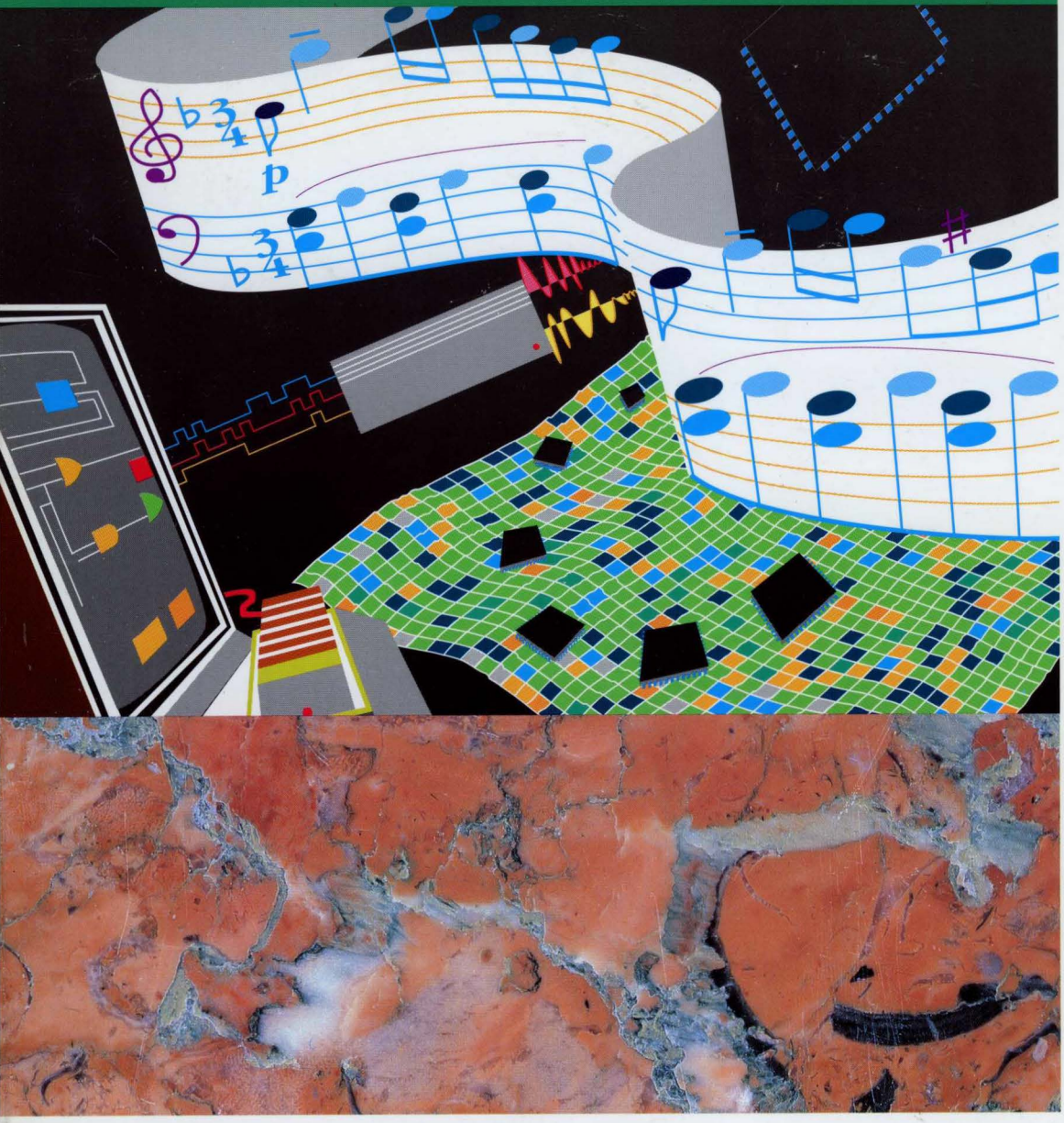


DIGITAL SIGNAL PROCESSOR (DSP) AND SPEECH PROCESSOR PRODUCTS DATA BOOK

NEC

Digital Signal Processor [DSP] and Speech Processor Products

1989–1990 DATA BOOK

June 1989
Document No. 50052

NEC

NEC

Contents

Section 4
DSP and Speech Development Tools (cont)

Section 5
Package Drawings

Numerical Index

NEC

SELECTION GUIDES

1

Section 1
Selection Guides

Part Numbering System

μPD72001L	Typical microdevice part number
μP	NEC monolithic silicon integrated circuit
D	Device type (D = digital MOS)
72001	Device identifier (alphanumeric)
L	Package type (L = PLCC)

A part number may include an alphanumeric suffix that identifies special device characteristics; for example, μPD72001L-11 has an 11-MHz data clock rating.

4-Bit, Single-Chip CMOS Microcomputers

Device, μPD	Features	Clock (MHz)	Supply Voltage (V)	ROM (X8)	RAM (X4)	I/O	# Package	Pins
7502	LCD controller/driver	0.4	2.5 to 6.0	2K	128	23	Miniflat	64
7503	LCD controller/driver	0.4	2.5 to 6.0	4K	224	23	Miniflat	64
7507	General-purpose	0.4	2.7 to 6.0	2K	128	32	DIP SDIP Miniflat	40 40 52
7508	General-purpose	0.4	2.7 to 6.0	4K	224	32	DIP SDIP Miniflat	40 40 52
75CG08	Piggyback EPROM	0.4	4.5 to 5.5	2K or 4K	224	32	Ceramic DIP	40
7514	LCD controller/driver	0.5	2.7 to 6.0	4K	256	31	Miniflat	80
7527A	FIP controller/driver	0.6	2.7 to 6.0	2K	128	35	DIP SDIP	42 42
7528A	FIP controller/driver	0.6	2.7 to 6.0	4K	160	35	DIP SDIP	42 42
75CG28	Piggyback EPROM; FIP controller/driver	0.5	4.5 to 5.5	4K	160	35	Ceramic DIP	42
7533	A/D converter	0.5	2.7 to 6.0	4K	160	30	DIP SDIP Miniflat	42 42 44
75CG33	Piggyback EPROM; A/D converter	0.5	4.5 to 5.5	4K	160	30	Ceramic DIP	42
7537A	FIP controller/driver	0.6	2.7 to 6.0	2K	128	35	DIP SDIP	42 42
7538A	FIP controller/driver	0.6	2.7 to 6.0	4K	160	35	DIP SDIP	42 42
75CG38	Piggyback EPROM; FIP controller/driver	0.5	4.5 to 5.5	4K	160	35	Ceramic DIP	42
7554	Serial I/O; external clock or RC oscillator	0.7	2.7 to 6.0	1K	64	16	SDIP SOP	20 20
75P54	Serial I/O; external clock or RC oscillator	0.7	4.5 to 6.0	1K OTPROM	64	16	SDIP SOP	20 20
7564	Serial I/O; ceramic oscillator	0.7	2.7 to 6.0	1K	64	16	SDIP SOP	20 20
75P64	Serial I/O; ceramic oscillator	0.7	4.5 to 6.0	1K OTPROM	64	16	SDIP SOP	20 20
7556	Comparator; external clock or RC oscillator	0.7	2.7 to 6.0	1K	64	20	SDIP SOP	24 24
75P56	Comparator; external clock or RC oscillator	0.7	4.5 to 6.0	1K OTPROM	64	20	SDIP SOP	24 24
7566	Comparator; ceramic oscillator	0.7	2.7 to 6.0	1K	64	19	SDIP SOP	24 24
75P66	Comparator; ceramic oscillator	0.7	4.5 to 6.0	1K OTPROM	64	19	SDIP SOP	24 24
75004	General-purpose	4.19	2.7 to 6.0	4K	512	34	SDIP Miniflat	42 44

Plastic unless ceramic (or cerdip) is specified.
* Under development; consult Microcontroller Marketing for availability.

4-Bit, Single-Chip CMOS Microcomputers (cont)

Device, μPD	Features	Clock (MHz)	Supply Voltage (V)	ROM (X8)	RAM (X4)	I/O	# Package	Pins
75006	General-purpose	4.19	2.7 to 6.0	6K	512	34	SDIP Miniflat	42 44
75008	General-purpose	4.19	2.7 to 6.0	8K	512	34	SDIP Miniflat	42 44
75P008	General-purpose	4.19	4.5 to 5.5	8K OTPROM	512	34	SDIP Miniflat	42 44
75028 *	A/D converter	4.19	2.7 to 6.0	8K	512	40	SDIP Miniflat	64 64
75P028 *	A/D converter	4.19	4.5 to 6.0	8K	512	40	SDIP Miniflat	64 64
75048 *	A/D converter; 0.5K EEPROM	4.19	2.7 to 6.0	8K	512	40	SDIP Miniflat	64 64
75104	High-end with 8-bit instruction	4.19	2.7 to 6.0	4096	320	58	SDIP Miniflat	64 64
75106	High-end with 8-bit instruction	4.19	2.7 to 6.0	6016	320	58	SDIP Miniflat	64 64
75108	High-end with 8-bit instruction	4.19	2.7 to 6.0	8064	512	58	SDIP Miniflat	64 64
75P108	High-end with 8-bit instruction; on-chip OTPROM or UVEPROM	4.19	4.5 to 5.5	8064	512	58	DIP Miniflat Shrink cerdip	64 64 64
75112	High-end with 8-bit instruction	4.19	2.7 to 6.0	12,032	512	58	SDIP Miniflat	64 64
75116	High-end with 8-bit instruction	4.19	2.7 to 6.0	16,128	512	58	SDIP Miniflat	64 64
75P116	High-end with 8-bit instruction	4.19	4.5 to 5.5	16,128 OTPROM	512	58	DIP Miniflat	64 64
75206	FIP controller/driver	4.19	2.7 to 6.0	6016	369	32	SDIP Miniflat	64 64
75208	FIP controller/driver	4.19	2.7 to 6.0	8064	497	32	SDIP Miniflat	64 64
75CG208	FIP controller/driver; piggyback EPROM	4.19	4.5 to 5.5	8064	512	32	Ceramic SDIP Ceramic flatpack	64 64
75212A	FIP controller/driver	4.19	2.7 to 6.0	12,160	512	32	SDIP Miniflat	64 64
75216A	FIP controller/driver	4.19	2.7 to 6.0	16,256	512	32	SDIP Miniflat	64 64
75CG216A	FIP controller/driver; piggyback EPROM	4.19	4.5 to 5.5	16,256	512	32	Ceramic SDIP Ceramic miniflat	64 64
75P216A *	FIP controller/driver	4.19	4.5 to 5.5	16,256 OTPROM	512	32	SDIP	64
75268 *	FIP controller/driver	4.19	2.7 to 6.0	8064	512	20	SDIP Flatpack	64 64
75304	LCD controller/driver	4.19	2.7 to 6.0	4K	512	68	Miniflat	80
75306	LCD controller/driver	4.19	2.7 to 6.0	6K	512	68	Miniflat	80
75308	LCD controller/driver	4.19	2.7 to 6.0	8K	512	68	Miniflat	80
75P308	LCD controller/driver; on-chip OTPROM or UVEPROM	4.19	4.5 to 5.5	8K	512	68	Miniflat Ceramic LCC	80 80
75312	LCD controller/driver	4.19	2.7 to 6.0	12K	512	68	Miniflat	80

4-Bit, Single-Chip CMOS Microcomputers (cont)

Device, μPD	Features	Clock (MHz)	Supply Voltage (V)	ROM (X8)	RAM (X4)	I/O	# Package	Pins
75316	LCD controller/driver	4.19	2.7 to 6.0	16K	512	68	Miniflat	80
75P316A *	LCD controller/driver; on-chip OTPROM or UVEPROM	4.19	2.7 to 6.0	16K	512	68	Miniflat Ceramic LCC	80 80
75328	LCD controller/driver; A/D converter	4.19	2.7 to 6.0	8064	512	24	Miniflat	80
75P328	LCD controller/driver; A/D converter	4.19	4.5 to 5.5	8064 OTPROM	512	24	Miniflat	80
75402	Low-end	4.19	2.7 to 6.0	1920	64	22	DIP SDIP Miniflat	28 28 44
75P402	Low-end	4.19	4.5 to 5.5	1920 OTPROM	64	22	DIP SDIP Miniflat	28 28 44
75516	High-end; A/D converter	4.19	2.7 to 6.0	16K	512	68	Miniflat	80
75P516	High-end; A/D converter	4.19	4.5 to 5.5	16K OTPROM	512	68	Miniflat LCC	80 80

8-Bit, Single-Chip NMOS/CMOS Microcomputers

Device, μPD	Features	Clock (MHz)	Supply Voltage (V)	ROM (X8)	RAM (X8)	I/O	# Package	Pins
7810H	NMOS; A/D converter	15	4.5 to 5.5	External	256	32	SDIP QUIP	64 64
7811H	NMOS; A/D converter	15	4.5 to 5.5	4K	256	44	SDIP QUIP	64 64
78PG11H	NMOS; A/D converter piggyback EPROM	15	4.5 to 5.5	4K	256	44	Ceramic QUIP	64
78C10/78C10A	CMOS; A/D converter	15	4.5 to 5.5	External	256	32	QUIP SDIP Miniflat PLCC	64 64 64 68
78C11/78C11A	CMOS; A/D converter	15	4.5 to 5.5	4K	256	44	QUIP SDIP Miniflat PLCC	64 64 64 68
78C12A	CMOS; A/D converter	15	4.5 to 5.5	8K	256	44	QUIP SDIP Miniflat PLCC	64 64 64 68
78C14	CMOS; A/D converter	15	4.5 to 5.5	16K	256	44	QUIP SDIP Miniflat PLCC	64 64 64 68
78CP14	CMOS; A/D converter	15	4.5 to 5.5	16K OTPROM	256	44	QUIP SDIP Miniflat PLCC	64 64 64 68
				16K UVEPROM	256	44	Ceramic QUIP Shrink cerdip	64 64

8-Bit, Single-Chip NMOS/CMOS Microcomputers (cont)

Device, μPD	Features	Clock (MHz)	Supply Voltage (V)	ROM (X8)	RAM (X8)	I/O	# Package	Pins
78213	CMOS; A/D converter; advanced peripherals	12	4.5 to 5.5	External	512	54	SDIP	64
							QUIP	64
							Miniflat	74
							PLCC	68
78214	CMOS; A/D converter; advanced peripherals	12	4.5 to 5.5	16K	512	54	SDIP	64
							QUIP	64
							Miniflat	74
							PLCC	68
78P214	CMOS; A/D converter; advanced peripherals	12	4.5 to 5.5	16K OTPROM	512	54	SDIP	64
							QUIP	64
							Miniflat	74
							PLCC	68
				16K UVEPROM	512	54	Shrink cerdip	64
							Ceramic QUIP	64
78220	CMOS; analog comparator; large I/O	12	4.5 to 5.5	External	640	71	PLCC	84
							Miniflat	94
78224	CMOS; analog comparator; large I/O	12	4.5 to 5.5	16K	640	71	PLCC	84
							Miniflat	94
78P224	CMOS; analog comparator; large I/O	12	4.5 to 5.5	16K OTPROM	640	71	PLCC	84
							Miniflat	94
78233 *	CMOS; real-time outputs; A/D and D/A converters	12	4.5 to 5.5	External	640	54	Miniflat	80
							Miniflat	94
							PLCC	84
78234 *	CMOS; real-time outputs; A/D and D/A converters	12	4.5 to 5.5	16K	640	54	Miniflat	80
							Miniflat	94
							PLCC	84
78P234 *	CMOS; real-time outputs; A/D and D/A converters	12	4.5 to 5.5	16K OTPROM	640	54	Miniflat	80
							Miniflat	94
							PLCC	84

16-Bit, Single-Chip CMOS Microcomputers

Device, μPD	Features	Clock (MHz)	Supply Voltage (V)	ROM (X8)	RAM (X8)	I/O	# Package	Pins
78310A	Real-time motor control	12	4.5 to 5.5	External	256	48	SDIP	64
							QUIP	64
							Miniflat	64
							PLCC	68
78312A	Real-time motor control	12	4.5 to 5.5	8K	256	48	SDIP	64
							QUIP	64
							Miniflat	64
							PLCC	68
78P312A	Real-time motor control	12	4.5 to 5.5	8K UVEPROM	256	48	Shrink cerdip	64
							Ceramic QUIP	64
				8K OTPROM	256	48	SDIP	64
							QUIP	64
							PLCC	68
78320	High-end; advanced analog and digital peripherals	16	4.5 to 5.5	External	640	55	Miniflat	64
							PLCC	68
78322	High-end; advanced analog and digital peripherals	16	4.5 to 5.5	16K	640	55	Miniflat	64
							PLCC	68

16-Bit, Single-Chip CMOS Microcomputers (cont)

Device, μPD	Features	Clock (MHz)	Supply Voltage (V)	ROM (X8)	RAM (X8)	I/O	# Package	Pins
78P322	High-end; advanced analog and digital peripherals	16	4.5 to 5.5	16K OTPROM	640	55	PLCC	68
71P301	Port and memory extender used with 7832X microcomputer family; UVEPROM or OTPROM	–	4.5 to 5.5	16K	1K	16	PLCC Miniflat Ceramic QUIP	44 64 64

8-Bit, Single-Chip Microcomputers

Device, μPD	Features	Clock (MHz)	Supply Voltage (V)	ROM (X8)	RAM (X8)	I/O	# Package	Pins
8035HL	HMOS	6	4.5 to 5.5	External	64	27	DIP	40
8039HL	HMOS	11	4.5 to 5.5	External	128	27	DIP	40
80C39H	CMOS	12	2.5 to 6.0	External	128	27	DIP Miniflat	40 44
80C40H	CMOS	12	2.5 to 6.0	External	256	27	DIP	40
8041AH	NMOS; universal PPI	11	4.5 to 5.5	1K	64	18	DIP	40
80C42	CMOS; universal PPI	12	4.5 to 5.5	2K	128	18	DIP Miniflat	40 44
8048H	HMOS	6	4.5 to 5.5	1K	64	27	DIP	40
8049H	HMOS	11	4.5 to 5.5	2K	128	27	DIP	40
80C49H	CMOS	12	2.5 to 6.0	2K	128	27	DIP	40
49H	CMOS	12	2.5 to 6.0	2K	128	27	Miniflat	44
80C50H	CMOS	12	2.5 to 6.0	4K	256	27	DIP	40
50H	CMOS	12	2.5 to 6.0	4K	256	27	Miniflat	44
8741A	NMOS; universal PPI; UVEPROM	6	4.5 to 5.5	1K	64	18	Cerdip	40
8748H	NMOS; OTPROM or UVEPROM	11	4.5 to 5.5	1K	64	27	DIP Cerdip	40 40
8749H	HMOS; OTPROM or UVEPROM	11	4.5 to 5.5	2K	128	27	DIP Cerdip	40 40

1

CMOS Microprocessors

Device, μPD	Features	Data Bits	Clock (MHz)	# Package	Pins
70008A	*Z80 microprocessor	8	8	DIP	40
				Miniflat	44
				PLCC	44
70108 (V20)	8088 compatible; enhanced	8/16	8 or 10	DIP	40
				Ceramic DIP	40
				Miniflat	52
				PLCC	44
70116 (V30)	8086 compatible; enhanced	16	8 or 10	DIP	40
				Ceramic DIP	40
				Miniflat	52
				PLCC	44
70208 (V40)	MS-DOS, V20 compatible CPU with peripherals	8/16	8 or 10	Ceramic PGA	68
				PLCC	68
				Miniflat	80
70216 (V50)	MS-DOS, V30 compatible CPU with peripherals	16/16	8 or 10	PGA	68
				PLCC	68
				Miniflat	80
70616 (V60)	32-bit; high-speed	16/32	16	PGA	68
70632 (V70)	32-bit; high-speed	32/32	20/25	PGA	132
70832 (V80)	32-bit; high-speed	32/32	25	Ceramic PGA	208
70136 (V33)	Hardwired, enhanced V30	16	12 or 16	PGA	68
				PLCC	68
70236 (V53)	V33 core-based; high-integration; DMA, serial I/O, interrupt controller, etc.	16	–	Ceramic PGA	132
70320 (V25)	MS-DOS compatible; high-integration; DMA, serial I/O, interrupt controller, etc.	8/16	5 or 8	PLCC	84
				Miniflat	94
70330 (V35)	MS-DOS compatible; high-integration; DMA, serial I/O, interrupt controller, etc.	16	8	PLCC	84
				Miniflat	94
70325 (V25+)	MS-DOS compatible; high-integration; high-speed DMA	8/16	8 or 10	PLCC	84
				Miniflat	94
70335 (V35+)	MS-DOS compatible; high-integration; high-speed DMA	16	8 or 10	PLCC	84
				Miniflat	94
70327 (V25 Software Guard)	MS-DOS compatible; high-integration; software protection	8/16	8	PLCC	84
				Miniflat	94
70337 (V35 Software Guard)	MS-DOS compatible; high-integration; software protection	16	8	PLCC	84
				Miniflat	94
79011 (V25 RTOS)	MS-DOS compatible; high-integration; real-time operating system	8/16	8	PLCC	84
				Miniflat	94
79021 (V35 RTOS)	MS-DOS compatible; high-integration; real-time operating system	16	8	PLCC	84
				Miniflat	94
70322 (V25 ROM)	MS-DOS compatible; high-integration; 16K-byte ROM	8/16	8	PLCC	84

Plastic unless ceramic (or cerdip) is specified.
* For additional information, refer to 1987 Microcomputer Data Book.

CMOS Microprocessors (cont)

Device, μPD	Features	Data Bits	Clock (MHz)	# Package	Pins
70P322	MS-DOS compatible; high-integration; 16K-byte UVEPROM; V25 or V35 mode	8/16	8	Ceramic LCC	84
70332 (V35 ROM)	MS-DOS compatible; high-integration; 16K-byte ROM	16	8	PLCC	84

NMOS and HMOS Microprocessors

Device, μPD	Features	Data Bits	Clock (MHz)	# Package	Pins
8085A	*8-bit microprocessor; NMOS or HMOS	8	5	DIP	40
8086	*16-bit microprocessor; HMOS	16	8	Cerdip	40
8088	*8-bit microprocessor; HMOS	8	8	Ceramic DIP	40

CMOS System Support Products

Device, μPD	Name	Data Bits	Clock (MHz)	# Package	Pins
71011	Clock Pulse Generator/Driver	–	20	DIP	18
				SOP	20
71037	Programmable DMA Controller	8	10	DIP	40
				Miniflat	40
				PLCC	44
71051	Serial Control Unit	8	8/10	DIP	28
				Miniflat	44
				PLCC	28
71054	Programmable Timer/Controller	8	8/10	DIP	24
				Miniflat	44
				PLCC	28
71055	Parallel Interface Unit	8	8/10	DIP	40
				Miniflat	44
				PLCC	44
71059	Interrupt Control Unit	8	8/10	DIP	28
				Miniflat	44
				PLCC	28
71071	DMA Controller	8/16	8/10	DIP	48
				Ceramic DIP	48
				Miniflat	52
				PLCC	52
71082	Transparent Latch	8	8	DIP	20
				SOP	20
71083	Transparent Latch	8	8	DIP	20
				SOP	20
71084	Clock Pulse Generator/Driver	–	25	DIP	18
				SOP	20
71086	Bus Buffer/Driver	8	8	DIP	18
				SOP	20
71087	Bus Buffer/Driver	8	8	DIP	20
				SOP	20

CMOS System Support Products (cont)

Device, μPD	Name	Data Bits	Clock (MHz)	# Package	Pins
71088	System Bus Controller	–	8/10	DIP	20
				SOP	20
82C43	*Input/Output Expander	–	5	DIP	24
				Skinny DIP	24

NMOS System Support Products

Device, μPD	Name	Data Bits	Clock (MHz)	# Package	Pins
8155H	*256 x 8 RAM; I/O ports and timer	8	3 or 5	DIP	40
8156H	*256 x 8 RAM; I/O ports and timer	8	3 or 5	DIP	40
8237A	*Programmable DMA Controller	8	5	DIP	40
8243	*Input/Output Expander	–	5	DIP	24
8251A	*Programmable Communications Interface	8	3/5	DIP	28
8253	*Programmable Internal Timer	8	5	DIP	24
8255A	*Programmable Peripheral Interface	8	5	DIP	40
8257	*Programmable DMA Controller	8	5	DIP	40
8259A	*Programmable Interrupt Controller	8	5	DIP	28
8279	*Programmable Keyboard/Display Interface	–	5	DIP	40

Communications Controllers

Device, μPD	Name	Description	Data Rate	# Package	Pins
7201A	Multiprotocol Serial Communications Controller	Dual full-duplex serial channels; four DMA channels; programmable interrupt vectors; asychronous COP and BOP support; NMOS	1 Mb/s	DIP Ceramic DIP	40 40
72001	CMOS, Advanced Multiprotocol Serial Communications Controller	Functional superset of 8530; 8086/V30 interface; two full-duplex serial channels; two digital phase-locked loops; two baud-rate generators per channel; loopback test mode; short frame and mark idle detection	2.2 Mb/s	DIP Miniflat PLCC	40 52 52
72002	CMOS, Advanced Multiprotocol Serial Communications Controller	Low-cost, single-channel version of 72001; software compatible; direct interface to 8237 DMA. Not included in 1989-1990 IPD Data Book; refer to 72002 data sheet.	2.2 Mb/s	DIP Miniflat PLCC	40 44 44
72101	CMOS, HDLC Controller	Single full-duplex serial channel; on-chip DMA Controller. Not included in 1989-1990 IPD Data Book; refer to 72101 data sheet	4 Mb/s	DIP PLCC	64 68

Graphics Controllers

Device, μPD	Name	Description	Drawing Rate	# Package	Pins
7220A	High-Performance Graphics Display Controller	General-purpose, high-integration controller; hardwired support for lines, arc/circles, rectangles, and graphics characters; 1024x1024 pixel display with four planes	500 ns/dot	Ceramic DIP	40
72020	Graphics Display Controller	CMOS 7220A with 2M video memory; dual-port RAM control; write-masking on any bit; enhanced external synch	500 ns/dot	DIP Miniflat	40 52
72022	Intelligent Display Processor	Display and image processing for text and sprites; three display modes; four-way horizontal split-screen display; CMOS	500 ns/dot	PLCC Miniflat	68 80
72120	Advanced Graphics Display Controller	High-speed graphics operations including paint, area fill, slant, arbitrary angle rotate, up to 16x enlargement and reduction; dual-port RAM control; CMOS	500 ns/dot	PLCC Miniflat	84 94
72123	Advanced Graphics Display Controller II	Enhanced 72120; expanded command set; improved painting performance; laser printer interface controls; CMOS	400 ns/dot	PLCC Miniflat	84 94

Advanced Compression/Expansion Engine

Device, μPD	Name	Description	# Package	Pins
72185	Advanced Compression/ Expansion Engine	High-speed CCITT Group 3/4 bit-map image compression/expansion (A4 test chart, 400 PPI x 400 LPI in under 1 second); 32K-pixel line length; 32-megabyte image memory; on-chip DMA and refresh timing generator; CMOS	SDIP PLCC	64 68

Plastic unless ceramic (or cerdip) is specified.

Floppy-Disk Controllers

Device, μPD	Name	Description	Transfer Rate	# Package	Pins
765A/B	Floppy-Disk Controller	Industry-standard controller supporting IBM 3740 and IBM System 34 double-density format; enhanced 765B supports multitasking applications	500 kb/s	DIP	40
71065/66	Floppy-Disk Interface	Compatible with 765-family controllers and others; supports multiple data rates from 125 to 500 kb/s	500 kb/s	SOP SDIP	28 30
72065/65B	CMOS Floppy-Disk Controller	100% 765A/B microcode compatible; compatible with 808x microprocessor families	1 Mb/s	DIP PLCC Miniflat	40 44 52
72067	Floppy-Disk Controller	CMOS; 765A/B microcode compatible; clock generation/ switching circuitry; selectable write precompensation; digital phase-locked loop	500 kb/s	DIP Miniflat PLCC	48 52 52
72068	Floppy-Disk Controller	All features of the 72067 plus IBM-PC, PC/XT, PC/AT, or PS/2 style registers; 24-ma high-current drivers	500 kb/s	Miniflat PLCC	80 84
72069	Floppy-Disk Controller	All features of the 72067/68 with substitution of high-performance analog phase-locked loop for digital PLL	1 Mb/s	PLCC Miniflat	84 100

Hard-Disk Controllers

Device, μPD	Name	Description	Read/Write Clock	# Package	Pins
7261A/B	Hard-Disk Controller	Supports eight drives in SMD mode, four drives in ST506 mode; error correction and detection	23 MHz	Ceramic DIP	40
7262	Enhanced Small-Disk Interface (ESDI) Controller	Serial-mode ESDI compatible; controls up to seven drives; supports up to 80 heads; hard and soft-sector interfacing	18 MHz	Ceramic DIP	40
72061	CMOS Hard-Disk Controller	Supports SMD/SMD-E and ST506/412 type drives	24 MHz	DIP Miniflat PLCC	40 52 52
72111	Small Computer System Interface (SCSI) Controller	Selectable 8/16 data bus width; 16 high-level commands for reduced CPU load; single-command automatic execution; 4-Mb sync/async; CMOS	16 MHz	SDIP Miniflat PLCC	64 74 68

1

Digital Signal Processors

Device, μPD	Description	Instruction Cycle (ns)	Instruction ROM (Bits)	Data ROM (Bits)	Data RAM (Bits)	# Package	Pins
7720A	16-bit, fixed-point DSP; NMOS	244	512 x 23	510 x 13	128 x 16	DIP PLCC	28 44
77C20A	16-bit, fixed-point DSP; CMOS	244	512 x 23	510 x 13	128 x 16	DIP PLCC PLCC	28 28 44
77P20	16-bit, fixed-point DSP; CMOS	244	512 x 23 UVEPROM	510 x 13 UVEPROM	128 x 16	Cerdip	28
77C25	16-bit, fixed-point DSP; CMOS	122	2048 x 24	1024 x 16	256 x 16	DIP PLCC	28 44
77P25	16-bit, fixed-point DSP; CMOS	122	2048 x 24 OTPROM	1024 x 16 OTPROM	256 x 16	DIP PLCC	28 44
			2048 x 24 UVEPROM	1024 x 16 UVEPROM	256 x 16	Cerdip	28
77220	24-bit, fixed-point DSP; CMOS	122	2048 x 32	1024 x 24	512 x 24	Ceramic PGA PLCC	68 68
77230AR	32-bit, floating-point DSP; CMOS	150	2048 x 32	1024 x 32	1024 x 32	Ceramic PGA	68
77230AR-003	32-bit, floating-point DSP; CMOS; standard library software	150	n/a	n/a	n/a	Ceramic PGA	68
77P230R	32-bit, floating-point DSP; CMOS	150	2048 x 32 UVEPROM	1024 x 32 UVEPROM	1024 x 32	Ceramic PGA	68
77810	16-bit fixed-point modem DSP; CMOS	181	2048 x 24	1024 x 16	256 x 16	Ceramic PGA PLCC	68 68
7281	Image pipelined processor; NMOS	5-MHz clock	n/a	n/a	512 x 18	Ceramic DIP	40
9305	Support device for μPD7281 processors; CMOS	10-MHz clock	n/a	n/a	n/a	Ceramic PGA	132

Speech Processors

Device, μPD	Name	Technology	Clock (MHz)	Data ROM (Bits)	# Package	Pins
7730	ADPCM Speech Encoder/Decoder	NMOS	8	–	DIP	28
77C30	ADPCM Speech Encoder/Decoder	NMOS	8	–	DIP PLCC	28 44
7755	ADPCM Speech Synthesizer	CMOS	0.7	96K	DIP SOP	18 24
7756	ADPCM Speech Synthesizer	CMOS	0.7	256K	DIP SOP	18 24
77P56	ADPCM Speech Synthesizer	CMOS	0.7	256K OTPROM	DIP SOP	20 24
7757	ADPCM Speech Synthesizer	CMOS	0.7	512K	DIP SOP	18 24
7759	ADPCM Speech Synthesizer	CMOS	0.7	1024K external	DIP Miniflat	40 52

Plastic unless ceramic (or cerdip) is specified.

V-Series Development Tools Selection Guide

Part Number (Note 1)	Full Emulator	Full Emulator Probe	Mini-IE Emulator	Mini-IE Probe	Evaluation Boards	EPROM/OTP Device	Relocatable Assembler (Note 13)	C Compiler (Note 14)
μPD70136GJ-12	IE-70136-A016	EP-70136L-A (Note 2)	IE-70136-PC	EP-70136L-PC (Note 2)	DDK-70136	–	RA70136	CC70136
μPD70136GJ-16	IE-70136-A016	EP-70136L-A (Note 2)	IE-70136-PC	EP-70136L-PC (Note 2)	DDK-70136	–	RA70136	CC70136
μPD70136L-16	IE-70136-A016	EP-70136L-A	IE-70136-PC	EP-70136L-PC	DDK-70136	–	RA70136	CC70136
μPD70136L-12	IE-70136-A016	EP-70136L-A	IE-70136-PC	EP-70136L-PC	DDK-70136	–	RA70136	CC70136
μPD70136R-12	IE-70136-A016	EP-70136L-A (Note 3)	IE-70136-PC	EP-70136L-PC (Note 3)	DDK-70136	–	RA70136	CC70136
μPD70136R-16	IE-70136-A016	EP-70136L-A (Note 3)	IE-70136-PC	EP-70136L-PC (Note 3)	DDK-70136	–	RA70136	CC70136
μPD70208GF-8	IE-70208-A010	(Note 12)	EB-V40MINI-IE	–	EB-70208	–	RA70116	CC70116
μPD70208GF-10	IE-70208-A010	(Note 12)	EB-V40MINI-IE	–	EB-70208	–	RA70116	CC70116
μPD70208L-8	IE-70208-A010	IE-70000-2958	EB-V40MINI-IE	ADAPT68PGA 68PLCC (Note 4)	EB-70208	–	RA70116	CC70116
μPD70208L-10	IE-70208-A010	IE-70000-2958	EB-V40MINI-IE	ADAPT68PGA 68PLCC (Note 4)	EB-70208	–	RA70116	CC70116
μPD70208R-8	IE-70208-A010	IE-70000-2959	EB-V40MINI-IE	(Note 4)	EB-70208	–	RA70116	CC70116
μPD70208R-10	IE-70208-A010	IE-70000-2959	EB-V40MINI-IE	(Note 4)	EB-70208	–	RA70116	CC70116
μPD70216GF-8	IE-70216-A010	(Note 12)	EB-V50MINI-IE	–	EB70216	–	RA70116	CC70116
μPD70216GF-10	IE-70216-A010	(Note 12)	EB-V50MINI-IE	–	EB70216	–	RA70116	CC70116
μPD70216L-8	IE-70216-A010	IE-70000-2958	EB-V50MINI-IE	ADAPT68PGA 68PLCC (Note 4)	EB70216	–	RA70116	CC70116
μPD70216L-10	IE-70216-A010	IE-70000-2958	EB-V50MINI-IE	ADAPT68PGA 68PLCC (Note 4)	EB70216	–	RA70116	CC70116
μPD70216R-8	IE-70216-A010	IE-70000-2959	EB-V50MINI-IE	(Note 4)	EB70216	–	RA70116	CC70116
μPD70216R-10	IE-70216-A010	IE-70000-2959	EB-V50MINI-IE	(Note 4)	EB70216	–	RA70116	CC70116
μPD70320GJ	IE-70320-A008	EP-70320GJ (Note 5)	EB-V25MINI-IE-P	EP-70320GJ (Note 6)	DDK-70320	–	RA70320	CC70116
μPD70320GJ-8	IE-70320-A008	EP-70320GJ (Note 5)	EB-V25MINI-IE-P	EP-70320GJ (Note 6)	DDK-70320	–	RA70320	CC70116
μPD70320L	IE-70320-A008	EP-70320L	EB-V25MINI-IE-P	(Note 7)	DDK-70320	–	RA70320	CC70116
μPD70320L-8	IE-70320-A008	EP-70320L	EB-V25MINI-IE-P	(Note 7)	DDK-70320	–	RA70320	CC70116
μPD70322GJ	IE-70320-A008	EP-70320GJ (Note 5)	EB-V25MINI-IE-P	EP-70320GJ (Note 6)	DDK-70320	–	RA70320	CC70116
μPD70322GJ-8	IE-70320-A008	EP-70320GJ (Note 5)	EB-V25MINI-IE-P	EP-70320GJ (Note 6)	DDK-70320	–	RA70320	CC70116
μPD70322L	IE-70320-A008	EP-70320L	EB-V25MINI-IE-P	(Note 7)	DDK-70320	70P322K (Note 10)	RA70320	CC70116

V-Series Development Tools Selection Guide (cont)

Part Number (Note 1)	Full Emulator	Full Emulator Probe	Mini-IE Emulator	Mini-IE Probe	Evaluation Boards	EPROM/OTP Device	Relocatable Assembler (Note 13)	C Compiler (Note 14)
μPD70322L-8	IE-70320-A008	EP-70320L	EB-V25MINI-IE-P	(Note 7)	DDK70320	70P322K (Note 10)	RA70320	CC70116
μPD70325GJ-8	IE-70325-A008	EP-70320GJ (Note 5)	(Note 12)	(Note 12)	DDK-70325	–	RA70320	CC70116
μPD70325GJ-10	IE-70325-A008 (Note 8)	EP-70320GJ (Note 5)	(Note 12)	(Note 12)	DDK-70325	–	RA70320	CC70116
μPD70325L-8	IE-70325-A008	EP-70320L	(Note 12)	(Note 12)	DDK-70325	–	RA70320	CC70116
μPD70325L-10	IE-70325-A008 (Note 8)	EP-70320L	(Note 12)	(Note 12)	DDK-70325	–	RA70320	CC70116
μPD70327GJ-8 (Note 9)	IE-70320-A008	EP-70320GJ (Note 5)	EB-V25MINI-IE-P	EP-70320GJ (Note 6)	–	–	RA70320	CC70116
μPD70327L-8 (Note 9)	IE-70320-A008	EP-70320L	EB-V25MINI-IE-P	(Note 7)	–	–	RA70320	CC70116
μPD70330GJ-8	IE-70330-A008	EP-70320GJ (Note 5)	EB-V35MINI-IE-P	EP-70320GJ (Note 6)	DDK-70330	–	RA70320	CC70116
μPD70330L-8	IE-70330-A008	EP-70320L	EB-V35MINI-IE-P	(Note 7)	DDK-70330	–	RA70320	CC70116
μPD70332GJ-8	IE-70330-A008	EP-70320GJ (Note 5)	EB-V35MINI-IE-P	EP-70320GJ (Note 6)	DDK-70330	–	RA70320	CC70116
μPD70332L-8	IE-70330-A008	EP-70320L	EB-V35MINI-IE-P	(Note 7)	DDK-70330	70P322K (Note 10)	RA70320	CC70116
μPD70335GJ-8	IE-70335-A008	EP-70320GJ (Note 5)	(Note 12)	(Note 12)	DDK-70330	–	RA70320	CC70116
μPD70335GJ-10	IE-70335-A008 (Note 8)	EP-70320GJ (Note 5)	(Note 12)	(Note 12)	DDK-70330	–	RA70320	CC70116
μPD70335L-8	IE-70335-A008	EP-70320L	(Note 12)	(Note 12)	DDK-70330	–	RA70320	CC70116
μPD70335L-10	IE-70335-A008 (Note 8)	EP-70320L	(Note 12)	(Note 12)	DDK-70330	–	RA70320	CC70116
μPD70337GJ-8 (Note 9)	IE-70330-A008	EP-70320GJ (Note 5)	EB-V35MINI-IE-P	EP-70320GJ (Note 6)	–	–	RA70320	CC70116
μPD70337L-8 (Note 9)	IE-70330-A008	EP-70320L	EB-V35MINI-IE-P	(Note 7)	–	–	RA70320	CC70116
μPD79011GJ-8 (Note 11)	IE-70320-A008	EP-70320GJ (Note 5)	(Note 12)	(Note 12)	–	–	RA70320	CC70116
μPD79011L-8 (Note 11)	+IE-70320-RTOS	EP-70320L	(Note 12)	(Note 12)	–	–	RA70320	CC70116
μPD79021L-8 (Note 11)	IE-70330-A008 +IE-70330-RTOS	EP-70320L	(Note 12)	(Note 12)	–	–	RA70320	CC70116

Notes:

(1) Packages:

Package	Description
GF	80-pin plastic miniflat
GJ	74-pin or 94-pin plastic miniflat
K	84-pin ceramic LCC with window
L	68-pin or 84-pin plastic LCC
R	68-pin PGA

(2) The EP-70136GL-A and EP-70136L-PC contain both a 68-pin PLCC probe and an adapter which converts the 68-pin PLCC probes to a 74-pin miniflat footprint.

(3) 68-pin PGA parts are supported by using the EP-70136L-A PLCC probe or EP-70136L-PC PLCC probe, plus a PLCC socket with a PGA-pinout. A PLCC socket of this type is supplied with the EP-70136L-A.

(4) The EB-V40 MINI-IE and EB-V50 MINI-IE support PGA packages directly; the ADAPT68PGA68PLCC adaptor converts the PGA-pinout on the MINI-IE to a PLCC footprint. This adaptor is supplied with the MINI-IE.

(5) The EP-70320GJ is an adaptor to the EP-70320L, which converts 84-pin PLCC probes to a 94-pin miniflat footprint. For GJ parts, both the PLCC probe and the adaptor are needed.

(6) The EP-70320GJ adaptor can be used to convert the supplied 84-pin PLCC cable of the EB-V25 MINI-IE-P or EB-V35 MINI-IE-P to a 94-pin miniflat.

(7) The EB-V25 MINI-IE-P and EB-V35 MINI-IE-P are supplied with an 84-pin PLCC cable.

(8) At the current time, the emulators for the μPD70325 and μPD70335 are specified to 8 MHz. Contact your local NEC Sales Office for the latest information on 10 MHz emulation.

(9) Development for the μPD70327 or μPD70337 can be done using the appropriate μPD70320 or μPD70330 tools; however, debugging of programs in the Software Guard mode is not supported at this time.

(10) The μPD70P322K EPROM device can be used for both μPD70322 and μPD70332 emulation. The μPD70P322K EPROM device can be programmed by using the PA-70P322L Programming Adapter and the PG-1500 EPROM Programmer.

(11) For emulation of μPD79011 or μPD79021, the base emulator (IE-70320 or IE-70330) plus Real-Time Operating System software IE-70320-RTOS or IE-70330-RTOS) is required.

(12) This emulation option is not currently supported, but may be available in the future. Contact your local NEC Sales Office for further information.

(13) The following relocatable assemblers are available:

RA70116-D52	For V20®/V30®/	(VS-DOS®)
RA70116-VVT1	V40™/V50™	(VAX/VMS™)
RA70116-VXT1		(VAX/UNIX™ 4.2 BSD or Ultrix™)
RA70136-D52	For V33™	(MS-DOS)
RA70136-VVT1		(VAX/VMS)
RA70136-VXT1		(VAX/UNIX 4.2 BSD or Ultrix)
RA70320-D52	For V25™ and V35™	(MS-DOS)
RA70320-VVT1		(VAX/VMS)
RA70320-VXT1		(VAX/UNIX 4.2 BSD or Ultrix)

(14) The following C compilers are available:

CC70116-D52	For V20/V30/	(MS-DOS)
CC70116-VVT1	V40/V50 and	(VAX/VMS)
CC70116-VXT1	V25/V35	(VAX/UNIX 4.2 BSD or Ultrix)
CC70136-D52	For V33	(MS-DOS)
CC70136-VVT1		(VAX/VMS)
CC70136-VXT1		(VAX/UNIX 4.2 BSD or Ultrix)
CC70320-D52	For V25 and V35	(MS-DOS)
CC70320-VVT1		(VAX/VMS)
CC70320-VXT1		(VAX/UNIX 4.2 BSD or Ultrix)

1

μPD75XX Series Development Tools Selection Guide

Part Number (Note 1)	Emulator*	Add-on Board*	System Evaluation Board	EPROM/OTP Device	PG-1500 Adapter (Note 2)	Absolute Assembler (Note 3)
μPD7501G-12	EVAKIT-7500B	EV7514	SE-7514A	–	–	ASM75
μPD7502G-12	EVAKIT-7500B	EV7514	SE-7514A	–	–	ASM75
μPD7503G-12	EVAKIT-7500B	EV7514	SE-7514A	–	–	ASM75
μPD7506C	EVAKIT-7500B	–	SE-7508	–	–	ASM75
μPD7506CT	EVAKIT-7500B	–	–	–	–	ASM75
μPD7506G-00	EVAKIT-7500B	–	–	–	–	ASM75
μPD7507C	EVAKIT-7500B	–	–	μPD78CG08E	–	ASM75
μPD7507CU	EVAKIT-7500B	–	–	–	–	ASM75
μPD7507G-00	EVAKIT-7500B	–	–	–	–	ASM75
μPD7507HC	EVAKIT-7500B	EV7508H	–	μPD75CG08HE	–	ASM75
μPD7507HCU	EVAKIT-7500B	EV7508H	–	–	–	ASM75
μPD7507HG-22	EVAKIT-7500B	EV7508H	–	–	–	ASM75
μPD7507SC	EVAKIT-7500B	–	SE-7508	–	–	ASM75
μPD7507SCT	EVAKIT-7500B	–	–	–	–	ASM75
μPD7508C	EVAKIT-7500B	–	–	μPD78CG08E	–	ASM75
μPD7508CU	EVAKIT-7500B	–	–	–	–	ASM75
μPD7508G-00	EVAKIT-7500B	–	–	–	–	ASM75
μPD75CG08E	EVAKIT-7500B	–	–	–	–	ASM75
μPD7508AC	EVAKIT-7500B	–	SE-7508	–	–	ASM75
μPD7508HC	EVAKIT-7500B	EV7508H	–	μPD75CG08HE	–	ASM75
μPD7508HCU	EVAKIT-7500B	EV7508H	–	–	–	ASM75
μPD7508HG-22	EVAKIT-7500B	EV7508H	–	–	–	ASM75
μPD75CG08HE	EVAKIT-7500B	EV7508H	–	–	–	ASM75
μPD7514G-12	EVAKIT-7500B	EV7514	SE-7514A	–	–	ASM75
μPD7516HCW	EVAKIT-7500B	EV7500FIP	–	–	–	ASM75
μPD7516HG-12	EVAKIT-7500B	EV7500FIP	–	–	–	ASM75
μPD7516HG-36	EVAKIT-7500B	EV7500FIP	–	μPD75CG16HE	–	ASM75
μPD75CG16HE	EVAKIT-7500B	EV7500FIP	–	–	–	ASM75
μPD7519HCW	EVAKIT-7500B	EV7500FIP	–	–	–	ASM75
μPD7519HG-12	EVAKIT-7500B	EV7500FIP	–	–	–	ASM75
μPD7519HG-36	EVAKIT-7500B	EV7500FIP	–	μPD75CG19HE	–	ASM75
μPD75CG19HE	EVAKIT-7500B	EV7500FIP	–	–	–	ASM75
μPD7527AC	EVAKIT-7500B	EV7528	–	μPD75CG28E	–	ASM75
μPD7527ACU	EVAKIT-7500B	EV7528	–	–	–	ASM75
μPD7528AC	EVAKIT-7500B	EV7528	–	μPD75CG28E	–	ASM75
μPD7528ACU	EVAKIT-7500B	EV7528	–	–	–	ASM75

* Required Tools

50174

μPD75XX Series Development Tools Selection Guide (cont)

Part Number (Note 1)	Emulator*	Add-on Board*	System Evaluation Board	EPROM/OTP Device	PG-1500 Adapter (Note 2)	Absolute Assembler (Note 3)
μPD75CG28E	EVAKIT-7500B	EV7528	–	–	–	ASM75
μPD7533C	EVAKIT-7500B	EV7533	–	μPD75CG33E	–	ASM75
μPD7533CU	EVAKIT-7500B	EV7533	–	–	–	ASM75
μPD7533G-22	EVAKIT-7500B	EV7533	–	–	–	ASM75
μPD75CG33E	EVAKIT-7500B	EV7533	–	–	–	ASM75
μPD7537AC	EVAKIT-7500B	EV7528	–	μPD75CG38E	–	ASM75
μPD7537ACU	EVAKIT-7500B	EV7528	–	–	–	ASM75
μPD7538AC	EVAKIT-7500B	EV7528	–	μPD75CG38E	–	ASM75
μPD7538ACU	EVAKIT-7500B	EV7528	–	–	–	ASM75
μPD75CG38E	EVAKIT-7500B	EV7528	–	–	–	ASM75
μPD7554CS	EVAKIT-7500B	EV7554A	SE-7554A	μPD75P54CS	PA-75P54CS	ASM75
μPD7554G	EVAKIT-7500B	EV7554A	SE-7554A	μPD75P54G	PA-75P54CS	ASM75
μPD75P54CS	EVAKIT-7500B	EV7554A	–	–	–	ASM75
μPD75P54G	EVAKIT-7500B	EV7554A	–	–	–	ASM75
μPD7556CS	EVAKIT-7500B	EV7554A	SE-7554A	μPD75P56CS	PA-75P56CS	ASM75
μPD7556G	EVAKIT-7500B	EV7554A	SE-7554A	μPD75P56G	PA-75P56CS	ASM75
μPD75P56CS	EVAKIT-7500B	EV7554A	–	–	–	ASM75
μPD75P56G	EVAKIT-7500B	EV7554A	–	–	–	ASM75
μPD7564CS	EVAKIT-7500B	EV7554A	SE-7554A	μPD75P64CS	PA-75P54CS	ASM75
μPD7564G	EVAKIT-7500B	EV7554A	SE-7554A	μPD75P64G	PA-75P54CS	ASM75
μPD75P64CS	EVAKIT-7500B	EV7554A	–	–	–	ASM75
μPD75P64G	EVAKIT-7500B	EV7554A	–	–	–	ASM75
μPD7566CS	EVAKIT-7500B	EV7554A	SE-7554A	μPD75P66CS	PA-75P56CS	ASM75
μPD7566G	EVAKIT-7500B	EV7554A	SE-7554A	μPD75P66G	PA-75P56CS	ASM75
μPD75P66CS	EVAKIT-7500B	EV7554A	–	–	–	ASM75
μPD75P66G	EVAKIT-7500B	EV7554A	–	–	–	ASM75

* Required tools

Notes:

(1) Packages:

Package	Description
C	28-pin plastic DIP (μPD7506/07S)
	40-pin plastic DIP (μPD7507/07H/08/08A/08H)
	42-pin plastic DIP (μPD7527A/28A/33/37A/38A)
CS	20-pin plastic shrink DIP (μPD7554/P54/64/P64)
	24-pin plastic shrink DIP (μPD7556/P56/66/P66)
CT	28-pin plastic shrink DIP
CU	40-pin plastic shrink DIP (μPD7507/07H/08/08H)
	42-pin plastic shrink DIP (μPD7527A/28A/33/37/37A/38A)
CW	64-pin plastic shrink DIP
E	40-pin ceramic piggy-back DIP (μPD75CG08/08H)
	42-pin ceramic piggy-back DIP (μPD75CG28/33/38)
	64-pin ceramic piggy-back QUIP (μPD75CG16H/19H)
G	20-pin plastic SO (μPD7554/P54/64/P64)
	24-pin plastic SO (μPD7556/P56/66/P66)
G-00	52-pin plastic miniflat
G-12	64-pin plastic miniflat (μPD7501/02/03/16H/19H)
	80-pin plastic miniflat (μPD7514)
G-22	44-pin plastic miniflat
G-36	64-pin plastic QUIP

(2) By using the specified adapter, the PG-1500 EPROM programmer can be used to program the OTP device.

(3) The ASM75 Absolute Assembler is provided to run under the MOS-DOS® operating system. (ASM75-D52).

MS-DOS is registered trademark of Microsoft Corporation.

1

μPD75XXX Series Development Tools Selection Guide

Part Number (Note 7)	Main Board Emulator*	Add-on Board*	Emulation Probe*	Optional Socket Adapter (Note 1)	EPROM/OTP Device (Note 2)	Relocatable Assembler (Note 5)	Structured Assembler (Note 6)
μPD75004CU	EVAKIT-75X	EV-75008	(Note 3)	–	μPD75P008CU/DU	RA75X	ST75X
μPD75006GB	EVAKIT-75X	EV-75008	EP-75008GB	EV-9200G-44	μPD75P008GB	RA75X	ST75X
μPD75006CU	EVAKIT-75X	EV-75008	(Note 3)	–	μPD75P008CU/DU	RA75X	ST75X
μPD75006GB	EVAKIT-75X	EV-75008	EP-75008GB	EV-9200G-44	μPD75P008GB	RA75X	ST75X
μPD75008CU	EVAKIT-75X	EV-75008	(Note 3)	–	μPD75P008CU/DU	RA75X	ST75X
μPD75008GB	EVAKIT-75X	EV-75008	EP-75008GB	EV-9200G-44	μPD75P008GB	RA75X	ST75X
μPD75P008CU	EVAKIT-75X	EV-75008	(Note 3)	–	–	RA75X	ST75X
μPD75P008DU	EVAKIT-75X	EV-75008	(Note 3)	–	–	RA75X	ST75X
μPD75P008GB	EVAKIT-75X	EV-75008	EP-75008GB	EV-9200G-44	–	RA75X	ST75X
μPD75028CW	EVAKIT-75X	EV-75048	(Note 4)	(Note 4)	μPD75P028CW	RA75X	ST75X
μPD75028GC	EVAKIT-75X	EV-75048	(Note 4)	(Note 4)	μPD75P028GC	RA75X	ST75X
μPD75P028CW	EVAKIT-75X	EV-75048	(Note 4)	(Note 4)	–	RA75X	ST75X
μPD75P028GC	EVAKIT-75X	EV-75048	(Note 4)	(Note 4)	–	RA75X	ST75X
μPD75048CW	EVAKIT-75X	EV-75048	(Note 4)	(Note 4)	–	RA75X	ST75X
μPD75048GC	EVAKIT-75X	EV-75048	(Note 4)	(Note 4)	–	RA75X	ST75X
μPD75104CW	EVAKIT-75X	EV-75108	(Note 3)	–	μPD75P108CW/DW	RA75X	ST75X
μPD75104G	EVAKIT-75X	EV-75108	EP-75108GF	EV-9200G-64	μPD75P108G/GF μPD75P116GF	RA75X	ST75X
μPD75104GF	EVAKIT-75X	EV-75108	EP-75108GF	EV-9200G-64	μPD75P108G/GF μPD75P116GF	RA75X	ST75X
μPD75104AGC	EVAKIT-75X	EV-75108	EP-75108GF	EV-9200G-64	–	RA75X	ST75X
μPD75106CW	EVAKIT-75X	EV-75108	(Note 3)	–	μPD75P108CW/DW	RA75X	ST75X
μPD75106G	EVAKIT-75X	EV-75108	EP-75108GF	EV-9200G-64	μPD75P108G/GF μPD75P116GF	RA75X	ST75X
μPD75106GF	EVAKIT-75X	EV-75108	EP-75108GF	EV-9200G-64	μPD75P108G/GF μPD75P116GF	RA75X	ST75X
μPD75108AG	EVAKIT-75X	EV-75108	EP-75108GF	EV-9200G-64	–	RA75X	ST75X
μPD75108AGC	EVAKIT-75X	EV-75108	EP-75108GF	EV-9200G-64	–	RA75X	ST75X
μPD75108CW	EVAKIT-75X	EV-75108	(Note 3)	–	μPD75P108CW/DW	RA75X	ST75X
μPD75108G	EVAKIT-75X	EV-75108	EP-75108GF	EV-9200G-64	μPD75P108G/GF μPD75P116GF	RA75X	ST75X
μPD75108GF	EVAKIT-75X	EV-75108	EP-75108GF	EV-9200G-64	μPD75P108G/GF μPD75P116GF	RA75X	ST75X
μPD75P108BCW	EVAKIT-75X	EV-75108	(Note 3)	–	–	RA75X	ST75X
μPD75P108CW	EVAKIT-75X	EV-75108	(Note 3)	–	–	RA75X	ST75X
μPD75P108DW	EVAKIT-75X	EV-75108	(Note 3)	–	–	RA75X	ST75X
μPD75P108G	EVAKIT-75X	EV-75108	EP-75108GF	EV-9200G-64	–	RA75X	ST75X
μPD75112CW	EVAKIT-75X	EV-75108	(Note 3)	–	μPD75P116CW	RA75X	ST75X

* Required Tools

50175

μPD75XXX Series Development Tools Selection Guide (cont)

Part Number (Note 7)	Main Board Emulator*	Add-on Board*	Emulation Probe*	Optional Socket Adapter (Note 1)	EPROM/OTP Device (Note 2)	Relocatable Assembler (Note 5)	Structured Assembler (Note 6)
μPD75112GF	EVAKIT-75X	EV-75108	EP-75108GF	EV-9200G-64	μPD75P116GF	RA75X	ST75X
μPD75116CW	EVAKIT-75X	EV-75108	(Note 3)	–	μPD75P116CW	RA75X	ST75X
μPD75116GF	EVAKIT-75X	EV-75108	EP-75108GF	EV-9200G-64	μPD75P116GF	RA75X	ST75X
μPD75P116BGF	EVAKIT-75X	EV-75108	EP-75108GF	EV-9200G-64	–	RA75X	ST75X
μPD75P116CW	EVAKIT-75X	EV-75108	(Note 3)	–	–	RA75X	ST75X
μPD75P116GF	EVAKIT-75X	EV-75108	EP-75108GF	EV-9200G-64	–	RA75X	ST75X
μPD75206CW	EVAKIT-75X	EV-75216A	(Note 3)	–	μPD75P216ACW	RA75X	ST75X
μPD75206G	EVAKIT-75X	EV-75216A	EP-75216AGF	EV-9200G-64	–	RA75X	ST75X
μPD75208CW	EVAKIT-75X	EV-75216A	(Note 3)	–	μPD75P216ACW	RA75X	ST75X
μPD75208G	EVAKIT-75X	EV-75216A	EP-75216AGF	EV-9200G-64	–	RA75X	ST75X
μPD75CG208AE	EVAKIT-75X	EV-75216A	(Note 3)	–	–	RA75X	ST75X
μPD75CG208AEA	EVAKIT-75X	EV-75216A	EP-75216AGF	EV-9200G-64	–	RA75X	ST75X
μPD75212ACW	EVAKIT-75X	EV-75216A	(Note 3)	–	μPD75P216ACW	RA75X	ST75X
μPD75212AGF	EVAKIT-75X	EV-75216A	EP-75216AGF	EV-9200G-64	–	RA75X	ST75X
μPD75216ACW	EVAKIT-75X	EV-75216A	(Note 3)	–	μPD75P216ACW	RA75X	ST75X
μPD75216AGF	EVAKIT-75X	EV-75216A	EP-75216AGF	EV-9200G-64	–	RA75X	ST75X
μPD75CG216AE	EVAKIT-75X	EV-75216A	(Note 3)	–	–	RA75X	ST75X
μPD75CG216AEA	EVAKIT-75X	EV-75216A	EP-75216AGF	EV-9200G-64	–	RA75X	ST75X
μPD75P216ACW	EVAKIT-75X	EV-75216A	(Note 3)	–	μPD75P216ACW	RA75X	ST75X
μPD75268CW	EVAKIT-75X	EV-75216A	(Note 3)	–	μPD75P216ACW	RA75X	ST75X
μPD75268GF	EVAKIT-75X	EV-75216A	EP-75216AGF	EV-9200G-64	–	RA75X	ST75X
μPD75304GF	EVAKIT-75X	EV-75308	(Note 3)	EV-9200G-80	μPD75P308GF/K	RA75X	ST75X
μPD75306GF	EVAKIT-75X	EV-75308	(Note 3)	EV-9200G-80	μPD75P308GF/K	RA75X	ST75X
μPD75308GF	EVAKIT-75X	EV-75308	(Note 3)	EV-9200G-80	μPD75P308GF/K	RA75X	ST75X
μPD75P308GF	EVAKIT-75X	EV-75308	(Note 3)	EV-9200G-80	–	RA75X	ST75X
μPD75P308K	EVAKIT-75X	EV-75308	(Note 3)	EV-9200G-80	–	RA75X	ST75X
μPD75312GF	EVAKIT-75X	EV-75308	(Note 3)	EV-9200G-80	μPD75P316GF	RA75X	ST75X
μPD75P316GF	EVAKIT-75X	EV-75308	(Note 3)	EV-9200G-80	–	RA75X	ST75X
μPD75328GC	EVAKIT-75X	EV-75328	(Note 3)	–	μPD75P328GC	RA75X	ST75X
μPD75P328GC	EVAKIT-75X	EV-75328	(Note 3)	–	–	RA75X	ST75X
μPD75402C	EVAKIT-75X	EV-75402	(Note 3)	–	μPD75P402C	RA75X	ST75X
μPD75402CT	EVAKIT-75X	EV-75402	(Note 3)	–	μPD75P402CT	RA75X	ST75X
μPD75402GB	EVAKIT-75X	EV-75402	EP-75402GB	EV-9200G-44	μPD75P402GB	RA75X	ST75X
μPD75P402C	EVAKIT-75X	EV-75402	(Note 3)	–	–	RA75X	ST75X
μPD75P402CT	EVAKIT-75X	EV-75402	(Note 3)	–	–	RA75X	ST75X
μPD75P402GB	EVAKIT-75X	EV-75402	EP-75402GB	EV-9200G-44	–	RA75X	ST75X
μPD75516GF	EVAKIT-75X	EV-75516	(Note 3)	–	μPD75P516GF/K	RA75X	ST75X
μPD75P516GF	EVAKIT-75X	EV-75516	(Note 3)	–	–	RA75X	ST75X
μPD75P516K	EVAKIT-75X	EV-75516	(Note 3)	–	–		

* Required Tools

Notes:

(1) The EV-9200G-XX is an LCC socket with the footprint of the flat package. One unit is supplied with the probe. Additional units are available as replacement parts in sets of five.

(2) All EPROM/OTP devices can be programmed using the NEC PG-1500. Refer to the PG-1500 Programming Socket Adapter Selection Guide for the appropriate socket adapter.

(3) The emulation probe is shipped with the add-on board.

(4) Preliminary information. Contact your NEC Sales Representative for further information and availability.

(5) The RA75X relocatable assembler package is provided for the following operating systems:
RA75X-D52 (MOS-DOS®)
RA75X-VVT1 (VAX/VMS™)

(6) The ST75X structures assembler preprocessor is provided with RA75X

(7) Packages:

Package	Description
C	28-pin plastic DIP
CT	28-pin plastic shrink DIP
CU	42-pin plastic shrink DIP
CW	64-pin plastic shrink DIP
DU	42-pin ceramic shrink DIP with window
DW	64-pin ceramic shrink DIP with window
E	64-pin ceramic piggy-back shrink DIP
EA	64-pin ceramic piggy-back miniflat
G	64-pin plastic miniflat
GB	44-pin plastic miniflat
GC	64 or 80-pin plastic miniflat
GF	64 or 80-pin plastic miniflat
K	80-pin plastic miniflat

1

MS-DOS is registered trademark of Microsoft Corporation.
VAX and VMS are trademarks of Digital Equipment Corporation.

1

μPD78XX Series Development Tools Selection Guide**

Part Number (Note 1)	Emulator*	Emulation Probe*	EPROM/OTP Device	PG-1500 Adapter (Note 2)	Relocatable Assembler (Note 10)	C Compiler (Note 10)
μPD7810HCW	IE-7811H-M	EV-9001-64 (Note 3)	–	–	RA87	CC87
μPD7810HG-36	IE-7811H-M	(Note 4)	–	–	RA87	CC87
μPD7811HCW	IE-7811H-M	EV-9001-64 (Note 3)	–	–	RA87	CC87
μPD7811HG-36	IE-7811H-M	(Note 4)	μPD78PG11HE (Note 6)	–	RA87	CC87
μPD78PG11HE	IE-7811H-M	(Note 4)	–	–	RA87	CC87
μPD78C10CW	IE-78C11-M	EV-9001-64 (Note 3)	–	–	RA87	CC87
μPD78C10G-36	IE-78C11-M	(Note 4)	–	–	RA87	CC87
μPD78C10G-1B	IE-78C11-M	(Note 5)	–	–	RA87	CC87
μPD78C10GF-3BE	IE-78C11-M	(Note 5)	–	–	RA87	CC87
μPD78C10L	IE-78C11-M	(Note 5)	–	–	RA87	CC87
μPD78C10ACW	IE-78C11-M (Note 8)	EV-9001-64 (Note 3)	–	–	RA87	CC87
μPD78C10AGQ-36	IE-78C11-M (Note 8)	(Note 4)	–	–	RA87	CC87
μPD78C10AGF-3BE	IE-78C11-M (Note 8)	(Note 5)	–	–	RA87	CC87
μPD78C10AL	IE-78C11-M (Note 8)	(Note 5)	–	–	RA87	CC87
μPD78C11CW	IE-78C11-M	EV-9001-64 (Note 3)	μPD78CP14CW/DW	PA-78CP14CW	RA87	CC87
μPD78C11G-36	IE-78C11-M	(Note 4)	μPD78CP14GQ-36/R μPD78CG14E	PA-78CP14GQ	RA87	CC87
μPD78C11G-1B	IE-78C11-M	(Note 5)	μPD78CP14GF-3BE	PA-78CP14GF	RA87	CC87
μPD78C11GF-3BE	IE-78C11-M	(Note 5)	μPD78CP14GF-3BE	PA-78CP14GF	RA87	CC87
μPD78C11L	IE-78C11-M	(Note 5)	μPD78CP14L	PA-78CP14L	RA87	CC87
μPD78C11ACW	IE-78C11-M (Note 8)	EV-9001-64 (Note 3)	μPD78CP14CW/DW (Note 7)	PA-78CP14CW	RA87	CC87
μPD78C11AGQ-36	IE-78C11-M (Note 8)	(Note 4)	μPD78CP14GQ-36/R (Note 7)	PA-78CP14GQ	RA87	CC87
μPD78C11AGF-3BE	IE-78C11-M (Note 8)	(Note 5)	μPD78CP14GF-3BE (Note 7)	PA-78CP14GF	RA87	CC87
μPD78C11AL	IE-78C11-M (Note 8)	(Note 5)	μPD78CP14L (Note 7)	PA-78CP14L	RA87	CC87
μPD78C12ACW	IE-78C11-M (Note 8)	EV-9001-64 (Note 3)	μPD78CP14CW/DW (Note 7)	PA-78CP14CW	RA87	CC87
μPD78C12AGQ-36	IE-78C11-M (Note 8)	(Note 4)	μPD78CP14GQ-36/R (Note 7)	PA-78CP14GQ	RA87	CC87

* Required Tools

** For all μPD78C1X devices, you may use the DDK-78C10 for evaluation purposes.

50176

μPD78XX Series Development Tools Selection Guide** (cont)

Part Number (Note 1)	Emulator*	Emulation Probe*	EPROM/OTP Device	PG-1500 Adapter (Note 2)	Relocatable Assembler (Note 10)	C Compiler (Note 10)
μPD78C12AGF-3BE	IE-78C11-M (Note 8)	(Note 5)	μPD78CP14GF-3BE (Note 7)	PA-78CP14GF	RA87	CC87
μPD78C12AL	IE-78C11-M (Note 8)	(Note 5)	μPD78CP14L (Note 7)	PA-78CP14L	RA87	CC87
μPD78C14CW	IE-78C11-M	EV-9001-64 (Note 3)	μPD78CP14CW/DW	PA-78CP14CW	RA87	CC87
μPD78C14G-36	IE-78C11-M	(Note 4)	μPD78CP14GQ-36/R μPD78CG14E	PA-78CP14GQ	RA87	CC87
μPD78C14G-1B	IE-78C11-M	(Note 5)	μPD78CP14GF	PA-78CP14GF	RA87	CC87
μPD78C14GF-3BE	IE-78C11-M	(Note 5)	μPD78CP14GF	PA-78CP14GF	RA87	CC87
μPD78C14L	IE-78C11-M	(Note 5)	μPD78CP14L	PA-78CP14L	RA87	CC87
μPD78C14AG-AB8	IE-78C11-M (Note 8)	(Note 5)	–	–	RA87	CC87
μPD78CG14E (Note 9)	IE-78C11-M	(Note 4)	–	–	RA87	CC87
μPD78CP14CW	IE-78C11-M	EV-9001-64 (Note 3)	–	PA-78CP14CW	RA87	CC87
μPD78CP14DW	IE-78C11-M	EV-9001-64 (Note 3)	–	PA-78CP14CW	RA87	CC87
μPD78CP14GQ-36	IE-78C11-M	(Note 4)	–	PA-78CP14GQ	RA87	CC87
μPD78CP14GF-3BE	IE-78C11-M	(Note 5)	–	PA-78CP14GF	RA87	CC87
μPD78CP14L	IE-78C11-M	(Note 5)	–	PA-78CP14L	RA87	CC87
μPD78CP14R	IE-78C11-M	(Note 4)	–	PA-78CP14GQ	RA87	CC87

* Required Tools

Notes:

(1) Packages

Package	Description
CW	64-pin plastic shrink DIP
DW	64-pin ceramic shrink DIP with window
E	64-pin ceramic piggyback QUIP
GF-1B	64-pin plastic miniflat (Resin Thickness: 2.05mm)
G-36	64-pin plastic QUIP
G-AB8	64-pin plastic miniflat (Interpin Pitch: 0.8mm)
GF-3BE	64-pin plastic miniflat (Resin Thickness: 2.7mm)
GQ-36	64-pin plastic QUIP
L	68-pin PLCC
R	64-pin ceramic QUIP with window

(2) By using the specified adapter, the PG-1500 EPROM programmer can be used to program the EPROM/OTP device.

(3) 64-pin shrink DIP adapter which plugs into the EP-7811HGQ emulation probe supplied with each IE.

(4) The emulation probe for the 64-pin QUIP package (EP-7811HGQ) is supplied with the IE.

MS-DOS is a trademark of Microsoft Corporation.
VAX, VMS and Ultrix are trademarks of Digital Equipment Corporation.
UNIX is a trademark of AT&T Bell Laboratories.

(5) No emulation probe available.

(6) The μPD78PG11HE is a piggy-back EPROM device in a ceramic QUIP package. It accepts 2764 EPROMs.

(7) The μPD78CP14 EPROM/OTP devices do not have pull-up resistors on ports A, B, and C.

(8) The IE-78C11-M can be used by replacing the μPD78C10G-36 with a μPD78C10AGQ-36. However, it will not be able to emulate the optional pull-up resistors on ports A, B, and C.

(9) The μPD78CG14E is a piggy-back EPROM device in a ceramic QUIP package. It accepts 27C256 and 27C256A EPROMS.

(10) The following relocatable assemblers and C Compilers are available:

RA87-D52 RA87-VVT1	(MS-DOS®) (VAX/VMS™)	Relocatable assemblers for 78XX series
CCMSD-I5DD-87 CCVMS-OT16-87 CCUNX-OT16-87	(MS-DOS) (VAX/VMS) (VAX/UNIX™) 4.2 BSD or Ultrix™)	C Compilers for 78XX Series

1

μPD78XXX Series Development Tools Selection Guide

Part Number (Note 1)	Emulator*	Emulation Probe*	EPROM/OTP Device	PG-1500 Adapter (Note 2)	Relocatable Assembler (Notes 11)	Structured Assembler (Note 12)	C Compiler (Note 13)
μPD78213CW	IE-78210-R	EP-78210CW	–	–	RA78K2	ST78K2	CC782XX
μPD78213GC-3B8	IE-78210-R	EP-78210GC	–	–	RA78K2	ST78K2	CC782XX
μPD78213GJ-5BJ	IE-78210-R	EP-78210GJ (Note 3)	–	–	RA78K2	ST78K2	CC782XX
μPD78213GQ-36	IE-78210-R	EP-78210GQ	–	–	RA78K2	ST78K2	CC782XX
μPD78213L	IE-78210-R	EP-78210L	–	–	RA78K2	ST78K2	CC782XX
μPD78214CW	IE-78210-R	EP-78210CW	μPD78P214CW/DW	PA-78P214CW	RA78K2	ST78K2	CC782XX
μPD78214GC-3B8	IE-78210-R	EP-78210GC	μPD78P214GC	PA-78P214GC	RA78K2	ST78K2	CC782XX
μPD78214GJ-5BJ	IE-78210-R	EP-78210GJ (Note 3)	μPD78P214GJ	PA-78P214GJ	RA78K2	ST78K2	CC782XX
μPD78214GQ-36	IE-78210-R	EP-78210GQ	μPD78P214GQ	PA-78P214GQ	RA78K2	ST78K2	CC782XX
μPD78214L	IE-78210-R	EP-78210L	μPD78P214L	PA-78P214L	RA78K2	ST78K2	CC782XX
μPD78P214CW	IE-78210-R	EP-78210CW	–	PA-78P214CW	RA78K2	ST78K2	CC782XX
μPD78P214DW	IE-78210-R	EP-78210CW	–	PA-78P214CW	RA78K2	ST78K2	CC782XX
μPD78P214GC-3B8	IE-78210-R	EP-78210GC	–	PA-78P214GC	RA78K2	ST78K2	CC782XX
μPD78P214GJ-5BJ	IE-78210-R	EP-78210GJ (Note 3)	–	PA-78P214GJ	RA78K2	ST78K2	CC782XX
μPD78P214GQ-36	IE-78210-R	EP-78210GQ	–	PA-78P214GQ	RA78K2	ST78K2	CC782XX
μPD78P214L	IE-78210-R	EP-78210L	–	PA-78P214L	RA78K2	ST78K2	CC782XX
μPD78220GJ-5BG	IE-78220-R	EP-78220GJ (Note 4)	–	–	RA78K2	ST78K2	CC782XX
μPD78220L	IE-78220-R	EP-78220L	–	–	RA78K2	ST78K2	CC782XX
μPD78224GJ-5BG	IE-78220-R	EP-78220GJ (Note 4)	μPD78P224GJ	PA-78P224GJ	RA78K2	ST78K2	CC782XX
μPD78224L	IE-78220-R	EP-78220L	μPD78P224L	PA-78P224L	RA78K2	ST78K2	CC782XX
μPD78P224GJ-5BG	IE-78220-R	EP-78220GJ (Note 4)	–	PA-78P224GJ	RA78K2	ST78K2	CC782XX
μPD78P224L	IE-78220-R	EP-78220L	–	PA-78P224L	RA78K2	ST78K2	CC782XX
μPD78310ACW	IE-78310A-R	(Note 5)	μPD78P312ACW/DW	PA-78P312CW	RA78K3	ST78K3	CC78K3 (Note 7)
μPD78310AGF-3BE	IE-78310A-R	EP-78310GF	μPD78P312AGF-3BE	PA-78P312GF	RA78K3	ST78K3	CC78K3 (Note 7)
μPD78310AGQ	IE-78310A-R	(Note 6)	μPD78P312AGQ/RQ	PA-78P312GQ	RA78K3	ST78K3	CC78K3 (Note 7)
μPD78310AL	IE-78310A-R	EP-78310L	μPD78P312AL	PA-78P312L	RA78K3	ST78K3	CC78K3 (Note 7)
μPD78312ACW	IE-78310A-R	(Note 5)	μPD78P312ACW/DW	PA-78P312CW	RA78K3	ST78K3	CC78K3 (Note 7)
μPD78312AGF-3BE	IE-78310A-R	EP-78310GF	μPD78P312AGF-3BE	PA-78P312GF	RA78K3	ST78K3	CC78K3 (Note 7)

* Required Tools

μPD78XXX Series Development Tools Selection Guide (cont)

Part Number (Note 1)	Emulator*	Emulation Probe*	EPROM/OTP Device	PG-1500 Adapter (Note 2)	Relocatable Assembler (Note 11)	Structured Assembler (Note 12)	C Compiler (Note 13)
μPD78312AGQ	IE-78310A-R	(Note 6)	μPD78P312AGQ/RQ	PA-78P312GQ	RA78K3	ST78K3	CC78K3 (Note 7)
μPD78312AL	IE-78310A-R	EP-78310L	μPD78P312AL	PA-78P312L	RA78K3	ST78K3	CC78K3 (Note 7)
μPD78P312ACW	IE-78310A-R	(Note 5)	–	PA-78P312CW	RA78K3	ST78K3	CC78K3 (Note 7)
μPD78P312ADW	IE-78310A-R	(Note 5)	–	PA-78P312CW	RA78K3	ST78K3	CC78K3 (Note 7)
μPD78P312AGF-3BE	IE-78310A-R	EP-78310GF	–	PA-78P312GF	RA78K3	ST78K3	CC78K3 (Note 7)
μPD78P312AGQ	IE-78310A-R	(Note 6)	–	PA-78P312GQ	RA78K3	ST78K3	CC78K3 (Note 7)
μPD78P312AL	IE-78310A-R	EP-78310L	–	PA-78P312L	RA78K3	ST78K3	CC78K3 (Note 7)
μPD78P312AR	IE-78310A-R	(Note 6)	–	PA-78P312GQ	RA78K3	ST78K3	CC78K3 (Note 7)
μPD78320GJ-5BJ	IE-78320-R	(Note 8)	(Note 8)	(Note 8)	RA78K3	ST78K3	CC78K3 (Note 7)
μPD78320L	IE-78320-R	(Note 8)	(Note 8)	(Note 8)	RA78K3	ST78K3	CC78K3 (Note 7)
μPD78322GJ-5BJ	IE-78320-R	(Note 8)	(Note 8)	(Note 8)	RA78K3	ST78K3	CC78K3 (Note 7)
μPD78322L	IE-78320-R	(Note 8)	(Note 8)	(Note 8)	RA78K3	ST78K3	CC78K3 (Note 7)
μPD71P301GF-3BE	–	–	–	PA-71P301GF	–	–	–
μPD71P301GQ-36	–	–	–	PA-71P301GQ	–	–	–
μPD71P301KA (Note 9)	–	–	–	PA-71P301KA	–	–	–
μPD71P301KB (Note 10)	–	–	–	PA-71P301KB	–	–	–
μPD71P301L	–	–	–	PA-71P301L	–	–	–

* Required Tools

Notes:

(1) Packages:

Package	Description
CW	64-pin plastic shrink DIP
DW	64-pin ceramic shrink DIP with window
GC-3B8	64-pin ceramic plastic miniflat (14mm x 14mm
GF-3BE	64-pin plastic miniflat (Resin Thickness: 2.7mm)
GJ-5BG	94-pin plastic miniflat
GJ-5BJ	74-pin plastic miniflat (20mm x 20mm)
GQ	64-pin plastic QUIP
GQ-36	64-pin plastic QUIP
KA	44-pin ceramic LCC with window
KB	64-pin ceramic LCC with window
L	44-pin PLCC (μPD71P301L) 68-pin PLCC (μPD78213/214/P214L, μPD78320/322L) 84 pin PLCC (μPD78220L, μPD78224L)
RQ	64-pin ceramic QUIP with window

(2) By using the specified adapter, the PG-1500 EPROM programmer can be used to program the EPROM/OTP device.

(3) The EP-78210GJ is a 68-pin PLCC to 74-pin miniflat package adapter for use with the EP-78210L emulation probe.

(4) The EP-78220GJ is a 84-pin PLCC to 94-pin miniflat package adapter for use with the EP-78220L emulation probe.

(5) The emulation probe for the 64-pin shrink DIP package (EP-78310CW) is supplied with the IE.

(6) The emulation probe for the 64-pin QUIP package (EP-78310GQ) is supplied with the IE.

(7) There are two C Compilers for the μPD783XX devices: CC78K3 from NEC Electronics and one from Lattice Corporation. A source code debugger is included with CC78K3 package.

(8) Please contact your NEC Sales Representative for further information.

(9) Sockets for the μPD71P301KA (44-pin LCC package) are available from Yamaichi (IC61-0444-030).

(10) Sockets for the μPD71P301KB (64-pin LCC package) are available from NEC Electronics (EV-9200G-64) in sets of five.

μPD78XXX Series Evaluation Boards Selection Guide

Part Number	Design/Development Boards	Evaluation Boards
μPD7821X	EB-78210-PC	DDK-78K2
μPD7822X	EB-78220-PC	DDK-78K2
μPD7831X	—	DDK-78310A
μPD7832X	EB-78320-PC	—

Notes:

(1) The following relocatable packages are available:

RA78K2-D52 RA78K2-VVT1	(MS-DOS®) (VAX/VMS™)	Relocatable assembler for 78XX series
RA78K3-D52 RA78K3-VVT1	(MS-DOS) (VAX/VMS)	Relocatable assembler for 78XX series

(2) The ST78K2 structured assembler processor is provided with RA78K2. The ST78K3 structured assembler preprocessor is provided with RA78K3 and CC78K3.

(3) The following C Compiler packages are available:

CCMSD-I5DD-782XX	(MS-DOS)	For μPD783XX series
CC78K3-D52	(MS-DOS)	
CC78K3-VVT1	(VAX/VMS)	

MOS-DOS is a trademark of Microsoft Corporation.
VAX and VMS are trademarks of Digital Equipment Corporation.

DSP and Speech Development Tools Selection Guide

Part Number (Note 7)	Emulator	Evaluation Board	Assembler (Note 1)	Simulator (Note 2)	EPROM/OTP Device	PG-1500 Adapter (Note 3)
μPD7720AC	EVAKIT-7720B	—	ASM77	SIM77	μPD77P20D	(Note 5)
μPD7720AL	EVAKIT-7720B (Note 4)	—	ASM77	SIM77	—	—
μPD77P20D	EVAKIT-7720B	—	ASM77	SIM77	—	—
μPD77C20AC	EVAKIT-7720B	—	ASM77	SIM77	μPD77P20D	(Note 5)
μPD77C20AL	EVAKIT-7720B (Note 4)	—	ASM77	SIM77	—	—
μPD77C20ALK	EVAKIT-7720B (Note 4)	—	ASM77	SIM77	—	—
μPD77220L	EVAKIT-77220	—	RA77230	SM77230	—	—
μPD77220R	EVAKIT-77220	—	RA77230	SM772230	μPD77P220R	PA-77P230R
μPD77P220R	EVAKIT-77220	—	RA77230	SM77230	—	PA-77P230R
μPD77230AR	EVAKIT-77230	DDK-77230	RA77230	SM77230	μPD77P230R	PA-77P230R
μPD77P230R	EVAKIT-77230	DDK-77230	RA77230	SM77230	—	PA-77P230R
μPD77C25C	EVAKIT-77C25	—	RA77C25	—	μPD77P25C/D	PA-77P25C
μPD77C25L	EVAKIT-77C25 (Note 4)	—	RA77C25	—	μPD77P25L	—
μPD77P25C	EVAKIT-77C25	—	RA77C25	—	—	PA-77P25C
μPD77P25D	EVAKIT-77C25	—	RA77C25	—	—	PA-77P25C
μPD77P25L	EVAKIT-77C25 (Note 4)	—	RA77C25	—	—	—
μPD7755C	NV-300 System	EB-7759	—	—	μPD77P56C	PA-77P56C
μPD7755G	NV-300 System	EB-7759 (Note 6)	—	—	μPD77P56G	PA-77P56C
μPD7756C	NV-300 System	EB-7759	—	—	μPD77P56C	PA-77P56C
μPD7756G	NV-300 System	EB-7759 (Note 6)	—	—	μPD77P56G	PA-77P56C
μPD77P56C	NV-300 System	EB-7759	—	—	—	PA-77P56C
μPD77P56G	NV-300 System	EB-7759 (Note 6)	—	—	—	PA-77P56C
μPD7757C	NV-300 System	EB-7759	—	—	—	—
μPD7757G	NV-300 System	EB-7759 (Note 6)	—	—	—	—
μPD7759C	NV-300 System	EB-7759	—	—	—	—
μPD7759GC	NV-300 System	EB-7759	—	—	—	—
μPD77810L	IE-77810	—	RA77810	—	—	—
μPD77810R	IE-77810	—	RA77810	—	—	—

Notes:

(1) The following assemblers are available:

Part Number	Description
ASM77-D52	Assembler for 7720 (MS-DOS®)
RA77C25-D52	Assembler for 77C25 (MS-DOS)
RA77C25-VVT1	Assembler for 77C25 (VAX/VMS™)
RA77230-D52	Assembler for 77230 (MS-DOS)
RA77230-VVT1	Assembler for 77230 (VAX/VMS)
RA77230-VXT1	Assembler for 77230 (VAX/UNIX ™) 4.2 BSD or Ultrix™)

(2) The following simulators are available:

Part Number	Description
SIM77-D52	Simulator for 7720 (MS-DOS)
SM77230-VVT1	Simulator for 77230 (VAX/UNIX)
SM77230-VXT1	Simulator for 77230 (VAX/UNIX 4.2 BSD or Ultrix)

(3) By using the specified adapter, the NEC PG-1500 EPROM programmer can be used to program the EPROM/OTP device.

(4) Please check with your NEC Sales Representative on the availability of a PLCC emulation probe.

(5) The μPD77P20D can be programmed using the EVAKIT-7720B.

(6) The EB-7759 comes with an emulation probe for only the 18-pin DIP.

(7) Packages:

Package	Description
C	18, 28, or 40-pin plastic DIP
D	28-pin ceramic DIP
G	24-pin plastic SOP
GC	52-pin plastic miniflat
L	44-or 68-pin PLCC
LK	28-pin PLCC
R	68-pin ceramic PGA

Socket Adapters and Adapter Modules

Target Chip	Socket Adapter (Note 1)	Adapter Module (Note 2)
Standard 27XXX EPROM Devices		
μPD27256 (12.5 V)	–	027A Board
μPD27256 (21 V)	–	027A Board
μPD27C256	–	027A Board
μPD27C256A	–	027A Board
μPD27C512	–	027A Board
μPD27C1000	–	027A Board
μPD27C1001	–	027A Board
μPD27C1024	–	027A Board
μPD75XX Series Devices		
μPD75P54CS	PA-75P54CS	04A Board
μPD75P54G	PA-75P54CS	04A Board
μPD75P56CS	PA-75P56CS	04A Board
μPD75P56G	PA-75P56CS	04A Board
μPD75P64CS	PA-75P54CS	04A Board
μPD75P64G	PA-75P54CS	04A Board
μPD75P66CS	PA-75P56CS	04A Board
μPD75P66G	PA-75P56CS	04A Board
μPD75XXX Series Devices		
μPD75P008CU	PA-75P008CU	04A Board
μPD75P008DU	PA-75P008CU	04A Board
μPD75P008GB	PA-75P008CU	04A Board
μPD75P028CW	PA-75P028CW	04A Board
μPD75P028GC	PA-75P028GC	04A Board
μPD75P108BCW	PA-75P108CW	04A Board
μPD75P108CW	PA-75P108CW	04A Board
μPD75P108DW	PA-75P108CW	04A Board
μPD75P108BGF	PA-75P116GF	04A Board
μPD75P108G	PA-75P116GF	04A Board
μPD75P116CW	PA-75P108CW	04A Board
μPD75P116GF	PA-75P116GF	04A Board
μPD75P216ACW	PA-75P216ACW	04A Board
μPD75P308GF	PA-75P308GF	04A Board
μPD75P308K	PA-75P308K	04A Board

Target Chip	Socket Adapter (Note 1)	Adapter Module (Note 2)
μPD75XXX Series Devices (cont)		
μPD75P316GF	PA-75P308GF	04A Board
μPD75P328GC	PA-75P328GC	04A Board
μPD75P402C	(Note 3)	027A Board
μPD75P402CT	PA-75P402CT	027A Board
μPD75P402GB	PA-75P402GB	027A Board
μPD75P516GF	PA-75P516GF	04A Board
μPD75P516K	PA-75P516K	04A Board
μPD78XX Series Devices		
μPD78CP14CW	PA-78CP14CW	027A Board
μPD78CP14DW	PA-78CP14CW	027A Board
μPD78CP14GQ	PA-78CP14GQ	027A Board
μPD78CP14GF	PA-78CP14GF	027A Board
μPD78CP14L	PA-78CP14L	027A Board
μPD78CP14R	PA-78CP14GQ	027A Board
μPD78XXX Series Devices		
μPD71P301GF	PA-71P301GF	027A Board
μPD71P301GQ	PA-71P301GQ	027A Board
μPD71P301KA	PA-71P301KA	027A Board
μPD71P301KB	PA-71P301KB	027A Board
μPD71P301L	PA-71P301L	027A Board
μPD78P214CW	PA-78P214CW	027A Board
μPD78P214GC	PA-78P214GC	027A Board
μPD78P214GJ	PA-78P214GJ	027A Board
μPD78P214GQ	PA-78P214GQ	027A Board
μPD78P214L	PA-78P214L	027A Board
μPD78P224GJ	PA-78P224GJ	027A Board
μPD78P224L	PA-78P224L	027A Board
μPD78P312ACW	PA-78P312CW	027A Board
μPD78P312ADW	PA-78P312CW	027A Board
μPD78P312AGF	PA-78P312GF	027A Board
μPD78P312AGQ	PA-78P312GQ	027A Board
μPD78P312AL	PA-78P312L	027A Board
μPD78P312AR	PA-78P312GQ	027A Board

Socket Adapters and Adapter Modules (cont)

Target Chip	Socket Adapter (Note 1)	Adapter Module (Note 2)
V-Series Devices		
μPD70P322K	PA-70P322L	027A Board
Digital Signal Processors		
μPD77P56C	PA-77P56C	04A Board
μPD77P56G	PA-77P56C	04A Board
μPD77P25C	PA-77P25C	027A Board
μPD77P25D	PA-77P25C	027A Board
μPD77P230R	PA-77P230R	027A Board

Notes:

(1) All socket adapters must be purchased separately.

(2) The 27A and 04A Adapter Modules are shipped with the PG-1500.

(3) The μPD75P402C does not require a programming socket adapter. It can be plugged directly into the 027A Board.

NEC

DIGITAL SIGNAL PROCESSORS

2

Section 2
Digital Signal Processors

μPD77C20A/7720A/77P20
DIGITAL SIGNAL PROCESSOR

Description

The μPD77C20A, μPD7720A, and μPD77P20—three signal processing interface (SPI) chips that are functionally the same—are advanced architecture microcomputers optimized for signal processing algorithms. Their speed and flexibility allow these SPIs to efficiently implement signal processing functions in a wide range of environments and applications.

The 7720A SPI, a revision of the 7720, the original mask ROM chip, uses a third less power than the 7720.

The 77C20A is a CMOS pin-for-pin compatible version of the NMOS version, 7720A. This advanced architecture CMOS microcomputer has power requirements 80 percent less than the 7720A. This low-power feature makes the 77C20A appropriate for portable applications and other designs requiring low power and low heat dissipation.

The 77P20 is an ultraviolet erasable and electrically programmable (EPROM) version of the 7720A. Program and data ROM, masked for the 7720A, are implemented in EPROM for the 77P20. The 77P20 is useful in prototype applications or in systems where product quantities are insufficient for masked ROM development.

Since the inception of 7720 and its companion EPROM version, 77P20, there have been several mask revisions to improve manufacturability and function. A 77P20 must always be used to verify the functions of a user's system before ROM code for 77C20A or 7720A is submitted, but certain early versions of 77P20 must not be used for final verification. Refer to the section on μPD77P20 for details.

Features

- ☐ Low-power CMOS: 24 mA typical current use (77C20A)
- ☐ Fast instruction execution: 240 ns with 8.333-MHz clock
- ☐ 16-bit data word
- ☐ Multi-operation instructions for fast program execution: multiply, accumulate, move data, adjust memory pointers—all in one instruction cycle
- ☐ Modified Harvard architecture with three separate memory areas
  - — Program ROM (512 x 23 bits)
  - — Data ROM (510 x 13 bits)
  - — Data RAM (128 x 16 bits)
- ☐ 16 x 16-bit multiplier; 31-bit product with every instruction
- ☐ Dual 16-bit accumulators
- ☐ External maskable interrupt
- ☐ Four-level stack for subroutines and/or interrupt
- ☐ Multiple I/O capabilities
  - — Serial: 8 or 16-bit (480 ns/bit)
  - — Parallel: 8 or 16-bit
  - — DMA
- ☐ Compatible with most μP's, including:
  - — μPD8080
  - — μPD8085
  - — μPD8086/88
  - — μPD780 (Z80®)
- ☐ Single +5-volt power supply
- ☐ Extended temperature (77C20A)
- ☐ NMOS technology (7720A, 77P20)
- ☐ Extended temperature range (7720A)

Z80 is a registered trademark of Zilog Corporation.

Applications

- ☐ Portable telecommunications equipment
- ☐ Digital filtering
- ☐ High-speed data modems
- ☐ Fast Fourier transforms (FFT)
- ☐ Speech synthesis and analysis
- ☐ Dual-tone multifrequency (DTMF) transmitters/receivers
- ☐ Equalizers
- ☐ Adaptive control
- ☐ Numerical processing

Performance Benchmarks

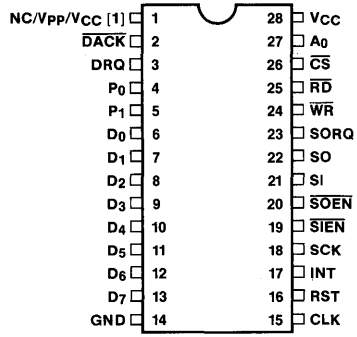
- ☐ Second-order digital filter (biquad): 2.21 μs
- ☐ Sin/cos of angles: 5.16 μs
- ☐ μ/A law to linear conversion: 0.49 μs
- ☐ FFT
  - — 32-point complex: 0.7 ms
  - — 64-point complex: 1.6 ms

Ordering Information

Part Number	Package Type	Max Frequency of Operation	Normal Temperature Range
μPD77C20AC	28-pin plastic DIP	8.33 MHz	−40 to +85°C
μPD77C20ALK	28-pin PLCC		
μPD77C20AL	44-pin PLCC		
μPD7720AC	28-pin plastic DIP	8.33 MHz	−10 to +70°C
μPD7720AL	44-pin PLCC		
μPD77P20D	28-pin cerdip	8.196 MHz	−10 to +70°C

Pin Configurations

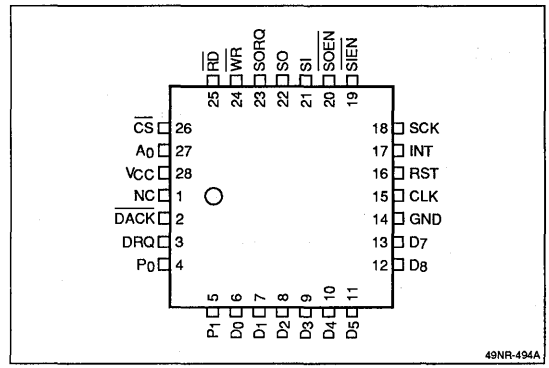
28-Pin Plastic DIP



Notes:
[1] **No connection: 77C20A, 7720A**
Must be connected for EPROM version; consult 77P20 specifications.

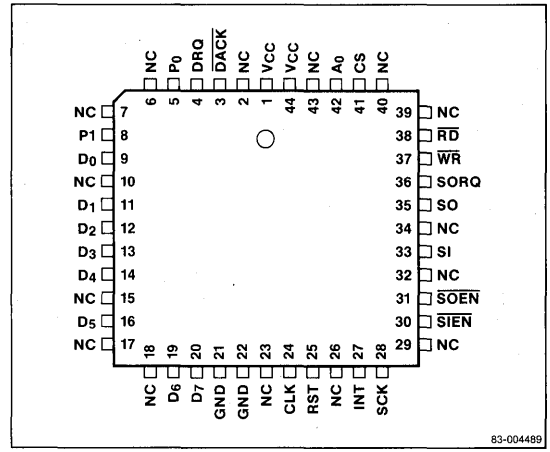
49-000905A

28-Pin PLCC



49NR-494A

44-Pin PLCC



83-004489

Pin Identification

Symbol	Function
A_0	Status/data register select input
CLK	Single-phase master clock input
$\overline{CS}$	Chip select input
D_0-D_7	Three-state I/O data bus
$\overline{DACK}$	DMA request acknowledge input
DRQ	DMA request output
INT	Interrupt input
P_0, P_1	General-purpose output control lines
$\overline{RD}$	Read control signal input
RST	Reset input
SCK	Serial data I/O clock input
SI	Serial data input
$\overline{SIEN}$	Serial input enable input
SO	Three-state serial data output
$\overline{SOEN}$	Serial output enable input
SORQ	Serial data output request
$\overline{WR}$	Write control signal input
GND	Ground
V_{CC}	+5 V power supply
$NC/V_{PP}/V_{CC}$	No connection (77C20A,7720A)/ programming voltage (77P20)

Pin Functions

A_0 [Status Data Register Select]

This input selects data register for read/write (low) or status register for read (high).

CLK

This is the single-phase master clock input.

$\overline{CS}$ [Chip Select]

This input enables data transfer through the data port with RD or WR.

D_0-D_7 [Data Bus]

This three-state I/O data bus transfers data between the data register or status register and the external data bus.

$\overline{DACK}$ [DMA Request Acknowledge]

This input indicates to the SPI that the data bus is ready for a DMA transfer (DACK = CS AND $A_0 = 0$).

DRQ [DMA Request]

This output signals that the SPI is requesting a data transfer on the data bus.

INT [Interrupt]

A low-to-high transition on this pin executes a call instruction to location 100H, if interrupts were previously enabled.

P_0, P_1

These pins are general-purpose output control lines.

$\overline{RD}$ [Read Control Signal]

This input latches data from the data or status register to the data port where it is read by an external device.

RST [Reset]

This input initializes the SPI internal logic and sets the PC to 0.

SCK [Serial Data I/O Clock]

When this input is high, a serial data bit is transferred.

SI [Serial Data Input]

This pin inputs 8- or 16-bit serial data words from an external device such as an A/D converter.

$\overline{SIEN}$ [Serial Input Enable]

This input enables the shift clock to the serial input register.

SO [Serial Data Output]

This three-state port outputs 8- or 16-bit data words to an external device such as a D/A converter.

$\overline{SOEN}$ [Serial Output Enable]

This input enables the shift clock to the serial output register.

SORQ [Serial Data Output Request]

This output specifies to an external device that the serial data register has been loaded and is ready for output. SORQ is reset when the entire 8- or 16-bit word has been transferred.

$\overline{WR}$ [Write Control Signal]

This input writes data from the data port into the data register.

GND

This is the connection to ground.

V_{CC} [Power Supply]

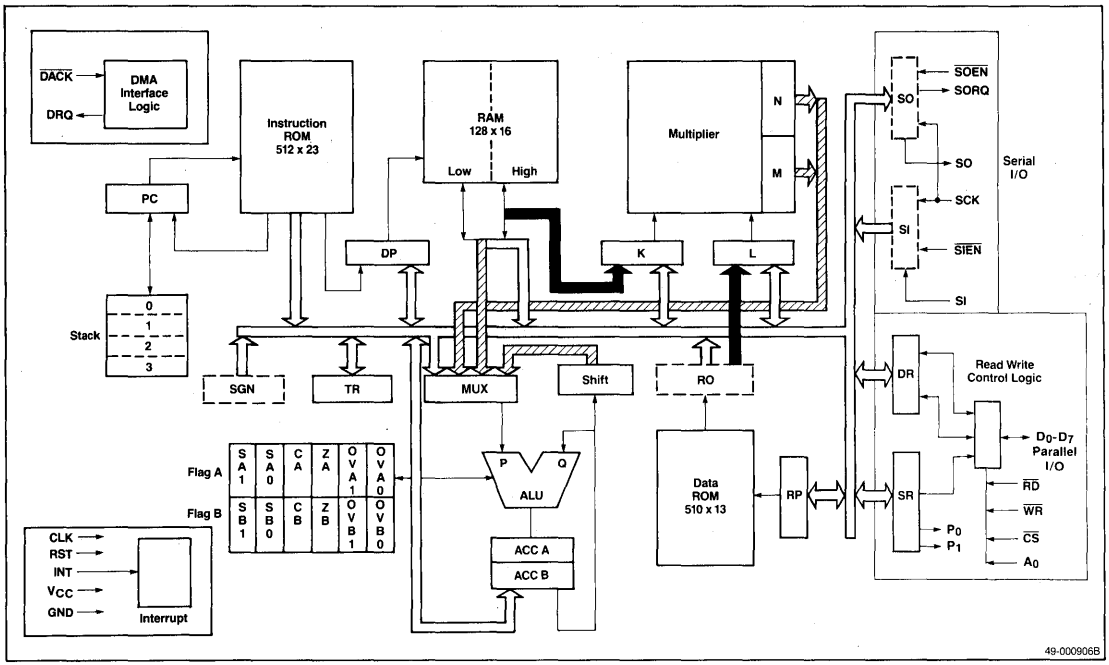
This pin is the +5-volt power supply.

NC/V_{PP}/V_{CC}

This pin is not internally connected in the 77C20A and 7720A. In the 77P20, this pin inputs the programming voltage (V_{PP}) when the part is being programmed.

This pin must be connected to V_{CC} for proper 77P20 operation. Consult the section on the μPD77P20 for details.

Block Diagram



Functional Description

The primary bus (unshaded in the block diagram) makes a data path between all of the registers (including I/O), memory, and the processing sections. This bus is referred to as the IDB (internal data bus). The multiplier input registers K and L can be loaded not only from the IDB but alternatively via buses (darkened in the block diagram) directly from RAM to the K register and directly from data ROM to the L register. Output from the multiplier in the M and N registers is typically added via buses (shaded in the block diagram) to either accumulator A or B as part of a multi-operation instruction.

The SPI is a complete 16-bit microcomputer on a single chip. ROM space provides program and coefficient storage; the on-chip RAM may be used for temporary data, coefficients, and results. A 16-bit arithmetic/logic unit (ALU) and a separate 16 x 16-bit, fully-parallel multiplier provide computational power. This combination allows the implementation of a "sum of products" operation in a single 240-ns instruction cycle. In addition, each arithmetic instruction allows a number of data movement operations to further increase throughput.

Two serial I/O ports interface to codecs and other serially-oriented devices; a parallel port provides both data and status information to conventional microprocessors. Handshaking signals, including DMA controls, allow the SPI to act as a sophisticated programmable peripheral as well as a stand-alone microcomputer.

Memory

Memory is divided into three types: instruction ROM, data ROM, and data RAM. The 512 x 23-bit words of instruction ROM are addressed by a 9-bit program counter that can be modified by an external reset, interrupt, call, jump, or return instruction.

The data ROM is organized in 510 x 13-bit words that are addressed through a 9-bit ROM pointer (RP register). The RP may be modified simultaneously with arithmetic instructions so that the next value is available for the next instruction. The data ROM is ideal for storing the necessary coefficients, conversion tables, and other constants for your processing needs.

Do not use data ROM locations 0 and 1 in the 77C20A or 7720A. These locations are reserved for storage of test pattern data. (When submitting code, set these locations to 0). Note that 77P20 allows use of these locations, but using them is not advised.

The data RAM is 128 x 16-bit words and is addressed through a 7-bit data pointer (DP register). The DP has extensive addressing features that operate simultaneously with arithmetic instructions, eliminating additional time for addressing or address modification.

Arithmetic Capabilities

One of the unique features of the SPI's architecture is its arithmetic facilities. With a separate multiplier, ALU, and multiple internal data paths, the SPI is capable of carrying out a multiply, an add or other arithmetic operation, and a data move between internal registers in a single instruction cycle.

ALU

The ALU is a 16-bit two's complement unit capable of executing 16 distinct operations on virtually any of the SPI's internal registers, thus giving the SPI both speed and versatility for efficient data management.

Accumulators [ACCA/ACCB]

Associated with the ALU are two 16-bit accumulators, each with its own set of flags, which are updated at the end of each arithmetic instruction (except NOP). Table 1 shows the ACC A/B flag registers. In addition to zero result, sign, carry, and overflow flags, the SPI incorporates auxiliary overflow and sign flags (SA1, SB1, OVA1, OVB1). These flags enable the detection of an overflow condition and maintain the correct sign after as many as three successive additions or subtractions.

Table 1. ACC A/B Flag Registers

Flag A	SA1	SA0	CA	ZA	OVA1	OVA0
Flag B	SB1	SB0	CB	ZB	OVB1	OVB0

Sign Register [SGN]

When OVA1 is set, the SA1 bit will hold the corrected sign of the overflow. The SGN register will use SA1 to automatically generate saturation constants 7FFFH(+) or 8000H(−) to permit efficient limiting of a calculated value. The SGN register is not affected by arithmetic operations on accumulator B, but flags SB1, SB0, CB, ZB, OVB1 and OVB0 are affected.

Multiplier

Thirty-one bit results are developed by a 16 x 16-bit two's complement multiplier in 240 ns. The result is automatically latched to two 16-bit registers, M and N, at the end of each instruction cycle. The sign bit and 15 higher bits are in M and the 15 lower bits are in N; the LSB in N is zero. A new product is available for use after every instruction cycle, providing significant advantages in maximizing processing speed for real-time signal processing.

Stack

The SPI contains a four-level program stack for efficient program usage and interrupt handling.

Interrupt

The SPI supports a single-level interrupt. Upon sensing a high level on the INT terminal, a subroutine call to location 100H is executed. The EI bit of the status register automatically resets to 0, disabling the interrupt facility until it is reenabled under program control.

Input/Output

General

The SPI has three communication ports, as shown in figure 1: two serial and one 8-bit parallel, each with its own control lines for interface handshaking. Parallel port operation is software-configurable to be in either polled mode or DMA mode. A general-purpose, two-line output port rounds out a full complement of interface capability.

Serial I/O

The two shift registers (SI, SO) are software-configurable to single- or double-byte transfers. The shift registers are externally clocked (SCK) to provide a simple interface between the SPI and serial peripherals such as A/D and D/A converters, codecs, or other SPIs. Figure 2 shows serial I/O timing

Figure 1. SPI Communication Ports

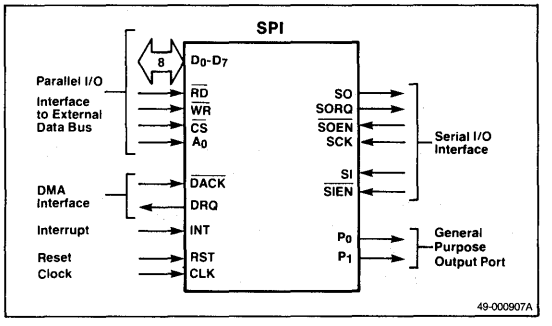
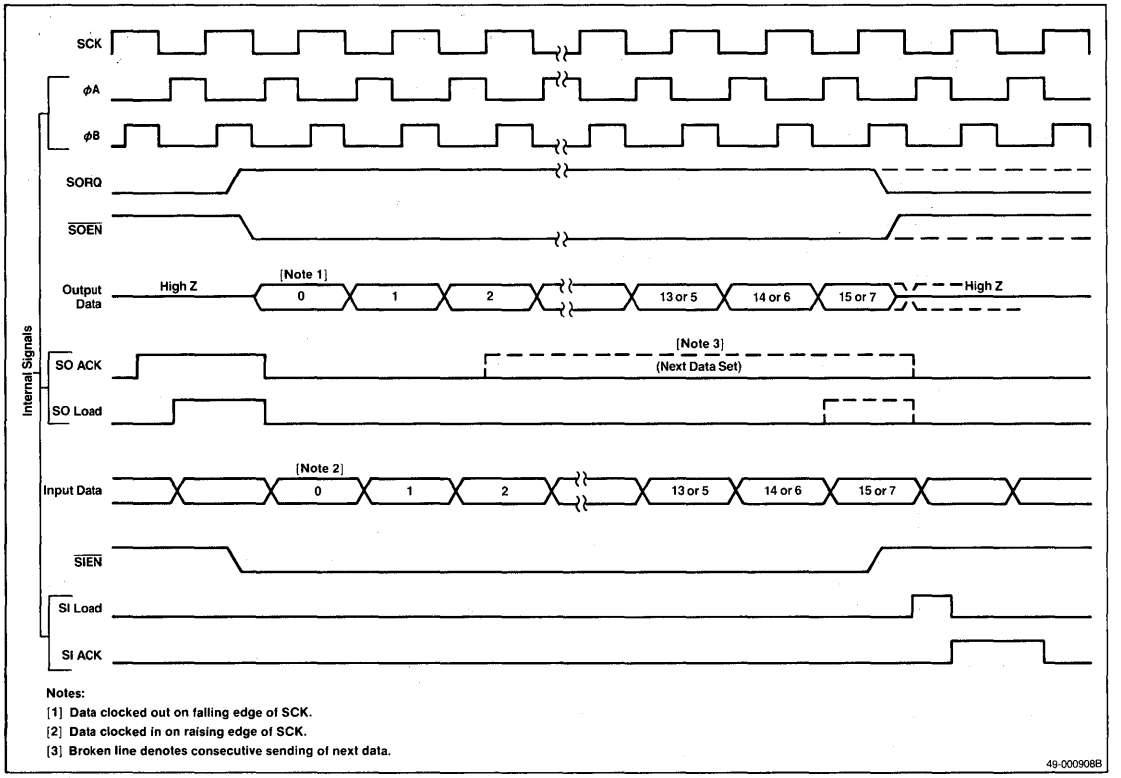


Figure 2. Serial I/O Timing



Parallel I/O

The 8-bit parallel I/O port may be used for transferring data or reading the SPI's status, as shown in table 2. Data transfer is handled through a 16-bit data register (DR) that is software-configurable for double- or single-byte data transfers. The port is ideally suited for operating with 8080, 8085, and 8086 processor buses and may be used with other processors and computer systems.

DMA Mode Option

Parallel data transfers may be controlled (optionally) via DMA control lines DRQ and $\overline{DACK}$. DMA mode allows high-speed transfers and reduced processor overhead. When in DMA mode, $\overline{DACK}$ input resets DRQ output when data transfer is completed. $\overline{DACK}$ does not affect any status register bit or flag bit.

Table 2. Parallel R/W Operation

$\overline{CS}$	A_0	$\overline{WR}$	$\overline{RD}$	Operation
1	X	X	X	No effect on internal operation; D_0-D_7 are at high impedance levels.
X	X	1	1	
0	0	0	1	Data from D_0-D_7 is latched to DR (Note 1)
0	0	1	0	Contents of DR are output to D_0-D_7 (Note 1)
0	1	0	1	Illegal (SR is read only)
0	1	1	0	Eight MSBs of SR are output to D_0-D_7
0	X	0	0	Illegal (may not read and write simultaneously)

Note:

(1) Eight MSBs or 8 LSBs of data register (DR) are used, depending on DR status bit (DRS). The condition of DACK = 0 is equivalent to A_0 = CS = 0.

Status Register

The status register, shown in figure 3, is a 16-bit register in which the eight most significant bits may be read by the system's microprocessor for the latest parallel data I/O status. The RQM and DRS bits can only be affected by parallel data moves. The other bits can be written to (or read) by the SPI's load immediate (LDI) or move (MOV) instructions. The EI bit is automatically reset when an interrupt is serviced.

Figure 3. Status Register (SR)

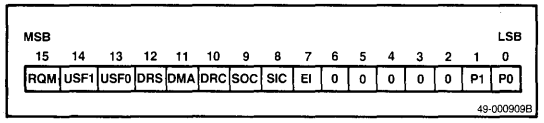


Table 3. Status Register Flags

Flag	Description
RQM (Request for Master)	A read or write from DR to IDB sets RQM = 1. An external read (write) resets RQM = 0.
USF1 and USF0 (User Flags 1 and 0)	General-purpose flags which may be read by an external processor for user-defined signaling
DRS (DR Status)	For 16-bit DR transfers (DRC = 0). DRS = 1 after first 8 bits have been transferred. DRS = 0 after all 16 bits have been transferred.
DMA (DMA Enable)	DMA = 0 (Non-DMA transfer mode) DMA = 1 (DMA transfer mode)
DRC (DR control)	DRC = 0 (16-bit mode) DRC = 1 (8-bit mode)
SOC (SO Control)	SOC = 0 (16-bit mode) SOC = 1 (8-bit mode)
SIC (SI Control)	SIC = 0 (16-bit mode) SIC = 1 (8-bit mode)
EI (Enable Interrupt)	EI = 0 (interrupts disabled) EI = 1 (interrupts enabled)
P1, P0 (Ports 0 and 1)	P0 and P1 directly control the state of output pins P_0 and P_1

Instructions

The SPI has three types of instructions: Load Immediate, Branch, and the multifunction OP instruction. Each type takes the form of a 23-bit word and executes in 240 ns.

Instruction Timing

To control the execution of instructions, the external 8-MHz clock is divided into four phases for internal execution. The various elements of the 23-bit instruction word are executed in a set order. Multiplication automatically begins first. Also, data moves from source to destination before other elements of the instruction. Data being moved on the internal data bus (IDB) is available for use in ALU operations (if P-select field of the instruction specifies IDB). However, if the accumulator specified in the ASL field is also specified as the destination of the data move, the ALU operation becomes a NOP, as the data move supersedes the ALU operation.

Pointer modifications occur at the end of the instruction cycle after their values have been used for data moves. The result of multiplication is available at the end of the instruction cycle for possible use in the next instruction. If a return is specified as part of an OP instruction, it is executed last.

An assembly language OP instruction may consist of what looks like one to six lines of assembly code, but all of these lines are assembled together into one 23-bit instruction word. Therefore, the order of the six lines makes no difference in the order of execution described above. However, for understanding the SPI's operation and to eliminate confusion, write assembly code in the order described; that is: data move, ALU operations, data pointer modifications, and then return.

OP/RT Instruction Field Specification

Figure 4 illustrates the OP/RT instruction field specification. There are two instructions of this type, both of which are capable of executing all ALU functions listed in table 5. The ALU functions operate on the value specified by the P-select field (see table 4).

Besides the arithmetic functions, these instructions can also (1) modify the RAM data pointer DP, (2) modify the data ROM pointer RP, and (3) move data along the on-chip data bus from a source register to a destination register (the possible source and destination registers are listed in tables 10 and 11, respectively). The difference in the two instructions of this type is that RT executes a subroutine or interrupt return at the end of the instruction cycle, but the OP does not. Tables 6, 7, 8, and 9 show the ASL, DPL, DPH, and RPDCR fields, respectively.

Figure 4. OP/RT Instruction Field

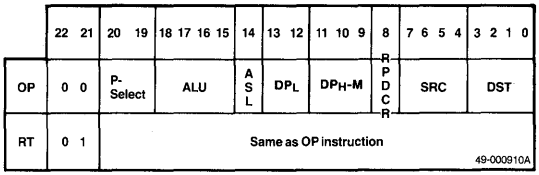


Table 4. P-Select Field

Mnemonic	D_{20}	D_{19}	ALU Input
RAM	0	0	RAM
IDB	0	1	Internal Data Bus (Note 1)
M	1	0	M Register
N	1	1	N Register

Note:

(1) Any value on the on-chip data bus. Value may be selected from any of the registers listed in table 6 source register selections.

Table 5. ALU Field

Mnemonic	D_{18}	D_{17}	D_{16}	D_{15}	ALU Function	SA1 SB1	SA0 SB0	CA CB	ZA ZB	OVA1 OVB1	OVA0 OVB0
NOP	0	0	0	0	No operation	—	—	—	—	—	—
OR	0	0	0	1	OR	x	Δ	0	Δ	0	0
AND	0	0	1	0	AND	x	Δ	0	Δ	0	0
XOR	0	0	1	1	Exclusive OR	x	Δ	0	Δ	0	0
SUB	0	1	0	0	Subtract	Δ	Δ	Δ	Δ	Δ	Δ
ADD	0	1	0	1	ADD	Δ	Δ	Δ	Δ	Δ	Δ
SBB	0	1	1	0	Subtract with borrow	Δ	Δ	Δ	Δ	Δ	Δ
ADC	0	1	1	1	Add with carry	Δ	Δ	Δ	Δ	Δ	Δ
DEC	1	0	0	0	Decrement ACC	Δ	Δ	Δ	Δ	Δ	Δ
INC	1	0	0	1	Increment ACC	Δ	Δ	Δ	Δ	Δ	Δ
CMP	1	0	1	0	Complement ACC (one's complement)	x	Δ	0	Δ	0	0
SHR1	1	0	1	1	1-Bit right shift	x	Δ	Δ	Δ	0	0
SHL1	1	1	0	0	1-Bit left shift	x	Δ	Δ	Δ	0	0
SHL2	1	1	0	1	2-Bit left shift	x	Δ	0	Δ	0	0
SHL4	1	1	1	0	4-Bit left shift	x	Δ	0	Δ	0	0
XCHG	1	1	1	1	8-Bit exchange	x	Δ	0	Δ	0	0

Note:

Δ May be affected, depending on the results
— Previous status can be held
0 Reset
x Indefinite

Table 6. ASL Field

Mnemonic	D_{14}	ACC Selection
ACCA	0	ACCA
ACCB	1	ACCB

Table 7. DPL Field

Mnemonic	D_{13}	D_{12}	Low DP Modify (DP_3-DP_0)
DPNOP	0	0	No operation
DPINC	0	1	Increment DPL
DPDEC	1	0	Decrement DPL
DPCLR	1	1	Clear DPL

Table 8. DPH Field

Mnemonic	D_{11}	D_{10}	D_9	High DP Modify
M0	0	0	0	Exclusive OR of DPH (DP_6-DP_4) with the mask defined by the three bits (D_{11}-D_9) of the DPH field
M1	0	0	1	
M2	0	1	0	
M3	0	1	1	
M4	1	0	0	
M5	1	0	1	
M6	1	1	0	
M7	1	1	1	

Table 9. RPDCR Field

Mnemonic	D_8	RP Operation
RPNOP	0	No operation
RPDEC	1	Decrement RP

Table 10. SRC Field

Mnemonic	D_7	D_6	D_5	D_4	Source Register
NON	0	0	0	0	No register
A	0	0	0	1	ACCA (Accumulator A)
B	0	0	1	0	ACCB (Accumulator B)
TR	0	0	1	1	TR temporary register
DP	0	1	0	0	DP data pointer
RP	0	1	0	1	RP ROM pointer
RO	0	1	1	0	RO ROM output data
SGN	0	1	1	1	SGN sign register
DR	1	0	0	0	DR data register
DRNF	1	0	0	1	DR no flag (Note 1)
SR	1	0	1	0	SR status register
SIM	1	0	1	1	SI serial in MSB (Note 2)
SIL	1	1	0	0	SI serial in LSB (Note 3)
K	1	1	0	1	K register
L	1	1	1	0	L register
MEM	1	1	1	1	RAM

Notes:

(1) DR to IDB, RQM not set. In DMA, DRQ not set.

(2) First bit in goes to MSB, last bit to LSB.

(3) First bit goes to LSB, last bit to MSB (bit reversed).

Jump/Call/Branch

Figure 5 shows the JP instruction field specification.

Three types of program counter modifications are accommodated by the processor and are listed in table 12. All the instructions, if unconditional or if the specified condition is true, take their next program execution address from the next address field (NA); otherwise PC = PC + 1.

For the conditional jump instruction, the condition field specifies the jump condition. Table 13 lists all the instruction mnemonics of the jump/call/branch codes.

Load Data [LDI]

Figure 6 shows the LD instruction field specification.

The load data instruction will take the 16-bit value contained in the immediate data field (ID) and place it in the location specified by the destination field (DST) (see table 11).

Table 11. DST Field

Mnemonic	D_3	D_2	D_1	D_0	Destination Register
@NON	0	0	0	0	No register
@A	0	0	0	1	ACCA (Accumulator A)
@B	0	0	1	0	ACCB (Accumulator B)
@TR	0	0	1	1	TR temporary register
@DP	0	1	0	0	DP data pointer
@RP	0	1	0	1	RP ROM pointer
@DR	0	1	1	0	DR data register
@SR	0	1	1	1	SR status register
@SOL	1	0	0	0	SO serial out LSB (Note 1)
@SOM	1	0	0	1	SO serial out MSB (Note 2)
@K	1	0	1	0	K (Mult)
@KLR	1	0	1	1	IDB → K, ROM → L (Note 3)
@KLM	1	1	0	0	Hi RAM → K, IDB → L (Note 4)
@L	1	1	0	1	L (Mult)
@NON	1	1	1	0	No register
@MEM	1	1	1	1	RAM

Notes:

(1) LSB is first bit out.

(2) MSB is first bit out.

(3) Internal data bus to K, and ROM to L register.

(4) Contents of RAM address specified by $DP_6 = 1$, is placed in K register, IDB is placed in L (that is, 1, DP_5, DP_4 DP_3-DP_0).

Table 12. BRCH Field

D_{C20}	D_{19}	D_{18}	Branch Instruction
1	0	0	Unconditional jump
1	0	1	Subroutine call
0	1	0	Conditional jump

Figure 5. JP Instruction Field Specification

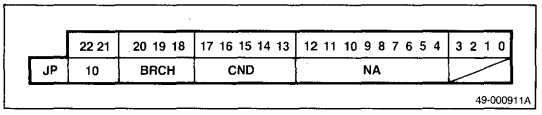


Figure 6. LD Instruction Field Specification

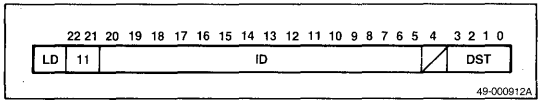


Table 13. BRCH/CND Fields

Mnemonic	D_{20}	D_{19}	D_{18}	D_{17}	D_{16}	D_{15}	D_{14}	D_{13}	Conditions (Note 1)
JMP	1	0	0	0	0	0	0	0	No condition
CALL	1	0	1	0	0	0	0	0	No condition
JNCA	0	1	0	0	0	0	0	0	CA = 0
JCA	0	1	0	0	0	0	0	1	CA = 1
JNCB	0	1	0	0	0	0	1	0	CB = 0
JCB	0	1	0	0	0	0	1	1	CB = 1
JNZA	0	1	0	0	0	1	0	0	ZA = 0
JZA	0	1	0	0	0	1	0	1	ZA = 1
JNZB	0	1	0	0	0	1	1	0	ZB = 0
JZB	0	1	0	0	0	1	1	1	ZB = 1
JNOVA0	0	1	0	0	1	0	0	0	OVA0 = 0
JOVA0	0	1	0	0	1	0	0	1	OVA0 = 1
JNOVB0	0	1	0	0	1	0	1	0	OVB0 = 0
JOVB0	0	1	0	0	1	0	1	1	OVB0 = 1
JNOVA1	0	1	0	0	1	1	0	0	OVA1 = 0
JOVA1	0	1	0	0	1	1	0	1	OVA1 = 1
JNOVB1	0	1	0	0	1	1	1	0	OVB1 = 0
JOVB1	0	1	0	0	1	1	1	1	OVB1 = 1
JNSA0	0	1	0	1	0	0	0	0	SA0 = 0
JSA0	0	1	0	1	0	0	0	1	SA0 = 1
JNSB0	0	1	0	1	0	0	1	0	SB0 = 0
JSB0	0	1	0	1	0	0	1	1	SB0 = 1
JNSA1	0	1	0	1	0	1	0	0	SA1 = 0
JSA1	0	1	0	1	0	1	0	1	SA1 = 1
JNSB1	0	1	0	1	0	1	1	0	SB1 = 0
JSB1	0	1	0	1	0	1	1	1	SB1 = 1
JDPL0	0	1	0	1	1	0	0	0	DPL = 0
JDPLF	0	1	0	1	1	0	0	1	DPL = FH
JNSIAK	0	1	0	1	1	0	1	0	SI ACK = 0
JSIAK	0	1	0	1	1	0	1	1	SI ACK = 1
JNSOAK	0	1	0	1	1	1	0	0	SO ACK = 0
JSOAK	0	1	0	1	1	1	0	1	SO ACK = 1
JNRQM	0	1	0	1	1	1	1	0	RQM = 0
JRQM	0	1	0	1	1	1	1	1	RQM = 1

Note:

(1) BRCH or CND values not in this table are prohibited.

Absolute Maximum Ratings

Supply voltage, V_{CC}	
77C20A	−0.5 to +7.0 V
7720A	−0.5 to +7.0 V
77P20	−0.3 to +7.0 V
Programming voltage, V_{PP} (77P20)	−0.3 to +22 V
Input voltage, V_I	
77C20A	−0.5 to V_{CC} + 0.5 V
7720A	−0.5 to +7.0 V
77P20	−0.3 to +7.0 V
Output voltage, V_O	
77C20A	−0.5 to V_{CC} + 0.5 V
7720A	−0.5 to +7.0 V
77P20	−0.3 V to +7.0 V
Operating temperature, T_{OPT}	
77C20A	−40 to +85°C
7720A, 77P20	−10 to +70°C
Storage temperature, T_{STG}	−65 to +150°C

Comment: Exposing the device to stresses above those listed in Absolute Maximum Ratings could cause permanent damage. The device is not meant to be operated under conditions outside the limits described in the operational sections of this specification. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

DC Characteristics

T_A = −10 to +70°C; V_{CC} = +5 V ±5%

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
Input low voltage	V_{IL}					
77C20A		−0.3		0.8	V	
7720A, 77P20		−0.5		0.8	V	
Input high voltage	V_{IH}					
77C20A		2.2		V_{CC} + 0.3	V	
7720A, 77P20		2.0		V_{CC} + 0.5	V	
CLK low voltage	$V_{\phi L}$					
77C20A		−0.3		0.45	V	
7720A, 77P20		−0.5		0.45	V	
CLK high voltage	$V_{\phi H}$					
77C20A		3.5		V_{CC} + 0.3	V	
7720A, 77P20		3.5		V_{CC} + 0.5	V	
Output low voltage	V_{OL}			0.45	V	I_{OL} = 2.0 mA
Output high voltage	V_{OH}	2.4			V	I_{OH} = −400 μA
Input load current	I_{LIL}			−10	μA	V_{IN} = 0 V
Input load current	I_{LIH}			10	μA	V_{IN} = V_{CC}

DC Characteristics (cont)

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
Output float leakage	I_{LOL}			−10	μA	$V_{OUT} = 0.47$ V
Output float leakage	I_{LOH}			10	μA	$V_{OUT} = V_{CC}$
Power supply current	I_{CC}					
77C20A			24	40	mA	$f_{CLK} = 8.192$ MHz
7720A			120	170	mA	
77P20			270	350	mA	
V_{PP} current (77P20 only)	I_{PP}			70	mA	Program mode max pulse current (Note 1)
		0.5		3.0	mA	Program verify, inhibit (Note 2)

Notes:

(1) $V_{PP} = 21 \pm 0.5$ V

(2) For K-level parts, V_{PP} max = $(V_{CC} - 0.6\text{ V}) + 0.25$ V
V_{PP} min = $(V_{CC} - 0.6\text{ V}) - 0.25$ V
For all other step levels: V_{PP} max = $V_{CC} + 0.25$ V
V_{PP} min = $V_{CC} - 0.85$ V

Capacitance

Parameter	Symbol	Limits Min	Max	Unit	Test Conditions
CLK, SCK capacitance	C_ϕ		20	pF	$f_c = 1$ MHz
Input pin capacitance	C_{IN}		10	pF	
Output pin capacitance	C_{OUT}		20	pF	

AC Characteristics

$T_A = -10$ to $+70$°C; $V_{CC} = +5$ V ±5%

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
CLK cycle time	ϕ_{CY}					
77C20A, 7720A		120		2000	ns	
77P20		122		2000	ns	
CLK pulse width	ϕ_D	60			ns	(Note 4)
CLK rise time	ϕ_R			10	ns	(Note 1)
CLK fall time	ϕ_F			10	ns	(Note 1)
Address setup time for $\overline{RD}$	t_{AR}	0			ns	
Address hold time for $\overline{RD}$	t_{RA}	0			ns	
$\overline{RD}$ pulse width	t_{RR}	250			ns	

AC Characteristics (cont)

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
Data delay from $\overline{RD}$	t_{RD}			150	ns	$C_L = 100$ pF
Read to data floating	t_{DF}	10		100	ns	$C_L = 100$ pF
Address setup time for $\overline{WR}$	t_{AW}	0			ns	
Address hold time for $\overline{WR}$	t_{WA}	0			ns	
$\overline{WR}$ pulse width	t_{WW}	250			ns	
Data setup time for $\overline{WR}$	t_{DW}	150			ns	
Data hold time for $\overline{WR}$	t_{WD}	0			ns	
$\overline{RD}$, $\overline{WR}$, recovery time	t_{RV}	250			ns	(Note 2)
DRQ delay	t_{AM}			150	ns	$C_L = 100$ pF
$\overline{DACK}$ delay time	t_{DACK}	1			ϕD	(Note 2)
$\overline{DACK}$ pulse width	t_{DD}					
77C20A		250			ns	
7720A		250		2000	ns	
77P20		250		50,000	ns	
SCK cycle time	t_{SCY}	480		DC	ns	
SCK pulse width	t_{SCK}	230			ns	
SCK rise/fall time	t_{RSC}/t_{FSC}			20	ns	
SORQ delay	t_{DRQ}	30		150	ns	$C_L = 100$ pF
$\overline{SOEN}$ hold time	t_{CSO}	30			ns	
$\overline{SOEN}$ setup time	t_{SOC}	50			ns	
SO delay from SCK = low	t_{DCK}			150	ns	
SO delay from SCK before 1st bit (Note 3)	t_{DZRQ}	20		300	ns	(Note 2)
SO delay from SCK	t_{DZSC}	20		300	ns	(Note 2)
SO delay for $\overline{SOEN}$	t_{DZE}	20		180	ns	(Note 2)
$\overline{SOEN}$ to SO floating	t_{HZE}	20		200	ns	(Note 2)
SCK to SO floating with SORQ high	t_{HZSC}	20		300	ns	(Note 2)
SO delay from SCK for last bit	t_{HZRQ}	70		300	ns	(Note 2)
$\overline{SIEN}$, SI setup time	t_{DC}	55			ns	(Note 2)
$\overline{SIEN}$, SI hold time	t_{CD}	30			ns	

2

AC Characteristics (cont)

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
P_0, P_1 delay	t_{DP}			ϕ_{CY} +150	ns	
RST pulse width	t_{RST}	4			ϕ_{CY}	
INT pulse width	t_{INT}	8			ϕ_{CY}	

Notes:

(1) Voltage at timing measuring point: 1.0 V and 3.0 V.

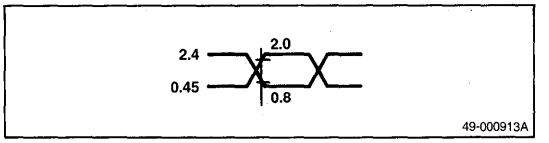
(2) Voltage at AC timing measuring point:
$V_{IL} = V_{OL} = 0.8$ V
$V_{IH} = V_{OH} = 2.0$ V

(3) SO goes out of tristate, but data is not valid yet.

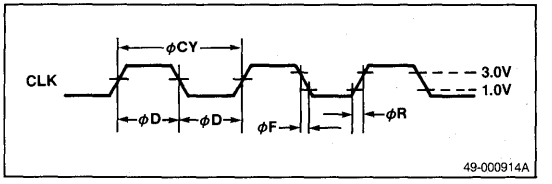
(4) Pulse width includes CLK rise and fall times. Refer to Clock Timing Waveform.

Timing Waveforms

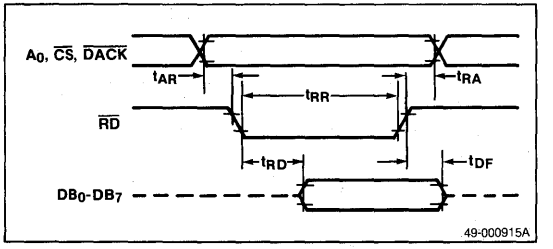
Input Waveform of AC Test (except CLK)



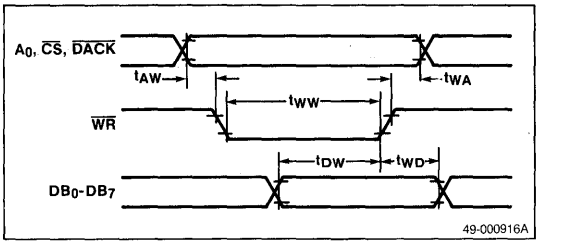
Clock



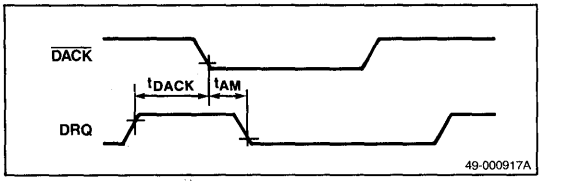
Read Operation



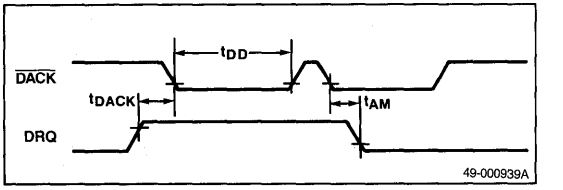
Write Operation



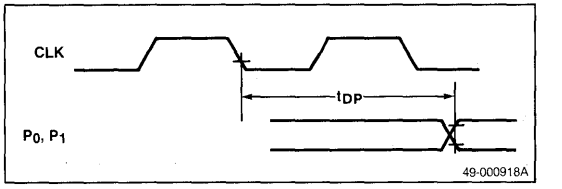
DMA Operation



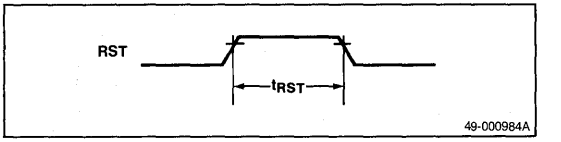
16-Bit Transfer Mode



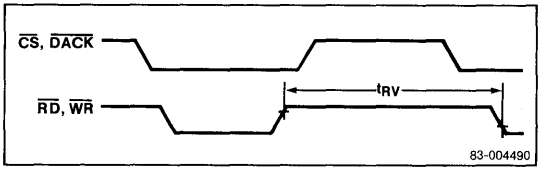
Port Output



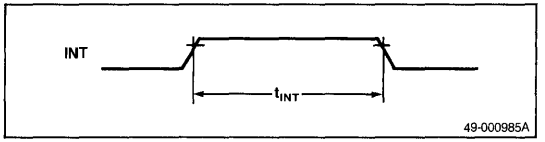
Reset



Read/Write Cycle



Interrupt



Serial Timing

Serial Output, Case 1

Figure 7 shows serial output timing when $\overline{SOEN}$ is asserted in response to SORQ when SCK is low. If $\overline{SOEN}$ is held inactive until after SORQ is asserted, and then $\overline{SOEN}$ is asserted while SCK is low ($\overline{SOEN}$ should be held inactive until the period of t_{CSO}, after the falling edge of SCK), SO will become active but not valid t_{DZSC} after the next rising edge of SCK. SO will become valid with the first bit t_{DCK} after the next falling edge of SCK, for use by an external device at the subsequent rising edge of SCK.

Subsequent bits will be shifted out t_{DCK} after subsequent falling edges of SCK, for use at subsequent rising edges of SCK. The last bit to be shifted out will also follow this pattern, and will be held valid t_{HZRQ} after the corresponding rising edge of SCK at which it is to be used. SORQ will be held t_{DRQ} after this same rising edge of SCK, and then removed. $\overline{SOEN}$ should be released at least t_{SOC} before the next falling edge of SCK.

Serial Output, Case 2

Figure 8 shows timing for serial output when $\overline{SOEN}$ is asserted in response to SORQ when SCK is high. If $\overline{SOEN}$ is held inactive until after SORQ is asserted, and then $\overline{SOEN}$ is asserted while SCK is high (at least t_{SOC} before the falling edge of SCK), SO will become active but not valid t_{DZE} after the falling edge of $\overline{SOEN}$. SO will become valid t_{DCK} after the falling edge of SCK, for use by an external device at the subsequent rising edge of SCK.

Note that, although figure 8 shows $\overline{SOEN}$ being asserted during a different SCK pulse than the one in which SORQ is asserted, it is permissible for these to occur during the same pulse of SCK, as long as $\overline{SOEN}$ is still asserted t_{SOC} before the falling edge of SCK. The timing for the second through the last bits is identical to the timing shown in figure 7.

Serial Output, Case 3

Figure 9 shows output timing when $\overline{SOEN}$ is active before SORQ is high. If $\overline{SOEN}$ is held active before SORQ is high, data will be shifted out whenever it becomes available in the serial output register (assuming previous data is already shifted out). In this case, SORQ will rise t_{DRQ} after a rising edge of SCK. SO will become active (but not valid yet) t_{DZRQ} after the same rising edge of SCK. The first valid SO bit occurs t_{DCK} after the next falling edge of SCK for use by an external device at the subsequent rising edge of SCK.

Subsequent bits will be shifted out t_{DCK} after subsequent falling edges of SCK, for use at subsequent rising edges of SCK. The last bit to be shifted out will also

Figure 7. Serial Output Case 1: $\overline{SOEN}$ Asserted in Response to SORQ When SCK Is Low

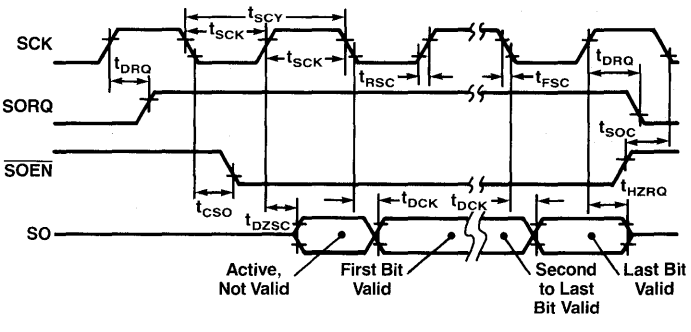


Figure 8. Serial Output Case 2: $\overline{SOEN}$ Asserted in Response to SORQ When SCK Is High

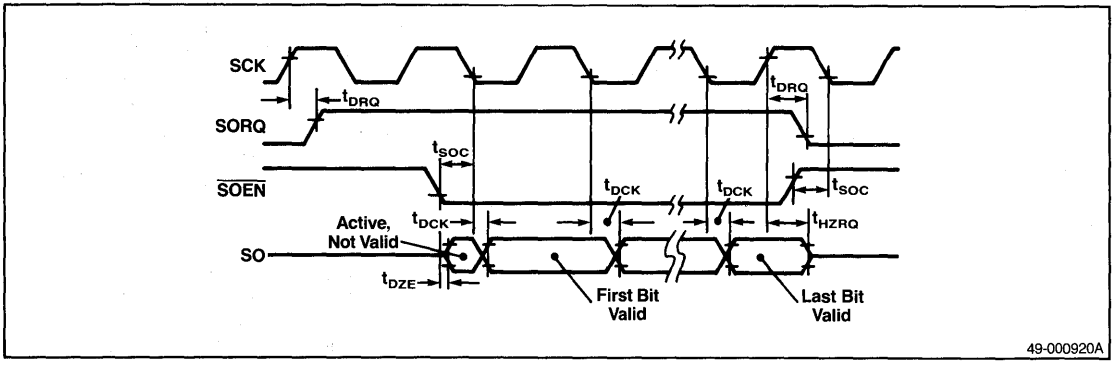
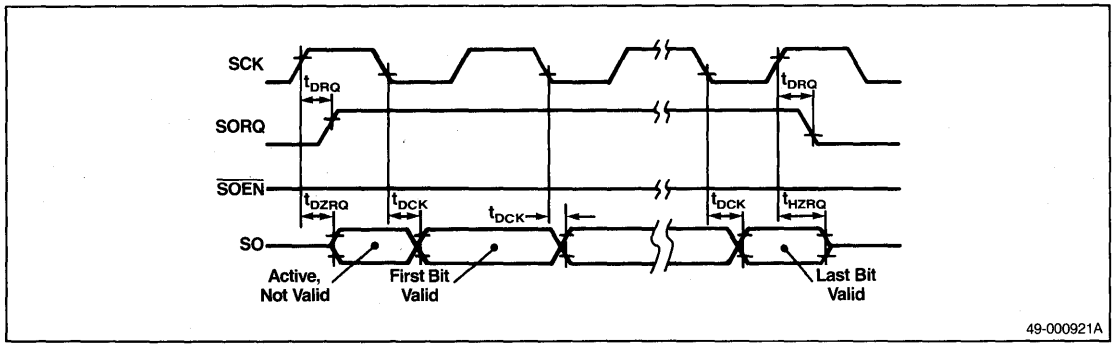


Figure 9. Serial Output Case 3: $\overline{SOEN}$ Active Before SORQ Is High



follow this pattern, and will be held valid t_{HZRQ} after the corresponding rising edge of SCK at which it is to be used. SORQ will be held t_{DRQ} after this same rising edge of SCK, and then removed.

Serial Output, Case 4A

Avoid releasing $\overline{SOEN}$ in the middle of a transfer (that is, before the last bit is shifted out), since this will stop the output shift operation, and, when $\overline{SOEN}$ is again asserted, the remainder of the transfer will be shifted out before the next transfer can begin. The next transfer will begin immediately without any indication of the byte/word boundary. If $\overline{SOEN}$ is released while SCK is high, as shown in figure 10, at least t_{SOC} before the falling edge of SCK, then SO will go inactive t_{HZE} after $\overline{SOEN}$ is released (which may be before or after the falling edge of SCK).

Serial Output, Case 4B

If $\overline{SOEN}$ is released while SCK is low, as in figure 11, at least t_{CSO} after the falling edge of SCK, then the next bit will be shifted out t_{DCK} after the falling edge of SCK, for use at the subsequent rising edge of SCK. SO will then go inactive t_{HZSC} after this rising edge of SCK.

Note: For all its uses, $\overline{SOEN}$ must not change state within t_{SOC} before or t_{CSO} after the falling edge of SCK; otherwise, the results will be indeterminate.

Serial Input

Serial input timing, shown in figure 12, is much simpler than serial output timing. Data bits are shifted in on the rising edge of SCK if $\overline{SIEN}$ is asserted. Both $\overline{SIEN}$ and SI must be stable at least t_{DC} before and t_{CD} after the rising edge of SCK; otherwise the results will be indeterminate.

Figure 10. Serial Output Case 4A: If $\overline{SOEN}$ Is Released in the Middle of a Transfer During SCK High

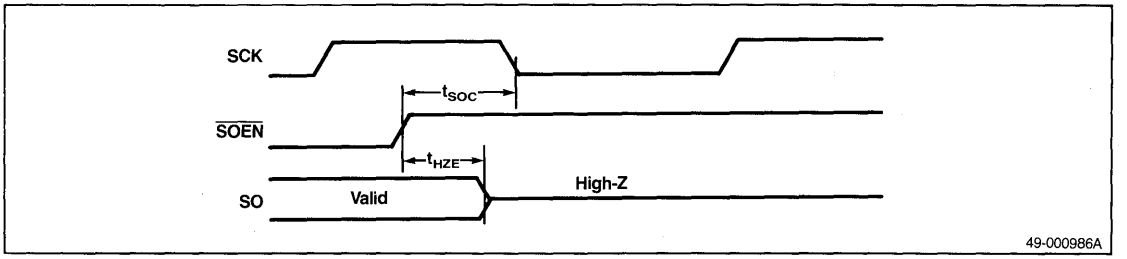


Figure 11. Serial Output Case 4B: If $\overline{SOEN}$ Is Released in the Middle of a Transfer During SCK Low

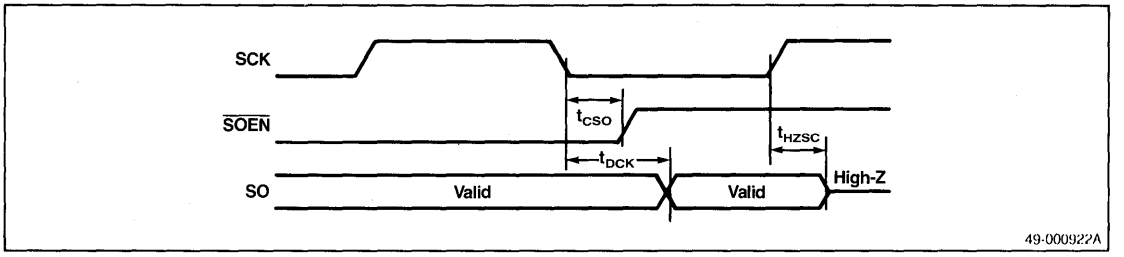
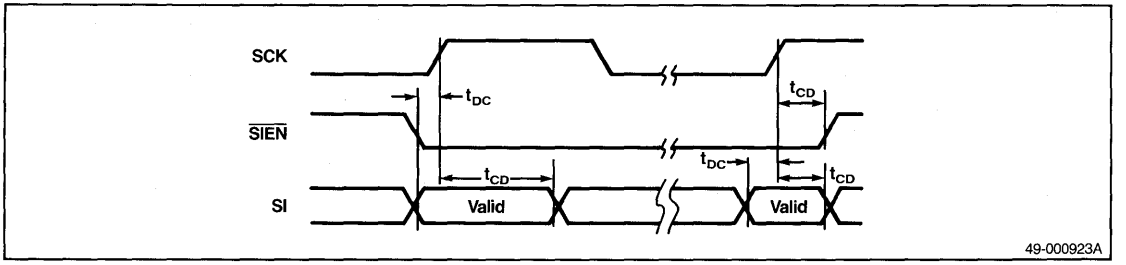


Figure 12. Serial Input



Serial Timing Example

Figure 13 shows serial timing of cascaded SPIs with a common SCK. SO from the first SPI equals SI of the second, and the first SPI's SORQ inverts to become $\overline{SIEN}$ of the second. $\overline{SOEN}$ of the first SPI is always asserted.

When cascading two SPIs in the described configuration, most of the timing involved is directly copied from the case of serial output with $\overline{SOEN}$ always enabled (figure 13). It must be shown that the results will be suitable for the serial input timing of the second SPI.

(1) SORQ (1) rises t_{DRQ} after a rising edge of SCK, and it is inverted (inverter has t_{PHL} delay time) to become $\overline{SIEN}$ (2), which must be stable t_{DC} before the next rising edge of SCK. It also must not change until t_{CD} after this first rising edge of SCK, as shown by case 2 in figure 8.

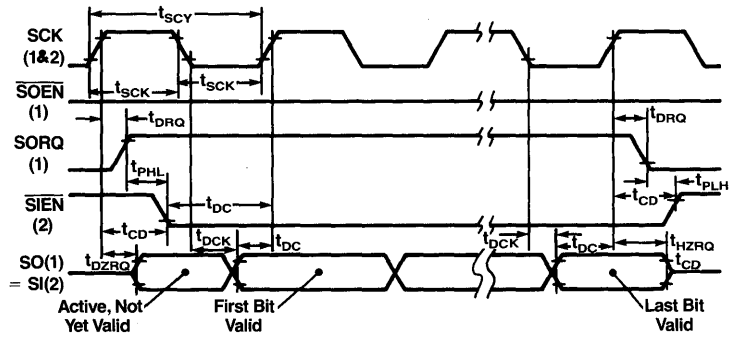
$$t_{DRQ}(max) + t_{PHL} + t_{DC}(min) \leq t_{SCY}(min)$$
$$t_{PHL}(max) \leq t_{SCY}(min) - t_{DC}(min) - t_{DRQ}(max)$$
$$\leq 480 - 55 - 150$$
$$\leq 275 \text{ ns (readily achieved by 74LS14, for example)}$$

(2) SORQ (1) is released t_{DRQ} after the last useful rising edge of SCK, and is inverted (inverter has t_{PHL} delay time) to become $\overline{SIEN}$ (2), which must remain stable t_{CD} after the rising edge of SCK.

$$t_{DRQ}(min) + t_{PLH}(min) \geq t_{CD}(min)$$
$$t_{PLH}(min) \geq t_{CD}(min) - t_{DRQ}(min)$$
$$\geq 30 - 30$$
$$\geq 0 \text{ (no problem, assuming causality)}$$

Note: This also shows $t_{PHL}(min) \geq 0$ for the rising edge of SORQ.

Figure 13. Serial Timing Example



(3) SO (1) is valid t_{DCK} after a falling edge of SCK; since it becomes SI (2), it must be valid t_{DC} before the next rising edge of SCK.

$$t_{DCK}(max) + t_{DC}(min) \leq t_{SCK}(min)$$
$$150 + 55 \leq 230$$
$$205 \leq 230 \text{ (this condition is satisfied)}$$

(4) SO (1) remains valid t_{HZRQ} after the last useful rising edge of SCK; since it becomes SI (2), it must remain valid t_{CD} after this rising edge of SCK.

$$t_{HZRQ}(min) \geq t_{CD}(min)$$
$$70 \geq 30 \text{ (this condition is satisfied)}$$

Note: The above calculations may need to be adjusted for rise and fall times, since t_{SCY} and t_{SCK} are measured for midpoints of wave slopes.

μPD77P20 UV Erasable EPROM Version

Function

The 77P20 operates from a single +5-volt power supply and can accordingly be used in any 77C20A/7720A masked ROM application.

Use of Evakit-7720

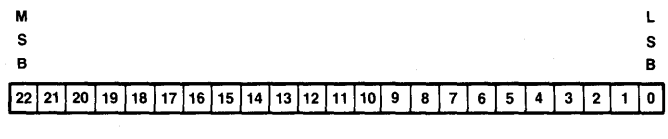
The following sections describe electrical conditions that are required for programming the 77P20. However, the Evakit-7720, NEC's hardware emulator development tool for the 77C20A/7720A/77P20, meets the electrical and timing specifications presented below. When the Evakit-7720 is used for programming 77P20, all data transfers and formatting are handled automatically by Evakit's monitor program. Please refer to the Evakit-7720(B) User's Manual for programing procedures.

The information presented below in the sections on Configuration, Operation, and Programming (and the various subsections) is required only for users who do NOT intend to use an Evakit to program the 77P20.

Configuration

Data transfer for programming and reading the internal ROM is partitioned into three bytes for each 23-bit wide instruction location and into two bytes for each 13-bit wide data location. Partitioning of data transfer into and out of the data port is shown in figure 14.

Figure 14. Instruction ROM Format



Instruction ROM

The instruction ROM data is transferred through the data port as a high byte, middle byte, and low byte as shown in figure 15. Bit 7 of the middle byte should be assigned a value of zero. Data is presented to the data port in a bit-reversed format. The LSB through the MSB of an instruction ROM byte is applied to the MSB through the LSB of the data port, respectively.

Data ROM

Figure 16 shows the data ROM format. The data ROM data is transferred through the data port as a low byte and a high byte as shown in figure 17. Bits 0, 1, and 2 of the low byte should be assigned a value of zero. Data is presented to the data port in corresponding order. The MSB through the LSB of a data ROM byte is applied to the MSB through the LSB of the data port, respectively.

Initially and after each erasure, all bits of the 77P20 are in the zero state.

Figure 15. Transfer of Instruction ROM Data

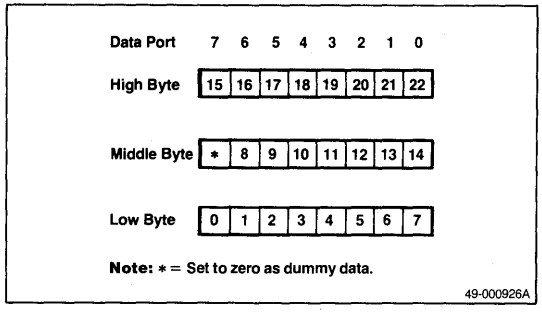


Figure 16. Data ROM Format

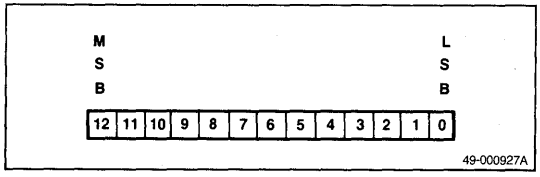
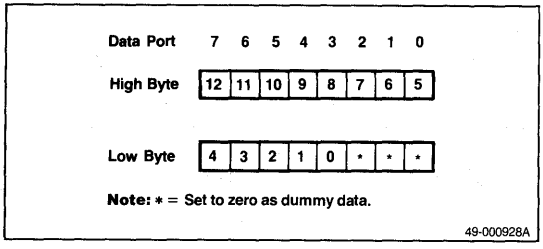


Figure 17. Transfer of Data ROM Data



Operating Modes

In order to read or write the instruction or data ROMs, the mode of operation of the 77P20 must be initially set. At the RST trailing edge, the $\overline{RD}$, $\overline{WR}$, and $\overline{CS}$ should be logical zero and the $\overline{DACK}$, A_0, and SI signals should be set to determine the mode of operation accordingly, as set out in table 14.

Table 14. μPD77P20 Operation Mode

$\overline{DACK}$	A_0	SI	
0	0	0	Write mode instruction and data ROM
0	0	1	Read the instruction ROM
0	1	0	Read the data ROM

Once set, the 77P20 will remain in the selected mode. A reset is required to transfer to another mode.

Write Mode

The individual instruction ROM and data ROM bytes are specified by control signals $\overline{RD}$, A_0, SI, and INT as set out in table 15. Before writing the EPROM location, the bytes should be loaded accordingly.

Table 15. Write Mode Specification of ROM Bytes

$\overline{RD}$	A_0	SI	INT	
1	0	0	1	Write instruction byte, high
1	0	1	0	Write instruction byte, middle
1	0	1	1	Write instruction byte, low
1	1	0	0	Write data byte, low
1	1	0	1	Write data byte, high

Read Mode

The instruction ROM and data ROM bytes are specified by the control signals $\overline{RD}$, A_0, SI, and INT as set out in table 16. Reading is accomplished by setting the control signals accordingly.

Table 16. Read Mode Specification of ROM Bytes

$\overline{RD}$	A_0	SI	INT	
0	0	0	1	Read instruction byte, high
0	0	1	0	Read instruction byte, middle
0	0	1	1	Read instruction byte, low
1	0	0	0	Read data byte, high and low

The instruction ROM and data ROM are addressed by the 9-bit program counter and the 9-bit ROM pointer respectively. The PC is reset to 000H and is automatically incremented to the end address 1FFH. The RP is reset to 1FFH and is automaticaly decremented to 000H.

Erasing

Programming can only occur when all data bits are in an erased or low (0) level state. Erase 77P20 programmed data by exposing it to light with wavelengths shorter than approximately 4,000 angstroms. Note that constant exposure to direct sunlight or room level fluorescent lighting could erase the 77P20. Consequently, if the 77P20 will be exposed to these types of lighting conditions for long periods of time, mask its window to prevent unintentional erasure.

The recommended erasure procedure for the 77P20 is exposure to ultraviolet light with wavelength of 2537 angstroms. The integrated dose (i.e., UV intensity x exposure time) for erasure should not be less than 15 W•s/cm^2. The erasure time is approximately 20 minutes using an ultraviolet lamp with a power rating of 12,000 μW/cm^2.

During erasure, place the 77P20 within 1 inch of the lamp tubes. If the lamp tubes have filters, remove the filters before erasure.

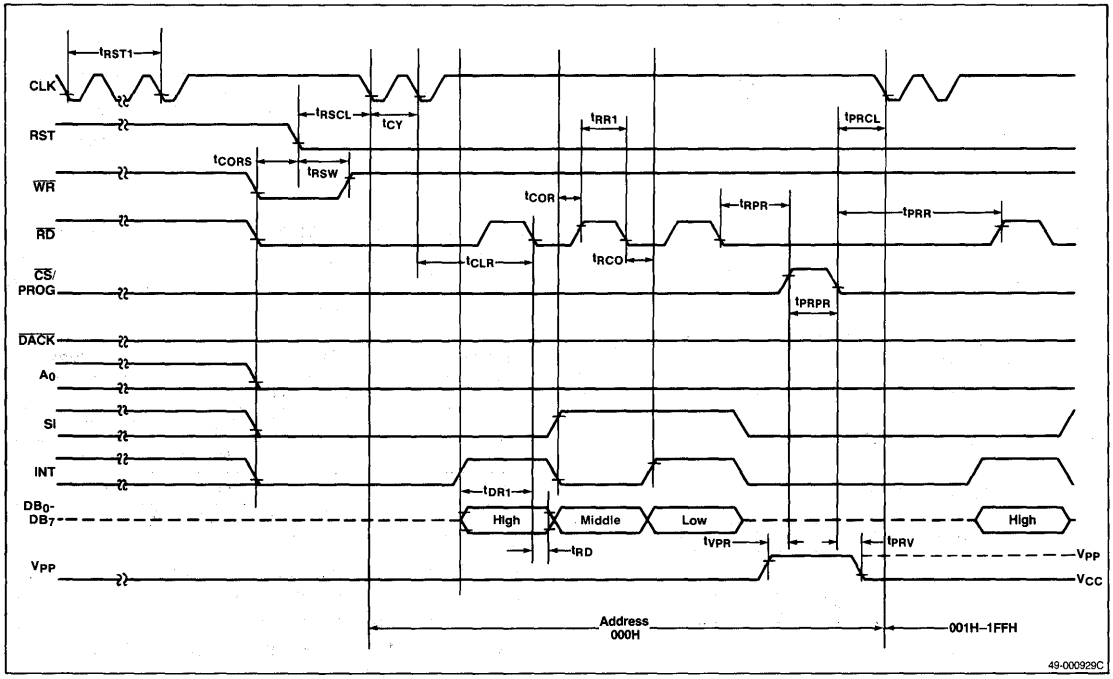
Programming

Programming of the 77P20 is achieved with a single 50-ms TTL pulse. Total programming time for the 11,776 bits of instruction EPROM and also for the 6630 bits of data EPROM is 26 seconds. Data is entered by programming a high (1) level in the chosen bit locations. Both instruction ROM and data ROM should be programmed since they cannot be erased independently. Both instruction ROM and data ROM programming modes are entered in the same manner.

The device must be reset initially before it can be placed into the programming mode. After reset, the $\overline{WR}$ signal and all other inputs ($\overline{RD}$, $\overline{CS}$/PROG, $\overline{DACK}$, A_0, SI, and INT) should be a TTL low (0) signal t_{RS} prior to the falling edge of RST. $\overline{WR}$ is then held for t_{RH} before being set to a TTL high (1) level signal. The device is now in a programming mode and will stay in this mode, allowing ROM locations to be sequentially programmed.

Programming Mode—Instruction ROM. Instruction ROM locations are sequentially programmed from address 000H to address 1FFH. The location address is incremented by the application of CLK for a duration of t_{CY}. Data bytes for each location as specified by control signals $\overline{RD}$, A_0, SI, and INT (table 15) are clocked into the device by the falling edge of $\overline{RD}$.

Figure 18. Programming Mode of Instruction ROM



After the three bytes have been loaded into the device, V_{PP} is raised to 21 V ±0.5 V, t_{VS} prior to $\overline{CS}$/PROG transitioning to a TTL high (1) level signal. V_{PP} is held for the duration of t_{PRPR} plus t_{PRV} before returning to the V_{CC} level. After t_{PRCL} the instruction ROM address can be incremented to program the next location. Figure 18 shows the programming mode of instruction ROM timing.

Programming Mode—Data ROM. Data ROM locations are sequentially programmed from address 1FFH to address 000H. The location address is decremented by the application of CLK for t_{CY}. The data bytes for each location as specified by control signals $\overline{RD}$, A_0, SI, and $\overline{INT}$ are clocked into the device by the falling edge of $\overline{RD}$.

After the two bytes have been loaded into the device, V_{PP} is raised to 21 V, ±0.5 V t_{VPR} prior to $\overline{CS}$/PROG transitioning to a TTL high (1) level signal. V_{PP} is held for the duration of t_{PRPR} plus t_{PRV} before returning to the V_{CC} level. After t_{PRCL} the data ROM address can be decremented to program the next location. Figure 19 shows programming mode of data ROM timing.

Read Mode. A read should be performed to verify that the data was programmed correctly. Prior to entering read mode the device must be reset.

Read Mode—Instruction ROM. This mode is entered by holding the $\overline{WR}$ signal at a TTL low (0) level with the SI signal at a TTL high (1) level and all other specified inputs ($\overline{RD}$, $\overline{CS}$/PROG, $\overline{DACK}$, A_0, INT) at TTL low (0) levels for t_{CORS} prior to the falling edge of RST. $\overline{WR}$ is then held for t_{RSW} before being set to a TTL high (1) level. The device is now in the instruction ROM read mode and will stay in this mode until reset.

Instruction ROM locations are sequentially read from address 000H through 1FFH. Application of CLK for t_{CY} will increment the location address. The three data bytes will be read as specified by the control signals $\overline{RD}$, A_0, SI and INT (table 16). Figure 20 shows read mode of instruction ROM timing.

Figure 19. Programming Mode of Data ROM

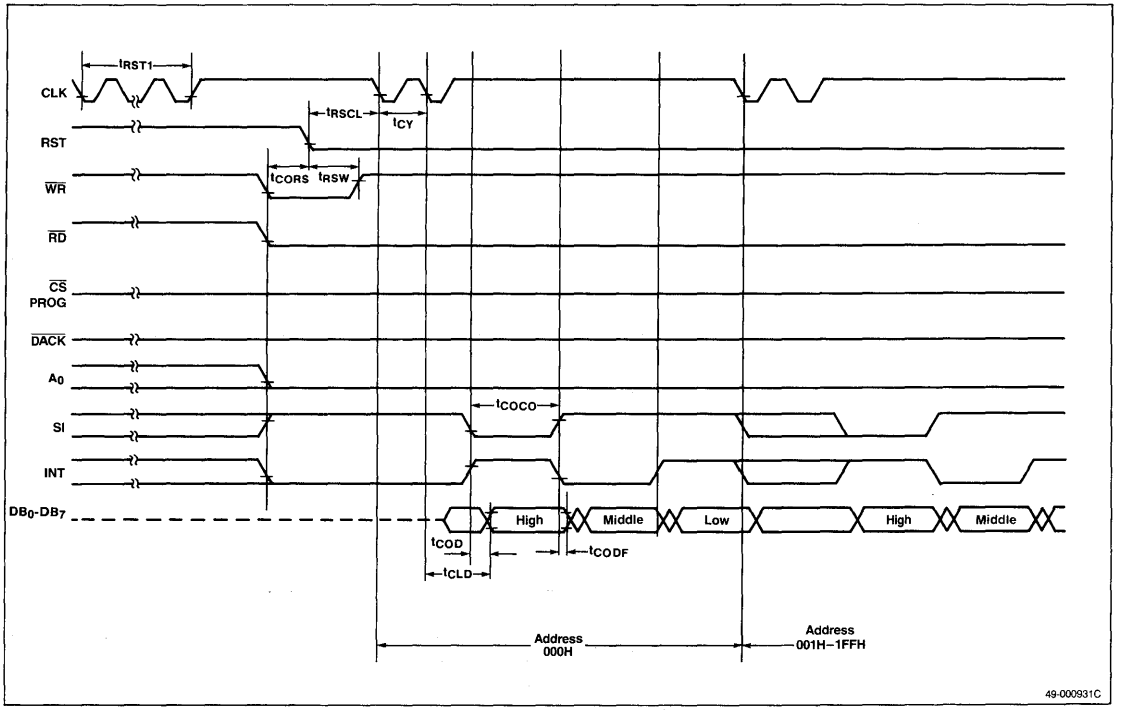


Figure 20. Read Mode of Instruction ROM

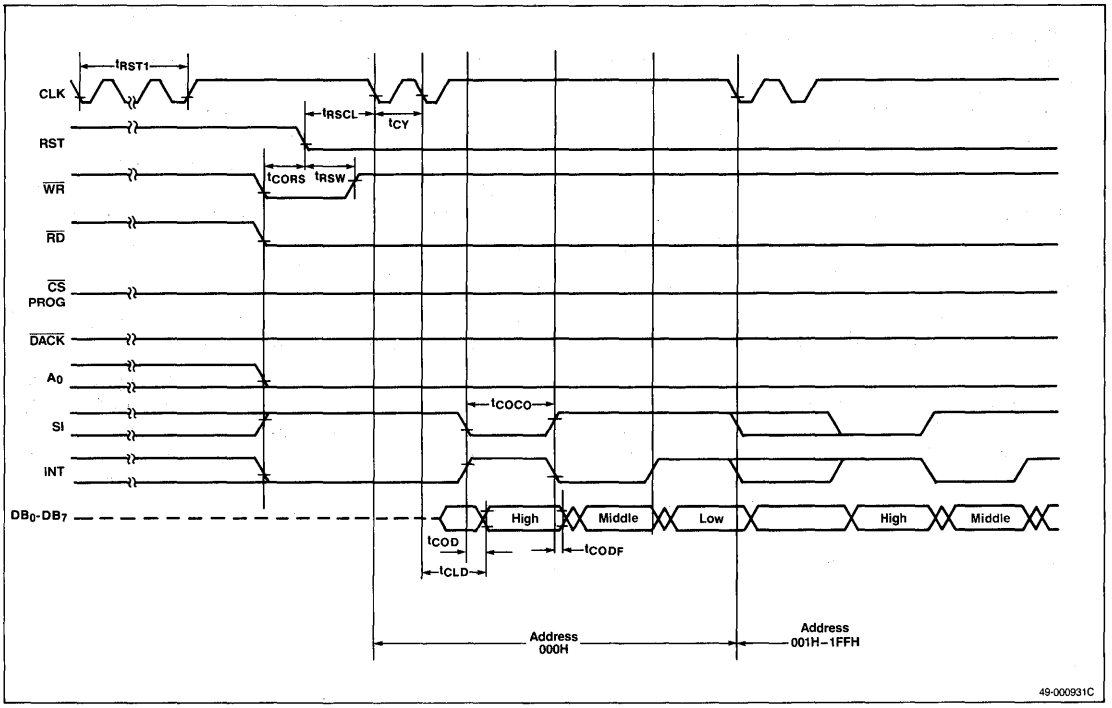
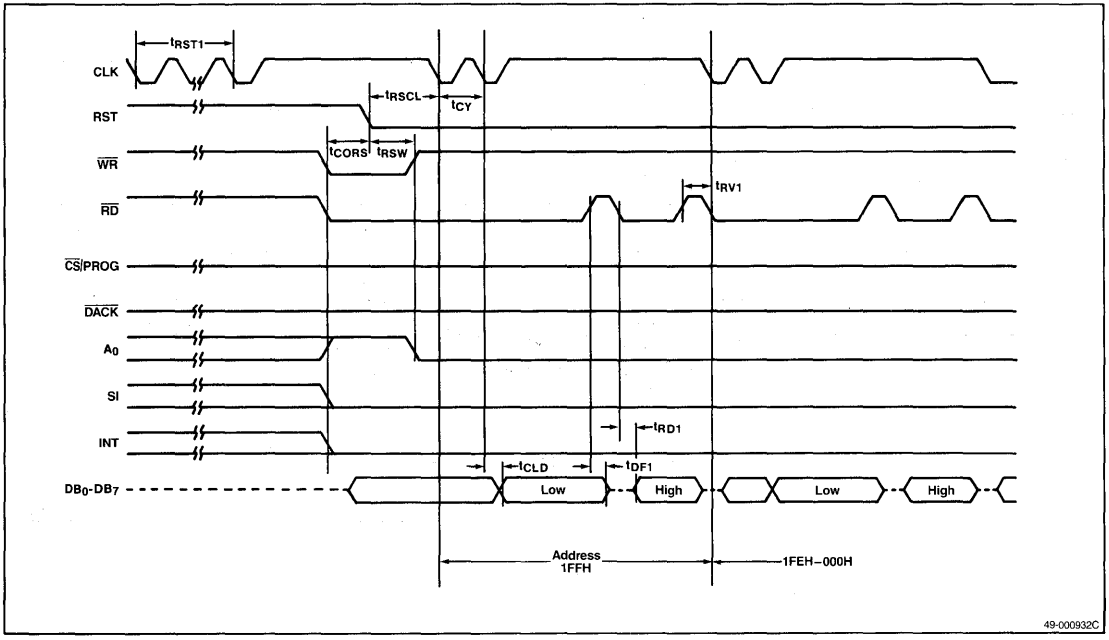


Figure 21. Read Mode of Data ROM



Read Mode—Data ROM. Figure 21 shows read mode of data ROM timing. This mode is entered by holding the $\overline{WR}$ signal at a TTL low (0) level with the A_0 signal at a TTL high (1) level and all other specified inputs ($\overline{RD}$, $\overline{CS}$/PROG, $\overline{DACK}$, SI, INT) at TTL low (0) levels for t_{CORS} prior to the falling edge of RST. $\overline{WR}$ and A_0 are then held for t_{RSW} prior to the falling edge of RST. $\overline{WR}$ and A_0 are then held for t_{RSW} before being set to a TTL high (1) level and TTL low (0) level, respectively. The device is now in the data ROM read mode and will stay in this mode until it is reset.

Data ROM locations are sequentially read from address 1FFH through 000H. Application of CLK for t_{CY} will decrement the location address. After the address has been decremented, the low byte of the current location will be available at the data port subsequent to a t_{CLD} delay. Application of $\overline{RD}$ will present the high byte t_{RD1} from the falling edge of the $\overline{RD}$ pulse. $\overline{RD}$ is then applied for t_{VR} to complete reading of the current location.

Read Operation, AC Characteristics

$T_A = 25°C \pm 5°C$; $V_{CC} = 5\ V \pm 5\%$; $V_{PP} = V_{CC} + 0.25\ V$ max
$V_{PP} = V_{CC} - 0.85\ V$ min

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
Data access time from CLK	t_{CLD}			1	μs	
Data delay time from SI, INT↑	t_{COD}			1	μs	
Data float time from SI, INT↑	t_{CODF}	0			ns	
SI, INT pulse width	t_{COCO}	1			μs	
$\overline{RD}$ recovery time	t_{RV1}	500			ns	
Data access time from $\overline{RD}$↓	t_{RD1}			150	ns	
Data float time from $\overline{RD}$↑	t_{DF1}	10			ns	

Programming Operation, AC Characteristics

$T_A = 25°C \pm 5°C$; $V_{CC} = 5\ V \pm 5\%$; $V_{PP} = 21\ V \pm 0.5\ V$

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
CLK cycle time	t_{CY}	240			ns	
CLK setup time to $\overline{RD}$↓	t_{CLR}	2			μs	
CLK hold time from RST↓	t_{RSCL}	6			μs	
CLK hold time from PROG↓	t_{PRCL}	200			ns	
Control signal set-up time to RST↓	t_{CORS}	1			μs	
$\overline{WR}$ hold time from RST↓	t_{RSW}	6			μs	
Data set-up time from $\overline{RD}$↓	t_{DRIO}	1			μs	
Data hold time from $\overline{RD}$↓	t_{RD}	100			ns	
$\overline{RD}$ pulse width	t_{RR1}	1			μs	
SI, INT set-up time from $\overline{RD}$↑	t_{COR}	100			ns	
SI, INT hold time from $\overline{RD}$↓	t_{RCO}	100			ns	
$\overline{RD}$ set-up time to PROG↑	t_{RPR}	100			ns	
$\overline{RD}$ hold time from PROG↓	t_{PRR}	2			μs	
V_{PP} set-up time to PROG↑	t_{VPR}	2			μs	
V_{PP} hold time from PROG↓	t_{PRV}	2			μs	
RST pulse width	t_{RST1}	4			t_{CY}	
RST setup time	t_{RS}	1			μs	
PROG pulse width	t_{PRPR}	45	50	55	ms	

Operation Mode

The 77P20 may be utilized in an operation mode after the instruction ROM and data ROM have been programmed. Since it was first introduced in 1982, the 77P20 has undergone several mask revisions to improve manufacturability and/or function. And since the purpose of the 77P20 is to run any program that may be programmed in the masked ROM 77C20A/7720A, it is important to know how to determine the step level, and the differences between them.

Step Level

The markings on the μPD77P20 package consist of three lines, as follows:

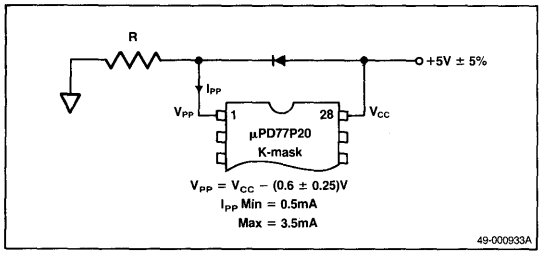
NEC JAPAN ← Manufacturer
D77P20D ← Part number
nnnnXnnnn ← Date code

In the date code, "X" identifies the step level of the part. Parts marked with step level K, E, or P should not be used for final system test by customers who are planning to submit code for the masked ROM 77C20A/7720A.

On all other 77P20 stepping versions, a slight functional change was made, and the change is incorporated in the 77C20A/7720A. The change allows the serial clock (SCK) to run asynchronously with CLK. Specified versions of 77P20 (i.e. K, E, P) and all Evakit-7720s and Evakit-7720Bs (Evaluation Systems for 77C20A/7720A/77P20) require that SCK run synchronously with CLK.

Because this functional change results in a slight change in internal serial timing, it is mandatory that code to be submitted for 77C20A/7720A be verified in customer's system using versions of 77P20 other than those listed above (i.e. K, E, P).

Figure 22. V_{PP} Circuitry for K Mask Version



Pin 1 Connection

The K mask version requires that the programming voltage V_{PP} be supplied in a different manner than for all later versions, as shown in figure 22. A silicon junction diode of 0.6 V forward voltage (V_F) should be used. R should be 800 to 1800 Ω to satisfy the V_{PP} and I_{PP} requirements.

In all mask versions other than K, pin 1 must be connected directly to V_{CC}.

Development Tools

For software development, assembly into object code, and debugging, an absolute assembler and simulator are available. The ASM77 Absolute Assembler and SIM77 Simulator for analyzing development code and I/O timing characteristics are available for systems supporting CP/M®, and CP/M-86®, ISIS-II®, or MS-DOS® operating systems. Additionally, the ASM77 Absolute Assembler is offered in Fortran source code for mini and main frame computer systems.

Once software development is complete, the code can be completely evaluated and debugged in hardware with the Evakit-7720 Evaluation System. The Evakit provides true in-circuit real-time emulation of the SPI for debugging and demonstrating your final system design. Code may be down-loaded to the Evakit from a development system via an RS232 port using the EVA communications program. This program is available in executable form for ISIS-II systems and many CP/M, CP/M-86, and MS-DOS systems. The EVA communications source code is also available for adapting the program to other systems.

The Evakit also serves to program the 77P20, a full-speed EPROM version of the SPI. A demonstration mask ROM chip, containing some common digital filtering routines, including N-stage IIR (biquadratic) and FIR (transversal filters), is available to test hardware interfaces to the SPI.

CP/M and CP/M-86 are registered trademarks of Digital Research Corp.

ISIS-II is a registered trademark of Intel Corp.

MS-DOS is a registered trademark of Microsoft Corp.

Further operational details of the SPI can be found in the μPD77C20A/7720A/77P20 Signal Processing Interface Design Manual. Operation of the SPI development tools is described in the Absolute Assembler User Manual, the Simulator Operating Manual, and Evakit-7720 User's Manual.

System Configuration

Figures 23, 24, 25, and 26 show typical system applications for the 77C20A/7720A/77P20 SPI.

Figure 23. Spectrum Analysis System

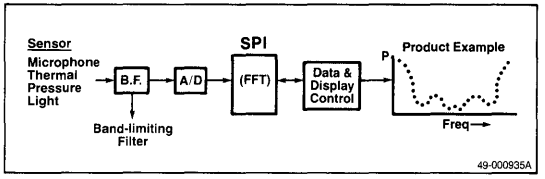


Figure 24. Analog-to-Analog Digital Processing System Using a Single SPI

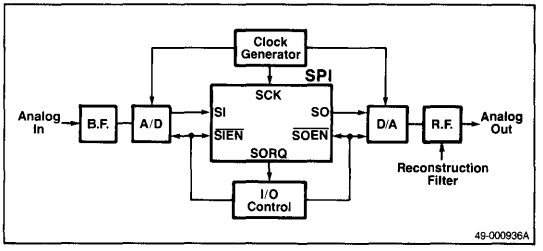


Figure 25. Signal Processing System Using Cascaded SPIs and Serial Communication

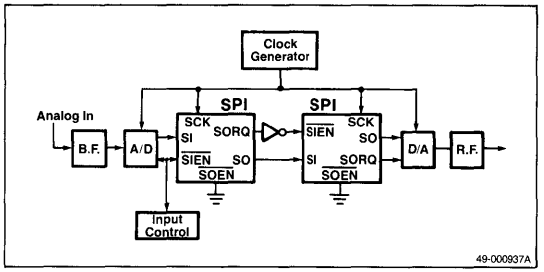
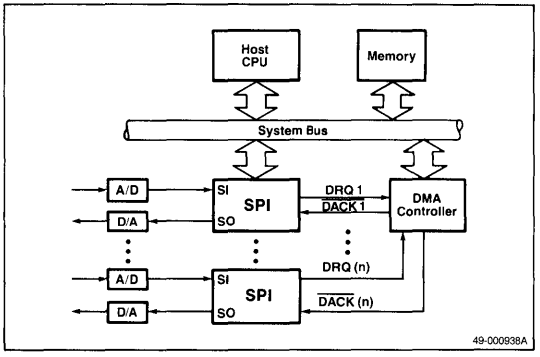


Figure 26. Signal Processing System Using SPIs as a Complex Computer Peripheral



2

Description

The μPD77C25 and μPD77P25 signal processing interface (SPI) chips are significant upgrades to the μPD7720 — the original member of NEC's family of digital signal processors. Designated the "SPI PLUS" (SPI+), these chips execute instructions twice as fast as the 77C20A/7720A. Additional instructions allow the SPI+ to execute common digital filter routines more efficiently and hence at more than twice the speed of a 7720 implementation.

In addition to doubled execution speed, the SPI+ has four times the instruction ROM space and twice the data ROM and RAM space of the 7720. Real savings are now possible, especially where one 77C25 can do the work of and replace two or more 7720s.

The external clock frequency (8.3 MHz maximum) remains the same as for 77C20A/7720A while the internal instruction execution speed is doubled. For most applications, the 77C25A/77P25 is plug-in compatible with the 77C20A/7720A/77P20.

The feature that distinguishes digital signal processing (DSP) chips from general-purpose microcomputers is the on-chip multiplier, necessary for high-speed signal processing algorithms. The SPI+ multiplier is very sophisticated, especially for a low-cost DSP chip, since both multiplier inputs can be loaded simultaneously from two separate memory areas. These loading operations are only two of nine operations that can occur during one 122-ns instruction cycle. (On competitive DSP chips, such operations require separate instructions). For a typical DSP filter application involving many successive multiplications, the SPI+ provides a new multiplication product for addition to a sum of products every 122 nanoseconds. Additionally, during the same instruction, memory data pointers are manipulated, and even a return from subroutine may be executed.

The μPD77C25 is the mask ROM version and μPD77P25D is the UVEPROM version. The μPD77P25C/77P25L are the one time programmable (OTP) EPROM device versions. All versions are CMOS and functionally identical.

Table 1 compares SPI+ chips with SPI chips.

Features

- ☐ Low-power CMOS: 25 mA typical current use (77C25)
- ☐ Fast instruction execution: 122 ns with 8.192 MHz clock
- ☐ All instructions execute in one instruction cycle
- ☐ Drop-in compatible with μPD77C20A/7720A/77P20
- ☐ 16-bit data word
- ☐ Multi-operation instructions for fast program execution: any part, any combination, or all of the following operations may constitute one instruction that executes in 122 ns.
  - — Load one multiplier input
  - — Load the other multiplier input
  - — Multiply (automatic)
  - — Load product to output registers (automatic)
  - — Add product to accumulator
  - — Move RAM column data pointer
  - — Move RAM row pointer
  - — Move data ROM pointer
  - — Return from subroutine
- ☐ Modified Harvard architecture with three separate memory areas
  - — Program ROM (2048 x 24 bits)
  - — Data ROM (1024 x 16 bits)
  - — Data RAM (256 x 16 bits)
- ☐ 16 x 16-bit multiplier; 31-bit product with every instruction
- ☐ Dual 16-bit accumulators
- ☐ External maskable interrupt
- ☐ Four-level stack for subroutines and/or interrupt
- ☐ Multiple I/O capabilities
  - — Serial: 8 or 16-bit (244 ns/bit)
  - — Parallel: 8 or 16-bit
  - — DMA
- ☐ Compatible with most μPs, including:
  - — μPD8080
  - — μPD8085
  - — μPD8086/88
  - — μPD780 (Z80®)
  - — μPD78xx family
- ☐ One-time programmable (OTP) version available in a 28-pin plastic DIP or a 44-pin PLCC
- ☐ Single +5-volt power supply

Z80 is a registered trademark of Zilog Corporation.

Applications

- ☐ Portable telecommunications equipment
- ☐ Digital filtering
- ☐ High-speed data modems
- ☐ Fast Fourier transforms (FFT)
- ☐ Speech synthesis and analysis
- ☐ Dual-tone multifrequency (DTMF) transmitters/receivers
- ☐ Equalizers
- ☐ Adaptive control
- ☐ Numerical processing

Performance Benchmarks

- ☐ Second-order digital filter (biquad): 1.1 μs
- ☐ Sin/cos of angles: 2.58 μs
- ☐ μ/A law to linear conversion: 0.24 μs
- ☐ FFT
  - — 32-point complex: 0.35 ms
  - — 64-point complex: 0.8 ms

Ordering Information

Part Number	Package Type	ROM	Normal Temperature Range
μPD77C25C	28-pin plastic DIP	Mask	−40 to +85°C
μPD77C25L	44-pin PLCC		
μPD77P25C	28-pin plastic DIP	OTP	−10 to +70°C
μPD77P25D	28-pin ceramic DIP	UV EPROM	
μPD77P25L	44-pin PLCC	OTP	

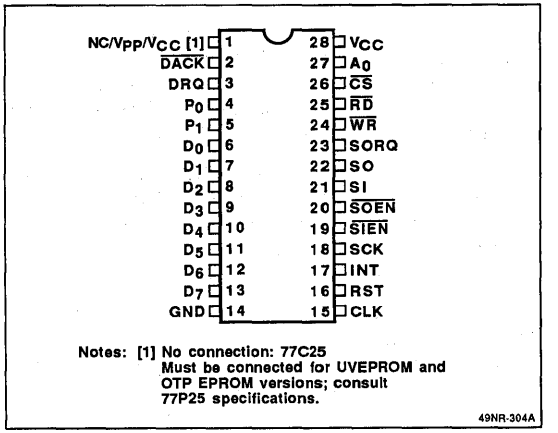
Table 1. Comparison of SPI+ and SPI Chips

	SPI+ 77C25/77P25	SPI 77C20A/77P20
Technology	CMOS/CMOS	CMOS/NMOS
Instruction cycle	122 ns	244 ns
Instruction ROM	2048 x 24 bits	512 x 23 bits
Data ROM	256 x 16 bits	510 x 13 bits
Data RAM	256 x 16 bits	128 x 16 bits
Fixed-point multiplier	16 bits x 16 bits → 31 bits	16 bits x 16 bits → 31 bits
ALU	16 bit fixed point	16 bit fixed point
Accumulator	2 x 16 bits	2 x 16 bits
Host CPU interface	8-bit bus	8-bit bus
Serial interface	One input and one output	One input and one output
	4 MHz	2 MHz
Temporary registers	two	one
Additional instructions	JDPLN0	—
	JDPLNF	—
	Modification of RAM column data pointer M8-MF	—
DMA mode	Fully implemented	Partially implemented
Package	28-pin DIP	28-pin DIP
	44-pin PLCC	44-pin PLCC
Power supply	5 V	5 V
Power consumption	50 mA (max) @ 8.192 MHz	40 mA (max) @ 8.192 MHz
Power saving mode (when idle)	Yes	No

Since the 77C25 executes an instruction in one external clock cycle (versus two cycles of the same 8.192-MHz clock for 77C20A), the 77C25 may be substituted for a 77C20A (or 7720A or 77P20) in a circuit without modification of that circuit. Hardware/software that implements data transfers—both serial and parallel—between the SPI+ and other devices in an existing 7720 design should use the handshake protocol described more fully in the 77C25 User's Manual.

Pin Configurations

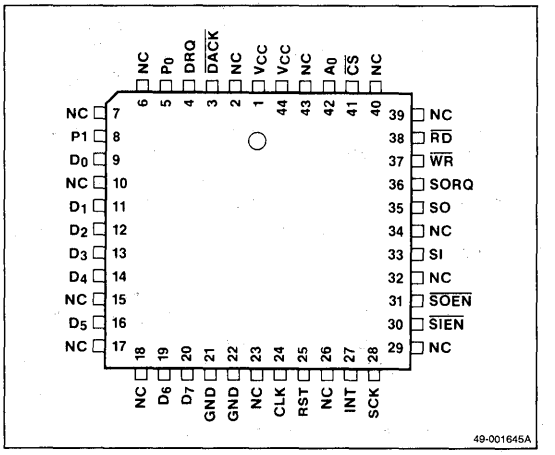
28-Pin DIP



Notes: [1] No connection: 77C25
Must be connected for UVEPROM and OTP EPROM versions; consult 77P25 specifications.

49NR-304A

44-Pin PLCC



49-001645A

Pin Identification

Symbol	Function
A_0	Status/data register select input
CLK	Single-phase master clock input
$\overline{CS}$	Chip select input
D_0-D_7	Three-state I/O data bus
$\overline{DACK}$	DMA request acknowledge input
DRQ	DMA request output
INT	Interrupt input
P_0, P_1	General-purpose output control lines
$\overline{RD}$	Read control signal input
RST	Reset input
SCK	Serial data I/O clock input
SI	Serial data input
$\overline{SIEN}$	Serial input enable input
SO	Three-state serial data output
$\overline{SOEN}$	Serial output enable input
SORQ	Serial data output request
$\overline{WR}$	Write control signal input
GND	Ground
V_{CC}	+5 V power supply
NC/V_{PP}/V_{CC}	77C25: no connection 77P25: +12.5 V programming 77P25: +5 V for normal operation

Pin Functions

A_0 [Status Data Register Select]

This input selects data register for read/write (low) or status register for read (high).

CLK

This is the single-phase master clock input.

$\overline{CS}$ [Chip Select]

This input enables data transfer through the data port with $\overline{RD}$ or $\overline{WR}$.

D_0-D_7 [Data Bus]

This three-state I/O data bus transfers data between the data register or status register and the external data bus.

$\overline{DACK}$ [DMA Request Acknowledge]

This input indicates to the SPI+ that the data bus is ready for a DMA transfer ($\overline{DACK}$ = CS and $A_0 = 0$).

DRQ [DMA Request]

This output signals that the SPI+ is requesting a data transfer on the data bus.

INT [Interrupt]

A low-to-high transition on this pin executes a call instruction to location 100H if interrupts were previously enabled.

P_0, P_1

These pins are general-purpose output control lines.

$\overline{RD}$ [Read Control Signal]

This input latches data from the data or status register to the data port where it is read by an external device.

RST [Reset]

This input initializes the SPI+ internal logic and sets the PC to 0.

SCK [Serial Data I/O Clock]

When this input is high, a serial data bit is transferred.

SI [Serial Data Input]

This pin inputs 8- or 16-bit serial data words from an external device such as an A/D converter.

$\overline{SIEN}$ [Serial Input Enable]

This input enables the shift clock to the serial input register.

SO [Serial Data Output]

This three-state port outputs 8- or 16-bit data words to an external device such as a D/A converter.

$\overline{SOEN}$ [Serial Output Enable]

This input enables the shift clock to the serial output register.

SORQ [Serial Data Output Request]

This output specifies to an external device that the serial data register has been loaded and is ready for output. SORQ is reset when the entire 8- or 16-bit word has been transferred.

$\overline{WR}$ [Write Control Signal]

This input writes data from the data port into the data register.

GND

This is the connection to ground.

V_{CC} [Power Supply]

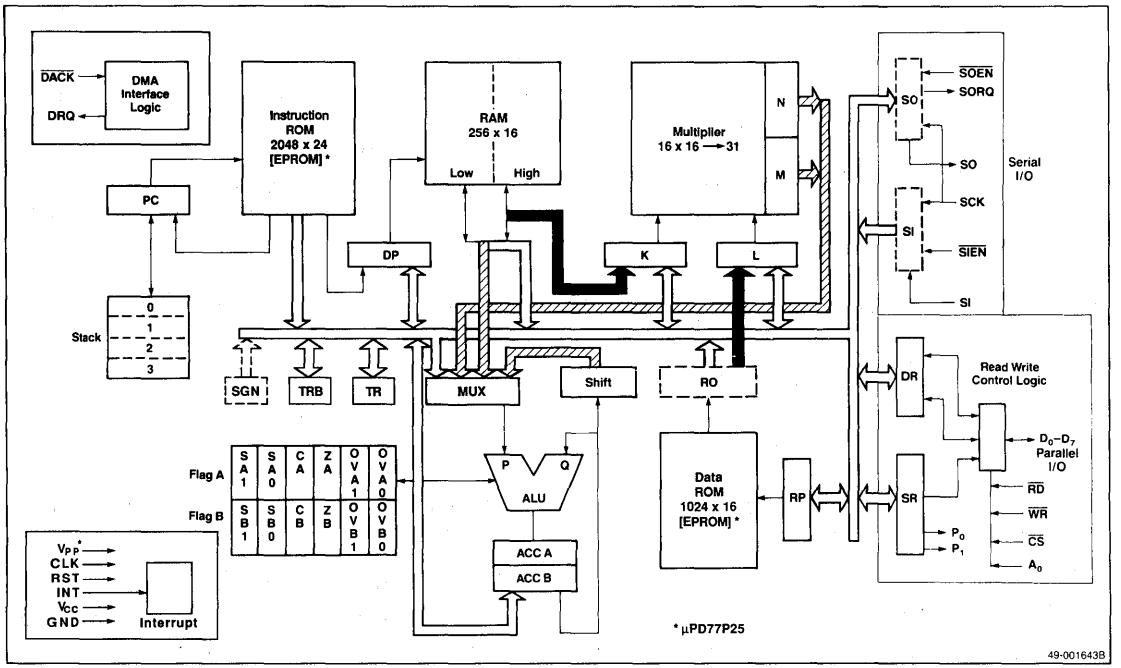
This pin is the +5-volt power supply.

NC/V_{PP}/V_{CC}

This pin is not internally connected in the 77C25. In the 77P25, this pin inputs the programming voltage (V_{PP}) when the part is being programmed.

This pin must be connected to V_{CC} for proper 77P25 operation.

Block Diagram



Functional Description

The primary bus (unshaded in the block diagram) makes a data path between all of the registers (including I/O), memory, and the processing sections. This bus is referred to as the IDB (internal data bus). The multiplier input registers K and L can be loaded not only from the IDB but alternatively via buses (darkened in the block diagram) directly from RAM to the K register and directly from data ROM to the L register. Output from the multiplier in the M and N registers is typically added via buses (shaded in the block diagram) to either accumulator A or B as part of a multi-operation instruction.

Memory

Memory is divided into three types: instruction ROM, data ROM, and data RAM. The 2048 x 24-bit words of instruction ROM are addressed by a 11-bit program counter that can be modified by an external reset, interrupt, call, jump, or return instruction.

The data ROM is organized in 1024 x 16-bit words that are addressed through a 10-bit ROM pointer (RP register). The RP may be modified simultaneously with arithmetic instructions so that the next value is available for the next instruction. The data ROM is ideal for storing the necessary coefficients, conversion tables, and other constants for signal and math processing.

The data RAM is 256 x 16-bit words and is addressed through an 8-bit data pointer (DP register). The DP has extensive addressing features that operate simultaneously with arithmetic instructions, eliminating additional time for addressing or address modification.

Arithmetic Capabilities

One of the unique features of the SPI+'s architecture is its arithmetic facilities. With a separate multiplier, ALU, and multiple internal data paths, the SPI+ is capable of carrying out a multiply, an add or other arithmetic operation, and a data move between internal registers in a single instruction cycle.

ALU

The ALU is a 16-bit two's complement unit capable of executing 16 distinct operations on data routed via the P and Q ALU inputs.

Accumulators [ACCA/ACCB]

Associated with the ALU are two 16-bit accumulators, each with its own set of flags, which are updated at the end of each arithmetic instruction. Table 2 shows the ACC A/B flag registers. In addition to zero result, sign, carry, and overflow flags, the SPI+ incorporates auxiliary overflow and sign flags (SA1, SB1, OVA1, OVB1). These flags enable the detection of an overflow condition and maintain the correct sign after as many as three successive additions or subtractions.

Table 2. ACC A/B Flag Registers

Flag A	SA1	SA0	CA	ZA	OVA1	OVA0
Flag B	SB1	SB0	CB	ZB	OVB1	OVB0

Sign Register [SGN]

When OVA1 is set, the SA1 bit will hold the corrected sign of the overflow. The SGN register will use SA1 to automatically generate saturation constants 7FFFH(+) or 8000H(−) to permit efficient limiting of a calculated value. The SGN register is not affected by arithmetic operations on accumulator B, but flags SB1, SB0, CB, ZB, OVB1, and OVB0 are affected.

Multiplier

Thirty-one bit results are developed by a 16 x 16-bit two's complement multiplier in 122 ns. The result is automatically latched to two 16-bit registers, M and N, at the end of each instruction cycle. The sign bit and 15 higher bits are in M and the 15 lower bits are in N; the LSB in N is zero. A new product is available for use after every instruction cycle, providing significant advantages in maximizing processing speed for real-time signal processing.

Stack

The SPI+ contains a four-level program stack for efficient program usage and interrupt handling.

Interrupt

The SPI+ supports a single-level interrupt. Upon sensing a high level on the INT terminal, a subroutine call to location 100H is executed. The EI bit of the status register automatically resets to 0, disabling the interrupt facility until it is reenabled under program control.

Input/Output

General

The SPI+ has three communication ports as shown in figure 1: two serial and one 8-bit parallel, each with its own control lines for interface handshaking. Parallel port operation is software-configurable to be in either polled mode or DMA mode. A general-purpose, two-line output port rounds out a full complement of interface capability.

Serial I/O

The two shift registers (SI, SO) are software-configurable to single- or double-byte transfers. The shift registers are externally clocked (SCK) to provide a simple interface between the SPI+ and serial peripherals such as A/D and D/A converters, codecs, or other SPI+s. Figure 2 shows serial I/O timing

Figure 1. SPI Communication Ports

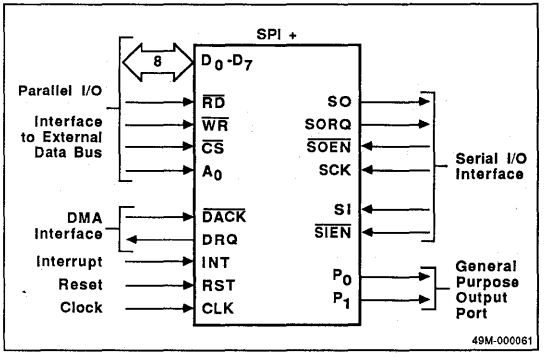
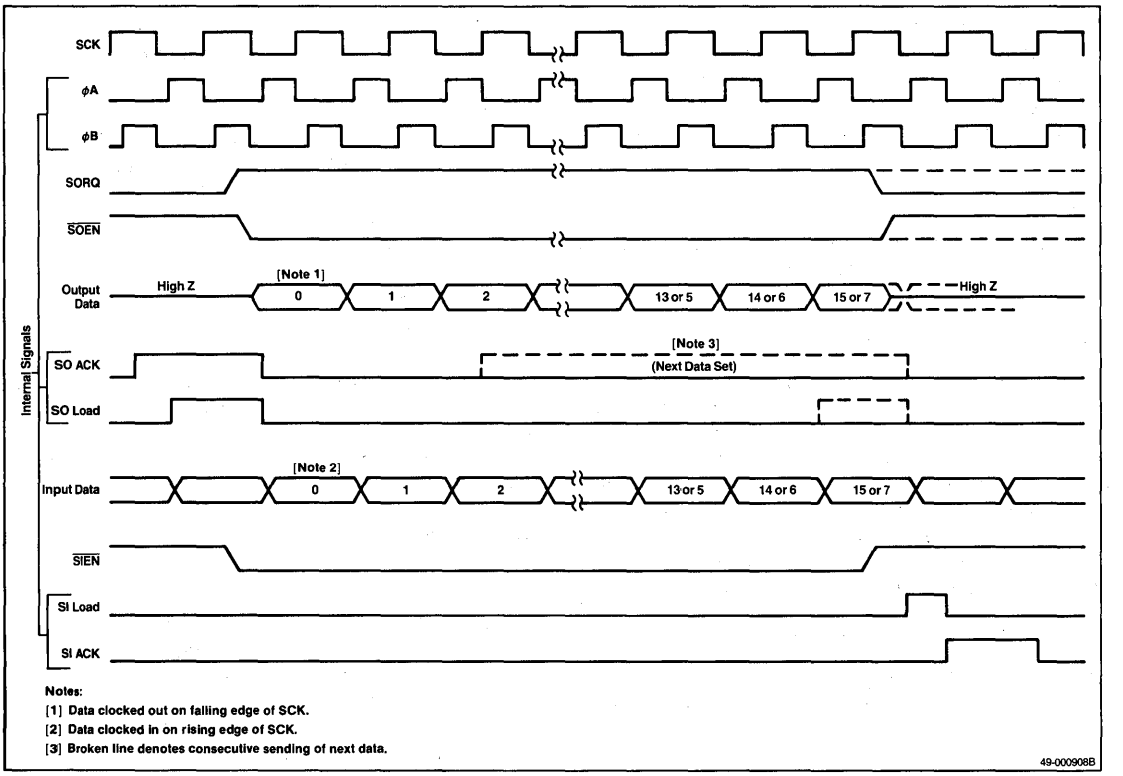


Figure 2. Serial I/O Timing



Parallel I/O

The 8-bit parallel I/O port may be used for transferring data or reading the SPI+ status as shown in table 3. Data transfer is handled through a 16-bit data register (DR) that is software-configurable for double- or single-byte data transfers. The port is ideally suited for operating with 8080, 8085, and 8086 processor buses and may be used with other processors and computer systems.

DMA Mode Option

Parallel data transfers may be controlled (optionally) via DMA control lines DRQ and $\overline{DACK}$. DMA mode allows high-speed transfers and reduced processor overhead. When in DMA mode, $\overline{DACK}$ input resets DRQ output when data transfer is completed.

Note: The RQM bit of the status register is affected by read/write operations in DMA mode the same as non-DMA mode. (In 7720 operation, RQM is not affected when in DMA mode.)

Table 3. Parallel R/W Operation

CS	A_0	WR	RD	Operation
1	X	X	X	No effect on internal operation; D_0-D_7 are at high impedance levels.
X	X	1	1	
0	0	0	1	Data from D_0-D_7 is latched to DR (Note 1)
0	0	1	0	Contents of DR are output to D_0-D_7 (Note 1)
0	1	0	1	Illegal (SR is read only)
0	1	1	0	Eight MSBs of SR are output to D_0-D_7
0	X	0	0	Illegal (may not read and write simultaneously)

Notes:

(1) Eight MSBs or 8 LSBs of data register (DR) are used, depending on DR status bit (DRS). The condition of $\overline{DACK} = 0$ is equivalent to $A_0 = CS = 0$.

Status Register

The status register, shown in figure 3, is a 16-bit register in which the 8 most significant bits may be read by the system's microprocessor for the latest parallel data I/O status. The RQM and DRS bits can only be affected by parallel data moves. The other bits can be written to (or read) by the SPI+ load immediate (LDI) or move (MOV) instructions. The EI bit is automatically reset when an interrupt is serviced.

Figure 3. Status Register

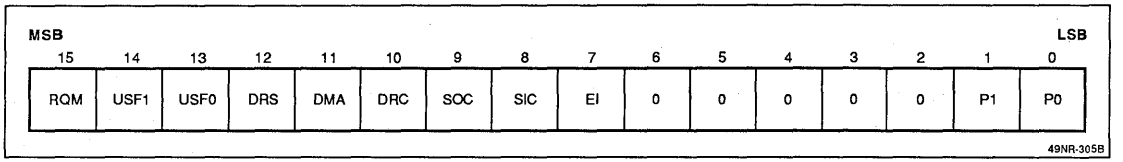


Table 4. Status Register Flags

Flag	Description
RQM (Request for master)	A read or write from DR to IDB sets RQM = 1. An external read (write) resets RQM = 0.
USF1 and USF0 (User flags 1 and 0)	General-purpose flags which may be read by an external processor for user-defined signaling
DRS (DR status)	For 16-bit DR transfers (DRC = 0). DRS = 1 after first 8 bits have been transferred. DRS = 0 after all 16 bits have been transferred.
DMA (DMA enable)	DMA = 0 (Non-DMA transfer mode) DMA = 1 (DMA transfer mode)
DRC (DR control)	DRC = 0 (16-bit mode) DRC = 1 (8-bit mode)
SOC (SO control)	SOC = 0 (16-bit mode) SOC = 1 (8-bit mode)
SIC (SI control)	SIC = 0 (16-bit mode) SIC = 1 (8-bit mode)
EI (Enable Interrupt)	EI = 0 (interrupts disabled) EI = 1 (interrupts enabled)
P1, P0 (Ports 0 and 1)	P0 and P1 directly control the state of output pins P_0 and P_1

Instructions

The SPI+ has three types of instructions: Load Immediate, Branch, and the multifunction OP instruction. Each type takes the form of a 24-bit word and executes in 122 ns.

Temporary Registers

The SPI+ has two 16-bit temporary registers.

Instruction Timing

To control the execution of instructions, the external 8-MHz clock is divided into phases for internal execution. The various elements of the 24-bit instruction word are executed in a set order. Multiplication automatically begins first. Also, data moves from source to destination before other elements of the instruction. Data being moved on the internal data bus (IDB) is available for use in ALU operations (if P-select field of the instruction specifies IDB). However, if the accumulator specified in the ASL field is also specified as the destination of the data move, the ALU operation becomes a NOP, as the data move supersedes the ALU operation.

Pointer modifications occur at the end of the instruction cycle after their values have been used for data moves. The result of multiplication is available at the end of the instruction cycle for possible use in the next instruction. If a return is specified as part of an OP instruction, it is executed last.

An assembly language OP instruction may consist of what looks like one to six lines of assembly code, but all of these lines are assembled together into one 24-bit instruction word. Therefore, the order of the six lines makes no difference in the order of execution described above. However, for understanding SPI+ operation and to eliminate confusion, assembly code should be written in the order described; that is: data move, ALU operations, data pointer modifications, and then return.

OP/RT Instruction Field Specification

Figure 4 illustrates the OP/RT instruction field specification. This is really one instruction type which is capable of executing all ALU functions listed in table 6. The ALU functions operate on the value specified by the P-select field (see table 5).

The "RT" indicates a 1-bit option in the instruction field which causes a return from subroutine or interrupt service.

Besides the arithmetic functions, this instruction can also (1) modify the RAM data pointer DP, (2) modify the data ROM pointer RP, and (3) move data along the on-chip data bus from a source register to a destination register. The possible source and destination registers are listed in tables 10 and 11 respectively. Tables 6, 7, 8, and 9 show the ASL, DPL, DPH, and RPDCR fields, respectively.

Figure 4. OP/RT Instruction Field

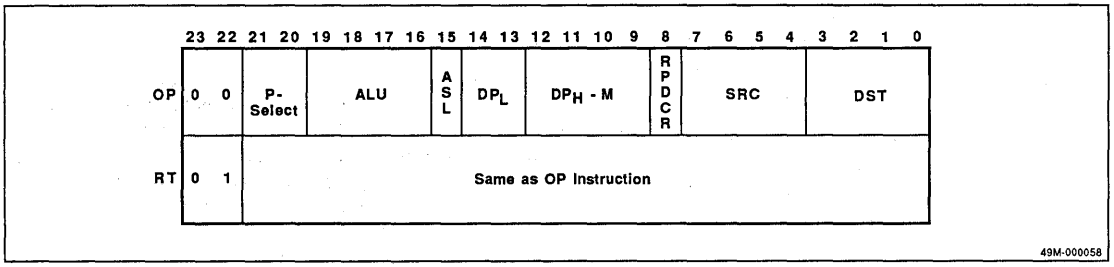


Table 5. P-Select Field

Mnemonic	D_{21}	D_{20}	ALU Input
RAM	0	0	RAM
IDB	0	1	Internal data bus (Note 1)
M	1	0	M register
N	1	1	N register

Notes:

(1) Any value on the on-chip data bus. Value may be selected from any of the registers listed in table 6 source register selections.

Table 6. ALU Field

Mnemonic	D_{19}	D_{18}	D_{17}	D_{16}	ALU Function	SA1 SB1	SA0 SB0	CA CB	ZA ZB	OVA1 OVB1	OVA0 OVB0
NOP	0	0	0	0	No operation	—	—	—	—	—	—
OR	0	0	0	1	OR	x	△	0	△	0	0
AND	0	0	1	0	AND	x	△	0	△	0	0
XOR	0	0	1	1	Exclusive OR	x	△	0	△	0	0
SUB	0	1	0	0	Subtract	△	△	△	△	△	△
ADD	0	1	0	1	Add	△	△	△	△	△	△
SBB	0	1	1	0	Subtract with borrow	△	△	△	△	△	△
ADC	0	1	1	1	Add with carry	△	△	△	△	△	△
DEC	1	0	0	0	Decrement ACC	△	△	△	△	△	△
INC	1	0	0	1	Increment ACC	△	△	△	△	△	△
CMP	1	0	1	0	Complement ACC (one's complement)	x	△	0	△	0	0
SHR1	1	0	1	1	1-bit right shift	x	△	△	△	0	0
SHL1	1	1	0	0	1-bit left shift	x	△	△	△	0	0
SHL2	1	1	0	1	2-bit left shift	x	△	0	△	0	0
SHL4	1	1	1	0	4-bit left shift	x	△	0	△	0	0
XCHG	1	1	1	1	8-bit exchange	x	△	0	△	0	0

Notes:

△ May be affected, depending on the results
— Previous status can be held
0 Reset
x Indefinite

Table 7. ASL Field

Mnemonic	D_{15}	ACC Selection
ACCA	0	ACCA
ACCB	1	ACCB

Table 8. DPL Field

Mnemonic	D_{14}	D_{13}	Low DP Modify (DP_3-DP_0)
DPNOP	0	0	No operation
DPINC	0	1	Increment DPL
DPDEC	1	0	Decrement DPL
DPCLR	1	1	Clear DPL

Table 9. DPH Field

Mnemonic	D_{12}	D_{11}	D_{10}	D_9	High DP Modify
M0	0	0	0	0	Exclusive OR of DPH (DP_7-DP_4) with the mask defined by the 4 bits (D_{12}-D_9) of the DPH field
M1	0	0	0	1	
M2	0	0	1	0	
M3	0	0	1	1	
M4	0	1	0	0	
M5	0	1	0	1	
M6	0	1	1	0	
M7	0	1	1	1	
M8	1	0	0	0	
M9	1	0	0	1	
MA	1	0	1	0	
MB	1	0	1	1	
MC	1	1	0	0	
MD	1	1	0	1	
ME	1	1	1	0	
MF	1	1	1	1	

Table 10. RPDCR Field

Mnemonic	D_8	RP Operation
RPNOP	0	No operation
RPDEC	1	Decrement RP

Table 11. SRC Field

Mnemonic	D_7	D_6	D_5	D_4	Source Register
NON/TRB	0	0	0	0	TRB (Note 1)
A	0	0	0	1	ACCA (Accumulator A)
B	0	0	1	0	ACCB (Accumulator B)
TR	0	0	1	1	TR temporary register
DP	0	1	0	0	DP data pointer
RP	0	1	0	1	RP ROM pointer
RO	0	1	1	0	RO ROM output data
SGN	0	1	1	1	SGN sign register
DR	1	0	0	0	DR data register
DRNF	1	0	0	1	DR no flag (Note 2)
SR	1	0	1	0	SR status register
SIM	1	0	1	1	SI serial in MSB (Note 2)
SIL	1	1	0	0	SI serial in LSB (Note 3)
K	1	1	0	1	K register
L	1	1	1	0	L register
MEM	1	1	1	1	RAM

Notes:

(1) DR to IDB, RQM not set. In DMA, DRQ not set.

(2) First bit in goes to MSB, last bit to LSB.

(3) First bit goes to LSB, last bit to MSB (bit reversed).

Jump/Call/Branch

Figure 5 shows the JP instruction field specification.

Three types of program counter modifications accommodated by the SPI+ are listed in table 12. All the instructions, if unconditional or if the specified condition is true, take their next program execution address from the next address field (NA); otherwise PC = PC + 1.

Table 12. BRCH Field

D_{21}	D_{20}	D_{19}	Branch Instruction
1	0	0	Unconditional jump
1	0	1	Subroutine call
0	1	0	Conditional jump

Load Data [LDI]

Figure 6 shows the LD instruction field specification.

The load data instruction will take the 16-bit value contained in the immediate data field (ID) and place it in the register specified by the destination field (DST). See table 13.

Figure 5. JP Instruction Field Specification

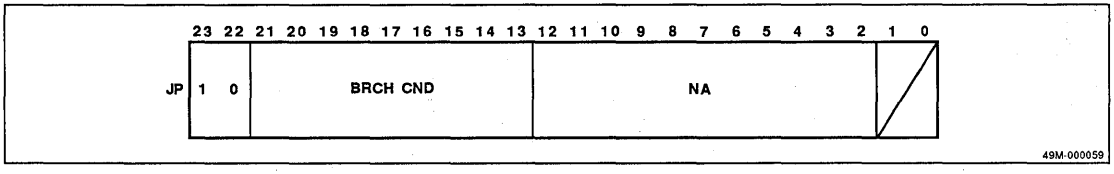


Figure 6. LD Instruction Field Specification

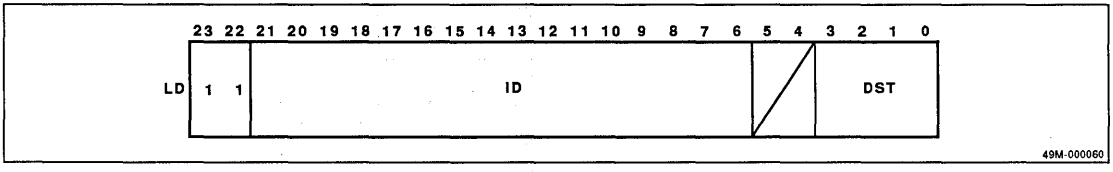


Table 13. DST Field

Mnemonic	D_3	D_2	D_1	D_0	Destination Register
@NON	0	0	0	0	No register
@A	0	0	0	1	ACCA (Accumulator A)
@B	0	0	1	0	ACCB (Accumulator B)
@TR	0	0	1	1	TR temporary register
@DP	0	1	0	0	DP data pointer
@RP	0	1	0	1	RP ROM pointer
@DR	0	1	1	0	DR data register
@SR	0	1	1	1	SR status register
@SOL	1	0	0	0	SO serial out LSB (Note 1)
@SOM	1	0	0	1	SO serial out MSB (Note 2)
@K	1	0	1	0	K (Mult)
@KLR	1	0	1	1	IDB → K, ROM → L (Note 3)
@KLM	1	1	0	0	Hi RAM → K, IDB → L (Note 4)
@L	1	1	0	1	L register
@TRB	1	1	1	0	TRB register
@MEM	1	1	1	1	RAM

Notes:

(1) LSB is first bit out.

(2) MSB is first bit out.

(3) Internal data bus to K, and ROM to L register.

(4) Contents of RAM address specified by $DP_6 = 1$, is placed in K register, IDB is placed in L (that is: 1, DP_5, DP_4, DP_3-DP_0).

Table 14. BRCH/CND Fields

Mnemonic	D_{21}	D_{20}	D_{19}	D_{18}	D_{17}	D_{16}	D_{15}	D_{14}	D_{13}	Conditions
JMP	1	0	0	0	0	0	0	0	0	No condition
CALL	1	0	1	0	0	0	0	0	0	No condition
JNCA	0	1	0	0	0	0	0	0	0	CA = 0
JCA	0	1	0	0	0	0	0	1	0	CA = 1
JNCB	0	1	0	0	0	0	1	0	0	CB = 0
JCB	0	1	0	0	0	0	1	1	0	CB = 1
JNZA	0	1	0	0	0	1	0	0	0	ZA = 0
JZA	0	1	0	0	0	1	0	1	0	ZA = 1
JNZB	0	1	0	0	0	1	1	0	0	ZB = 0
JZB	0	1	0	0	0	1	1	1	0	ZB = 1
JNOVA0	0	1	0	0	1	0	0	0	0	OVA0 = 0
JOVA0	0	1	0	0	1	0	0	1	0	OVA0 = 1
JNOVB0	0	1	0	0	1	0	1	0	0	OVB0 = 0
JOVB0	0	1	0	0	1	0	1	1	0	OVB0 = 1
JNOVA1	0	1	0	0	1	1	0	0	0	OVA1 = 0
JOVA1	0	1	0	0	1	1	0	1	0	OVA1 1
JNOVB1	0	1	0	0	1	1	1	0	0	OVB1 0
JOVB1	0	1	0	0	1	1	1	1	0	OVB1 = 1
JNSA0	0	1	0	1	0	0	0	0	0	SA0 = 0
JSA0	0	1	0	1	0	0	0	1	0	SA0 = 1
JNSB0	0	1	0	1	0	0	1	0	0	SB0 = 0
JSB0	0	1	0	1	0	0	1	1	0	SB0 = 1
JNSA1	0	1	0	1	0	1	0	0	0	SA1 = 0
JSA1	0	1	0	1	0	1	0	1	0	SA1 = 1
JNSB1	0	1	0	1	0	1	1	0	0	SB1 = 0
JSB1	0	1	0	1	0	1	1	1	0	SB1 = 1
JDPL0	0	1	0	1	1	0	0	0	0	DPL = 0
JDPLN0	0	1	0	1	1	0	0	0	1	DPL ≠ 0
JDPLF	0	1	0	1	1	0	0	1	0	DPL = FH
JDPLNF	0	1	0	1	1	0	0	1	1	DPL ≠ FH
JNSIAK	0	1	0	1	1	0	1	0	0	SI ACK = 0
JSIAK	0	1	0	1	1	0	1	1	0	SI ACK = 1
JNSOAK	0	1	0	1	1	1	0	0	0	SO ACK = 0
JSOAK	0	1	0	1	1	1	0	1	0	SO ACK = 1
JNRQM	0	1	0	1	1	1	1	0	0	RQM = 0
JRQM	0	1	0	1	1	1	1	1	0	RQM = 1

Absolute Maximum Ratings

T_A = 25°C unless otherwise specified

Supply voltage, V_{CC}	−0.5 to +7.0 V
V_{PP}*	−0.5 to 13.5 V
Input voltage, V_I	−0.5 to V_{DD} +0.5 V
V_{RST}*	−0.5 to V_{DD} +13 V
Output Voltage, V_O	−0.5 to V_{DD} +0.5 V
Storage temperature, T_{STG}	−65 to 150°C

*For μPD77P25

Comment: Exposing the device to stresses above those listed in Absolute Maximum Ratings could cause permanent damage. The device is not meant to be operated under conditions outside the limits described in the operational sections of this specification. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Recommended Operating Conditions

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
Supply voltage	V_{CC}	4.5	5.0	5.5	V	For normal operation
		5.75	6.0	6.25	V	For writing*
	V_{PP}*	4.5	5.0	5.5	V	For reading and normal operation
		12.2	12.5	12.8	V	For writing
Input voltage, low	V_{IL}	−0.3		0.8	V	
Input voltage, high	V_{IH}	2.2		V_{DD} + 0.3	V	
CLK input voltage, low	V_{ILC}	−0.3		0.5	V	
CLK input voltage, high	V_{IHC}	3.5		V_{DD} + 0.3	V	
Input voltage for setting Prom Mode	V_{RST}*	11.5	12.0	12.5	V	For reading and writing
Operating temperature	T_{OPT}	−40	25	85	°C	Still air (μPD77C25)
		−10	25	70	°C	Still air (μPD77P25)
		20	25	30	°C	For Prom Mode*

*For μPD77P25

DC Characteristics, Normal

T_A = −40 to +85°C (μPD77C25), −10 to +70°C (μPD77P25); V_{CC} = 4.5 to 5.5 V

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
Output voltage, low	V_{OL}			0.45	V	I_{OL} = 2.0 mA
Output voltage, high	V_{OH}	0.7 V_{DD}			V	I_{OH} = 400 μA
Input leakage current, low	I_{LIL}			−10	μA	V_{IN} = 0 V
Input leakage current, high	I_{LIH}			10	μA	V_{IN} = V_{DD}
Output leakage current, low	I_{LOL}			−10	μA	V_{OUT} = 0.47 V
Output leakage current, high	I_{LOH}			10	μA	V_{OUT} = V_{DD}
Supply current (μPD77C25)	I_{CC}		25	50	mA	f_{CLK} = 8.192 MHz
			15	25	mA	f_{CLK} = 8.192 MHz; RST = "1"
			6		mA	f_{CLK} = 500 KHz; RST = "1"
Supply current (μPD77P25)	I_{CC}		35	60	mA	f_{CLK} = 8.192 MHz
			20	35	mA	f_{CLK} = 8.192 MHz; RST = "1"
	I_{PP}			1	mA	

DC Characteristics, Prom Mode

T_A = +20 to +30°C; V_{CC} = 5.75 to 6.25 V

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
Input leakage current	I_{RST}			30	μA	V_{RST} = 12.0 ± 0.5 V
Supply current	I_{CC}			60	mA	
	I_{PP}			30	mA	

Capacitance

T_A = 25°C; V_{CC} = 0 V

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
CLK, SCK capacitance	C_ϕ			20	pF	fc = 1 MHz
Input capacitance	C_{IN}			20	pF	
Output capacitance	C_{OUT}			20	pF	

AC Characteristics

T_A = −40 to +85°C (μPD77C25), −10 to +70°C (μPD77P25);
V_{CC} = 4.5 to 5.5 V

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
Clock						
CLK cycle time	t_{CYC}	120	122	2000	ns	
CLK pulse width						Measuring at 2 V
μPD77C25	t_{CC}	55			ns	
μPD77P25	t_{CC}	60			ns	
CLK rise time	t_{CR}			10	ns	Measuring at 1 and 3 V
CLK fall time	t_{CF}			10	ns	
SCK cycle time	t_{CYS}	240	244		ns	
SCK high pulse width	t_{SSH}	100			ns	
SCK low pulse width	t_{SSL}	100			ns	
SCK rise time	t_{SR}			20	ns	
SCK fall time	t_{SF}			20	ns	
Host Interface Timing						
A0, $\overline{CS}$, $\overline{DACK}$ setup time for $\overline{RD}$	t_{SAR}	0			ns	
A0, $\overline{CS}$, $\overline{DACK}$ hold time for $\overline{RD}$	t_{HRA}	0			ns	
$\overline{RD}$ pulse width	t_{WRD}	120			ns	
A0, $\overline{CS}$, $\overline{DACK}$ setup time for $\overline{WR}$	t_{SAW}	0			ns	
A0, $\overline{CS}$, $\overline{DACK}$ hold time for $\overline{WR}$	t_{HWA}	0			ns	
$\overline{WR}$ pulse width	t_{WWR}	120			ns	
Data setup time for $\overline{WR}$	t_{SDW}	100			ns	
Data hold time for $\overline{WR}$	t_{HWD}	0			ns	
$\overline{RD}$, $\overline{WR}$ recovery time	t_{RV}	100			ns	
$\overline{DACK}$ hold time for DRQ	t_{HRQA}	0.5 t_{CYC}			ns	
$\overline{RD}$, $\overline{WR}$ setup time for CLK	t_{SRWC}	50			ns	(Note 1)
$\overline{RD}$, $\overline{WR}$ hold time for CLK	t_{HCRW}	50			ns	(Note 1)

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
Host Interface Switching						
$\overline{RD}$ ↓ → data delay time	t_{DRD}			100	ns	
$\overline{RD}$ ↑ → data float time	t_{FRD}	10		65	ns	
CLK ↑ → DRQ delay time	t_{DCRQ}			80	ns	
$\overline{DACK}$ ↓ → DRQ delay time	t_{DARQ}			110	ns	
CLK ↑ → P0, P1 delay time	t_{DCP}			100	ns	
Interrupt Reset Timing						
RST setup time for CLK	t_{SRSC}	50			ns	(Note 1)
RST hold time for CLK	t_{HCRS}	50			ns	(Note 1)
RST pulse width	t_{RST}	2 t_{CYC}			ns	For system reset
		3 t_{CYC}			ns	For enter power saving state
INT setup time for CLK	t_{SINC}	50			ns	(Note 1)
INT hold time for CLK	t_{HCIN}	50			ns	(Note 1)
INT pulse width	t_{INT}	3 t_{CYC}			ns	
INT recovery time	t_{RINT}	2 t_{CYC}			ns	
Interrupt Reset Switching						
CLK ↑ → reset state delay time	t_{DCRS}			100	ns	
Serial Interface Timing						
$\overline{SIEN}$, SI setup time for SCK	t_{SSIS}	50			ns	
$\overline{SIEN}$, SI hold time for SCK	t_{HSSI}	30			ns	
$\overline{SOEN}$ setup time for SCK	t_{SSES}	50			ns	

AC Characteristics (cont)

$T_A = -40$ to $+85°C$; $V_{CC} = 4.5$ to 5.5 V

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
Serial Interface Timing (cont)						
SOEN hold time for SCK	t_{HSSE}	30			ns	
CLK setup time for SCK	t_{SCS}	50			ns	(Note 1)
CLK hold time for SCK	t_{HSC}	50			ns	(Note 1)
SCK setup time for CLK	t_{SSC}	50			ns	(Note 1)
SCK hold time for CLK	t_{HCS}	50			ns	(Note 1)
Serial Interface Switching						
SCK↑ → SORQ delay time	t_{DSSQ}	30		150	ns	
SCK↓ → SO delay time	t_{DSLSO}			60	ns	
SCK↓ → SO hold time	t_{HSLSO}	0			ns	
SCK↓ → SO float time	t_{FSSO}			60	ns	

Notes:

(1) Setup and hold requirement for asynchronous signal only guarantees recognition at next CLK.

UVPROM Programming Timing (Read)

$T_A = 25 \pm 5°C$; $V_{CC} = 5.0 \pm 0.5$ V; $V_{PP} = V_{CC}$; $V_{IHR} = 12.0 \pm 0.5$ V

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
Data Read (Prom Mode)						
CE setup time for RST	t_{SRSCE}	2			μs	
OE setup time for RST	t_{SRSOE}	2			μs	
Data Read Switching (Prom Mode)						
Address to output delay	t_{DAD}			200	ns	
CE to output delay	t_{DCD}			200	ns	
OE to output delay	t_{DODR}			75	ns	
OE high to output float	t_{FCD}	0		60	ns	
Address to output hold	t_{HAD}	0			ns	

UVPROM Programming Timing (Write)

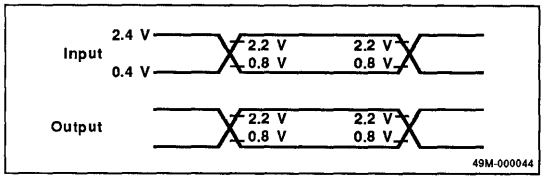
$T_A = 25 \pm 5°C$; $V_{CC} = 6.0 \pm 0.25$ V; $V_{PP} = 12.5 \pm 0.3$ V; $V_{IHR} = 12.0 \pm 0.5$ V

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
Data Write (Prom Mode)						
CE setup time for RST	t_{SRSCE}	2			μs	
CE setup time for address	t_{SAC}	2			μs	
CE setup time for data	t_{SDC}	2			μs	
CE setup time for V_{PP}	t_{SVPC}	2			μs	
CE setup time for V_{DD}	t_{SVDC}	2			μs	
OE setup time for data	t_{SDO}	2			μs	
Address hold time	t_{HCA}	2			μs	
Data hold time	t_{HCD}	2			μs	
Initial program pulse width	t_{WCO}	0.95	1.0	1.05	ms	
Overprogram pulse width	t_{WC1}*	2.85		78.75	ms	
Data Write Switching (Prom Mode)						
OE to output float time	t_{FOD}	0		130	ns	
OE to output delay	t_{DODW}			150	ns	

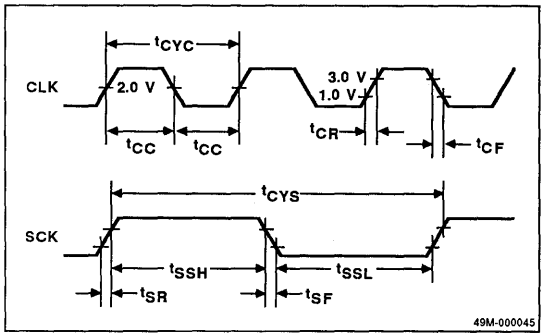
*$t_{WC1} = 3nt_{WCO}$ assuming initial program pulse is applied n times.

Timing Waveforms

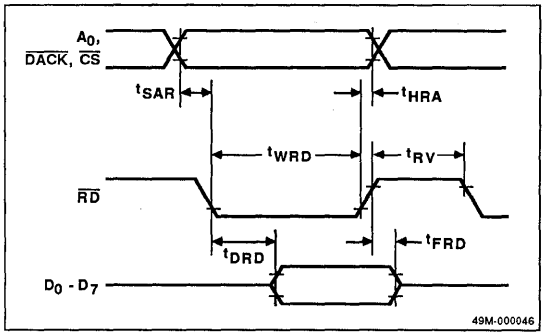
Input/Output Voltage Reference Levels



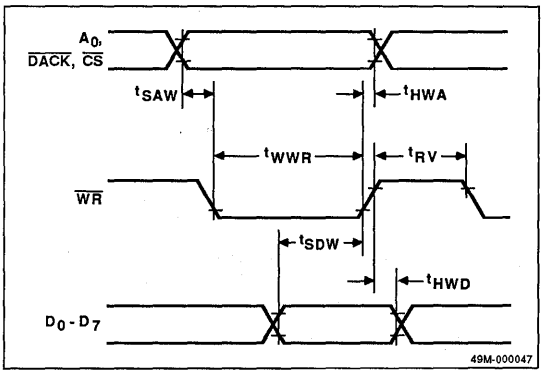
Clock Timing



Host Read Operation



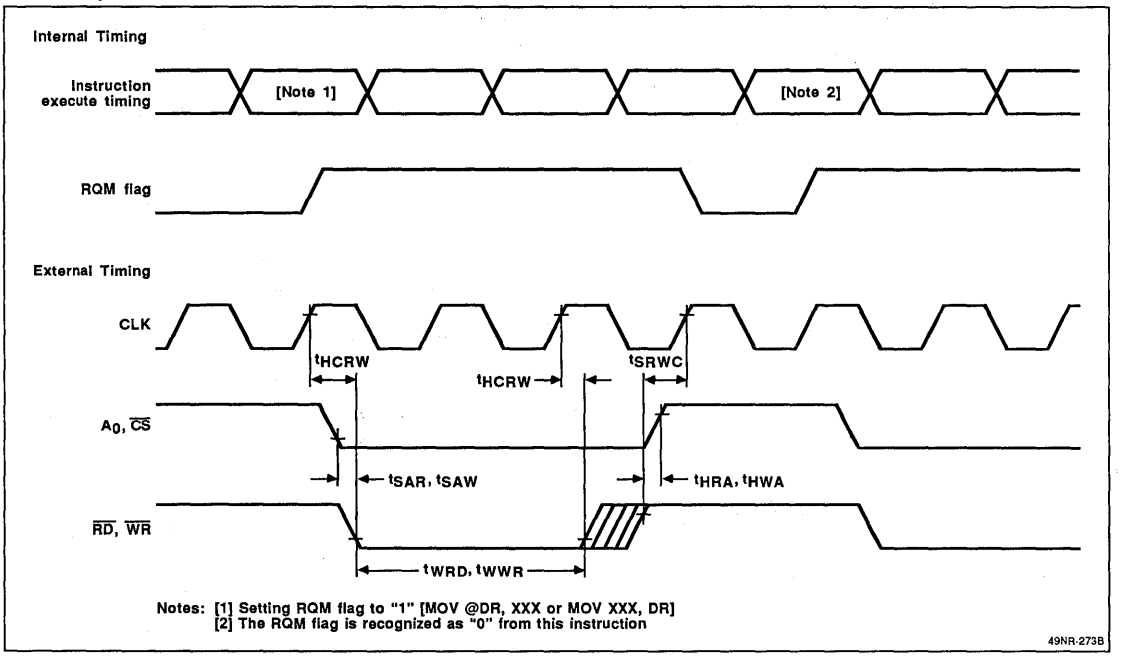
Host Write Operation



2

Timing Waveforms (cont)

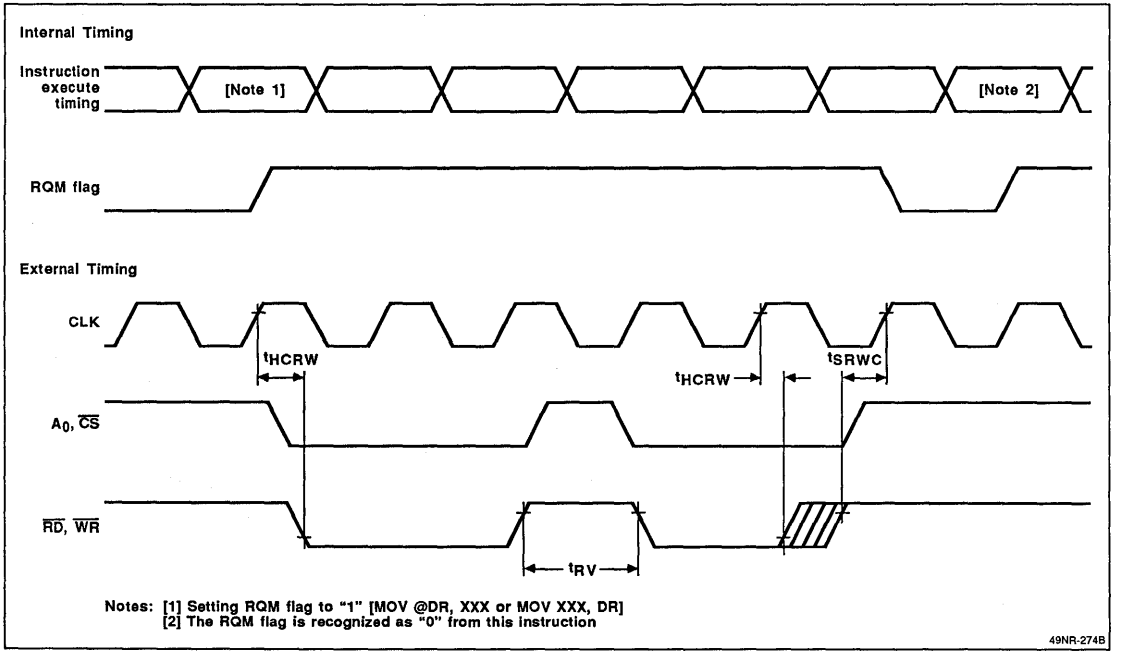
Normal Operation #1, 8-Bit Mode



Notes: [1] Setting RQM flag to "1" [MOV @DR, XXX or MOV XXX, DR]
[2] The RQM flag is recognized as "0" from this instruction

49NR-273B

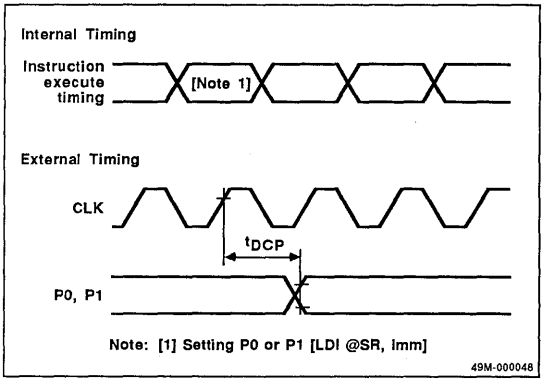
Normal Operation #2, 16-Bit Mode



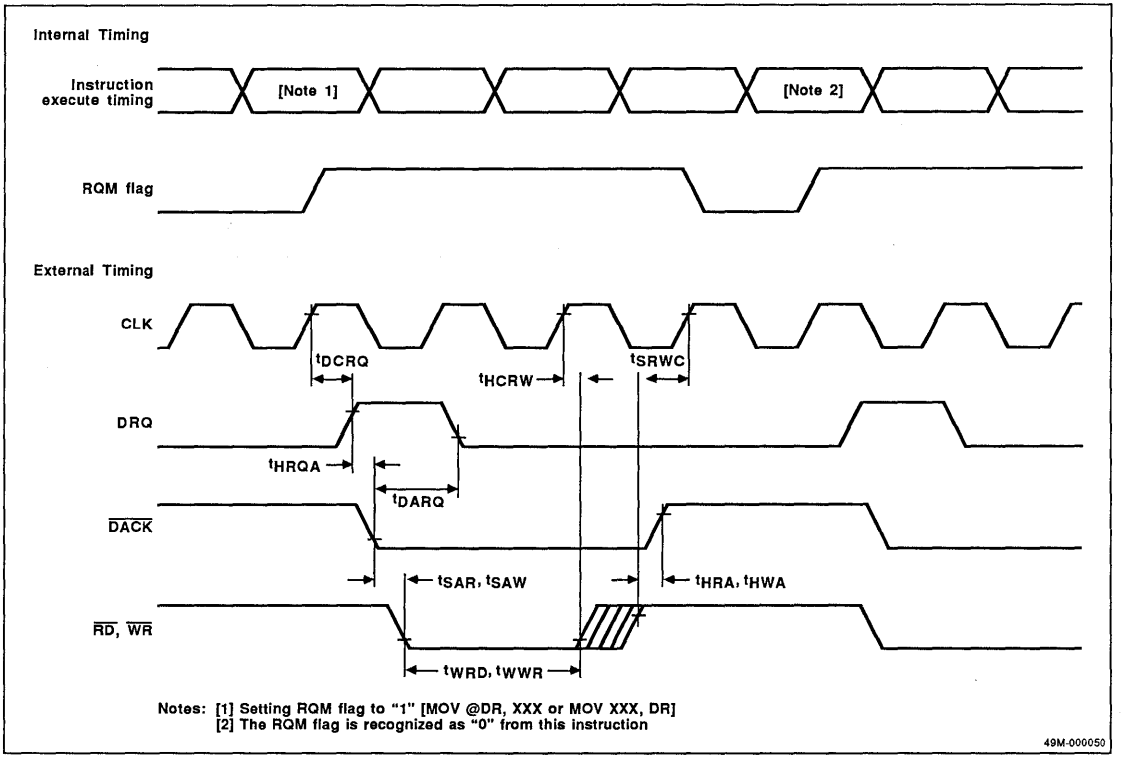
Notes: [1] Setting RQM flag to "1" [MOV @DR, XXX or MOV XXX, DR]
[2] The RQM flag is recognized as "0" from this instruction

49NR-274B

Port Operation

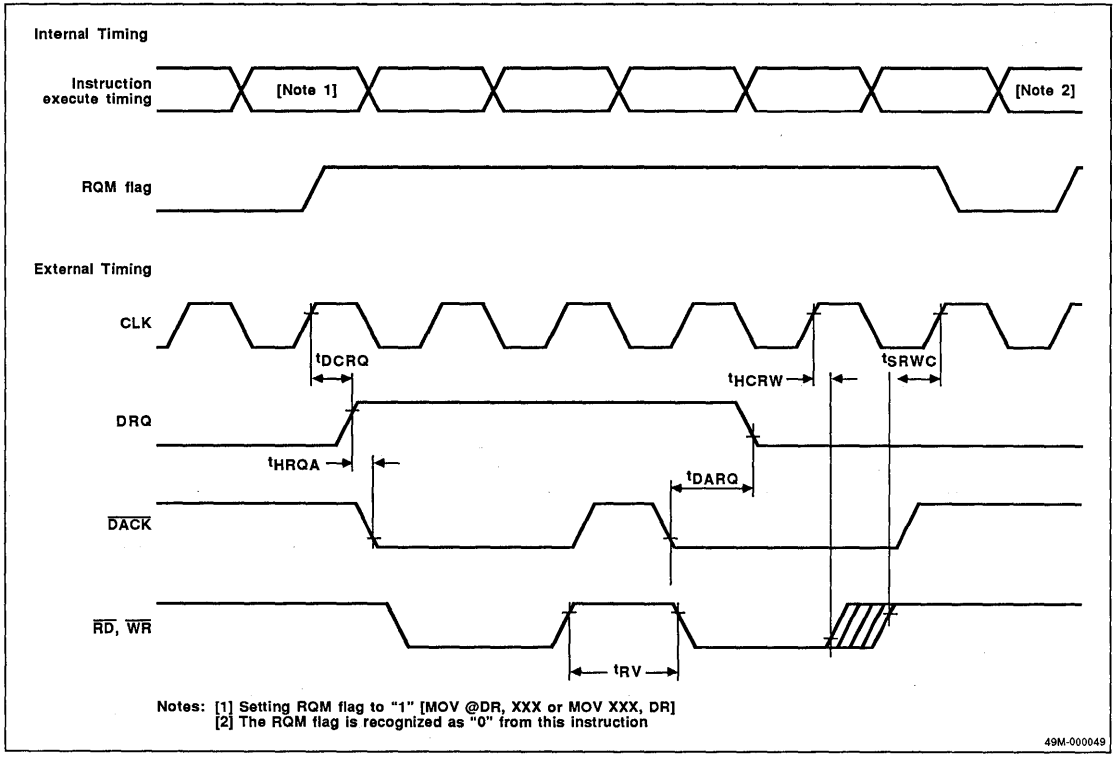


DMA Operation #1, 8-Bit Mode



Timing Waveforms (cont)

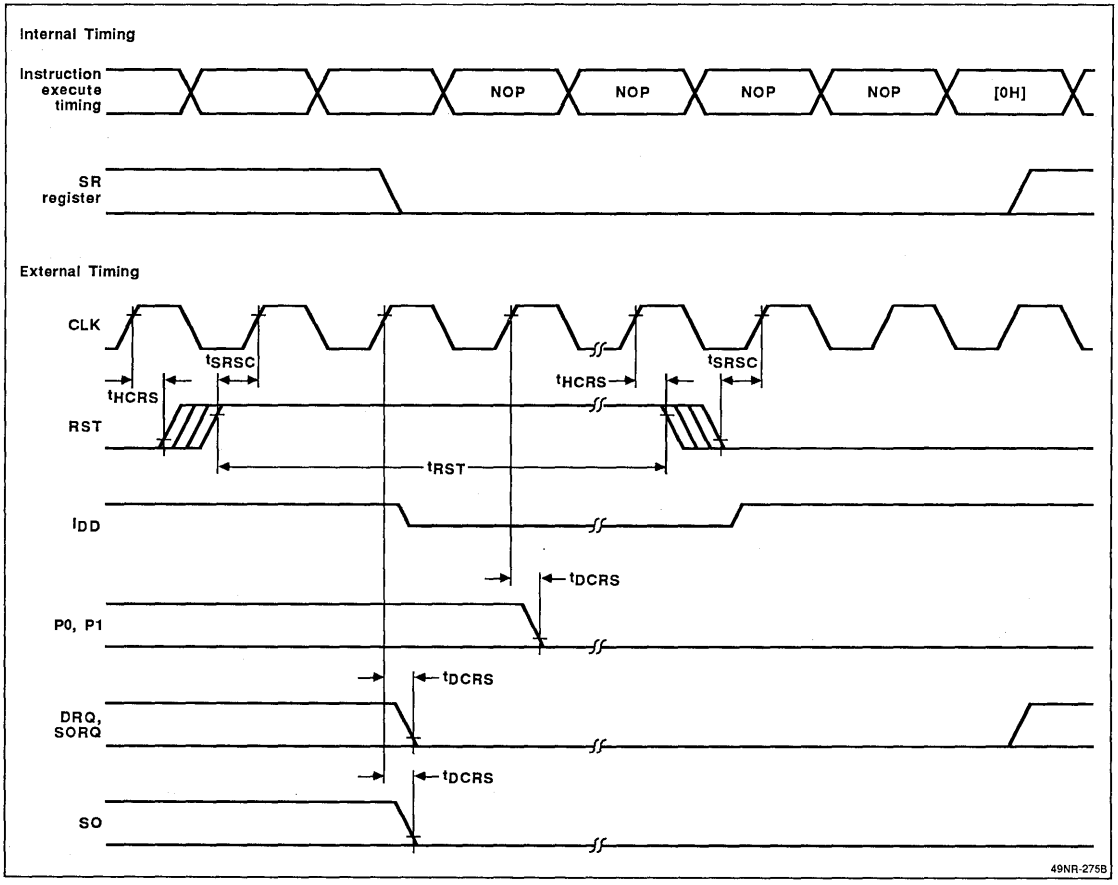
DMA Operation #2, 16-Bit Mode



Notes: [1] Setting RQM flag to "1" [MOV @DR, XXX or MOV XXX, DR]
[2] The RQM flag is recognized as "0" from this instruction

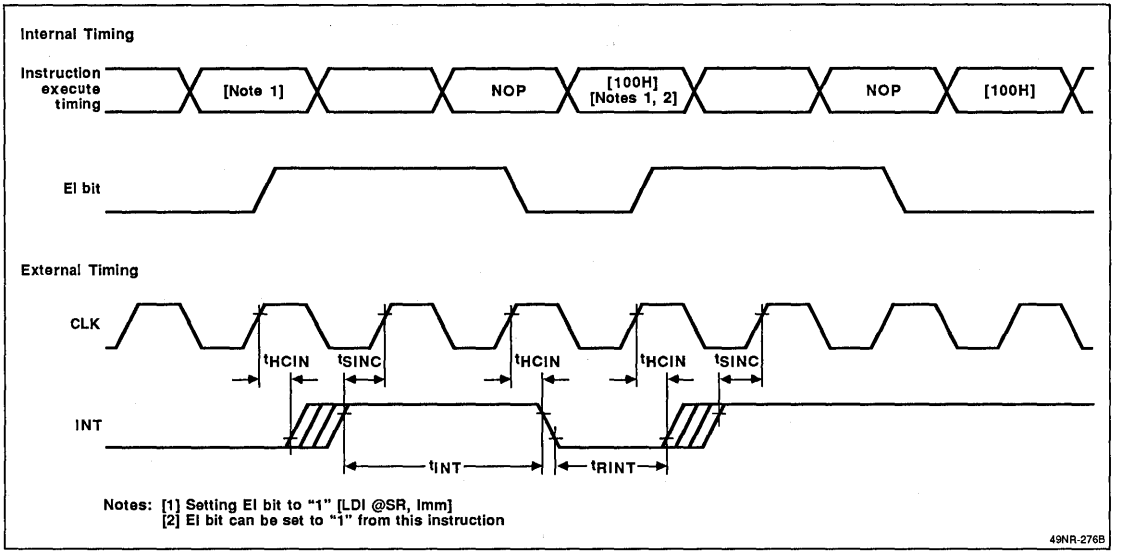
49M-000049

Reset Operation

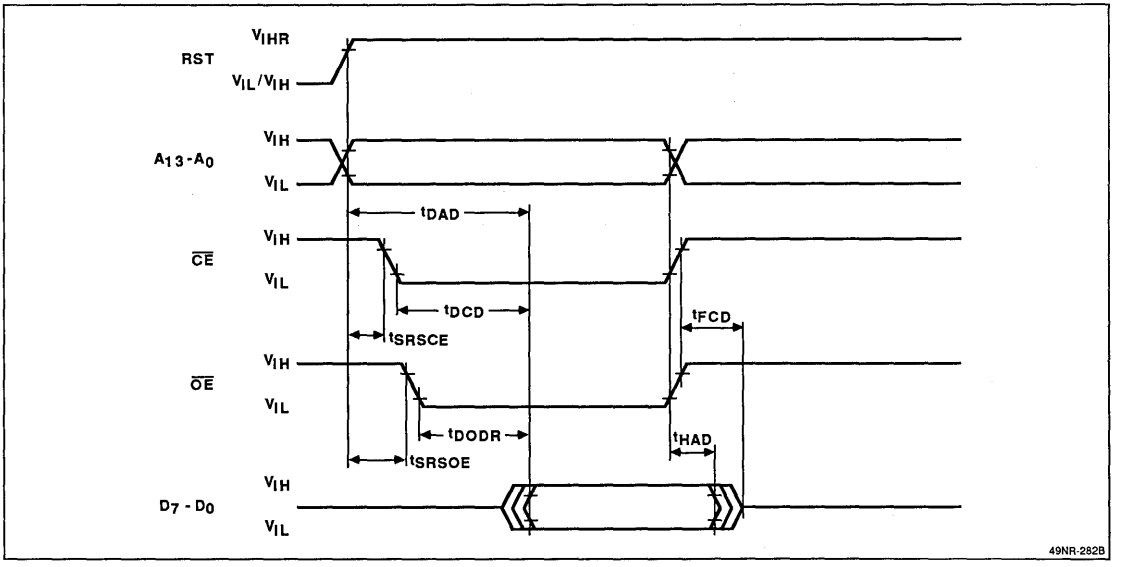


Timing Waveforms (cont)

Interrupt Operation



On-Chip UVEPROM Read Timing



On-Chip UVEPROM Write Timing

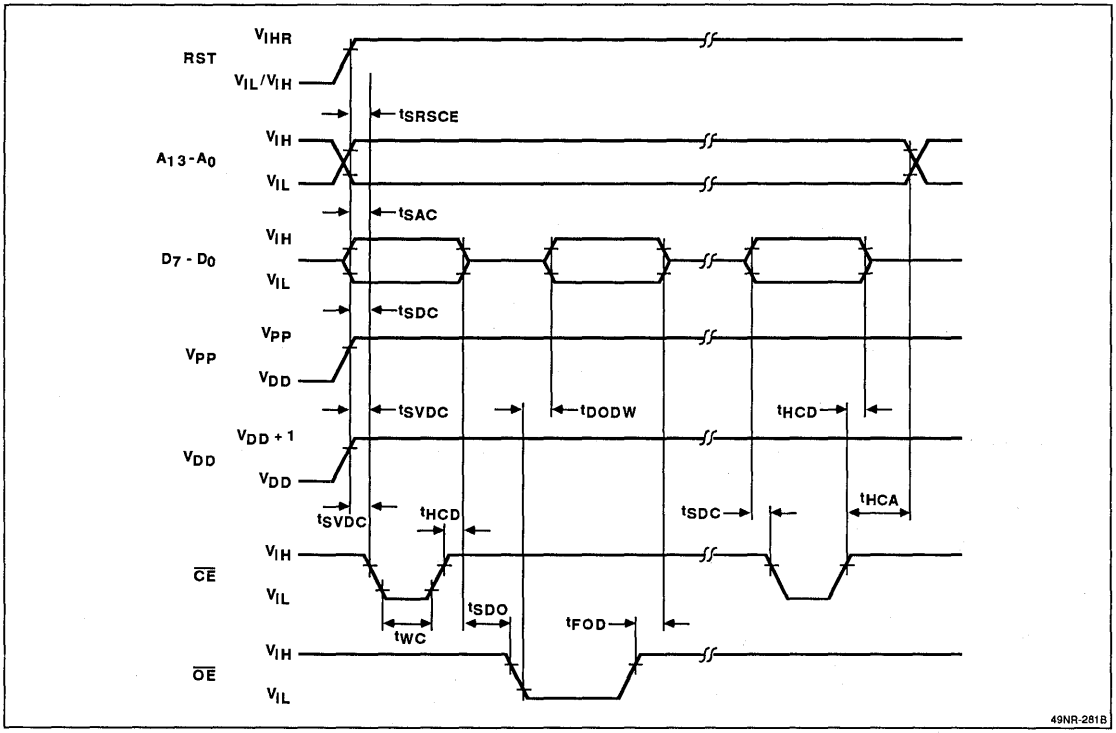
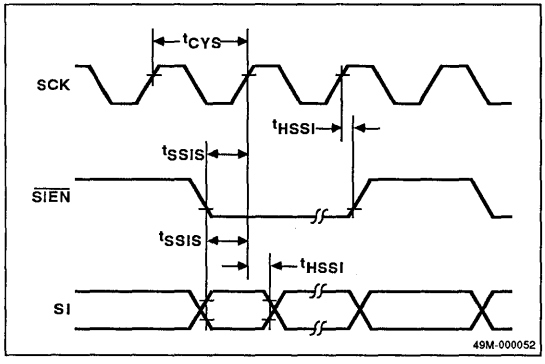


Figure 7. Serial Input Operation



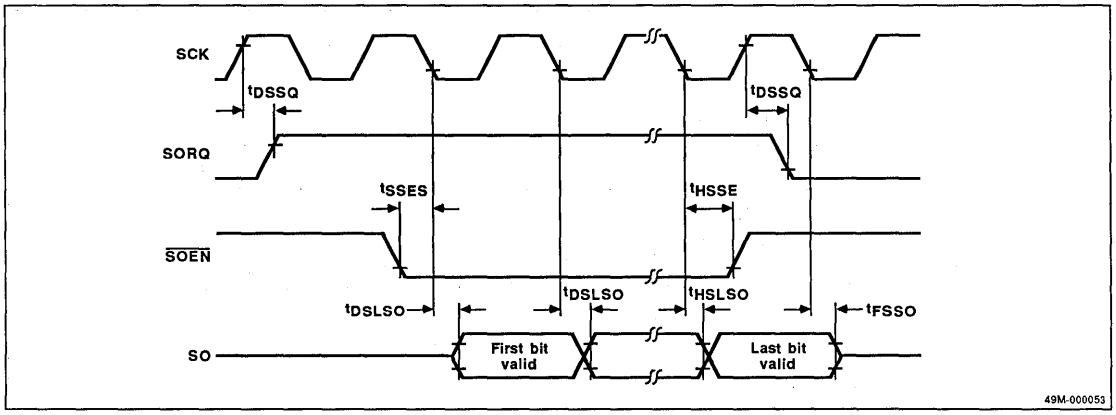
Serial Timing

Serial Output Case 1: $\overline{SOEN}$ Asserted in Response to SORQ

Figure 8 shows timing for serial output when $\overline{SOEN}$ is asserted in response to SORQ. If $\overline{SOEN}$ is held inactive until after SORQ is asserted, and then $\overline{SOEN}$ is asserted at least t_{SSES} before the falling edge of SCK, SO will become valid t_{DSLSO} after the falling edge of SCK for use by an external device at the subsequent rising edge of SCK.

Note that, although figure 8 shows $\overline{SOEN}$ being asserted during a different SCK pulse than the one in which SORQ is asserted, it is permissible for these to occur during the same pulse of SCK as long as $\overline{SOEN}$ is still asserted t_{SSES} before the falling edge of SCK.

Figure 8. Serial Output Case #1



Serial Output Case 2: $\overline{SOEN}$ Active before SORQ is High

Figure 9 shows output timing when $\overline{SOEN}$ is active before SORQ is high. If $\overline{SOEN}$ is held active before SORQ is high, data will be shifted out whenever it becomes available in the serial output register (assuming previous data is already shifted out). In this case, SORQ will rise t_{DSSQ} after a rising edge of SCK. The first SO bit occurs t_{DSLSO} after the next falling edge of SCK for use by an external device at the subsequent rising edge of SCK.

Subsequent bits will be shifted out t_{DSLSO} after subsequent falling edges of SCK for use at subsequent rising edges of SCK. The last bit to be shifted out will also follow this pattern, and will be held valid t_{FSSO} after the corresponding falling edge of SCK at which it is to be used. SORQ will be held t_{DSSQ} after this same rising edge of SCK, and then removed.

Serial Output Case 3: $\overline{SOEN}$ Released in Middle of a Transfer

If $\overline{SOEN}$ is released while SCK is in the middle of a transfer, as shown in figure 10, at least t_{HSSE} after the falling edge of SCK, then the next bit will be shifted out t_{DSLSO} after the falling edge of SCK for use at the subsequent rising edge of SCK. SO will go inactive t_{FSSO} after the falling edge of SCK.

Note: For all its uses, $\overline{SOEN}$ must not change state within t_{SSES} before or t_{HSSE} after the falling edge of SCK; otherwise the results will be indeterminate.

Figure 9. Serial Output Case #2

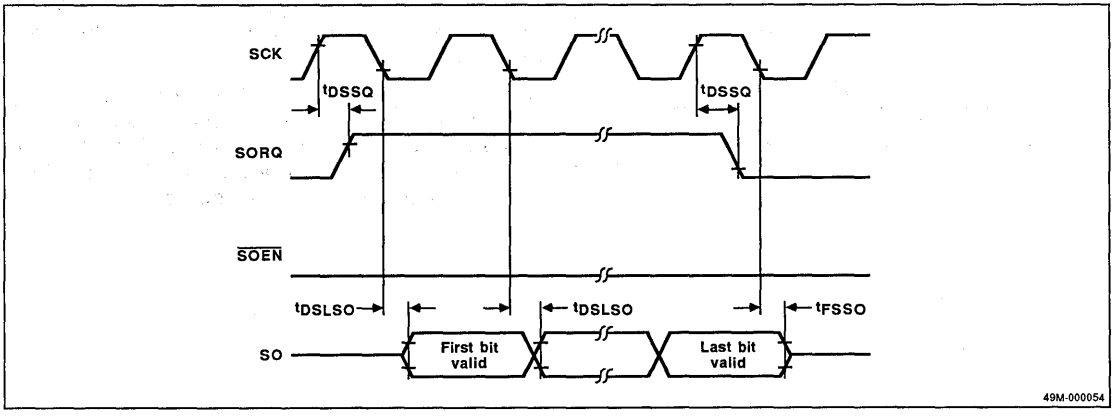
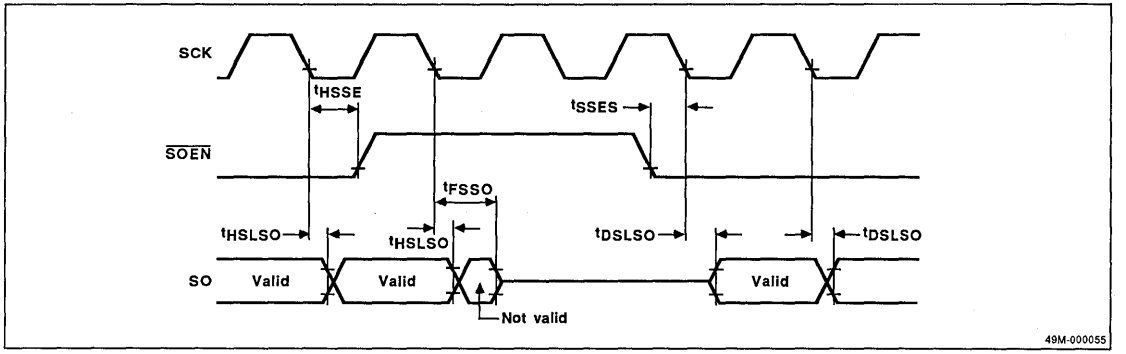


Figure 10. Serial Output Case #3



Serial Input

Serial input timing, shown in figure 11, is much simpler than serial output timing, shown in figure 12. Data bits are shifted in on the rising edge of SCK if SIEN is asserted. Both SIEN and SI must be stable at least t_{SSIS} before and t_{HSSI} after the rising edge of SCK; otherwise the results will be indeterminate.

Figure 11. Serial Input Timing Example

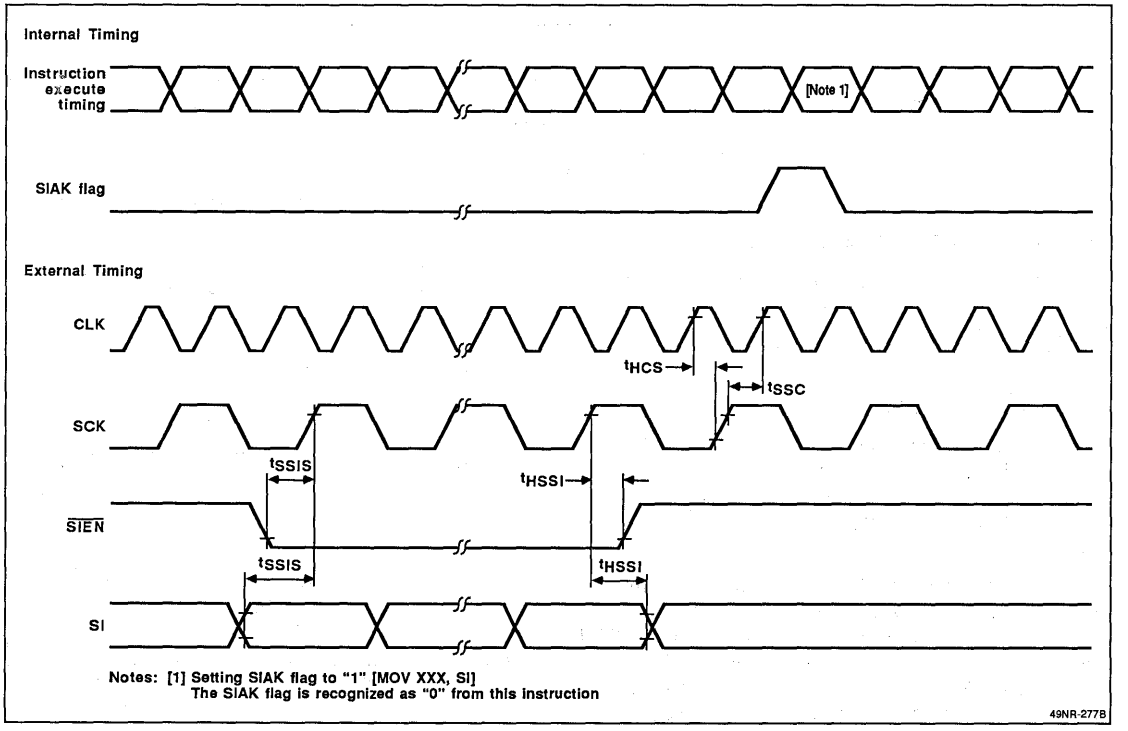
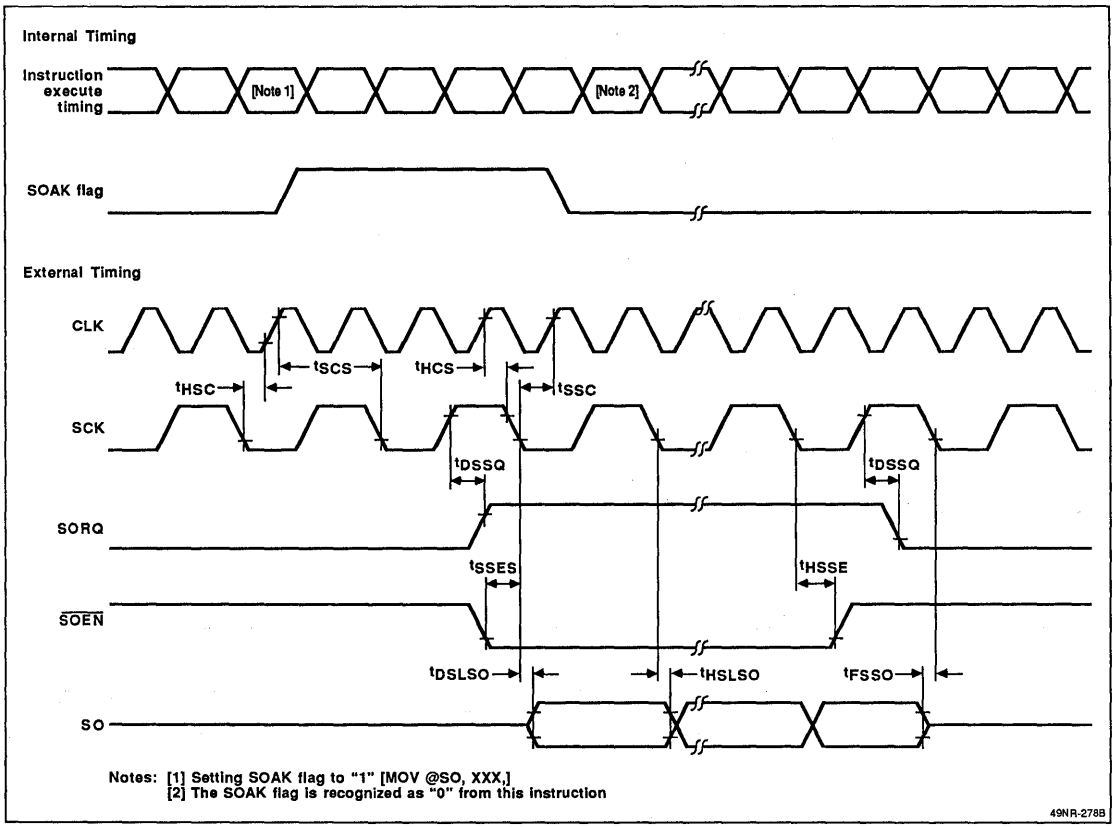


Figure 12. Serial Output Timing Example



μPD77P25 UVEPROM Interface

Both the instruction ROM and data ROM of the μPD77P25 are an ultraviolet-light erasable UVEPROM. The following describes how to write to and read from the UVEPROM.

Input/Output Data Format

One word of the instruction ROM consists of 24 bits, while 16 bits make up one data ROM word. Data are written to or read from these UVEPROMs in units of bytes or 8 bits. Therefore, special addresses are assigned to the UVEPROMs.

Figure 13 shows the address assignment. Addresses 0H through 1FFFH are assigned to the 2 K-word instruction ROM. The addresses 2000H through 27FFH are assigned to the 1 K-word data ROM. Since the instruction ROM is configured on a 1-word-for-24-bit basis, one dummy byte address is provided per word.

For example, data in word address 0H of the instruction ROM is equivalent to three bytes of byte addresses 0H to 2H. 3H is a dummy address. Data ROM is shown in figure 14.

Figure 13. Instruction ROM

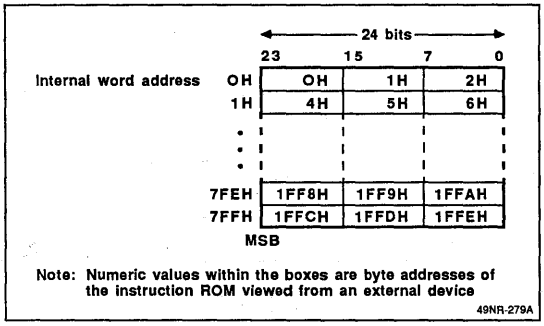
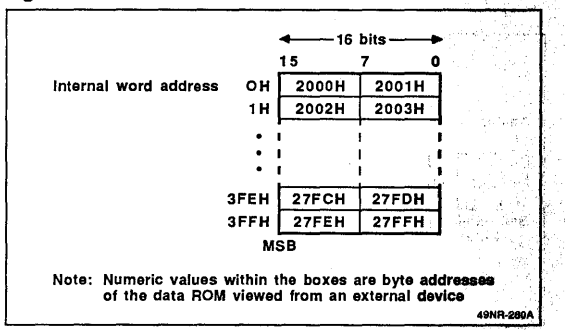


Figure 14. Data ROM



Erasing

The data in the μPD77P25's UVEPROMs can be erased by exposing them to a light with a wavelength shorter than 400 nm. All data in the UVEPROMs are set to "1s" after the erasure. If the μPD77P25 is exposed to direct sunlight or fluorescent light for a long time, the data might be erased. To prevent this, the UVEPROM window must be masked with a cover or film to shield it from the ultraviolet light. Usually, the UVEPROMs are erased when exposed to ultraviolet light with a wavelength of 254 nm. The total light quantity required to completely erase the written data is 15WS/cm^2 (UV intensity x erase time) that is equivalent to exposure to a UV lamp with a wavelength of 12000μW/cm^2 for about 15 to 20 minutes. However, a longer erasing time may be required due to such factors as the life of the UV lamp and stains on the window of the package. The μPD77P25 must be positioned within one inch away from the UV lamp.

Write Mode

To write data, the μPD77P25 UVEPROMs must be first erased as described in the preceding section. Perform the writing operation observing the following procedures. Table 15 shows reassigned pin functions when writing/reading UVEPROMs.

(1) Apply +12.5 V to RST (pin 16), + 6 V to V_{CC}, and +12.5 V to V_{PP}. This causes the UVEPROMs to enter write mode.

(2) Specify the desired ROM byte address from address input pins A_0 to A_{13}.

(3) Write the data on the data bus (D_0-D_7) by applying "0" to $\overline{CE}$ while $\overline{OE}$ is "1" (program mode).

(4) Output the written data to the data bus (D_0-D_7) by applying "0" to $\overline{OE}$ while $\overline{CE}$ is "1" (program verify mode).

(5) Repeat steps 2 through 4, 25 times maximum until the data is properly written to the specified address.

(6) After verifying that the data has been properly written, apply additional pulses by setting $\overline{OE}$ to "1" (clear $\overline{CE}$ to "0"). The pulse width is 3X ms if the number of repetitions in 3 and 4 is X.

The above procedure completes writing one byte of data. If the data will not be properly written even after steps 2 to 4 have been repeated more than 25 times, the μPD77C25 is defective.

Table 15. Pin Functions for Writing/ Reading UVEPROM

Pin Name	DIP Pin Number	Pin Name For Normal Operation	Function
A_0	27	A_0	Input address (viewed from external device) for writing/ reading UVEPROM (instruction ROM and data ROM).
A_1	24	$\overline{WR}$	
A_2	23	SORQ	
A_3	22	SO	
A_4	21	SI	
A_5	20	$\overline{SOEN}$	
A_6	19	$\overline{SIEN}$	
A_7	18	SCI	
A_8	17	INT	
A_9	15	CLK	
A_{10}	5	P1	
A_{11}	4	P0	
A_{12}	3	DRQ	
A_{13}	2	$\overline{DACK}$	
D_0-D_7	6-13	D_0-D_7	Inputs/outputs data for UVEPROM (instruction ROM and data ROM)
$\overline{CE}$	26	$\overline{CS}$	UVEPROM write strobe signal (active low)
$\overline{OE}$	25	$\overline{RD}$	UVEPROM read strobe signal (active low)
V_{PP}	1	V_{PP}	Power pin for writing UVEPROM; Apply +12.5 V for writing and +5 V for reading.
V_{CC}	28	V_{CC}	Power pin; Apply +6 V for writing and +5 V for reading.
GND	14	GND	Ground pin
—	16	RST	Sets UVEPROM write or read mode. Mode is set when +12.5 V is applied.

Read Mode

(1) Apply +12.5 V to RST (pin 16), + 5 V to V_{CC}, and +5.0 V to V_{PP}. This causes the UVEPROMs to enter read mode.

(2) Specify the desired ROM byte address from the address input pins A_0 to A_{13}.

(3) Data will be output to the data bus (D_0-D_7) by clearing $\overline{OE}$ and $\overline{CE}$ to "0".

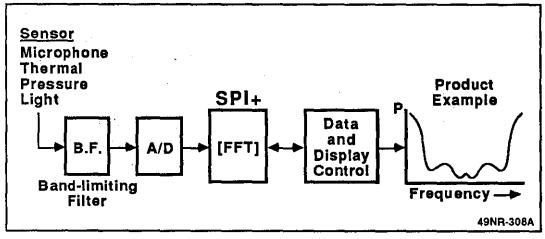
Development Tools

For software development and assembly into object code, a relocatable assembler (RA77C25) is available. This software is available to run on MS-DOS®, CP/M®, VAX™/VMX™, and VAX/UNIX™ systems. For debugging, a hardware emulator (EVAKIT-77C25) provides in-circuit, real-time emulation of the SPI+. Some of the features of the EVAKIT-77C25 include: break/step emulation, symbolic debugging, and on-line assembly/disassembly of code. The EVAKIT-77C25 connects via a probe to your target system for test and demonstration of your final system design. The EVAKIT also connects to your host development system via an RS232 port. Using Kermit or NEC's EVA communications program, code can be down-loaded or up-loaded between development system and EVAKIT. By connecting to a PROM programmer, the EVAKIT is also used to prepare μPD77P25 UVEPROMs which are intended for prototyping and small volume applications. A program adaptor, PA-77P25, is provided for use with the data I/O programmer. Code submittal for the mask ROM μPD77C25 is accomplished by preparing a 27C256A or μPD77P25 PROM using the same programming device.

System Configuration

Figures 15, 16, 17, and 18 show typical system applications for the 77C25/77P25.

Figure 15. Spectrum Analysis System



MS-DOS is a registered trademark of Microsoft Corporation.
CP/M is a registered trademark of Digital Research, Incorporated.
VAX and VMX are trademarks of Digital Equipment Corporation.
UNIX is a trademark of AT&T Bell Laboratories.

Figure 16. Analog-to-Analog Digital Processing System Using a Single SPI+

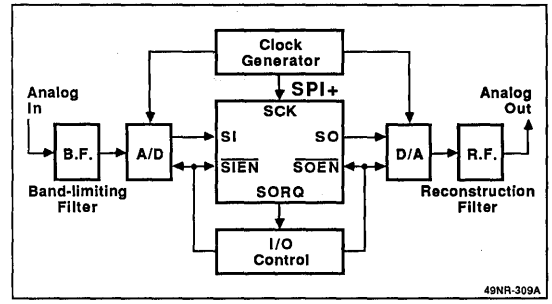


Figure 17. Signal Processing System Using Cascaded SPI+s and Serial Communication

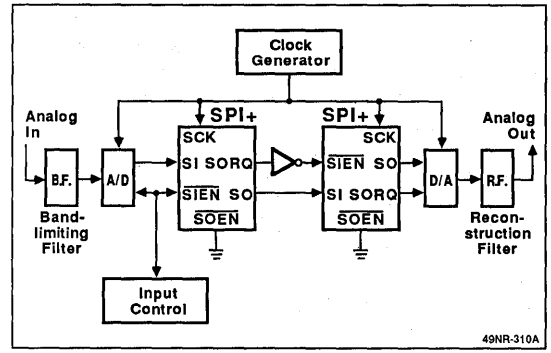
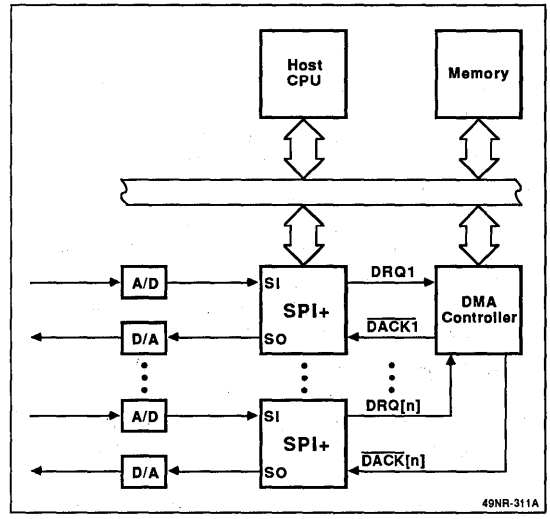


Figure 18. Signal Processing System Using SPI+s as a Complex Computer Peripheral



μPD77220
24-Bit, Fixed-Point Digital Signal Processor

Description

The μPD77220 is a Digital Signal Processor (DSP) developed for digital signal processing requiring high accuracy. This unit processes 24-bit fixed-point data at the rate of 122 ns/instruction.

The internal circuit consists of a multiplier (24 x 24 bits), instruction ROM (2K words x 32 bits), data ROM (1K word x 24 bits), and two independent data RAMs (256 words x 24 bits each).

The μPD77220 has two operation modes: master and slave. These modes can be set using external pins. In master mode, an external 8K-word memory can be added, and 4K words in the memory can be used as an instruction area. In slave mode, the μPD77220 operates as an I/O processor for the host CPU. An external 8K-byte data memory can be added.

Features

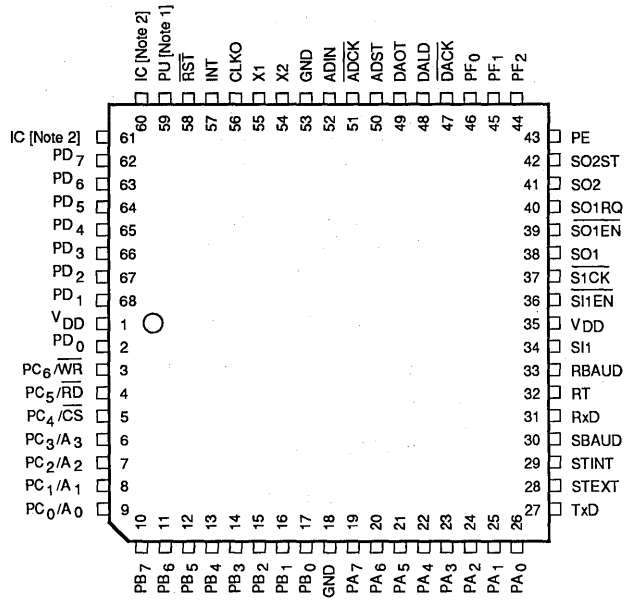
- Processes 24-bit fixed-point data
  - 24-bit fixed-point multiplication circuit
    24 bits x 24 bits → 47 bits
  - 47-bit ALU with eight working registers
  - 47-bit barrel shifter
- High-speed operation and efficient data transfer
  - Instruction cycle 122 ns
  - Three-stage pipeline processing
  - Dedicated data buses in the internal RAM, multiplication circuit, and ALU
- Architecture suitable for digital signal processing
  - Two built-in independent data RAMs and data RAM pointers
  - Each data RAM pointer consists of a base pointer and index register: the base pointer performs a ring count operation in any range
  - Data ROM pointer steps forward in two-step increments (2N) in addition to normal autoincrement/autodecrement addressing
- Flexible external interfaces
- Two modes of operation: master or slave
  - In master mode, 4K word by 32-bit instruction area
  - High-speed access to external memory
    Master mode: 4K words by 24 bits
    Slave mode: 4K words by 8 bits
- CMOS process
- 5 V single power supply
- 68-pin grid array and PLCC packages

Ordering Information

Part Number	Package
μPD77220R	68-pin ceramic PGA
μPD77220L	68-pin PLCC

Pin Configuration

68-Pin PLCC



Notes:

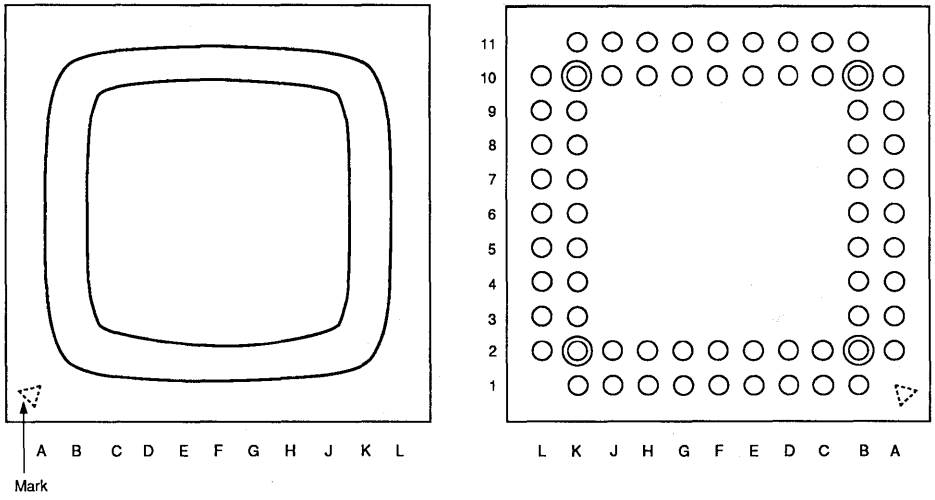
[1] In normal operation connect the Pull Up (PU) pin to the V_{DD} pin through a pull up resistor.

[2] In normal operation the Internal Connection (IC) pin must be open.

83ML5819B

Pin Configuration (cont)

68-Pin Ceramic PGA



Terminal No.	Terminal Mnemonic: Master Mode	Terminal Mnemonic: Slave Mode	Terminal No.	Terminal Mnemonic: Master Mode	Terminal Mnemonic: Slave Mode	Terminal No.	Terminal Mnemonic: Master Mode	Terminal Mnemonic: Slave Mode	Terminal No.	Terminal Mnemonic: Master Mode	Terminal Mnemonic: Slave Mode
A_2	A_7		B_9	D_{13}	I/O_5	F_{10}	D_{23}	I/O_{15}	K_4	V_{DD}	
A_3	A_9		B_{10}	D_{14}	I/O_6	F_{11}	NC		K_5	$\overline{SOEN}$	
A_4	A_{10}		B_{11}	D_{16}	I/O_8	G_1	D_7		K_6	GND	
A_5	A_{AX}		C_1	A_4		G_2	D_6		K_7	$\overline{WR}$	
A_6	D_8	I/O_0	C_2	A_3		G_{10}	D_{24}	$\overline{HRD}$	K_8	V_{DD}	
A_7	D_{10}	I/O_2	C_{10}	D_{17}	I/O_9	G_{11}	D_{25}	$\overline{HWR}$	K_9	X2	
A_8	D_{11}	I/O_3	C_{11}	D_{18}	I/O_{10}	H_1	D_5		K_{10}	D_{30}	P_2
A_9	D_{12}	I/O_4	D_1	A_2		H_2	D_4		K_{11}	D_{31}	P_3
A_{10}	D_{15}	I/O_7	D_2	A_1		H_{10}	D_{26}	$\overline{CS}$	L_2	D_0	
A_1	A_6		D_{10}	D_{19}	I/O_{11}	H_{11}	D_{27}	RQM	L_3	SOCK	
B_2	A_5		D_{11}	D_{20}	I/O_{12}	J_1	D_3		L_4	SO	
B_3	A_8		E_1	A_0		J_2	D_2		L_5	SI	
B_4	GND		E_2	$\overline{M/S}$		J_{10}	D_{28}	P_0	L_6	$\overline{SIEN}$	
B_5	AI1		E_{10}	D_{21}	I/O_{13}	J_{11}	D_{29}	P_1	L_7	$\overline{RESET}$	
B_6	V_{DD}		E_{11}	D_{22}	I/O_{14}	K_1	D_1		L_8	$\overline{RD}$	
B_7	D_9	I/O_1	F_1	$\overline{INT}$		K_2	SORQ		L_9	CLKOUT	
B_8	GND		F_2	$\overline{NMI}$		K_3	SICK		L_{10}	X1	

83ML-5981 B

μPD77220 and μPD77230A Comparison

The μPD77220 is a 24-bit fixed-point signal processor; the μPD77230 is a 32-bit floating-point signal processor. The two processors are generally compatible on an object level. However, the following μPD77230A instructions are not available on the μPD77220.

- ADDF, SUBF, NORM, CVT (OP field)
- TRNORM, RDNORM, FLTFIX, FIXMA (CNT field)
- SPIE, IESP (CNT field)
- WRBEL8, WRBL8E (CNT field)
- JEV0, JEV1 (C field)
- TRE (DST field)

Also, the CMP instruction on the μPD77220 treats data as 47-bit fixed-point data at the time of comparison (as opposed to 55-bit floating-point data on the μPD77230A).

Internal memory differences between the two processors are shown in Table 1. Table 2 describes the differences in the data lengths between the μPD77220 and the μPD77230A.

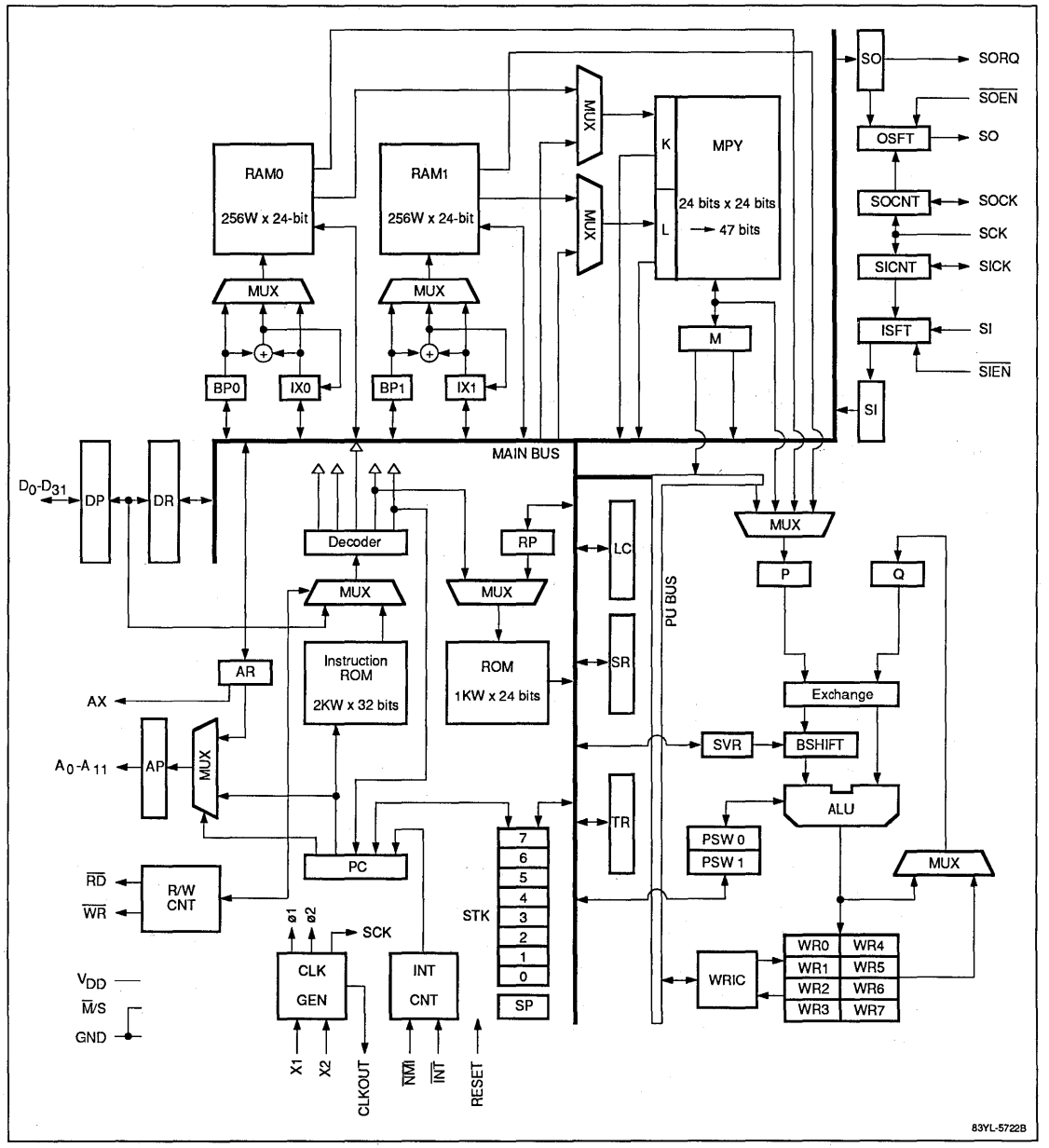
Table 1. Internal Memory Differences between μPD77220 and μPD77230A

Memory Type	μPD77220	μPD77230A
Instruction ROM	2K words x 32 bits	2K words x 32 bits
Data ROM	1K words x 24 bits	1K words x 32 bits
RAM 0	256 x 24 bits	512 x 32 bits
RAM 1	256 x 24 bits	512 x 32 bits

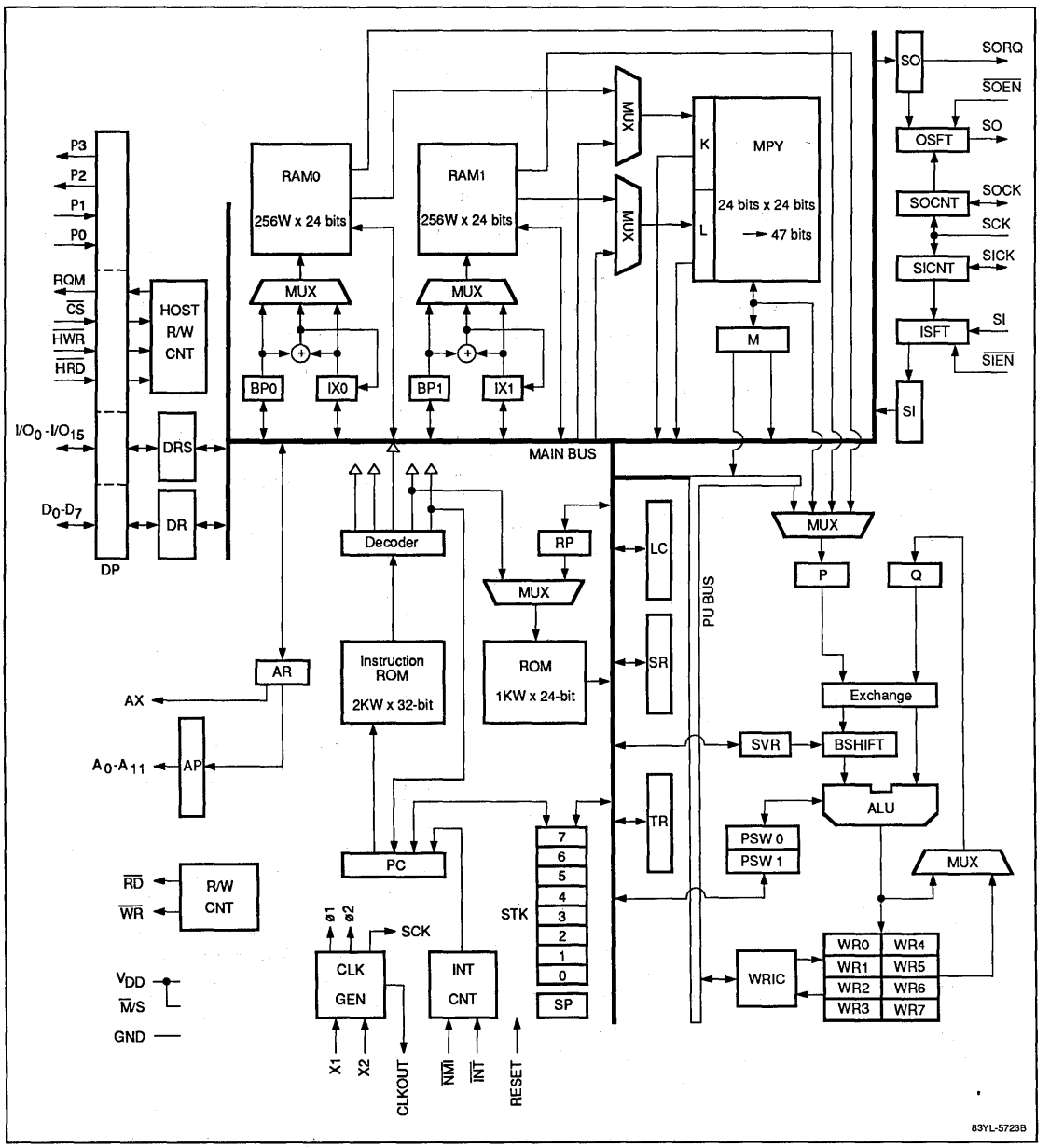
Table 2. Data Length Differences between μPD77220 and μPD77230A

Item	μPD77220	μPD77230A
MAIN BUS	24 bits	32 bits
P, Q	47 bits	55 bits
PSW0, PSW1	4 bits (OVFE not present)	5 bits
RAM0, RAM1	24 bits	32 bits
IX0, IX1	9 bits	←
RP	10 bits	←
M	47 bits	55 bits
DRS	32 bits	←
SI, SO	32 bits	←
LC	10 bits	←
TR	24 bits	32 bits
PU BUS	47 bits	55 bits
WR0 - WR7	47 bits	55 bits
SVR	7 bits	←
BP0, BP1	9 bits	←
ROM	24 bits	32 bits
K, L	24 bits	32 bits
DR	32 bits	←
AR	13 bits	←
STK	13 bits	←
SR	20 bits	←

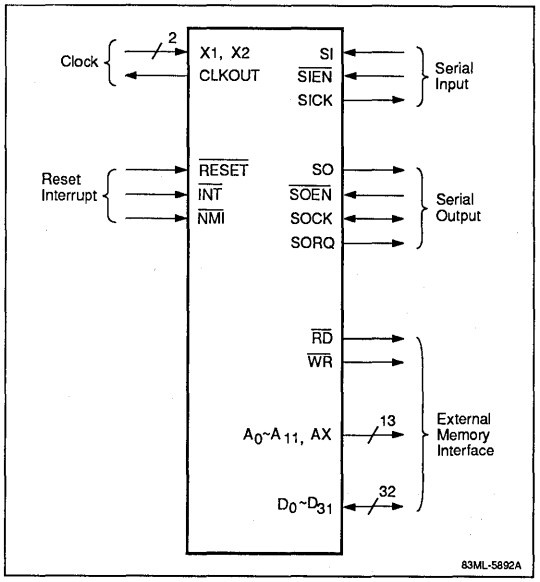
Master Mode Block Diagram



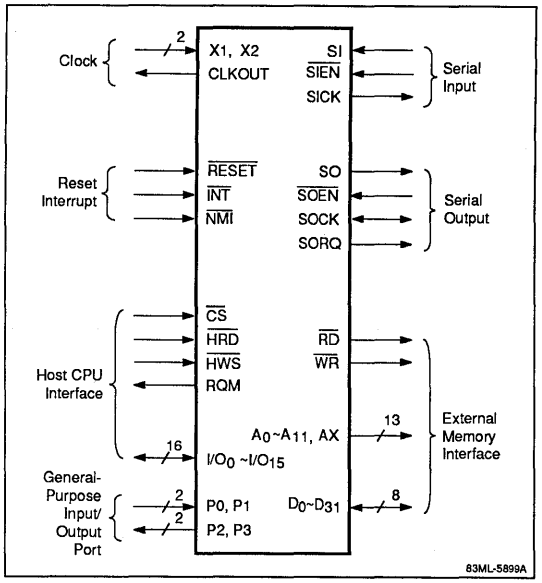
Slave Mode Block Diagram



Master Mode Operation



Slave Mode Operation



Pin Functions

Symbol	PGA Pin Location	PLCC Pin Number	I/O	Function
Power Supply				
V_{DD}	B6	36	—	+5 V power supply Be sure to connect these three pins
	K4	66	—	
	K8	6	—	
GND	B4	40	—	Ground terminals Be sure to ground these three pins
	B8	32	—	
	K6	2	—	
Setting Modes				
$\overline{M}$/S	E2	51	I	Operation mode; mode cannot be changed during operation 0: Master mode 1: Slave mode
Clocks				
X1	L10	9	I	Input pins for crystal oscillator connection If an external clock is used, connect it to the X1 pin and leave X2 open
X2	K9	8	—	
CLKOUT	L9	7	O	μPD77220 internal system clock output. The output signal frequency is half the frequency of the crystal oscillator connected to the X1 or X2 pin

Pin Functions (cont)

Symbol	PGA Pin Location	PLCC Pin Number	I/O	Function
Reset and Interrupt				
$\overline{RESET}$	L7	3	I	Internal system reset signal input (low-level active) — Requires latitude of more than three system clock (CLKOUT) cycles
$\overline{NMI}$	F2	53	I	Non-maskable interrupt input (low-level active) — Requires latitude of more than three system clock (CLKOUT) cycles — Fall edge detection — The interrupt address is 10H
$\overline{INT}$	F1	52	I	Maskable interrupt input (low-level active) — Requires latitude of more than three system clock (CLKOUT) cycles — Fall edge detection — The interrupt address is 100H
Serial Interfaces				
SOCK	L3	63	I/O	Serial output data clock I/O — Serial data is output synchronously when the clock to be input or output at this pin rises — Whether the external clock is to be input or the internal clock to be output depends on the status register
SORQ	K2	62	O	Serial output request (high-level active) — When output data is in the SO register, set to 1 When output is terminated, set to 0
$\overline{SOEN}$	K5	68	I	Serial output enable (low-level active) — Enables serial data output from the SO pin
SO	L4	65	O (3 state)	Serial data output — Serial data output is synchronized with the leading edge of the SOCK signal
SICK	K3	64	I/O	Serial input data clock I/O — Serial data is latched internally at the trailing edge of the clock input to or output from this terminal — The status register determines whether to input the external clock or to output the internal clock
$\overline{SIEN}$	L6	1	I	Serial input enable (low-level active) — Enables serial data input from the SI pin
SI	L5	67	I	Serial data input — Inputs serial data synchronously when SICK falls

Note: The system clock is an internal clock generated by CLK GEN on the basis of the clock (master clock hereafter) input in X1. Its frequency is half of that of the master clock.

External Memory Interfaces (Master Mode Only)				
$\overline{WR}$	K7	4	O	Write output (low-level active) — Write control output for external memory. If 0 is set, the output address is valid and data is output to data bus (D0 to D31)
$\overline{RD}$	L8	5	O	Read output (low-level active) — Read output control for the external memory. If 0 is set, the output address is valid and data is output to data bus (D0 to D31)
AX	A5	37	O	Highest-order memory address output — If the external instruction memory is accessed (highest-order bit PC12 of the internal program counter is 1), 0 is output — If the external data memory is accessed, the value of the highest-order bit AR12 of the internal address register is output 0: High-speed access area 1: Low-speed access area

Pin Functions (cont)

Symbol	PGA Pin Location	PLCC Pin Number	I/O	Function
External Memory Interfaces (Master Mode Only) (cont)				
A0 - A11	See PGA pin configuration diagram	See PLCC pin configuration diagram	O (3 state)	Memory address output — Address output when the external memory is accessed — If the external instruction memory is accessed, the value of low-order 12 bits of the internal program counter is output — If the external data memory is accessed, the value of low-order 12 bits of the address register is output
D0 - D31	See PGA pin configuration diagram	See PLCC pin configuration diagram	I/O (3 state)	32-bit data bus for the external memory
Host CPU Interfaces (Slave Mode Only)				
$\overline{CS}$	H10	15	I	Chip select input (low-level active) — If 0 is set, read/write from host CPU through 16-bit data bus (I/O0 to I/O15) is enabled
$\overline{HWR}$	G11	16	I	Host CPU write input (low-level active) — If 0 is set, 16-bit data bus (I/O0 to I/O15) is ready for input (also $\overline{CS}$ = 0)
$\overline{HRD}$	G10	17	I	Host CPU read input (low-level active) — If 0 is set, 16-bit data bus (I/O0 to I/O15) is ready for output (for $\overline{CS}$ = 0)
I/O0 to I/O15	See PGA pin configuration diagram	See PLCC pin configuration diagram	I/O (3 state)	16-bit data buses for host CPU — Bidirectional buses that input and output data according to control signals $\overline{CS}$, $\overline{HWR}$, and $\overline{HRD}$ from the host CPU — 16-bit or 32-bit I/O data transfer format can be set in the internal status register
RQM	H11	14	O	Host request input — Signal that indicates a read or write request to host CPU
External Data Memory Interfaces (Slave Mode Only)				
$\overline{WR}$	K7	4	O	Write data output (low-level active) — Write control output for the external memory. If set to 0, the output address is valid and data is output to data buses (D0 to D7)
$\overline{RD}$	L8	5	O	Read data output (low-level active) — Read control output for the external memory. If set to 0, the output address is valid and data is input through data buses (D0 to D7)
AX	A5	37	O	Highest-order memory address output — If the external memory is accessed, the value of the highest-order bit AR12 of the internal address register is output 0: High-speed access area 1: Low-speed access area
A0 - A11	See PGA pin configuration diagram	See PLCC pin configuration diagram	O	Memory address output — Address output when the external memory is accessed. The value of the low-order 12 bits of the internal address resister is output from this address
D0 - D7	See PGA pin configuration diagram	See PLCC pin configuration diagram	I/O (3 state)	8-bit data bus for external memory — 1-byte, 2-byte, 3-byte, or 4-byte I/O data transfer format can be set in the internal status register

Pin Functions (cont)

Symbol	PGA Pin Location	PLCC Pin Number	I/O	Function
General-Purpose I/O Ports (Slave Mode Only)				
P0, P1	J10, J11	13, 12	I	General-purpose input port — The status of these general-purpose input ports can be determined by an instruction
P2, P3	K10, K11	11,10	O	General-purpose output port — Data to be output from these general-purpose output pins can be set using an instruction; the data is stored unless the set value is changed

Internal Functions

Mnemonic	Multiplier	Description
Multiplier Peripheral Circuits		
MPY	Multiplier	24-bit fixed-point data multiplier
		24 bits x 24 bits → 47 bits
K	K Register	MPY input data storage register (24 bits)
L	L Register	MPY input data storage register (24 bits)
M	M Register	MPY multiplication result storage register (47 bits)
ALU Peripheral Circuits		
ALU	Arithmetic Logic Unit	47-bit data logical operation circuit
P	P Register	ALU input data storage register (47 bits)
Q	Q Register	ALU input data storage register (47 bits)
EXCHANGE	Data Exchanger	Selects P or Q from which the fixed-point data is to be input to the barrel shifter
BSHIFT	Barrel Shifter	Barrel shifter for fixed-point data in the P or Q register
SVR	Shift Value Register	Shift value set register
WRIC	Working Register Interface Circuit	Specifies the format of data transfer between the working register and PU bus
WR0 - WR7	Working Register (0-7)	ALU operation result storage register (47 bits)
PSW0	Program Status Word 0	ALU operation result status register
PSW1	Program Status Word 1	ALU operation result status register
Data Memory Peripheral Circuits		
ROM	Data ROM	Fixed-data storage ROM (1 kW x 24 bits)
RP	ROM Pointer	Register specifying ROM address (10 bits)
RAM0	Data RAM0	Data storage RAM0 (256 W x 24 bits)
BP0	Base Pointer 0	Register specifying RAM0 base address (9 bits)
IX0	Index Register 0	Register specifying RAM0 index address (9 bits)
RAM1	Data RAM1	Data storage RAM1 (256 W x 24 bits)
BP1	Base Pointer 1	Register specifying RAM1 base address (9 bits)
IX1	Index Register 1	Register specifying RAM1 index address (9 bits)
Instruction ROM Peripheral Circuits		
INSTRUCTION ROM	Instruction ROM	Instruction storage ROM (2 kW x 32 bits)
PC	Program Counter	Register specifying instruction ROM address (13 bits)
STK	Stack	8-level 13-bit stack

Internal Functions (cont)

Mnemonic	Multiplier	Description
Instruction ROM Peripheral Circuits (cont)		
SP	Stack Pointer	Pointer indicating stack address
DECODER	Instruction Decoder	Instruction decoding circuit
Parallel Interface Buses		
DP	Data Port	**Master mode:** — 32-bit parallel data bus for the external memory
		Slave mode: — 8-bit parallel data bus for the external data memory — 16-bit parallel data bus for host CPU — Read/write control signal for host CPU — General-purpose I/O port
AP	Address Port	**Master mode:** — Address bus for the external memory
		Slave mode: — Address bus for the external data memory
DR	Data Register	**Master mode:** — Register for interface between mode DP and internal data bus (main bus) (32 bits)
		Slave mode: — Register for interface between mode DP (8-bit parallel data bus for the external data memory) and main bus (32 bits)
DRS	Data Register for Slave	**Slave mode:** — Register for interface between mode DP (16-bit parallel data bus for host CPU) and main bus (32 bits)
AR	Address Register	Register specifies external data memory address (13 bits)
HOST R/W CNT	Host CPU Read/Write Control Circuit	Slave Host CPU interface control mode circuit
R/W CNT	Read/Write Control Circuit	External memory read/write control circuit
Serial Input/Output Interfaces		
SO	Serial Output Data Register	Serial output data storage register (32 bits)
OSFT	Output Shift Register	Shift register - outputs SO data serially
SOCNT	Serial Output Control Circuit	Serial output control circuit
SI	Serial Input Data Register	Serial input data storage register (32 bits)
ISFT	Input Shift Register	Shift register - inputs serial data
SICNT	Serial Input Control Circuit	Serial input control circuit
Control Circuits		
CLK GEN	Clock Generator	Circuit for generating internal system clock and serial I/O clock
INT CNT	Interrupt Controller	Internal interrupt control circuit
TR	Temporary Register	General-purpose register (24 bits)
LC	Loop Counter	Register which sets program loop count (10 bits)
SR	Status Register	Register which specifies or indicates operation mode (20 bits)

Absolute Maximum Ratings

T_A = +25°C

Operating temperature, T_{OPT}	−10 to +70°C
Storage temperature, T_{STG}	−65 to +150°C
Output voltage, V_O	−0.5 to V_{DD} +0.5 V
Input voltage, V_I	−0.5 to V_{DD} +0.5 V
Power supply voltage, V_{DD}	−0.5 to +6.5 V

Exposure to Absolute Maximum Ratings for extended periods may affect device reliability; exceeding the ratings could cause permanent damage.

Capacitance

T_A = +25°C; V_{DD} = 0 V

Parameter	Symbol	Max	Unit	Conditions
Input capacitance	C_{IN}	10	pF	f_c = 1 MHz
Output capacitance	C_{OUT}	20	pF	

Recommended Operating Conditions

Parameter	Symbol	Min	Typ	Max	Unit
Power supply voltage	V_{DD}	4.75	5.0	5.25	V
Low-level input voltage	V_{IL}	−0.3		0.8	V
High-level input voltage	V_{IH}	2.2		V_{DD} +0.3	V
Low-level X1 input voltage	V_{ILX}	−0.3		0.5	V
High-level X1 input voltage	V_{IHX}	3.9		V_{DD} +0.3	V
Operating temperature	T_{OPT}	−10	+25	+70	°C

DC Characteristics

T_A = −10 to +70°C; V_{DD} = 5 V ± 5%

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
Low-level output voltage	V_{OL}			0.45	V	I_{OL} = 2.0 mA
High-level output voltage	V_{OH}	0.7 V_{DD}			V	I_{OH} = −400 μA
Low-level input current	I_{IL}			−400	μA	$\overline{RESET}$, SICK, SOCK VIN = 0 V
High-level input current	I_{IH}			400	μA	$\overline{M}$/S V_{IN} = V_{DD}
Low-level input leak current	I_{LIL}			−10	μA	Except $\overline{RESET}$, SICK, SOCK, V_{IN} = 0 V
High-level input leak current	I_{LIH}			10	μA	Except $\overline{M}$/S, V_{IN} = V_{DD}
Low-level output leak current	I_{LOL}			−10	μA	V_{OUT} = 0 V
High-level output leak current	I_{LOH}			10	μA	V_{OUT} = V_{DD}
X1 input current	I_{IX1}			400	μA	X1 pin, external clock input
Power supply current	I_{DD}		140	200	mA	f_{CYX} = 16.384 MHz

Crystal Oscillator Connection Conditions

T_A = −10 to +70°C; V_{DD} = 5 V ± 5%

Parameter	Mnemonic	Min	Typ	Max	Unit	Conditions
Oscillation frequency	f_{CYX}	1.0	16.384	16.667	MHz	See figure 1.
C1, C2 capacity			15		pF	

Timing Requirements

T_A = −10 to +70°C; V_{DD} = 5 V ± 5%

Parameter	Mnemonic	Min	Typ	Max	Unit	Conditions
X1 cycle time	t_{CYX}	60	61	1000	ns	See figure 2. See figure 3; switching characteristics timing measurement voltage = 1.0 and 3.0 V
X1 high pulse width	t_{XXH}	25			ns	
X1 low pulse width	t_{XXL}	25			ns	
X1 rise time	t_{XR}			5	ns	
X1 fall time	t_{XF}			5	ns	
SICK, SOCK cycle time	t_{CYS}	240	244		ns	
SICK, SOCK high pulse width	t_{SSH}	100			ns	
SICK, SOCK low pulse width	t_{SSL}	100			ns	
SICK, SOCK rise time	t_{SR}			20	ns	
SICK, SOCK fall time	t_{SF}			20	ns	

Switching Characteristics

T_A = −10 to +70°C; V_{DD} = 5 V ± 5%; C_L = 100 pF

Item	Mnemonic	Min	Max	Unit
X1 ↑ → $\overline{RD}$ delay time	t_{DXC}		50	ns
X1 ↑ → CLKOUT hold time	t_{HXC}	0		ns
SCK cycle time	t_{CYS}	$8t_{CYX}$		ns
SCK high pulse width	t_{SSH}	$4t_{CYX}$−65		ns
SCK low pulse width	t_{SSL}	$4t_{CYX}$−65		ns
SCK rise time	t_{SR}		20	ns
SCK fall time	t_{SF}		20	ns
X1 ↑ → SCK ↑ delay time	t_{DXS}	10	120	ns

Figure 1. Oscillation Circuit Diagram

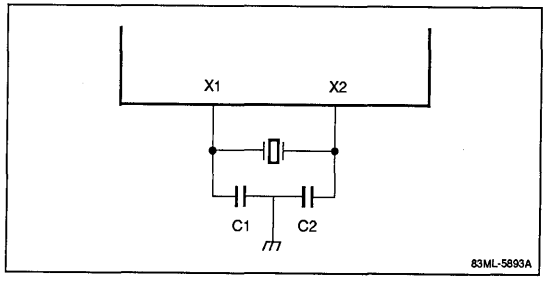


Figure 2. External Clock Connection Diagram

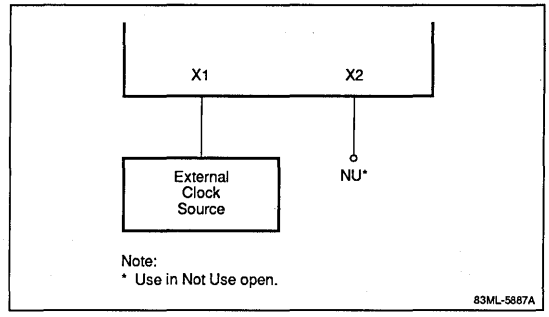
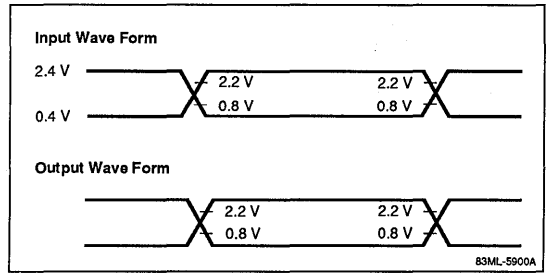
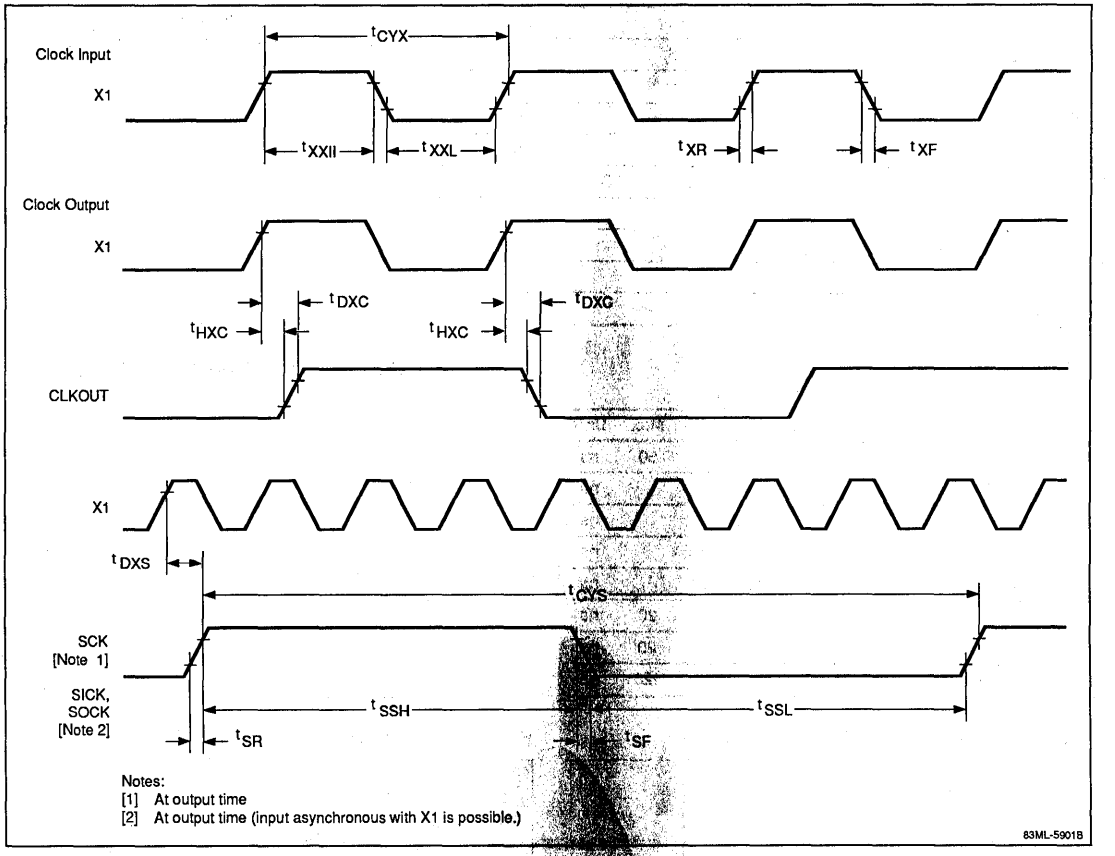


Figure 3. Switching Characteristics



Timing Waveforms

Clock Input/Output



External Memory Access Timing

T_A = −10 to +70°C; V_{DD} = 5 V ± 5%

Item	Mnemonic	Min	Max	Unit	Conditions
Data set time (for address)	t_{SADI}		$2t_{CYX}$−85	ns	When an instruction is read
Data set time (for $\overline{RD}$ ↓)	t_{SRDI}		t_{CYX}−25	ns	
Data hold time (for $\overline{RD}$ ↑)	t_{HRDI}	0		ns	
Data set time (for address)	t_{SAD1}		$4t_{CYX}$−135	ns	Applied to high-speed access area
	t_{SAD2}		$8t_{CYX}$−135	ns	Applied to low-speed access area
Data set time (for $\overline{DR}$ ↓)	t_{SRD1}		$3t_{CYX}$−75	ns	Applied to high-speed access area
	t_{SRD2}		$7t_{CYX}$−75	ns	Applied to low-speed access area
Data hold time (for $\overline{RD}$ ↑)	t_{HRD}	0		ns	

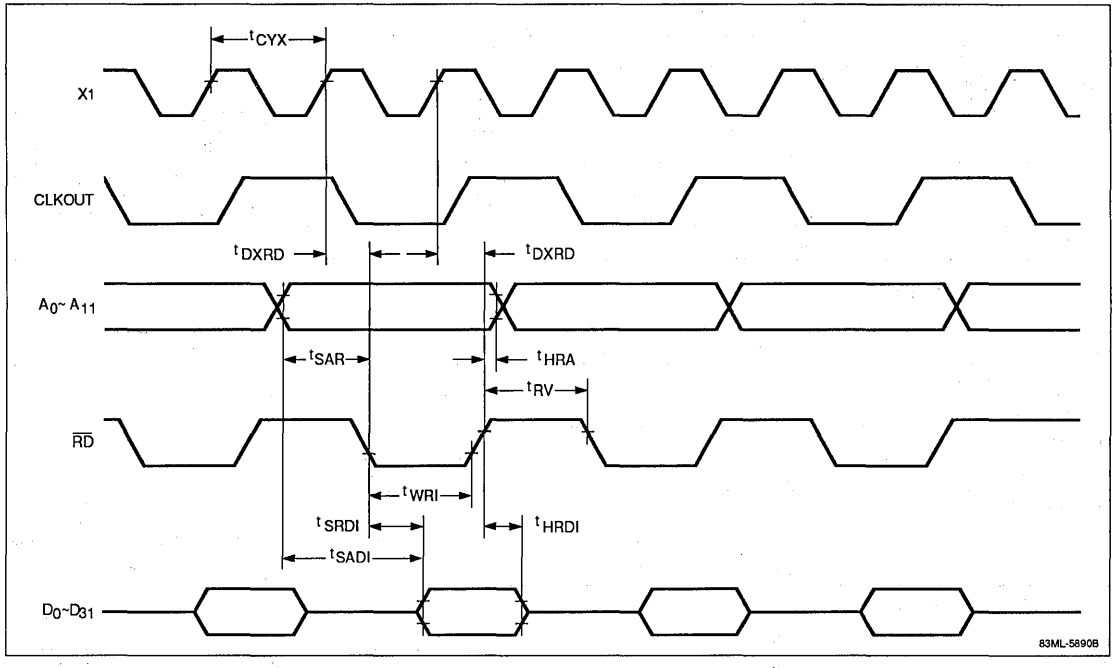
Switching Characteristics

T_A = −10 to +70°C; V_{DD} = 5 V ± 5%; C_L = 100 pF

Item	Mnemonic	Min	Max	Unit	Conditions
X1 ↑ → $\overline{RD}$ delay time	t_{DXRD}		55	ns	
X1 ↑ → $\overline{WR}$ delay time	t_{DXWR}		55	ns	
Address set time (for $\overline{RD}$ ↓)	t_{SAR}	t_{CYX}−50		ns	
Address hold time (for $\overline{RD}$ ↑)	t_{HRA}	5		ns	
$\overline{RD}$ low-level width	t_{WR1}	t_{CYX}−20		ns	When an instruction is read
	t_{WR2}	$3t_{CYX}$−30		ns	Applied to high-speed access area
	t_{WR3}	$7t_{CYX}$−30		ns	Applied to low-speed access area
Address set time (For $\overline{WR}$ ↓)	t_{SAW}	t_{CYX}−45		ns	
Address hold time (for $\overline{WR}$ ↑)	t_{HWA}	5		ns	
$\overline{WR}$ low-level width	t_{WW1}	$3t_{CYX}$−50		ns	Applied to high-speed access area
	t_{WW2}	$7t_{CYX}$−50		ns	Applied to low-speed access area
Data set time (for $\overline{WR}$ ↑)	t_{SDW1}	$3t_{CYX}$−100		ns	Applied to high-speed access area
	t_{SDW2}	$7t_{CYX}$−100		ns	Applied to low-speed access area
$\overline{WR}$ ↓ → data delay time	t_{DWD}	0		ns	
Data float time (for $\overline{WR}$ ↑)	t_{FWD}	10	50	ns	
$\overline{RD}$, $\overline{WR}$ recovery time	t_{RV}	t_{CYX}−30		ns	At time of continuous operation

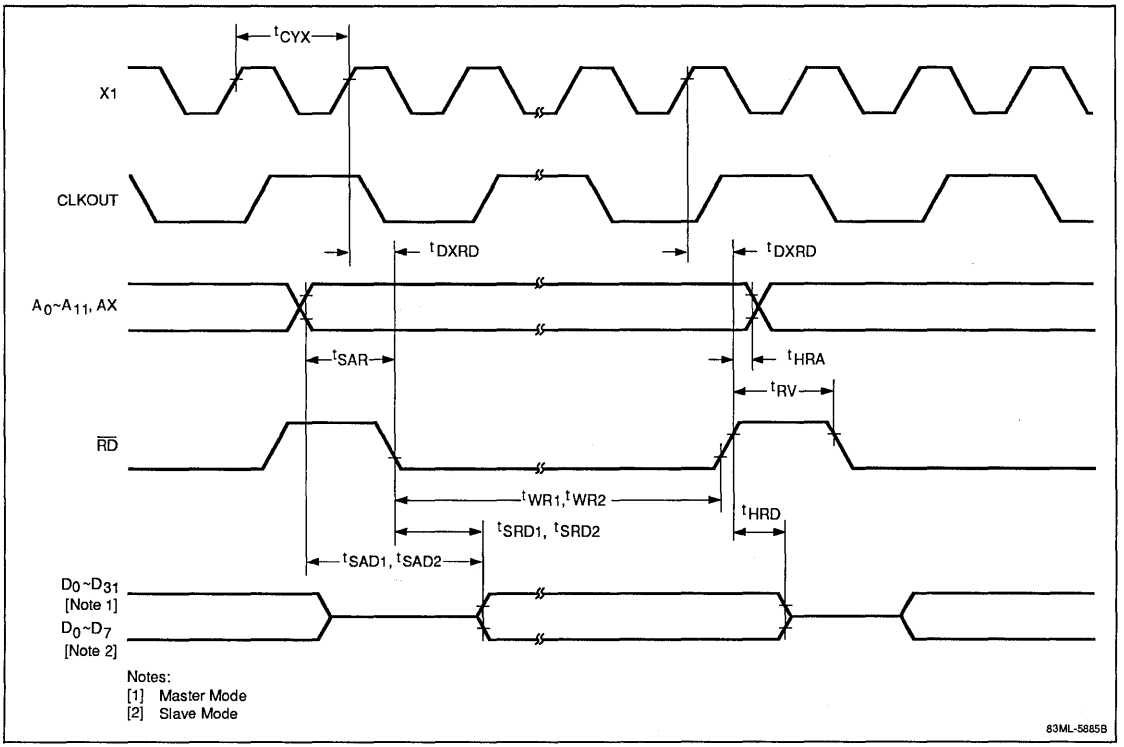
Timing Waveforms (cont)

Instruction Read Operation (Master Mode Only)



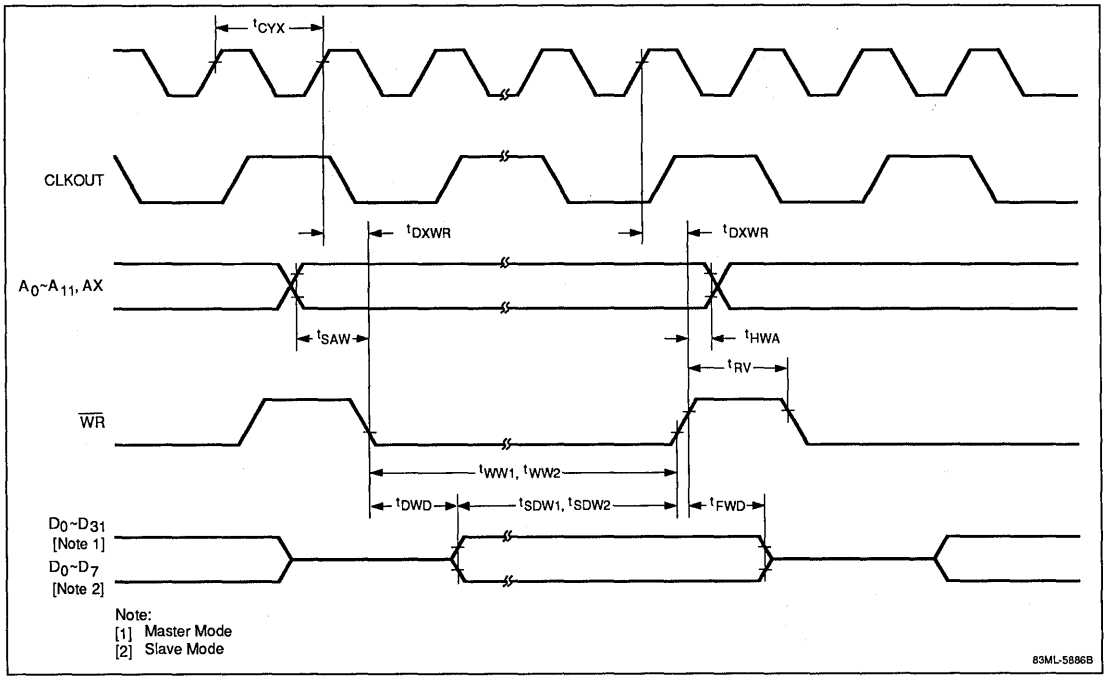
Timing Waveforms (cont)

Data Read Operation

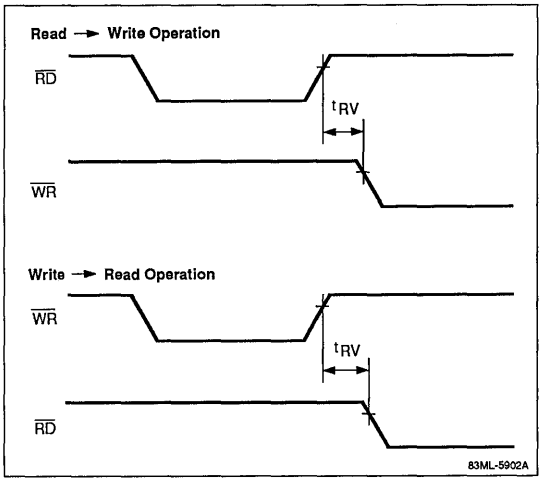


Timing Waveforms (cont)

Data Write Operation



Data Read/Write Operation



Host Interface Timing (Slave Mode)

T_A = −10 to +70°C; V_{DD} = 5 V ± 5%

Item	Mnemonic	Min	Unit
$\overline{CS}$ set time (for $\overline{HRD}$ ↓)	t_{SCR}	0	ns
$\overline{CS}$ hold time (for $\overline{HRD}$ ↑)	t_{HRC}	0	ns
$\overline{HRD}$ low-level width	t_{WHRD}	150	ns
$\overline{CS}$ set time (for $\overline{HWR}$ ↓)	t_{SCW}	0	ns
$\overline{CS}$ hold time (for $\overline{HWR}$ ↑)	t_{HWC}	0	ns
$\overline{HWR}$ low-level width	t_{WHWR}	150	ns
Data set time (for $\overline{HWR}$ ↓)	t_{SIHW}	100	ns
Data hold time (for $\overline{HHR}$ ↑)	t_{HHWI}	0	ns
$\overline{HRD}$, $\overline{HWR}$ recovery time	t_{HRV}	100	ns
$\overline{HRD}$, $\overline{HWR}$ hold time (for RQM ↑)	t_{HRH}	t_{CYX}	ns
P0, P1 set time (for X1 ↑)	t_{SPX}	t_{CYX}	ns
P0, P1 hold time (for X1 ↑)	t_{HXP}	t_{CYX}	ns

Switching Characteristics

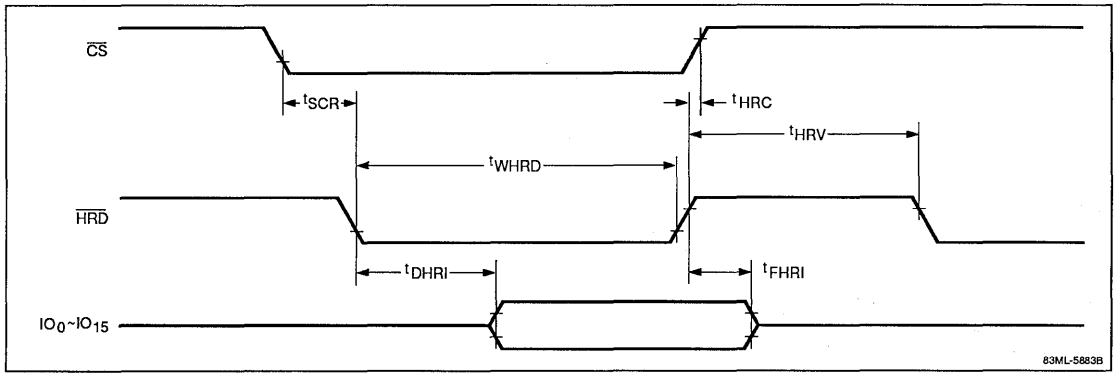
T_A = −10 to +70°C; V_{DD} = 5 V ± 5%; C_L = 100 pF

Item	Mnemonic	Min	Max	Unit
$\overline{HRD}$ ↓ → data delay time	t_{DHRI}		100	ns
$\overline{HRD}$ ↓ → data float time	t_{FHRI}	10	65	ns
X1 ↑ → RQM ↑ delay time	t_{DXRH}		100	ns
X1 ↑ → RQM ↓ delay time	t_{DXRL}		100	ns
$\overline{HRD}$, $\overline{HWR}$ ↑ → RQM ↓ delay time	t_{DHR}		$2t_{CYX}+100$	ns
X1 ↑ → P2, P3 delay time	t_{DXP}		100	ns

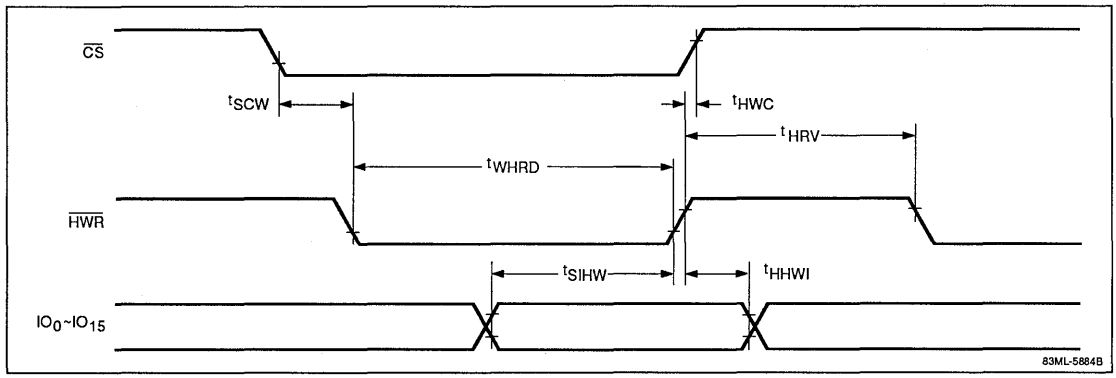
2

Timing Waveforms (cont)

Host Read Operation

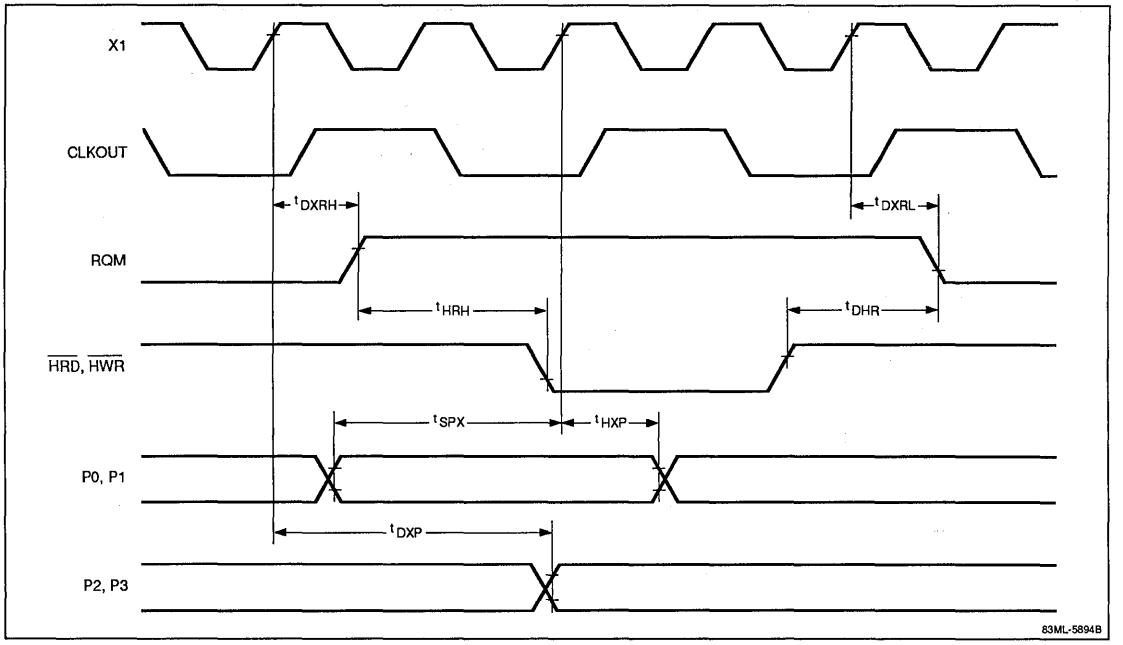


Host Write Operation



Timing Waveforms (cont)

RQM Port



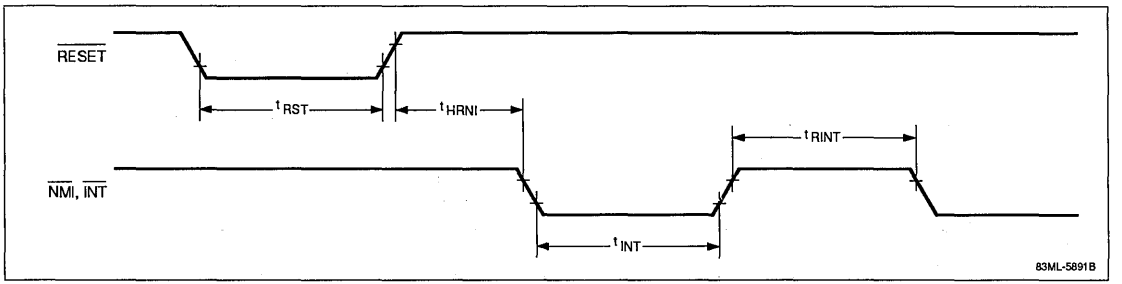
Interrupt Reset Timing

T_A = −10 to +70°C; V_{DD} = 5 V ± 5%

Item	Mnemonic	Min	Unit
$\overline{RESET}$ low-level width	t_{RST}	$6t_{CYX}$	ns
$\overline{NMI}$, $\overline{INT}$ hold time (for $\overline{RESET}$ ↑)	t_{HRNI}	$6t_{CYX}$	ns
$\overline{NMI}$, $\overline{INT}$ low-level width	t_{INT}	$6t_{CYX}$	ns
$\overline{NMI}$, $\overline{INT}$ recovery time	t_{RINT}	$6t_{CYX}$	ns

Timing Waveforms (cont)

Interrupt Reset Timing Chart



Serial Interface Timing

T_A = −10 to +70°C; V_{DD} = 5 V ± 5%

Item	Mnemonic	Min	Unit
$\overline{SIEN}$, SI set time (for SCK ↓)	t_{SSIS}	55	ns
$\overline{SIEN}$, SI hold time (for SCK ↓)	t_{HSSI}	30	ns
$\overline{SOEN}$ set time (for SCK ↑)	t_{SSES}	50	ns
$\overline{SOEN}$ hold time (for SCK ↑)	t_{HSSE}	30	ns
$\overline{SIEN}$, $\overline{SOEN}$ recovery time	t_{SRV}	t_{CYS}	ns

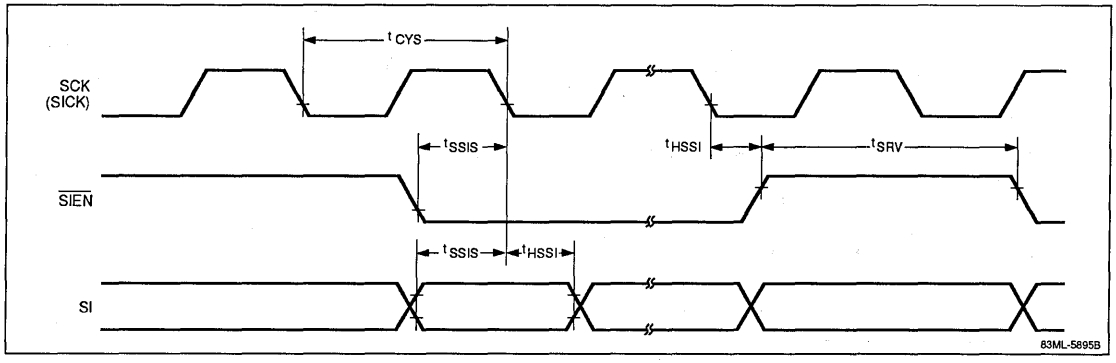
Switching Characteristics

T_A = −10 to +70°C; V_{DD} = 5 V ± 5%; C_L = 100 pF

Item	Mnemonic	Min	Max	Unit
SCK ↓ → SORQ ↑ delay time	t_{DSSQ}	30	150	ns
$\overline{SOEN}$ ↓ → SO delay time	t_{DSESO}		60	ns
$\overline{SOEN}$ ↑ → SO float time	t_{FSESO}	10	100	ns
SCK ↑ → SO delay time	t_{DSHSO}		60	ns
SCK ↓ → SO hold time	t_{HSHSO}	0		ns
SCK ↓ → SO delay time	t_{DSLSO}		60	ns
Switching Characteristics				
SCK ↓ → SO float time (at time of SORQ ↓)	t_{FSSO}	10	100	ns

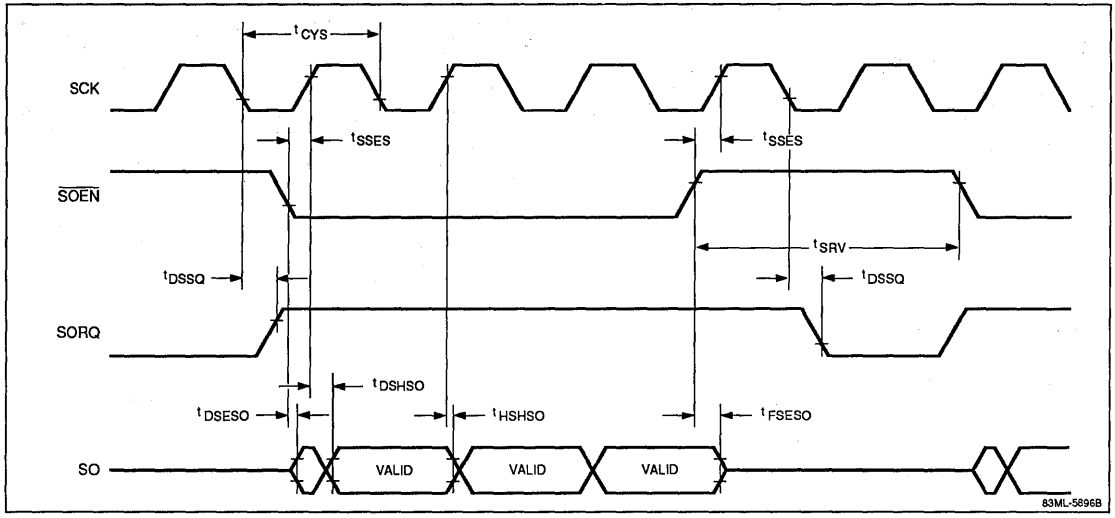
Timing Waveforms (cont)

Serial In

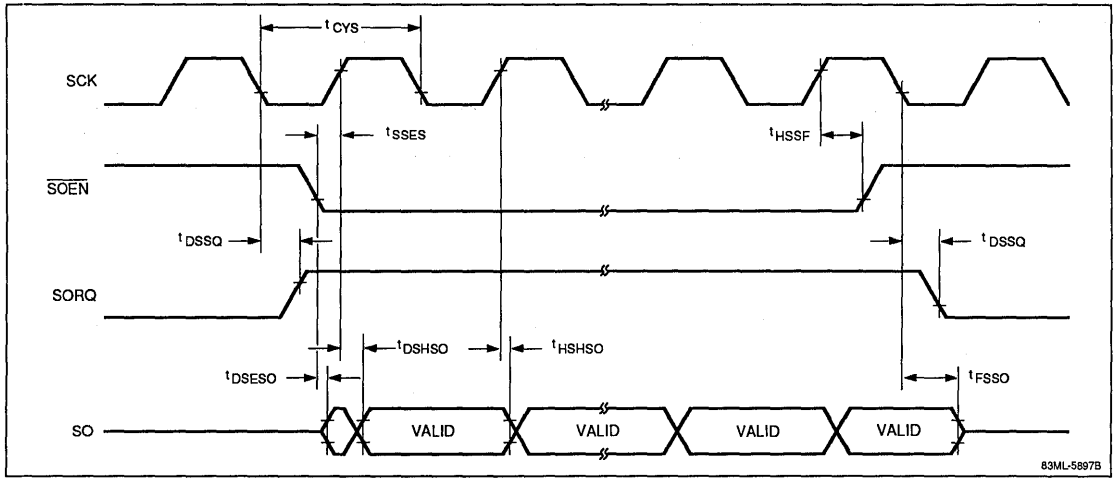


Timing Waveforms (cont)

Serial OUT 1 (SOEN Interrupt Control)

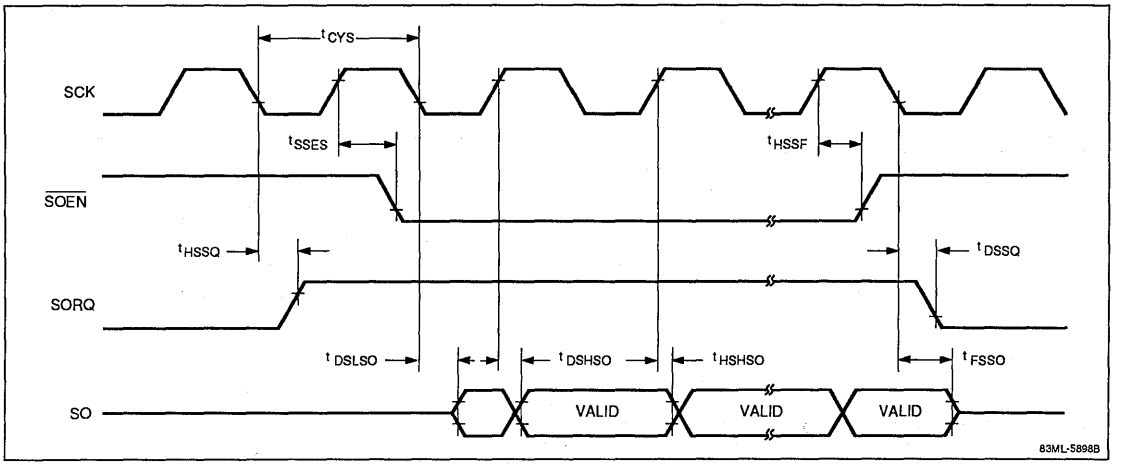


Serial OUT 2 (SOEN Control: SOEN Low AT SCK is Low Level)



Timing Waveforms (cont)

Serial OUT 3 (SOEN Control: SOEN Low AT SCK is High Level)



Data Format

The μPD77220 can process fixed-point data. Data is represented by a 2's complement, and the highest-order bit of fixed-point data indicates the sign. See figure 4. Table 3 shows the 24-bit fixed-point data format. Table 4 shows the 47-bit fixed point data.

Numeric data is processed in fixed-point data format, and the decimal point is positioned between the sign bit and the following bit.

Figure 4. Fixed-Point Data Format

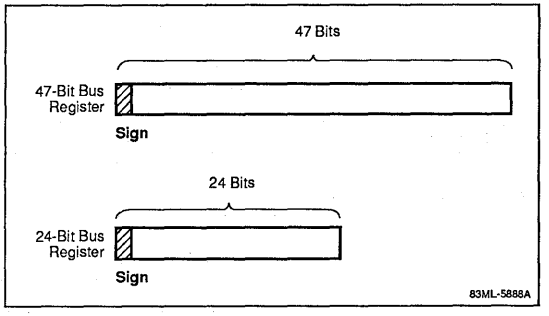


Table 3. 24-Bit Fixed-Point Internal Data Format

Value	Binary Notation	Hexadecimal Notation	Conversion to Decimal Number
Maximum Positive Value	0111 1111	$7FFFFF_H$	$1.0 - 2^{-23} \approx 1.0$
	0111 1110	$7FFFFF_H$	$1.0 - 2^{-22}$
	:	:	:
	0100 0000	400000_H	$1.0 - 2^{-1} = 0.5$
	:	:	:
Minimum Positive Value	0000 0001	000001_H	$2^{-23} \approx 1.2 \times 10^{-7}$
Zero	0000 0000	000000_H	0.0
Maximum Negative Value	1111 1111	$FFFFFF_H$	$-(2^{-23}) \approx -1.2 \times 10^{-7}$
	:	:	:
	1100 0000	$C00000_H$	$-(2^{-1}) = -0.5$
	:	:	:
	1000 0001	800001_H	$-1.0 + 2^{-23}$
Minimum Negative Value	1000 0000	800000_H	−1.0

Table 4. 47-Bit Fixed-Point Internal Data Format

Value	Binary Notation	Hexadecimal Notation	Conversion to Decimal Number
Maximum Positive Value	0111 1111	$7FFFFFFFFFFE_H$	$1.0 - 2^{-46} \approx 1.0$
	0111 1110	$7FFFFFFFFFFC_H$	$1.0 - 2^{-45}$
	:	:	:
	0100 0000	400000000000_H	$1.0 - 2^{-1} = 0.5$
	:	:	:
Minimum Positive Value	0000 0001	000000000002_H	$2^{-46} \approx 1.4 \times 10^{-14}$
Zero	0000 0000	000000000000_H	0.0
Maximum Negative Value	1111 1111	$FFFFFFFFFFFE_H$	$-(2^{-46}) \approx -1.4 \times 10^{-14}$
	:	:	:
	1100 0000	$C00000000000_H$	$-(2^{-1}) = -0.5$
	:	:	:
	1000 0001	800000000002_H	$-1.0 + 2^{-46}$
Minimum Negative Value	1000 0000	800000000000_H	−1.0

Conversion of data (47 bits) into hexadecimal format ranges from the highest-order bit (sign bit) to the lowest-order bit sequentially.

Instructions

All μPD77220 instructions consist of a 32-bit word. The instructions fall into three categories:

- Operation instructions
- Branch instructions
- Load instructions

Operation Instructions

An operation (OP) instruction is an ALU operation instruction where 22 different operations may be specified in the upper five bits. Figure 5 shows the bit format.

Pointer modifications may be specified in the CNT field. Transfers may also be specified within the SRC and DST fields of an OP instruction. When all fields are specified in an OP instruction, several different tasks are performed simultaneously.

OP Field. The 5-bit OP field specifies the operation type in the ALU. Table 5 lists the 22 types of operations it may contain.

CNT (Control) Field. The CNT field is 12 bits long and specifies a pointer, flag operation, register switch-over, data transfer format, and loop counter decrement.

The control field bit configuration is shown in figure 6. The field has 22 types of subfields. Table 6 describes the subfields.

Figure 5. Operation Instruction Format

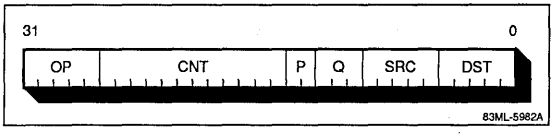


Table 5. OP Field Specifications

Mnemonic	OP Field (31-27)	Operation
NOP	00000	No operation
INC	00001	Increment
DEC	00010	Decrement
ABS	00011	Absolute
NOT	00100	Not
NEG	00101	Negate
SHLC	00110	Shift left with carry for double precision
SHRC	00111	Shift right with carry for double precision
ROL	01000	Rotate left
ROR	01001	Rotate right
SHLM	01010	Shift left multiple (see note)
SHRM	01011	Shift right multiple (see note)
SHRAM	01100	Shift right arithmetic multiple see note)
CLR	01101	Clear
ADD	10000	Add fixed-point data
SUB	10001	Subtract fixed-point data
ADDC	10010	Add fixed-point data with carry
SUBC	10011	Subtract fixed-point data with carry
CMP	10100	Compare
AND	10101	AND
OR	10110	OR
XOR	10111	Exclusive OR

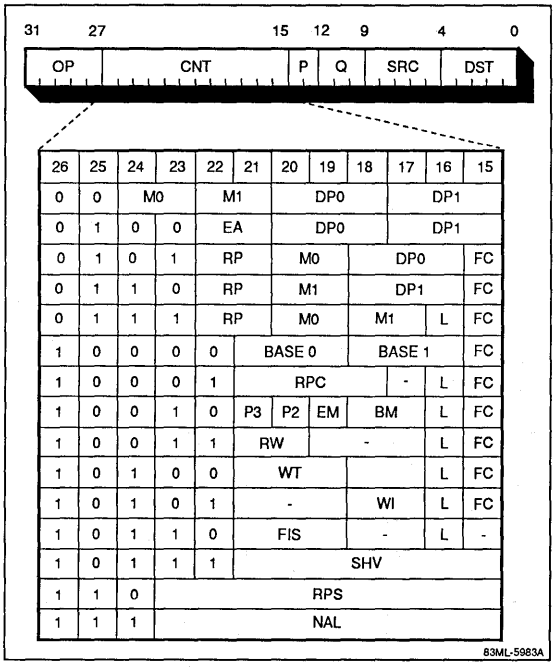
Multiple value is in SVR or specification value of SHV bit.

Table 6. Control Field Specifications

Group	Field	Function	Effective
Interrupt	EM	Enables/disables maskable interrupt	→
	BM	Sets and clears maskable interrupt input flag	→
PSW	FIS	PSW control	*/→
	FC	Switches over PSW0, PSW1	*
ROM pointer	RP	Rules ROM pointer count operations	→
	RPC	Specifies n value in ROM pointer operation	→
	RPS	Specifies low-order nine bits of data ROM address	→
RAM0/RAM1 pointers	M0	Specifies base pointer 0 and index register 0	→
	M1	Specifies base pointer 1 and index register 1	→
	DP0	Rules count operations of base pointer 0 and index register 0	→
	DP1	Rules count operations of base pointer 1 and index register 1	→
	BASE0	Specifies counter length of modulo counter base pointer 0	→
	BASE1	Specifies counter length of modulo counter counter base pointer 1	→
Data format conversion	WI	Specifies transfer format when working register is specified in the DST field	→
	WT	Specifies transfer format when working register is specified in the SRC field	→
Shift specification	SHV	Specifies amount of shift for 47-bit fixed-point data	*
Data memory access	RW	Specifies input/output operation for external memory	*
	EA	Address register increment and decrement	*/→
General-purpose output port	P2	Controls signal output of pins with the same name	→
	P3	Controls signal output of pins with the same name	→
Loop counter	L	Loop counter decrement	*
Jump	NAL	Specifies unconditional jump address	*

* Effective starting with current instruction.
→ Effective starting with the next instruction.

Figure 6. CNT Field Bit Configuration



26	25	24	23	22	21	20	19	18	17	16	15
0	0	M0		M1		DP0			DP1		
0	1	0	0	EA		DP0			DP1		
0	1	0	1	RP		M0		DP0			FC
0	1	1	0	RP		M1		DP1			FC
0	1	1	1	RP		M0		M1		L	FC
1	0	0	0	0	BASE 0			BASE 1			FC
1	0	0	0	1	RPC				-	L	FC
1	0	0	1	0	P3	P2	EM	BM		L	FC
1	0	0	1	1	RW			-		L	FC
1	0	1	0	0	WT					L	FC
1	0	1	0	1	-			WI		L	FC
1	0	1	1	0	FIS			-		L	-
1	0	1	1	1	SHV						
1	1	0	RPS								
1	1	1	NAL								

P Field. The 2-bit P field (bits 14, 13) specifies the source of input to the register, which is used as an input to the ALU for operations requiring two operands. The internal data bus, MPY output, RAM0, or RAM1 can be specified. Table 7 shows the field specifications.

Table 7. P Field Specifications

Mnemonic	Bit 14	Bit 13	Input of P Register
IB	0	0	PU bus
M4	0	1	Multiplier output register (MPY output)
RAM0	1	0	RAM block 0
RAM1	1	1	RAM block 1

Q Field. The 3-bit Q field (bits 12-10) specifies the source of input to the Q register, which is the second of two ALU input registers.

One of the working registers, WR0 to WR7, must be specified in the Q field. The result of the operation is placed in the working register specified in the Q field. Table 8 provides the Q field specifications.

Table 8. Q Field Specifications

Mnemonic	Bit 12	Bit 11	Bit 10	Register
WR0	0	0	0	Working register 0
WR1	0	0	1	Working register 1
WR2	0	1	0	Working register 2
WR3	0	1	1	Working register 3
WR4	1	0	0	Working register 4
WR5	1	0	1	Working register 5
WR6	1	1	0	Working register 6
WR7	1	1	1	Working register 7

SRC (Source) Field. The 5-bit SRC field (bits 9-5) holds the source register for a transfer instruction. Table 9 lists the 32 types of registers that may be specified in this field.

Table 9. SRC Field Specifications

Mnemonic	SRC Field (9-5)	Selected Source Register
NON	00000	Non-selection
RP	00001	ROM pointer
PSW0	00010	Program status word 0
PSW1	00011	Program status word 1
SVR	00100	SVR (shift value register)
SR	00101	Status register
LC	00110	Loop counter
STK	00111	Stack
M	01000	M register
ML	01001	Low 24 bits of M register
ROM	01010	Data ROM
TR	01011	Temporary register
AR	01100	Address register
SI	01101	Serial input register
DR	01110	Data register
DRS	01111	Data register for slave
WR0	10000	Working register 0
WR1	10001	Working register 1
WR2	10010	Working register 2
WR3	10011	Working register 3
WR4	10100	Working register 4
WR5	10101	Working resister 5
WR6	10110	Working register 6
WR7	10111	Working register 7
RAM0	11000	RAM 0
RAM1	11001	RAM 1
BP0	11010	Base pointer 0
BP1	11011	Base pointer 1
IX0	11100	Index register 0
IX1	11101	Index register 1
K	11110	K register
L	11111	L register

DST (Destination) Field. The DST field (bits 4-0) is 5 bits long and specifies the destination register for a transfer instruction. Table 10 lists the 31 destinations that may be specified in the DST field.

Table 10. DST Field Specifications

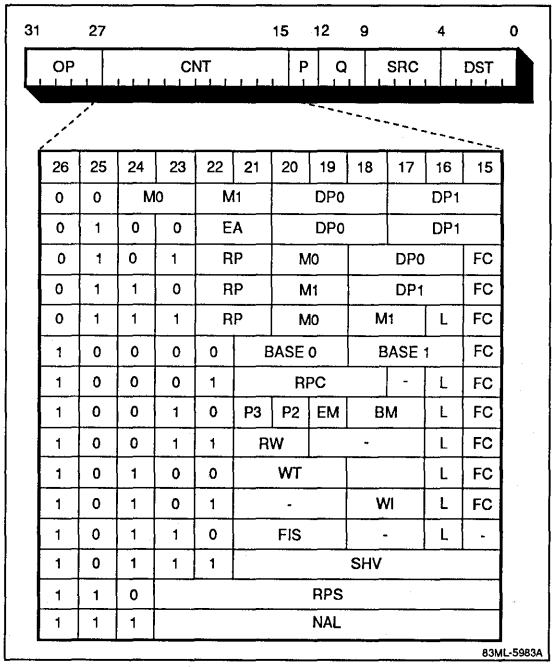
Mnemonic	DST Field (4-0)	Selected Destination Register
NON	00000	Non-selection
RP	00001	ROM pointer
PSW0	00010	Program status word 0
PSW1	00011	Program status word 1
SVR	00100	SVR (shift value register)
SR	00101	Status register
LC	00110	Loop counter
STK	00111	Stack
LKR0	01000	L register (RAM 0 to K register)
KLR1	01001	K register (RAM 1 to L register)
TR	01011	Temporary register
AR	01100	Address register
SO	01101	Serial output register
DR	01110	Data register
DRS	01111	Data register for slave
WR0	10000	Working register 0
WR1	10001	Working register 1
WR2	10010	Working register 2
WR3	10011	Working register 3
WR4	10100	Working register 4
WR5	10101	Working resister 5
WR6	10110	Working register 6
WR7	10111	Working register 7
RAM0	11000	RAM 0
RAM1	11001	RAM 1
BP0	11010	Base pointer 0
BP1	11011	Base pointer 1
IX0	11100	Index register 0
IX1	11101	Index register 1
K	11110	K register
L	11111	L register

Branch Instructions

Branch instructions specify a conditional jump, an unconditional jump, subroutine call, or return. The format of the branch instruction, consisting of five fields, is shown in figure 7.

Note that the SRC and DST fields may be included as part of the branch instruction. This data transfer will take place regardless of any condition upon which a jump may be dependent.

Figure 6. CNT Field Bit Configuration



26	25	24	23	22	21	20	19	18	17	16	15
0	0	M0		M1		DP0			DP1		
0	1	0	0	EA		DP0			DP1		
0	1	0	1	RP		M0		DP0			FC
0	1	1	0	RP		M1		DP1			FC
0	1	1	1	RP		M0		M1		L	FC
1	0	0	0	0	BASE 0			BASE 1			FC
1	0	0	0	1	RPC				-	L	FC
1	0	0	1	0	P3	P2	EM	BM		L	FC
1	0	0	1	1	RW			-		L	FC
1	0	1	0	0	WT					L	FC
1	0	1	0	1	-			WI		L	FC
1	0	1	1	0	FIS			-		L	-
1	0	1	1	1	SHV						
1	1	0	RPS								
1	1	1	NAL								

P Field. The 2-bit P field (bits 14, 13) specifies the source of input to the register, which is used as an input to the ALU for operations requiring two operands. The internal data bus, MPY output, RAM0, or RAM1 can be specified. Table 7 shows the field specifications.

Table 7. P Field Specifications

Mnemonic	Bit 14	Bit 13	Input of P Register
IB	0	0	PU bus
M4	0	1	Multiplier output register (MPY output)
RAM0	1	0	RAM block 0
RAM1	1	1	RAM block 1

Q Field. The 3-bit Q field (bits 12-10) specifies the source of input to the Q register, which is the second of two ALU input registers.

One of the working registers, WR0 to WR7, must be specified in the Q field. The result of the operation is placed in the working register specified in the Q field. Table 8 provides the Q field specifications.

Table 8. Q Field Specifications

Mnemonic	Bit 12	Bit 11	Bit 10	Register
WR0	0	0	0	Working register 0
WR1	0	0	1	Working register 1
WR2	0	1	0	Working register 2
WR3	0	1	1	Working register 3
WR4	1	0	0	Working register 4
WR5	1	0	1	Working register 5
WR6	1	1	0	Working register 6
WR7	1	1	1	Working register 7

SRC (Source) Field. The 5-bit SRC field (bits 9-5) holds the source register for a transfer instruction. Table 9 lists the 32 types of registers that may be specified in this field.

Table 9. SRC Field Specifications

Mnemonic	SRC Field (9-5)	Selected Source Register
NON	00000	Non-selection
RP	00001	ROM pointer
PSW0	00010	Program status word 0
PSW1	00011	Program status word 1
SVR	00100	SVR (shift value register)
SR	00101	Status register
LC	00110	Loop counter
STK	00111	Stack
M	01000	M register
ML	01001	Low 24 bits of M register
ROM	01010	Data ROM
TR	01011	Temporary register
AR	01100	Address register
SI	01101	Serial input register
DR	01110	Data register
DRS	01111	Data register for slave
WR0	10000	Working register 0
WR1	10001	Working register 1
WR2	10010	Working register 2
WR3	10011	Working register 3
WR4	10100	Working register 4
WR5	10101	Working resister 5
WR6	10110	Working register 6
WR7	10111	Working register 7
RAM0	11000	RAM 0
RAM1	11001	RAM 1
BP0	11010	Base pointer 0
BP1	11011	Base pointer 1
IX0	11100	Index register 0
IX1	11101	Index register 1
K	11110	K register
L	11111	L register

DST (Destination) Field. The DST field (bits 4-0) is 5 bits long and specifies the destination register for a transfer instruction. Table 10 lists the 31 destinations that may be specified in the DST field.

Table 10. DST Field Specifications

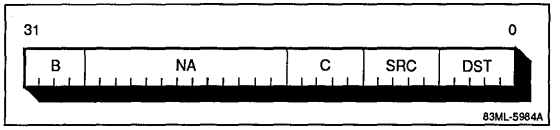
Mnemonic	DST Field (4-0)	Selected Destination Register
NON	00000	Non-selection
RP	00001	ROM pointer
PSW0	00010	Program status word 0
PSW1	00011	Program status word 1
SVR	00100	SVR (shift value register)
SR	00101	Status register
LC	00110	Loop counter
STK	00111	Stack
LKR0	01000	L register (RAM 0 to K register)
KLR1	01001	K register (RAM 1 to L register)
TR	01011	Temporary register
AR	01100	Address register
SO	01101	Serial output register
DR	01110	Data register
DRS	01111	Data register for slave
WR0	10000	Working register 0
WR1	10001	Working register 1
WR2	10010	Working register 2
WR3	10011	Working register 3
WR4	10100	Working register 4
WR5	10101	Working resister 5
WR6	10110	Working register 6
WR7	10111	Working register 7
RAM0	11000	RAM 0
RAM1	11001	RAM 1
BP0	11010	Base pointer 0
BP1	11011	Base pointer 1
IX0	11100	Index register 0
IX1	11101	Index register 1
K	11110	K register
L	11111	L register

Branch Instructions

Branch instructions specify a conditional jump, an unconditional jump, subroutine call, or return. The format of the branch instruction, consisting of five fields, is shown in figure 7.

Note that the SRC and DST fields may be included as part of the branch instruction. This data transfer will take place regardless of any condition upon which a jump may be dependent.

Figure 7. Branch Instruction Format



B Field. This field (bits 31-28) indicates a branch instruction. The value of this field is always 1101.

C Field. This 5-bit field (bits 14-10) indicates the nature of the branch instruction. Table 11 summaries the branch conditions that can be specified.

NA Field. The destination address of the branch is contained in the 13-bit NA field (bits 27-15). Note that the most significant bit of the NA field is used to determine whether the destination address is in internal or external instruction memory.

SRC Field. The SRC field (bits 9-5) specifies a type of source register for a transfer instruction. There are 32 possible types.

DST Field. The DST field (bits 4-0) indicates the type of destination register to be used for a transfer instruction. There are 31 possible types.

Load Instructions

The load instruction consists of three fields as shown in figure 8. This instruction loads 24-bit data specified in the IM field into the register specified in the DST field. The data is input to each register through the main bus.

The value of the LDI field is always 111.

Figure 8. Load Instruction Format

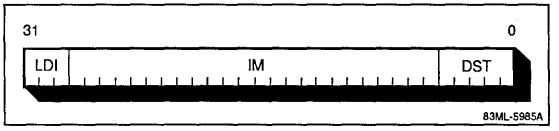


Table 11. Branch Condition Summary

Mnemonic	C Field (14-10)	Jump with Condition
JMP	00000	Jump with no condition
CALL	00001	Subroutine call
RET	00010	Return
JNZRP	00011	Jump if ROM pointer is not zero
JZ0	00100	Jump if zero flag 0 is set
JNZ0	00101	Jump if zero flag 0 is reset
JZ1	00110	Jump if zero flag 1 is set
JNZ1	00111	Jump if zero flag 1 is reset
JC0	01000	Jump if carry flag 0 is set
JNC0	01001	Jump if carry flag 0 is reset
JC1	01010	Jump if carry flag 1 is set
JNC1	01011	Jump if carry flag 1 is reset
JS0	01100	Jump if sign flag 0 is set
JNS0	01101	Jump if sign flag 0 is reset
JS1	01110	Jump if sign flag 1 is set
JNS1	01111	Jump if sign flag 1 is reset
JV0	10000	Jump if overflow flag 0 is set
JNV0	10001	Jump if overflow flag 0 is reset
JV1	10010	Jump if overflow flag 1 is set
JNV1	10011	Jump if overflow flag 1 is reset
JNFSI	10110	Jump if SI register is not full
JNESO	10111	Jump if SO register is not empty
JIP0*	11000	Jump if input port 0 is on
JIP1*	11001	Jump if input port 1 is on
JNZIX0	11010	Jump if index register 0 is nonzero
JNZIX1	11011	Jump if index register 1 is nonzero
JNZBP0	11100	Jump if base pointer 0 is nonzero
JNZBP1	11101	Jump if base pointer 1 is nonzero
JRDY	11110	Jump if ready is on
JRQM*	11111	Jump if request for master is on

* Valid for slave mode only.

μPD77230/77P230
32-Bit, Floating-Point Advanced Signal Processor

Description

The μPD77230 Advanced Signal Processor (ASP) is the high-end member of a new third-generation family of 32-bit digital signal processors. This CMOS chip implements 32-bit full floating-point arithmetic, and is intended for digital signal processing and other applications requiring high speed and high precision.

The μPD77230 has on-chip instruction and data ROM. These ROM areas can be mask ROM (μPD77230AR) or EPROM (μPD77P230R). The mask ROM is also available as a standard part with a standard, general-purpose DSP library (μPD77230AR-003).

All instructions execute in one instruction cycle. The μPD77230 executes a 32-bit by 32-bit floating point multiply with 55-bit product, sum of products, data move, and multiple data pointer manipulations—all in one 150-ns instruction cycle.

Features

- ☐ Fast instruction cycle: 150 ns using 13.3-MHz clock
- ☐ All instructions execute in one cycle
- ☐ 32- x 32-bit floating point arithmetic
- ☐ Large on-chip memory (32-bit words)
  - — 1K data RAM (two 512-word blocks)
  - — 1K data coefficient ROM
  - — 2K instruction ROM
- ☐ 8K- x 32-bit external memory; 4K may be instruction memory
- ☐ 1.5-μm CMOS technology
- ☐ 32-bit internal bus
- ☐ 55-bit ALU bus
- ☐ Dedicated internal buses for RAM, multiplier, and ALU
- ☐ Eight accumulators/working registers (55 bits)
- ☐ 47-bit bidirectional barrel shifter
- ☐ Two independent data RAM pointers
- ☐ Modulo 2^n incrementing for circular RAM buffers
- ☐ Base and index addressing of internal RAM
- ☐ Data ROM capable of 2^n incrementing
- ☐ Loop counter for repetitive processing
- ☐ Eight-level stack accessible to internal bus
- ☐ Two interrupts: maskable and nonmaskable (NMI)
- ☐ Serial I/O (4 MHz)
- ☐ Master/slave mode operation
- ☐ Three-stage instruction pipeline
- ☐ Single +5-volt power supply
- ☐ Approximately 1.2 watts

Ordering Information

Part Number	ROM	Package Type
μPD77230AR	Mask ROM	68-pin ceramic PGA
μPD77230AR-003	Mask ROM (Standard library)	
μPD77P230R	EPROM	

Applications

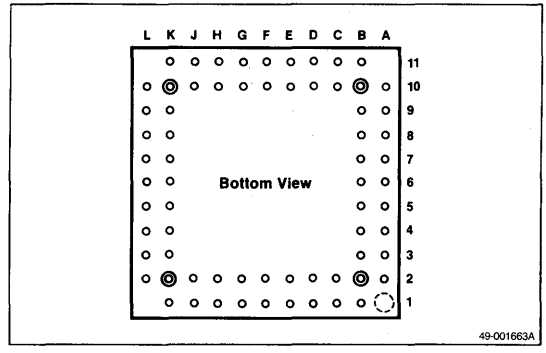
- ☐ General-purpose digital filtering (FIR, IIR, FFT)
- ☐ High-speed data modems
- ☐ Adaptive equalization (CCITT)
- ☐ Echo cancelling
- ☐ High-speed controls
- ☐ Image processing
- ☐ Graphic transformations
- ☐ Instrumentation electronics
- ☐ Numerical processing
- ☐ Speech processing
- ☐ Sonar/radar signal processing
- ☐ Waveform generation

Floating-Point Performance Benchmarks

Second-order digital filter (biquad)	0.9 μs
32-tap finite impulse response filter	5.25 μs
Fast Fourier transform (FFT)	
32-point complex (radix 2)	0.15 ms
512-point complex FFT	4.7 ms
1024-point complex FFT	11.78 ms
4096-point complex FFT	69.51 ms
Square root	6.0 μs

Pin Configuration

68-Pin Ceramic PGA



Pin Identification

No.	Master	*Slave	No.	Master	*Slave
A2	A_7		F10	D_{23}	I/O_{15}
A3	A_9		F11	NC (No connection)	
A4	A_{10}		G1	D_7	
A5	A_X		G2	D_6	
A6	D_8	I/O_0	G10	D_{24}	$\overline{HRD}$
A7	D_{10}	I/O_2	G11	D_{25}	$\overline{HWR}$
A8	D_{11}	I/O_3	H1	D_5	
A9	D_{12}	I/O_4	H2	D_4	
A10	D_{15}	I/O_7	H10	D_{26}	$\overline{CS}$
B1	A_6		H11	D_{27}	RQM
B2	A_5		J1	D_3	
B3	A_8		J2	D_2	
B4	GND		J10	D_{28}	P0
B5	A_{11}		J11	D_{29}	P1
B6	V_{DD}		K1	D1	
B7	D_9	I/O_1	K2	SORQ	
B8	GND		K3	SICK	
B9	D_{13}	I/O_5	K4	V_{DD}	
B10	D_{14}	I/O_6	K5	$\overline{SOEN}$	
B11	D_{16}	I/O_8	K6	GND	
C1	A_4		K7	$\overline{WR}$	
C2	A_3		K8	V_{DD}	
C10	D_{17}	I/O_9	K9	X2	
C11	D_{18}	I/O_{10}	K10	D_{30}	P2
D1	A_2		K11	D_{31}	P3
D2	A_1		L2	D_0	
D10	D_{19}	I/O_{11}	L3	SOCK	
D11	D_{20}	I/O_{12}	L4	SO	
E1	A_0		L5	SI	
E2	M/S		L6	$\overline{SIEN}$	
E10	D_{21}	I/O_{13}	L7	$\overline{RESET}$	
E11	D_{22}	I/O_{14}	L8	$\overline{RD}$	
F1	$\overline{INT}$		L9	CLKOUT	
F2	$\overline{NMI}$		L10	X1	

*If slave-mode pin identification is not specified, it is the same as master-mode.

Pin Function Summary

Symbol	I/O	Function
A_0-A_{11}	O	Address bus to external memory
A_X	O	Highest bit of memory address
CLKOUT	O	Internal system clock
$\overline{CS}$	I	Chip select
D_0-D_7	I/O*	Data bus for access to external memory in slave mode.
D_0-D_{31}	I/O*	Data bus for access to external memory (data or instruction) in master mode.
GND		Ground (Connect ground to all GND pins.)
$\overline{HRD}$	I	Host CPU read
$\overline{HWR}$	I	Host CPU write
I/O_0-I/O_{15}	I/O*	Port to host CPU data bus
$\overline{INT}$	I	Maskable interrupt
$\overline{NMI}$	I	Nonmaskable interrupt
M/S	I	Operation mode select
P0, P1	I	General-purpose input port
P2, P3	O	General-purpose output port
$\overline{RD}$	O	Controls data read from external memory
$\overline{RESET}$	I	System reset
RQM	O	Data read/write request
SI	I	Serial input data
SICK	I/O	Clock for serial input data
$\overline{SIEN}$	I	Serial input data enable
SO	O*	Serial output data
SOCK	I/O	Clock for serial output data
$\overline{SOEN}$	I	Serial output data enable
SORQ	O	Serial output request
V_{DD}		+5-volt power (Connect +5 V to all V_{DD} pins.)
$\overline{WR}$	O	Controls data write to external memory
X1, X2	I	External clock (X1) or crystal (X1, X2)

*These pins have a high-impedance inactive state.

Pin Functions

Paragraphs below supplement the brief descriptions in the preceding table. Pin symbols are in alphabetical order within several master and slave mode categories.

Master and Slave Modes

CLKOUT [System Clock]. Outputs internal system clock. Output signal frequency is half the oscillation frequency of crystal connected across X1 and X2 pins.

$\overline{INT}$ [Maskable Interrupt]. Inputs maskable interrupt signal, which is active-low and must be at least three system clock pulses wide. Interrupt signal is detected at falling edge. Interrupt address is 100H.

M/S [Mode Select]. Selects operation mode. Operation mode must not be switched during operation, however. Master = 0; slave = 1.

$\overline{NMI}$ [Nonmaskable Interrupt]. Inputs nonmaskable interrupt signal, which is active-low and must be at least three system clock pulses wide. Interrupt signal is detected at falling edge. Interrupt address is 10H.

$\overline{RESET}$ [System Reset]. Inputs internal system reset signal, which is active-low and must be at least three system clock pulses wide.

SI [Serial Input Data]. Inputs serial data synchronized with falling edge of SICK.

SICK [Serial Input Clock]. Inputs or outputs clock for serial input data. Serial data is internally latched at the falling edge of the clock that is input to or output from this pin. Whether the clock is to be input from an external source or the internal clock is to be output is determined by the status register setting.

$\overline{SIEN}$ [Serial Input Enable]. Enables SI pin to input serial data. This pin is active-low.

SO [Serial Output Data]. Outputs serial data synchronized with rising edge of SOCK pin. When inactive, this pin becomes high impedance.

SOCK [Serial Output Clock]. Inputs or outputs clock for serial output data. The serial output data is synchronized with the clock that is input to or output from this pin. Whether the clock is to be input from an external source or the internal clock is to be output is determined by the status register setting.

$\overline{SOEN}$ [Serial Output Enable]. Enables SO pin to output serial data. This pin is active-low.

SORQ [Serial Output Request]. Outputs serial output request signal, which is active-high. When data is ready in the serial output register, this signal becomes 1. It will become 0 after data has been output.

X1, X2 [External Clock]. Connection to external oscillator crystal (X1, X2) or external clock (X1).

Master Mode, External Memory Interface

A_0-A_{11} [Address Bus]. Address bus for access to external memory. When accessing external instruction memory, the lower 12 bits of the program counter are output to these pins. When accessing external data memory, the lower 12 bits of the external address register are output to these pins.

A_X [Highest Address Bit]. Outputs the highest bit of the memory address. When accessing external instruction memory, the highest bit of the program counter (PC_{12}) is output to this pin. When accessing external data memory, the highest bit of the external address register is output to this pin. High-speed memory area = 0; low-speed memory area = 1.

D_0-D_{31} [Data Bus]. These pins form a 32-bit, three-state data bus for external memory (data or instruction).

$\overline{RD}$ [Data Read]. Controls data read from external memory. This signal becomes 0 after the output address is valid, and data is input at the rising edge to the data port formed by pins D_0 to D_{31}.

$\overline{WR}$ [Data Write]. Controls data write to external memory. This signal becomes 0 after the output address is valid and data is output to the data port formed by pins D_0 to D_{31}.

Slave Mode, External Memory Interface

A_0-A_{11} [Address Bus]. Address bus for accessing external memory. When accessing external data memory, the lower 12 bits of the external address register are output to these pins.

A_X [Highest Address Bit]. When accessing external data memory, the highest bit of the external address register is output to this pin. High-speed memory area = 0; low-speed memory area = 1.

D_0-D_7 [Data Bus]. These pins form an 8-bit, three-state data bus for external data memory access. Data may be transferred in one of four formats (1-, 2-, 3-, or 4-byte words), depending on the status register setting.

$\overline{RD}$ [Data Read]. Controls data read from external memory. This signal becomes 0 after the output address is valid, and data is input at the rising edge to the data port formed by pins D_0 to D_7.

$\overline{WR}$ [Data Write]. Controls data write to external memory. This signal becomes 0 after the output address is valid and data is output to the data port formed by pins D_0 to D_7.

Slave Mode, Host CPU Interface

$\overline{CS}$ [Chip Select]. Active-low chip select input signal. When this pin becomes 0, the host CPU may perform read/write operations on the 16-bit port formed by pins I/O_0 through I/O_{15}.

$\overline{HRD}$ [Host CPU Read]. Active-low host read input signal. In conjunction with CS, this signal allows the host CPU to read data from the DRS register via the 16-bit port formed by pins I/O_0 to I/O_{15}.

$\overline{HWR}$ [Host CPU Write]. Active-low host write input signal. In conjunction with CS, this signal allows the host CPU to write data into the DRS register via the 16-bit port formed by pins I/O_0 to I/O_{15}.

I/O_0-I/O_{15} [Data Port]. These pins form an I/O port to the host CPU bidirectional data bus. It is used for input to or output from the DRS register under control of host CPU signals $\overline{CS}$, $\overline{HWR}$, and $\overline{HRD}$. Data transfer format can be specified in the status register as either a 16-bit or a 32-bit transfer.

RQM [Read/Write Request]. Requests host CPU to read or write data via the host CPU data bus.

Slave Mode, I/O Port

P0, P1 [Input Port]. These pins form a general-purpose input port. Status of either of these pins may be tested by a conditional branch instruction.

P2, P3 [Output Port]. These pins form a general-purpose output port. Data output by these pins can be set directly by an instruction and will be retained until explicitly changed.

Functional Description

Figure 1 is the functional block diagram of the μPD77230 in its master mode configuration. The main internal bus (32 bits) ties together all the functional blocks of the μPD77230, including the ALU area. The 55-bit processing unit (PU) bus links the ALU input to the 55-bit multiplier output register and the eight 55-bit working registers. Thus, the full 55 bits of precision can be maintained during extensive calculations.

In addition to the main bus and the PU bus, there is a sub-bus linking each of the two RAM areas to both the ALU input and the multiplier input registers. This allows simultaneous loading of the multiplier input registers in parallel with ALU operations and in parallel with data transfer operations, which make use of the main bus. There is a sub-bus connecting the ALU input to the 55-bit multiplier output and another sub-bus that can route the working registers' contents back to the ALU input.

Architecture

The μPD77230 has a Harvard-type architecture, with structions are executed in a single cycle, even if the instruction is stored in the external instruction memory expansion area.

Instruction Memory

The μPD77230 has an internal instruction ROM that holds 2K 32-bit instruction words. An additional 4K word external memory expansion is also available. A 13-bit program counter (PC) contains the current instruction address; the most significant bit of the PC determines whether on-chip or external instructions are to be fetched. An eight-level stack holds subroutine and interrupt return addresses, and it is accessible to/from the main internal bus.

Data Memory

The data ROM area on the μPD77230 holds 1K 32-bit words. The ROM pointer (RP) contains the current ROM address, which can also be specified within an instruction field. The ROM pointer has auto-increment and auto-decrement features and an add 2^n to the RP option.

There are two separate and independently addressable data RAM areas, each 512 words by 32 bits. Each RAM area can be addressed by a base register, an index register, or the sum of the two. The base register and/or the index register may be incremented, decremented, or cleared. In addition, the base pointer can operate in a modulo count mode, and the index register contents may be replaced by the sum of the index and base registers.

Data memory may be expanded by the addition of 8K words of external memory. External data memory is divided into a high-speed half, which is accessed in two instruction cycles, and a low-speed half, which is accessed in four instruction cycles. Both high-speed and low-speed memory accesses occur in parallel with normal program execution.

Multiplier and ALU

The floating point multiplier has two 32-bit input registers, called the K and L registers, which are accessible both to and from the main bus. The multiplier produces the 55-bit product of the K and L register contents automatically in a single instruction cycle (there is no multiply instruction). The 55-bit result is stored in the M register in 8-bit exponent, 47-bit mantissa format. The contents of the M register can be transferred to the main bus (32 bits) or to the ALU via the processing unit bus (55 bits). The multiplier consists of a 24- by 24-bit fixed-point multiplier and an

exponent adder, so that it can also be used for fixed-point multiplications.

The 55-bit floating-point ALU is capable of a full set of arithmetic and logical operations (see Instruction Set section). There is a 47-bit bidirectional barrel shifter, which can perform general-purpose shifting in addition to the mantissa alignments required for floating-point arithmetic. A separate exponent ALU (EALU) determines shift values in floating-point work. The ALU status is reflected in one of two identical processor status words (PSW) that contain carry, zero, sign, and overflow flags. The results of the ALU operation are stored in one of eight 55-bit accumulators or "working registers."

There are two 55-bit input registers to the ALU called the P register and the Q register. The Q register input is selected from one of the eight working registers, while the P register input is selected from among the 32-bit main bus, data RAM 0, data RAM 1, and the 55-bit M register.

A loop counter is included in the design of the μPD77230. This loop counter is a 10-bit register, attached to the main bus, which can be decremented by a control bit built into an ordinary ALU instruction. When the loop counter is decremented to zero, the instruction following the one that decremented it will be skipped.

System Control

The master system clock may be provided to the μPD77230 via either an external crystal or an already available clock signal. The internal clock of the μPD77230 contains two phases, and is obtained by dividing the master clock frequency by 2. If desired, the serial input and output clocks can be derived from the master clock by dividing it by 8.

Both a maskable and nonmaskable interrupt are available in the μPD77230. The maskable interrupt can be "memorized," so that if an interrupt occurs while it is in the interrupt disabled condition, then it may be acted upon (or disregarded) at a later time. The status of the interrupts and other aspects of the μPD77230 are determined by or reflected in the 20-bit status register.

Serial I/O

The serial input and output circuitry in the μPD77230 is designed for easy interfacing to codecs and other μPD77230s. The input and output circuits are independently clocked by either an internal clock or an external clock up to 4 MHz. The length of the serial input and output data words can be independently programmed to be 8, 16, 24, or 32 bits.

The parallel I/O capabilities in the μPD77230 can be used for external instruction and data memory expansion and for interaction with a host processor. The difference between master mode and slave mode operation must be defined to further discuss the nature of the parallel interface in the μPD77230.

Master/Slave Modes

The master mode parallel interface is shown in figure 1. In this mode, the μPD77230 is intended to act as a standalone processor with the parallel interface allowing access to external memory, memory-mapped I/O devices, and/or a system-level bus. Master mode operation allows for external instruction memory expansion and external data memory expansion. There is an 8K external memory space. The lower 4K can be shared between instructions and data, while the upper 4K can be used for data only.

The slave mode parallel interface is shown in figure 2. In this mode, the μPD77230 is a "peripheral" to a host processor. The full 8K external memory space is available for data memory expansion, but instruction memory expansion is not allowed in slave mode. The 8-bit external data bus is used to assemble words in the data register (DR), which can be 8, 16, 24 or 32 bits wide. Communication with the host occurs across the 16-bit host data bus. Word lengths of 16 or 32 bits can be transferred between the μPD77230 and the host. Four pins can be used in slave mode as general-purpose I/O ports: two input pins and two output pins.

Figure 3 shows the functional pin groups in master mode and slave mode.

Figure 1. Master Mode Block Diagram

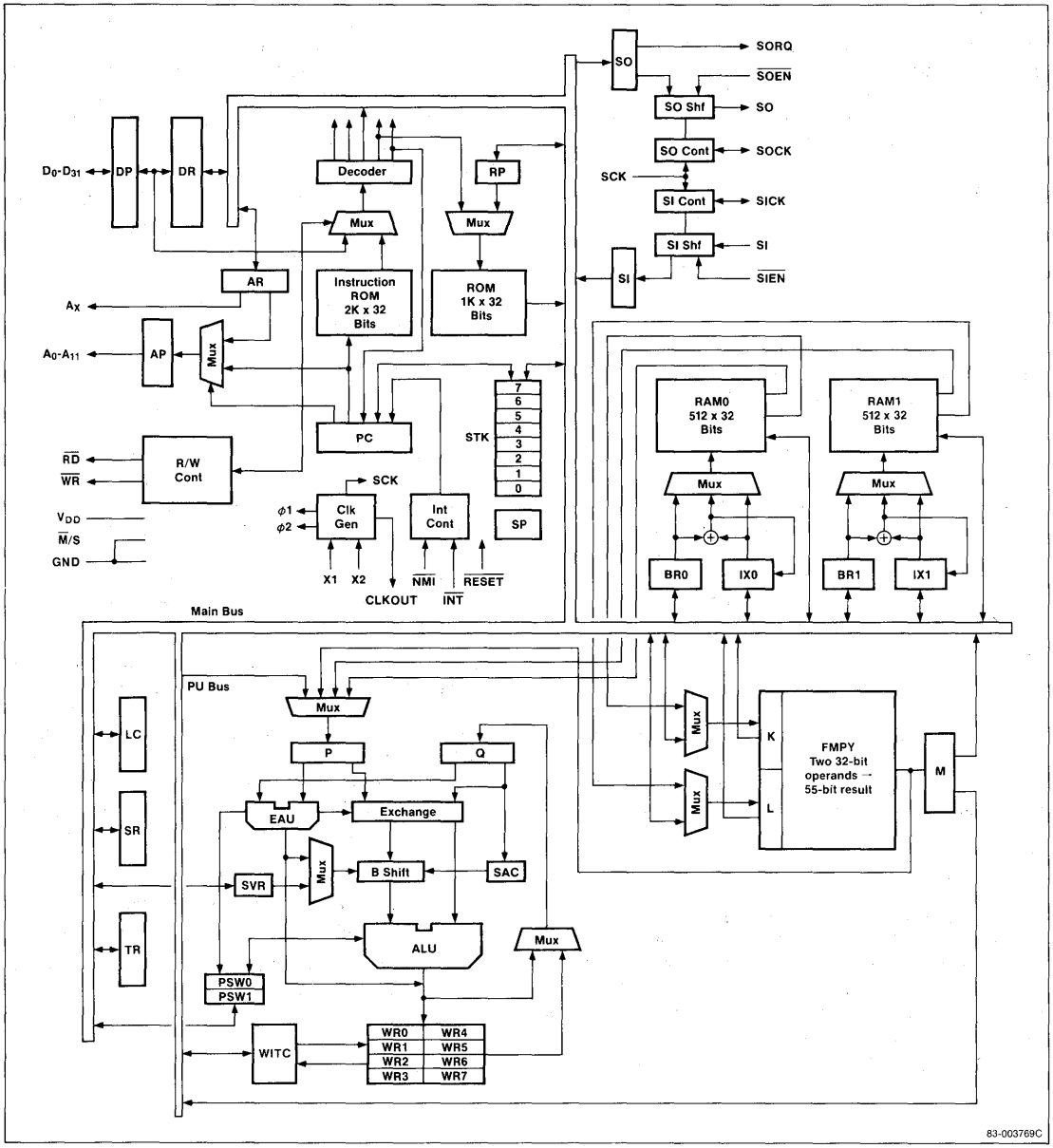


Figure 2. Slave Mode Block Diagram

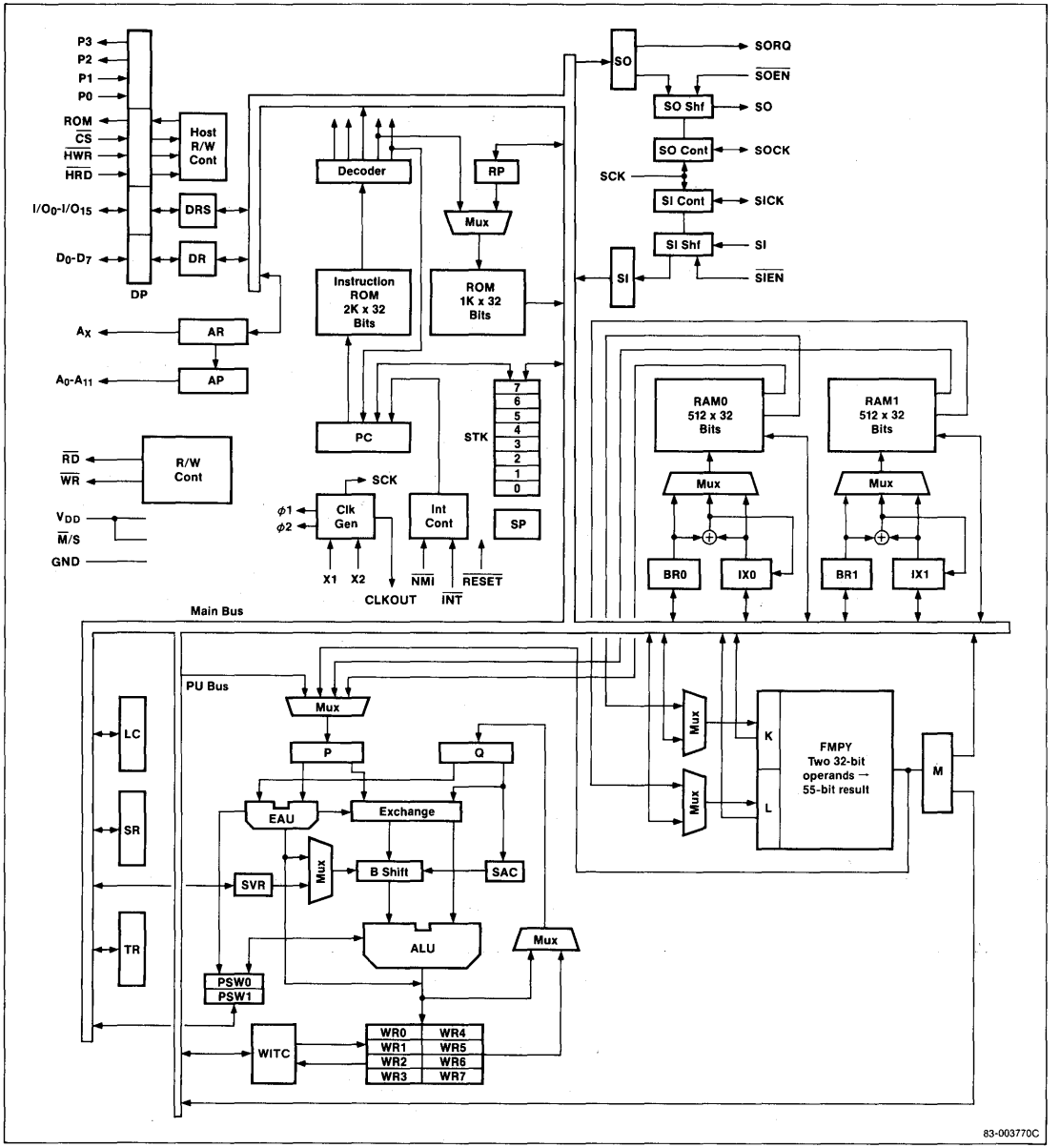
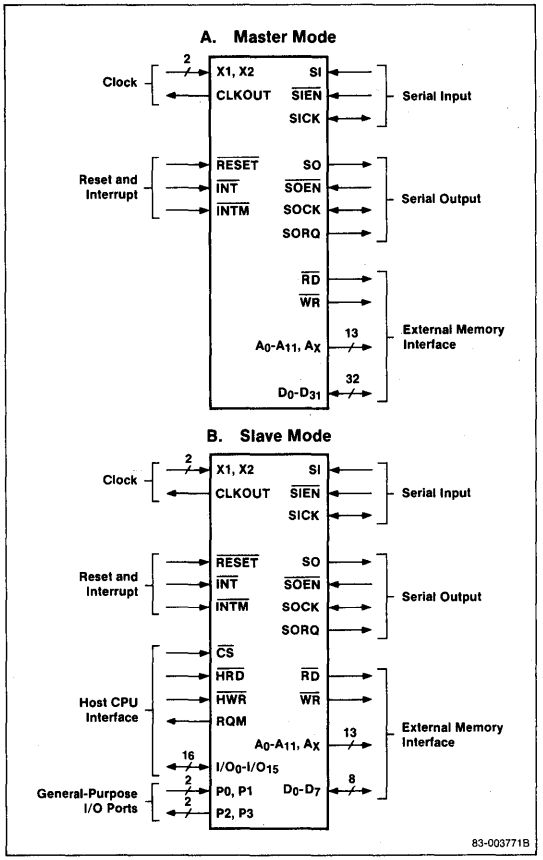


Figure 3. Functional Pin Groups



Instruction Set

All μPD77230 instructions consist of a single 32-bit word. Figure 4 shows the bit format for the three basic types of instructions.

OP Type Instruction

This is an ALU operation instruction where 26 different operations may be specified in the upper five bits (figure 4). Pointer modifications may be specified in the CNT field. Transfers may also be specified within an OP instruction by use of the SRC and DST fields. When all fields are specified in an OP instruction, several different tasks are performed at once. The high five bits make up the OP field, summarized in table 1.

Table 2 summarizes the effect on bits in the PSW resulting from ALU operations.

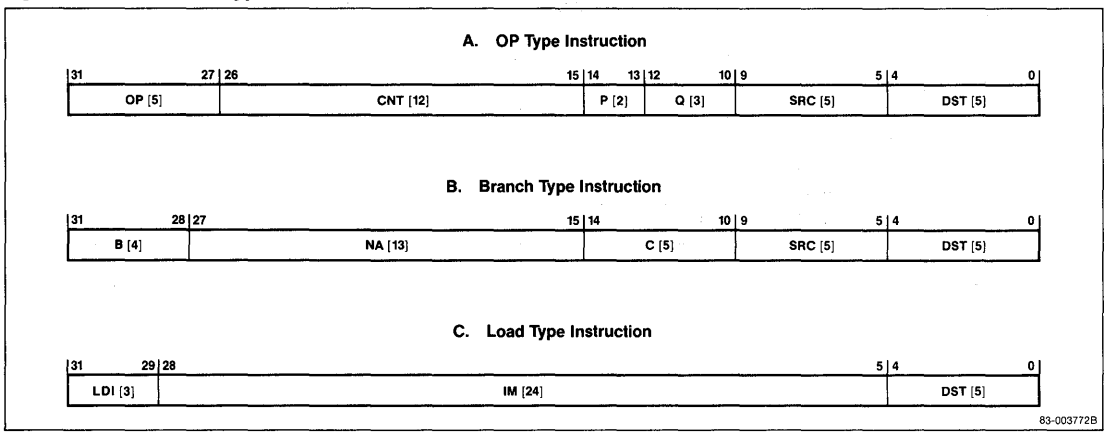
Control Field [CNT]

This 12-bit field contains specifications for control modes and pointer modifications. Figure 5 summarizes the bit field format, table 3 summarizes the function of CNT field groups, and table 4 summarizes the function of each mnemonic within the 23 groups. Table 5 shows the possible combinations of control field instructions.

P Field

The two-bit P field specifies the source of input to the P register, which is used as an input to the ALU for operations requiring two operands. See table 6.

Figure 4. Instruction Type Formats



2

Table 1. OP Field Specifications

Mnemonic	OP Field (31-27)	Operation
NOP	00000	No operation
INC	00001	Increment
DEC	00010	Decrement
ABS	00011	Absolute value
NOT	00100	Not-one's complement
NEG	00101	Negate-two's complement
SHLC	00110	Shift left with carry
SHRC	00111	Shift right with carry
ROL	01000	Rotate left
ROR	01001	Rotate right
SHLM	01010	Shift left multiple
SHRM	01011	Shift right multiple
SHRAM	01100	Shift right arithmetic multiple
CLR	01101	Clear
NORM	01110	Normalize
CVT	01111	Convert floating point format
ADD	10000	Fixed-point add
SUB	10001	Fixed-point subtract
ADDC	10010	Fixed-point add with carry
SUBC	10011	Fixed-point subtract with borrow
CMP	10100	Compare (floating point)
AND	10101	Logical AND
OR	10110	Logical OR
XOR	10111	Logical exclusive OR
ADDF	11000	Floating-point add
SUBF	11001	Floating-point subtract

Table 2. Effects of ALU Operations on PSW Flags

ALU Operation	Contents of PSW				
	OVFE	C	Z	S	OVFM
NOP	*	*	*	*	*
INC	*	$	$	$	$
DEC	*	$	$	$	$
ABS	*	$	$	0	$+
NOT	*	0	$	$	0
NEG	*	$	$	$	$+
SHLC	*	$	$	$	0
SHRC	*	$	$	$	0
ROL	*	0	*	$	0
ROR	*	0	*	$	0
SHLM	*	0	$	$	0
SHRM	*	0	$	$	0
SHRAM	*	0	$	$	0
CLR	0	0	1	0	0
NORM (NORM.)	$	0	$	$	0
(ROUNDING)	$	$	$	$	$
(FLT-FIX)	*	0	$	$	$
(FIX M.A.)	*	0	$	$	$
CVT	X	0	$	$	0
ADD	*	$	$	$	$
SUB	*	$	$	$	$
ADDC	*	$	$	$	$
SUBC	*	$	$	$	$
CMP	$	$	$	$	$
AND	*	0	$	$	0
OR	*	0	$	$	0
XOR	*	0	$	$	0
ADDF	$	$	$	$	$
SUBF	$	$	$	$	$

$ Flag will be affected by result of operation.

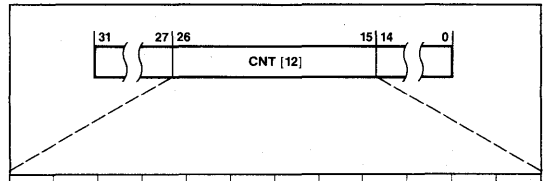
0 Flag will be reset to 0.

1 Flag will be reset to 1.

* Previous condition of flag will be preserved.

\+ If the original data in the mantissa was 80---0H, OVFM = 1 after operation.

Figure 5. Control Field Bit Format



26	25	24	23	22	21	20	19	18	17	16	15
0	0	M0		M1		DP0			DP1		
0	1	0	0	EA		DP0			DP1		
0	1	0	1	RP		M0		DP0			FC
0	1	1	0	RP		M1		DP1			FC
0	1	1	1	RP		M0		M1		L	FC
1	0	0	0	0	BAS0			BAS1			FC
1	0	0	0	1	RPC					L	FC
1	0	0	1	0	P3	P2	EM	BM		L	FC
1	0	0	1	1	RW					L	FC
1	0	1	0	0	WT					L	FC
1	0	1	0	1	NF			WI		L	FC
1	0	1	1	0	FIS			FD		L	
1	0	1	1	1	SHV						
1	1	0	RPS								
1	1	1	NAL								

83-003773A

Table 3. Control Field Function Summary

Group	Field	Function	Effective
Interrupt	EM, BM	Enable and disable maskable interrupt, and control interrupt memorization.	→
PSW	FIS	PSW control (select and clear)	*
	FC	Select other PSW	*
Data ROM pointer	RP	Controls ROM pointer operation	→
	RPC	Specifies n value for special manipulation of ROM pointer	→
	RPS	Specifies 9 lower bits of data ROM address	→
Data RAM0 and RAM1 pointers	M0	Specifies RAM0 addressing mode	→
	M1	Specifies RAM1 addressing mode	→
	DP0	Controls modification of base pointer 0 and index register 0	→
	DP1	Controls modification of base pointer 1 and index register 1	→
	BASE0	Specifies counter length of modulo count operation of base pointer 0	→
	BASE1	Specifies counter length of modulo count operation of base pointer 1	→
Data format conversion	FD	Controls conversion mode for floating point CVT.	*
	WI	Controls transfer format when working register is specified in DST field.	→
	WT	Controls transfer format when working register is specified in SRC field.	→
Normalization specification	NF	Specifies normalization, normalization with rounding, floating-point to fixed-point conversion, or digit alignment.	*
Shift specification	SHV	Controls amount of shift for 47-bit mantissa	*
Data memory access	RW	Specifies read/write operation for external memory.	*
	EA	Increments or decrements external address register	*
General-purpose output port	P2	Controls state of P2 pin	→
	P3	Controls state of P3 pin	→
Loop counter	L	Decrements loop counter	→
Jump	NAL	Specifies unconditional local jump address	*

* Effective starting with current instruction.

→ Effective starting with next instruction.

Table 4. Control Field Mnemonic Summary

Operation	Mnemonic		Code
EM, BM Field (19-17)			
Maskable interrupt	EM	BM	
No operation	(NOP)	(NOP)	000
Clear booking flag	(NOP)	CLRBM	001
Set booking flag	(NOP)	SETBM	010
Interrupt disabled	DI	(NOP)	011
Interrupt enabled	EI	(NOP)	100
Interrupt enabled and clear booking flag	EI	CLRM	101
Interrupt enabled and set booking flag	EI	SETBM	110
Use prohibited	—	—	111

* Default: interrupt disabled and clear booking flag.

* Writing (NOP) is not necessary, just useful for remembering the available combinations and their effects.

FIS Field (21-19)		
Flag initialize and select		
No operation	(NOP)	000
Specify PSW 0 for operation (default)	SPCPSW0	001
Specify PSW 1 for operation	SPCPSW1	010
Clear PSW 0	CLRPSW0	100
Clear PSW 1	CLRPSW1	101
Clear PSW 0 and PSW 1	CLRPSW	110
FC Bit (15)		
Flag change operation		
No operation	(NOP)	0
Exchange PSW for operation	XCHPSW	1
RP Field (22, 21)		
ROM pointer modification		
No operation	(NOP)	00
Increment ROM pointer	INCRP	01
Decrement ROM pointer	DECRP	10
Increment specified bit of ROM pointer (that is, add 2^N)	INCBRP	11
RPC Field (21-18)		
Specify N for adding 2^N to ROM pointer	BITRP imm	(imm)B
*imm (= n) is 0 through 9		
RPS Field (23-15)		
Specify immediate ROM address	SPCRA imm	(imm)B
$*0 \leq imm \leq 511$		

Table 4. Control Field Mnemonic Summary (cont)

Operation	Mnemonic	Code
M0 Field		
Specify RAM pointer		
No change in specification	(NON)	00
Base pointer 0	SPCBP0	01
Index register 0	SPCIX0	10
Base pointer 0 + index register 0 (default)	SPCBI0	11
M1 Field		
Specify RAM pointer		
No change in specification	(NON)	00
Base pointer 1	SPCBP1	01
Index register 1	SPCIX1	10
Base pointer 1 + index register 1 (default)	SPCBI1	11
DP0 Field		
Pointer modification operation		
No operation	(NOP)	000
Increment base pointer 0	INCBP0	001
Decrement base pointer 0	DECBP0	010
Clear base pointer 0	CLRBP0	011
Store base + index to index register 0	STIX0	100
Increment index register 0	INCIX0	101
Decrement index register 0	DECIX0	110
Clear index register 0	CLRIX0	111
DP1 Field		
Pointer modification operation		
No operation	(NOP)	000
Increment base pointer 1	INCBP1	001
Decrement base pointer 1	DECBP1	010
Clear base pointer 1	CLRBP1	011
Store base + index to index register 1	STIX1	100
Increment index register 1	INCIX1	101
Decrement index register 1	DECIX1	110
Clear index register 1	CLRIX1	111

Operation	Mnemonic		Code
BASE0 Field (21-19)			
Specify modulo count number (2^N) for incrementing base pointer 0	MCNBP0	imm	(imm)B
*imm (=n) is 1 through 7; 0 specifies ordinary count			
BASE1 Field (18-16)			
Specify modulo count number (2^N) for incrementing base pointer 1	MCNBP1	imm	(imm)B
*imm (=n) is 1 through 7; 0 specifies ordinary count			
FD Field			
Data conversion format specification			
No change of specification	(NON)		00
Conversion of ASP format to IEEE format (default)	SPIE		01
Conversion of IEEE format to ASP format	IESP		10
Use prohibited			11
WI Field (18, 17)			
Specification of transfer format when data is moved from IB to WR			
No change of specification	(NON)		00
Transfer low 24 bits of mantissa to high 24 bits	BWRL24		01
Ordinary transfer (default)	BWRORD		10
Use prohibited			11
WT Field (21-19)			
Specification of transfer format when data is moved from WR to IB			
No change of specification	(NON)		000
Ordinary transfer (default)	WRBORD		001
Low 24 bits of mantissa to high 24	WRBL24		010
Low 23 bits (bit 23 = 0) to high 24	WRBL23		011
Exponent part to mantissa low 8 bits	WRBEL8		100
Mantissa low 8 bits to exponent part	WRBL8E		101
Exchange high 8 bits of mantissa with low 8 bits of mantissa	WRBXCH		110
Bit reverse entire mantissa	WRBBRV		111

Table 4. Control Field Mnemonic Summary (cont)

Operation	Mnemonic	Code
NF Field (21-19)		
Normalization format specification		
No change of specification	(NON)	000
Truncating normalization (default)	TRNORM	010
Rounding normalization	RDNORM	100
Convert floating to fixed point	FLTFIX	110
Fixed point multiple alignment (multiple value is in SVR)	FIXMA	111
SHV Field (21-15)		
Set shift value to SVR		
imm bits left shift (default)	SETSVL imm	0 (imm)B
imm bits right shift	SETSVR imm	1 (imm)B
*$0 \leq imm \leq 46$		
RW Field (21, 20)		
Operation for external data memory		
No operation	(NOP)	00
Read	RD	01
Write	WR	10
Use prohibited		11
EA Field (22, 21)		
Operation for external address register		
No operation	(NOP)	00
Increment external address register	INCAR	01
Decrement external address register	DECAR	10
Use prohibited		11

Operation	Mnemonic	Code
P2 Bit (20)		
P2 pin control (slave mode only)		
Clear output port pin 2	CLRP2	0
Set output port pin 2	SETP2	1
P3 Bit (21)		
P3 pin control (slave mode only)		
Clear output port pin 3	CLRP3	0
Set output port pin 3	SETP3	1
L Bit (16)		
Loop counter operation		
No operation	(NOP)	0
Decrement loop counter	DECLC	1
NAL Bit (23-15)		
Local branch; jump to imm address in local block	JBLK imm	(imm)B
*$0 \leq imm \leq 511$		

Table 5. Control Field Instruction Combinations

Case 1	SPCBP0 SPCIX0 SPCBI0	SPCBP1 SPCIX1 SPCBI1	INCBP0 DECBP0 CLRBP0 STIX0 INCIX0 DECIX0 CLRIX0	INCBP1 DECBP1 CLRBP1 STIX1 INCIX1 DECIX1 CLRIX1		
Case 2	INCAR DECAR	INCBP0 DECBP0 CLRBP0 STIX0 INCIX0 DECIX0 CLRIX0	INCBP1 DECBP1 CLRBP1 STIX1 INCIX1 DECIX1 CLRIX1			
Case 3	INCRP DECRP INCBRP	SPCBP0 SPCIX0 SPCBI0	INCBP0 DECBP0 CLRBP0 STIX0 INCIX0 DECIX0 CLRIX0	XCHPSW		
Case 4	INCRP DECRP INCBRP	SPCBP1 SPCIX1 SPCBI1	INCBP1 DECBP1 CLRBP1 STIX1 INCIX1 DECIX1 CLRIX1	XCHPSW		
Case 5	INCRP DECRP INCBRP	SPCBP0 SPCIX0 SPCBI0	SPCBP1 SPCIX1 SPCBI1	DECLC	XCHPSW	
Case 6	MCNBP0 imm	MCNBP1 imm	XCHPSW			
Case 7	BITRP imm	DECLC	XCHPSW			
Case 8	CLRP2 SETP2	CLRP3 SETP3	EI DI	CLRBM SETBM	DECLC	XCHPSW
Case 9	RD SR	DECLC	XCHPSW			
Case 10	WRBORD WRBL24 WRBL23 WRBEL8 WRBL8E WRBXCH WRBBRV	DECLC	XCHPSW			
Case 11	TRNORM RDNORM FLTFIX FIXMA	BWRL24 BWRORD	DECLC	XCHPSW		
Case 12	SPCPSW0 SPCPSW1 CLRPSW0 CLRPSW1 CLRPSW	SPIE IESP	DECLC			
Case 13	SETSVL imm SETSVR imm					
Case 14	SPCRA imm					
Case 15	JBLK imm					

Table 6. P Field Specifications

Mnemonic	P Field (14, 13)	Input of P Register
IB	00	Internal bus
M	01	Multiplier output register
RAM0	10	RAM block 0
RAM1	11	RAM block 1

Q Field

The three-bit Q field specifies the source of input to the Q register, which is the other of two ALU input registers. See table 7.

Table 7. Q Field Specifications

Mnemonic	Q Field (12-10)	Register
WR0	000	Working register 0
WR1	001	Working register 1
WR2	010	Working register 2
WR3	011	Working register 3
WR4	100	Working register 4
WR5	101	Working register 5
WR6	110	Working register 6
WR7	111	Working register 7

Source Field

Table 8 lists 32 source registers that may be specified in the source field.

Destination Field

Table 9 lists 32 destinations that may be specified in the DST field. Note that the LKR0 and KLR1 specifications will simultaneously load, as destinations, both the K and L registers.

Branch Instruction

The branch instruction type is used for a jump, conditional jump, call, or return. The format of the branch instruction is shown in figure 4. The destination address of the branch is contained in the 13-bit NA field. Note that the most significant bit of the NA field is used to determine whether the destination address is in internal or external instruction memory. The five-bit C field summarized in table 10 determines the nature of the branch.

Note also that an SRC and DST may be included as part of the branch instruction. This data transfer will take place regardless of any condition upon which a jump may be dependent.

LDI Instruction

Figure 4 shows the format of the LDI instruction type. The 24-bit IM (immediate) field contains the data that will be loaded into the register specified by the DST field. It is also possible to load a 32-bit floating-point number using this instruction in conjunction with the TRE destination field specification.

Table 8. SRC Field Specifications

Mnemonic	SRC Field (9-5)	Selected Source Register
NON	00000	No source selected
RP	00001	ROM pointer
PSW0	00010	Program status word 0
PSW1	00011	Program status word 1
SVR	00100	SVR (shift value register)
SR	00101	Status register
LC	00110	Loop counter
STK	00111	Top of stack
M	01000	M register (multiplier output)
ML	01001	Low 24 bits of M register
ROM	01010	Data ROM output
TR	01011	Temporary register
AR	01100	External address register
SI	01101	Serial input register
DR	01110	Data register
DRS	01111	Data register for slave
WR0	10000	Working register 0
WR1	10001	Working register 1
WR2	10010	Working register 2
WR3	10011	Working register 3
WR4	10100	Working register 4
WR5	10101	Working register 5
WR6	10110	Working register 6
WR7	10111	Working register 7
RAM0	11000	RAM block 0
RAM1	11001	RAM block 1
BP0	11010	Base pointer 0
BP1	11011	Base pointer 1
IX0	11100	Index register 0
IX1	11101	Index register 1
K	11110	K register
L	11111	L register

Table 9. DST Field Specifications

Mnemonic	DST Field (4-0)	Selected Destination Register
NON	00000	No destination selected
RP	00001	ROM pointer
PSW0	00010	Program status word 0
PSW1	00011	Program status word 1
SVR	00100	SVR (shift value register)
SR	00101	Status register
LC	00110	Loop counter
STK	00111	Top of stack
LKR0	01000	L register (RAM 0 to K register)
KLR1	01001	K register (RAM 1 to L register)
TRE	01010	Exponent part of temporary register
TR	01011	Temporary register
AR	01100	External address register
SO	01101	Serial output register
DR	01110	Data register
DRS	01111	Data register for slave
WR0	10000	Working register 0
WR1	10001	Working register 1
WR2	10010	Working register 2
WR3	10011	Working register 3
WR4	10100	Working register 4
WR5	10101	Working register 5
WR6	10110	Working register 6
WR7	10111	Working register 7
RAM0	11000	RAM block 0
RAM1	11001	RAM block 1
BP0	11010	Base pointer 0
BP1	11011	Base pointer 1
IX0	11100	Index register 0
IX1	11101	Index register 1
K	11110	K register
L	11111	L register

Table 10. Branch Condition Summary (C Field)

Mnemonic	C Field (14-10)	Jump with Condition
JMP	00000	Jump unconditionally
CALL	00001	Subroutine call
RET	00010	Return from interrupt or subroutine
JNZRP	00011	Jump if ROM pointer not zero
JZ0	00100	Jump if zero flag 0 is set
JNZ0	00101	Jump if zero flag 0 is reset
JZ1	00110	Jump if zero flag 1 is set
JNZ1	00111	Jump if zero flag 1 is reset
JC0	01000	Jump if carry flag 0 is set
JNC0	01001	Jump if carry flag 0 is reset
JC1	01010	Jump if carry flag 1 is set
JNC1	01011	Jump if carry flag 1 is reset
JS0	01100	Jump if sign flag 0 is set
JNS0	01101	Jump if sign flag 0 is reset
JS1	01110	Jump if sign flag 1 is set
JNS1	01111	Jump if sign flag 1 is reset
JV0	10000	Jump if overflow flag 0 is set
JNV0	10001	Jump if overflow flag 0 is reset
JV1	10010	Jump if overflow flag 1 is set
JNV1	10011	Jump if overflow flag 1 is reset
JEV0	10100	Jump if exponent overflow flag 0 is set
JEV1	10101	Jump if exponent overflow flag 1 is set
JNFSI	10110	Jump if SI register is not full
JNESO	10111	Jump if SO register is not empty
JIP0	11000	Jump if input port 0 is on
JIP1	11001	Jump if input port 1 is on
JNZIX0	11010	Jump if index register 0 nonzero
JNZIX1	11011	Jump if index register 1 nonzero
JNZBP0	11100	Jump if base pointer 0 nonzero
JNZBP1	11101	Jump if base pointer 1 nonzero
JRDY	11110	Jump if ready is on
JROM	11111	Jump if request for master is on

System Configurations

The μPD77230 may be configured in a variety of ways, from simple systems to complex. Figure 6 is the simplest example showing the μPD77230 as a stand-alone processor performing a preset filtering function. The only other devices needed are A/D and D/A converters, which can be a single-chip combo device as shown in the figure plus necessary clock and timing circuitry. Figure 7 shows the same stand-alone operation with external memory and memory-mapped I/O to implement various control functions along with processing the signal itself.

Figure 6. Stand-Alone μPD77230 with Codec

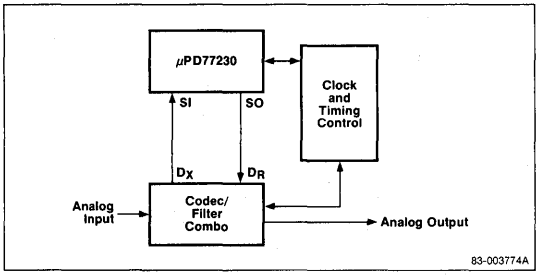


Figure 7. Stand-Alone μPD77230 with Codec, External Memory, and I/O

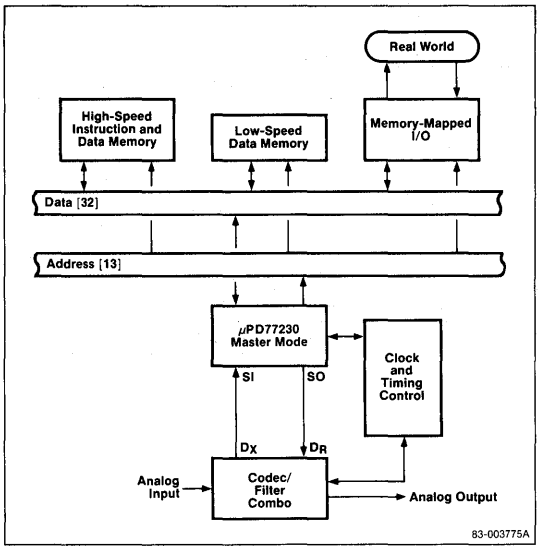


Figure 8 shows a μPD77230 in a slave mode as a peripheral to a host processor. Note that in slave mode, the μPD77230 can still be the "master" of its local bus with the four general-purpose I/O pins available for use.

Figure 9 shows how to cascade multiple μPD77230s to increase system throughput. The cascading is done by using only the serial ports so that the μPD77230s themselves can be in any mode of operation desired. For example, they may all be in master mode, they may all be slaves to the same host processor, they may all be slaves to different hosts, or one may be the master with the others as slaves to it.

Figure 8. Slave μPD77230 as Peripheral to Host Processor

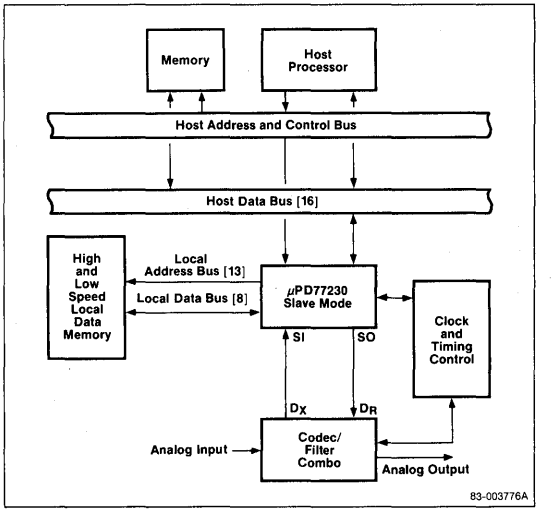


Figure 9. μPD77230s Cascaded Through Serial I/O Ports

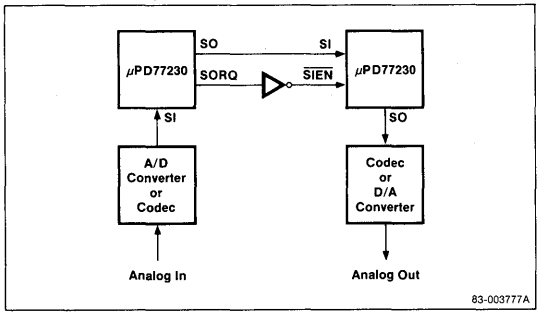


Figure 10. Large System with Many Options

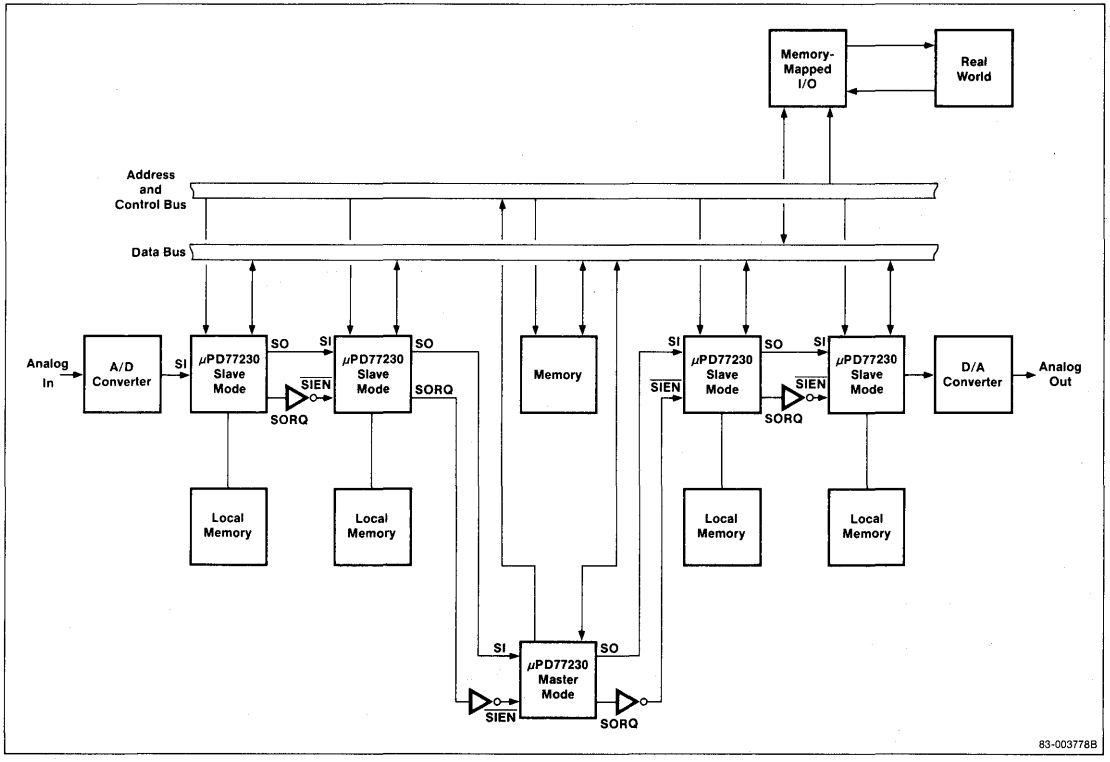


Figure 10 shows an arbitrarily large system with cascading master mode and slave mode μPD77230s. In this example, the master μPD77230 might do little actual signal processing. Instead, it will be an overall system controller gathering information from inputs in the I/O block, from the slave μPD77230 I/O ports, and from its own processing of the signal. It will then control the other μPD77230s and the system outputs of the I/O block.

Support Tools

The μPD77230 has a wide variety of development and software support tools. Both absolute and relocatable assemblers, with powerful pre-assembler options are available. In addition, a software simulator and in-circuit emulator will aid the designer in performance evaluation and hardware integration. The software tools options are as follows:

- Assembler: MS-DOS, CP/M-86, VAX VMS, VAX UNIX
- Simulator: VAX VMS, VAX UNIX

Absolute Maximum Ratings

$T_A = 25°C$

Supply voltage, V_{DD}	−0.5 to +6.5 V
Voltage on any input pin, V_I	−0.5 to V_{DD} + 0.5 V
Voltage on any output pin, V_O	−0.5 to V_{DD} + 0.5 V
Storage temperature, T_{STG}	−65 to +150°C

Note: Voltages are with respect to ground.

Recommended Operating Conditions

Parameter	Symbol	Limits Min	Typ	Max	Unit
Supply voltage	V_{DD}	4.75	5.0	5.25	V
Low-level input voltage	V_{IL}	−0.3		0.8	V
High-level input voltage	V_{IH}	2.2		V_{DD} + 0.3	V
Low-level X1 input voltage	V_{ILX}	−0.3		0.5	V
High-level X1 input voltage	V_{IHX}	3.9		V_{DD} + 0.3	V
Operating free-air temperature	T_{OPT}	−10	25	70	°C

DC Characteristics

$T_A = -10$ to +70°C; $V_{DD} = 5$ V ±5%

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
Low-level output voltage	V_{OL}			0.45	V	$I_{OL} = 2.0$ mA
High-level output voltage	V_{OH}	0.7V_{DD}			V	$I_{OH} = -400$ μA
Low-level input current	I_{IL}			−400	μA	$V_{IN} = 0$V; $\overline{RESET}$, SICK, SOCK
High-level input current	I_{IH}			400	μA	$V_{IN} = V_{DD}$; $\overline{M}/S$
Low-level input leak current	I_{LIL}			−10	μA	$V_{IN} = 0$ V, except $\overline{RESET}$, SICK, SOCK
High-level input leak current	I_{LIH}			10	μA	$V_{IN} = V_{DD}$, except $\overline{M}/S$
Low-level output leak current	I_{LOL}			−10	μA	$V_{OUT} = 0$ V
High-level output leak current	I_{LOH}			10	μA	$V_{OUT} = V_{DD}$
X1 input current	I_{IX1}			400	μA	External clock input
Supply current	I_{DD}		200	300	mA	f_{CYX} =13.3333 MHz

Capacitance

$T_A = 25°C$; $V_{DD} = 0$ V

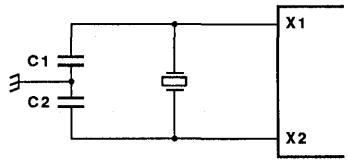
Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
Input capacitance	C_{IN}			10	pF	$f_C = 1$ MHz
Output capacitance	C_{OUT}			20	pF	

Clock Timing

Internal Clock

$T_A = -10$ to +70°C; $V_{DD} = 5$ V ±5%

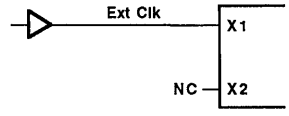
Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
Input clock frequency	f_{CYX}	1	13.3333	13.513	MHz	See diagram below
C1, C2 capacitance			15		pF	



External Clock

$T_A = -10$ to +70°C; $V_{DD} = 5$ V ±5%

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
X1 cycle time	t_{CYX}	74	75	1000	ns	See diagram below; measured at 1.0 V and 3.0 V
X1 high pulse width	t_{XXH}	27			ns	
X1 low pulse width	t_{XXL}	27			ns	
X1 rise time	t_{XR}			10	ns	
X1 fall time	t_{XF}			10	ns	
SICK, SOCK cycle time	t_{CYS}	242	244		ns	
SICK, SOCK high pulse width	t_{SSH}	101			ns	
SICK, SOCK low pulse width	t_{SSL}	101			ns	
SICK, SOCK rise time	t_{SR}			20	ns	
SICK, SOCK fall time	t_{SF}			20	ns	



Clock Timing (cont)

Switching

$T_A = -10$ to $+70°C$; $V_{DD} = 5\ V \pm 5\%$; $C_L = 100\ pF$

Parameter	Symbol	Limits Min	Limits Max	Unit
X1 ↑ → CLKOUT delay time	t_{DXC}		50	ns
X1 ↑ → CLKOUT hold time	t_{HXC}	0		ns
SCK cycle time	t_{CYS}	$8t_{CYX}$		ns
SCK high pulse width	t_{SSH}	$4t_{CYX} - 65$		ns
SCK low pulse width	t_{SSL}	$4t_{CYX} - 65$		ns
SCK rise time	t_{SR}		20	ns
SCK fall time	t_{SF}		20	ns
S1 → SCK ↑ delay time	t_{DXS}	10	120	ns

External Memory Access Timing

Setup and Hold

$T_A = -10$ to $+70°C$; $V_{DD} = 5\ V \pm 5\%$

Parameter	Symbol	Limits Min	Limits Max	Unit	Test Conditions
Data setup time for address	t_{SADI}		$2t_{CYX} - 95$	ns	Instruction read
Data setup time for $\overline{RD}$	t_{SRDI}		$2t_{CYX} - 35$	ns	
Data hold time for $\overline{RD}$	t_{HRDI}	0		ns	
Data setup time for Address	t_{SAD1}		$4t_{CYX} - 135$	ns	High-speed
	t_{SAD2}		$8t_{CYX} - 135$	ns	Low-speed
Data setup time for $\overline{RD}$	t_{SRD1}		$3t_{CYX} - 75$	ns	High-speed
	t_{SRD2}		$8t_{CYX} - 75$	ns	Low-speed
Data hold time for $\overline{RD}$	t_{HRD}	0		ns	

Switching

$T_A = -10$ to $+70°C$; $V_{DD} = 5\ V \pm 5\%$; $C_L = 100\ pF$

Parameter	Symbol	Limits Min	Limits Max	Unit	Test Conditions
X1 ↑ → $\overline{RD}$ delay time	t_{DXRD}		70	ns	
X1 ↑ → $\overline{WR}$ delay time	t_{DXWR}		70	ns	
Address setup time for $\overline{RD}$	t_{SAR}	$t_{CYX} - 60$		ns	
Address hold time for $\overline{RD}$	t_{HRA}	5		ns	
$\overline{RD}$ pulse width	t_{WRI}	$t_{CYX} - 30$		ns	Instruction read
	t_{WR1}	$3t_{CYX} - 30$		ns	High-speed
	t_{WR2}	$7t_{CYX} - 30$		ns	Low-speed
Address setup time for $\overline{WR}$	t_{SAW}	$t_{CYX} - 55$		ns	
Address hold time for $\overline{WR}$	t_{HWA}	5		ns	
$\overline{WR}$ pulse width	t_{WW1}	$3t_{CYX} - 50$		ns	High-speed
	t_{WW2}	$7t_{CYX} - 50$		ns	Low-speed
Data setup time for $\overline{WR}$	t_{SDW1}	$3t_{CYX} - 100$		ns	High-speed
	t_{SDW2}	$7t_{CYX} - 100$		ns	Low-speed
$\overline{WR}$ ↓ → Data delay time	t_{DWD}	0		ns	
$\overline{WR}$ ↑ → Data float time	t_{FWD}	10	50	ns	
$\overline{RD}$, $\overline{WR}$ recovery time	t_{RV}	$t_{CYX} - 35$		ns	

Host Interface Timing, Slave Mode

Setup and Hold

$T_A = -10$ to $+70°C$; $V_{DD} = 5\ V \pm 5\%$

Parameter	Symbol	Limits Min	Limits Max	Unit
$\overline{CS}$ setup time for $\overline{HRD}$	t_{SCR}	0		ns
$\overline{CS}$ hold time for $\overline{HRD}$	t_{HRC}	0		ns
$\overline{HRD}$ pulse width	t_{WHRD}	150		ns
$\overline{CS}$ setup time for $\overline{HWR}$	t_{SCW}	0		ns
$\overline{CS}$ hold time for $\overline{HWR}$	t_{HWC}	0		ns
$\overline{HWR}$ pulse width	t_{WHWR}	150		ns
Data setup time for $\overline{HWR}$	t_{SIHW}	100		ns
Data hold time for $\overline{HWR}$	t_{HHWI}	0		ns
$\overline{HRD}$, $\overline{HWR}$ recovery time	t_{HRV}	100		ns
$\overline{HRD}$, $\overline{HWR}$ hold time for RQM	t_{HRH}	t_{CYX}		ns
P0, P1 setup time for X1	t_{SPX}	t_{CYX}		ns
P0, P1 hold time for X1	t_{HXP}	t_{CYX}		ns

Host Interface Timing, Slave Mode (cont)

Switching

$T_A = -10$ to $+70°C$; $V_{DD} = 5\ V \pm 5\%$; $C_L = 100\ pF$

Parameter	Symbol	Limits Min	Limits Max	Unit
$\overline{HRD}$ ↓ → Data delay time	t_{DHRI}		100	ns
$\overline{HRD}$ ↑ → Data float time	t_{FHRI}	10	65	ns
X1 ↑ → RQM ↑ delay time	t_{DXRH}		100	ns
X1 ↑ → RQM ↓ delay time	t_{DXRL}		100	ns
$\overline{HRD}$, $\overline{HWR}$ ↑ → RQM ↓ delay time	t_{DHR}		$2t_{CYX} + 100$	ns
X1 ↑ → P2, P3 delay time	t_{DXP}		100	ns

Interrupt Reset Timing

$T_A = -10$ to $+70°C$; $V_{DD} = 5\ V \pm 5\%$

Parameter	Symbol	Limits Min	Limits Max	Unit
$\overline{NMI}$, $\overline{INT}$ pulse width	t_{INT}	$6t_{CYX}$		ns
$\overline{NMI}$, $\overline{INT}$ hold time for $\overline{RESET}$ ↑	t_{HRNI}	$6t_{CYX}$		ns
$\overline{NMI}$, $\overline{INT}$ recovery time	t_{RINT}	$6t_{CYX}$		ns
$\overline{RESET}$ pulse width	t_{RST}	$6t_{CYX}$		ns

Serial Interface Timing

Setup and Hold

$T_A = -10$ to $+70°C$; $V_{DD} = 5\ V \pm 5\%$

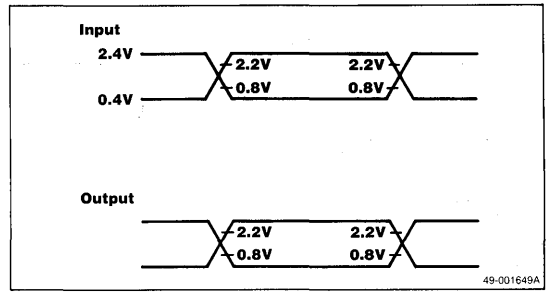
Parameter	Symbol	Limits Min	Limits Max	Unit
$\overline{SIEN}$, SI setup time for SCK ↓	t_{SSIS}	55		ns
$\overline{SIEN}$, SI hold time for SCK ↓	t_{HSSI}	30		ns
$\overline{SOEN}$ setup time for SCK ↑	t_{SSES}	50		ns
$\overline{SOEN}$ hold time for SCK	t_{HSSE}	30		ns
$\overline{SIEN}$, $\overline{SOEN}$ recovery time	t_{SRV}	t_{CYS}		ns

Switching

$T_A = -10$ to $+70°C$; $V_{DD} = 5\ V \pm 5\%$; $C_L = 100\ pF$

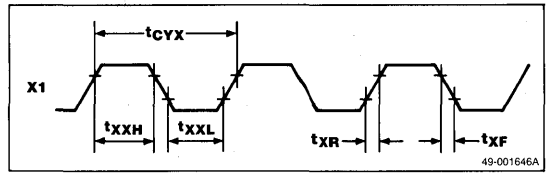
Parameter	Symbol	Limits Min	Limits Max	Unit
SCK ↓ → SORQ delay time	t_{DSSQ}	30	150	ns
$\overline{SOEN}$ ↓ → SO delay time	t_{DSESO}		60	ns
$\overline{SOEN}$ ↑ → SO float time	t_{FSESO}	10	100	ns
SCK ↑ → SO delay time	t_{DSLSO}		60	ns
SCK ↑ → SO hold time	t_{HSHSO}	0		ns
SCK ↑ → SO delay time	t_{DSHSO}		60	ns
SCK ↑ → SO float time (SORQ ↓)	t_{FSSO}	10	100	ns

Timing Measurement Points

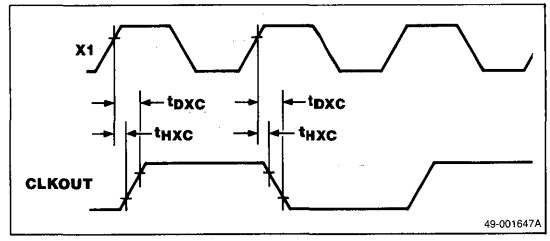


Clock Timing Waveforms

Master Clock

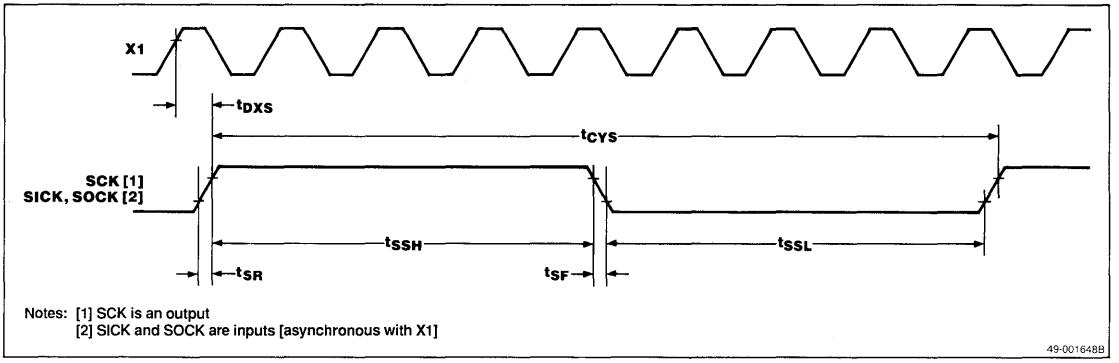


Clock Output



Clock Timing Waveforms (cont)

Switching

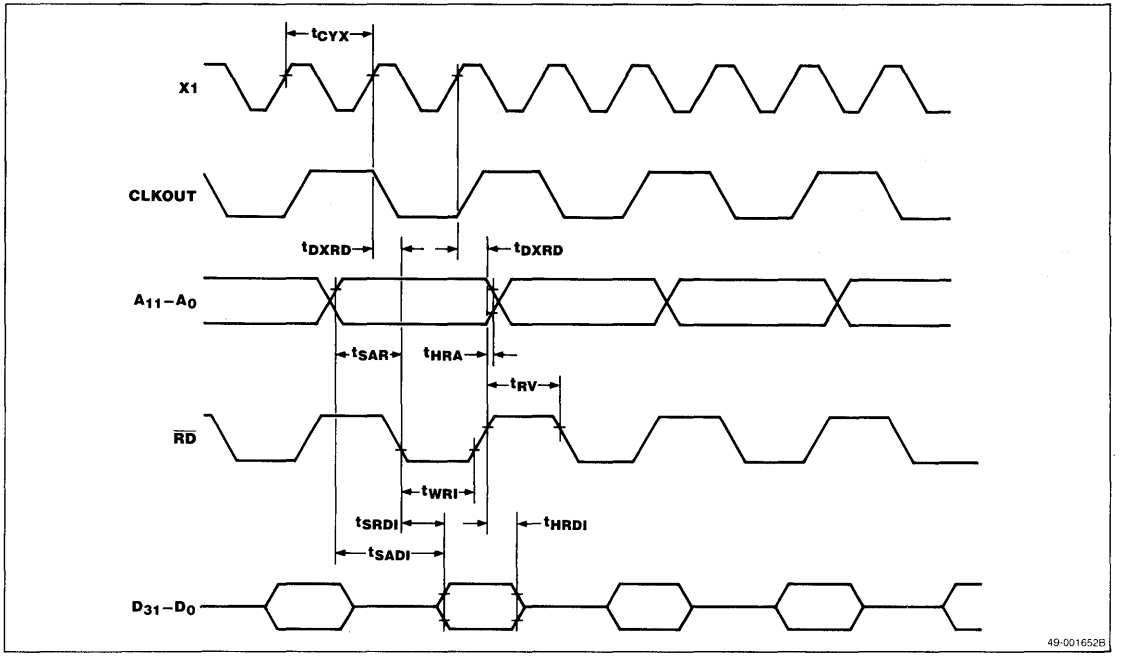


Notes: [1] SCK is an output
[2] SICK and SOCK are inputs [asynchronous with X1]

49-001648B

External Memory Access Timing Waveforms

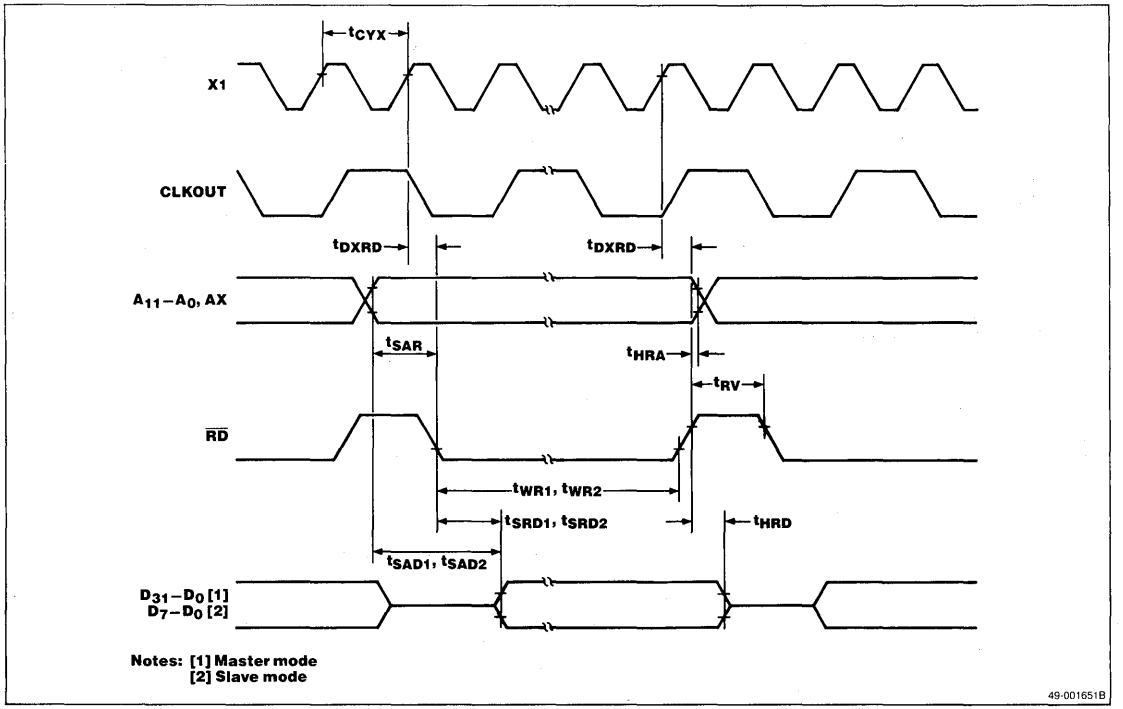
Instruction Read (Master)



49-001652B

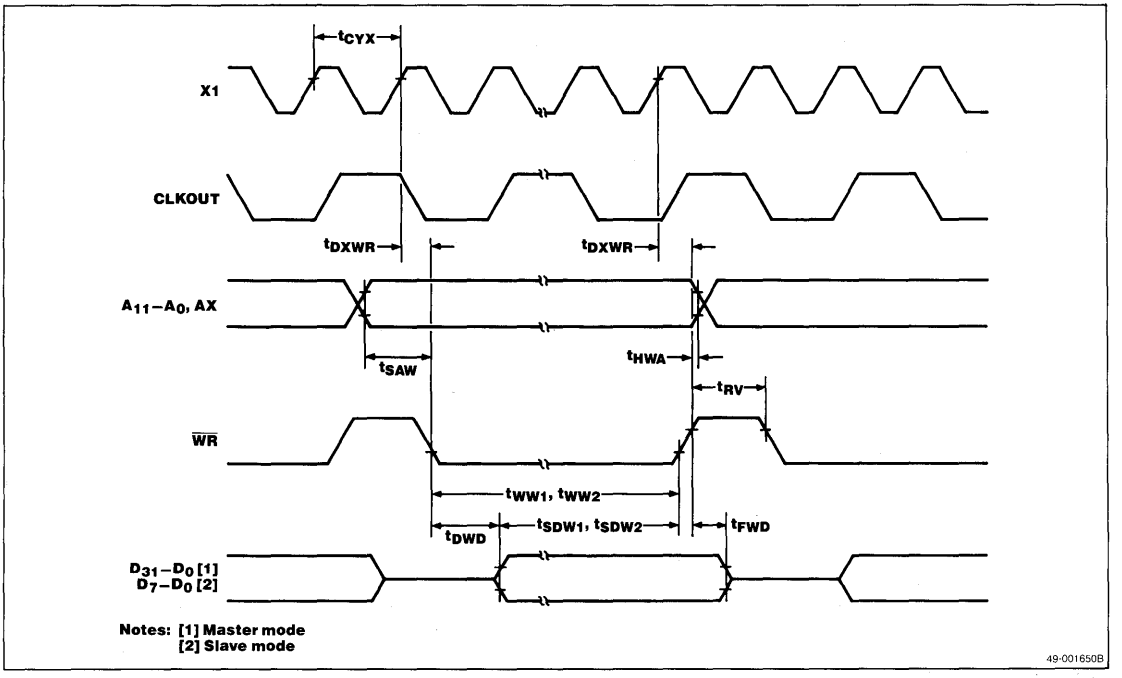
External Memory Access Timing Waveforms (cont)

Data Read

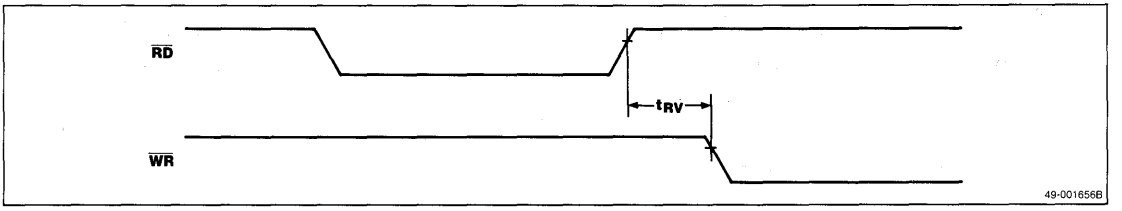


External Memory Access Timing Waveforms (cont)

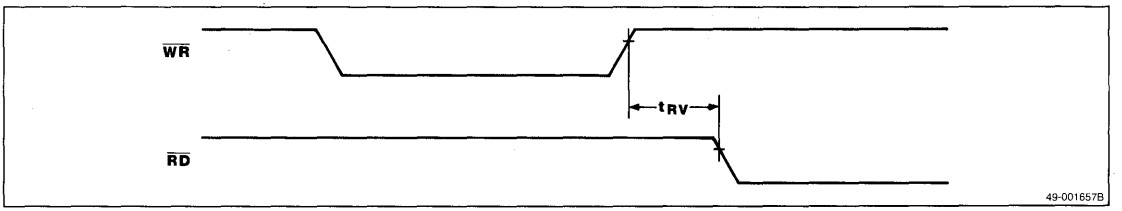
Data Write



Read → Write

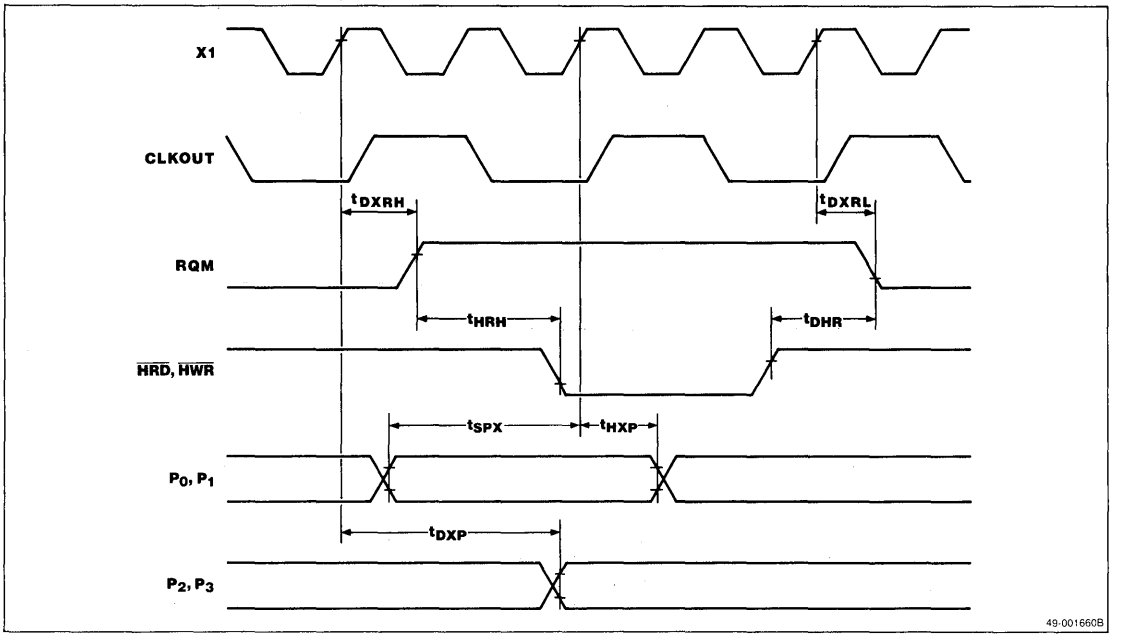


Write → Read

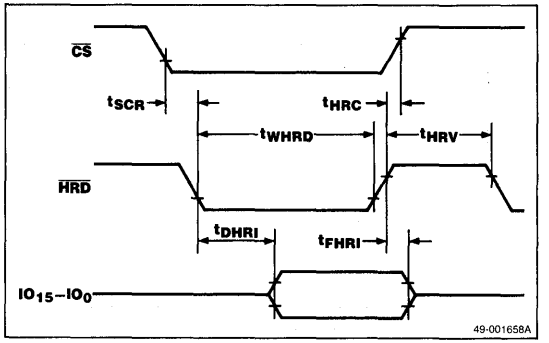


Host Interface Timing Waveforms, Slave Mode

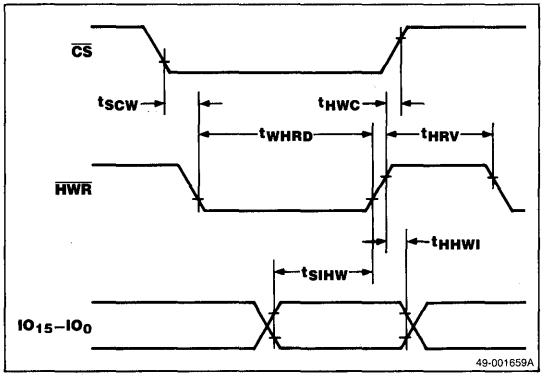
RQM Port



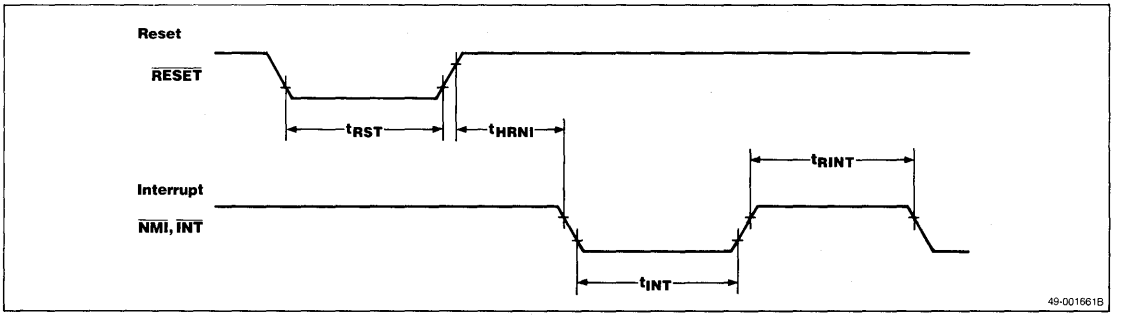
Host Read



Host Write

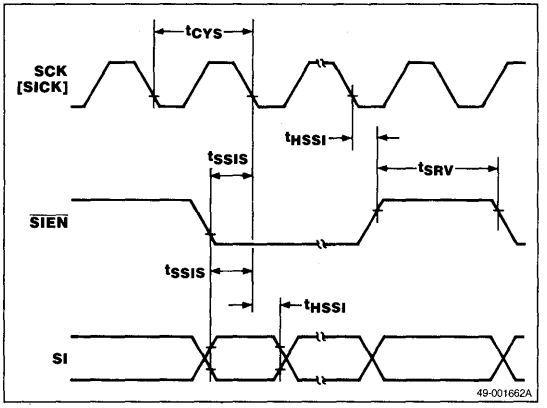


Interrupt Reset Timing Waveform

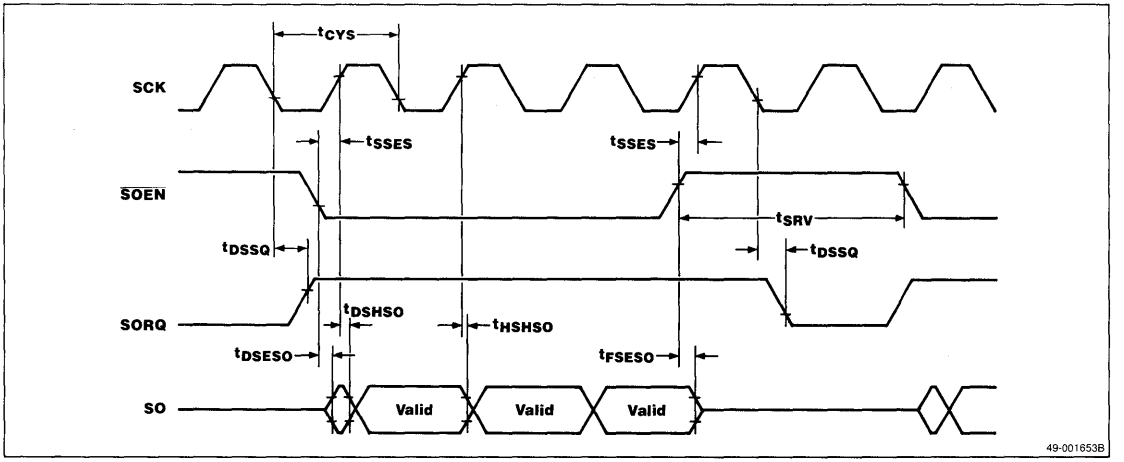


Serial Interface Timing Waveforms

Serial In

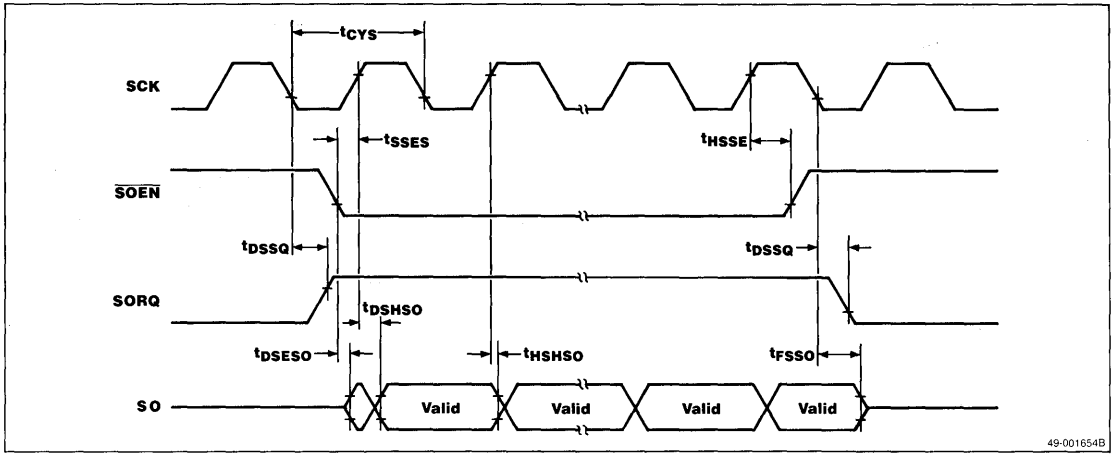


Serial Out, Case 1 (SOEN interrupt control)

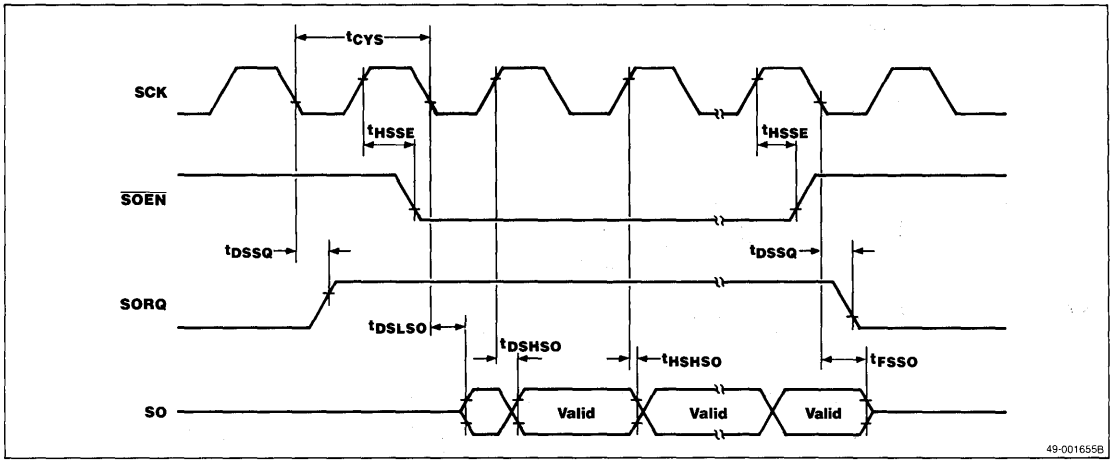


Serial Interface Timing Waveforms (cont)

Serial Out, Case 2 (SOEN control: SOEN low at SCK low)



Serial Out, Case 3 (SOEN control: SOEN low at SCK high)



μPD77810
Modem Digital Signal Processor

Description

The μPD77810 is a CMOS 16-bit signal processor designed for modem applications. It provides a compact digital signal processing system for modulation and demodulation and features low power consumption and high reliability at low cost. The μPD77810 consists of a dual processor and a modem function block. The dual processor comprises a μPD77C25 digital signal processor (DSP) and μCOM78K/I general purpose processor (GPP).

The μPD77810 is software compatible with both the μPD77C25 and μCOM78K/I families.

Features

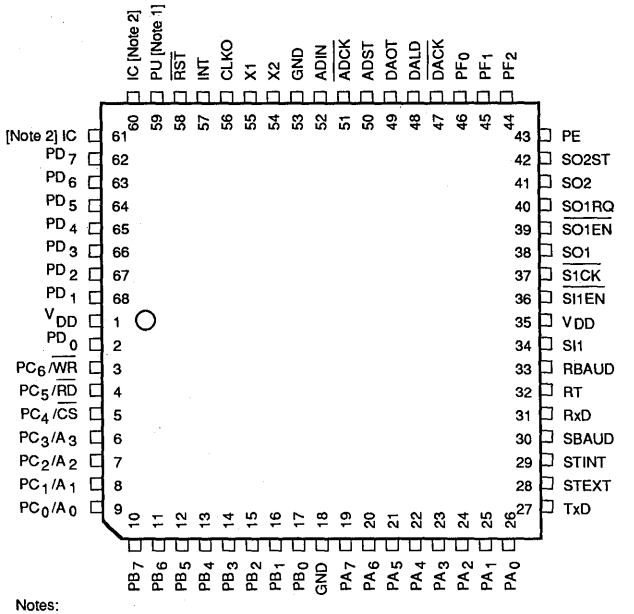
- Dual Processor
  - DSP (μPD77C25)
    - Minimum instruction execution time (181 ns with 5.5296 MHz clock)
    - Dedicated built-in 16-bit multiplier (31 bits)
    - Instruction ROM (2048 words x 24 bits)
    - Data ROM (1024 words x 16 bits)
    - Data RAM (256 words x 16 bits)
  - GPP (μCOM78K/I)
    - Minimum instruction execution time (362 ns with 5.5296 MHz clock)
    - Memory mapped built-in peripheral hardware (special function register)
    - Powerful interrupt functions
    - Non-maskable interrupt (1 type)
    - Maskable interrupt (9 types)
    - Internal ROM (16,384 words x 8 bits)
    - Internal RAM (192 words x 8 bits)
    - Control RAM (16 words x 8 bits)
- Modem Function Block
  - Built-in scrambler and descrambler
    - CCITT V Series Recommendations
    - Built-in hardware for the V.22, V.22bis, V.26, V.27, V.27bis, V.27ter, V.29 and V.32
  - Built-in transmit and receive PLLs (TxPLL and RxPLL)
  - Built-in synchronous/asynchronous serial communication interfaces (ASC, SAC, and UART)
  - Built-in A/D and D/A converter serial interfaces (8 or 16 bits)
- Software Compatibility
  - DSP (μPD77C25)
    - Compatible at assembler source program level
    - Upward compatible with the μPD7720 at assembler source program level
  - GPP (μCOM78K/I)
    - Compatible at assembler source program level
- Built-in clock generator (11.0592 MHz)
- CMOS
- Single +5 V power supply
- 68-pin PLCC
- 68-pin PGA

Ordering Information

Part Number	Package Type
μPD77810L	68-pin PLCC
μPD77810R	68-pin PGA

Pin Configurations

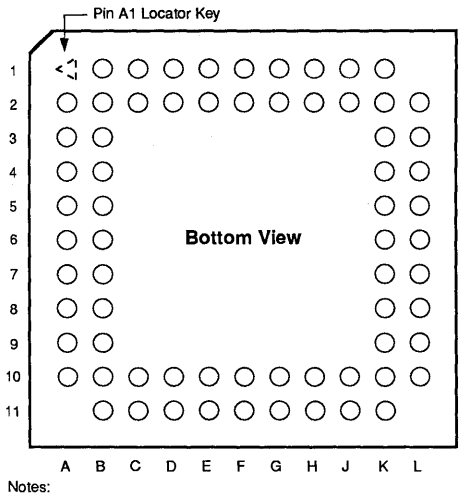
68-Pin PLCC



Notes:
[1] In normal operation connect the Pull Up (PU) pin to the V_{DD} pin through a pull up resistor.
[2] In normal operation the Internal Connection (IC) pin must be open.

83ML-5819B

68-Pin PGA



Pin	Symbol	Pin	Symbol	Pin	Symbol	Pin	Symbol
A2	PE	B9	SBAUD	F10	PA_7	K4	PD_3
A3	SO2	B10	STEXT	F11	GND	K5	PD_1
A4	$\overline{SO1EN}$	B11	PA_0	G1	X1	K6	PD_0
A5	$\overline{S1CK}$	C1	PF_0	G2	X2	K7	$PC_5/\overline{RD}$
A6	V_{DD}	C2	$\overline{DACK}$	G10	PB_0	K8	PC_3/A_3
A7	RBAUD	C10	PA_1	G11	PB_1	K9	PC_1/A_1
A8	RxD	C11	PA_2	H1	CLKO	K10	PB_6
A9	STINT	D1	DALD	H2	INT	K11	PB_7
A10	TxD	D2	DAOT	H10	PB_2	L2	IC [Note 2]
B1	PF_1	D10	PA_3	H11	PB_3	L3	PD_6
B2	PF_2	D11	PA_4	J1	$\overline{RST}$	L4	PD_4
B3	SO2ST	E1	ADST	J2	PU [Note 1]	L5	PD_2
B4	SO1RQ	E2	$\overline{ADCK}$	J10	PB_4	L6	V_{DD}
B5	SO1	E10	PA_5	J11	PB_5	L7	$PC_6/\overline{WR}$
B6	$\overline{SI1EN}$	E11	PA_6	K1	IC [Note 2]	L8	$PC_4/\overline{CS}$
B7	SI1	F1	ADIN	K2	PD_7	L9	PC_2/A_2
B8	RT	F2	GND	K3	PD_5	L10	PC_0/A_0

Notes:
[1] In normal operation connect the Pull Up (PU) pin to the V_{DD} Pin through a pull up resistor.
[2] In normal operation the Internal Connection (IC) pin must be open.

83ML-5904B

Pin Identification

Symbol	I/O	Function
General-Purpose Parallel Port		
A_0-A_3/ PC_0-PC_3	In	Address A0 to A3: Address input. Used to specify the C-RAM address.
$\overline{CS}$/PC_4	In	Chip Select: Chip Select Input.
D_0-D_7/ PD_0-PD_7	I/O	Data Bus D_0-D_7: Data Bus. When specified as a bus by the PCMR register, D_0-D_7 are used as a tri-state data bus. In this case, port C is used as an address bus or control bus, or inputs a Chip Select signal.
PA_0-PA_7	I/O	Port A: 8-bit general-purpose I/O port. I/O is selectable on a four-bit basis. It can be specified by the PTMR register.
PB_0-PB_7	I/O	Port B: 8-bit general-purpose I/O port. I/O is selectable on a two-bit basis. It can be specified by the PTMR register.
PC_0-PC_6	I/O	Port C: 7-bit general-purpose I/O port. I/O is selectable as a general-purpose I/O port on a bit basis. Port C inputs an address or a read/write signal from the host computer when port D is used as the bus. I/O can be specified by the PCMR register.
PD_0-PD_7	I/O	Port D: 8-bit general-purpose I/O port. I/O is selectable as a general-purpose I/O port on a bit basis. It can be specified by the PDMR register. Port D has a data bus function, transferring data to and from an external unit in the 16-byte C-RAM space. The bus/port can be specified by the PCMR register.
PE	In	Port E: 1-bit general-purpose input port.
PF_0-PF_2	Out	Port F: 3-bit general-purpose input port.
$\overline{RD}$/PC_5	In	Read Strobe: Used to input Read Strobe from the host computer.
$\overline{WR}$/PC_6	In	Write Strobe: Used to input Write Strobe from the host computer.
General-Purpose Serial Port		
SI1	In	Serial Input 1: General-purpose serial input pin (16 bits). The pin reads data input to $\overline{SI1}$ in synchronization with the rising edge of the $\overline{S1CK}$ serial clock when the $\overline{S1EN}$ pin is 0.
$\overline{SI1CK}$	In	Serial Clock for SI1 and SO1: Input pin for SI1 and SO1 serial clock. The I/O serial data is in synchronization with $\overline{S1CK}$.
$\overline{SI1EN}$	In	Serial Input 1 Enable: SI1 serial input enable pin. When this pin is 0, SI1 serial input is enabled.
SO1	Out	Serial Output 1: General-purpose serial output pin (16 bits). The pin outputs data in synchronization with the falling edge of the $\overline{S1CK}$ serial clock when the $\overline{SO1EN}$ pin is 0.

Symbol	I/O	Function
General-Purpose Serial Port (cont)		
$\overline{SO1EN}$	In	Serial Output 1 Enable: Enable pin for SO1 serial input. When this pin is 0, SO1 serial output is enabled.
SO1RQ	Out	Serial Output 1 Request: Request pin for SO1 serial output. This pin is set to 1 when a serial output instruction to SO1 is executed. When inverted, SO1RQ can be input to $\overline{SO1EN}$.
SO2	Out	Serial Output 2: DSP serial output pin (16 bits). This pin outputs serial data with an instruction in synchronization with the falling edge of the $\overline{ADCK}$ serial clock when SO2ST is 1.
SO2ST	Out	Serial Output 2 Strobe: Request pin for SO2 serial output. This pin is set to 1 when a serial output instruction to SO2 is executed.
A/D and D/A Serial Interface		
$\overline{ADCK}$	Out	A/D Serial Clock: A/D conversion serial clock. Data is input to the ADIN pin in synchronization with the falling edge of the $\overline{ADCK}$.
ADIN	In	A/D Data Input: Input pin for A/D conversion data. Data input to this ADIN pin is input from MSB in synchronization with the rising edge of the $\overline{ADCK}$ serial clock when ADST is 1. The ADIN pin is serial input of the DSP portion.
ADST	Out	A/D Start Strobe: Output pin for A/D conversion start strobe. This ADST pin is enable signal for the ADIN serial input. It can combine receive PLL with $\overline{ADCK}$.
$\overline{DACK}$	Out	D/A Serial Clock: D/A conversion serial clock. Data is output from the DAOT pin in synchronization with the falling edge of $\overline{DACK}$.
DALD	Out	D/A Data Load Strobe: Output pin for D/A conversion load strobe. This pin can combine transmit PLL together with $\overline{DACK}$.
DAOT	Out	D/A Data Output: Output pin for D/A conversion data. The pin outputs D/A conversion data from MSB in synchronization with the falling edge of the $\overline{DACK}$ serial clock when DALD pin output is 1.
Serial Control		
RBAUD (PG_0)	I/O	RX Baud Rate Clock: Received data baud rate clock output. This pin is also used as an input port (PG_0) depending on the PLLMR1 mode setting.
RT	Out	RX Clock: Received data bit rate clock output.
RxD	Out	Received Data: Received data serial output or output port. The received data is output from LSB using the bit string synchronous to the RT received clock as a start-stop signal.

Pin Identification (cont)

Symbol	I/O	Function
Serial Control (cont)		
SBAUD (PG_1)	I/O	TX Baud Rate Clock: Transmitted data baud rate clock output. This pin is also used as an input port (PG_1) depending on the PLLMR1 mode setting.
STEXT	In	TX Clock External: Transmitted data bit rate clock input.
STINT	Out	TX Clock Internal: Transmitted data bit rate clock output.
TxD	In	Transmitted Data: Transmitted data serial input or input port. The transmitted data is input from LSB using the start-stop signal input to the TxD pin as a transmit clock or in synchronization with STINT or STEXT.
Circuit Control		
CLKO	Out	Clock Out: CLKO is a 3.6864 MHz (50% duty) output pin, one third of a system clock (11.0592 MHz).
INT	In	Interrupt: Maskable interrupt input. The interrupt address is 14H.
$\overline{RST}$	In	Reset: Low-level active system reset input. $\overline{RST}$ takes priority over any other operations. After reset, the GPP and DSP start programs from address 0.
X1	In	Crystal oscillator (11.0592 MHz ± 100 ppm) connector.
X2	Out	
V_{DD}		Power Supply: +5 V ±10%.
GND		Ground: GND (common).

Absolute Maximum Ratings

T_A = 25 °C

Supply voltage, V_{DD}	−0.5 to +7.0 V
Input voltage, V_I	−0.5 to V_{DD} +0.5 V
Output voltage, V_O	−0.5 to V_{DD} +0.5 V
Operating temperature, T_{OPT}	−10 to +70 °C
Storage temperature, T_{STG}	−65 to +150 °C

Exposure to Absolute Maximum Ratings for extended periods may affect device reliability; exceeding the ratings could cause permanent damage.

Capacitance

T_A = 25 °C; V_{DD} = 0 V

Parameter	Symbol	Min	Max	Unit	Conditions
X1, SCK capacitance	C_ϕ		20	pF	f_C = 1 MHz. All pins are grounded except measuring pins.
Input capacitance	C_I		20	pF	
Output capacitance	C_O		20	pF	

DC Characteristics

T_A = −10 to +70 °C; V_{DD} = +5 V ±10%; f_{OSC} = 11.0592 MHz

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
Input voltage, low	V_{IL}	−0.3		0.8	V	
Input voltage, high	V_{IH}	2.2		V_{DD} + 0.3	V	
X1 input voltage low	V_{ILC}	−0.3		0.8	V	
X1 input voltage high	V_{IHC}	2.2		V_{DD} + 0.3	V	
Output voltage low	V_{OL}			0.45	V	I_{OL} = 2.0 mA
Output voltage, high	V_{OH}	0.7 V_{DD}			V	I_{OH} = −400 μA
Input leak current, low	I_{LIL}			−10	μA	V_I = 0 V
Input leak current, high	I_{LIH}			10	μA	V_I = V_{DD}
Output leak current,low	I_{LOL}			−10	μA	V_O = 0.47 V
Output leak current, high	I_{LOH}			10	μA	V_O = V_{DD}
Supply current	I_{DD}		80		mA	

AC Characteristics

T_A = −10 to +70 °C; V_{DD} = +5 V ±10%

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
X1 cycle time	t_{CYC}		90		ns	11.0592 MHz ± 100 ppm
X1 pulse width, high	t_{CCH}		35		ns	
X1 pulse width, low	t_{CCL}		35		ns	
X1 rise time	t_{CR}			10	ns	(Note 4)
X1 fall time	t_{CF}			10	ns	
CLKO cycle time	t_{COCY}		271		ns	
CLKO width, high	t_{COCH}		115		ns	
CLKO width, low	t_{COCL}		115		ns	
Address set time for $\overline{RD}$	t_{AR}	0			ns	
Address hold time for $\overline{RD}$	t_{RA}	0			ns	
$\overline{RD}$ width	t_{RR}	170			ns	
Data access time $\overline{RD}$	t_{RD}		110		ns	C_L = 100 pF
Data float time for $\overline{RD}$	t_{DF}	0		50	ns	C_L = 20 pF, R_L = 2 kΩ
Access set time for $\overline{WR}$	t_{AW}	0			ns	
Address hold time for $\overline{WR}$	t_{WA}	0			ns	
$\overline{WR}$ pulse width	t_{WW}	150			ns	
Data set time $\overline{WR}$	t_{DW}	100			ns	
Data hold time $\overline{WR}$	t_{WD}	0			ns	
$\overline{RD}$ and $\overline{WR}$ recovery time	t_{RV}	180			ns	
$\overline{ADCK}$ cycle time	t_{ADCY}		1065		ns	
$\overline{ADCK}$ pulse width, high	t_{ADCH}		532		ns	
$\overline{ADCK}$ pulse width, low	t_{ADCL}		532		ns	
$\overline{DACK}$ cycle time	t_{DACY}		1085		ns	
$\overline{DACK}$ pulse width, high	t_{DACH}		532		ns	
$\overline{DACK}$ pulse width, low	t_{DACL}		532		ns	
Serial I/O request delay time	t_{DRQ}	50		150	ns	
Serial input set time for $\overline{SCK}$	t_{DC}	50			ns	
Serial input hold time for $\overline{SCK}$	t_{CD}	30			ns	
SO1EN set time for $\overline{SCK}$	t_{SOC}	50			ns	
SO1EN hold time for $\overline{SCK}$	t_{CSO}	30			ns	
Serial output delay time for $\overline{SCK}$	t_{DCK}			60	ns	
Serial output hold time for $\overline{SCK}$	t_{HCK}	0			ns	
Serial output float time for $\overline{SCK}$	t_{HZCK}			60	ns	
Reset pulse width	t_{RST}	10			μs	
INT pulse width	t_{INT}	8		t_{CYC}		

Notes:

(1) $\overline{SCK}$ includes $\overline{S1CK}$, $\overline{ADCK}$, and $\overline{DACK}$.

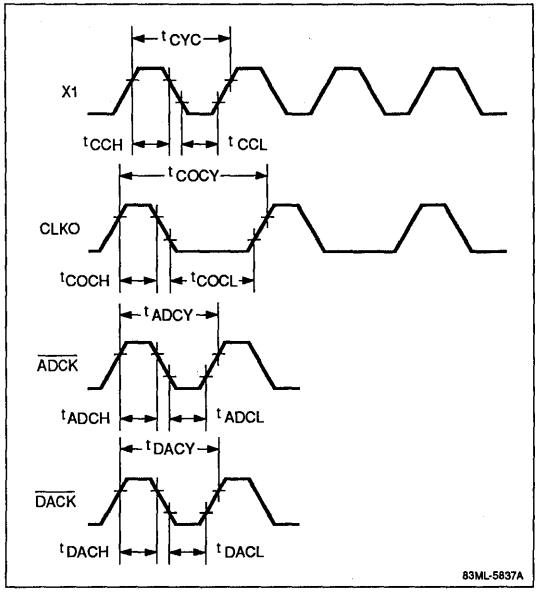
(2) Serial input includes ADIN and SI1.

(3) Serial output includes DAOT, SO1, and SO2.

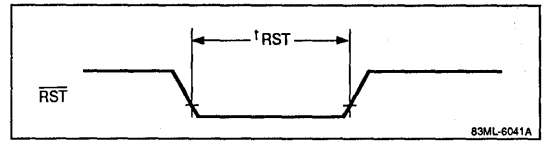
(4) Voltage at timing measuring point: 1.0 V and 3.0 V.

Timing Waveforms

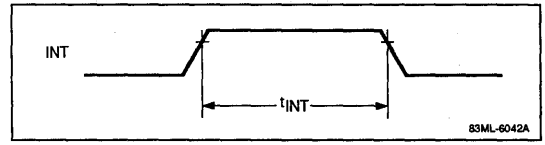
Clock



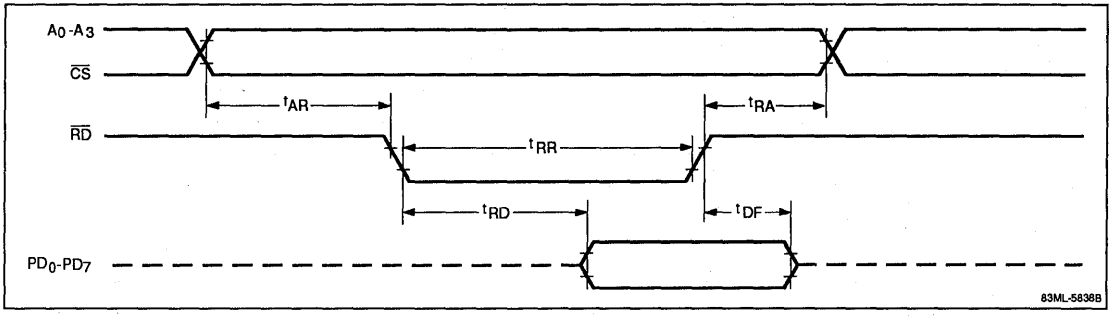
Reset



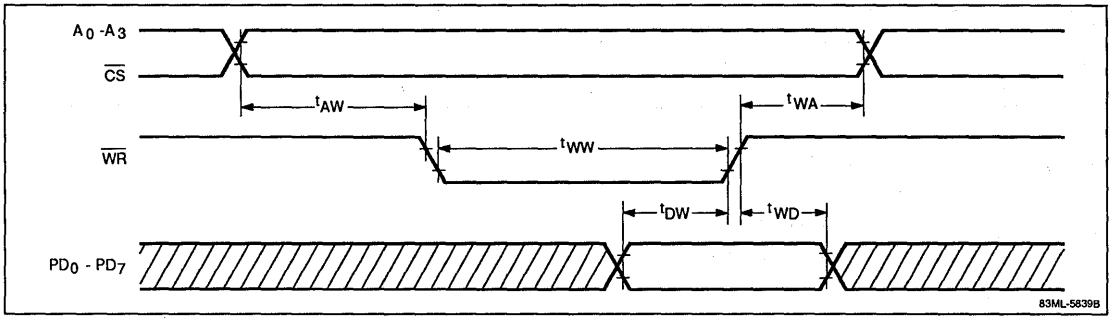
Interrupt



Read Operation

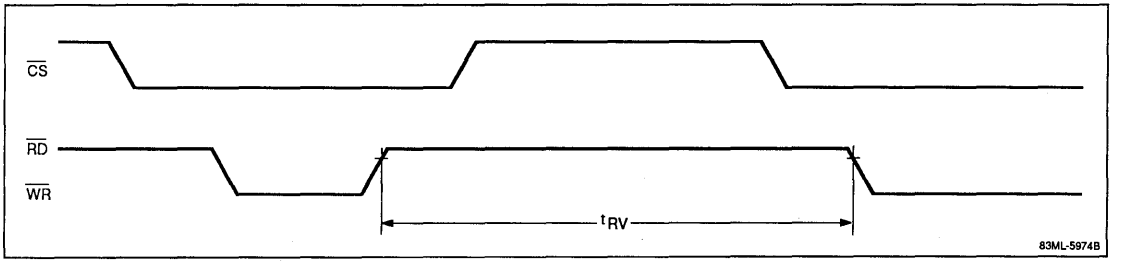


Write Operation

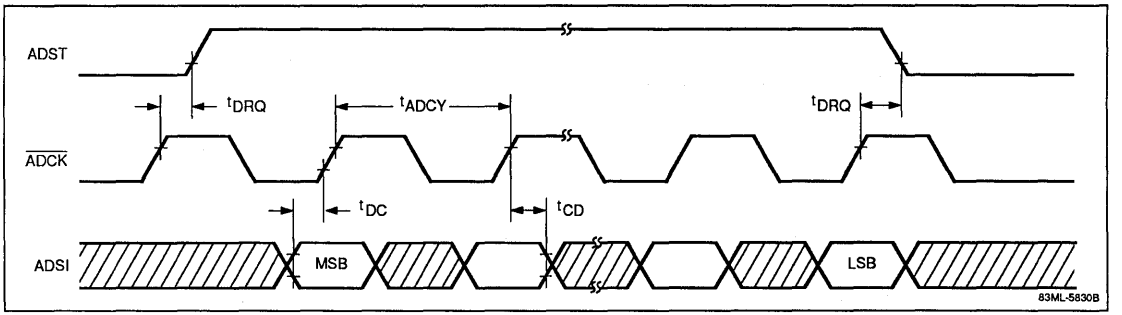


Timing Waveforms (cont)

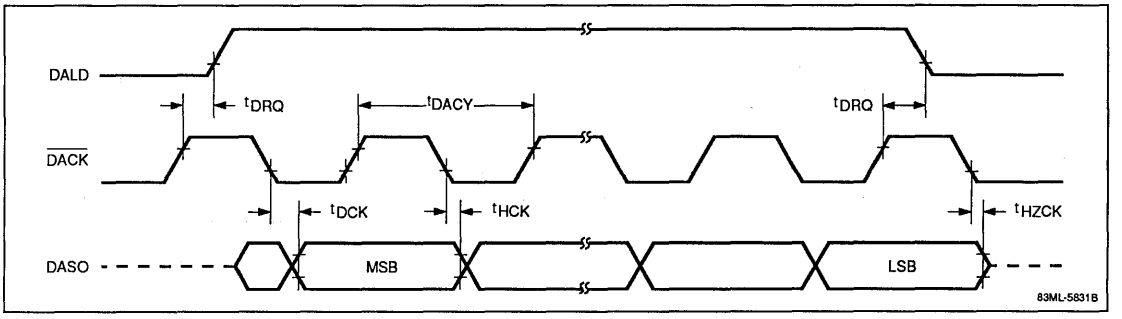
Read/Write Cycle Timing



A/D Serial Input

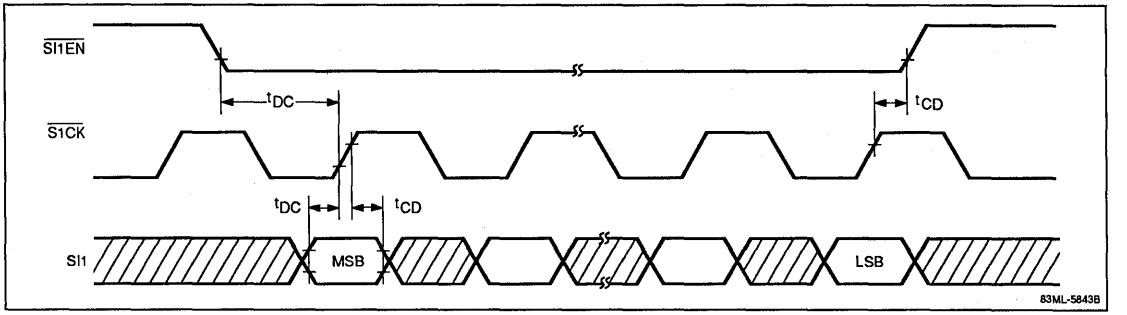


D/A Serial Output

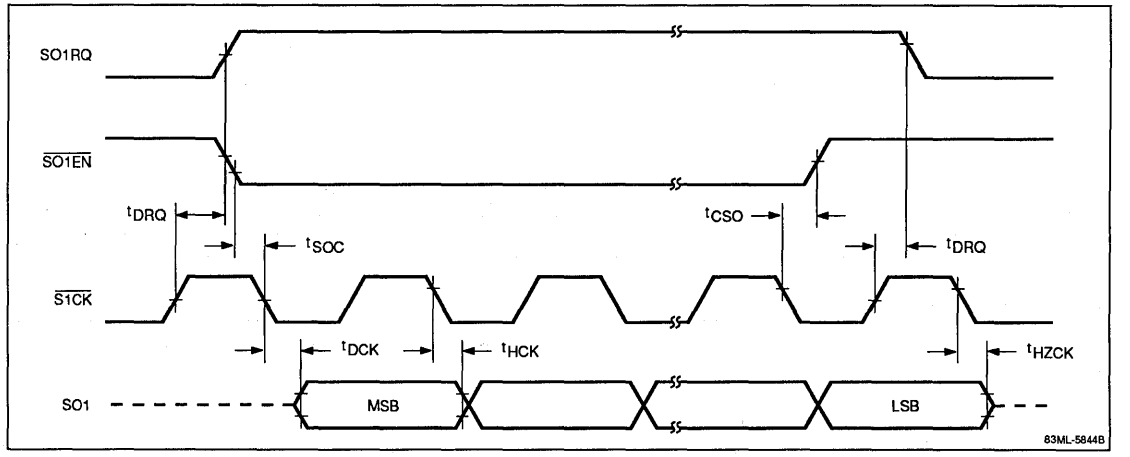


Timing Waveforms (cont)

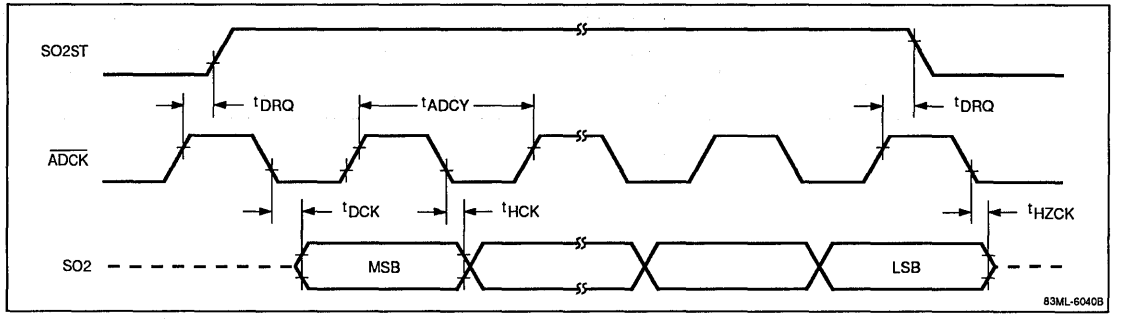
Serial Input SI1



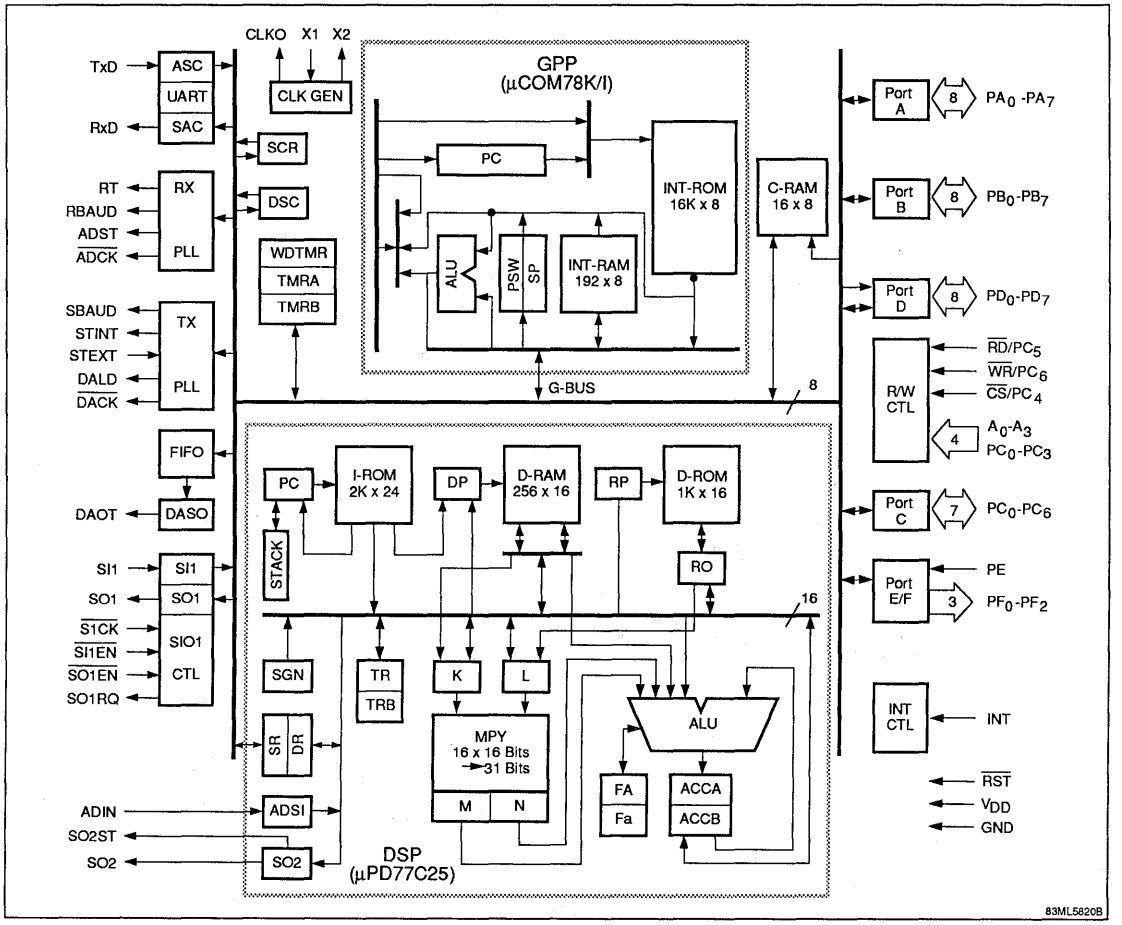
Serial Output SO1



Serial Output SO2



Block Diagram



μPD77810 Functional Units

The μPD77810 contains the following functional units:

- DSP (μPD77C25)
- GPP (μCOM78K/I)
- Modem Function Block
  - Timers: WDTMR and TMR
  - Control RAM
  - Scrambler and Descrambler
  - UART, SAC, and ASC
  - Phase-Locked Loops: TxPLL and RxPLL
  - Interface to A/D and D/A
  - Serial I/O
  - Parallel I/O

Figure 1 shows an overview of the μPD77810. Figure 2 shows the functional pin groups of the μPD77810.

DSP FUNCTIONAL DESCRIPTION

Figure 3 is the block diagram of the DSP. The DSP consists of the following:

- Multiplier
- ALU Peripheral
- Data Memory with Data ROM and RAM
- Instruction ROM
- Parallel Interface
- Serial Interface
- G-bus Interface

Figure 1. Overview of the μPD77810

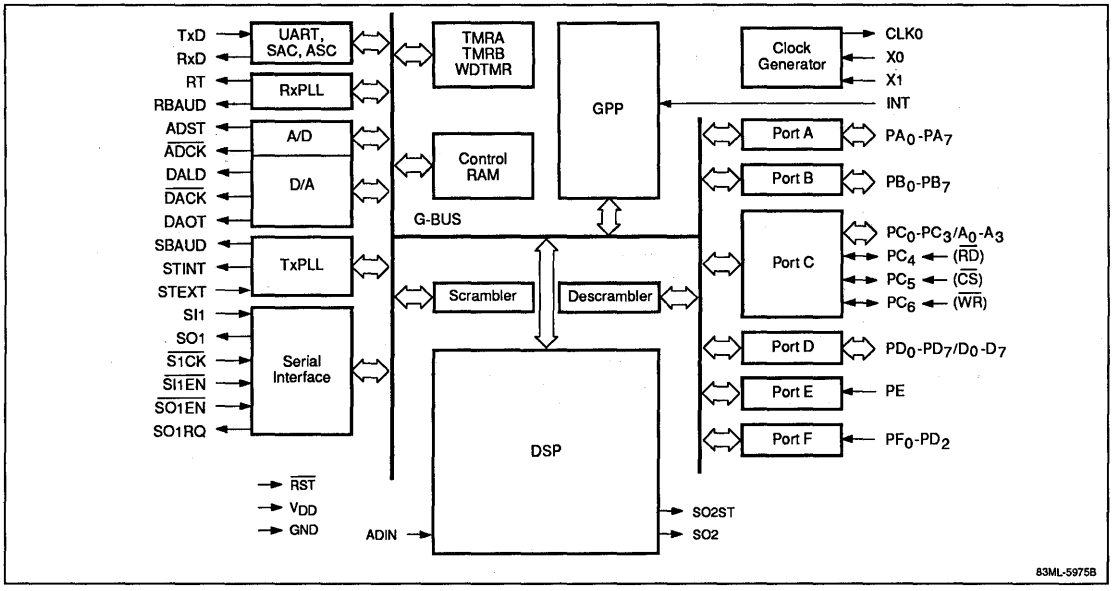
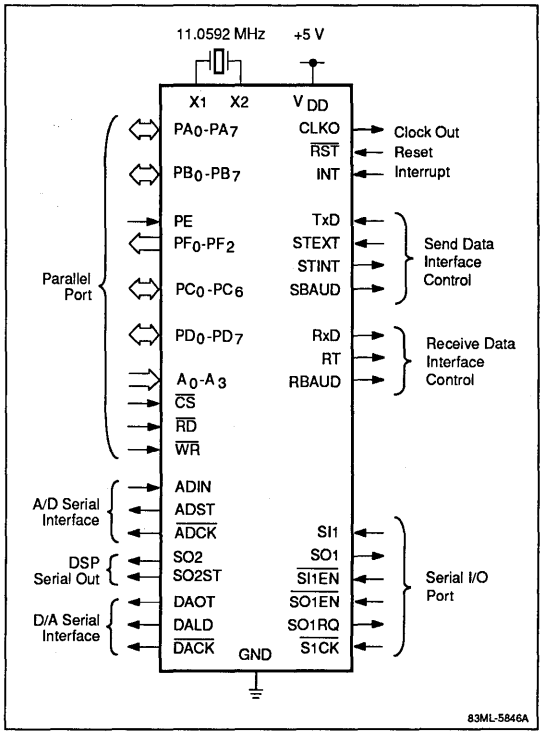


Figure 2. Functional Pin Groups of the μPD77810



Differences Between the μPD77810 and μPD7720 and μPD77C25 Families.

The DSP was designed on the basis of the μPD7720 and μPD77C25 16-bit signal processor families, allowing the μPD77810 to be compatible with these families at the assembler source program level. Table 1 lists the differences between the μPD77810 and the μPD7720 and μPD77C25 families.

DSP Internal Functions

Instruction ROM. The instruction ROM is a 2048 word x 24 bit mask programmable ROM that stores programs. Its addressing is generated by the Program Counter (PC).

Program Counter [PC]. The program counter is an 11-bit binary counter that addresses the instruction ROM. The PC is incremented during every instruction fetch cycle and instructions are read from the ROM sequentially. When a jump or subroutine call instruction is executed, the contents of the address field (NA field) of the instruction are transferred to the PC. When a return instruction is executed, the contents of the stack register are transferred to the PC and when a interrupt is issued, the fixed address 100H is transferred.During a reset, the PC is set to the start address 000H.

Stack. The 4 x 11 bit stack memory stores the return address when a subroutine call instruction is executed or an interrupt is issued. It has a four-level last-in first-out (LIFO) memory. When a return instruction is executed, the return address is read from the stack memory to the PC.

RAM. The 256 word x 16 bit RAM stores data. Its address is set by the data pointer (DP). Data is transferred between the RAM and internal data bus and also to the ALU P input. Data at the RAM address specified as $DP_6 = 1$ can be directly output to the K register.

Data Pointer [DP]. The 8-bit data pointer specifies the RAM address. The DP is connected to the low-order eight bits of the internal data bus and is transferred to and from other registers via the bus.

Figure 3. DSP Block Diagram

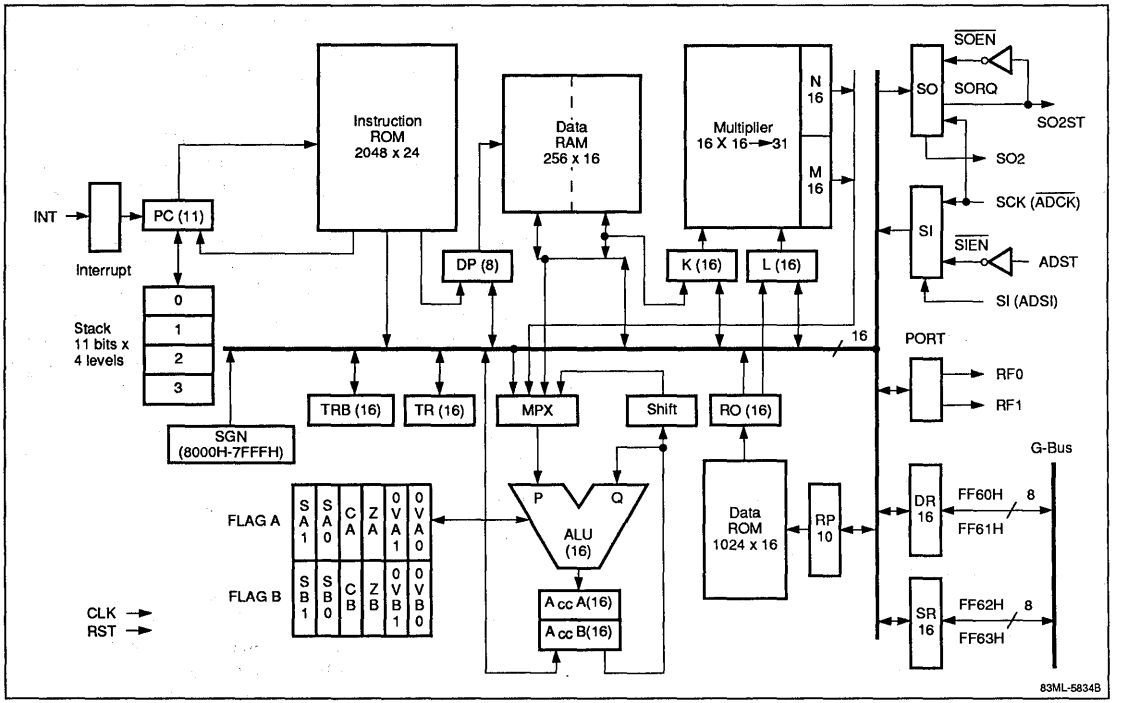


Table 1. Differences Between the μPD77810 and the μPD7720 and μPD77C25 Families

Member		μPD7720	μPD77C25	μPD77810 DSP
Memory	Instruction ROM	512 x 23 bits	2048 x 24 bits	2048 x 24 bits
	Data ROM	510 x 13 bits	1024 x 16 bits	1024 x 16 bits
	RAM	128 x 16 bits	256 x 16 bits	256 x 16 bits
Registers	PC	9 bits	11 bits	11 bits
	STACK	9 bits x 4 levels	11 bits x 4 levels	11 bits x 4 levels
	RP	9 bits	10 bits	10 bits
	RO	13 bits	16 bits	16 bits
	DP	7 bits	8 bits	8 bits
	Additional register		TRB	TRB
Instruction length		23 bits (DP$_H$/M field, 3 bits)	24 bits (DP$_H$/M field, 4 bits)	24 bits (DP$_H$/M field, 4 bits)
Additional instruction			JDPLNO JDPLNF M8-MF (DP modified)	JDPLNO JDPLNF M8-MF (DP modified)

Table 1. Differences Between the μPD77810 and the μPD7720 and μPD77C25 Families (cont)

Member	μPD7720	μPD77C25	μPD77810 DSP
DMA mode	Available	Available	Unavailable
Operation clock (instruction cycle)	8.192 MHz (244 ns)	8.192 MHz (122 ns)	5.5296 MHz (181 ns)
Other			• SR (status register) bits 0 and 1 have been changed to R_XPLL decremental data setting port output. • SR (status register) bit 11 has been changed to USFO.

The high-order four bits (DP_H) of DP can be modified by exclusive OR of four bits of the DP_H/M field in an instruction.

The low-order four bits (DP_L) of DP are assigned to an increment/decrement counter. The DP increments, decrements, or clears DP_L field of an instruction.

Data ROM. The 1024 word x 16 bits mask ROM stores fixed data; for example, digital filter coefficients and data used to decode μ-law or A-law compressed non-linear data. The data ROM address is set by the RP register. ROM data is output to the internal data bus via the RO register.

Addresses 0 and 1 that were not accessible to the μPD7720 family user are available for the μPD77810.

ROM Pointer [RP]. The ROM pointer specifies the data ROM address. RP consists of a 10-bit decrement counter. It can transfer data to and from the low-order ten bits of the internal data. The RP register can be decremented by the RPDCR bit of an instruction.

ROM Output Buffer [RO]. The ROM output buffer (RO) is a 16-bit register that stores the ROM output data. RO data is output to the internal data bus or directly output to the L register.

Multiplier. The parallel multiplier using the Second Order Booth algorithms multiplies 16-bit data of two's compliments notation. The result is a sign bit plus 30 bits of data. The sign bit plus the low-order 15 bits are output to the M register and the lower-order 15 bits without the sign bit are output to the high-order of the N register. Bit 0 of the N register is set to 0. The multiplier inputs data from the K and L registers.

K and L Registers. The K and L registers are 16-bit registers that store the multiplier and multiplicand that are to be input to the multiplier. The K register also inputs RAM output data and the L register inputs data ROM output data. Immediately after input data is set in the K and L registers, it is input to the multiplier for processing.

M and N Registers. The M and N registers are multiplier output registers. Of the multiplier result , the signed bit and the high-order 15 bits are output to the M register and the low-order 15 bits are output to the high-order of the N register. Bit 0 of the N register is set to 0. The M and N register output is connected to the ALU P input.

ALU, A_{CC}A and A_{CC} B. The ALU is a 16-bit arithmetic and logical unit, which performs the following operations for its P and Q data inputs:

- OR
- AND
- XOR (Exclusive OR)
- SUB
- ADD
- Shift [A_{CC}A, A_{CC}B only]
- 1's complement [A_{CC}A, A_{CC}B only]

P input: RAM, internal data bus, M register, N register, shift register, 0000H
Q input: A_{CC}A, A_{CC}B

A_{CC}A and A_{CC}B are 16-bit registers that store the result of the ALU operation. It can also input data from the internal data bus. The ASL bit of an instruction specifies whether the ALU output is input to A_{CC}A or A_{CC}B. Register data can be output to the internal data bus or to the shift register together with the ALU Q input.

Shift. The shift register shifts 16-bits of data that is input from $A_{CC}A$ and $A_{CC}B$. One-bit right shifting, left shifting on one-, two, and four bit basis, and 8-bit replacement are available.

Flag A and Flag B Registers. Flag A is a register used to store flags generated when $A_{CC}A$ is selected. Similarly, flag B is the register which stores flags when $A_{CC}B$ is selected. Table 2 shows the flags changed by the results of ALU operations. The flag A and flag B register contain flag bits as shown below.

FLAG A	SA1	SA0	CA	ZA	OVA1	OVA0
FLAG B	SB1	SB0	CB	ZB	OVB1	OVB0

CA and CB [Carry]: CA and CB are flags that store the carries that occur from the results of an operation. The operations are SUB, ADD, SBB, ADC, DEC, and INC.

ZA and ZB [Zero]: When data to be stored in the A_{CC} is 0 after an operation, excluding NOP, a 1 is set in the ZA or ZB flag.

SA0 and SB0 [Sign 0]: SA0 and SB0 store the MSB of the data to be stored in A_{CC}, when an operation excluding NOP is executed.

OVA0 and OVB0 [Overflow 0]: OVA0 and OVB0 store the exclusive ORed results of carries that occur in ALU bits 15 and 14 when SUB, ADD, SBB, ADC, DEC, or INC is executed.

OVA1 and OVB1 [Overflow 1]: OVA1 and OVB1 flags are designed for effective overflow processing from the results of up to three operations. The operations are SUB, ADD, SBB, ADC, DEC, and INC.

SA1 and SB1 [Sign1]: SA1 and SB1 are used in conjunction with OVA1 and OVB1 flags. The flags are designed for effective overflow processing and indicate the direction in which the overflow occurred.

Table 2. Flags Changed by Results of ALU Operations

Mnemonic	SA1/ SB1	SA0/ SB0	CA/ CB	ZA/ ZB	OVA1/ OVB1	OVA0/ OVB0
NOP	•	•	•	•	•	•
OR	X	$	0	$	0	0
AND	X	$	0	$	0	0
XOR	X	$	0	$	0	0
SUB	$	$	$	$	$	$
ADD	$	$	$	$	$	$
SBB	$	$	$	$	$	$
ADC	$	$	$	$	$	$
DEC	$	$	$	$	$	$
INC	$	$	$	$	$	$
CMP	X	$	0	$	0	0
SHR1	X	$	$	$	0	0
SHL1	X	$	$	$	0	0
SHL2	X	$	0	$	0	0
SHL4	X	$	0	$	0	0
XCHG	X	$	0	$	0	0

Symbols:
$ = The flag is changed by the result of operation.
• = The flag remains unchanged.
0 = Flag is reset.
X = Undefined

Temporary Register [TR and TRB]. TR and TRB are 16-bit general-purpose registers that can be used to latch data temporarily.

Sign Register [SGN]. The SGN register stores 8000H when the SA1 flag is 0 and 7FFFH when it is 1. If an overflow occurs, overflow correction can be performed with only one instruction.

Status Register [SR]. The SR register stores interface information for the GPP. Internally, it is handled as a 16-bit register. Of the 16 bits of data, eight bits can be read by the GPP by specifying the SFR address FF62H or FF6H.

The SR register consists of 16-bits as shown below.

MSB							LSB
RQM	USF2	USF1	DRS	USFO	DRC	SOC	SIC

MSB							LSB
EI	0	0	0	0	0	RF1	RF0

RF0 and RF1: RF0 and RF1 correspond to output ports RF0 and RF1. The values set in the bits are output directly to the ports.

Bits RF0 and RF1 specify the value to be set in the decrementer in RxPLL of the modem function block.

EI [Enable Interrupt]: The EI bit specifies whether an interrupt request input to the INT pin is enabled.
0 = Disabled
1 = Enabled

SIC [SI Control]: The SIC bit specifies the length of serial data to be input to the ADIN A/D conversion input pin.
0 = Serial input data is 16 bits
1 = Serial input data is 8 bits

SOC [SO Control]: The SOC bit specifies the length of serial data to be output to the SO serial output pin.
0 = Serial output data is 16 bits
1 = Serial output data is 8 bits

DRC [DR Control]: The DRC bit sets the DR register configuration for GPP as eight or 16 bits.
0 = The DR register is treated as a 16-bit register
1 = The DR register is treated as a 8-bit register.

DRS [DR Status]: The DRS bit indicates the DR register transfer status.
0 = End of data transfer
1 = Data is being transferred

When DRC = 1, the DRS bit is always set to 0.

USF0, USF1, and USF2 [User's Flag]: USF0, USF1, and USF2 are flag bits which can be used freely. They are used as a status bit in an interface with an external unit.

Request for Master [RQM]: RQM is a flag bit used to transfer data between the DR register and GPP.

Data Register [DR]. DR is a 16-bit register used to transfer data to and from the GPP. One of its sides is connected to the 8-bit bus and reads or writes data from an external unit in two operations. Internally, it transfers data in one operation (16 bits). When the DR register is defined as an 8-bit register by the DRC bit, only the low-order eight bits of DR can be transferred.

Serial Input Register [SI]. The SI register inputs serial data from an external unit. Serial data is input to DSP ADSI from the ADIN pin at the rising edge of the ADCK serial clock, converted to parallel data by SI, and output to the internal data bus with an instruction. Serial data can be handled from either the LSB or MSB.

Serial Output Register [SO]. The SO register loads parallel data to be output from the internal data bus, converts to serial data, and outputs to an external unit. Serial data can be handled from the either the LSB or MSB. It is output at the rising edge of the ADCK serial clock.

Interrupt. An interrupt is accepted with an instruction from the GPP, when interrupt is enabled (EI bit of SR register = 1). Program control jumps to the interrupt address 100H and executes an interrupt process.

Reset [RST]. $\overline{RST}$ initializes the following by SFR INTDSP0 (0) of the GPP:

- PC
- Flags A and B
- SR register
- ADSI ASK flag and SO ACK flag

DSP Instructions

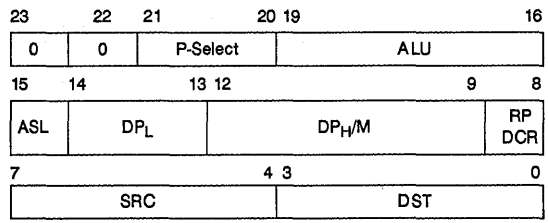
All DSP instructions consist of a single 24-bit word. Four types of instructions are available and are distinguished by the OP code which are the highest two bits of an instruction.

- OP instruction: Normal operations and transfer
- RT instruction: Return instruction
- JP instruction: Jump instructions including unconditional jump, conditional jump, and subroutine call
- LD instruction: Immediate data load instruction

See table 3 for DSP instruction codes.

OP Instruction. The OP instruction has the following functions:

- Performs operations specified by six fields and two bits.
- Increments the current address set in the program counter by one.



P-SELECT Field: The P-SELECT field selects ALU P input. See table 4 for P-SELECT field specifications.

ALU Field: The ALU field specifies an ALU operation. See table 5 for ALU field specifications.

ASL [A_{CC} Selection] Bit: The ASL bit specifies whether $A_{CC}A$ or $A_{CC}B$ is selected to the ALU input/output. See table 6 for ASL bit specifications.

DP_L Field: The DP_L field specifies the operation of the low-order four bits of the data pointer. The changed DP_L is valid from the next instruction. See table 7 for DP_L field specifications.

DP_H/MP [DP_H Modify] Field: The DP_H/M field modifies the high-order four bits of the data pointer. The OP instruction performs exclusive OR of DP_H four bits with the value in the field for each bit. The modified DP_H value is valid from the next instruction. See table 8 for DP_H/M field specifications.

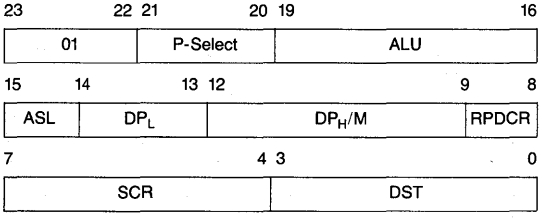
RPDCR [RP Decrement] Bit: The RPDCR bit specifies whether RP data is decremented or not decremented. The decremented value is valid from the next instruction. See table 9 for RPDCR bit specifications.

SRC [Source] Field: The SCR field specifies the register that outputs data to the internal data bus. See table 10 for SCR field specifications.

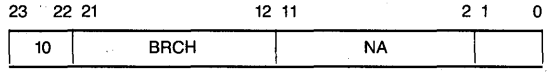
DST [Destination] Field: The DST field specifies the register that inputs data from the internal data bus. This is source data from the register specified in the SRC field. See table 11 for DST field specifications.

RT Instruction. The RT instruction has the following functions:

- Performs operations specified by six fields and two bits, similar to the OP instruction. Therefore, RT has the same function as that of the OP instruction.
- Sets the program counter as the stacked return address. See table 4 through table 11 for RT instruction specifications.



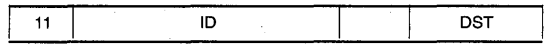
JP Instruction. The JP instruction includes three functions, such as unconditional jump, conditional jump, and subroutine call.



BRCH (Branch) Field: The BRCH field selects the instruction to be executed from unconditional jump, conditional jump, and subroutine call. See table 12 for BRCH field specifications.

NA (Next Address) Field: The NA field specifies the address of the jump destination. See table 13 for NA field specifications.

LD Instruction. The LD instruction transfers imediate data to the specified register.



ID (Immediate Data) Field: The 16-bit ID field sets immediate data. Immediate data is transferred to the register specified in the DST field. See table 14 for ID field specifications.

DST (Destination) Field: The DST field specifies the register where data in the ID field is transferred. The DST field is the same as that of the OP instruction. See table 11 for DST field specifications.

Table 3. DSP Instruction Codes

	OP Field		
Instruction	**23**	**22**	**Meaning**
OP	0	0	Operation and transfer
RT	0	1	Return
JP	1	0	Jump
LD	1	1	Immediate data loading

Table 4. P-Select Field Specifications

	P-Select Field		
Mnemonic	**21**	**20**	**ALU-P Input***
RAM	0	0	RAM
IDB	0	1	Internal data bus
M	1	0	M register
N	1	1	N register

Note:

* The input is valid when the ALU field specifies an instruction other than Shift, INC A_{CC}, DEC A_{CC}, and Complement A_{CC}.

Table 5. ALU Field Specifications

Mnemonic	ALU Field 19	18	17	16	Operation	
NOP	0	0	0	0	No operation	
OR	0	0	0	1	OR	$(A_{CC}) \leftarrow (A_{CC})\ V\ (P)$
AND	0	0	1	0	AND	$(A_{CC}) \leftarrow (A_{CC})\ V\ (P)$
XOR	0	0	1	1	Exclusive OR	$(A_{CC}) \leftarrow (A_{CC}) \forall (P)$
SUB	0	1	0	0	Subtract	$(A_{CC}) \leftarrow (A_{CC}) - (P)$
ADD	0	1	0	1	Add	$(A_{CC}) \leftarrow (A_{CC}) + (P)$
SBB	0	1	1	0	Subtract with borrow	$(A_{CC}) \leftarrow (A_{CC}) - (P) - (C)$
ADC	0	1	1	1	Add with carry	$(A_{CC}) \leftarrow (A_{CC}) + (P) + (C)$
DEC	1	0	0	0	Decrement A_{CC}	$(A_{CC}) \leftarrow (A_{CC}) - 1$
INC	1	0	0	1	Increment A_{CC}	$(A_{CC}) \leftarrow (A_{CC}) + 1$
CMP	1	0	1	0	Complement A_{CC} (1's complement)	$(A_{CC}) \leftarrow (\overline{A_{CC}})$
SHR1	1	0	1	1	1-bit R-shift	
SHL1	1	1	0	0	1-bit L-shift	
SHL2	1	1	0	1	2-bit L-shift	
SHL4	1	1	1	0	4-bit L-shift	
XCHG	1	1	1	1	8-bit exchange	

Symbols:
P = Input selected in the P-Select field; C = Carry flag not selected by the ASL bit.

Table 6. ASL Bit Specifications

Mnemonic	ASL Bit 15	A_{CC} Selection
ACCA	0	A_{CC} A
ACCB	1	A_{CC} B

Table 7. DP_L Field Specifications

Mnemonic	DP_L Field 14	13	Operation
DPNOP	0	0	No operation
DPINC	0	1	Increment DP_L
DPDEC	1	0	Decrement DP_L
DPCLR	1	1	Clear DP_L

Table 8. DP_H/M Field Specifications

Mnemonic	DP_H/M Field 12	11	10	9	Exclusive OR
M0	0	0	0	0	$(DP_7\ DP_6\ DP_5\ DP_4) \forall (0\ 0\ 0\ 0)$
M1	0	0	0	1	$(DP_7\ DP_6\ DP_5\ DP_4) \forall (0\ 0\ 0\ 1)$
M2	0	0	1	0	$(DP_7\ DP_6\ DP_5\ DP_4) \forall (0\ 0\ 1\ 0)$
M3	0	0	1	1	$(DP_7\ DP_6\ DP_5\ DP_4) \forall (0\ 0\ 1\ 1)$
M4	0	1	0	0	$(DP_7\ DP_6\ DP_5\ DP_4) \forall (0\ 1\ 0\ 0)$
M5	0	1	0	1	$(DP_7\ DP_6\ DP_5\ DP_4) \forall (0\ 1\ 0\ 1)$
M6	0	1	1	0	$(DP_7\ DP_6\ DP_5\ DP_4) \forall (0\ 1\ 1\ 0)$
M7	0	1	1	1	$(DP_7\ DP_6\ DP_5\ DP_4) \forall (0\ 1\ 1\ 1)$
M8	1	0	0	0	$(DP_7\ DP_6\ DP_5\ DP_4) \forall (1\ 0\ 0\ 0)$
M9	1	0	0	1	$(DP_7\ DP_6\ DP_5\ DP_4) \forall (1\ 0\ 0\ 1)$
MA	1	0	1	0	$(DP_7\ DP_6\ DP_5\ DP_4) \forall (1\ 0\ 1\ 0)$
MB	1	0	1	1	$(DP_7\ DP_6\ DP_5\ DP_4) \forall (1\ 0\ 1\ 1)$
MC	1	1	0	0	$(DP_7\ DP_6\ DP_5\ DP_4) \forall (1\ 1\ 0\ 0)$
MD	1	1	0	1	$(DP_7\ DP_6\ DP_5\ DP_4) \forall (1\ 1\ 0\ 1)$
ME	1	1	1	0	$(DP_7\ DP_6\ DP_5\ DP_4) \forall (1\ 1\ 1\ 0)$
MF	1	1	1	1	$(DP_7\ DP_6\ DP_5\ DP_4) \forall (1\ 1\ 1\ 1)$

Table 9. RPDCR Bit Specifications

Mnemonic	RPDCR Bit 8	Operation
RPNOP	0	No operation
RPDEC	1	Decrement RP

Table 10. SCR Field Specifications

	SRC Field				
Mnemonic	7	6	5	4	Source Register
NON, TRB (Note 1)	0	0	0	0	TRB
A	0	0	0	1	A_{CC} A
B	0	0	1	0	A_{CC} B
TR	0	0	1	1	TR
DP	0	1	0	0	DP
RP	0	1	0	1	RP register
RO	0	1	1	0	RO register
SGN	0	1	1	1	SGN register
DR	1	0	0	0	DR register
DRNF	1	0	0	1	DR register (Note 2)
SR	1	0	1	0	SR register
SIM	1	0	1	1	ADSI register (Note 3)
SIL	1	1	0	0	ADSI register (Note 4)
K	1	1	0	1	K register
L	1	1	1	0	L register
MEM	1	1	1	1	RAM

Notes:
(1) TRB register data is output to the internal data bus even when NON is specified.
(2) DR register data is output to the internal data bus but the ROM flag is not set.
(3) For 16-bit data, the first serial input data is output to the highest bit (MSB) and the last is output to the lowest bit (LSB).
(4) For 16-bit data, the first serial input data is output to the LSB of the internal data bus and the last is output to the MSB.

Table 11. DST Field Specifications

	DST Field				
Mnemonic	3	2	1	0	Destination Register
@ NON	0	0	0	0	No register
@ A	0	0	0	1	A_{CC} A (accumulator A)
@ B	0	0	1	0	A_{CC} B (accumulator B)
@ TR	0	0	1	1	TR (temporary register)
@ DP	0	1	0	0	DP (data pointer)
@ RP	0	1	0	1	RP register
@ DR	0	1	1	0	DR register
@ SR	0	1	1	1	SR register
@ SOL	1	0	0	0	SO register serial out LSB (Note 1)
@ SOM	1	0	0	1	SO register serial out MSB (Note 2)
@ K	1	0	1	0	K register
@ KLR	1	0	1	1	KLR (Note 3)
@ KLM	1	1	0	0	KLM (Note 4)
@ L	1	1	0	1	L register
@ TRB	1	1	1	0	TRB register
@ MEM	1	1	1	1	RAM

Notes:
(1) For 16-bit serial data, serial data is output from the LSB of the internal data bus sequentially.
(2) For 16-bit data, serial data is output from the MSB of the internal data bus sequentially.
(3) The K register stores data on the internal data bus and the L register stores the RO register (ROM) output.
(4) The L register stores data on the internal data bus and the K register stores RAM data specified by $DP_6 = 1$ (DP_7, 1, DP_5, DP_4, DP_3, DP_2, DP_1, and DP_0).

Table 12. BRCH Field Specifications

	BRCH Field*									
Mnemonic	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**Conditions**
JMP	1	0	0	0	0	0	0	0	0	Unconditional
CALL	1	0	1	0	0	0	0	0	0	Unconditional
JNCA	0	1	0	0	0	0	0	0	0	CA = 0
JCA	0	1	0	0	0	0	0	1	0	CA = 1
JNCB	0	1	0	0	0	0	1	0	0	CB = 0
JCB	0	1	0	0	0	0	1	1	0	CB = 1
JNZA	0	1	0	0	0	1	0	0	0	ZA = 0
JZA	0	1	0	0	0	1	0	1	0	ZA = 1
JNZB	0	1	0	0	0	1	1	0	0	ZB = 0
JZB	0	1	0	0	0	1	1	1	0	ZB = 1
JNOVA0	0	1	0	0	1	0	0	0	0	OVA0 = 0
JOVA0	0	1	0	0	1	0	0	1	0	OVA0 = 1
JNOVB0	0	1	0	0	1	0	1	0	0	OVB0 = 0
JOVB0	0	1	0	0	1	0	1	1	0	OVB0 = 1
JNOVA1	0	1	0	0	1	1	0	0	0	OVA1 = 0
JOVA1	0	1	0	0	1	1	0	1	0	OVA1 = 1
JNOVB1	0	1	0	0	1	1	1	0	0	OVB1 = 0
JOVB1	0	1	0	0	1	1	1	1	0	OVB1 = 1
JNSA0	0	1	0	1	0	0	0	0	0	SA0 = 0
JSA0	0	1	0	1	0	0	0	1	0	SA0 = 1
JNSB0	0	1	0	1	0	0	1	0	0	SB0 = 0
JSB0	0	1	0	1	0	0	1	1	0	SB0 = 1
JNSA1	0	1	0	1	0	1	0	0	0	SA1 = 0
JSA1	0	1	0	1	0	1	0	1	0	SA1 = 1
JNSB1	0	1	0	1	0	1	1	0	0	SB1 = 0
JSB1	0	1	0	1	0	1	1	1	0	SB1 = 1
JDPL0	0	1	0	1	1	0	0	0	0	$DP_L = 0$
JDPLN0	0	1	0	1	1	0	0	0	1	$DP_L \neq 0$
JDPLF	0	1	0	1	1	0	0	1	0	DP_L = F (HEX)
JDPLNF	0	1	0	1	1	0	0	1	1	$DP_L \neq$ F (HEX)
JNSIAK	0	1	0	1	1	0	1	0	0	SI ACK = 0
JSIAK	0	1	0	1	1	0	1	1	0	SI ACK = 1
JNSOAK	0	1	0	1	1	1	0	0	0	SO ACK = 0
JSOAK	0	1	0	1	1	1	0	1	0	SO ACK = 1
JNRQM	0	1	0	1	1	1	1	0	0	RQM = 0
JRQM	0	1	0	1	1	1	1	1	0	RQM = 1

Note:
* The BRCH field values not listed in this table are prohibited.

Table 13. NA Field Specifications

NA Field											
12	11	10	9	8	7	6	5	4	3	2	Jump Address
0	0	0	0	0	0	0	0	0	0	0	Address 0
0	0	0	0	0	0	0	0	0	0	1	Address 1
0	0	0	0	0	0	0	0	0	1	0	Address 2
					≀						≀
1	1	1	1	1	1	1	1	1	1	1	Address 2047

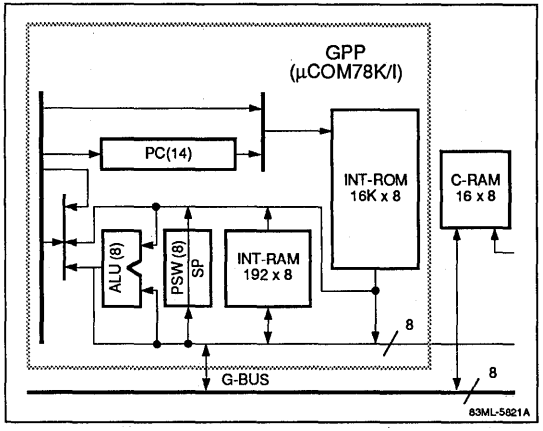
Table 14. ID Field Specifications

ID Field																
21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	HEX
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0001
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0002
							≀									≀
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	FFFF

GPP FUNCTIONAL DESCRIPTION

Figure 4 is the block diagram of the GPP.

Figure 4. GPP Block Diagram



Memory Map

The general purpose processor (GPP) has a 64 K byte address space (16-bit address). Figure 5 shows memory mapping of the GPP.

The GPP address space consists of the following:

- 16,384 byte internal program memory (INT-ROM) space.
- 192 byte internal data memory (INT-RAM) space.
- 256 byte special function register (SFR) space.

Internal Program Memory Space [INT-ROM]. A 16,384 word × 8-bit mask programmable ROM occupies an area of addresses from 0000H to 3FFFH. The ROM can be used for storing programs and data. The internal program memory space is allocated as follows:

Vector Table Area: The 22 bytes from 0000H to 0015H holds vectors for reset and interrupts. The low-order eight bits of a 14-bit address are stored in an even-numbered address and the high order six bits are stored in an odd-numbered address. See table 15 for the interrupt-vector address.

Figure 5. GPP Memory Mapping

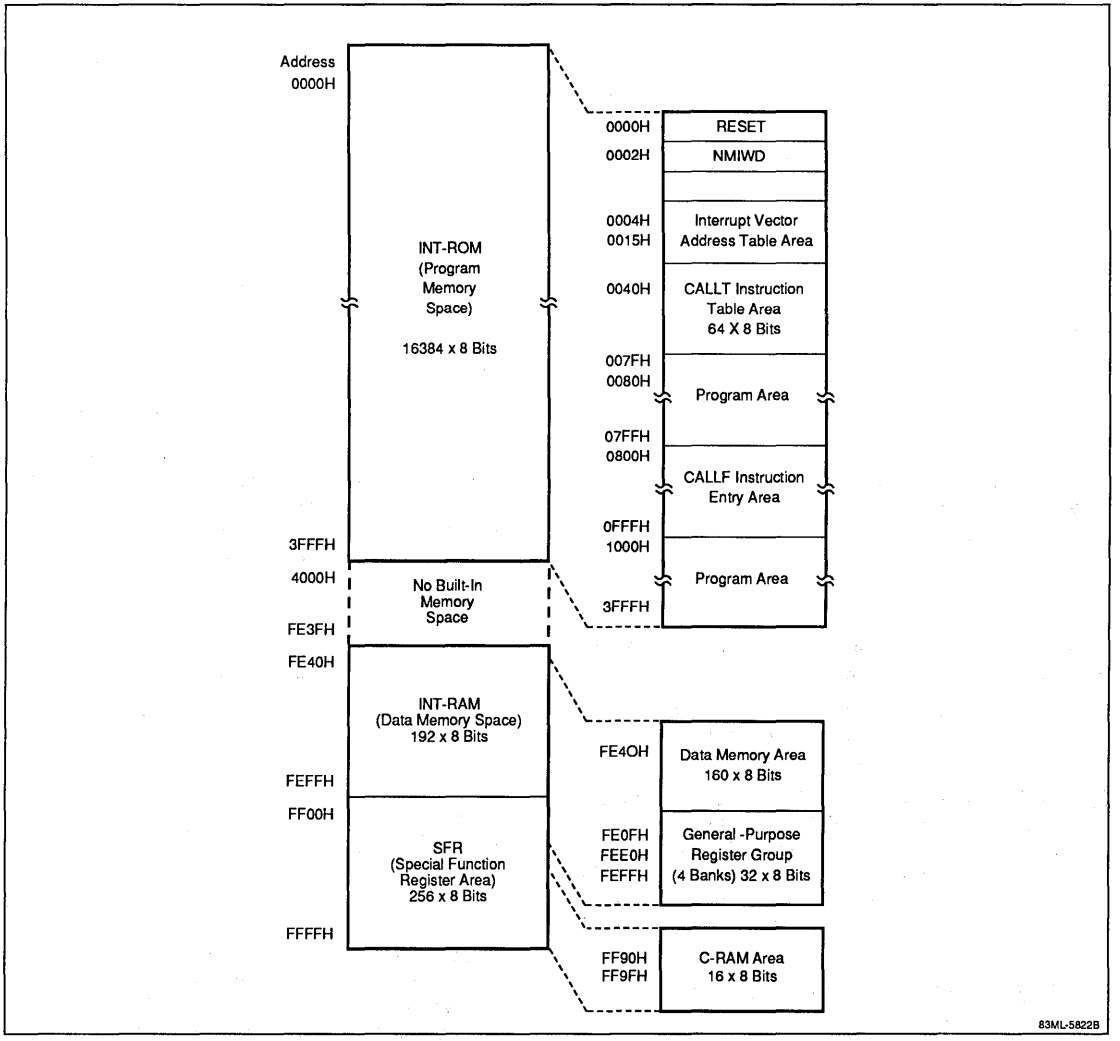


Table 15. Interrupt Vector Address

Interrupt Vector Address	Interrupt Source Flag Name	Interrupt Source Condition
0000H		Reset ($\overline{RESET}$) input
0002H	NMIWD	Watch dog timer
0004H	IST	STINT rising edge
0006H	IRT	RT rising edge
0008H	IIU	Data was input to URTI, or a break signal was detected.
000AH	IOU	Data was input to URTO
000CH	IFIFO	Data was read from FIFO, or four levels of FIFO data were output.
000EH	IAT	TMRA is 0
0010H	IBT	TMRB is 0
0012H	IS1	Data is input to SI1
0014H	INT	Interrupt (INT) input

CALLT Instruction Table Area: A 64-byte area from 0040H to 007FH stores a one-byte call instruction (CALLT) subroutine entry address.

CALLF Instruction Entry Area: An area from 0800H to 0FFFH stores a two-byte call instruction (CALLF) which calls a subroutine directly.

Internal Data Memory Space (INT-RAM). A memory area from FE40H to FEFFH is allocated to a 192-byte RAM.

In the RAM's 32-byte area from FEE0H to FEFFH a four-bank general-purpose register group is mapped. Data memory is also used as stack memory.

Special Function Register (SFR) Space. A 61-byte area within a 256-byte area from FF00H to FFFFH stores a special function register (SFR) of on-chip peripheral hardware. The addresses not mapped with SFR are not accessible. C-RAM is also mapped within the SFR space. Note that it is possible for C-RAM to be externally accessible. See I/O port and C-RAM.

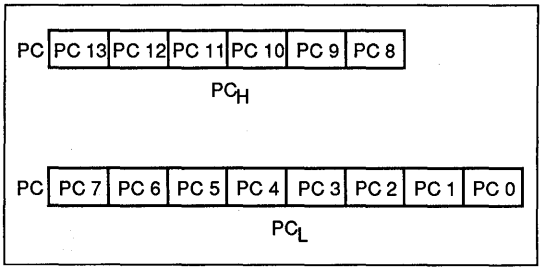
C-RAM which is able to write externally in the slave mode, is also allocated in the SFR space.

Registers

Program Counter [PC]. The program counter is a 14-bit binary counter containing address information of the next program to be executed. It is incremented automatically depending on the number of bytes of the instruction to be fetched. When a branch instruction is executed, immediate data or the contents of a register is set in the counter.

When the $\overline{RESET}$ signal is input, the PC is initialized with the data at addresses 0000H and 0001H in INT-ROM; the data at address 0000H are placed in the low-order eight bits of the PC, and the low-order six bits of the data at 0001H are placed in the high-order six bits of the PC. See Figure 6.

Figure 6. Program Counter Configuration



Program Status Word [PSW]. The program status word is an 8-bit register consisting of flags. See Figure 7. It can be read or written on an eight-bit basis. The flags area is operated by bit operation instructions. The PSW data is saved into a stack area when an interrupt request is issued or a PUSH instruction is executed and is restored with a RETI or POP instruction.

When $\overline{RESET}$ is input, all flags are cleared and PSW is set to 02H.

Figure 7. Program Status Word Configuration

	7	6	5	4	3	2	1	0
PSW	IE	Z	RBS1	AC	RBS0	0	1	CY

Note: Bit 2 and Bit 1 must be set as follows:
Bit 2 = 0
Bit 1 = 1

Carry Flag [CY]: The carry flag (CY) stores overflow or underflow when arithmetic instructions are executed. The flag stores the value shifted out when a shift rotate instruction is executed and performs as a bit accumulator when a bit operation instruction is executed.

Register Bank Select Flags [RBS_0 and RBS_1]: RBS_0 and RBS_1 are used to select one of the four register banks. See table 16.

Table 16. Register Bank Selection

RBS_1	RBS_0	Register Bank
0	0	Register bank 0
0	1	Register bank 1
1	0	Register bank 2
1	1	Register bank 3

Figure 8. Stack Pointer Configuration

	7	6	5	4	3	2	1	0
SP	SP 7	SP 6	SP 5	SP 4	SP 3	SP 2	SP 1	SP 0

Auxiliary Carry Flag [AC]: The auxiliary carry flag is set to 1 when a bit 3 carry occurs at the end of an operation or when a bit 3 borrow occurs. Otherwise it is reset to 0. The AC flag is used when a BCD correct instruction is executed.

Zero Flag [Z]: The zero flag is set to 1 when the result of an operation is 0. If the result of an operation is not 0, the Z flag is reset to 0. The Z flag can be tested with a conditional branch instruction.

Interrupt Request Enable Flag [IE]: The interrupt request enable flag controls whether a CPU interrupt request (maskable vector interrupt) is accepted. When the flag is set to 0, the processor is set to the DI state and all interrupts except a non-maskable interrupt (watch dog timer interrupt) are disabled. When the flag is set to 1, the processor is set to the EI state and interrupt requests are controlled by the interrupt mask flag for each interrupt request. The EI flag is set to 1 when an EI instruction is executed and reset to 0 when a DI instruction is executed or an interrupt is accepted.

Stack Pointer [SP]. The stack pointer is an 8-bit register used to retain the low-order eight bits of the return address in a stack area (LIFO form). The high-order eight bits of an address in this area are always FEH. The stack memory is allocated to any area in data memory (FE40H to FEFFH). When the SP value is set SP data is not stored from 00H to 3FH. SP data is decremented when a write (save) operation is performed to stack memory and incremented when data is read (restored) from stack memory. SP is accessible with a dedicated instruction. SP data is not acted upon when $\overline{RESET}$ is input. $\overline{RESET}$ must initialize the SP before a subroutine call. See Figures 8, 9, and 10.

Figure 9. Data Saved to the Stack Memory

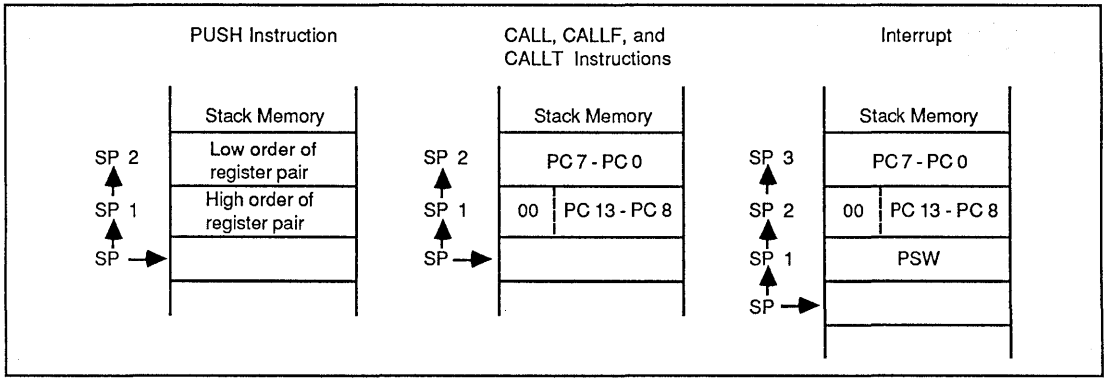
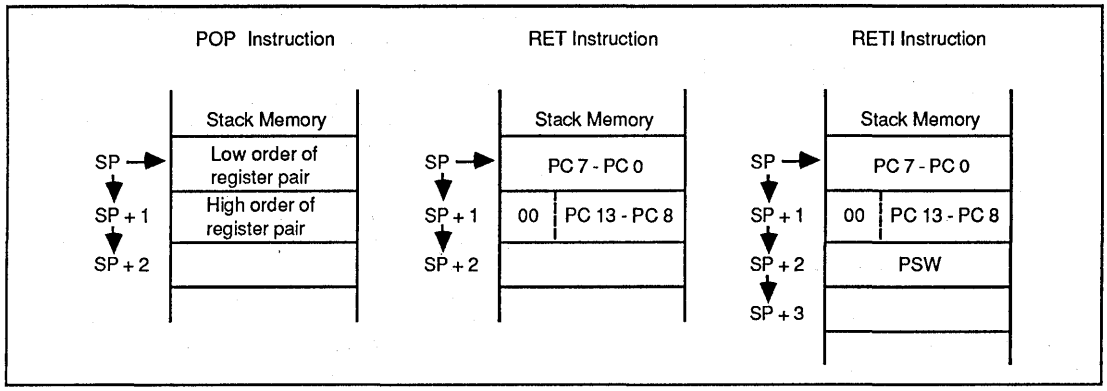


Figure 10. Data Restored From the Stack Memory



General-Purpose Registers. General-purpose registers are mapped to special addresses in the INT-RAM (FEE0H to FEFFH). The registers consist of four bank registers; each having eight 8-bit registers (X, A, C, B, E, D, L, and H). The actual register bank in operation is determined by RBS0 and RBS1 of PSW.

Normally, general-purpose registers are operated on an eight-bit basis. These can also be operated on a 16-bit basis as a pair of 8-bit registers (AX, BC, DE, and HL).See Figure 11.

Registers have functional names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL) as well as absolute names (R0 to R7 and RP0 to RP3). See table 17 for the relationship between functional names and absolute names.

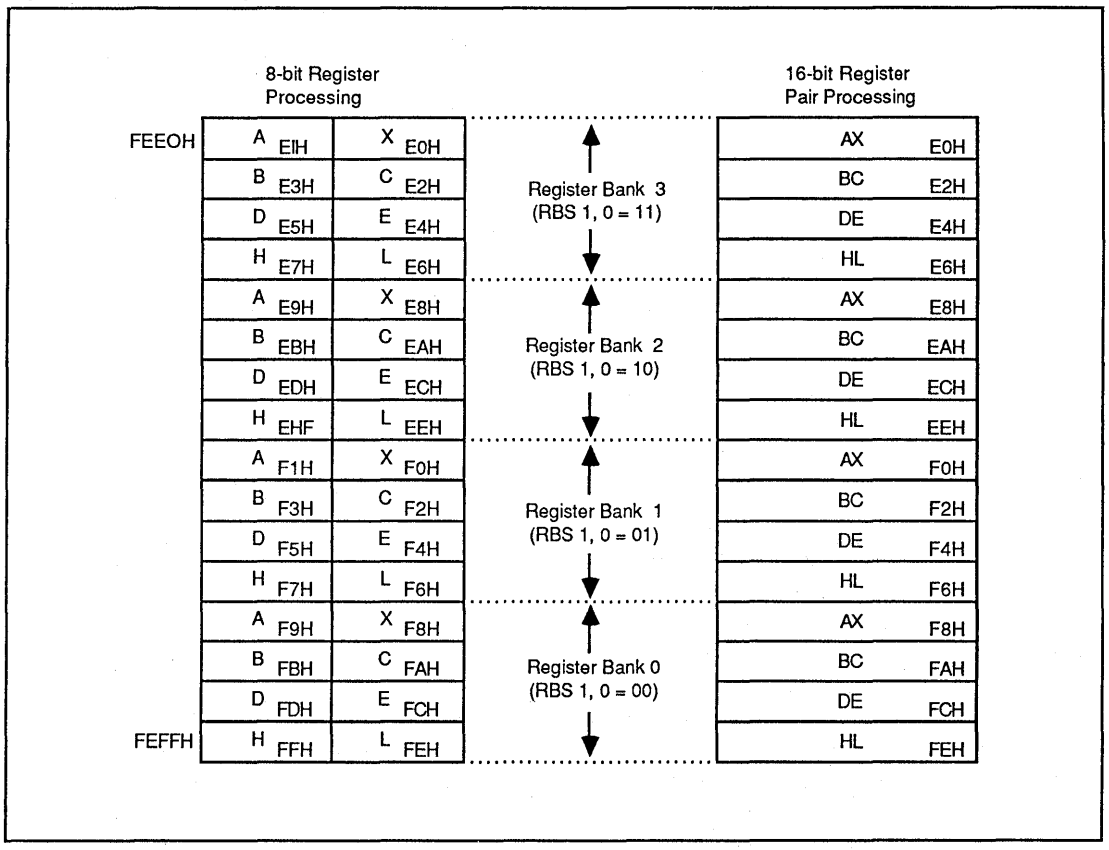
The general-purpose register area is accessible by specifying a normal data memory address. It does not have to be used as a register area.

The GPP has four register banks and the user can use different register banks for efficient programming of normal and interrupt operations.

Table 17. Relationship Between Functional Names and Absolute Names

Functional Name	Absolute Name
X	R0
A	R1
C	R2
B	R3
E	R4
D	R5
L	R6
H	R7
AX	RP0
BC	RP1
DE	RP2
HL	RP3

Figure 11. General-Purpose Register Configuration



Special Function Register [SFR]

The special function registers are assigned to special functions like the built-in peripheral hardware mode register and control registers. They are mapped to 61 bytes in the 256-byte area from FF00H to FFFFH.

SFRs are instruction operands which can be used for transfer instructions, bit operation instructions, and arithmetic instructions.

Note that only addresses assigned for the SFR are accessible. If an address not assigned for the SFR is accessed, the processor may malfunction.

Table 18 lists the SFRs.

Table 18. Special Function Register (SFR) List

Functional Area	SFR Name	Mnemonic	R/W	Status at Reset	Address
Port	Port mode register (PTMR)	PTMR	R/W	3FH	FF28H (8-bits)
	Port C mode register (PCMR)	PCMR		7FH	FF29H (8-bits)
	Port D mode register (PDMR)	PDMR		FFH	FF2AH (8-bits)
	Port A (PORTA) (Note 1)	PA		00H	FF2CH (8-bits)
	Port B (PORTB) (Note 1)	PB		00H	FF2DH (8-bits)
	Port C (PORTC) (Note 1)	PC		00H	FF2EH (low-order 7-bits)
	Port D (PORTD) (Note 1)	PD		00H	FF2FH (8-bits)
	Port E (PORTE)	PE	R	0H	FF57H (low-order 1-bit)
	Port F (PORTF)	PF	R/W	0H	FF5CH (low-order 3-bits)
Interrupt	Interrupt request flag register (IF0)	IF0	R/W	00H 00H	FFE0H (16-bits) FFE1H
	Interrupt mask register (MK0)	MK0		FFH FFH	FFE4H (16-bits) FFE5H
	DSP interrupt register (INTDSP) (Note 8)	INTDSP		0H	FF64H (low-order 2-bits)
Scrambler/descrambler	Mode register (SCRMR)	SCRMR	R/W	00H	FF40H (8-bits)
	Scrambler port (SCR) (Note 3)	SCR		Undefined	FF41H (low-order 1-bit)
	Descrambler port (DSC) (Note 3)	DSC			FF42H (low-order 1-bit)
	Scrambler control register (SCRM)	SCRM		0H	FF65H (low-order 4-bits)
	Descrambler control register (DSCM)	DSCM		0H	FF66H (low-order 3-bits)
Transmit PLL/receive PLL	PPL mode register 1 (PLLMR1)	PLLMR1	R/W	00H	FF44H (8-bits)
	PPL mode register 2 (PLLMR2)	PLLMR2		33H	FF7EH (8-bits)
	SBAUD, RBAUD status register (BAUDSR)	BAUDSR	R	0H	FF45H (low-order 2-bits)
Serial communication interface ASC, SAC, UART	Synchronous/asynchronous mode register (ASMR)	ASMR	R/W	00H	FF49H (8-bits)
	UART mode register (URTMR)	URTMR		00H	FF4AH (low-order 7-bits)
	UART status register (URTSR) (Note 4)	URTSR	R	0H	FF4BH (low-order 4-bits)
	ASC register (ASCR)	ASCR		Undefined	FF4CH (8-bits)
	SAC register (SACR)	SACR	R/W		FF4DH (8-bits)
	URO register (URO)	URO			FF3EH (8-bits)
	URI register (URI)	URI	R		FF3FH (8-bits)
A/D, D/A interface	D/A mode register (DAMR)	DAMR	R/W	00H	FF4EH (low-order 6-bits)
	FIFO read address (FFRA)	FFRW		0H	FF4FH (high-order 3-bits)
	FIFO write address (FFWA)	FFRW		0H	FF4FH (low-order 3-bits)
	FIFO (FIFO) (Note 5)	FIFO		Undefined	FF54H (16-bits) FF55H
Serial I/O	Status register (S1SR)	S1SR	R	0H	FF56H (2-bits)
	Serial input port 1 (SI1)	SI1	R/W	0000H	FF58H (16-bits) FF59H
	Serial output port 1 (SO1)	SO1		0000H	FF5AH (16-bits) FF5BH
Timer	Timer mode register (TMMR) (Note 6)	TMMR	R/W	00H	FF5DH (8-bits)
	Timer A (TMRA)	TMRA		FFH	FF5EH (8-bits)
	Watch dog timer control register (WDMSR) (Note 7)	WDMSR		00H	FF6DH (8-bits)

Table 18. Special Function Register (SFR) List (cont)

Functional Area	SFR Name	Mnemonic	R/W	Status at Reset	Address
DSP interface	Data register (DR) (Note 2)	DR	R/W	Undefined	FF60H (16-bits) FF61H
	Status register (SR)	SR	R	00H	FF62H (8-bits) FF63H
C-RAM	Control RAM (C-RAM)		R/W	Undefined	FF90H (8-bits) FF9FH (8-bits)

Notes:
(1) Write operation is invalid when the register is used as an input port.
(2) The DSP status (RQM flag) is changed by a Read/Write signal.
(3) The shift register of the scrambler/descrambler is shifted one bit by a Write signal to the SCR and DSC.
(4) URTSR is reset after it is read.
(5) The FFWA write address is incremented by a Write signal to the FIFO.
(6) This register is reset to 0 by TMRA.
(7) The write operation is performed with special instructions (MOV WDMSR, #byte).
(8) INTDSP is reset six clocks after it is set to 1.
(9) The 16-bit SFR registers must be accessed one byte at a time. For high byte access the symbol H is appended to the SFR mnemonic and for the low byte access the symbol L is used.

Interrupt Functions

The GPP has one non-maskable interrupt and nine maskable vector interrupts.

The vector interrupt saves status information (PC and PSW information) of the program being executed. The status information is stored in memory specified by the stack pointer when an interrupt request is accepted. Then data is stored at the address of the interrupt request (vector table address) in the PC as vector address information and starts the interrupt service program. Control is returned from the interrupt service program by transferring the program counter value and status information from stack memory to the PC and PSW with the RETI instruction.

Maskable Vector Interrupt. Maskable vector interrupt processing indicates when an interrupt is enabled by the interrupt mask register [MK0]. The interrupt source state can be checked by the interrupt request flag register [IF0]. Maskable vector interrupt operations are explained below. The interrupt enable state indicates that the IE bit of PSW is 1 and the corresponding bit of the interrupt mask register MK0 is 0.

- When an interrupt source is detected, the corresponding bit of IF0 is set.
- When interrupt processing starts, the corresponding bit of IF0 is reset.
- When an interrupt source is detected while interrupt is enabled, interrupt processing starts.
- If two or more interrupt sources are detected, priority is given to the lowest interrupt vector address.
- If an interrupt request is detected during interrupt processing, it is nested when interrupt is enabled.

The GPP has a total of ten interrupt request sources; nine maskable interrupts and one non-maskable interrupt. Of the ten sources the maskable interrupt request sources are listed in table 19.

Table 20 lists the interrupt vector table addresses. Table 21 lists the IF0 and MK0 SFR addresses.

Table 19. Maskable Interrupt Request Sources

Interrupt Source	Interrupt Signal	Condition
T_XPLL	IST	STINT rising edge
P_XPLL	IRT	RT rising edge
TIMRA	IAT	Timer TMRA is set to 0
TIMRB	IBT	Timer TMRB is set to 0
FIFO	IFIFO	Data was read from FIFO. Or, four levels of FIFO data were output from FIFO.
SI1	IS1	Data was input to SI1
UART	IIU	Data was input to URTI. Or, a break signal was detected.
	IOU	URTO data was output
External	INT	External interrupt

Table 20. Interrupt Vector Table Address

Interrupt Request Type	Vector Table Address	Default Priorities	Interrupt Request Signal	IF0 Corresponding Bit	MK0 Corresponding Bit
Maskable	0004H	1	IST	0	0
	0006H	2	IRT	1	1
	0008H	3	IIU	2	2
	000AH	4	IOU	3	3
	000CH	5	IFIFO	4	4
	000EH	6	IAT	5	5
	0010H	7	IBT	6	6
	0012H	8	IS1	7	7
	0014H	9	INT	8	8
Non-maskable	0002H	0	Watch dog timer interrupt	–	–

Table 21. IF0 and MK0, SFR Addresses

Mnemonic	SFR Address	Function
IF0	FFE0, FFE1H	Interrupt request flag register (16-bits)
MK0	FFE4, FFE5H	Interrupt mask register (16-bits)

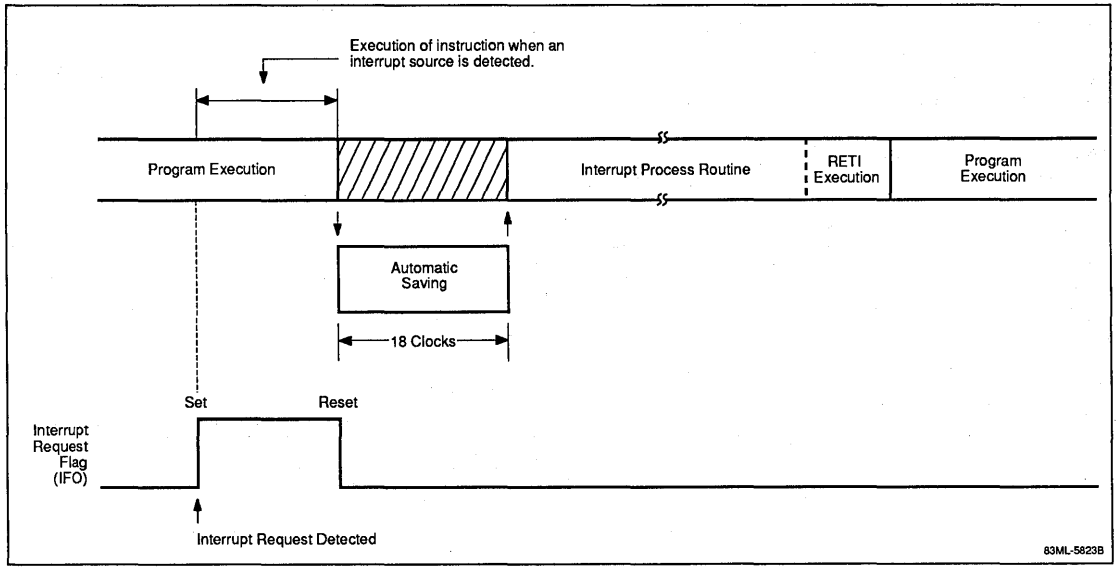
Interrupt Request Flag Register [F0]: The interrupt request flag register is a 16-bit register. It consists of the interrupt source flags listed in table 20. The flags in the interrupt request flag register are set when a corresponding interrupt source is detected and reset when it is processed. Flags are reset to 0 when $\overline{RST}$ is input. The low-order seven bits are always 0.

Interrupt Mask Register [MK0]: The interrupt mask register is a 16-bit register. It sets even if interrupt is enabled when an interrupt source flag is set. See table 20 for interrupt source flags. The flags of the MK0 are set to 0 to enable interrupt and set to 1 to disable interrupt.

The low-order seven bits are always 1. Flags are initialized to 1 when $\overline{RST}$ is input.

Vector Interrupt Processing: The vector interrupt processing sequence is shown in Figure 12. It is automatically executed internally. The latency in the interrupt process routine gaining control is 18 clocks (approximately 3.3 μs).

Figure 12. Vector Interrupt Operation



Non-Maskable Interrupt. The processor has a watch dog timer interrupt function as a non-maskable interrupt. This interrupt is executed immediately when a source is detected. Interrupt execution does not affect the IE flag of PSW.

Interrupt to the DSP. The GPP has reset and interrupt functions to the DSP. These functions are specified by the 2-bit INTDSP register. INTDSP is initialized to 0 when Reset is input. See table 22 and table 23.

Table 22. INTDSP Function

INTDSP	Function
INTDSP (bit 1)	When INTDSP is set to 1 an interrupt request is issued to the DSP. After being issued INTDSP resets automatically.
INTDSO (bit 0)	When INTDSO is a 1, DSP is reset

Table 23. INTDSP SFR Address

Mnemonic	SFR Address	Function
INTDSP	FF64H	DSP reset/interrupt request register

Addressing

GPP addressing includes the following:

- Data memory addressing
- Instruction addressing

Data Memory Addressing. Figure 13 shows the data memory map, SFR memory map, and applicable addressing.

Register Addressing: Addresses a general-purpose register mapped at a specific address in data memory. The general-purpose register in the register bank specified by RBS0 and RBS1 flags in the PSW is registered.

Coding example follows:

```
XCH  A, r
```

To specify the C register as r, code as follows:

```
XCH  A, C
```

Short and Direct Addressing: Addresses an area from FE40H to FEFFH in the internal data memory and an area from FF00H to FF1FH in the SFR. To access 16-bit data, 2-byte data specified by continuous even-numbered and odd-numbered addresses is specified.

Coding example follows:

```
ADDC  saddr, A
```

To specify address FE50H as saddr, code as follows:

```
ADDC  0FE50H, A
```

SFR Addressing: Addresses a special function register (SFR) mapped to the SFR area (FF00H to FFFH).

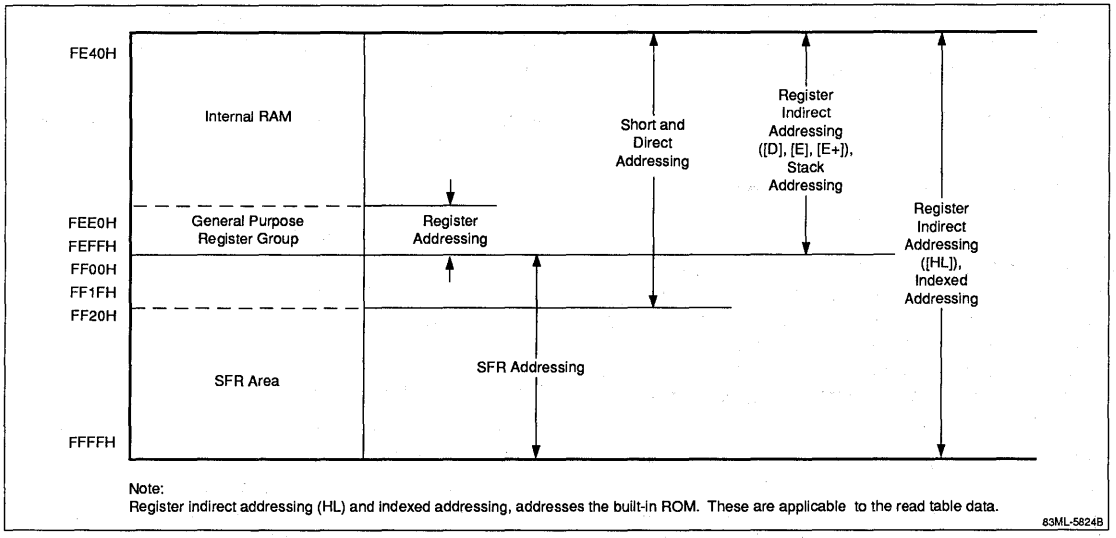
Coding example follows:

```
MOV  A, sfr
```

To specify the PTMR register as sfr, code as follows:

```
MOV  A, PTMR
```

Figure 13. Data Memory Map and Addressing



Register Indirect Addressing: Addresses data memory indirectly by the contents of the register stored in the operand. The register in the register bank specified by the RBS0 and RBS1 flags in the PSW is specified. Only when the E register is specified with the MOV instruction are the contents of the register automatically incremented by one after the instruction is executed. In this case, the operand is coded as [E+]. Register indirect addressing using the HL register pair can address the overall space including the internal ROM.

Coding example follows:

SUB A, [r4]

To specify the E register a r4, code as follows:

SUB A, [E]

Indexed Addressing: Addresses data as the result of an addition of 16-bit immediate data and 8-bit register data. The 8-bit register is in the register bank specified by the RBS0 and RBS1 flags of the PSW. This technique can address the overall space including the internal ROM.

Coding example follows:

MOV A, word [r1]

To specify FEA0H as word and the B register as r1, code as follows:

MOV A, OFEA0H [B]

Stack Indirect Addressing: Addresses internal memory data (FE40H to FEFFH) indirectly by the contents of the stack pointer (SP).

This technique is applicable when executing PUSH and POP instructions, save or restore operations by interrupt processing, and subroutine call and return.

Coding example follows:

PUSH rp

To specify the DE register pair as rp, code as follows:

PUSH DE

Instruction Addressing. The instruction address is determined by the program counter (PC) value. Normally, the PC is automatically incremented by one (for one byte) depending on the number of bytes to be fetched every time an instruction is executed. If a branch instruction is executed, branch destination information is set in the PC by distinct addressing, as shown below:

Relative Addressing: The first address of a subsequent instruction is added by 8-bit immediate data (displacement value: jdisp) of an instruction code and transferred to the PC. Then program control branches to the address set in the PC. The displacement value is handled as signed two's complements (−128 to +128) and bit 7 is used as a sign bit.

Relative addressing is applicable for the BR S addr 14 instruction and a branch instruction.

Immediate Addressing: Immediate data in an instruction word is transferred to the PC and program control branches to the address set in the PC.

Immediate addressing is applicable for the CALL laddr14, BR laddr14, and CALLF laddr11 instructions. For the CALLF laddr11 instruction, program control branches to the fixed area of the low-order 2-bit address.

Table Indirect Addressing: The contents of a specific location table (branch destination address) addressed by immediate data of the low-order five bits of an instruction code are transferred to the PC and program control branches to the address set in PC.

Table indirect addressing is applicable for the CALLT [addr5] instruction.

Register Addressing: The contents of a register pair (RP3 to RP0) specified by an instruction word is transferred to the PC and program control branches to the address set in PC. Register addressing is applicable for the BR rp instruction.

INSTRUCTION SET

Tables 24 through 27 and figure 14 define the operands, symbols, and codes that appear in table 28. Table 28 lists the instruction encodings and shows all the legitimate combinations of operands. The instruction set terminology is as follows:

Operands and Coding Requirements: In the operand field of an instruction, operands are accepted according to their value. An operand having two or more values can have only one selected. Uppercase letters and symbols like +, #, !, $, /, and [ ] are keywords and must be written as they are presented. The symbols have the following meanings:

+	=	Automatic increment
#	=	Immediate data
!	=	Absolute address
$	=	Relative address
/	=	Bit reverse
[ ]	=	Indirect addressing

For immediate data, write an appropriate numeric value or label. When a label is used, it must be defined elsewhere.

The clock column symbols are as follows:

- n in the clock column of a shift rotate instruction indicates the number of bits to be shifted.
- The value enclosed in ( ) in the clock column of a conditional branch instruction indicates the number of clocks when program control does not branch.

- When accessing SFR by register indirect addressing ([HL]) and indexed addressing (word [r1]), the number of clocks is set to the one shown after a slash (/) in the column.
- If the result of word +r1 overflows in indexed addressing, the number of clocks is increased to the value enclosed in ( ).

Table 24. Operand Values

Operand	Value
r	X (R0), A (R1), C (R2), B (R3), E (R4), D (R5), L (R6), H (R7)
r1	A, B
r2	B, C
r3	D, E, E+
r4	D, E
rp	AX (RP0), BC (RP1), DE (RP2), HL (RP3)
sfr	Special function register abbreviation (see table 16)
sfrp	Special function register abbreviation (16-bit operable register, see table 16)
saddr	FE40H to FE1FH immediate data or label
saddrp	FE40H to FE1FH immediate data (bit 0 = 0) or label (for 16-bit data)
!addr14	0000H to 3FFFH immediate data or label: immediate addressing
$addr13	0000H to 1FFFH immediate data or label: relative addressing
addr11	800H to FFFH immediate data or label
addr5	40H to 7EH immediate data (bit 0 = 0) or label
word	16-bit immediate data or label
byte	8-bit immediate data or label
bit	3-bit immediate data or label
n	3-bit immediate data (0 to 7)
RBn	RB0 to RB3

Note:
r and rp can be coded with a functional name (X, A, C, B, E, D, L, H, AX, BC, DE, and HL) as well as an absolute name (R0 to R7 and RP0 to RP3).

Table 25. Abbreviations

Identifier	Description
A	A register (8-bit accumulator)
X	X register
B	B register
C	C register
D	D register
E	E register
H	H register
L	L register
R0 to R7	Register 0 to register 7 (absolute names)
AX	Register pair AX (16-bit accumulator)
BC	Register pair BC
DE	Register pair DE
HL	Register pair HL

Table 25. Abbreviations (cont)

Identifier	Description
RP0 to RP3	Register pair 0 to register pair 3 (absolute names)
PC	Program counter
SP	Stack pointer
PSW	Program status word
CY	Carry flag
AC	Auxiliary carry flag
Z	Zero flag
RBS0 to RBS1	Register bank select flag
IE	Interrupt request enable flag
WDMSR	Watch dog timer control register
( )	Memory data indicated by the address in ( ) or register data
xxH	Hexadecimal number
X_H, X_L	High-order and low-order 8-bits of 16-bit register pair

Table 26. Flag Symbols

Symbol	Description
(Blank)	Flag not affected
0	Data was cleared to 0
1	Data was set to 1
x	Data was set or cleared according to the result of operation
R	The previous saved value was restored

Table 27. Instruction Code Field Identifiers

Identifier	Description
Bn	Immediate data corresponding to bits
Nn	Immediate data corresponding to n
Data	8-bit immediate data corresponding to bytes
Low/high/byte	16-bit immediate data corresponding to words
Saddr-offset	Low-order 8-bit offset data of 16-bit address corresponding to saddr
Sfr-offset	Low-order 8-bit offset data of 16-bit address of special function register (sfr)
Low/high offset	16-bit offset data corresponding to words in indexed addressing
Low/high addr	16-bit immediate data corresponding to addr 14
jdisp	Signed two's complements of the difference between the first address of the following instruction and the branch destination address (8-bits)
fa	Low-order 11-bits of immediate data corresponding to addr 11
ta	Low-order 5-bits of immediate data corresponding to (addr5 × 1/2)

Figure 14. Operand Register Selection Codes

r

R_2	R_1	R_0	Register	
R_6	R_5	R_4		
0	0	0	R0	X
0	0	1	R1	A
0	1	0	R2	C
0	1	1	R3	B
1	0	0	R4	E
1	0	1	R5	D
1	1	0	R6	L
1	1	1	R7	H

r1

R_5	Register
0	A
1	B

r2

R_0	Register
0	C
1	B

r3

R_1	R_0	Register
0	0	E
0	1	E+
1	0	D

r4

R_1	
R_2	Register
R_4	
0	E
1	D

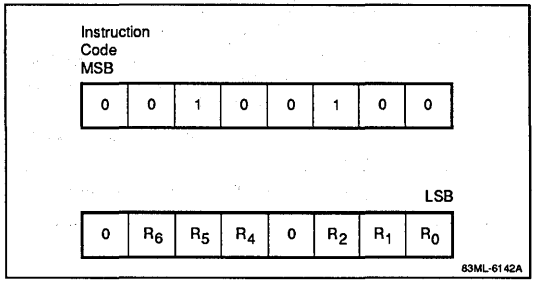
rp

P_1	P_0	Register-Pair	
P_2	P_1		
P_6	P_5		
0	0	RP0	AX
0	1	RP1	BC
1	0	RP2	DE
1	1	RP3	HL

Example of Machine Code and Operands: When both the first and second operands are arranged as registers or register pairs in the operand field, the instruction code is structured as follows:

Of a register byte, the high-order four bits are used to specify the second operand and the low-order four bits are used to specify the first operand.

MOV r, r



To specify the first operand as A register and the second operand as L register, code as follows:

MOV A, L

In this case, the instruction code is set as shown below.

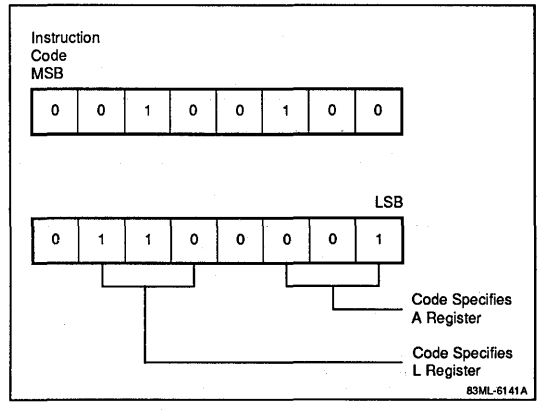


Table 28. Instruction Encodings

Mnemonic	Operand	Instruction Code B1/B3	Instruction Code B2/B4	Bytes	Clocks	Operation	Flags Z AC CY
8-Bit Data Transfer Instructions							
MOV	r, #byte	1011 $1R_2R_1R_0$	Data	2	2	r ← byte	
	saddr, #byte	0011 1010 Data	Saddr-offset	3	3	(saddr) ← byte	
	sfr, #byte (Note 1)	0010 1011 Data	Sfr-offset	3	5	sfr ← byte	
	r, r	0010 0100	$0R_6R_5R_4$ $0R_2R_1R_0$	2	2	r ← r	
	A, r	1101 $0R_2R_1R_0$		1	2	A ← r	
	A, saddr	0010 0000	Saddr-offset	2	2	A ← (saddr)	
	saddr, A	0010 0010	Saddr-offset	2	3	(saddr) ← A	
	A, sfr	0001 0000	Sfr-offset	2	4	A ← sfr	
	sfr, A	0001 0010	Sfr-offset	2	5	sfr ← A	
	A, [r3] (Note 2)	0111 $11R_1R_0$		1	5/6	A ← (FE00H + r3) r3 = 40H-FFH	
	[r3], A (Note 2)	0111 $10R_1R_0$		1	5/6	(FE00H + r3) ← A r3 = 40H-FFH	
	A, [HL]	0101 1101		1	5/7	A ← (HL)	
	[HL], A	0101 0101		1	5/7	(HL) ← A	
	A, word [r1]	0000 1010 Low offset	$00R_51$ 0000 High offset	4	7(8)/ 9(10)	A ← (word + r1)	
	word [r1], A	0000 1010 Low offset	$10R_51$ 0000 High offset	4	7(8)/ 9(10)	(word + r1) ← A	
	PSW, #byte	0010 1011 Data	1111 1110	3	5	PSW ← byte	X X X
	PSW, A	0001 0010	1111 1110	2	5	PSW ← A	X X X
	A, PSW	0001 0000	1111 1110	2	4	A ← PSW	
XCH	A, r	1101 $1R_2R_1R_0$		1	4	A ⟷ r	
	A, saddr	0010 0001	Saddr-offset	2	4	A ⟷ (saddr)	
	A, sfr	0000 0001 Sfr-offset	0010 0001	3	10	A ⟷ sfr	
	A, [r4]	0111 $1R_211$		1	8	A ⟷ (FE00H + r4) r4 = 40H-FFH	

Notes:

(1) When sfr is coded as WDMSR, MOV is used as another dedicated instruction. In this case, the numbers of bytes and clocks are different from MOV (see CPU control instruction).

(2) When r3 is coded as E+, the E register is automatically incremented by one after the instruction is executed and the number of clocks is set to 6.

Table 28. Instruction Encodings (cont)

Mnemonic	Operand	Instruction Code B1/B3	Instruction Code B2/B4	Bytes	Clocks	Operation	Flags Z	AC	CY
16-Bit Data Transfer Instructions									
MOVW	rp, #word	0110 $0P_2P_1$0 High byte	Low byte	3	3	rp ← word			
	saddrp, #word	0000 1100 Low byte	Saddr-offset High byte	4	4	(saddrp + 1) (saddrp) ← word			
	sfrp, #word	0000 1011 Low byte	Sfr-offset High byte	4	8	sfrp ← word			
	rp, rp	0010 0100	$0P_6P_5$0 $1P_2P_1$0	2	4	rp ← rp			
	AX, saddrp	0001 1100	Saddr-offset	2	6	AX ← (saddrp + 1) (saddrp)			
	saddrp, AX	0001 1010	Saddr-offset	2	5	(saddrp + 1) (saddrp) ← AX			
	AX, sfrp	0001 0001	Sfr-offset	2	10	AX ← sfrp			
	sfrp, AX	0001 0011	Sfr-offset	2	9	sfrp ← AX			
8-Bit Operation Instructions									
ADD	A, #byte	1010 1000	Data	2	2	A, CY ← A + byte	X	X	X
	saddr, #byte	0110 1000 Data	Saddr-offset	3	3	(saddr), CY ← (saddr) + byte	X	X	X
	sfr, #byte	0000 0001 Sfr-offset	0110 1000 Data	4	9	sfr, CY ← sfr + byte	X	X	X
	r, r	1000 1000	$0R_6R_5R_4$ $0R_2R_1R_0$	2	3	r, CY ← r + r	X	X	X
	A, saddr	1001 1000	Saddr-offset	2	3	A, CY ← A + (saddr)	X	X	X
	A, sfr	0000 0001 Sfr-offset	1001 1000	3	7	A, CY ← A + sfr	X	X	X
	A, [r4]	0001 0110	$011R_4$ 1000	2	7	A, CY ← A + (FE00H + r4) r4 = 40H-FFH	X	X	X
	A, [HL]	0001 0110	0101 1000	2	8/10	A, CY ← A + (HL)	X	X	X
ADDC	A, #byte	1010 1001	Data	2	2	A, CY ← A + byte + CY	X	X	X
	saddr, #byte	0110 1001 Data	Saddr-offset	3	3	(saddr), CY ← (saddr) + byte + CY	X	X	X
	sfr, #byte	0000 0001 Sfr-offset	0110 1001 Data	4	9	sfr, CY ← sfr + byte + CY	X	X	X
	r, r	1000 1001	$0R_6R_5R_4$ $0R_2R_1R_0$	2	3	r, CY ← r + r + CY	X	X	X
	A, saddr	1001 1001	Saddr-offset	2	3	A, CY ← A + (saddr) + CY	X	X	X
	A, sfr	0000 0001 Sfr-offset	1001 1001	3	7	A, CY ← A + sfr + CY	X	X	X
	A, [r4]	0001 0110	$011R_4$ 1001	2	7	A, CY ← A + (FE00H + r4) + CY r4 = 40H-FFH	X	X	X
	A, [HL]	0001 0110	0101 1001	2	8/10	A, CY ← A + (HL) + CY	X	X	X

Table 28. Instruction Encodings (cont)

Mnemonic	Operand	Instruction Code B1/B3	Instruction Code B2/B4	Bytes	Clocks	Operation	Flags Z	Flags AC	Flags CY
8-Bit Operation Instructions (cont)									
SUB	A, #byte	1010 1010	Data	2	2	A, CY ← A − byte	X	X	X
	saddr, #byte	0110 1010	Saddr-offset	3	3	(saddr), CY ← (saddr) − byte	X	X	X
		Data							
	sfr, #byte	0000 0001	0110 1010	4	9	sfr, CY ← sfr − byte	X	X	X
		Sfr-offset	Data						
	r, r	1000 1010	$0R_6R_5R_4$ $0R_2R_1R_0$	2	3	r, CY ← r − r	X	X	X
	A, saddr	1001 1010	Saddr-offset	2	3	A, CY ← A − (saddr)	X	X	X
	A, sfr	0000 0001	1001 1010	3	7	A, CY ← A − sfr	X	X	X
		Sfr-offset							
	A, [r4]	0001 0110	$011R_4$ 1010	2	7	A, CY ← A − (FE00H + r4) r4 = 40H-FFH	X	X	X
	A, [HL]	0001 0110	0101 1010	2	8/10	A, CY ← A − (HL)	X	X	X
SUBC	A, #byte	1010 1011	Data	2	2	A, CY ← A − byte − CY	X	X	X
	saddr, #byte	0110 1011	Saddr-offset	3	3	(saddr), CY ← (saddr) − byte − CY	X	X	X
		Data							
	sfr, #byte	0000 0001	0110 1011	4	9	sfr, CY ← sfr − byte − CY	X	X	X
		Sfr-offset	Data						
	r, r	1000 1011	$0R_6R_5R_4$ $0R_2R_1R_0$	2	3	r, CY ← r − r − CY	X	X	X
	A, saddr	1001 1011	Saddr-offset	2	3	A, CY ← A − (saddr) − CY	X	X	X
	A, sfr	0000 0001	1001 1011	3	7	A, CY ← A − sfr − CY	X	X	X
		Sfr-offset							
	A, [r4]	0001 0110	$011R_4$ 1011	2	7	A, CY ← A − (FE00H + r4) − CY r4 = 40H-FFH	X	X	X
	A, [HL]	0001 0110	0101 1011	2	8/10	A, CY ← A − (HL) − CY	X	X	X
AND	A, #byte	1010 1100	Data	2	2	A ← A ∧ byte	X		
	saddr, #byte	0110 1100	Saddr-offset	3	3	(saddr) ← (saddr) ∧ byte	X		
		Data							
	sfr, #byte	0000 0001	0110 1110	4	9	sfr ← sfr ∧ byte	X		
		Sfr-offset	Data						
	r, r	1000 1100	$0R_6R_5R_4$ $0R_2R_1R_0$	2	3	r ← r ∧ r	X		
	A, saddr	1001 1100	Saddr-offset	2	3	A ← A ∧ (saddr)	X		
	A, sfr	0000 0001	1001 1100	3	7	A ← A ∧ sfr	X		
		Sfr-offset							
	A, [r4]	0001 0110	$011R_4$ 1100	2	7	A ← A ∧ (FE00H + r4) r4 = 40H-FFH	X		
	A, [HL]	0001 0110	0101 1100	2	8/10	A ← A ∧ (HL)			

Table 28. Instruction Encodings (cont)

Mnemonic	Operand	Instruction Code B1/B3	Instruction Code B2/B4	Bytes	Clocks	Operation	Flags Z	AC	CY
8-Bit Operation Instructions (cont)									
OR	A, #byte	1010 1110	Data	2	2	A ← A V byte	X		
	saddr, #byte	0110 1110 Data	Saddr-offset	3	3	(saddr) ← (saddr) V byte	X		
	sfr, #byte	0000 0001 Sfr-offset	0110 1110 Data	4	9	sfr ← sfr V byte	X		
	r, r	1000 1110	$0R_6R_5R_4$ $0R_2R_1R_0$	2	3	r ← r V r	X		
	A, saddr	1001 1110	Saddr-offset	2	3	A ← A V (saddr)	X		
	A, sfr	0000 0001 Sfr-offset	1001 1110	3	7	A ← A V sfr	X		
	A, [r4]	0001 0110	$011R_4$ 1110	2	7	A ← A V (FE00H + r4) r4 = 40H-FFH	X		
	A, [HL]	0001 0110	0101 1110	2	8/10	A ← A V (HL)	X		
XOR	A, #byte	1010 1101	Data	2	2	A ← A ⊻ byte	X		
	saddr, #byte	0110 1101 Data	Saddr-offset	3	3	(saddr) ← (saddr) ⊻ byte	X		
	sfr, #byte	0000 0001 Sfr-offset	0110 1101 Data	4	9	sfr ← sfr ⊻ byte	V		
	r, r	1000 1101	$0R_6R_5R_4$ $0R_2R_1R_0$	2	3	r ← r ⊻ r	X		
	A, saddr	1001 1101	Saddr-offset	2	3	A ← A ⊻ (saddr)	X		
	A, sfr	0000 0001 Sfr-offset	1001 1101	3	7	A ← A ⊻ sfr	X		
	A, [r4]	0001 0110	$011R_4$ 1101	2	7	A ← A ⊻ (FE00H + r4) r4 = 40H-FFH	X		
	A, [HL]	0001 0110	0101 1101	2	8/10	A ← A ⊻ (HL)	X		
CMP	A, #byte	1010 1111	Data	2	2	A − byte	X	X	X
	saddr, #byte	0110 1111 Data	Saddr-offset	3	3	(saddr) − byte	X	X	X
	sfr, #byte	0000 0001 Sfr-offset	0110 1111 Data	4	7	sfr − byte	X	X	X
	r, r	1000 1111	$0R_6R_5R_4$ $0R_2R_1R_0$	2	3	r − r	X	X	X
	A, saddr	1001 1111	Saddr-offset	2	3	A − (saddr)	X	X	X
	A, sfr	0000 0001 Sfr-offset	1001 1111	3	7	A − sfr	X	X	X
	A, [r4]	0001 0110	$011R_4$ 1111	2	7	A − (FE00H + r4) r4 = 40H-FFH	X	X	X
	A, [HL]	0001 0110	0101 1111	2	8/10	A − (HL)	X	X	X
16-Bit Operation Instructions									
ADDW	AX, #word	0010 1101 High byte	Low byte	3	4	AX, CY ← AX + word	X	X	X
	AX, rp	1000 1000	$0000\ 1P_2P_10$	2	6	AX, CY ← AX + rp	X	X	X
	AX, saddrp	0001 1101	Saddr-offset	2	7	AX, CY ← AX + (saddrp + 1) (saddrp)	X	X	X
	AX, sfrp	0000 0001 Sfr-offset	0001 1101	3	13	AX, CY ← AX + sfrp	X	X	X

Table 28. Instruction Encodings (cont)

Mnemonic	Operand	Instruction Code B1/B3	Instruction Code B2/B4	Bytes	Clocks	Operation	Flags Z	AC	CY
16-Bit Operation Instructions (cont)									
SUBW	AX, #word	0010 1110	Low byte	3	4	AX, CY ← AX − word	X	X	X
		High byte							
	AX, rp	1000 1010	0000 $1P_2P_1$0	2	6	AX, CY ← AX − rp	X	X	X
	AX, saddrp	0001 1110	Saddr-offset	2	7	AX, CY ← AX − (saddrp + 1) (saddrp)	X	X	X
	AX, sfrp	0000 0001	0001 1110	3	13	AX, CY ← AX − sfrp	X	X	X
		Sfr-offset							
CMPW	AX, #word	0010 1111	Low byte	3	3	AX − word	X	X	X
		High byte							
	AX, rp	1000 1111	0000 $1P_2P_1$0	2	5	AX − rp	X	X	X
	AX, saddrp	0001 1111	Saddr-offset	2	6	AX − (saddrp + 1) (saddrp)	X	X	X
	AX, sfrp	0000 0001	0001 1111	3	12	AX − sfrp	X	X	X
		Sfr-offset							
Multiplication/Division Instructions									
MULUW	r	0000 0101	0000 $0R_2R_1R_0$	2	43	AX (high-order 16 bits), r (low-order 8 bits) ← AX x r			
DIVUW	r	0000 0101	0001 $1R_2R_1R_0$	2	71	AX (dividend), r (remainder) ← AX ÷ r			
Increment and Decrement Instructions									
INC	r	1100 $0R_2R_1R_0$		1	2	r ← r + 1	X	X	
	saddr	0010 0110	Saddr-offset	2	2	(saddr) ← (saddr) + 1	X	X	
DEC	r	1100 $1R_2R_1R_0$		1	2	r ← r − 1	X	X	
	saddr	0010 0111	Saddr-offset	2	2	(saddr) ← (saddr) − 1	X	X	
INCW	rp	0100 $01P_1P_0$		1	3	rp ← rp + 1			
DECW	rp	0100 $11P_1P_0$		1	3	rp ← rp − 1			
Shift Rotate Instructions									
ROR	r, n	0011 0000	$01N_2N_1$ $N_0R_2R_1R_0$	2	3+2n	(CY, $r_7 \leftarrow r_0$, $r_{m-1} \leftarrow r_m$) x n times n = 0-7			X
ROL	r, n	0011 0001	$01N_2N_1$ $N_0R_2R_1R_0$	2	3+2n	(CY, $r_0 \leftarrow r_7$, $r_{m+1} \leftarrow r_m$) x n times n = 0-7			X
RORC	r, n	0011 0000	$00N_2N_1$ $N_0R_2R_1R_0$	2	3+2n	(CY ← r_0, r_7 ← CY, $r_{m-1} \leftarrow r_m$) x n times n = 0-7			X
ROLC	r, n	0011 0001	$00N_2N_1$ $N_0R_2R_1R_0$	2	3+2n	(CY ← r_7, r_0 ← CY, $r_{m+1} \leftarrow r_m$) x n times n = 0-7			X
SHR	r, n	0011 0000	$10N_2N_1$ $N_0R_2R_1R_0$	2	3+2n	(CY ← r_0, r_7 ← 0, $r_{m-1} \leftarrow r_m$) x n times n = 0-7	X	O	X
SHL	r, n	0011 0001	$10N_2N_1$ $N_0R_2R_1R_0$	2	3+2n	(CY ← r_7, r_0 ← 0, $r_{m+1} \leftarrow r_m$) x n times n = 0-7	X	O	X
SHRW	rp, n	0011 0000	$11N_2N_1$ $N_0P_2P_1$0	2	3+3n	(CY ← rp_0, rp_{15} ← 0, $rp_{m-1} \leftarrow rp_m$) x n times n = 0-7	X	O	X
SHLW	rp, n	0011 0001	$11N_2N_1$ $N_0P_2P_1$0	2	3+3n	(CY ← rp_{15}, rp ← 0, $rp_{m+1} \leftarrow rp_m$) x n times n = 0-7	X	O	X
ROR4	[r4]	0000 0101	1000 $10R_1$0	2	22	$A_{3-0} \leftarrow (FE00 + r4)_{3-0}$, $(FE00 + r4)_{7-4} \leftarrow A_{3-0}$, $(FE00 + r4)_{3-0} \leftarrow (FE00 + r4)_{7-4}$			
ROL4	[r4]	0000 0101	1001 $10R_1$1	2	23	$A_{3-0} \leftarrow (FE00 + r4)_{7-4}$, $(FE00 + r4)_{3-0} \leftarrow A_{3-0}$, $(FE00 + r4)_{7-4} \leftarrow (FE00 + r4)_{3-0}$			

Table 28. Instruction Encodings (cont)

Mnemonic	Operand	Instruction Code B1/B3	Instruction Code B2/B4	Bytes	Clocks	Operation	Flags Z	Flags AC	Flags CY
BCD Correct Instructions									
ADJBA		0000 1110		1	3	Decimal adjust accumulator after addition	X	X	X
ADJBS		0000 1111		1	3	Decimal adjust accumulator after subtract	X	X	X
Bit Operation Instructions									
MOV1	CY, saddr. bit	0000 1000 Saddr-offset	0000 $0B_2B_1B_0$	3	5	CY ← (saddr. bit)			X
	CY, sfr. bit	0000 1000 Sfr-offset	0000 $1B_2B_1B_0$	3	7	CY ← sfr.bit			X
	CY, A. bit	0000 0011	0000 $1B_2B_1B_0$	2	5	CY ← A. bit			X
	CY, X. bit	0000 0011	0000 $0B_2B_1B_0$	2	5	CY ← X. bit			X
	CY, PSW. bit	0000 0010	0000 $0B_2B_1B_0$	2	5	CY ← PSW. bit			X
	saddr. bit, CY	0000 1000 Saddr-offset	0001 $0B_2B_1B_0$	3	8	(saddr. bit) ← CY			
	sfr. bit, CY	0000 1000 Sfr-offset	0001 $1B_2B_1B_0$	3	12	sfr. bit ← CY			
	A. bit, CY	0000 0011	0001 $1B_2B_1B_0$	2	8	A. bit ← CY			
	X, bit, CY	0000 0011	0001 $0B_2B_1B_0$	2	8	X. bit ← CY			
	PSW. bit, CY	0000 0010	0001 $0B_2B_1B_0$	2	7	PSW. bit ← CY	X	X	
AND1	CY, saddr. bit	0000 1000 Saddr-offset	0010 $0B_2B_1B_0$	3	5	CY ← CY ∧ (saddr. bit)			X
	CY, /saddr. bit	0000 1000 Saddr-offset	0011 $0B_2B_1B_0$	3	5	CY ← CY ∧ $\overline{\text{(saddr. bit)}}$			X
	CY, sfr. bit	0000 1000 Sfr-offset	0010 $1B_2B_1B_0$	3	7	CY ← CY ∧ sfr. bit			X
	CY, /sfr. bit	0000 1000 Sfr-offset	0011 $1B_2B_1B_0$	3	7	CY ← CY ∧ $\overline{\text{sfr. bit}}$			X
	CY, A. bit	0000 0011	0010 $1B_2B_1B_0$	2	5	CY ← CY ∧ A. bit			X
	CY, /A. bit	0000 0011	0011 $1B_2B_1B_0$	2	5	CY ← CY ∧ $\overline{\text{A. bit}}$			X
	CY, X. bit	0000 0011	0010 $0B_2B_1B_0$	2	5	CY ← CY ∧ X. bit			X
	CY, /X. bit	0000 0011	0011 $0B_2B_1B_0$	2	5	CY ← CY ∧ $\overline{\text{X. bit}}$			X
	CY, PSW. bit	0000 0010	0010 $0B_2B_1B_0$	2	5	CY ← CY ∧ PSW. bit			X
	CY, /PSW. bit	0000 0010	0011 $0B_2B_1B_0$	2	5	CY ← CY ∧ $\overline{\text{PSW. bit}}$			X

Table 28. Instruction Encodings (cont)

Mnemonic	Operand	Instruction Code B1/B3	Instruction Code B2/B4	Bytes	Clocks	Operation	Flags Z	AC	CY
Bit Operation Instructions (cont)									
OR1	CY, saddr. bit	0000 1000 Saddr-offset	0100 $0B_2B_1B_0$	3	5	CY ← CY V (saddr. bit)			X
	CY, /saddr. bit	0000 1000 Saddr-offset	0101 $0B_2B_1B_0$	3	5	CY ← CY V $\overline{\text{(saddr. bit)}}$			X
	CY, sfr. bit	0000 1000 Sfr-offset	0100 $1B_2B_1B_0$	3	7	CY ← CY V sfr. bit			X
	CY, /sfr. bit	0000 1000 Saddr-offset	0101 $1B_2B_1B_0$	3	7	CY ← CY V $\overline{\text{sfr. bit}}$			X
	CY, A. bit	0000 0011	0100 $1B_2B_1B_0$	2	5	CY ← CY V A. bit			X
	CY, /A. bit	0000 0011	0101 $1B_2B_1B_0$	2	5	CY ← CY V $\overline{\text{A. bit}}$			X
	CY, X. bit	0000 0011	0100 $0B_2B_1B_0$	2	5	CY ← CY V X. bit			X
	CY, /X. bit	0000 0011	0101 $0B_2B_1B_0$	2	5	CY ← CY V $\overline{\text{X. bit}}$			X
	CY, PSW. bit	0000 0010	0100 $0B_2B_1B_0$	2	5	CY ← CY V PSW. bit			X
	CY, /PSW. bit	0000 0010	0101 $0B_2B_1B_0$	2	5	CY ← CY V $\overline{\text{PSW. bit}}$			X
XOR1	CY, saddr. bit	0000 1000 Saddr-offset	0110 $0B_2B_1B_0$	3	5	CY ← CY ⊻ (saddr, bit)			X
	CY, sfr. bit	0000 1000 Sfr-offset	0110 $1B_2B_1B_0$	3	7	CY ← CY ⊻ sfr. bit			X
	CY, A. bit	0000 0011	0110 $1B_2B_1B_0$	2	5	CY ← CY ⊻ A. bit			X
	CY, X. bit	0000 0011	0110 $0B_2B_1B_0$	2	5	CY ← CY ⊻ X. bit			X
	CY, PSW. bit	0000 0010	0110 $0B_2B_1B_0$	2	5	CY ← CY ⊻ PSW. bit			X
SET1	saddr. bit	1011 $0B_2B_1B_0$	Saddr-offset	2	3	(saddr. bit) ← 1			
	sfr. bit	0000 1000 Sfr-offset	1000 $1B_2B_1B_0$	3	10	sfr. bit ← 1			
	A. bit	0000 0011	1000 $1B_2B_1B_0$	2	6	A. bit ← 1			
	X. bit	0000 0011	1000 $0B_2B_1B_0$	2	6	X. bit ← 1			
	PSW. bit	0000 0010	1000 $0B_2B_1B_0$	2	5	PSW. bit ← 1	X	X	X
CLR1	saddr. bit	1010 $0B_2B_1B_0$	Saddr-offset	2	3	(saddr. bit) ← 0			
	sfr. bit	0000 1000 Sfr-offset	1001 $1B_2B_1B_0$	3	10	sfr. bit ← 0			
	A. bit	0000 0011	1001 $1B_2B_1B_0$	2	6	A. bit ← 0			
	X. bit	0000 0011	1001 $0B_2B_1B_0$	2	6	X. bit ← 0			
	PSW. bit	0000 0010	1001 $0B_2B_1B_0$	2	5	PSW. bit ← 0	X	X	X
NOT1	saddr. bit	0000 1000 Saddr-offset	0111 $1B_2B_1B_0$	3	6	(saddr. bit) ← $\overline{\text{(saddr. bit)}}$			
	sfr. bit	0000 1000 Sfr-offset	0111 $1B_2B_1B_0$	3	10	sfr. bit ← $\overline{\text{sfr. bit}}$			
	A. bit	0000 0011	0111 $1B_2B_1B_0$	2	6	A. bit ← $\overline{\text{A. bit}}$			
	X. bit	0000 0011	0111 $0B_2B_1B_0$	2	6	X. bit ← $\overline{\text{X. bit}}$			
	PSW. bit	0000 0010	0111 $0B_2B_1B_0$	2	5	PSW. bit ← $\overline{\text{PSW. bit}}$	X	X	X

Table 28. Instruction Encodings (cont)

Mnemonic	Operand	Instruction Code B1/B3	Instruction Code B2/B4	Bytes	Clocks	Operation	Flags Z	AC	CY
Bit Operation Instructions (cont)									
SET1	CY	0100 0001		1	2	CY ← 1			1
CLR1	CY	0100 0000		1	2	CY ← 0			0
NOT1	CY	0100 0010		1	2	CY ← $\overline{CY}$			X
Call Return Instructions									
CALL	laddr14	0010 1000 High addr	Low addr	3	9	(SP−1)(SP−2) ← PC + 3, PC ← addr14, SP ← SP−2			
CALLF	laddr11	1001 0 ← fa		2	9	(SP−1)(SP−2) ← PC + 2, PC_{12-11} ← 01, PC_{10-0} ← laddr11, SP ← SP−2			
CALLT	[addr5]	111 ← ta →		1	12	(SP−1)(SP−2) ← PC + 1, PC_H ← (addr5 + 1), PC_L ← (addr5), SP ← SP−2			
RET		0101 0110		1	8	PC_L ← (SP), PC_H ← (SP + 1), SP ← SP + 2			
RETI		0101 0111		1	10	PC_L ← (SP), PC_H ← (SP + 1), PSW ← (SP + 2), SP ← SP + 3	R	R	R
Stack Operation Instructions									
PUSH	rp	0011 11P_1P_0		1	7	(SP−1) ← rp_H, (SP−2) ← rp_L, SP ← SP−2			
	PSW	0100 1001		1	3	(SP−1) ← PSW, SP ← SP−1			
POP	rp	0011 01P_1P_0		1	8	rp_L ← (SP), rp_H ← (SP + 1), SP ← SP + 2			
	PSW	0100 1000		1	4	PSW ← (SP), SP ← SP + 1	R	R	R
MOV	SP. #byte	0010 1011 Data	1111 1100	3	5	SP ← byte			
	SP. A	0001 0010	1111 1100	2	5	SP ← A			
	A. SP	0001 0000	1111 1100	2	4	A ← SP			
Unconditional Branch Instructions									
BR	laddr14	0010 1100 High addr	Low addr	3	5	PC ← laddr14			
	rp	0000 0101	0100 1$P_2P_1$0	2	5	PC_H ← rp_H, PC_L ← rp_L			
	$addr14	0001 0100	jdisp	2	4	PC ← $addr14			
Conditional Branch Instructions									
BC BL	$addr14	1000 0011	jdisp	2	4(2)	PC ← $addr14 if CY = 1			
BNC BNL	$addr14	1000 0010	jdisp	2	4(2)	PC ← $addr14 if CY = 0			
BZ BE	$addr14	1000 0001	jdisp	2	4(2)	PC ← $addr14 if Z = 1			
BNZ BNE	$addr14	1000 0000	jdisp	2	4(2)	PC ← $addr14 if Z = 0			

Table 28. Instruction Encodings (cont)

Mnemonic	Operand	Instruction Code B1/B3	Instruction Code B2/B4	Bytes	Clocks	Operation	Flags Z AC CY
Conditional Branch Instructions (cont)							
BT	saddr. bit, $addr14	0111 $0B_2B_1B_0$ jdisp	Saddr-offset	3	6(4)	PC ← $addr 14 if (saddr. bit) = 1	
	sfr. bit, $addr14	0000 1000 Sfr-offset	1011 $1B_2B_1B_0$ jdisp	4	9(7)	PC ← $addr14 if sfr. bit = 1	
	A. bit, $addr14	0000 0011 jdisp	1011 $1B_2B_1B_0$	3	7(5)	PC ← $addr14 if A. bit = 1	
	X. bit, $addr14	0000 0011 jdisp	1011 $0B_2B_1B_0$	3	7(5)	PC ← $addr14 if X. bit = 1	
	PSW, bit, $addr14	0000 0010 jdisp	1011 $0B_2B_1B_0$	3	7(5)	PC ← $addr14 if PSW. bit = 1	
BF	saddr. bit, $addr14	0000 1000 Saddr-offset	1010 $0B_2B_1B_0$ jdisp	4	7(5)	PC ← $addr14 if (saddr. bit) = 0	
	sfr. bit, $addr14	0000 1000 Sfr-offset	1010 $1B_2B_1B_0$ jdisp	4	9(7)	PC ← $addr14 if sfr. bit = 0	
	A. bit, $addr14	0000 0011 jdisp	1010 $1B_2B_1B_0$	3	7(5)	PC ← $addr14 if A. bit = 0	
	X. bit, $addr14	0000 0011 jdisp	1010 $0B_2B_1B_0$	3	7(5)	PC ← $addr14 if X. bit = 0	
	PSW. bit, $addr14	0000 0010 jdisp	1010 $0B_2B_1B_0$	3	7(5)	PC ← $addr14 if PSW. bit = 0	
BTCLR	saddr. bit, $addr14	0000 0011 Saddr-offset	1101 $0B_2B_1B_0$ jdisp	4	9(5)	PC ← $addr14 if (saddr. bit) = 1 then reset (saddr. bit)	
	sfr. bit, $addr14	0000 1000 Sfr-offset	1101 $1B_2B_1B_0$ jdisp	4	13(7)	PC ← $addr14 if sfr. bit = 1 then reset sfr. bit	
	A. bit, $addr14	0000 0011 jdisp	1101 $1B_2B_1B_0$	3	9(5)	PC ← $addr14 if A. bit = 1 then reset A. bit	
	X. bit, $addr14	0000 0011 jdisp	1101 $0B_2B_1B_0$	3	9(5)	PC ← $addr14 if X. bit = 1 then reset X. bit	
	PSW. bit, $addr14	0000 0010 jdisp	1101 $0B_2B_1B_0$	3	8(5)	PC ← $addr14 if PSW. bit = 1 then reset PSW. bit	X X X
DBNZ	r2, $addr14	0011 $001R_0$	jdisp	2	5(3)	r2 ← r2 − 1, then PC ← addr14 if r2 = 0	
	saddr, $addr14	0011 1011 jdisp	Saddr-offset	3	6(4)	saddr ← saddr − 1, then PC ← $addr14 if saddr ≠ 0	
CPU Control Instructions							
MOV	WDMSR, #byte	0000 1001 Data	0110 1101 Data	4	12	WDMSR ← byte	
SEL	RBn	0000 0101	1010 $10N_1N_0$	2	2	RBS1−0 ← n n = 0-3	
NOP		0000 0000		1	2	No operation	
EI		0100 1011		1	2	IE ← 1 (enable interrupt)	
DI		0100 1010		1	2	IE ← 0 (disable interrupt)	

MODEM FUNCTION BLOCK FUNCTIONAL DESCRIPTION

For a general overview of the modem function block units, see Figure 1.

Parallel I/O Port and C-RAM

General. The modem function block, ports A, B, and D are 8-bit general-purpose I/O ports. Port C is a 7-bit general-purpose I/O port. These ports are selectively set to be used as an input or output port by the internal program using three mode registers PTMR, PCMR, and PDMR.

Port D has a data bus function for transferring data to and from an external unit. The processor has an internal C-RAM used as a 16-byte buffer to transfer data to and from an external unit.

Port E is a one-bit input port and port F is a 3-bit output port. See table 29.

Table 29. Parallel I/O Ports and C-RAM SFR Addresses and Descriptions

Unit	SFR Address	Description
PTMR	FF28H	Ports A, B, TxD, and Rxd mode registers
PCMR	FF29H	Port C mode register
PDMR	FF2AH	Port D mode register
Port A	FF2CH	8-bit general-purpose I/O port
Port B	FF2DH	8-bit general-purpose I/O port
Port C	FF2EH	7-bit general-purpose I/O port
Port D	FF2FH	8-bit general-purpose I/O port
Port E	FF57H	1-bit general-purpose input port
Port F	FF5CH	3-bit general-purpose output port
C-RAM	FF90H to FF9FH	Memory used to transfer data to and from an external unit

Functions. Port A (PA_0 to PA_7): Port A is a general-purpose I/O port consisting of an 8-bit register. Its SFR address is FF2CH. It is used as an input port when the processor is reset and 00H is the initial value of the internal buffer. Bits 4 and 5 of the PTMR ($PTMR_4$ and $PTMR_5$) determines whether each of two 4-bit groups are input or output.

Port B (PB_0 to PB_7): Port B is a general-purpose I/O port consisting of an 8-bit register. Its SFR address is FF2DH. It is used as an input port when the processor is reset and 00H is the initial value of the internal buffer. Bits 0 through 3 of the PTMR ($PTMR_0$ through $PTMR_3$) determines whether each of three 2-bit groups are input or output.

Port C (PC_0 to PC_6): Port C is a general-purpose I/O port when PCMR bit 7 = 0 and consists of a 7-bit register. Its SFR address is FF2EH. It is used as an input port when the processor is reset and 00H is the initial value of the internal buffer. I/O is selectable on a bit basis as a general-purpose I/O port. It is determined by the seven bits in the PCMR register (bits 0 through 6).

Port C functions as bus control when PCMR bit 7 = 1. In this mode, the port inputs C-RAM addresses and Read/Write signals from the host computer. See table 30.

Table 30. Port C Functions

Pin Symbol as a Port	Pin Symbol as a Bus	Function as a Bus
PC_0–PC_3	A_0–A_3	C-RAM address input
PC_4	$\overline{CS}$	Chip select input
PC_5	$\overline{RD}$	Read strobe input from host computer
PC_6	$\overline{WR}$	Write strobe input from host computer

Port D (PD_0 to PD_7): Port D is a general-purpose I/O when PCMR bit 7 = 0 and consists of an 8-bit register. Its SFR address is FF2FH. It is used as an input port when the processor is reset and 00H is the initial value of the internal buffer. I/O is selectable on a bit basis as a general-purpose I/O port. It is determined by eight bits (PDMR bit 0 through PDMR bit 7) in the PDMR register. See table 33.

Port D functions as a data bus when PCMR bit 7 = 1. In this mode, port C is used as an address and control bus for an 16-byte data bus of C-RAM and its address is specified by A_0 through A_3 of port C.

Ports A through D are all selected as input ports when the processor is initialized or reset. The ports when selected as input or output ports, can be either written or read. When used as an output port, the following applies:

- Write – GPP data will be written to the external port.
- Read – the most recent data written to the port from the GPP will be read back to the GPP.

When used as an input port, the following applies:

- Write – GPP data will be written to the external port.
- Read – the external data that is input to the port is read into the GPP.

Port E (PE): Port E is a general-purpose input port consisting of a 1-bit register. Its SFR address is FF57H. The input value can be read from bit 0 of the G-bus. The remaining bits 1 through 7, contain 0's. No data can be written to port E.

Port F (PF_0 to PF_2): Port F is a general-purpose output port consisting of a 3-bit register. Its SFR address is FF5CH. The internal register is set as 0H when the processor is reset.

Bits 0 through 2 of the G-bus are output bits. The port can also be read. Bits 3 through 7 contain 0's.

Port Mode Register (PTMR): The PTMR consists of an 8-bit register. It selects the mode of ports A and B and the RxD and TxD pins. Its address is FF28H and is 3FH when the processor is initialized and reset. See table 31.

Table 31. PTMR Functions

PTMR	Value	Function
Bit 7	0	RxD is an output port and TxD is an input port
	1	RxD and TxD are not used as a port
Bit 6		Not used
Bit 5	0	PA_4 to PA_7 (high-order four bits) are output ports
	1	PA_4 to PA_7 (high-order four bits) are input ports
Bit 4	0	PA_0 to PA_3 (low-order four bits) are output ports
	1	PA_0 to PA_3 (low-order four bits) are input ports
Bit 3	0	PB_7 and PB_6 are output ports
	1	PB_7 and PB_6 are input ports
Bit 2	0	PB_5 and PB_4 are output ports
	1	PB_5 and PB_4 are input ports
Bit 1	0	PB_3 and PB_2 are output ports
	1	PB_3 and PB_2 are input ports
Bit 0	0	PB_1 and PB_0 are output ports
	1	PB_1 and PB_0 are input ports

Port C Mode Register (PCMR): The PCMR consists of an 8-bit register. Its address is F29H and is 7FH when the processor is reset. See table 32.

Table 32. PCMR Functions

PCMR	Value	Function
Bit 7	0	Ports C and D are set as a port
	1	Ports C and D are set as a bus
Bit n n = 0–6	0	PCn is an output port (valid when PCMR bit 7 = 0)
	1	PCn is an input port (valid when PCMR bit 7 = 0)

Port D Mode Register (PDMR): The PDMR consists of an 8-bit register. It selects port D in the input or output mode. Its SFR address is FF29H and is FFH when the processor is reset. See table 33.

Table 33. PDMR Functions

PDMR	Value	Function
Bit n n = 0–7	0	PDn is an output port (valid when PCMR bit 7 = 0)
	1	PDn is an input port (valid when PCMR bit 7 = 0)

Control RAM (C-RAM): C-RAM is a 16-byte by 8-bit memory. It is mapped as SFR addresses FF90H through FF9FH. Table 34 shows the relationship between C-RAM and SFR addresses, when PC_0 to PC_3 of port C are used as an address bus.

Table 34. C-RAM SFR Address Mapping

External Address (PC_3–PC_0)					
A3	A2	A1	A0	(HEX)	SFR Address
0	0	0	0	(0H)	FF90H
0	0	0	1	(1H)	FF91H
	≀				≀
1	1	1	0	(EH)	FF9EH
1	1	1	1	(FH)	FF9FH

Addresses 7H and FH (C-RAM memory external addresses) store information indicating the C-RAM status. 7H indicates the status of 0H to 6H and FH indicates the status of 8H to EH. See table 35.

Table 35. C-RAM Status Functions

7H Memory Bit	Function	FH Memory Bit	Function
Bit 0	0H memory status	Bit 0	8H memory status
Bit 1	1H memory status	Bit 1	9H memory status
≀	≀	≀	≀
Bit 6	6H memory status	Bit 6	EH memory status
Bit 7	0	Bit 7	0

The state of a memory status bit, indicates the following:

0 = No write or read request is issued by the μPD77810.
1 = A write or read request is issued by the μPD77810.

Each status bit is set to 1 by a GPP transfer instruction, but cannot be set to 0 by a GPP transfer instruction. When an external host computer accesses 0H to 6H and 8H to EH, the corresponding status bit is set to 0. All bits are set to 0 when the processor is reset.

A read access to the C-RAM can be performed by the GPP and an external host computer simultaneously, but simultaneous write access is denied.

Scrambler (SCR) and Descrambler (DSC)

Both the scrambler (SCR) and descrambler (DSC) consist of a polynomial counter, a protection circuit, and an SCRMR used to set the mode. Registers SCR and DSC are one-bit registers corresponding to the LSB of the data bus. They also have a 4-bit SCRM register and 3-bit DSCM register as a control register. See table 36.

Table 36. Scrambler and Descrambler SFR Addesses and Descriptions

Unit	SFR Address	Description
SCRMR	FF40H	Mode register
SCR	FF41H	Scrambler port
DSC	FF42H	Descrambler port
SCRM	FF65H	Scrambler control register
DSCM	FF66H	Descrambler control register

Mode Register (SCRMR). The SCRMR is an 8-bit register. Each bit (bit 7 = MSB and bit 0 = LSB) specifies the multiplexer mode as shown in Figures 15 and 16. Bits 7 through 2 are shared by the SCR and DSC. Bit 1 is used only by DSC and bit 0 is used only by the SCR. Table 37 shows the relationship between the SCRMR bit patterns and CCITT V series recommendations.

Table 37. SCRMR Functions

CCITT Recommendation	Bits 7	6	5	4	3	2	1	0	Generating Function
V.22, V.22bis	0	0	0	0	0	0	–	–	$1 + X^{-14} + X^{-17}$ (Note 1)
V.27	1	1	1	0	1	0	–	–	$1 + X^{-6} + X^{-7}$ (Note 2)
V.27bis, V.27ter	1	1	1	1	1	0	–	–	$1 + X^{-6} + X^{-7}$ (Note 3)
V.29	1	–	–	–	1	1	0	0	$1 + X^{-18} + X^{-23}$
V.26/V.32 call	1	–	–	–	1	1	1	0	$1 + X^{-18} + X^{-23}$: SCR $1 + X^{-5} + X^{-23}$: DSC
V.26/V.32 answer	1	–	–	–	1	1	0	1	$1 + X^{-5} + X^{-23}$: SCR $1 + X^{-18} + X^{-23}$: DSC

Notes:

(1) The processor has a protection circuit (conforming to Recommendation V.22) that reverses the next scrambler input, when 1 is output continuously to the scrambler 64 times.

(2) The processor has a protection circuit (conforming to Recommendation V.27) that protects repeated patterns of 1, 2, 3, 4, 6, 9, and 12 in bits 2 through 6. Example of 45 bits of transmitted bit strings follows:

$$P(x) = \sum_{(i=0)}^{32} a(i)\, x^i$$

Where, a (i) = 0 or 1
a (i) = a (i + 9) or a (i + 12)
Bit data is inverted before transmission.

(3) The processor has a protection circuit (conforming to V.27bis and V.27ter Recommendations) that protects repeat patterns of 1, 2, 3, 4, 6, 8, 9, and 12 in bits 2 through 6.

Control Registers (SCRM and DSCM). Table 38 shows the functions of the 4-bit SCRM and the 3-bit DSCM.

Table 38. SCRM and DSCM Functions

Bit	Name	Function
4-Bit SCRM Functions		
3	SCRM.INT	Initial data loading (when the bit changes from 0 to 1)
2	SCRM.CLR	Scrambler clear (when the bit is 1)
1	SCRM.STT	Scrambler protection circuit start (when the bit changes from 0 to 1)
0	SCRM.RST	Scrambler protection circuit reset (when the bit is 1)
3-Bit DSCM Functions		
2	DSCM.CLR	Descrambler clear (when the bit is 1)
1	DSCM.STT	Descrambler protection circuit start (when the bit changes from 0 to 1)
0	DSCM.RST	Descrambler protection circuit reset (when the bit is 1)

T_XPLL and R_XPLL

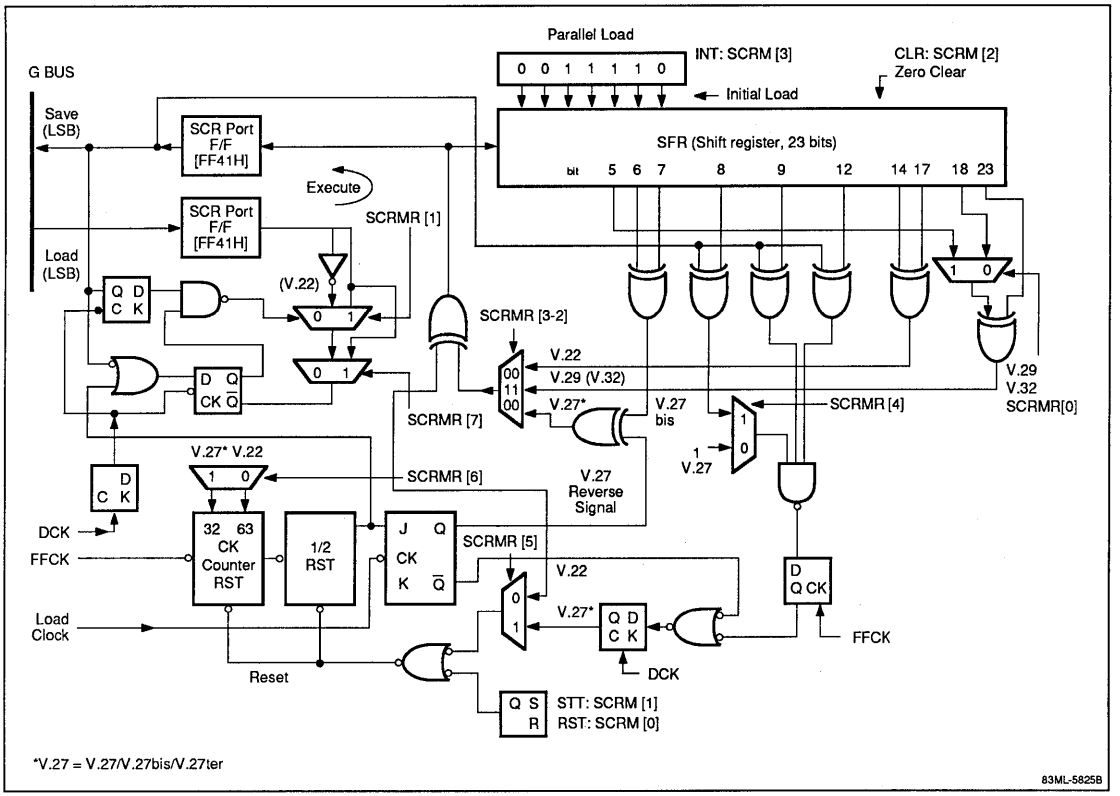
The transmitting phase-locked loop (T_XPLL) and the receiving phase-locked loop (R_XPLL) consist of a group of counters, including some that are only partially resettable, a preset controller, and a PLL mode register (PLLMR) to set the mode. See Figure 17. The T_XPLL adjusts the phase to the external bit rate clock. R_XPLL adjusts it to the phase detected internally. The A/D and D/A precisions (bit length) are set by the SR register (see DSP internal functions) and DAMR register (see A/D and D/A interface) in DSP.

Table 39 lists the clocks used by T_XPLL and R_XPLL.

Table 39. T_XPLL and R_XPLL Clocks

Pin Symbol	Clock Function
ADST	A/D sampling clock
$\overline{\text{ADCK}}$	A/D data serial clock
RT	Received data bit rate clock (1 in asynchronous mode)
RBAUD	Received data baud rate clock
DALD	D/A data load strobe clock
$\overline{\text{DACK}}$	D/A data serial clock
STINT	Transmited data bit rate clock (1 in asynchronous mode)
SBAUD	Transmitted data baud rate clock
ST16	16-time clock of transmitted bit rate used in ASC block
RT16	16-time clock of received bit rate used in SAC block

Figure 15. SCR Block Diagram



T_XPLL. T_XPLL is a PLL whose theory of operation is based on a frequency divider with an adjustable ratio. The incrementer (INCR) shown in Figure 18 is incremented by one at an input clock rate of 5.5296 MHz. When INCR reaches the number 6 it inputs either 0, 1, or 2 as determined by the 2-bit FLIP/FLOP (TF0 and TF1). Table 40 shows the TF0 and TF1 values.

Table 40. TF0/RF0 and TF1/RF1 Values

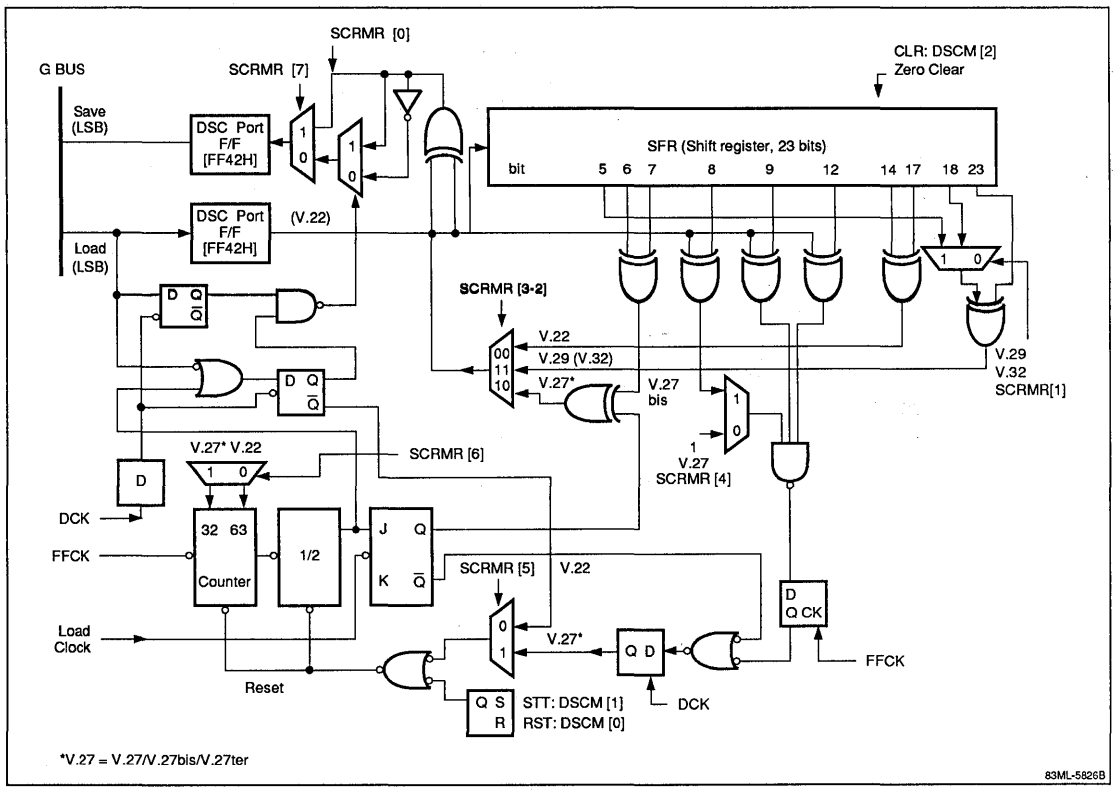
TF0/RF0	TF1/RF1	Data to be Loaded in INCR
0	0	1 (1/6)
1	0	2 (1/5)
0	1	0 (1/7)
1	1	Inhibited

In Figure 18, CMP is a phase comparator that outputs the phase of STEXT or RT at the rising edge of SBAUD. CNT is an incremental/decremental counter. It is incremented or decremented by one according to the CMP output. When CMP reaches +3 or more, it issues a TF0 set signal; when −3 or less, it issues a TF1 set signal. TF0 and TF1 are updated every time an SBAUD is generated. TF0 and TF1 select the multiplexer output and change the timing for the INCR division ratio to 2400 Hz/9600 Hz, selectable by PLLMR.

At the rising edge of STINT, an interrupt signal IST is output.

R_XPLL. R_XPLL is a PLL whose theory of operation is based on a frequency divider with an adjustable ratio.

Figure 16. DSC Block Diagram



In Figure 19, INCR is an incrementer that is incremented by an input clock rate of 5.5296 MHz. When INCR reaches the number 6, it inputs 0, 1, or 2 at the next increment. Data to be loaded is determined by the 2-bit FLIP/FLOP (RF0 and RF1). Table 40 shows the RF0 and RF1 values.

RF0 and RF1 are set or reset with a DSP instruction (write instruction to the SR register). These bits select the multiplexer output and change the timing for the INCR division ratio to 2400 Hz/9600 Hz, selectable by PLLMR. At the rising edge of RTINT, an interrupt signal IRT is output.

Mode Registers. Table 41 shows the PLLMR1, PLLMR2, and BAUDSR register SFR addresses.

Table 41. PLLMR1, PLLMR2, and BAUDSR Register SFR Addresses

Unit	SFR Address	Description
PLLMR1	FF44H	PLL mode register 1 (8 bits)
PLLMR2	FF7EH	PLL mode register 2 (8 bits)
BAUDSR	FF45H (low-order 2 bits)	SBAUD and RBAUD status register (2 bits)

Mode Register PLLMR1: The PLLMR1 mode register is an 8-bit register. Each bit (bit 0 = LSB) specifies the multiplexer mode. PLLMR1 specifies whether the SBAUD and RBAUD pins are used as a baud rate clock output pin or input port (PG_0 and PG_1). PLLMR1 also performs as an input register when the pins are used as input ports.

Bit 7 (MSB) of the PLLMR1 controls TF0 and TF1 of T_XPLL and bit 6 controls RF0 and RF1 of R_XPLL. Bits 5 and 4 specify the clock source of the transmitting PLL and bit 3 specifies the mode of the SBAUD and RBAUD pins. Bits 1 and 0 enables the SBAUD and RBAUD pins, when the pins are used as input ports. Figure 17 shows the PLLMR1 functions.

The PLLMR1 register uses the SFR address of FF44H. PLLMR1 bits 2 through 7 are set to 0, and bits 0 and 1 are set to an undefined value when the processor is reset.

When PLLMR1 is read immediately after it is written an incorrect value may occur in bits 6 and 7.

Figure 17. PLLMR1 Functions

PLLMR1	
Bit 7	TFO and TF1 Update Cycle
0	2400 Hz
1	9600 Hz

PLLMR1	
Bit 6	RF0 and RF1 Update Cycle
0	2400 Hz
1	9600 Hz

PLLMR1	
Bit 3	SBAUD and RBAUD Pin Mode
0	Input Port
1	Baud Rate Clock Output

PLLMR1	
Bit	Function
1	SBAUD Pin Input Bit
0	RBAUD Pin Input Bit

PLLMR1		
Bit		Transmitter Clock
5	4	
0	0	Internal Clock (STINT) (Self Run)
0	1	External Clock (STEXT)
1	0	Slave Clock (RT)
1	1	Inhibited

Note:

(1) A Frequency Rate of 2400 Hz cannot be used for the update cycle clock in phase control of the Tx PLL.

Mode Register PLLMR2: The PLLMR2 mode register is an 8-bit register. Each bit selects a multiplexer mode. The high-order four bits of PLLMR2 select the transmit (T_XPLL) clock rate and the low-order four bits select the receive (R_XPLL) clock rate.

Table 42 lists the PLLMR2 functions. The PLLMR2 register has an SFR address of FF7EH. The SFR address is 33H when the processor is reset.

SBAUD and RBAUD Status Register BAUDSR: The BAUDSR is a 2-bit read-only register that indicates the SBAUD and RBAUD status. Bit 1 indicates the SBAUD status and bit 0 indicates the RBAUD status 1 or 0.

The BAUDSR register has an SFR address of FF45H. The SFR address is set as 0H when the processor is reset. In read mode, bits 2 through 7 output 0s.

Table 42. PLLMR2 Functions

Bits							Bits						
7	6	5	4	SBAUD (Hz)	STINT (Hz)	ST 16 (Hz)	3	2	1	0	RBAUD (Hz)	RT (Hz)	RT 16 (Hz)
0	0	0	0	300	300	4800	0	0	0	0	300	300	4800
0	1	0	0	600	600	9600	0	1	0	0	600	600	9600
0	0	0	1	600	1200	19200	0	0	0	1	600	1200	19200
0	0	1	1	600	2400	38400	0	0	1	1	600	2400	38400
0	0	1	0	1200	1200	19200	0	0	1	0	1200	1200	19200
0	1	1	0	1200	2400	38400	0	1	1	0	1200	2400	38400
0	1	1	1	1600	4800	76800	0	1	1	1	1600	4800	76800
1	0	0	0	2400	2400	38400	1	0	0	0	2400	2400	38400
1	0	0	1	2400	4800	76800	1	0	0	1	2400	4800	76800
1	0	1	1	2400	9600	153600	1	0	1	1	2400	9600	153600
1	0	1	0	2400	7200	115200	1	0	1	0	2400	7200	115200
1	1	1	0	2400	14400	230400	1	1	1	0	2400	14400	230400
1	1	1	1	2400	19200	307200	1	1	1	1	2400	19200	307200

Note:

(1) There is a possibility that erroneous data could occur if the GPP is allowed to write and read data to the PLLMR2 simultaneously.

ASC, SAC, and UART

This circuit provides a serial interface asynchronously with the DTE. The circuit consists of an asynchronous-to-synchronous converter (ASC), and synchronous-to-asynchronous converter (SAC), and a universal asynchronous receiver transmitter (UART) (URTI for input and URTO for output).

The mode of the ASC and SAC is selected by the ASMR mode register. In synchronous mode, the serial clock is RT and ST. The mode and control of UART is selected by the URTMR mode register and its status is retained in the URTSR register.

This circuit is invalid when PTMR bit 7 = 0. The R_XD pin is used a an output port and the T_XD pin is used a an input port. The LSB of SACR is input to R_XD and T_XD outputs data to the MSB of ASCR.

Table 43 lists the SFR addresses of the ASC, SAC, and UART register. Figure 20 shows the block diagram of the ASC, SAC, and UART.

Table 43. ASC, SAC, UART Register SFR Addresses

Unit	SFR Address	Description
ASMR	FF49H	Asynchronous/synchronous mode register
URTMR	FF4AH (low-order 7 bits)	UART mode register
URTSR	FF4BH (low-order 4 bits)	UART status register
ASCR	FF4CH	ASC register
SACR	FF4DH	SAC register
URO	FF3EH	URO register
URI	FF3FH	URI register

The asynchronous/synchronous mode register (ASMR) is an 8-bit register. It inputs ASC serial data and URTI serial data, selects the R_XD pin output signal, selects the character length and signaling rate range, sets a loop from R_XD to T_XD, and selects aynchronous or synchronous mode. The register contains 00H when the processor is reset. Figure 21 lists the ASMR functions.

Figure 18. T_XPLL Block Diagram

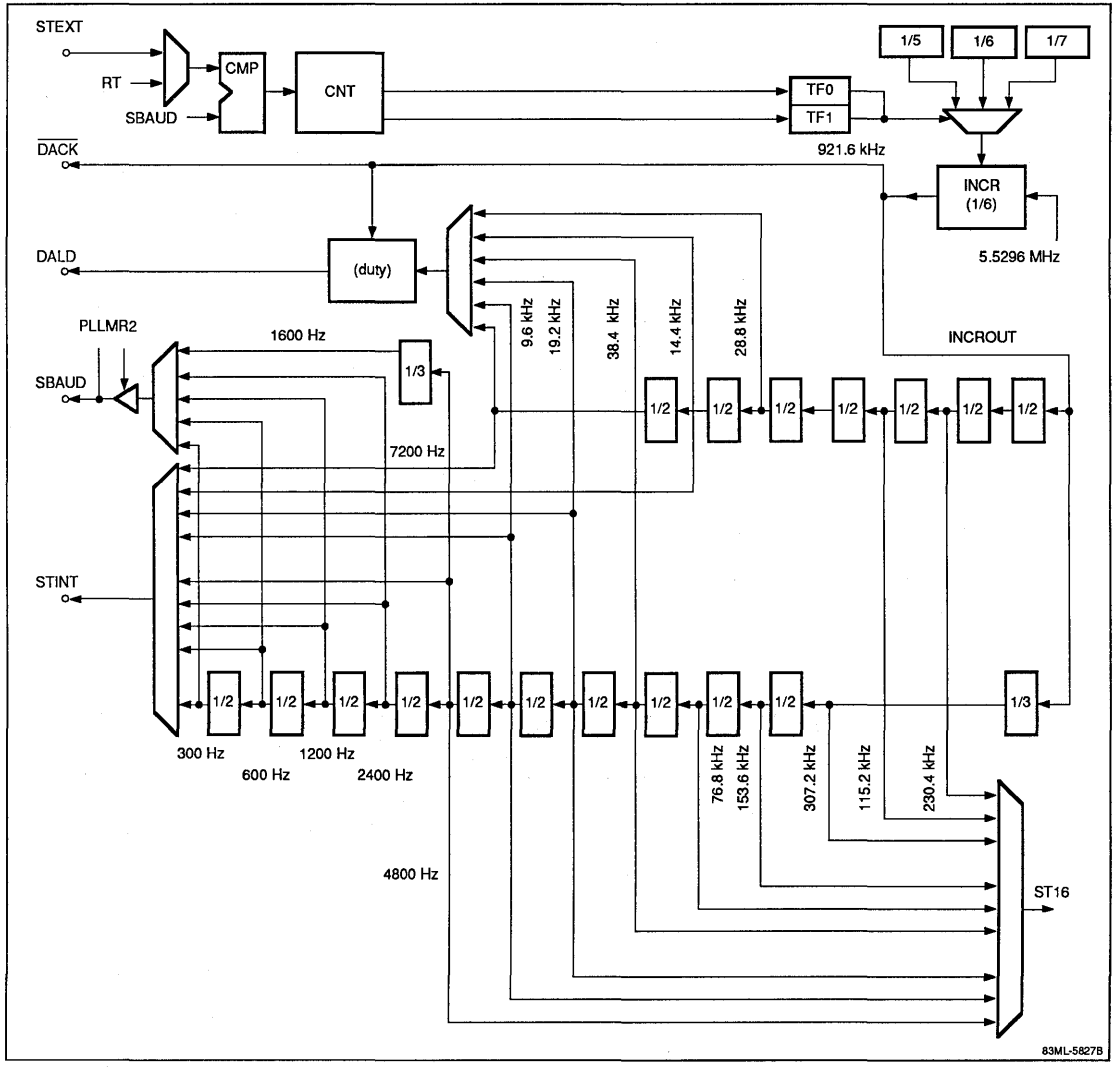


Figure 19. R_XPLL Block Diagram

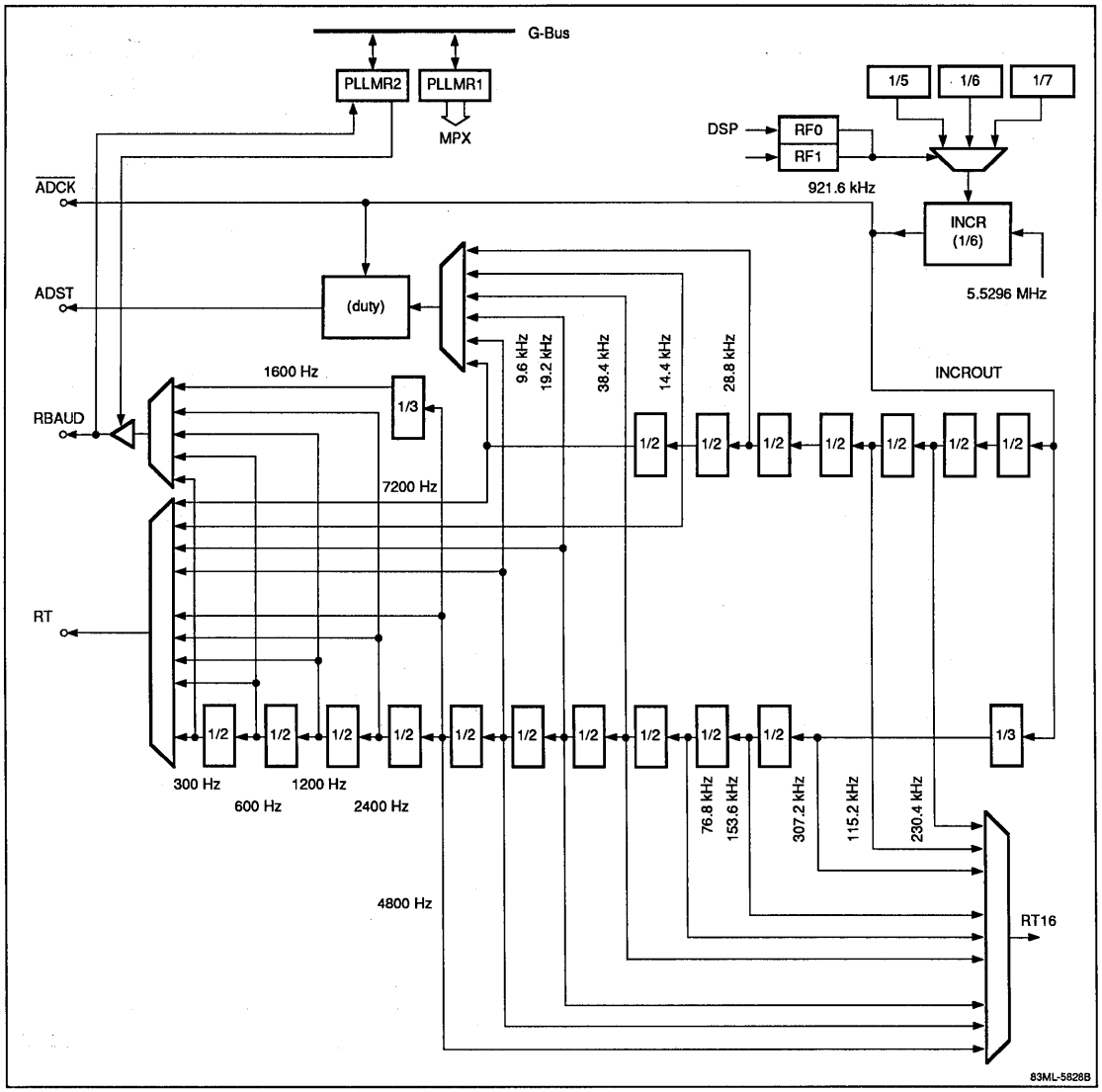
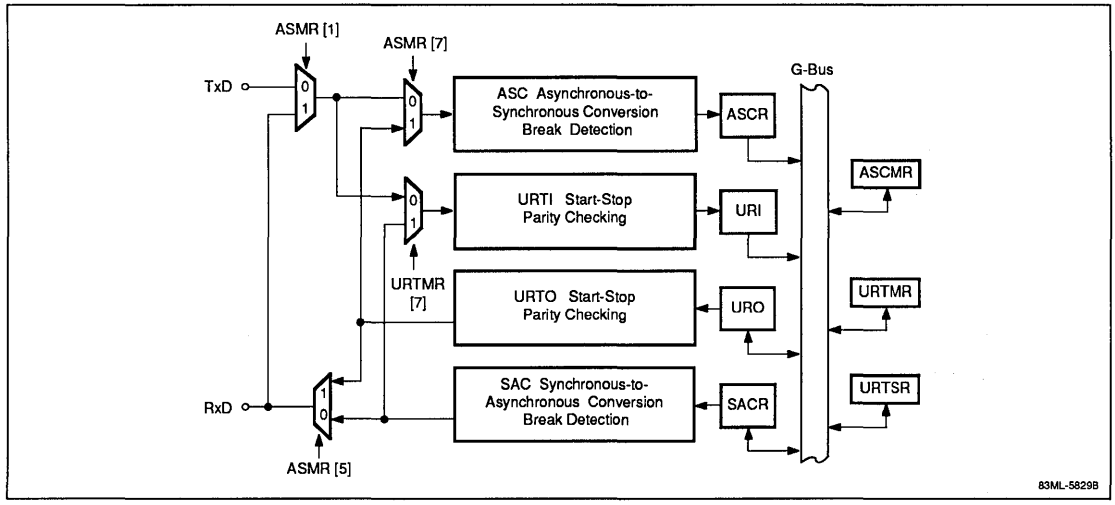


Figure 20. ASC, SAC, UART Block Diagram



ASC. The asynchronous-to-synchronous converter (ASC) converts a start-stop signal that is being input to the T_XD pin to a bit string, which is synchronous to the transmit clock ST of the modem. If the rate of the input signal is high (1% or 2.3%), it deletes the stop bit. ASC also has a break character detection function. When a break character is detected, it generates break signals for 2M + 3 bits (M indicates the character length including the start and stop bits). ASC has an 8-bit ASCR output register that can output data to the G-bus. ASCR inputs data converted from asynchronous to synchronous from the MSB. When data is processed bit by bit, the ASCR MSB has valid data. Figure 22 provides a diagram of the ASC break signal.

Figure 21. ASMR Functions

ASMR	
Bit 7	ASC Serial Input
0	T_XD pin
1	URTO output

ASMR	
Bit 6	ASC Control
0	Input disable
1	Input enable

ASMR	
Bit 5	R_XD Pin Output
0	SAC output
1	URTO output

ASMR		
Bit 4	Bit 3	Character Length M (1)
0	0	8
0	1	9
1	0	10
1	1	11

ASMR	
Bit 1	Loop to R_XD to T_XD
0	No loop
1	Loop (3)

ASMR	
Bit 2	Signaling Rate Range
0	Basic
1	Expanded

ASMR	
Bit 0	Asynchronous/Synchronous
0	Asynchronous (2)
1	Synchronous (2)

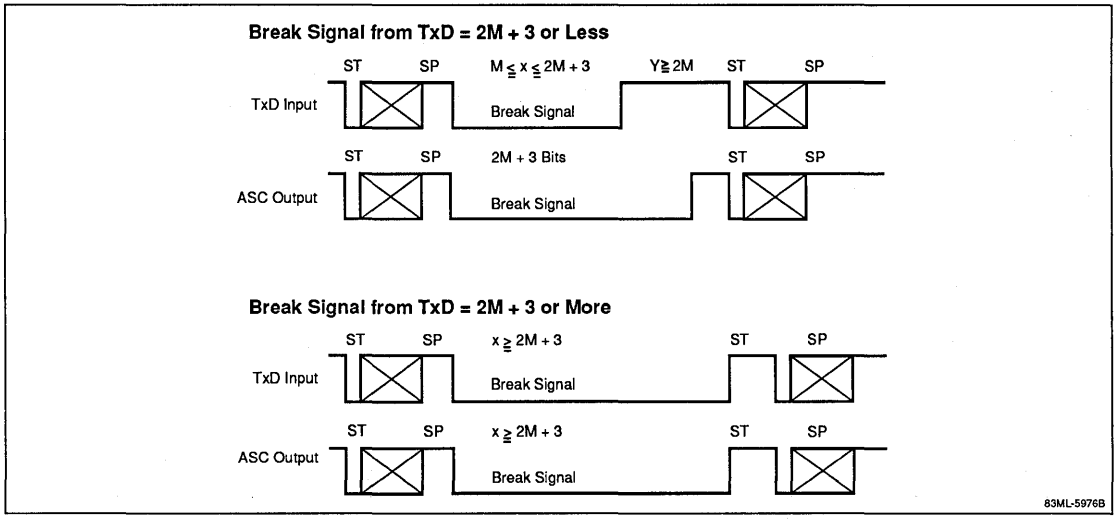
Notes:

(1) Includes Start and Stop Bits

(2) RT and STINT Pins = 1

(3) R_XD Outputs all 1s

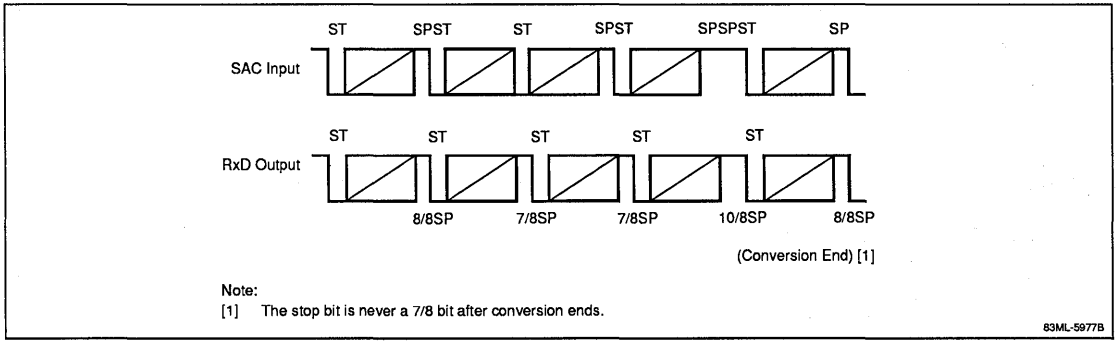
Figure 22. ASC Break Signal Diagram



SAC. The synchronous-to-asynchronous converter (SAC) inserts a stop bit if it is deleted in a circuit that outputs a bit string, which is synchronous to the RT receive clock from the R_XD pin as a start-stop signal. The width of the stop bit to be inserted is shorter than the original stop bit by 1/8 (1/4 in extension mode) and is retained until conversion ends.

If two null codes with a deleted stop bit are continuous (start bit length = 2M – 2 bits), they must be distinguished from a break signal (start bit length = 2M + 3 bits or more). SAC has an 8-bit input register than can input data from the G-bus. SAC converts data from the LSB of SACR. Figure 23 shows a diagram of the SAC stop bit insertion.

Figure 23. SAC Stop Bit Insertion



UART. The universal asynchronous receiver transmitter (UART) consists of a URTI serial input and URTO output. URTI extracts a character from the start-stop data, deletes the start, stop, and parity bits, and inputs only data to the 8-bit URI register. URTI also performs parity checking if specified. URTO adds the start, stop, and parity bits to URO data and outputs it serially.

The UART mode register (URTMR) is an 8-bit register. The UART functions are shown in Figure 24. ASC and SAC are independent of the UART.

The UART status register (URTSR) is a 4-bit register. All the UART bits are cleared when its status is output to the G-bus. The URTSR functions are shown in Figure 25.

Figure 24. URTMR Functions

URTMR	
Bit 7	URTI Serial Input
0	T_XD pin
1	SAC output

URTMR	
Bit 5	URTO Control
0	Serial output disable
1	Serial output enable

URTMR	
Bit 6	Break Signal
0	Not sent
1	Sent (continuously)

URTMR		
Bit 3	Bit 2	Parity
0	–	No check/generate
1	0	Odd
1	1	Even

URTMR	
Bit 4	URTI Control
0	Serial input disable
1	Serial input enable

URTMR	
Bit1	Data Length (1)
0	7
1	8

URTMR	
Bit 0	Stop Bit Length
0	1
1	2

Note:

(1) The data length does not include the start, stop, and parity bits.

Figure 25. URTSR Functions

URTSR	
Bit 3	Parity Error
0	No parity error
1	Parity error

URTSR	
Bit 2	Framing Error
0	No framing error
1	Framing error

URTSR	
Bit 1	Overrun Error
0	No overrun error
1	Overrun error

URTSR	
Bit0	Break Signal
0	No break signal
1	Break signal (2)

Notes:

(1) If data is input to the URTI serially or a break signal is detected, an interrupt request (IIU) is issued. URTSR data must be checked every time an IIU is issued. If data is output to the URTO serially, an interrupt signal IOU is issued.

(2) The URTSR determines that a break signal is issued when two or more continous characters (excluding the start, stop, and parity bits) are 0.

A/D and D/A Interface

This circuit interfaces the A/D and D/A converters. It consists of a variable-length serial I/O and FIFO, a mode register used to reset the mode, and a DAMR. The A/D serial input signal ADIN inputs DSP ADSI.

The circuit uses the ADST pin to output the A/D conversion start strobe and uses the ADIN pin to input A/D data serially. The circuit also uses the $\overline{\text{ADCK}}$ pin for the A/D conversion serial clock. The circuit inputs data from the ADIN pin in synchronization with the rising edge of $\overline{\text{ADCK}}$. The DALD pin is used to output a D/A conversion load strobe signal. The circuit outputs data from the DAOT pin in synchronization with the $\overline{\text{DACK}}$ D/A conversion serial clock.

Serial data is input or output from the MSB. The data length is selectable between 8 or 16 bits. Table 44 lists the A/D and D/A SFR addresses

Table 44. A/D and D/A SFR Addresses

Unit	SFR Address	Description
DAMR	FF4EH (low-order 6 bits)	A/D and D/A mode register
FFRA	FF4FH (high-order 3 bits)	FIFO read address
FFWA	FF4FH (low-order 3 bits)	FIFO write address
FIFO	FF54, FF55H	FIFO

D/A Mode Register (DAMR). The DAMR is a 6-bit register. It controls the FIFO read address and selects the A/D and D/A previous bit length and sampling cycle. Its SFR address is FF4EH which corresponds to the low-order six bits of the G-bus. DAMR changes the width of the serial enable signal ADST or DALD, depending on the A/D bit length and the duty of the ADST signal. However, the data width for actual processing is selected by the SR register (SIC bit) of the DSP. DAMR bit 4 and SIC bits must be identical. Figure 26 shows the DAMR functions.

Figure 26. DAMR Functions

DAMR Bit 5	FFRA Control
0	Data is not output from FIFO
1	FFRA changes depending on FIFO read

DAMR Bit 4	A/D Precision
0	16
1	8

DAMR Bit 3	A/D Precision
0	16
1	8

DAMR Bit			A/D, D/A Sampling Frequency
2	1	0	
0	0	0	9.6 kHz
0	0	1	19.2 kHz
0	1	0	38.4 kHz
0	1	1	Inhibited
1	0	0	14.4 kHz
1	0	1	28.8 kHz
1	1	0	7.2 kHz
1	1	1	Inhibited

Notes:

(1) There is a possibility that erroneous data could be read in bits 0, 1, and 2 if the GPP reads DMAR immediately after it is written.

(2) The DAOT pin outputs on F when DAMR bit 5 is 0.

(3) Writes to the FFRA are inhibited when DAMR bit 5 is 1.

FIFO. FIFO is an eight-level stack memory. When data is read from the FIFO and output to an external unit by the DASO, the next data is read. If data is read from the level 4 of the FIFO or all the data is read from the FIFO (write address = read address), an interrupt request from the FIFO is issued. The write address is selected by FFWA and the read address by FFRA. Both FFWA and FFRA are three-bit registers.

The FIFO SFR address is FF54H (low-order eight bits) and FF55H (high-order eight bits). FFRA and FFWA have the same address of FF4FH. FFRA corresponds to bits 6, 5, and 4 of the G-bus and FFWA corresponds to bits 2, 1, and 0.

When the D/A precision is eight bits, data is written into the FIFO by an instruction to write in the low-order eight bits (MOV FIFO, xx). When it is 16 bits, data is written into the FIFO by an instruction to write in the high-order 8 bits (MOV FIFO + 1, xx). When a 16-bit transfer instruction (MOVW FIFO, xx) is executed, data is written in the low-order eight bits and then in the high-order eight bits. When FIFO data (FF54H, FF55H) is read to the G-bus, the data is also immediately read from the G-bus. The operation does not affect FFWA and FFRA. Note that data is stored in a buffer before it is written into FIFO and data in the buffer is read when the G-bus is read. Also, at FIFO levels 2 and 3 immediately after DAMR bit 5 is changed from 0 to 1, a 1 is read from the FIFO.

Refer to timing waveforms for the A/D serial input and D/A serial output timing.

Serial Interface [SI1, SO1, S1SR]

General. The serial input port 1 (SI1) and serial output port 1 (SO1) are 16-bit serial I/O interfaces. The serial interface has an internal status register (S1SR) used to indicate the status of the SI1 and SO1 interfaces.

Both the SI1 and SO1 consist of a four-bit counter, a 16-bit shift register, and a 16-bit register buffer. The S1SR consists of a two-bit register.

Table 45 lists the serial interface SFR addresses.

Table 45. Serial Interface SFR Addresses

Unit	SFR Address	Description
S1SR	FF56H, (2 bits)	Status register
SI1	FF58, FF59H	Serial input port 1
SO1	FF5A, FF5BH	Serial output port 1

Interface Functions. The S1SR indicates the SI1 and SO serial interface status. It consists of two bits, and is set to 0H when the processor is reset.

Table 46 lists the S1SR functions. The SFR address is FF56H.

Table 46. S1SR

S1SR	Value	Functions
Bit 0	1	Data was input to SI1
Bit 1	1	SO1 buffer is full

SI1: SI1 is a 16-bit serial input interface. It is comprised of a 16-bit shift register, an SI1 register (buffer), and a 4-bit counter. The SFR address of the SI1 register is FF58H (low-order eight bits) and FF59H (high-order eight bits). The SFR address is undefined when the processor is reset. SI1 counts 16 bits of serial input data at the rising edge of $\overline{S1CK}$ and inputs them to the shift register when the $\overline{SI1EN}$ pin goes active. After the 16 bits of serial data are input, the register resets the counter with a carry and transfers the contents of the shift register to the SI1 register.

This sets S1SR bit 0 to 1 and issues an SI1 interrupt. A read signal then allows the contents of the SI1 register to be output to the G-bus. At this instant of time, data at FF59H (high-order eight bits) is read and S1SR bit 0 goes to 0. This completes the execution of the serial input.

To read SI1 data, the SFR address FF58H (low-order eight bits) must be read first and then SFR address FF59H (high-order eight bits). When S1SR0 is 0, serial input is disabled, so the same data will be read repeatedly from SI1.

SO1: SO1 is a 16-bit serial output interface. It consists of a 16-bit shift register, an SO1 register (buffer), and a 4-bit counter. The SFR address of the SO1 register is FF5AH (low-order eight bits) and FF5BH (high-order eight bits). The SFR address is undefined when the processor is reset. SO1 writes serial output data from the G-bus (SO1 register) by a Write signal generated from the G-bus interface. When data is written in the high-order eight bits (FF5BH), S1SR bit 1 (SO1 buffer full) is set to 1. SO1 register data is transferred to the shift register when it is not in the output mode and S1SR bit 1 (SO1 buffer full) is set to 0. When data is input to the shift register, SO1 automatically outputs serial output request signal SO1RQ from the SO1RQ pin, using $\overline{S1CK}$ as a serial clock. When the $\overline{SO1EN}$ pin goes active, SO1 outputs 16 bits of serial data from the SO1 pin at the falling edge of the $\overline{S1CK}$ serial clock. SO1 stores output data in the buffer before transferring it to the shift register. It stores the next data in the buffer when the buffer becomes free. The buffer status is checked by the S1SR register and SO1 can output bytes of serial data continuously.

To write serial output data to the SO1 register, the low-order eight bits must be written first and then the high-order eight bits. Data is then transferred to the shift register.

When S1SR bit 1 is a 1, the write operation is disabled. Consequently, in this type of occurrence, the address is rewritten. Figure 27 shows the SI1 timing diagram and Figure 28 shows the SO1 timing diagram.

Figure 27. SI1 Timing Diagram

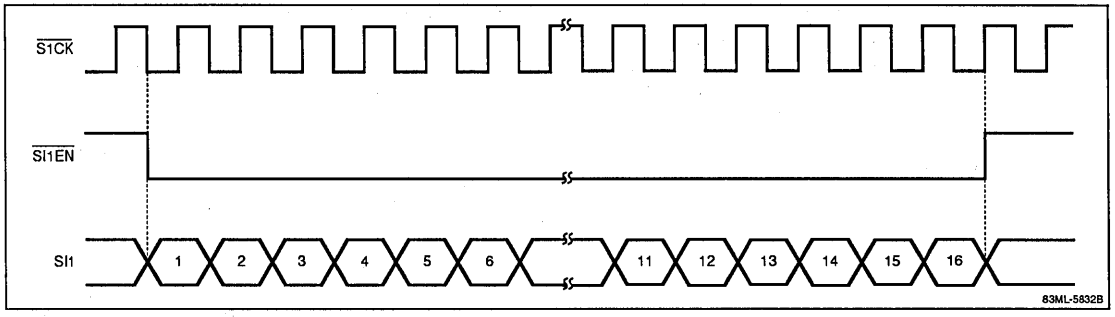
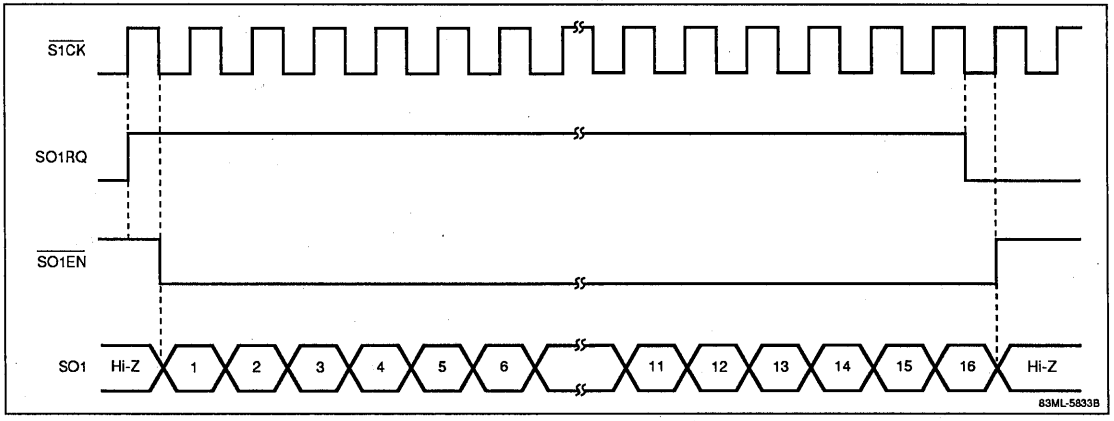


Figure 28. SO1 Timing Diagram



General-Purpose Timer and Watch Dog Timer [TMMR, TMRA, TMRB, WDMSR, WDTMR]

General. TMRA is a general-purpose timer consisting of an 8-bit decrement counter. TMRB is an interval timer consisting of an 8-bit decrement counter. TMMR is a control register for TMRA and TMRB.

WDTMR is an 8-bit watch dog timer that monitors software hangup. If the specified time is expired, it issues a non-maskable interrupt (MNIWD) to GPP. WDMSR is a register used to specify the mode of WDTMR. Table 47 lists the timer SFR addresses.

Table 47. Timer SFR Addresses

Unit	SFR Address	Description
TMMR	FF5DH	General-purpose timer control register
TMRA	FF5EH	8-bit general-purpose timer

Functions. General-Purpose Timer Control Register (TMMR): TMMR is a 5-bit register used to control the TMRA general-purpose timer and the TMRB interval timer. TMMR bit 0 specifies the TMRA operation; bits 4 through 6 specify the TMRB interval clock; and bit 7 specifies the TMRB initialization.

When TMMR bit 0 is changed from 0 to 1, TMMR loads the data stored in the buffer into TMRA and decrements it at the rising edge of the timer clock (230.4 kHz). When the counter value reaches 0, TMMR sets bit 0 to 0 and issues an interrupt signal to stop the counter.

When bit 7 is changed from 0 to 1, TMMR clears TMRB and increments the counter at the rising edge of the timer clock (921.6 kHz, 460.8 kHz, 230.4 kHz, or 115.2 kHz). If the counter overflows, TMMR issues an interrupt signal. Bit 7 is cleared to 0 at the same time the timer starts operation. The TMMR initial value and reset value is 00H.

The TMMR SFR address is FF5DH. In TMMR read mode, a 0 is output to G-bus unassigned bits 3 through 7. Table 48 shows the TMMR functions.

Table 48. TMMR Functions

TMMR	Name	Contents				Initial Value
Bit 7	TBI	When bit 7 is 1, TMRB is initialized				0
Bits 6-4	TBS	TMRB interval timer clock selection				000
		Bits				
		6	5	4	Clock Frequency	
		0	0	0	921.6 kHz	
		0	1	1	460.8 kHz	
		1	0	1	230.4 kHz	
		1	1	1	115.2 kHz	
Bits 3-1	—	Not used (0 is output if read)				—
Bit 0	TAE	When bit 0 is 1, TMRA is enabled				0

General-Purpose Timer (TMRA): TMRA consists of a buffer register and a counter. The TMRA SFR address is FF5EH. Buffer register TMRA is set to FFH when reset, however other values can be written to the buffer. A value of 0 may cause the TMRA to malfunction.

When TMRA is enabled by the TAE = 1, data from the buffer register is loaded to the counter, which decrements at a 230.4 kHz (4.34 μs) frequency rate generated by T_XPLL. When the counter is decremented to 0, TMRA issues a timer interrupt signal IAT, sets the TAE bit to 0, and stops the counter. When TMRA is read by the GPP, the counter value is output if the counter is in the operation mode. If the counter is not in the operation mode, the buffer register value is output.

Interval Timer (TMRB): TMRB consists of an interval timer clock selector and a counter. The counter is reset to 00H. When TMMR bit 7 (initialization signal TBI) is 1, TMRB clears the counter and decrements it at the frequency selected by the TBS (interval timer clock selection bit) of TMMR bits 6 through 4. Four interval times are available: 0.28 ms, 0.55 ms, 1.1 ms, and 2.2 ms.

If the counter overflows, TMRB issues a timer interrupt signal IBT. The TMRB counter value cannot be read by the GPP.

Watch Dog Timer Control Register (WDMSR): The WDMSR is an 8-bit register used to control the watch dog timer (WDTMR). Its SFR address is FF6DH. It is set to 00H when reset and the watch dog timer stops. WDMSR bit 0 and WDMSR bit 1 specify the WDTMR interval time (ITV0 and ITV1). WDMSR bit 7 enables the WDTMR and WDMSR bit 2 through WDMSR bit 6 (five bits) are not defined, but when read 0 is output to the G-bus.

Table 49 shows the WDMSR SFR address. Figure 29 shows the WDMSR functions.

Table 49. WDMSR SFR Address

Unit	SFR Address	Description
WDMSR	FF6DH	Watch dog timer control register

Watch Dog Timer Counter Register (WDTMR): The WDTMR monitors software hangup. If the time, specified by WDMSR expires, WDTMR issues a non-maskable interrupt signal (MNIWD). WDTMR consists of an 8-bit increment counter and a decoder. WDTMR is set to 00H when initialized or reset. WDTMR is enabled by WDR = 1 (operation enable signal) of WDMSR, and starts incrementing at the clock rate of 75 Hz. The decoder decodes a carry from bits 6 through 8 of the counter. The internal signal ITV bit 0/ ITV bit 1 which is output from WDMSR, selects the interval time, and issues a non-maskable signal (NMIWD). Next, the increment counter is reset by NMIWD and starts incrementing again at 75 Hz. The watch dog timer has no address and cannot be read and written.

WDTMR is reset and starts counting every time data is written into the WDMSR. Figure 30 shows the 8-bit increment counter and decoder.

Figure 29. WDMSR Functions

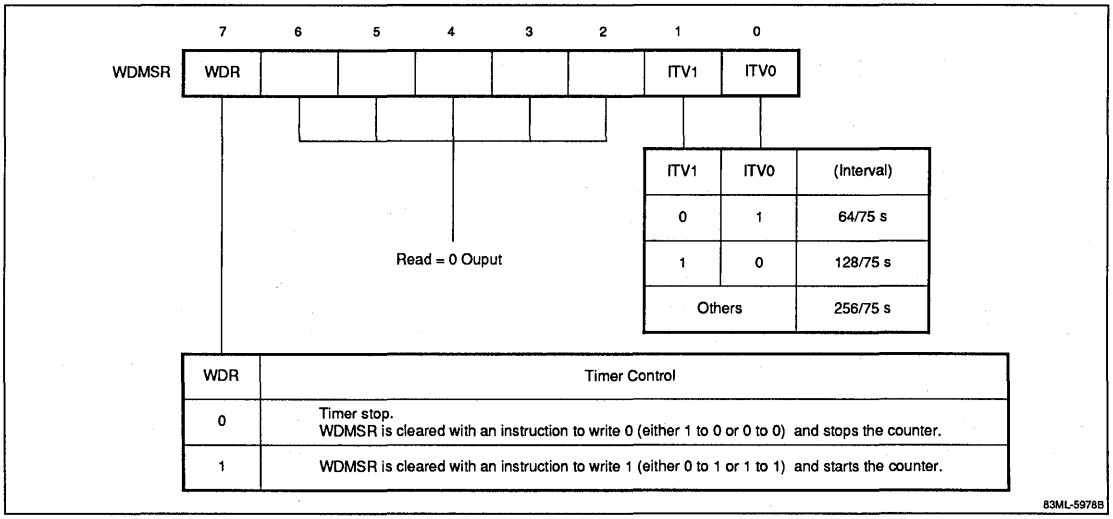
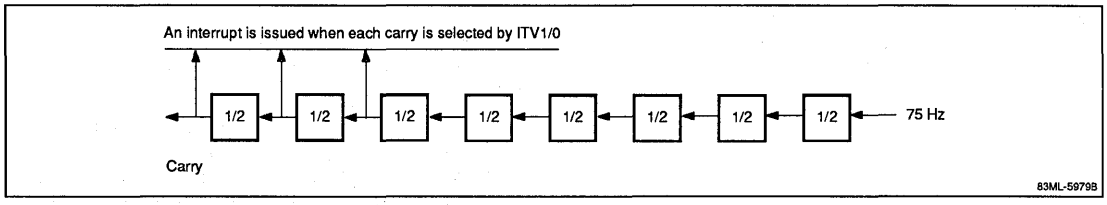


Figure 30. WDTMR 8-Bit Increment Counter and Decoder



DSP Interface

General. The DSP interface consists of an INTDSP register that issues an interrupt and reset to the DSP, a data register (DR) that inputs and outputs data to and from the DSP, and a status register (SR). The INTDSP register, DR, and SR are all mapped in memory as SFR of the GPP.

DSP Functions

DSP Reset and Interrupt: The INTDSP register issues reset and interrupt requests to the DSP. The INTDSP register is a 2-bit register, which is set to 00H when initialized or reset. Its address is SFR FF64H, which corresponds to the low-order 2 bits of the G-bus. 0 is output from the G-bus bits 2 through 7 when the interface is read.

Table 50 shows the INTDSP SFR address and table 51 shows the INTDSP functions.

Table 50. INTDSP SFR Address

Unit	SFR Address	Description
INTDSP	FF64H	DSP reset and interrupt request register

Table 51. INTDSP Functions

INTDSP	Function
Bit 0	When this bit = 1, INTDSP resets DSP
Bit 1	When this bit is changed from 0 to 1, INTDSP issues an interrupt request to DSP. After the interrupt request is issued, the bit is automatically reset.

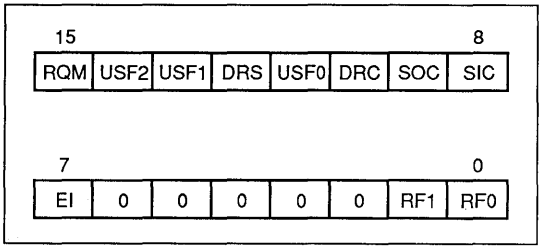
Data Input/Output Between the GPP and DSP: The SR register stores the DSP status. The SR register consists of an 11-bit status register. Internally, it is handled as a 16-bit register. The high-order eight bits can be read by the GPP by specifying the SFR address FF62H or FF63H.

The SR register is set to 00H when the processor is reset. Table 52 shows the SR register SFR address and Figure 31 shows the status register configuration. See DSP Status Register (SR) for functional details.

Table 52. SR Register SFR Address

Unit	SFR Address	Description
SR	FF62H or FF63H	DSP SR register

Figure 31. Status Register Configuration



The DR register is a 16-bit register. It can be used as a data transfer register to and from the DSP. Since the GPP is eight bits, DR transfers 16-bit data in two operations. Internally, 16-bit data is transferred in one operation. For 16-bit transfer, DR first transfers the low-order eight bits then the high-order eight bits. When the DR register is defined as an 8-bit register by the DRC bit of the status register (SR), only the low-order eight bits of DR are transferred. The high-order eight bits are not defined (or their value is the one previous to being changed). The DR register can be read and written by the GPP by specifying the SFR address FF60H or FF61H. Table 53 shows the DR register SFR address. See DSP Data Register (DR) for the functional details.

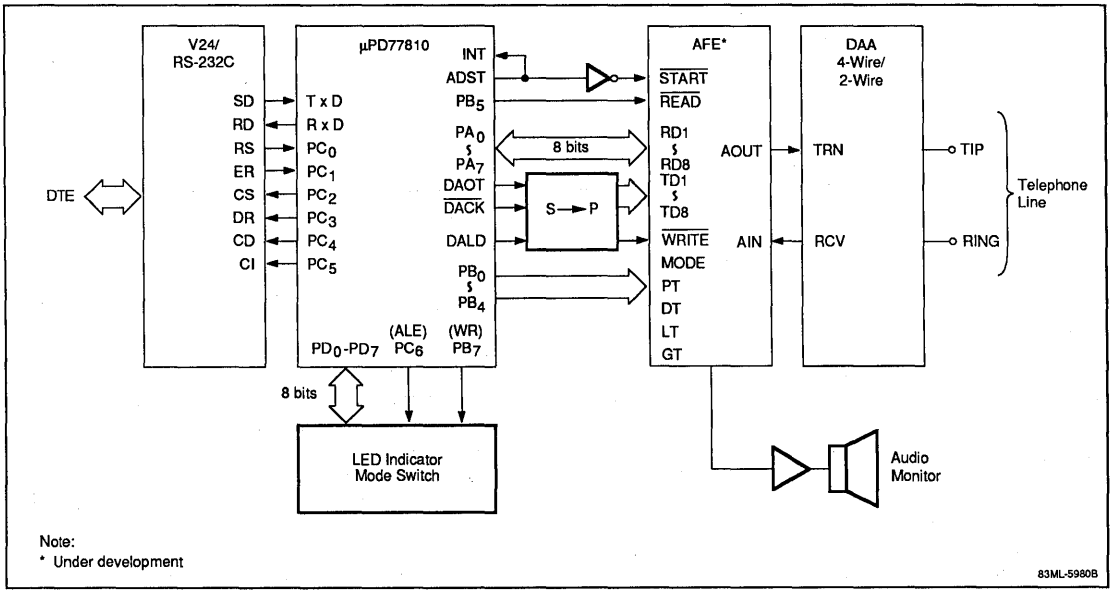
Table 53. DR Register SFR Address

Unit	SFR Address	Description
DR	FF60H or FF61H	DSP DR register

SYSTEM CONFIGURATION

Figure 32 shows a typical V.22bis system application for the μPD77810.

Figure 32. V.22bis System Application Example



μPD7281
IMAGE PIPELINED PROCESSOR

Description

The NEC μPD7281 Image Pipelined Processor is a high-speed digital signal processor specifically designed for digital image processing such as restoration, enhancement, compression, and pattern recognition. The μPD7281 employs token-based data-flow and pipelined architecture to achieve a very high throughput rate. A high-speed on-chip multiplier speeds calculations. More than one μPD7281 can easily be cascaded with a minimum amount of interface hardware to increase the throughput rate even further. The μPD7281 is designed to be used as a peripheral processor for minicomputers or microcomputers, thereby relieving the host processor from the burden of time-intensive computations. The μPD7281 has a very powerful instruction set designed specifically for digital image processing algorithms. The Image Pipelined Processor can also be used as either a general purpose digital signal processor or a numeric processor.

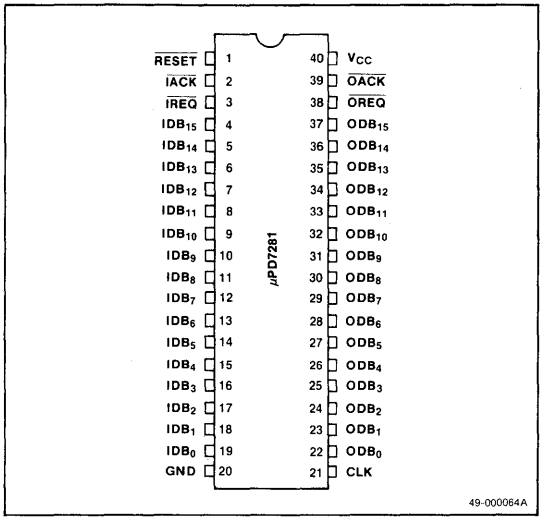
Features

- ☐ Token-based data-flow architecture
- ☐ Internal pipelined ring architecture
- ☐ Powerful instruction set for image processing
- ☐ 17 x 17-bit (including sign bits) fast multiplier: 200 ns
- ☐ High-speed data I/O handling
  - — Asynchronous two-wire handshaking protocols
  - — Separate data input and output pins
- ☐ Easy multiple-processor configuration
- ☐ Rewritable program stores
- ☐ On-chip memories:
  - — Link Table (LT): 128 x 16 bits
  - — Function Table (FT): 64 x 40 bits
  - — Data Memory (DM): 512 x 18 bits
  - — Data Queue (DQ): 32 x 60 bits
  - — Generator Queue (GQ): 16 x 60 bits
  - — Output Queue (OQ): 8 x 32 bits
- ☐ NMOS technology
- ☐ Single +5 V power supply
- ☐ 40-pin DIP

Applications

- ☐ Digital image restoration
- ☐ Digital image enhancement
- ☐ Pattern recognition
- ☐ Digital image data compression
- ☐ Radar and sonar processing
- ☐ Fast Fourier Transforms (FFT)
- ☐ Digital filtering
- ☐ Speech processing
- ☐ Numeric processing

Pin Configuration



Performance Benchmarks

Operation	1 μPD7281	3 μPD7281s	Note
Rotation	1.5 sec	0.6 sec	512 x 512 binary image
1/2 Shrinking	80 ms	30 ms	512 x 512 binary image
Smoothing	1.1 sec	0.4 sec	512 x 512 binary image
3x3 Convolution	3.0 sec	1.1 sec	512 x 512 grey scale image
64-stage FIR Filter	50 μs	18 μs	17-bit fixed point
cos(x)	40 μs	15 μs	33-bit fixed point

Ordering Information

Part Number	Package Type
μPD7281D	40-pin ceramic DIP

Pin Identification

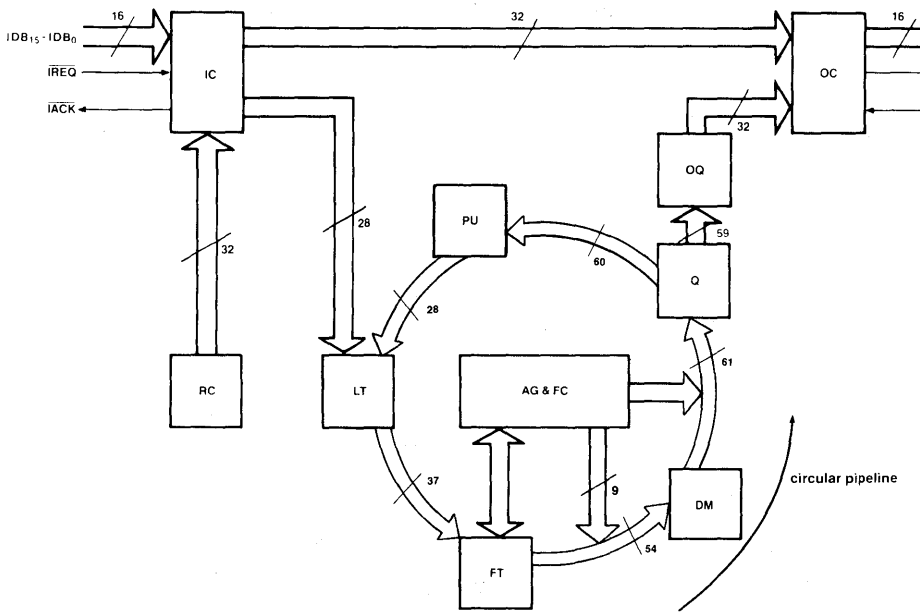
No.	Signal	I/O	At RESET	Description
1	$\overline{RESET}$	In		System Reset: A low signal on this pin initializes μPD7281. During the reset, a 4-bit module number should be placed on IDB_{15} - IDB_{12}.
2	$\overline{IACK}$	Out	High	Input Acknowledge: This acknowledge signal is output by the μPD7281 to notify the external data source that a 16-bit data transfer has been completed.
3	$\overline{IREQ}$	In		Input Request: This input signal requests a data transfer from an external device to μPD7281.
4-19	IDB_{15} - IDB_0	In		16-bit input data bus: 32-bit input data tokens are input to the Input Controller as two 16-bit words.
20	GND			Power ground
21	CLK	In		System clock input (10 MHz: target spec)
22-37	ODB_{15} - ODB_0	Out	High Impedance	16-bit output data bus: 32-bit output data tokens are output by the Output Controller as two 16-bit words.
38	$\overline{OREQ}$	Out	High	Output Request: This signal informs an external device that a 16-bit data word is ready to be transferred out of μPD7281.
39	$\overline{OACK}$	In		Output Acknowledge: This acknowledge signal input by the external data destination notifies μPD7281 that a 16-bit data transfer may occur.
40	V_{CC}			+5 V power supply

Architecture

The μPD7281 utilizes a token-based, data-flow architecture. This novel architecture not only provides multiprocessing capability without complex external hardware, but also offers high computational efficiency within each processor. Taking advantage of the multiprocessing capability of data-flow architecture, almost any processing speed requirements can be satisfied by using as many μPD7281s as needed in the system. Within each μPD7281, the data-flow architecture provides high computational efficiency through concurrent operations. For example, while the Processing Unit (or ALU) spends its time for actual computations only, the internal memory address calculations, internal memory read and write operations and input/output operations are all being done concurrently. Furthermore, in contrast to conventional von Neumann processors, a data-flow processor doesn't fetch instructions, perform subroutine stack operations or do data transfers between registers. Therefore, it does not spend the time required for these operations.

The μPD7281 also utilizes an internally pipelined architecture. As shown in the block diagram, a circular pipeline is formed by five functional blocks: the Link Table (LT), the Function Table (FT), the Data Memory (DM), the Queue (Q), and the Processing Unit (PU). A token entered through the Input Controller (IC) is passed on to the Link Table to be processed around the pipelined ring as many times as needed. When a token is finished being processed, it is queued into Output Queue (OQ) and then output via the Output Controller (OC).

Block Diagram



IC: Input Controller. Controls input data tokens and determines whether or not an input data token should be sent to the circular pipeline for processing.

OC: Output Controller. Controls output data tokens.

LT: Link Table (128 words x 16 bits). Stores instruction parameters.

FT: Function Table (64 words x 40 bits). Stores instruction parameters.

DM: Data Memory (512 words x 18 bits). Stores constants or temporary data.

Q: Queue (48 words x 60 bits). FIFO queue. Data Queue: 32 words x 60 bits. Generator Queue: 16 words x 60 bits.

PU: Processing Unit. Executes logical, arithmetic and bit operations.

OQ: Output Queue (8 words x 32 bits). FIFO queue for the output tokens.

AG/FC: Address Generator and Flow Controller. Generates addresses for DM and controls the flow of tokens.

RC: Refresh Controller. Generates refresh tokens for internal DRAMs.

49-000132B

Absolute Maximum Ratings

$T_A = +25°C$

Supply voltage, V_{DD}	−0.5 V to +7.0 V
Input voltage, V_I	−0.5 V to +7.0 V
Output voltage, V_O	−0.5 V to +7.0 V
Operating temperature, T_{OPT1} (2 m/s air flow)	0°C to +70°C
Operating temperature, T_{OPT2} (No air flow)	0°C to +45°C
Storage temperature, T_{STG}	−65°C to +150°C

Comment: Exposing the device to stresses above those listed in Absolute Maximum Ratings could cause permanent damage. The device is not meant to be operated under conditions outside the limits described in the operational sections of this specification. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Capacitance

$T_A = +25°C$

Parameter	Symbol	Limits Min	Limits Max	Unit	Test Conditions
CLK capacitance	C_K		20	pF	fc = 1 MHz
Input capacitance	C_I		10	pF	(All other pins
Output capacitance	C_O		20	pF	at 0 V)

DC Characteristics

$T_A = 0°C$ to $+70°C$, $V_{DD} = 5\ V \pm 10\%$

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
Input low voltage 1 ($\overline{RESET}$, IDB_{15-0})	V_{IL1}	−0.5		0.7	V	
Input high voltage 1 ($\overline{RESET}$, IDB_{15-0})	V_{IH1}	2.0		$V_{DD}+0.5$	V	
Input low voltage 2 ($\overline{IREQ}$, $\overline{OACK}$, CLK)	V_{IL2}	−0.5		0.45	V	
Input high voltage 2 ($\overline{IREQ}$, $\overline{OACK}$, CLK)	V_{IH2}	3.5		$V_{DD}+0.5$	V	
Output low voltage	V_{OL}			0.45	V	$I_{OL} = 2.0\ mA$
Output high voltage	V_{OH}	2.4			V	$I_{OH} = -400\ \mu A$
Input leakage current	I_{LI}			±10	μA	$0\ V \le V_I \le V_{DD}$
Output leakage current	I_{LO}		±10		μA	$0\ V \le V_O \le V_{DD}$
Supply current	I_{DD}		320	500	mA	

AC Characteristics

$T_A = 0°C$ to $+70°C$, $V_{DD} = 5\ V \pm 10\%$

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
CLK cycle time	t_{CLK}	100		500	ns	Measured at 2 V
CLK pulse width high	t_{WKH}	40			ns	
CLK pulse width low	t_{WKL}	40			ns	
CLK rise time	t_{KR}			10	ns	
CLK fall time	t_{KF}			10	ns	
$\overline{IACK}$ delay time 1 (from $\overline{IREQ}$ down) (Note 1)	t_{DIAL1}	20		50	ns	
$\overline{IACK}$ delay time 1 (from $\overline{IREQ}$ up) (Note 2)	t_{DIAH1}	20		55	ns	
$\overline{IACK}$ delay time 2 (from $\overline{IREQ}$ down)	t_{DIAL2}	20		70	ns	
$\overline{IACK}$ delay time 2 (from $\overline{IREQ}$ up)	t_{DIAH2}	20		70	ns	
Min time between transitions on $\overline{IREQ}$ and $\overline{IACK}$	t_{HIQ}	15			ns	
$\overline{IREQ}$ rise time	t_{IQR}			10	ns	

AC Characteristics (cont)

$T_A = 0°C$ to $+70°C$, $V_{DD} = 5\ V \pm 10\%$

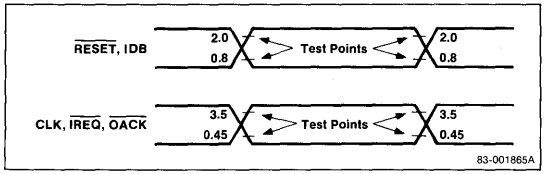
Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
$\overline{IREQ}$ fall time	t_{IQF}			10	ns	
Data set up time (before $\overline{IREQ}$ up)	t_{SID}	40			ns	
Data hold time (after $\overline{IREQ}$ up)	t_{HID}	0			ns	
$\overline{OREQ}$ delay time 1 (from $\overline{OACK}$ down)	t_{DOQH}	15		35	ns	
$\overline{OREQ}$ delay time 1 (from $\overline{OACK}$ up)	t_{DOQL}	15		45	ns	
Min time between transitions on $\overline{OREQ}$ and $\overline{OACK}$	t_{DOA}	15			ns	
$\overline{OACK}$ rise time	t_{OAR}			10	ns	
$\overline{OACK}$ fall time	t_{OAF}			10	ns	
Data access time (after $\overline{OREQ}$ down)	t_{DOD}			25	ns	
Data float time (after $\overline{OREQ}$ up)	t_{FOD}	10		35	ns	
Pre $\overline{RESET}$ high time	t_{RVRST}	t_{CLK}			ns	
$\overline{RESET}$ low time	t_{WRST}	$6t_{CLK}$			ns	
Module number data setup time (after $\overline{RESET}$ down)	t_{DMD}			$2t_{CLK}$	ns	
Module number data hold time (after $\overline{RESET}$ up)	t_{HMD}	0			ns	
Reset delay from CLK down	t_{DRST}			$(1/2)t_{CLK}$	ns	

Notes:

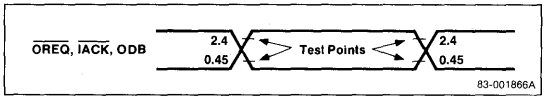
(1) "Down" = on falling edge
(2) "Up" = on rising edge
(3) Output load capacitance: $\overline{IACK}$, $\overline{OREQ}$ = 50 pF; ODB_{15-0} = 100 pF

Timing Waveforms

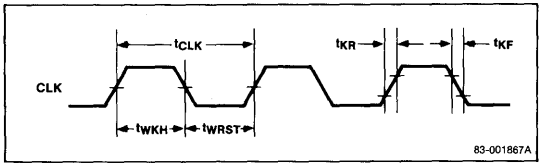
AC Test Input Voltage



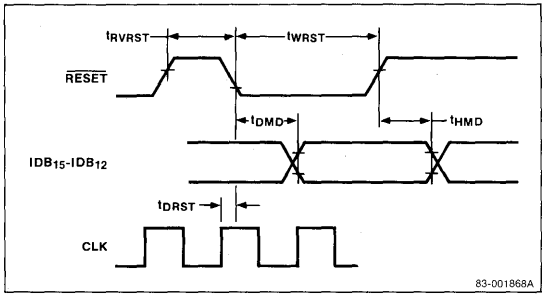
AC Test Output Voltage



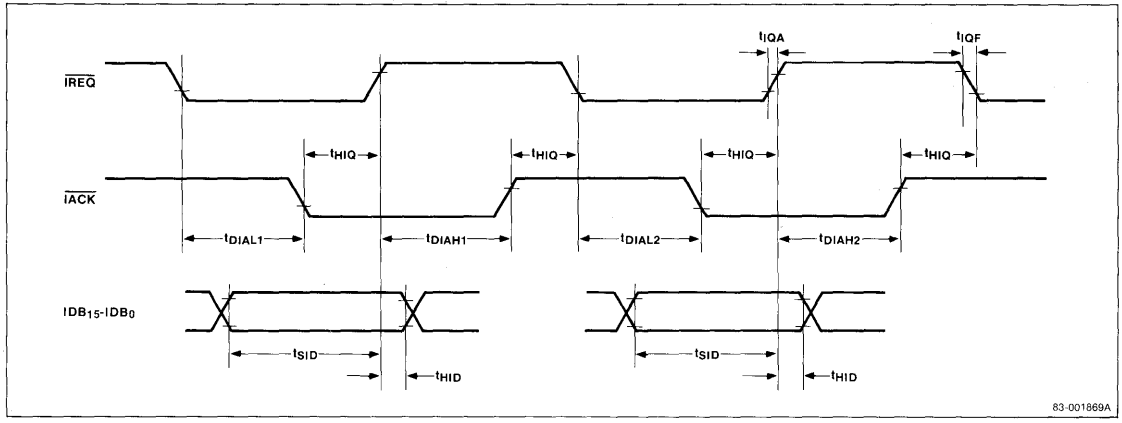
Clock Timing



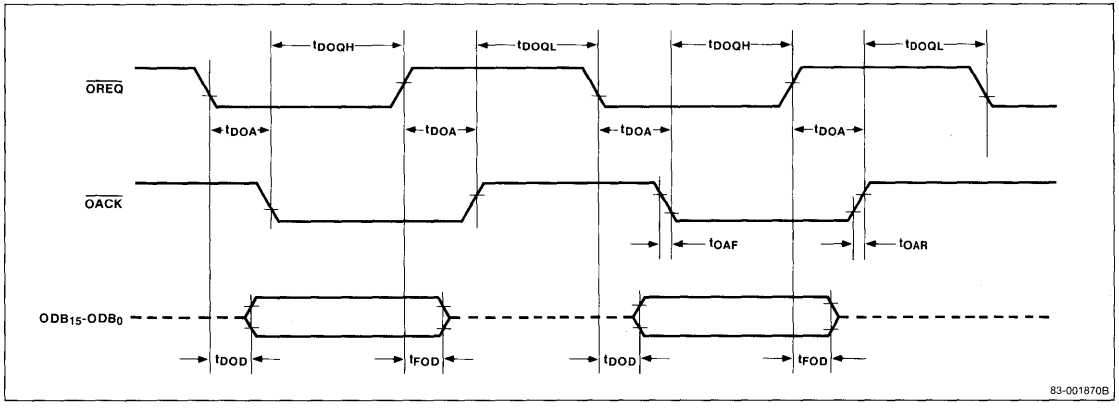
Module Number and RESET Timing



Input Handshake Timing



Output Handshake Timing



Functional Description

As shown in the block diagram, the μPD7281 consists of 10 functional blocks. Before any processing occurs, the host processor down-loads the object code into the Link Table and the Function Table of the μPD7281 by using specially formatted input tokens. At this time, constants may also be sent to the Data Memory to be stored. The contents of the Link Table and the Function Table are closely related to a computational graph. When a computational process is represented graphically, it usually forms a directed data-flow graph. In such a graph, the arcs (or edges, links, etc.) represent the entries in the Link Table and the nodes represent the entries in the Function Table. An arc between any two nodes has a data value, called a "token", and is identified by a corresponding entry in the Link Table. A node in the directed data-flow graph signifies an operation, and the type of operation is logged into the Function Table along with the identification information about the outgoing arc.

A minimal amount of interface hardware is required to configure μPD7281s in a multiprocessor system. As many as 14 μPD7281s can be cascaded together, as shown in figure 1. Each μPD7281 must be assigned a Module Number (MN) during reset. Figure 2 shows the timing diagram for assigning the module number.

When any token enters a μPD7281, regardless of the total number of μPD7281s used in the system, the Input Controller of that μPD7281 discerns whether or not the entering token is to be processed by checking the Module Number (MN) field of the token. If the Module Number is not the same as the Module Number assigned during reset, the token is passed to the Output Controller so that it can be sent out via the Output Data Bus. However, if the token has the same Module Number, then the Input Controller strips off the MN field and sends the remaining part of the token to the Link Table for processing.

Once a token enters the circular pipeline by accessing the Link Table, it requires seven pipeline clock cycles for the token to fully circulate around the ring. One pipeline clock cycle is needed for the Link Table, the Function Table, or the Data Memory to process an incoming token, and two pipeline clock cycles are needed for the Queue or the Processing Unit to process a token. The Queue requires one pipeline

Figure 1. Connecting Multiple μPD7281s

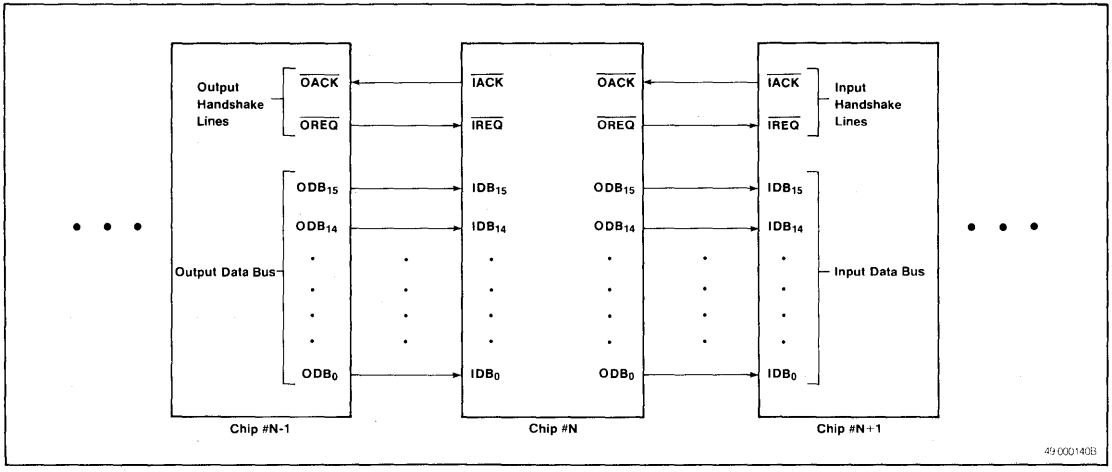
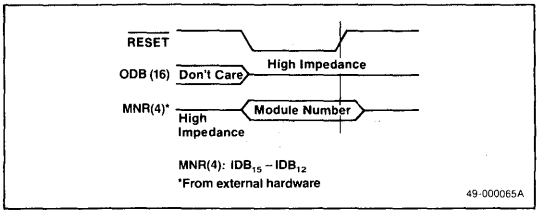


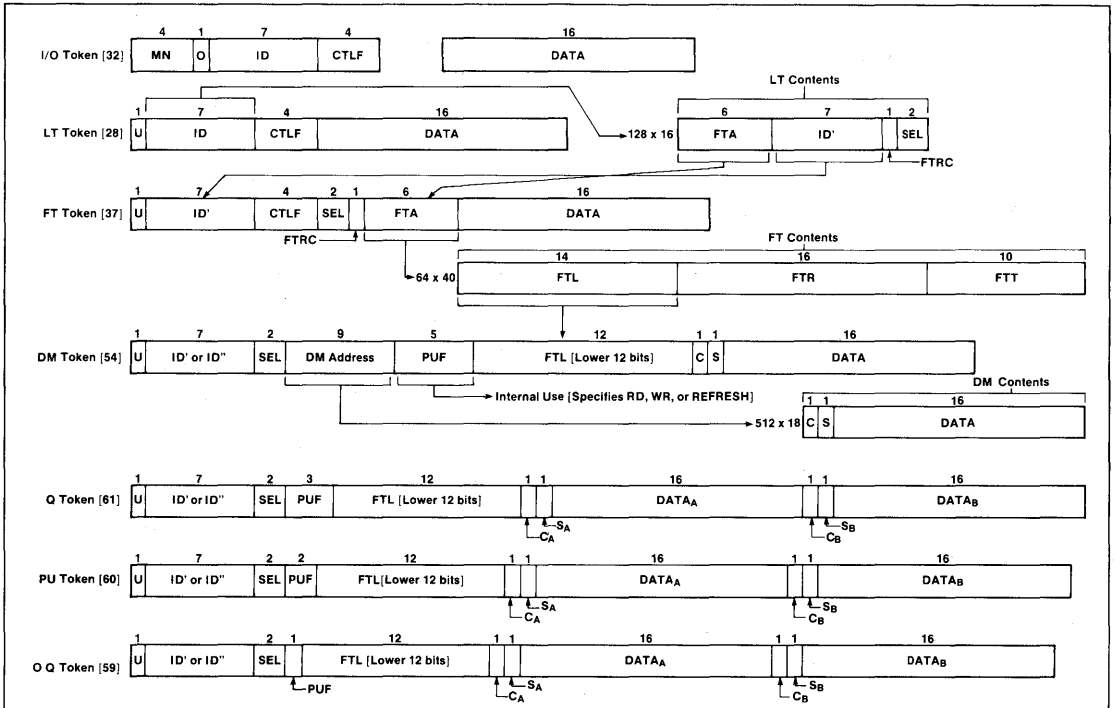
Figure 2. Timing Diagram for Assigning Module Numbers During RESET



clock cycle to write and one cycle to read. Similarly, the Processing Unit requires one pipeline clock cycle to execute and one clock cycle to output the result. In other words, both the Processing Unit and the Queue are made of two-stage pipelines. Therefore, when seven tokens exist simultaneously in the circular pipeline, the pipeline is full and full parallel processing is achieved.

When a data token flows through each functional block in a given μPD7281, the format of the token changes significantly. The actual transitions of a token format through different functional blocks are shown in figure 3. A data token flowing within the circular pipeline must have at least a 7-bit Identifier (ID) field and an 18-bit data field. The ID field is used as an address to access the Link Table memory. When a token accesses the LT memory, the ID field of the token is replaced by a new ID (shown as ID' in figure 3) previously stored in the LT memory. As a result, every time a data token accesses LT memory, its ID field is renewed. The data field of a token consists of a control bit, a sign bit and a 16-bit data. A token may have up to two data fields, as well as other fields (OP code, control, etc.) if necessary.

Figure 3. Token Formats and Transitions



I/O Token	Input or Output Token
LT Token	A token accessing the Link Table
FT Token	A token accessing the Function Table
DM Token	A token accessing the Data Memory
Q Token	A token accessing the Queue
PU Token	A token accessing the Processing Unit
OQ Token	A token accessing the Output Queue

MN	Module Number
ID	Identifier for an input or output token
ID'	Identifier for a token which exits LT for the first time
ID"	Identifier for a token which exits LT for the second time
CTLF	Control field
DATA	16-bit data field
FTA	Function Table Address

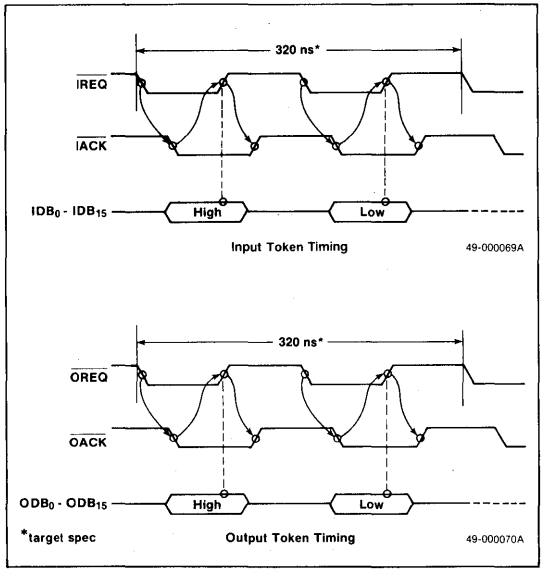
SEL	SELection field
FTRC	Function Table Right field Control bit
FTL	Function Table Left field
FTR	Function Table Right field
FTT	Function Table Temporary field
PUF	Processing Unit Flags
C	Control bit for a 16-bit data
C_A	Control bit for a 16-bit data from A side
C_B	Control bit for a 16-bit data from B side
S	Sign-bit for a 16-bit data
S_A	Sign-bit for a 16-bit data from A side
S_B	Sign-bit for a 16-bit data from B side
$DATA_A$	16-bit data from A side
$DATA_B$	16-bit data from B side

49-000133B

Input Controller [IC]

A 32-bit token is entered into a μPD7281 in two 16-bit halves using a two-signal request/acknowledge handshake method, as shown in figure 4. The input/output token format is shown in figure 7. After a token is accepted by the IC, the MN field of the token is compared to the Module Number of μPD7281 which was assigned at reset. If the Module Number of the accepted token is not the same, the IC passes the token directly to the Output Controller. If the MN field of the accepted token is the same, then the IC strips off the Module Number and sends the remaining part of the token to the Link Table. The IC also monitors the status of the Processing Unit. If it is busy, the IC delays accepting another token until it is no longer busy. The IC also accepts the refresh tokens from the Refresh Controller (RC) and sends them to the Link Table.

Figure 4. Handshake Timing Waveforms



Output Controller [OC]

The OC outputs 32-bit tokens in two 16-bit halves using a two-signal request/acknowledge handshake method, as shown in figure 4. The types of tokens output by the OC are as follows: output data tokens from the Output Queue, error status data tokens generated internally by OC, DUMP tokens, and passing data tokens from the Input Controller.

Link Table [LT]

The LT is a 128 x 16-bit dynamic RAM. The ID field of an incoming LT token is used to access the LT memory. The contents of an LT memory location consist of a 6-bit Function Table Address (FTA), a 7-bit ID, a 1-bit Function Table Right Field Control (FTRC), and a 2-bit Selection (SEL) field. When a token accesses LT memory, its ID field is replaced by the new ID field contained in the memory location being accessed. Therefore, every time a token accesses LT memory, it is given a new ID. The FTA field is used to access FT memory locations. The FTRC bit and the SEL field are used to specify the type of instruction. By using specially formatted tokens, the contents of the LT can either be set during a program download or be read during a diagnosis.

Function Table [FT]

The FT is a 64 x 40-bit dynamic RAM. As for the case of the Link Table, the contents can either be set during a program download or be read during a diagnosis by using specially formatted tokens.

Each FT memory location consists of a 14-bit Function Table Left field (FTL), a 16-bit Function Table Right field (FTR), and a 10-bit Function Table Temporary field (FTT). These fields contain control information for different types of instructions.

Address Generator and Flow Controller [AG/FC]

The AG/FC generates the addresses to access the Data Memory (DM) and controls the writing of data to and the reading of data from the Data Memory. AG/FC determines whether the incoming token contains a one-operand instruction or a two-operand instruction. One-operand instruction tokens can be sent directly to the Queue. However, if the token contains a two-operand instruction, then both operands must be available before they can be sent to the Queue. For a two-operand instruction, the token which arrives at the Data Memory first is temporarily stored until the second operand token arrives. When the second operand token exits the Function Table, the AG/FC generates the Data Memory address which contains the first operand. Then, the second operand token and the first operand data read out from the Data Memory are sent to the Queue together.

Data Memory [DM]

The DM is a 512 x 18-bit dynamic RAM which is used to queue the first operand for a two-operand instruction until the second operand arrives. DM can also be used as a temporary memory or as a buffer memory for I/O data.

Queue [Q]

The Q is a FIFO memory configured with a 48 x 60-bit dynamic RAM. The Q is used to temporarily store the Processing Unit-bound and the Output Queue-bound tokens. The Q is further divided into two different FIFO memories: a 32 x 60-bit Data Queue (DQ) and a 16 x 60-bit Generator Queue (GQ). The DQ is used for the

PU, OUT and AG/FC instructions. The DQ temporarily stores the PU and AG/FC tokens before they are sent to the Processing Unit for processing. The DQ also temporarily stores the Output Queue tokens before they are sent to the Output Queue. The GQ is used for Generate (GE) instructions only. The DQ will not output tokens to the Output Queue if it is full, and the DQ or GQ will not output tokens to the Processing Unit if the Processing Unit is busy.

In order to control the number of tokens in the circular pipeline to prevent Q overflow, the Q is further restricted by the following two situation rules: when the DQ has eight or more tokens stored, the read from the GQ is inhibited, and when the DQ has fewer than eight tokens stored, the read from the GQ has a higher priority than the read from the DQ. Since instructions stored in the GQ generate tokens, restricting the number of GQ tokens is important in order to keep the Q from overflowing. In case the internal processing speed is slower than the rate of incoming data tokens, the DQ posseses a potential overflow condition. To prevent overflow, the processor is put into restrict/inhibit mode when the DQ reaches a level greater than 23.

Output Queue [OQ]

The OQ is a first-in first-out (FIFO) memory configured in an 8 x 32-bit static RAM. The OQ is used to temporarily store the output data tokens from the Data Queue so that they can be output by the Output Controller via the output data bus. When OQ is full, it sends a signal to the Data Queue to delay accepting further tokens.

Processing Unit [PU]

The PU executes two types of instructions: PU and GE. PU instructions include logical, arithmetic (add, subtract and multiply), barrel-shift, compare, data-exchange, bit-manipulation, bit-checking, data-conversion, double-precision adjust, and other operations. The control information for a PU instruction is contained in the Function Table Left field of the PU token. The GE instructions are used to generate a new token, multiple copies of a token, or block copies of a token. They can also be used to set the Control field (CTLF) of a token and to generate external memory addresses. If the current PU operation cannot be completed within a pipeline clock cycle, the PU sends a signal to the Queue and the Input Controller to prevent them from releasing any more tokens.

Refresh Controller [RC]

The RC automatically generates refresh tokens for the dynamic RAMs used in the circular pipeline, i.e. the LT, FT, DM, and Q. Each RC token, generated periodically, is sent to the Input Controller and is propagated through the LT, FT, DM and Q, in that order. The RC tokens are deleted after reaching the Q.

Operation Modes

There are three different modes in which the μPD7281 can operate: Normal, Test, and Break (see figure 5). After an external hardware reset, the μPD7281 is in the Normal mode of operation. The μPD7281 can enter the Test mode for program debugging by inputting a SETBRK token (see figure 6) while the processor is in the Normal mode. If an overflow occurs in the Data Queue or the Generator Queue, the processor enters into the Break mode so that the internal contents of the processor can be examined; see table 1. Table 2 describes the effects of software and hardware resets.

Table 1. DUMPD Output Token Format

MN	Z	ID	CTLF	DATA (16-bit field)
0000	0	0000 000	0111	xxxxx(5) GQ Size(5 bits) DQ Size(6 bits)
0000	0	0000 001	0111	xxxx(4) u(1) ID(7) CTLF(4)
0000	0	0000 010	0111	DATA(16)
0000	0	0000 011	0111	xxx (3) u(1) ID(7) x(1) C_B, S_B, C_A, S_A
0000	0	0000 100	0111	xx(2) FTL (Lower 12 bits) xx(2)
0000	0	0000 101	0111	$DATA_A$(16)
0000	0	0000 110	0111	$DATA_B$(16)
0000	0	0000 111	0111	xxxxxxxxx(9) ID(7)

x: Don't care u: Unused

Table 2. Effects of Reset Operation

	Hardware Reset	Software Reset
MN	μPD7281 reads in MN	No Change
High/Low Word Flip-flop	Reset	No Change
Input Inhibit Control	Reset (No constraint)	No Change
LT Break State	Reset	Reset
Internal Operation	Stopped	Stopped
DQ, GQ, and OQ Pointers	Set to 0	Set to 0

Figure 5. μPD7281 Operation Modes

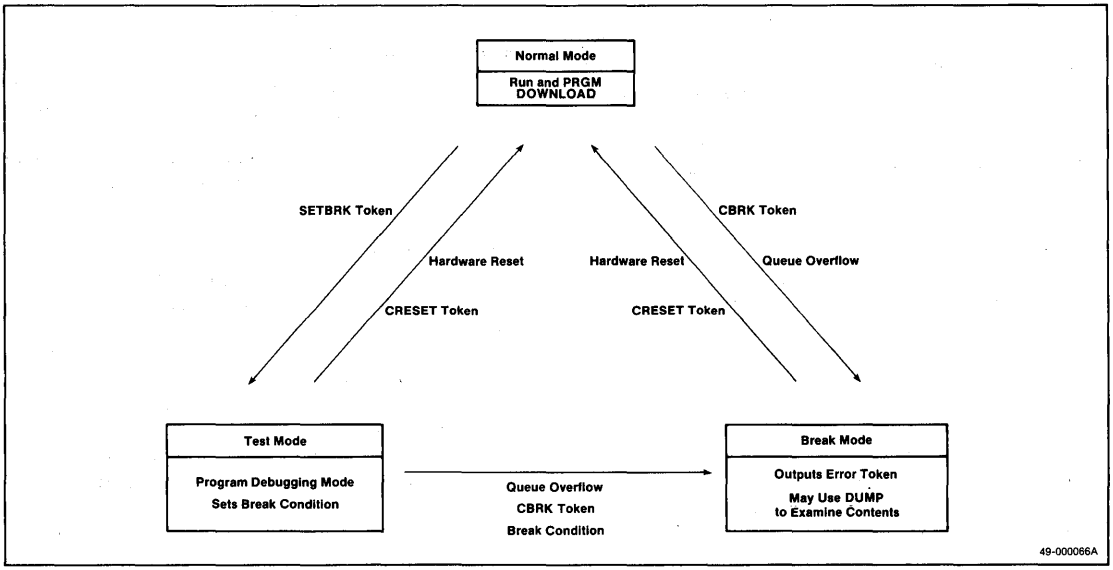
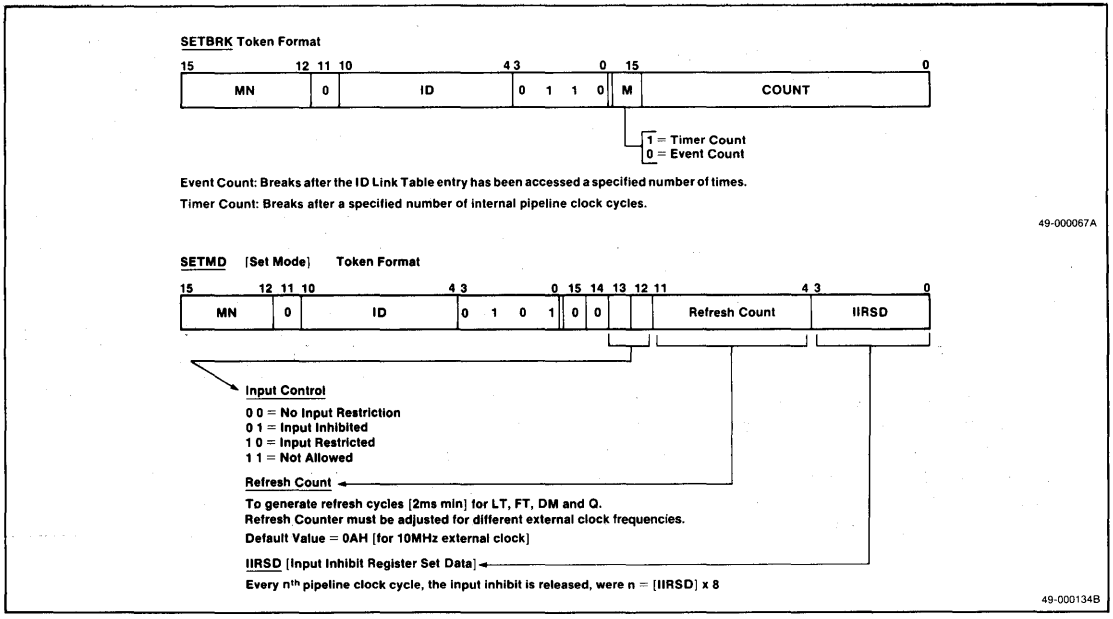


Figure 6. SETBRK (Set Break Condition) and SETMD (Set Mode) Token Formats



Input/Output Tokens

The only way any external device can communicate with the μPD7281 is by using the I/O tokens (see figure 7). Both the input and the output tokens have the same format so that a token may flow through a series of multiple processors without a format change. A 32-bit I/O token is divided into upper and lower 16-bit words and input to or output from the μPD7281 a 16-bit word at a time. Object code is down-loaded into the Link Table and the Function Table using SETLT, SETFTR, SETFTL and SETFTT input tokens. The contents of the Function Table and the Link Table can also be read using RDLT, RDFTR, RDFTL and RDFTT tokens. In order to write or read a value to and from the Data Memory, a program must be down-loaded and executed. Once object code is down-loaded into the μPD7281, data tokens are input to the processor, thereby initiating the processing. For a description of the input and output tokens, see tables 3 and 4.

2

Figure 7. Input/Output Token Format

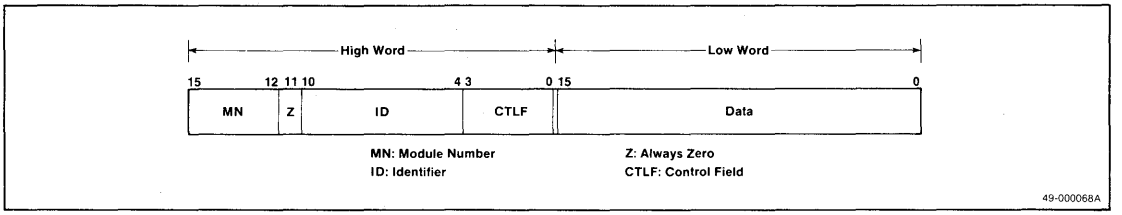


Table 3. Input Token Format

Input Token	High Word (16)				Low Word (16)	Remarks
	MN (4) 15 12	Z (1) 11	ID (7) 10 4	CTLF (4) 3 0	DATA (16) 15 0	
SETLT	MN	0	LT address	1 1 0 0	Data to be set in LT	Set LT
SETFTR	MN	0	FT address	1 1 0 1	Data to be set in FTR	Set FT Right Field
SETFTL	MN	0	FT address	1 1 1 0	Data to be set in FTL	Set FT Left Field
SETFTT	MN	0	FT address	1 1 1 1	Data to be set in FTT	Set FT Temporary Field
RDLT	MN	0	LT address	1 0 0 0		Read LT
RDFTR	MN	0	FT address	1 0 0 1		Read FT Right Field
RDFTL	MN	0	FT address	1 0 1 0		Read FT Left Field
RDFTT	MN	0	FT address	1 0 1 1		Read FT Temporary Field
CRESET	MN	0		0 1 0 0		Command Reset
SETMD	MN	0		0 1 0 1	Mode set data	Set Operation Mode
SETBRK	MN	0	ID	0 1 1 0	M (1) Count (15)	Set Break Condition
DUMP	MN	0	xxxx(4) DUMP (3)	0 1 1 1		Dump
CBRK	0 0 0 0	0		0 1 0 0		Command Break
VAN	1 1 1 1	0				Vanish Data
PASS	MN*	0				Pass Data
EXEC	MN	0	ID	0 0 C S	Data	Normal Execution Data

* When MN is not the current module number
x: Don't care

Table 4. Output Token Format

Output Token	Upper-Order Word (16)				Lower-Order Word (16)		Remarks
	MN (4) 15 12	Z (1) 11	ID (7) 10 4	CTLF (4) 3 0	DATA (16) 15 0		
LTRDD	0 0 0 0	0	LT address	1 0 0 0	Data read from LT		FT Read Data
FTRRDD	0 0 0 0	0	FT address	1 0 0 1	Data read from FTR		FT Right Field Read Data
FTLRDD	0 0 0 0	0	FT address	1 0 1 0	Data read from FTL		FT Left Field Read Data
FTTRDD	0 0 0 0	0	FT address	1 0 1 1	Data read from FTT		FT Temporary Field Read Data
PASSD	MN	0	ID	CTLFD	Data		Pass Data
ERR	0 0 0 0	0	0 0 0 0 0 0 0	0 1 0 0	MN(4)MODE(4) 0 0 0 STATUS(5)		Error Data
DUMPD	0 0 0 0	0	0 0 0 0 DUMP(3)	0 1 1 1	Dump data		Dumped Data
OUTD	MN	0	ID	0 0 C S	Data		Output Data

Instruction Set Summary

Tables 5 through 8 summarize the instruction set.

Table 5. AG/FC Instructions

Mnemonic	Instruction
QUEUE	Queue
RDCYCS	Read cyclic short
RDCYCL	Read cyclic long
WRCYCS	Write cyclic short
WRCYCL	Write cyclic long
RDWR	Read/Write Data Memory
RDIDX	Read Data Memory with index
PICKUP	Pickup data stream
COUNT	Count data stream
CONVO	Convolve
CNTGE	Count generation
DIVCYC	Divide cyclic
DIV	Divide
DIST	Distribute
SAVE	Save ID
CUT	Cut data stream

Table 6. PU Instructions

Mnemonic	Instruction
OR	Logical OR
AND	Logical AND
XOR	Logical EXCLUSIVE-OR
ANDNOT	Logical INVERT an operand then AND: ($\bar{A}$•B)
NOT	Invert
ADD	Add
SUB	Subtract

Table 6. PU Instructions (cont)

Mnemonic	Instruction
MUL	Multiply
NOP	No operation
ADDSC	Add and shift count
SUBSC	Subtract and shift count
MULSC	Multiply and shift count
NOPSC	NOP and shift count
INC	Increment
DEC	Decrement
SHR	Shift right
SHL	Shift left
SHRBRV	Shift right with bit reverse
SHLBRV	Shift left with bit reverse
CMPNOM	Compare and normalize
CMP	Compare
CMPXCH	Compare and exchange
GET1	Get one bit
SET1	Set one bit
CLR1	Clear one bit
ANDMSK	Mask a word with logical AND
ORMSK	Mask a word with logical OR
CVT2AB	Convert 2's complement to sign-magnitude
CVTAB2	Convert sign-magnitude to 2's complement
ADJL	Adjust long (for double precision numbers)
ACC	Accumulate
COPYC	Copy control bit

Table 7. GE Instructions

Mnemonic	Instruction
COPYBK	Copy block
COPYM	Copy multiple
SETCTL	Set control field

Table 8. OUT Instructions

Mnemonic	Instruction
OUT1	Output 1 token
OUT2	Output 2 tokens

There are four different types of instructions which can be specified by the SEL field of an FT token. See table 9.

Table 9. SEL Field of an FT Token

SEL	Type	Description
11	AG/FC	Executes instructions specified by the Function Table Right field while monitoring the Function Table Temporary field.
01	PU	Performs arithmetic, logical, barrel-shift, bit-manipulation, data-conversion, etc.
10	GE	Generates a block or multiple new tokens from a token. Sets the control field of a token. Increments or decrements the data field of a token.
00	OUT	Outputs data tokens from the circular pipeline to the Output Queue after the tokens are finished being processed.

AG/FC Instructions

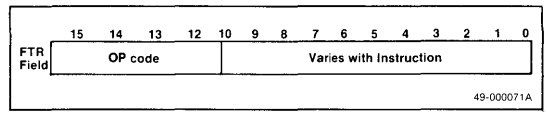
There are 16 AG/FC instructions (see table 10). They can be grouped into three types: Address Generator (AG), Flow Controller (FC), and AG/FC type.

AG type: RDCYCS, RDCYCL, WRCYCS, WRCYCL, RDWR, RDIDX

FC type: PICKUP, COUNT, CUT, DIVCYC, DIV, DIST, CONVO, SAVE, CNTGE

AG/FC type: QUEUE

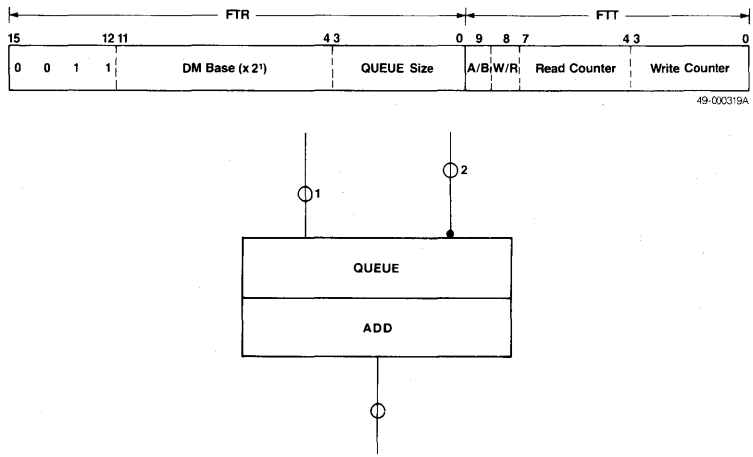
A 4-bit OP code in the Function Table right field specifies the instruction to be executed.



QUEUE

For a two-operand instruction, a QUEUE instruction is used to temporarily store the first operand token in the Data Memory until the second operand token arrives. The maximum Queue size is 16. See figure 8.

Figure 8. QUEUE Instruction



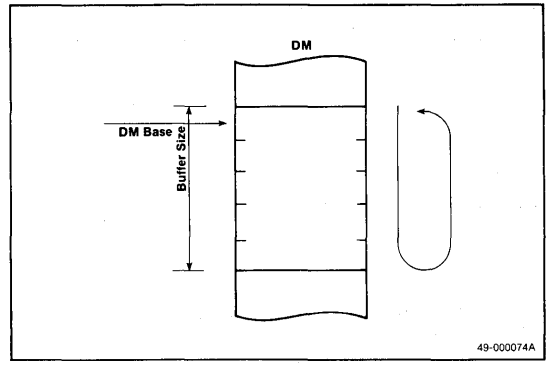
Note: See Data-flow Graph Explanation [figure 27] for the explanation of figures.

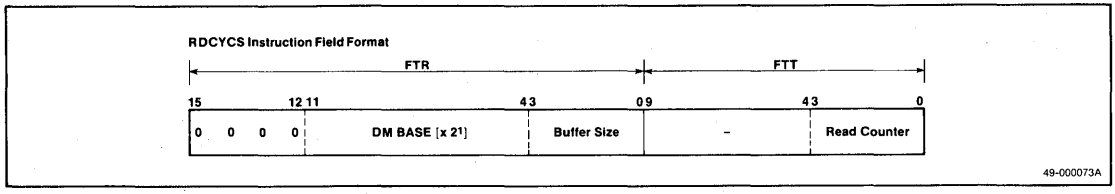
49-000072A

RDCYCS [Read Cyclic Short]

RDCYCS reads 18-bit data words from the Data Memory cyclically (see figure 9). The first data to be read is specified by the DM Base address. The last data to be read is specified by the buffer size. The Read Counter (RC) contains the offset address from Data Memory Base (DMB) address. It is incremented each time the Data Memory is accessed. The maximum buffer size is 16.

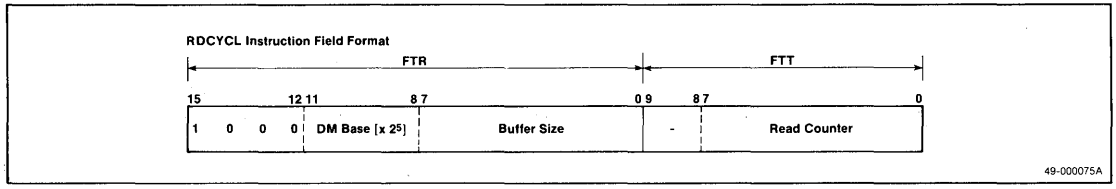
Figure 9. RDCYCS Instruction Operation





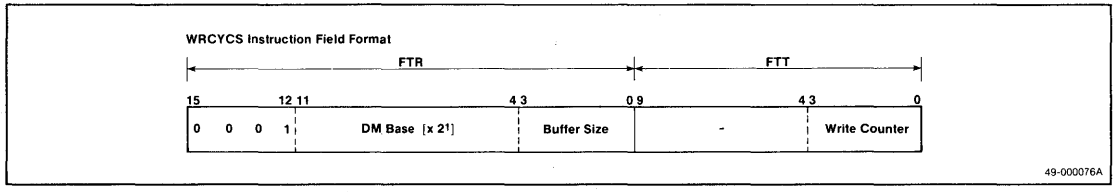
RDCYCL [Read Cyclic Long]

RDCYCL reads 18-bit data words from the Data Memory in a cyclic manner like RDCYCS but has a longer cyclic range. The first data to be read is specified by the DM Base address. The last data to be read is specified by the buffer size. The maximum buffer size is 256.



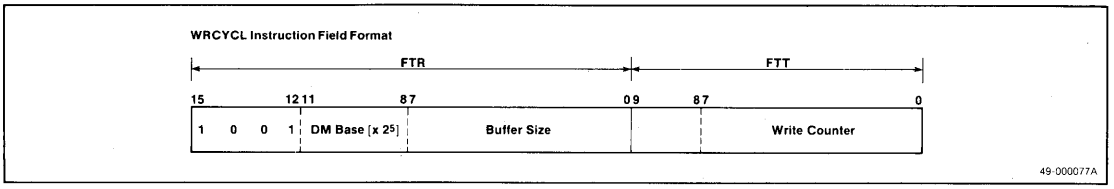
WRCYCS [Write Cyclic Short]

WRCYCS writes 18-bit data words into the Data Memory cyclically. The first the Data Memory address is specified by the DM Base address. The last address is specified by the buffer size. The maximum buffer size is 16.



WRCYCL [Write Cyclic Long]

WRCYCL writes 18-bit data words into the data memory in a cyclic manner similar to WRCYCS but has a longer cyclic range. The first DM address is specified by the DM Base address. The last address is specified by the buffer size. The maximum buffer size is 256.



RDWR [Read/Write Data Memory]

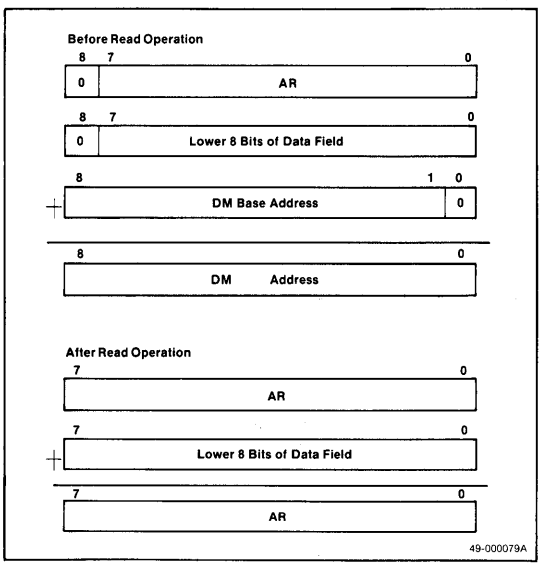
RDWR is used to write or read data to and from the Data Memory. This instruction reads/modifies/writes the Data Memory with the Address Register as index.

If a token arriving at the instruction has FTRC bit = 0, then the instruction performs a DM read operation. If it has FTRC bit = 1, then the instruction performs a DM write operation.

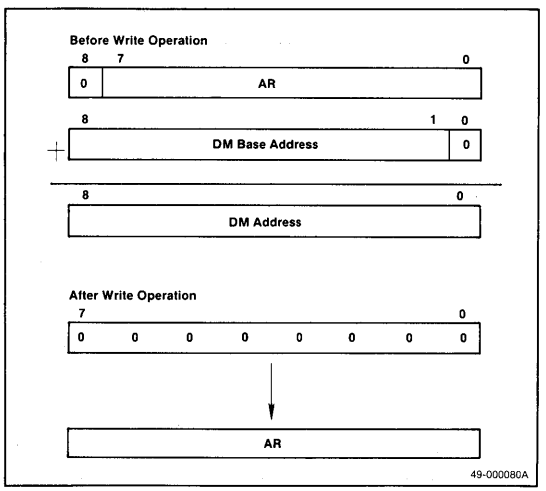
For a token with the FTRC bit = 0, the actual DM address location to be read is determined by the sum of the following three values: 8-bit Address Register (AR), the lower eight bits of the data field of the token, and the DM Base address. After the read operation, the lower eight bits of the token's data field is added to the value of AR. Additionally, the data field of the token is replaced by the contents read from the Data Memory location.

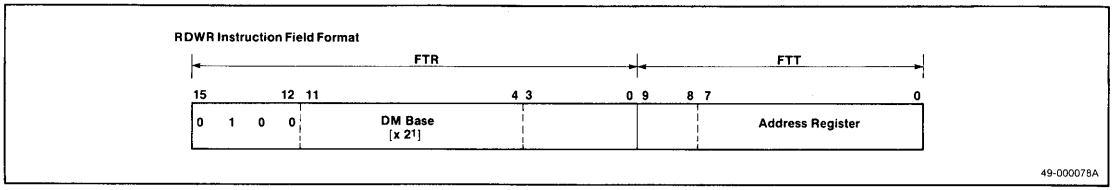
If a token with FTRC bit = 1 is used along with RDWR, a write operation is performed. The Data Memory address location is determined by the sum of 8-bit AR and DM Base address. The 18-bit data from the token is written into the DM address calculated. After the write operation, AR is reset to 00H.

FTRC = 0



FTRC = 1



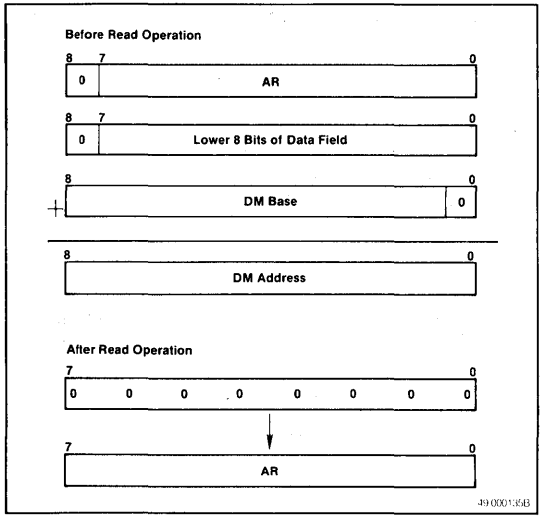


RDIDX [Read Data Memory with Index]

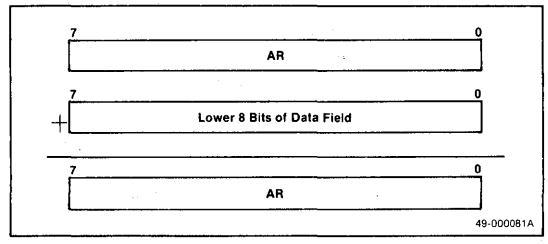
RDIDX is used to read the contents of the Data Memory. This instruction is most useful when a part of the Data Memory is used as a look-up table. The RDIDX instruction performs different operations depending upon the FTRC bit of the token using the instruction. If the FTRC bit = 0, then the instruction reads a Data Memory location. The DM address location to be read is determined by the sum of the following three values: the 8-bit AR, the lower eight bits of the token's data field, and the DM Base address. After the read operation, the data field of the token is replaced by the contents of the Data Memory location read. The value of AR is reset to zero after the operation.

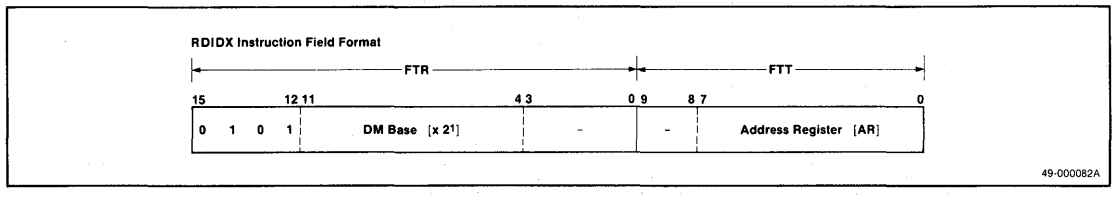
If the FTRC bit = 1, no operation is performed on the Data Memory. However, the token's AR contents are replaced by the modulo-256 sum of the lower eight bits of data field and the current contents of AR.

FTRC = 0



FTRC = 1





PICKUP [Pickup Data Stream]

This instruction picks up every $(n+1)^{th}$ token from a stream of incoming tokens and increments the $(n+1)^{th}$ token's ID field by one. The number n is specified by the Count Size (CS) of the Function Table Right field.

Figure 10 illustrates the PICKUP instruction with CS = 3.

Note: These figures use the data-flow graph convention. See figure 27, Data-flow Graph Explanation for the explanation of figures.

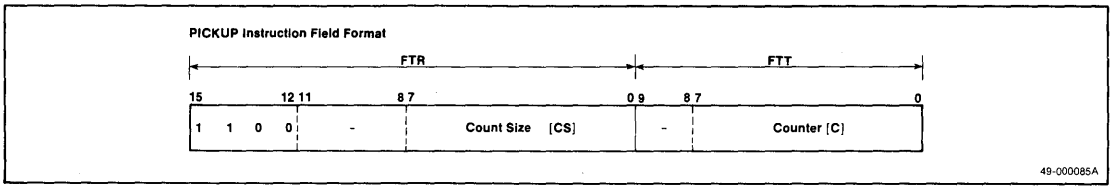
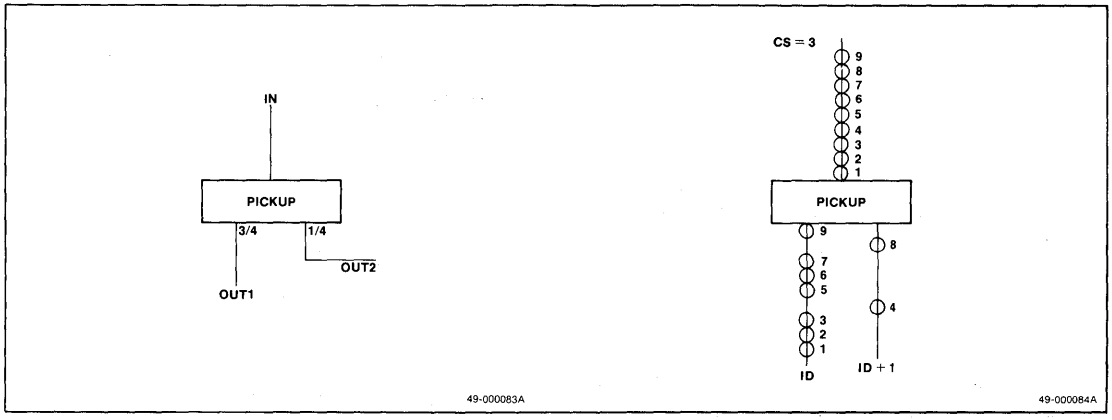


Figure 10. Pickup Instruction



COUNT [Count Data Stream]

COUNT copies every $(n+1)^{th}$ token from a stream of incoming tokens and increments the copied token's ID field by one. The number n is specified by CS of the Function Table Right field. Figure 11 illustrates the COUNT instruction with CS = 3.

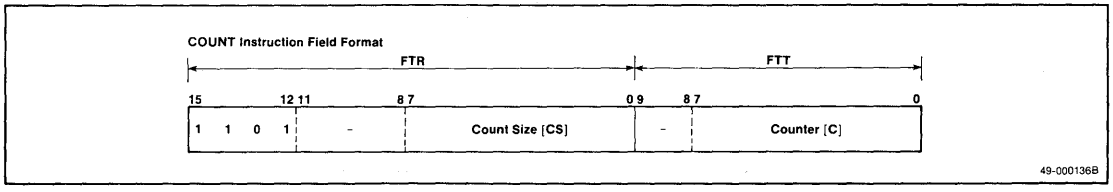
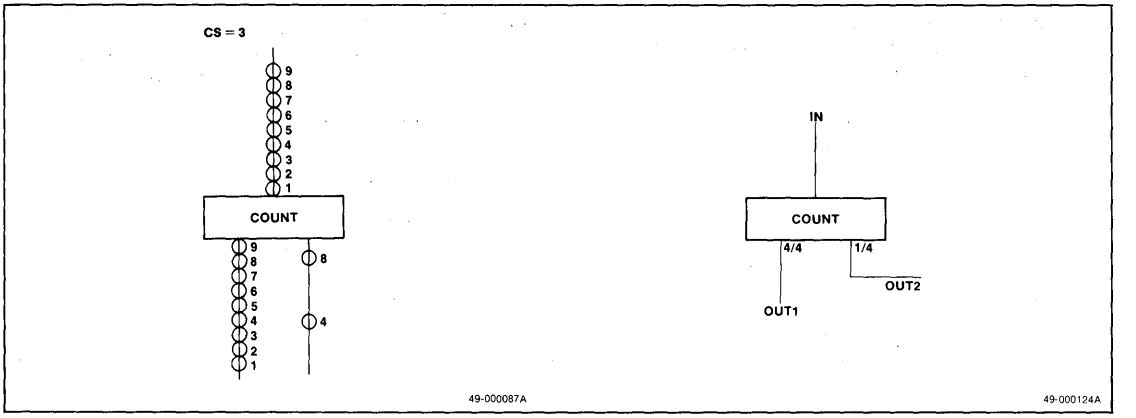


Figure 11. COUNT Instruction



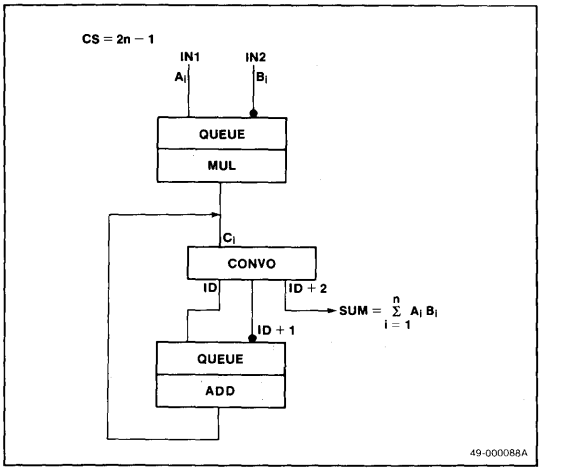
CONVO [Convolve]

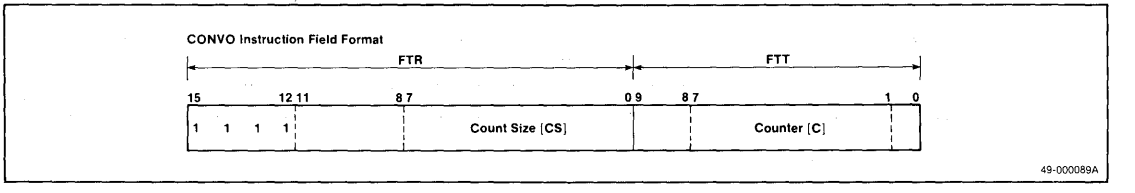
CONVO instruction is used to perform cumulative operations such as ΣA_i or ΠA_i. The CONVO instruction is best suited for convolving two sequences of the same length. Figure 12 illustrates the CONVO instruction by computing

$$SUM = \sum_{i=1}^{n} A_i B_i.$$

The A_i sequence is input to IN1 while the B_i sequence is input to IN2. Together they are queued and multiplied to form the C_i sequence. The C_i's arriving at CONVO instruction are queued and added together to form the final answer SUM. The length of the summation, n, is specified by the CS.

Figure 12. CONVO Instruction

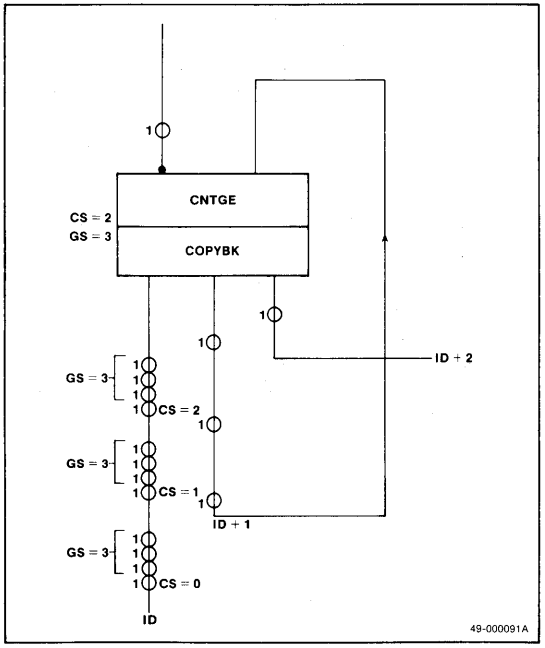


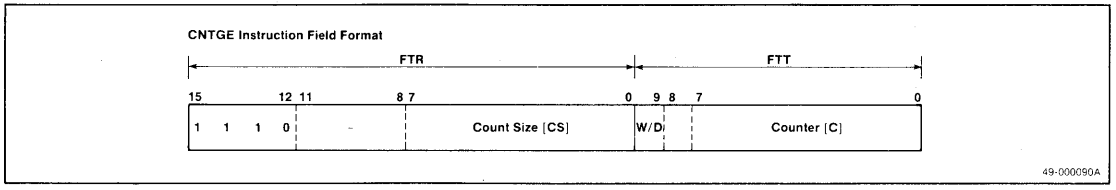


CNTGE [Count Generation]

CNTGE is normally used with COPYBK (Copy Block) to generate more than 16 copies of a single token (see figure 13). This instruction has both the dead (inactive) state and the wait (active) state. The instruction starts in the dead state. The FTRC bit = 0 tokens that arrive during the dead state of instruction are output to the ID + 2 token stream. It enters the wait state when a token with FTRC bit = 1 arrives and the token is output to ID token stream. Once the instruction is in the wait state, it counts the number of tokens arriving with FTRC bit = 0, outputting them to the ID token stream, until the number exceeds the number specified by CS. If Counter (C) reaches the number specified by Count Size (CS), the instruction automatically enters the dead state. Tokens with the FTRC bit = 1 arriving at CNTGE while the instruction is in the wait state are deleted by the instruction. Once the instruction enters the dead state, it can be reactivated by the arrival of a token with FTRC bit = 1.

Figure 13. CNTGE Instruction



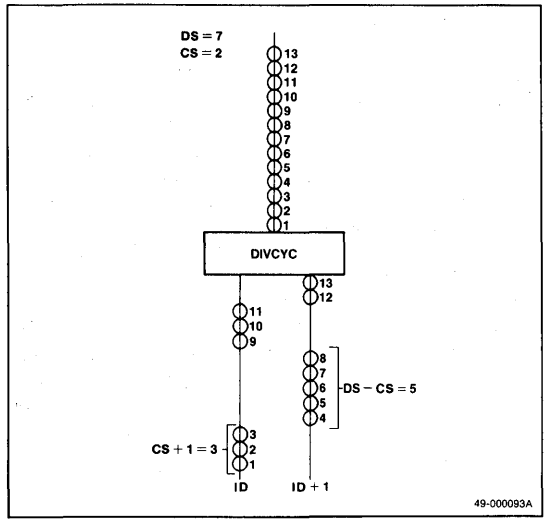


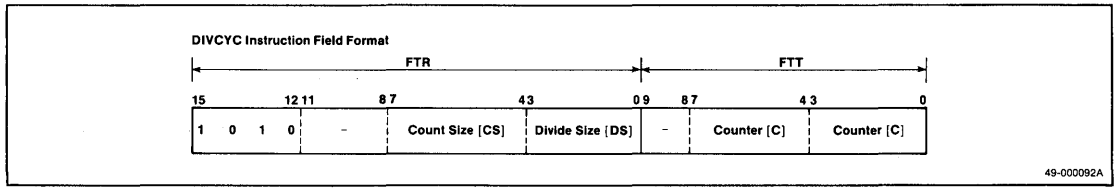
DIVCYC [Divide Cyclic]

DIVCYC divides an incoming stream of tokens into two streams of tokens: an ID token stream and an ID + 1 token stream. The pattern in which the incoming tokens are divided is specified by the Divide Size (DS) and Count Size (CS). The DS specifies cycle size whereas CS specifies the number of consecutive tokens to be in the ID stream. The first CS + 1 tokens are output to the ID token stream. The following consecutive (DS - CS) tokens are output to the ID + 1 token stream.

Figure 14 illustrates the DIVCYC instruction with DS = 7 and CS = 2. Note that an incoming stream of tokens is divided into a stream of ID tokens and a stream of ID + 1 tokens with a cycle of 8 tokens. Since CS = 2, the number of ID tokens in one cycle is 3, the number of ID + 1 tokens in a cycle is 5.

Figure 14. DIVCYC Instruction

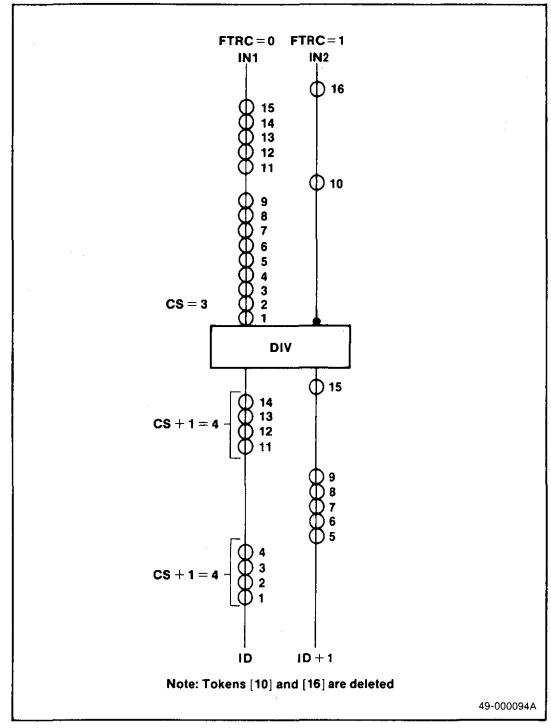


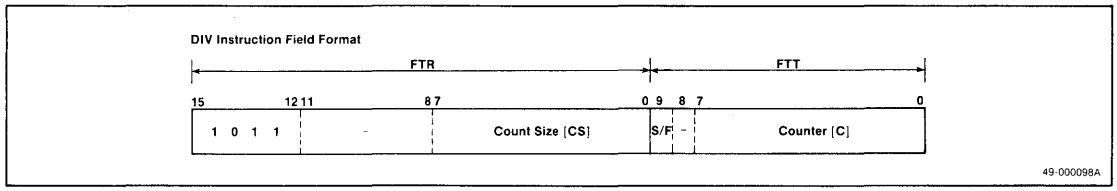


DIV [Divide]

DIV with CS = n divides an incoming stream of tokens with FTRC bit = 0 into two streams of tokens: ID tokens and ID + 1 tokens. The first (n + 1) incoming tokens with FTRC bit = 0 are output as the ID tokens, and the rest of the incoming tokens with FTRC bit = 0 are output as ID + 1 tokens. An incoming token with FTRC bit = 1 is used to reinitialize the DIV instruction. The stream of input tokens with FTRC bit = 0 after the reinitialization is again divided into a stream of (n + 1) ID tokens followed by ID + 1 tokens. A token with FTRC bit = 1 which reinitializes the DIV instruction is deleted from the output token stream. A DIV instruction with CS = 3 is illustrated in figure 15. The 10th and 16th input tokens have FTRC bit = 1, so they reinitialize the DIV instruction.

Figure 15. DIV Instruction

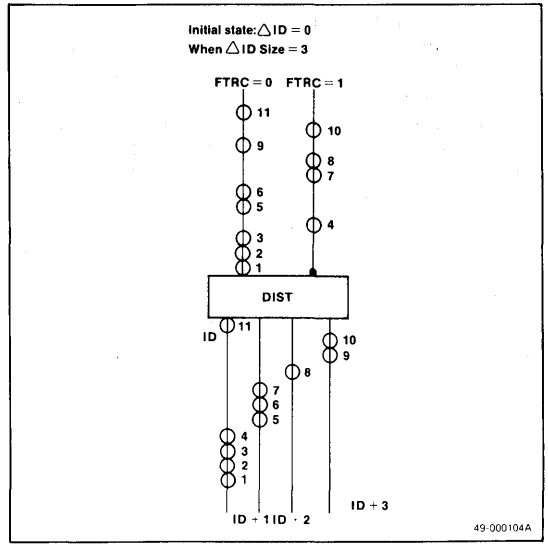


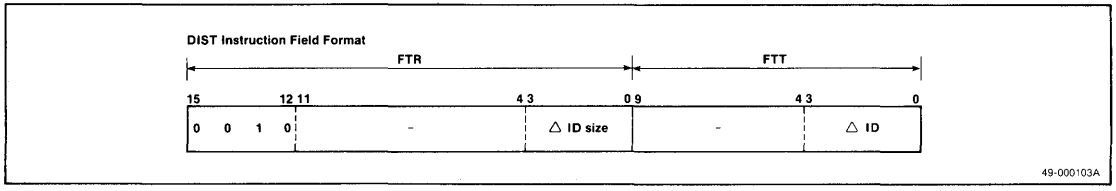


DIST [Distribute]

DIST is used to divide a stream of incoming tokens with the same ID into more than one stream of tokens with different IDs (see figure 16). The ΔID size determines the maximum number of output token streams the instruction can have. ΔID is the value added to an incoming token's ID field to form the ID field of the output token. The ΔID field is initially set to zero, and it is incremented by one after a token with FTRC bit = 1 passes through the instruction. However, a token with FTRC bit = 0 has no effect on the value of ΔID field. If the value of ΔID before being incremented by a token with the FTRC bit = 1 is equal to the contents of the ΔID size field, the ΔID field will be reset to zero.

Figure 16. DIST Instruction



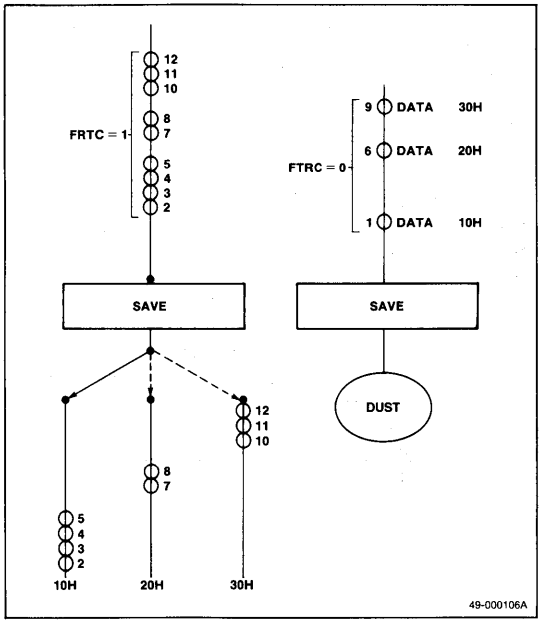


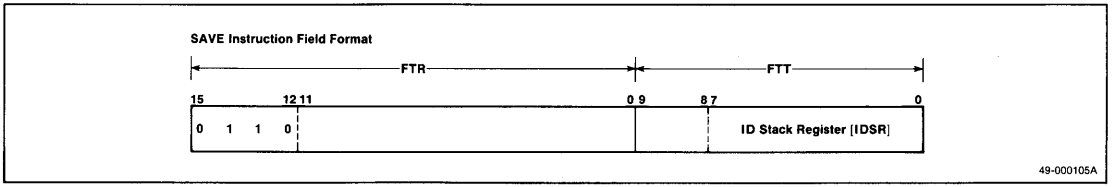
SAVE [Save ID]

SAVE is used to set the value of the ID field of a token. The instruction performs two different operations depending on whether the token's FTRC bit is 1 or 0. If the token's FTRC bit = 0, the instruction copies the lower eight bits of the data field into the Identifier Stack Register (IDSR) field. However, if the token's FTRC bit is 1, the instruction replaces the token's ID field with the contents of IDSR.

Figure 17 illustrates the use of the SAVE instruction. Token 1 assigns an ID field value of 10H to tokens 2, 3, 4 and 5, token 6 assigns an ID field value of 20H to tokens 7 and 8, and token 9 assigns an ID field value of 30H to tokens 10, 11 and 12. In this example, tokens 1, 6 and 9 are deleted after SAVE instruction.

Figure 17. SAVE Instruction

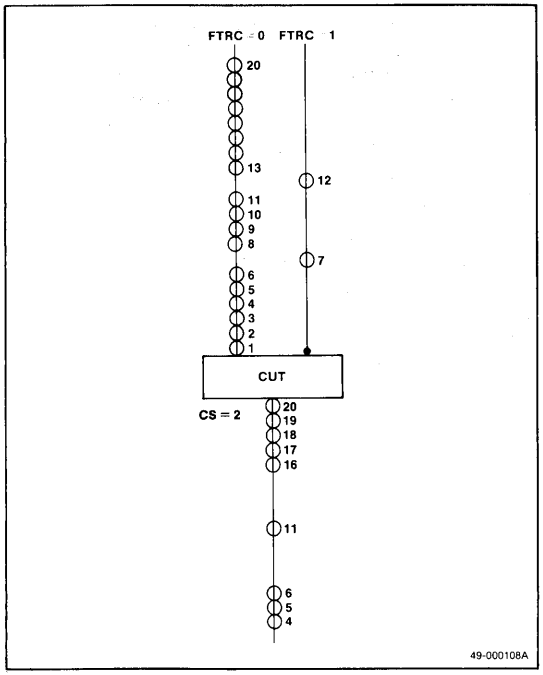




CUT [Cut Data Stream]

CUT is used to delete unnecessary tokens from a series of incoming tokens. The first n tokens arriving at the instruction are deleted, where n is the value contained in the CS field of the instruction. Initially the S/F bit and the Counter (C) are set to zero. When a token with its FTRC bit = 0 enters the instruction while S/F bit is zero, the token increments the Counter by one and the token itself is deleted. As the first (n + 1) tokens are deleted by the instruction, the Counter has the same value as n, the contents of CS field. This condition sets the S/F bit to 1. When the S/F bit is 1, a token with its FTRC bit = 0 can pass through the instruction without being deleted. However, if a token with its FTRC bit = 1 passes through the instruction, it resets the S/F bit to 0, thereby reinitializing the instruction. The token with its FTRC bit = 1 is also deleted after reinitializing the instruction. Figure 18 illustrates the use of CUT to delete tokens 7 and 12 and the three tokens following them.

Figure 18. CUT Instruction



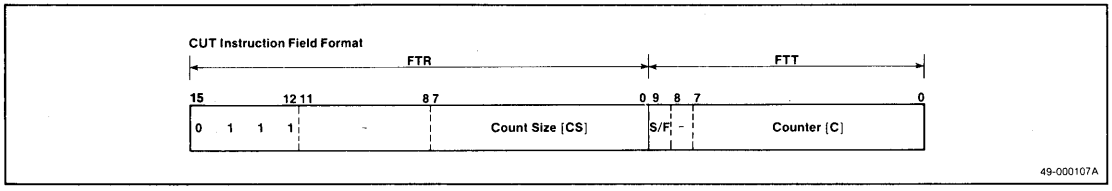
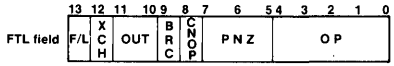


Table 10. AG and FC Instructions

Mnemonic	FTR (16) 15 14 13 12	11 10 9 8 7 6 5 4 3 2 1 0	FTT (10) 9 8	7 6 5 4 3 2 1 0	FTRC (1)	Operation
QUEUE	0 0 1 1	DM Base (x 2^1) (8) / Queue Size (4)	A/B W/R	Read Counter (4) / Write Counter (4)		Synchronize two tokens
RDCYCS	0 0 0 0	DM Base (x 2^1) (DMB) (8) / Buffer Size (BS) (4)	(6)	Read Counter (RC) (4)	0	DATA ← (DMB + RC), RC ← RC + 1
					1	DATA ← (DMB + RC), RC ← RC + 1, when BS = RC, copy with ID + 1
RDCYCL	1 0 0 0	DM Base (x 2^5) (4) / Buffer Size (8)	(2)	Read Counter (8)	0	DATA ← (DMB + RC), RC ← RC + 1
					1	DATA ← (DMB + RC), RC ← RC + 1, when BS = RC, copy with ID + 1
WRCYCS	0 0 0 1	Base (x 2^1) (8) / Buffer Size (4)	(6)	Write Counter (WC) (4)	0	(DMB + WC) ← DATA, WC ← WC + 1, delete token
					1	(DMB + WC) ← DATA, WC ← WC + 1, when BS = WC, token not deleted
WRCYCL	1 0 0 1	DM Base (x 2^5) (4) / Buffer Size (8)	(2)	Write Counter (8)	0	(DMB + WC) ← DATA, WC ← WC + 1, delete token
					1	(DMB + WC) ← DATA, WC ← WC + 1, when BS = WC, token not deleted
RDWR	0 1 0 0	DM Base (x 2^1) (8) / (4)	(2)	Address Register (AR) (8)	0	DATA ← (DMB + AR + DATA), AR ← AR + DATA
					1	(DMB + AR) ← DATA, AR ← 0
RDIDX	0 1 0 1	DM Base (x 2^1) (8) / (4)	(2)	Address Register (8)	0	DATA ← (DMB + AR + DATA), AR ← 0
					1	AR ← AR + DATA
PICKUP	1 1 0 0	(4) / Count Size (CS) (8)	(2)	Counter (C) (8)	0	When CS≠C, C←C+1; when CS=C, distribute, C←0
					1	C ← C + DATA, token deleted
COUNT	1 1 0 1	(4) / Count Size (8)	(2)	Counter (8)	0	When CS ≠ C, C ← C + 1; when CS = C, copy token, C ← 0
					1	C ← C + DATA, token deleted
CUT	0 1 1 1	(4) / Count Size (8)	S/F (1)	Counter (8)	0	When S/F = 0 and C ≤ CS, C ← C + 1, delete token; when S/F = 0 and C > CS, or when S/F = 1, C ← C+1, token not deleted
					1	S/F ← 0, C ← 0, token deleted
DIVCYC	1 0 1 0	(4) / Count Size (4) / Divide Size (4)	(2)	Counter (4) / Counter (4)	0	When C ≤ CS, C ← C + 1; when C> CS, distribute, C ← C + 1; C←C. When C = DS, C ← 0
					1	C ← C + DATA, token deleted
DIV	1 0 1 1	(4) / Count Size (8)	S/F (1)	Counter (8)	0	When S/F=0 and C ≤ CS, C ← C+1; when S/F = 0 and C > CS, or when S/F = 1, distribute, C ← C + 1;
					1	S/F ← 0, C ← 0, token deleted
DIST	0 0 1 0	(8) / Δ ID Size (4)	(6)	ΔID (4)	0	ID ← (ID + ΔID) modulo ΔID size
					1	When ΔID ≠ ΔID size, ID←(ID+ΔID) modulo ΔID size, ΔID ← ΔID + 1. When ΔID = ΔID size, ΔID ← 0
CONVO	1 1 1 1	(4) / Count Size (8)	(2)	Counter (7) (1)		When CS ≠ C, ID ← ID + C (modulo 2), C ← C + 1; when CS = C, ID ← ID + 2, C ← 0
SAVE	0 1 1 0	(12)	(2)	ID Stack Register (8) (IDSR)	0	IDSR ← Lower 8-bit of DATA
					1	ID ← IDSR
CNTGE	1 1 1 0	(4) / Count Size (8)	W/D (1)	Counter (8)	0	When dead, ID ← ID + 2; when wait, if C = CS, C ← 0, W/D = 0; when wait, if C ≠ CS, C ← C + 1
					1	When dead, initialization; when wait, delete token

PU Instructions

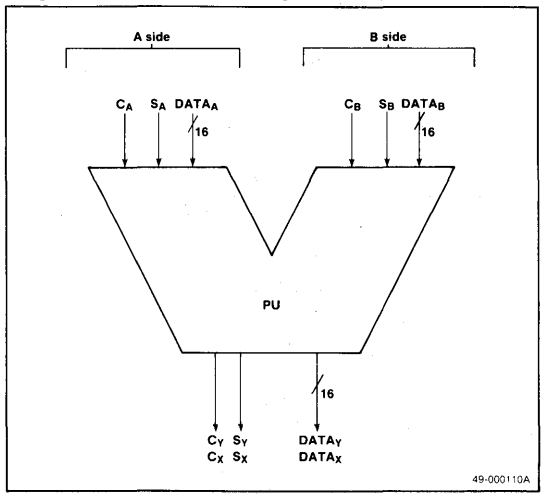


F/L: Full/Left
XCH: Exchange
OUT:Output
BRC: Branch Control
CNOP: C-Bit NOP
PNZ: Positive, Negative, or Zero
OP: Op code

49-000109A

PU instructions (see table 20) are stored in the Function Table Left field of the Function Table memory. The bits 0 through 11 are used as control information for the Processing Unit. The bits 12 and 13 are deleted before the token arrives at the Processing Unit. Two operands from the A and B sides are operated on by the Processing Unit and the result is output to the X and Y sides (see figure 19).

Figure 19. The Processing Unit



49-000110A

Bit Assignments

F/L [Full/Left]: F/L bit = 0 indicates that the PU instruction is a one-operand instruction, and only the Function Table Left field is meaningful. F/L bit = 1 indicates that the PU instruction is a two-operand instruction, and both the Function Table Left field and the Function Table Right field are meaningful. Therefore, when F/L bit = 1, the PU instruction is used in conjunction with an AG/FC instruction.

XCH [Exchange]: This bit controls the exchange operation. Operands will be exchanged just before the two tokens enter the QUEUE when XCH = 1.

OUT [Output]: There are four different PU output token formats. The two OUT bits specify the output token format. See table 11.

Table 11. OUT Bits

OUT Bits	No. of Outputs	First Output ID	First Output DATA, C, S	Second Output ID	Second Output DATA, C, S
0 0	1	ID	X[1]		
0 1	1	ID	Y[2]		
1 0	2	ID	X	ID + 1	X
1 1	2	ID	X	ID + 1	Y

Notes: 1. This is the 18-bit result of the operation output to the X side. It includes the C_X and S_X bits.
2. This is the 18-bit result of the operation output to the Y side. It includes the C_Y and S_Y bits.

BRC [Branch Control]: The BRC bit controls the flow of the PU output data token. The output data token may be output to either the ID token stream or the ID + 1 token stream. When the BRC bit is set to 1 and the C bit of the PU output data token is also 1, the output data token is sent to the ID + 1 token stream. But when the BRC bit is set to 1 and the C bit of the output data token is 0, the token is sent to the ID token stream. Therefore, using the BRC bit implements a conditional branch on C.

CNOP Bit: This bit informs the Processing Unit whether or not the incoming token should be processed. If the CNOP bit is set, and the C_A bit is not equal to the C_B bit, then the token passes through the Processing Unit with no operation performed. See table 12.

Table 12. CNOP Bit

C_A	C_B	PU Operation
0	0	Processing specified by the OP code is performed.
0	1	Token passes through the Processing Unit as NOP.
1	0	Token passes through the Processing Unit as NOP.
1	1	Processing specified by the OP code is performed.

PNZ [Positive, Negative, Zero] Field: The PNZ field is used to test the resulting condition of the PU operation. If the resulting condition matches the condition set by the PNZ field, then the C bit of the output data token is set to 1. See table 13.

Table 13. PNZ Field

P	N	Z	Condition	C_X	C_Y	Assembler Description	
0	0	0	No condition set	C_A	C_B		
0	0	1	Result of operation = 0	1	1	EQ	True
			Result of operation ≠ 0	0	0		False
0	1	0	Result of operation < 0	1	1	LT	True
			Result of operation ≥ 0	0	0		False
0	1	1	Result of operation ≤ 0	1	1	LE	True
			Result of operation > 0	0	0		False
1	0	0	Result of operation > 0	1	1	GT	True
			Result of operation ≤ 0	0	0		False
1	0	1	Result of operation ≥ 0	1	1	GE	True
			Result of operation < 0	0	0		False
1	1	0	Result of operation ≠ 0	1	1	NE	True
			Result of operation = 0	0	0		False
1	1	1	Overflow generated	1	1	OVF	True
			No overflow generated	0	0		False

OP Code Field: This 5-bit OP code field specifies the PU operations to be performed. See table 14

Table 14. OP Code Field

Instruction	Mnemonic	Opcode
Logical	OR	00000
	AND	00001
	XOR	00010
	ANDNOT	00011
	NOT	01100
Arithmetic	ADD	11000
	ADDSC	11100
	SUB	11001
	SUBSC	11101
	MUL	11010
	MULSC	11110
	NOP	11011
	NOPSC	11111
	INC	01010
	DEC	01011
Shift	SHL	00100
	SHLBRV	00101
	SHR	00110
	SHRBRV	00111
Compare	CMPNOM	01000
	CMP	01001
	CMPXCH	10001
Bit manipulation	GET1	10101
	SET1	10110
	CLR1	10111
Bit check	ANDMSK	01101
	ORMSK	10000
Data conversion	CVT2AB	01110
	CVTAB2	01111
Double precision adjust	ADJL	10100
Accumulative addition	ACC	10010
C bit copy	COPYC	10011

Logical Instructions

These instructions perform 16-bit logical operations on $DATA_A$ and $DATA_B$. Usually there are no changes in C and S bits between the input token and the output token, however C bits can be affected by PNZ condition when specified.

OR, AND, XOR: These instructions perform 16-bit logical OR, AND, and XOR operations using input data tokens from the A and B sides of the Processing Unit. The 16 bit result is output to the X side.

ANDNOT: This instruction first complements $DATA_A$ and then performs logical AND operation with $DATA_B$. The 16-bit result is output to the X side.

NOT: This is a one-operand instruction which requires 16-bit data input from the A side only. The B side input is ignored. This instruction complements the 16-bit input data from the A side. The 16-bit result is output to the X side.

Arithmetic Instructions

These instructions perform 17-bit (including the sign bit) arithmetic operations on $DATA_A$ and $DATA_B$. When a PNZ condition is specified, the C bits of output data, C_X and C_Y, reflect the setting. However, if no PNZ condition is specified (i.e., PNZ = 000), then $C_X \leftarrow C_A$ and $C_Y \leftarrow C_B$.

ADD, SUB: These instructions perform addition or subtraction on $DATA_A$ and $DATA_B$ along with the sign bits, S_A and S_B. The result is output to the X side. $DATA_Y$ is normally 0000H. However, if an overflow occurs, then $DATA_Y$ is equal to +0001H ($S_Y = 0$). If an underflow occcurs, then the $DATA_Y$ is equal to −0001H ($S_Y = 1$).

MUL: This instruction multiplies $DATA_A$ and $DATA_B$. The correct sign bit for the product is determined from S_A and S_B. The 33-bit result including a sign bit is output as two 17-bit words, S_X and $DATA_X$, followed by S_Y and $DATA_Y$. $DATA_X$ is the upper 16-bit word and $DATA_Y$ is the lower 16-bit word. S_X holds the resulting sign bit, and S_Y is a mere duplicate of S_X.

NOP: This instruction performs no operation on the input token. The input data from A and B sides are output to the X and Y sides, respectively, without any change in their contents. If any control other than the OP code (such as PNZ control, BRC control, etc.) has been specified, the output complies with the control.

Shift Count Instructions

These four Shift Count (SC) instructions first perform the normal operations, then count the number of leading zeros in $DATA_X$ of the result, and finally output the number of zeros as $DATA_Y$ (see table 15). These instructions are provided for easy floating point processing.

ADDSC, SUBSC, NOPSC: These instructions perform addition, subtraction, or no operation. The number of preceding zeros in $DATA_X$ of the result is output as $DATA_Y$. If an overflow or an underflow occurs as a result of an operation, $DATA_Y$ contains + 0001H ($S_Y = 0$) or −0001H ($S_Y = 1$), respectively.

MULSC: This instruction performs a normal multiplication operation using the two 17-bit data. The upper order 16-bit data and its sign bit are output as $DATA_X$ and S_X, but the lower 16-bit data is not output as $DATA_Y$. Instead, the number of preceding zeros in $DATA_X$ are counted and output as $DATA_Y$. The S_Y bit is always zero.

Table 15. Shift Count Operation

$DATA_X$ After Operation																SC Output (Y)				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	S_Y	Y Data			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0 H
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	F H
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	x	0	0	0	0	E H
0	0	0	0	0	0	0	0	0	0	0	0	0	1	x	x	0	0	0	0	D H
0	0	0	0	0	0	0	0	0	0	0	0	1	x	x	x	0	0	0	0	C H
0	0	0	0	0	0	0	0	0	0	0	1	x	x	x	x	0	0	0	0	B H
0	0	0	0	0	0	0	0	0	0	1	x	x	x	x	x	0	0	0	0	A H
0	0	0	0	0	0	0	0	0	1	x	x	x	x	x	x	0	0	0	0	9 H
0	0	0	0	0	0	0	0	1	x	x	x	x	x	x	x	0	0	0	0	8 H
0	0	0	0	0	0	0	1	x	x	x	x	x	x	x	x	0	0	0	0	7 H
0	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	0	0	0	0	6 H
0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x	0	0	0	0	5 H
0	0	0	0	1	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	4 H
0	0	0	1	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	3 H
0	0	1	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	2 H
0	1	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	1 H
1	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0 H*

Notes: * When an overflow or underflow has occurred
x don't care

Increment and Decrement Instructions

INC, DEC: These instructions increment or decrement the 17-bit data from the A side (S_A and $DATA_A$), and outputs the result to X side as S_X and $DATA_X$. The S_Y and $DATA_Y$ are normally zero. However, if an overflow or an underflow occurs, then the Y side outputs + 0001H ($S_Y = 0$) or − 0001H ($S_Y = 1$), respectively.

Shift Instructions

SHR [Shift Right], SHL [Shift Left]: SHR or SHL instructions perform a barrel-shifting operation on the 16-bit data, $DATA_A$. The actual number of shifts and the direction is further specified by the lower five bits of $DATA_B$ and S_B, respectively. See figure 20 for detailed operation explanations.

Figure 20. SHR and SHL

Right Shift [SHR execution]

S_B	Lower 5 bits of $DATA_B$ [No. of shifts]	$DATA_X$	$DATA_Y$
0	00000	$A_{15}A_{14}...A_1A_0$	0...0
0	00001	0 | $A_{15}A_{14}...A_1$	A_0 | 0...0
0	00010	0 0 | $A_{15}...A_2$	A_1 A_0 | 0...0
0	00011	0...0 | $A_{15}...A_3$	A_2...A_0 | 0...0
0	00100	0...0 | $A_{15}...A_4$	$A_3...A_0$ | 0...0
0	00101	0...0 | $A_{15}...A_5$	$A_4...A_0$ | 0...0
0	00110	0...0 | $A_{15}...A_6$	$A_5...A_0$ | 0...0
0	00111	0...0 | $A_{15}...A_7$	$A_6...A_0$ | 0...0
0	01000	0...0 | $A_{15}...A_8$	$A_7...A_0$ | 0...0
0	01001	0...0 | $A_{15}...A_9$	$A_8...A_0$ | 0...0
0	01010	0...0 | $A_{15}...A_{10}$	$A_9...A_0$ | 0...0
0	01011	0...0 | $A_{15}...A_{11}$	$A_{10}...A_0$ | 0...0
0	01100	0...0 | A_{15}...A_{12}	$A_{11}...A_0$ | 0...0
0	01101	0...0 | A_{15}...A_{13}	$A_{12}...A_0$ | 0...0
0	01110	0...0 | A_{15} A_{14}	$A_{13}...A_0$ | 0 0
0	01111	0...0 | A_{15}	$A_{14}...A_0$ | 0
0	1XXXX	0...0	$A_{15}...A_1$ A_0
1	00000	A_{15} $A_{14}...A_1$ A_0	0...0
1	00001	$A_{14}...A_0$ | 0	0...0 | A_{15}
1	00010	$A_{13}...A_0$ | 0 0	0...0 | A_{15} A_{14}
1	00011	$A_{12}...A_0$ | 0...0	0...0 | A_{15}...A_{13}
1	00100	$A_{11}...A_0$ | 0...0	0...0 | A_{15}...A_{12}
1	00101	$A_{10}...A_0$ | 0...0	0...0 | $A_{15}...A_{11}$
1	00110	$A_9...A_0$ | 0...0	0...0 | $A_{15}...A_{10}$
1	00111	$A_8...A_0$ | 0...0	0...0 | $A_{15}...A_9$
1	01000	$A_7...A_0$ | 0...0	0...0 | $A_{15}...A_8$
1	01001	$A_6...A_0$ | 0...0	0...0 | $A_{15}...A_7$
1	01010	$A_5...A_0$ | 0...0	0...0 | $A_{15}...A_6$
1	01011	$A_4...A_0$ | 0...0	0...0 | $A_{15}...A_5$
1	01100	$A_3...A_0$ | 0...0	0...0 | $A_{15}...A_4$
1	01101	A_2...A_0 | 0...0	0...0 | $A_{15}...A_3$
1	01110	A_1 A_0 | 0...0	0 0 | $A_{15}...A_2$
1	01111	A_0 | 0...0	0 | $A_{15}...A_1$
1	1XXXX	0...0	$A_{15}...A_0$

49-000137C

Left Shift [SHL execution]

S_B	Lower 5 bits of $DATA_B$ [No. of shifts]	$DATA_X$	$DATA_Y$
0	00000	A_{15} $A_{14}...A_1$ A_0	0...0
0	00001	$A_{14}...A_0$ | 0	0...0 | A_{15}
0	00010	$A_{13}...A_0$ | 0 0	0...0 | A_{15} A_{14}
0	00011	$A_{12}...A_0$ | 0...0	0...0 | A_{15}...A_{13}
0	00100	$A_{11}...A_0$ | 0...0	0...0 | A_{15}...A_{12}
0	00101	$A_{10}...A_0$ | 0...0	0...0 | $A_{15}...A_{11}$
0	00110	$A_9...A_0$ | 0...0	0...0 | $A_{15}...A_{10}$
0	00111	$A_8...A_0$ | 0...0	0...0 | $A_{15}...A_9$
0	01000	$A_7...A_0$ | 0...0	0...0 | $A_{15}...A_8$
0	01001	$A_6...A_0$ | 0...0	0...0 | $A_{15}...A_7$
0	01010	$A_5...A_0$ | 0...0	0...0 | $A_{15}...A_6$
0	01011	$A_4...A_0$ | 0...0	0...0 | $A_{15}...A_5$
0	01100	$A_3...A_0$ | 0...0	0...0 | $A_{15}...A_4$
0	01101	A_2...A_0 | 0...0	0...0 | $A_{15}...A_3$
0	01110	A_1 A_0 | 0...0	0 0 | $A_{15}...A_2$
0	01111	A_0 | 0...0	0 | $A_{15}...A_1$
0	1XXXX	0...0	$A_{15}...A_0$
1	00000	$A_{15}A_{14}...A_1A_0$	0...0
1	00001	0 | $A_{15}A_{14}...A_1$	A_0 | 0...0
1	00010	0 0 | $A_{15}...A_2$	A_1 A_0 | 0...0
1	00011	0...0 | $A_{15}...A_3$	A_2...A_0 | 0...0
1	00100	0...0 | $A_{15}...A_4$	$A_3...A_0$ | 0...0
1	00101	0...0 | $A_{15}...A_5$	$A_4...A_0$ | 0...0
1	00110	0...0 | $A_{15}...A_6$	$A_5...A_0$ | 0...0
1	00111	0...0 | $A_{15}...A_7$	$A_6...A_0$ | 0...0
1	01000	0...0 | $A_{15}...A_8$	$A_7...A_0$ | 0...0
1	01001	0...0 | $A_{15}...A_9$	$A_8...A_0$ | 0...0
1	01010	0...0 | $A_{15}...A_{10}$	$A_9...A_0$ | 0...0
1	01011	0...0 | $A_{15}...A_{11}$	$A_{10}...A_0$ | 0...0
1	01100	0...0 | A_{15}...A_{12}	$A_{11}...A_0$ | 0...0
1	01101	0...0 | A_{15}...A_{13}	$A_{12}...A_0$ | 0...0
1	01110	0...0 | A_{15} A_{14}	$A_{13}...A_0$ | 0 0
1	01111	0...0 | A_{15}	$A_{14}...A_0$ | 0
1	1XXXX	0...0	$A_{15}...A_1$ A_0

49-000138C

SHRBRV [Shift Right with Bit Reverse], SHLBRV [Shift Left with Bit Reverse]: SHRBRV or SHLBRV first reverses the order of the bits in $DATA_A$ and then performs a normal SHR or SHL operation, respectively. See figure 21.

Compare Instructions

The Compare instructions (see table 16) are different from other PU instructions in that PNZ conditions must be specified along with the instructions. When a compare instruction is used along with a specified PNZ field, the Processing Unit performs a subtract operation. This subtract operation produces a set of PNZ flags, which are compared against the PNZ field specified by the instruction. When these two PNZ fields coincide, the specified PNZ conditions are said to be true. When they do not coincide, the specified PNZ conditions are said to be false (see table 17). The output data from the Processing Unit differs significantly depending on the PNZ conditions. The following three instructions compare the 17-bit data (S_A and $DATA_A$) from the A side against the 17-bit data (S_B and $DATA_B$) from the B side.

CMPNOM [Compare and normalize]: If the specified PNZ conditions are false, then the control bits, sign bits and data for both the X and Y sides are set to zero. If the PNZ conditions are true, then C_X and C_Y are set to one, S_X and S_X are set to zero, $DATA_X$ is set to 0001H, and $DATA_Y$ is set to 0000H.

CMP [Compare]: This instruction outputs the 17-bit data words from the A and B sides to the X and Y sides without any change in their contents. It only alters the control bits. If the specified PNZ conditions are true, then C_X and C_Y are set to one. If the PNZ conditions are false, then C_X is set to one and C_Y is set to zero.

CMPXCH [Compare and exchange]: If the specified PNZ conditions are true, then both the input data from the A side and B side are unchanged and output to the X side and Y side, respectively, including their sign bits and the control bits. However, if the PNZ conditions are false, then the input data from the A side is exchanged with the input data from the B side, including the control and sign bits.

Figure 21. Bit Reversal Operations in SHRBRV and SHLBRV

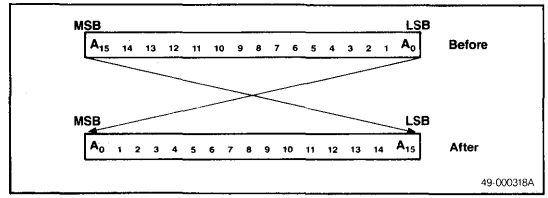


Table 17. PNZ Field Conditions for Compare Instructions

P	N	Z	Condition	True/ False	Function	Mnemonic
0	0	1	$S_A\ DATA_A = S_B\ DATA_B$	True	Equal	EQ
			$S_A\ DATA_A \neq S_B\ DATA_B$	False	Not equal	
0	1	0	$S_A\ DATA_A < S_B\ DATA_B$	True	Less than	LT
			$S_A\ DATA_A \geq S_B\ DATA_B$	False	Greater or equal	
0	1	1	$S_A\ DATA_A \leq S_B\ DATA_B$	True	Less or equal	LE
			$S_A\ DATA_A > S_B\ DATA_B$	False	Greater than	
1	0	0	$S_A\ DATA_A > S_B\ DATA_B$	True	Greater than	GT
			$S_A\ DATA_A \leq S_B\ DATA_B$	False	Less or equal	
1	0	1	$S_A\ DATA_A \geq S_B\ DATA_B$	True	Greater or equal	GE
			$S_A\ DATA_A < S_B\ DATA_B$	False	Less than	
1	1	0	$S_A\ DATA_A \neq S_B\ DATA_B$	True	Not equal	NE
			$S_A\ DATA_A = S_B\ DATA_B$	False	Equal	

Note: The significance of the PNZ bits when Compare instructions are executed differs from that of other instructions. Here, the use of PNZ = 111 or 000 is prohibited.

Table 16. Compare Instructions

Mnemonic	Input						Output						Notes
	C_A	S_A	$DATA_A$	C_B	S_B	$DATA_B$	C_X	S_X	$DATA_X$	C_Y	S_Y	$DATA_Y$	
CMPNOM	C_A	S_A	A	C_B	S_B	B	0	0	0000H	0	0	0000H	When PNZ is False
	C_A	S_A	A	C_B	S_B	B	1	0	0001H	1	0	0000H	When PNZ is true
CMP	C_A	S_A	A	C_B	S_B	B	0	S_A	A	0	S_B	B	When PNZ is false
	C_A	S_A	A	C_B	S_B	B	1	S_A	A	1	S_B	B	When PNZ is true
CMPXCH	C_A	S_A	A	C_B	S_B	B	C_A	S_A	A	C_B	S_B	B	When PNZ is true
	C_A	S_A	A	C_B	S_B	B	C_B	S_B	A	C_A	S_B	A	When PNZ is false

Bit Manipulation Instructions

GET1 [Get one bit]: This instruction is used to read a particular bit from $DATA_A$ (see table 18). A bit of $DATA_A$ specified by the lower 4 bits of $DATA_B$ is output as the least significant bit of $DATA_X$. All other bits of $DATA_X$ are set to zero. $DATA_Y$ is also set to zero. The control bits and the sign bits of $DATA_X$ and $DATA_Y$ are as follows: $C_X \leftarrow C_A$, $C_Y \leftarrow C_B$, $S_X \leftarrow S_A$, $S_Y \leftarrow 0$.

SET1 [Set one bit]: This instruction is used to set a particular bit of $DATA_A$. The bit of $DATA_A$ to be set is specified by the lower 4 bits of $DATA_B$. After the bit is set, the 16-bit result is output as $DATA_X$. $DATA_Y$ is always output as zero. The control bits and the sign bits of $DATA_X$ and $DATA_Y$ are as follows: $C_X \leftarrow C_A$, $C_Y \leftarrow C_B$, $S_X \leftarrow S_A$, $S_Y \leftarrow 0$.

CLR1 [Clear one bit]: This instruction is used to reset a particular bit of $DATA_A$. The bit of $DATA_A$ to be reset is specified by the lower 4 bits of $DATA_B$. After the bit is reset (cleared), the 16-bit result is output as $DATA_X$. $DATA_Y$ is always output as zero. The control bits and the sign bits of $DATA_X$ and $DATA_Y$ are as follows: $C_X \leftarrow C_A$, $C_Y \leftarrow C_B$, $S_X \leftarrow S_A$, $S_Y \leftarrow 0$.

Table 18. Bit Addressing

$DATA_B$ Bit				
3	2	1	0	$DATA_A$ Bit Position
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11
1	1	0	0	12
1	1	0	1	13
1	1	1	0	14
1	1	1	1	15

Bit Check Instructions

ANDMSK [Mask a word with logical AND]: This instruction tests certain bits in $DATA_A$. The bits in $DATA_A$ to be tested are first masked with a bit pattern in $DATA_B$. Only those bits in $DATA_A$ corresponding to the one bits of $DATA_B$ are considered. Then only those masked bits of $DATA_A$ are ANDed together to set or reset the control bits, C_X and C_Y. If the result of the AND operation is 1, then both the C_X and C_Y are set to 1. If the result of the operation is 0, then the both C_X and C_Y are set to 0. The rest of the output data fields are the following: $S_X \leftarrow S_A$, $S_Y \leftarrow S_B$, $DATA_X \leftarrow DATA_A$, $DATA_Y \leftarrow DATA_B$.

ORMSK [Mask a word with logical OR]: This instruction tests certain bits in $DATA_A$. The bits in $DATA_A$ to be tested are first masked with a bit pattern in $DATA_B$. Only those bits in $DATA_A$ corresponding to the one bits of $DATA_B$ are considered. Then only those masked bits of $DATA_A$ are ORed together to set or reset the control bits, C_X and C_Y. If the result of the OR operation is 1, then both C_X and C_Y are set to 1. If the result of the operation is 0, then the both C_X and C_Y are set to 0. The rest of the output data fields are the following: $S_X \leftarrow S_A$, $S_Y \leftarrow S_B$, $DATA_X \leftarrow DATA_A$, $DATA_Y \leftarrow DATA_B$.

Data Conversion Instructions

CVT2AB [Convert two's complement to sign-magnitude]: This instruction converts a 16-bit number in two's complement form to a 17-bit number in sign-magnitude form. The sign of the two's complement number is output as the S_X bit.

CVTAB2 [Convert sign-magnitude to two's complement]: This instruction converts a 17-bit number in sign-magnitude form to a 16-bit number in two's complement form. This operation has the potential danger of an overflow or an underflow. If an overflow or an underflow occurs, the C_X bit is set to 1.

Double Precision Adjustment Instruction

ADJL [Adjust long]: This instruction is used to adjust a double precision number, in which the sign bits of the upper and lower words are different. This situation may occur after a double precision arithmetic operation. The examples in table 19 illustrate the adjustments of double precision numbers.

Table 19. Double Precision Adjustment Examples

	Input/Output	Sign	Data
Input	High (A data)	0	1234H
	Low (B data)	0	5678H
Output	High (X data)	0	1234H
	Low (Y data)	0	5678H
Input	High (A data)	0	1234H
	Low (B data)	1	5678H
Output	High (X data)	0	1233H
	Low (Y data)	0	A988H
Input	High (A data)	1	1234H
	Low (B data)	0	5678H
Output	High (X data)	1	1233H
	Low (Y data)	1	A988H

Accumulative Addition Instruction

ACC [Accumulate]: This instruction (see figure 22) performs cumulative additions of incoming tokens' data fields. The incoming tokens are classified into type 1 and type 2 tokens. A type 1 token is deleted after the ACC operation, but a type 2 token is not. Moreover, a type 2 token reads the contents of the ACC register, which contains the accumulated sum of tokens. When a type 2 token reads the contents of the ACC register, the ID field of the token is unchanged. However, if an overflow has occurred prior to the arrival of a type 2 token, the ID field is incremented by one. Only the following three tokens qualify as type 2 tokens.

1. If the ACC instruction is used along with RDCYCS instruction, and the token's FTRC bit = 1, and the Buffer Size and Read Counter of RDCYCS instruction are equal.
2. If the ACC instruction is used along with RDCYCL instruction, and the token's FTRC bit = 1, and the Buffer size and Read Counter of RDCYCL instruction are equal.
3. If the ACC instruction is used along with COUNT instruction, and the token's FTRC bit = 0, and the Count Size and Counter of COUNT instruction are equal.

C Bit Copy Instruction

COPYC [Copy control bit]: This instruction copies the control bit of the A side and outputs it as C_Y.

$C_X \leftarrow C_A$, $S_X \leftarrow S_A$, $DATA_X \leftarrow DATA_A$, $C_Y \leftarrow C_A$, $S_Y \leftarrow S_B$, $DATA_Y \leftarrow DATA_B$.

Figure 22. ACC Instruction

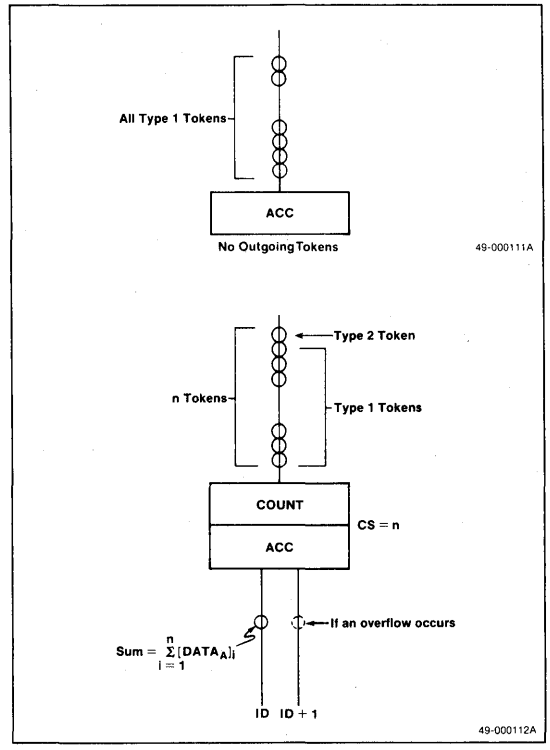


Table 20. PU Instruction (Sheet 1 of 3)

Mnemonic	OP Code	Input C_A	S_A	$DATA_A$	C_B	S_B	$DATA_B$	Output C_X	S_X	$DATA_X$	C_Y	SY	$DATA_Y$	Notes
Logical Operations														
OR	00000	C_A	S_A	A	C_B	S_B	B	C_X	S_A	A OR B	C_Y	0	0000H	
AND	00001	C_A	S_A	A	C_B	S_B	B	C_X	S_A	A AND B	C_Y	0	0000H	
XOR	00010	C_A	S_A	A	C_B	S_B	B	C_X	S_A	A XOR B	C_Y	0	0000H	
ANDNOT	00011	C_A	S_A	A	C_B	S_B	B	C_X	S_A	$\overline{A}$ AND B	C_Y	0	0000H	
NOT	01100	C_A	S_A	A				C_X	S_A	$\overline{A}$	C_Y	0	0000H	
Arithmetic Operations														
ADD	11000	C_A	0	A	C_B	0	B	C_X	0	A + B	C_Y	0	*	
		C_A	0	A	C_B	1	B	C_X	0	A − B	C_Y	0	0000H	When A ≥ B, S_X = 0
								C_X	1	B − A	C_Y	1	0000H	When A < B, S_X = 1
		C_A	1	A	C_B	0	B	C_X	0	B − A	C_Y	0	0000H	When A < B, S_X = 0
								C_X	1	A − B	C_Y	1	0000H	When A ≥ B, S_X = 1
		C_A	1	A	C_B	1	B	C_X	1	A + B	C_Y	1	*	
ADDSC	11100	C_A	0	A	C_B	0	B	C_X	0	A + B	C_Y	S_S	No. of shifts †	
		C_A	0	A	C_B	1	B	C_X	0	A − B	C_Y	S_S	*	When A ≥ B, S_X = 0
								C_X	1	B − A	C_Y	S_S	No. of shifts †	When A < B, S_X = 1
		C_A	1	A	C_B	0	B	C_X	0	B − A	C_Y	S_S	*	When A < B, S_X = 0
								C_X	1	A − B	C_Y	S_S	No. of shifts †	When A ≥ B, S_X = 1
		C_A	1	A	C_B	1	B	C_X	1	A + B	C_Y	S_S	No. of shifts†	
SUB	11001	C_A	0	A	C_B	0	B	C_X	0	A − B	C_Y	0	0000H	When A > B, S_X = 0
								C_X	1	B − A	C_Y	1	0000H	When A < B, S_X = 1
		C_A	0	A	C_B	1	B	C_X	0	A + B	C_Y	0	*	
		C_A	1	A	C_B	0	B	C_X	1	A + B	C_Y	1	*	
		C_A	1	A	C_B	1	B	C_X	0	B − A	C_Y	0	0000H	When A < B, S_X = 0
								C_X	1	A − B	C_Y	1	0000H	When A ≥ B, S_X = 1
SUBSC	11101	C_A	0	A	C_B	0	B	C_X	0	A − B	C_Y	S_S	No. of shifts †	When A ≥ B, S_X = 0
								C_X	1	B − A	C_Y	S_S	No. of shifts †	When A < B, S_X = 1
		C_A	0	A	C_B	1	B	C_X	0	A + B	C_Y	S_S	No. of shifts †	
		C_A	1	A	C_B	0	B	C_X	1	A + B	C_Y	S_S	No. of shifts †	
		C_A	1	A	C_B	1	B	C_X	0	B − A	C_Y	S_S	No. of shifts †	When A < B, S_X = 0
								C_X	1	A − B	C_Y	S_S	No of shifts †	When A ≥ B, S_X = 1
MUL	11010	C_A	S_A	A	C_B	S_B	B	C_X	S_X	A x B High	C_Y	S_X	A x B Low	$S_X = S_A$ OR S_B (logical OR)
MULSC	11110	C_A	S_A	A	C_B	S_B	B	C_X	S_X	A x B High	C_Y	S_S	No. of shifts †	$S_X = S_A$ OR S_B (logical OR)

Table 20. PU Instruction (Sheet 2 of 3)

Mnemonic	OP code	Input						Output						Notes
		C_A	S_A	$DATA_A$	C_B	S_B	$DATA_B$	C_X	S_X	$DATA_X$	C_Y	SY	$DATA_Y$	
Arithmetic Operations														
NOP	11011	C_A	S_A	A	C_B	S_B	B	C_X	S_A	A	C_Y	S_B	B	
NOPSC	11111	C_A	S_A	A	C_B	S_B	B	C_X	S_A	A	C_Y	S_S	No. of shifts †	
INC	01010	C_A	0	A				C_X	0	A + 1	C_Y	0	*	
		C_A	1	A				C_X	0	1	C_Y	0	0000H	When A = 0, S_X = 0
								C_X	1	A − 1	C_Y	1	0000H	When A ≥ 1, S_X = 1
DEC	01011	C_A	0	A				C_X	0	A − 1	C_Y	0	0000H	When A ≥ 0, S_X = 0
								C_X	1	1	C_Y	1	0000H	When A = 0, S_X = 1
		C_A	1	A				C_X	1	A + 1	C_Y	1	*	
Shift														
SHL	00100	C_A	S_A	A	C_B	0	No. of shifts	C_X	S_A	Shift A left	C_Y	S_A	Shift A left	
		C_A	S_A	A	C_B	1	No. of shifts	C_X	S_A	Shift A right	C_Y	S_A	Shift A right	
SHLBRV	00101	C_A	S_A	A	C_B	0	No. of shifts	C_X	S_A	Reverse A and shift left	C_Y	S_A	Reverse A and shift left	
		C_A	S_A	A	C_B	1	No. of shifts	C_X	S_A	Reverse A and shift right	C_Y	S_A	Reverse A and shift right	
SHR	00110	C_A	S_A	A	C_B	0	No. of shifts	C_X	S_A	Shift A right	C_Y	S_A	Shift A right	
		C_A	S_A	A	C_B	1	No. of shifts	C_X	S_A	Shift A left	C_Y	S_A	Shift A left	
SHRBRV	00111	C_A	S_A	A	C_B	0	No. of shifts	C_X	S_A	Reverse A and shift right	C_Y	S_A	Reverse A and shift right	
		C_A	S_A	A	C_B	1	No. of shifts	C_X	S_A	Reverse A and shift left	C_Y	S_A	Reverse A and shift left	
Comparison														
CMPNOM	01000	C_A	S_A	A	C_B	S_B	B	0	0	0000H	0	0	0000H	When PNZ is false
		C_A	S_A	A	C_B	S_B	B	1	0	0001H	1	0	0000H	When PNZ is true
CMP	01001	C_A	S_A	A	C_B	S_B	B	0	S_A	A	0	S_B	B	When PNZ is false
		C_A	S_A	A	C_B	S_B	B	1	S_A	A	1	S_B	B	When PNZ is true
CMPXCH	10001	C_A	S_A	A	C_B	S_B	B	C_A	S_A	A	C_B	S_B	B	When PNZ is true
		C_A	S_A	A	C_B	S_B	B	C_B	S_B	B	C_A	S_A	A	When PNZ is false
Accumulative Addition														
ACC	10010	C_A	S_A	A	C_B	S_B	B	C_X	S_X	ΣA				Used as a pair with AG & FC instruction COUNT
C Bit Copy														
COPYC	10011	C_A	S_A	A	C_B	S_B	B	C_A	S_A	A	C_A	S_B	B	

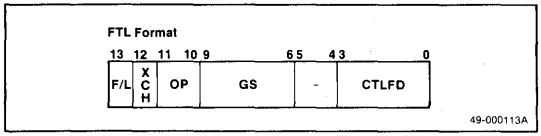
Table 20. PU Instruction (Sheet 3 of 3)

		Input						Output						
Mnemonic	**OP code**	**C_A**	**S_A**	**$DATA_A$**	**C_B**	**S_B**	**$DATA_B$**	**C_X**	**S_X**	**$DATA_X$**	**C_Y**	**SY**	**$DATA_Y$**	**Notes**
Bit Operations														
GET1	10101	C_A	S_A	A	C_B	S_B	Bit position	C_X	S_A	0000H	C_Y	0	0000H	When the bit specified by the lower 4 bits of $DATA_B$ is 0
		C_A	S_A	A	C_B	S_B	Bit position	C_X	S_A	0001H	C_Y	0	0000H	When the bit specified by the lower 4 bits of $DATA_B$ is 1
SET1	10110	C_A	S_A	A	C_B	S_B	Bit position	C_X	S_A	A bit in $DATA_A$ is set	C_Y	0	0000H	Bit specification by the lower 4 bits of $DATA_B$
CLR1	10111	C_A	S_A	A	C_B	S_B	Bit position	C_X	S_A	A bit in $DATA_A$ is cleared	C_Y	0	0000H	Bit specification by the lower 4 bits of $DATA_B$
Bit Check														
ANDMSK	01101	C_A	S_A	A	C_B	S_B	B	0	S_A	A	0	S_B	B	If ANDMSK = 0
		C_A	S_A	A	C_B	S_B	B	1	S_A	A	1	S_B	B	If ANDMSK = 1
ORMSK	10000	C_A	S_A	A	C_B	S_B	B	0	S_A	A	0	S_B	B	If ORMSK = 0
		C_A	S_A	A	C_B	S_B	B	1	S_A	A	1	S_B	B	If ORMSK = 1
Data Conversion														
CVT2AB	01110	C_A	S_A	A	C_B	S_B	B	C_X	S_X	Converted A data	C_Y	0	0000H	Absolute value ← twos complement
CVTAB2	01111	C_A	S_A	A	C_B	S_B	B	C_X	S_X	Converted A data	C_Y	0	0000H	Twos complement ← absolute value
Adjustment of Double Precision Numbers														
ADJL	10100	C_A	0	A	C_B	1	B	C_X	0	A − 1	C_Y	0	0000H-B	A ≠ 0 AND B ≠ 0
		C_A	1	A	C_B	0	B	C_X	1	A − 1	C_Y	1	0000H-B	A ≠ 0 AND B ≠ 0
		C_A	0	A	C_B	1	0000H	C_X	0	A	C_Y	0	0000H	
		C_A	0	0000H	C_B	1	B	C_X	1	0000H	C_Y	1	B	B ≠ 0
		C_A	1	A	C_B	0	0000H	C_X	1	A	C_Y	1	0000H	
		C_A	1	0000H	C_B	0	B	C_X	0	0000H	C_Y	0	B	B ≠ 0
		C_A	0	A	C_B	0	B	C_X	0	A	C_Y	0	B	
		C_A	1	A	C_B	1	B	C_X	1	A	C_Y	1	B	

Notes: * If an overflow occurs as the result of A + B, $DATA_Y$ = 0001H and if no overflow, $DATA_Y$ = 0000H.

† This indicates the number of consecutive zeros from the MSB of $DATA_X$. This number is used to calculate the number of shifts to be performed by subsequent processing.

GE Instructions



Bit Assignments

F/L [Full/Left]: F/L bit = 0 indicates that the GE instruction is used alone, whereas F/L bit = 1 indicates that the GE instruction is used in conjunction with an AG/FC instruction.

XCH [Exchange]: XCH bit = 1 indicates that the data from A side and B side are to be exchanged before the two data tokens enter the Queue.

OP [OP code]: These two bits select an operation to be performed. See table 21.

Table 21. OP Bits

OP	Operation
00	COPYBK (Copy block)
01	COPYM (Copy multiple)
11	SETCTL (Set control field)

GS [Generation Size]: These four bits determine the number of copies of a token to be made. A minimum of 2 and a maximum of 17 copies can be made using a GE instruction.

CTLFD [Control Field]: This field is used with Set Control Field (SETCTL) instruction. The data in CTLFD field further specifies the types of operations to be performed by the SETCTL instruction.

COPYBK [Copy Block]

COPYBK is used to duplicate a block of tokens from a single token. These duplicated tokens have exactly the same ID as the original token except the token copied last which has the original token's ID plus one. The number of tokens to be generated is specified by the GS field, and the COPYBK instruction generates exactly GS + 2 tokens. The data fields of the tokens being duplicated can also be incremented or decremented in a systematic manner. The incremental (or decremental) step value is contained in $DATA_B$. The tokens generated are sent to the Link Table. The series of LT tokens output by the instruction is shown in figure 23.

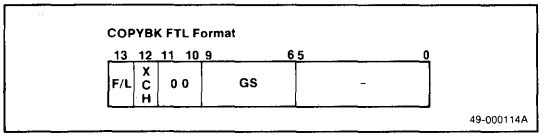
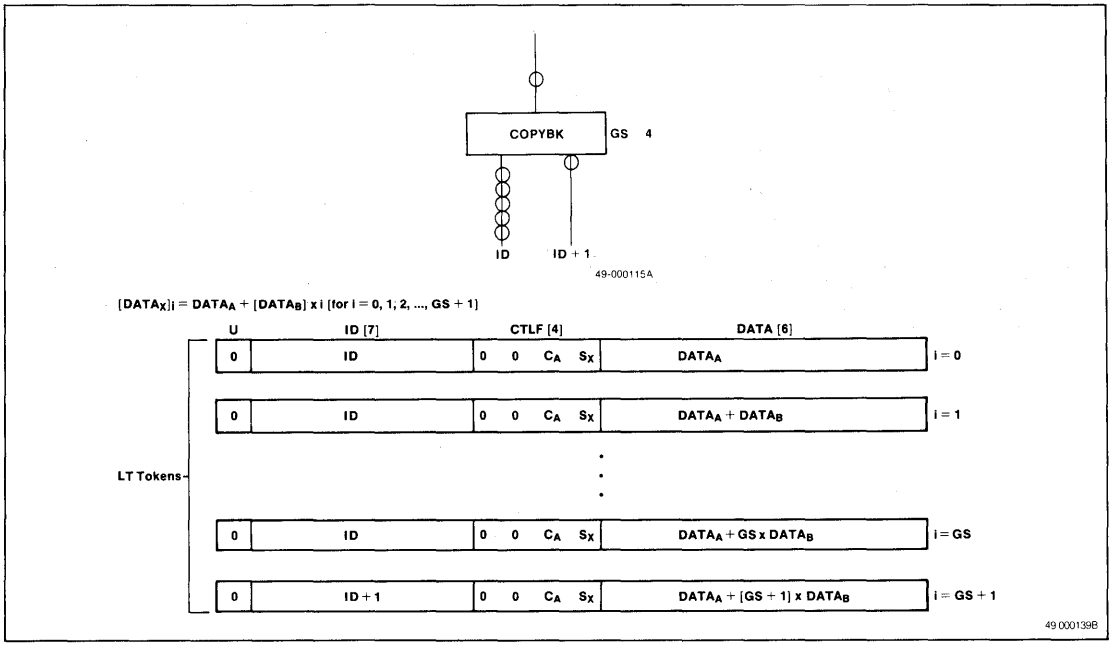


Figure 23. COPYBK Instruction Output



COPYM [Copy Multiple]

COPYM is used to generate multiple tokens from a single token. Each generated token has a different ID value. The number of tokens generated from the original token is GS + 2. The data field of the tokens being generated can also be incremented or decremented in a systematic manner. The incremental (or decremental) step value is contained in $DATA_B$. The generated tokens are sent to the Link Table as LT tokens. The series of LT tokens output by the COPYM instruction is shown in figure 24.

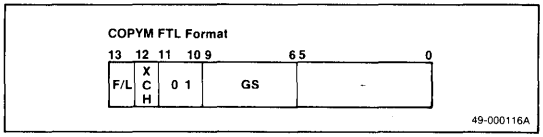
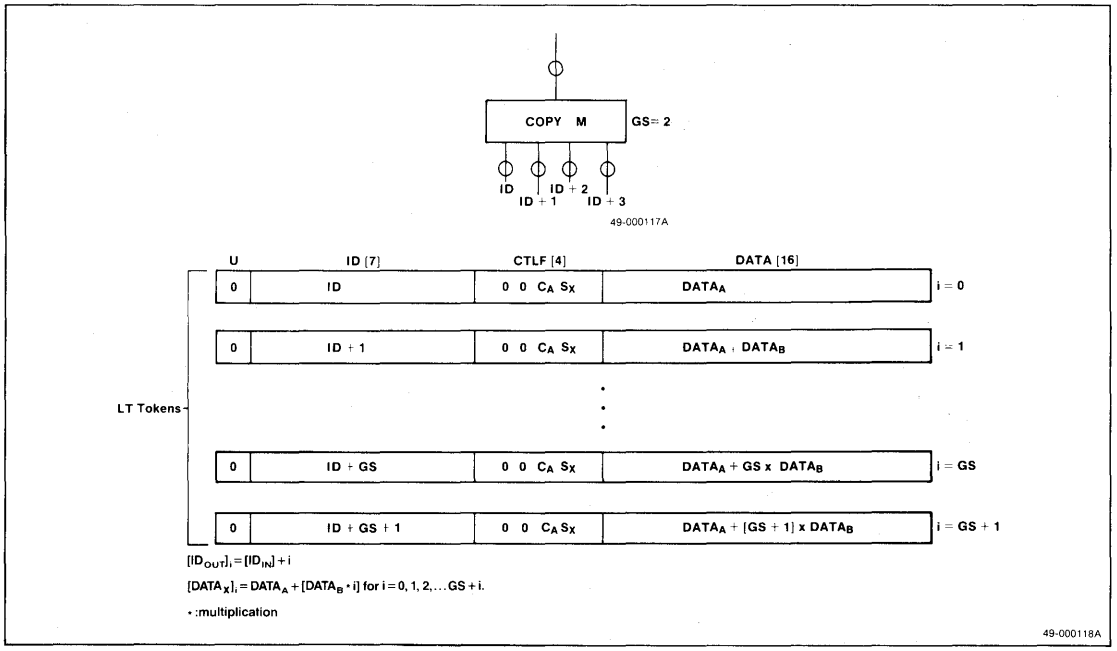
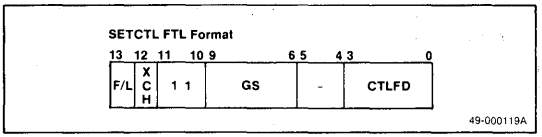


Figure 24. COPYM Instruction Output Tokens



SETCTL [Set Control Field]



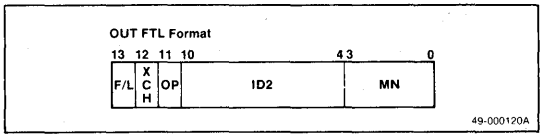
SETCTL is used to read and rewrite the contents of the Link Table and the Function Table. Since it can change the contents of the Link Table and the Function Table, this instruction can be used to write a self-modifying code. The type of operation to be performed is further specified by the contents of CTLFD field, as shown in table 22.

Table 22. SETCTL Instruction Control Field Operation

CTLFD				Operation
0	0	C	S	Normal data. Operation is exactly the same as COPYM.
1	1	0	0	The data field of this token is used to set a location in the Link Table memory (C and S bits are not included.) After the data is set, the token is deleted.
1	1	0	1	The data field of this token is used to set a location in the Function Table Right field. After the data is set, the token is deleted.
1	1	1	0	The lower 14 bits of the data field of this token are used to set a location in the Function Table Left field (higher bits are ignored.) After the data is set, the token is deleted.
1	1	1	1	The lower 10 bits of the data field of this token are used to set a location in the Function Table Temporary field (higher bits are ignored.) After the data is set, the token is deleted.
1	0	0	0	This token reads the LT address indicated by the ID field and outputs the contents.
1	0	0	1	This token reads the Function Table Right field address indicated by the ID field and outputs the contents.
1	0	1	0	This token reads the Function Table Left field address indicated by the ID field and outputs the contents.
1	0	1	1	This token reads the Function Table Temporary field address indicated by the ID field and outputs the contents.
0	1	0	0	These tokens should not be generated by the Processing
0	1	0	1	Unit. They are operating-mode-related tokens.
0	1	1	0	
0	1	1	1	

Note: The set or write operation is performed at the address indicated by the ID field of the token.

OUT Instructions



Bit Assignments

F/L [Full/Left]: F/L bit = 0 indicates that the OUT instruction is to be used alone. F/L bit = 1 indicates that the OUT instruction is to be used in conjunction with an AG/FC instruction.

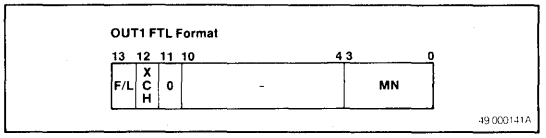
XCH [Exchange]: If XCH bit = 1, the output data tokens from the A side are exchanged with those from the B side before they go to the Output Queue. If XCH bit = 0, no exchange operation is performed.

OP [OP Code]: This bit is used to further specify the OUT instruction. If OP = 0, then OUT1 instruction is performed, whereas if OP = 1, OUT2 instruction is performed.

ID2 [Second ID]: This field is used only by the OUT2 instruction. ID2 is the ID of the second output data token.

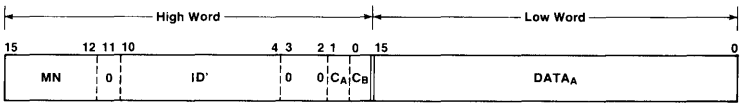
MN [Module Number]: This field indicates the destination module of the output data token.

OUT1



This instruction outputs a 32-bit data token via the Output Data Bus (ODB). Since the size of the ODB is 16 bits, a 32-bit output data token is divided into two 16-bit words and output one 16-bit word at a time. The format of an output data token is shown in figure 24.

Figure 25. OUT1 Output Token Format



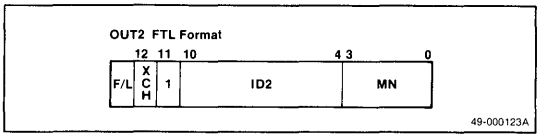
MN: Determined by the lower 4 bits of FTL contents
ID': 7-bit ID comes from the contents of LT referenced by the OUT1 instruction
C_A, S_A: Control bit and sign bit of $DATA_A$
$DATA_A$: 16-bit output data

49-000121A

2

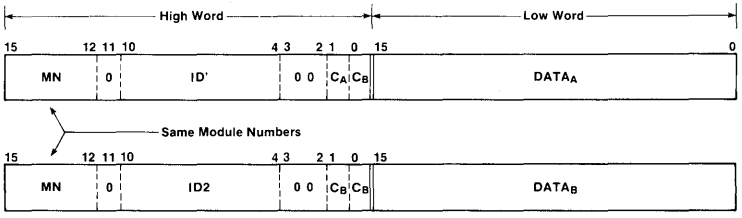
OUT2

This instruction outputs two 32-bit data tokens via ODB. Since the ODB is 16 bits wide, each 32-bit token is divided into two 16-bit words and output one 16-bit word at a time. This instruction is useful when a double precision number is to be output. The formats of two output data tokens are shown in figure 25.



49-000123A

Figure 26. OUT2 Output Tokens Format



Note: First and second tokens must have the same module numbers

MN:	Determined by the lower four bits of the FTL contents.
ID':	7-bit ID coming from the contents of the Link Table referenced by the OUT2 instruction.
ID2:	7-bit ID comes from the FTL field of the OUT2 instruction.
C_A, S_A:	Control bit and sign bit of $DATA_A$.
$DATA_A$:	First 16-bit output data.
C_B, S_B:	Control bit and sign bit of $DATA_B$.
$DATA_B$:	Second 16-bit output data.

49-000122A

Figure 27. Data-Flow Graph Explanation

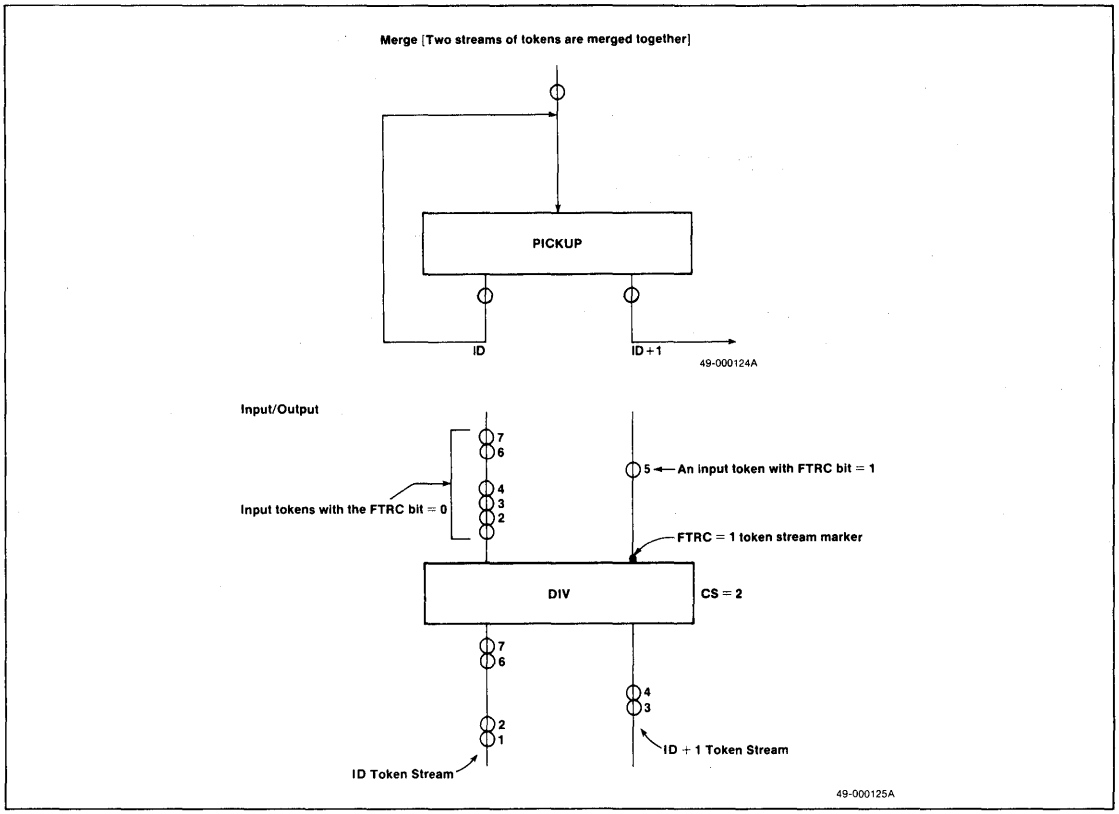
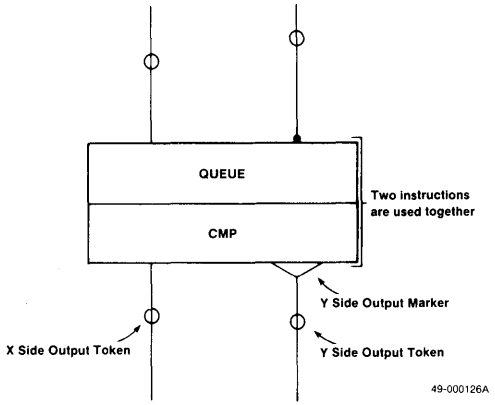
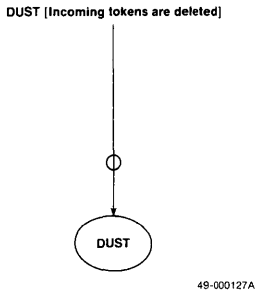
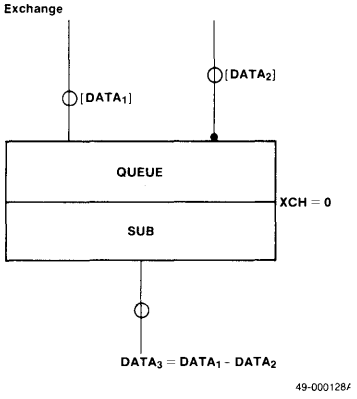
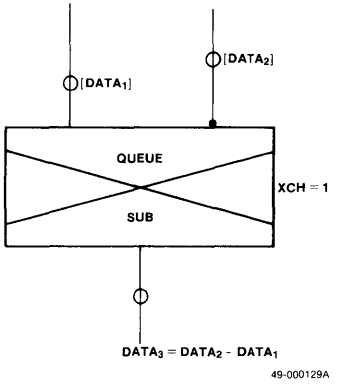


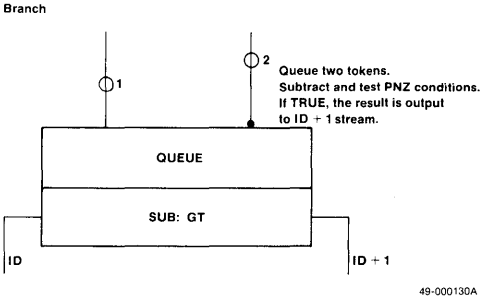
Figure 27. Data-Flow Graph Explanation (cont)

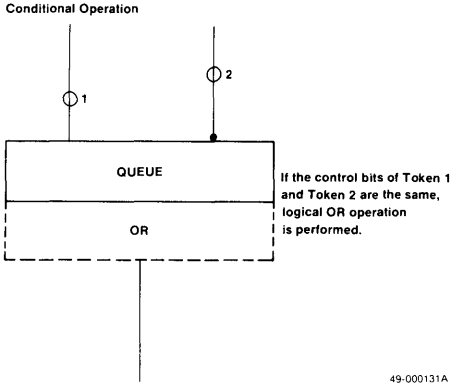












NEC

NEC Electronics Inc.

μPD9305 MEMORY ACCESS AND GENERAL BUS INTERFACE FOR THE μPD7281

PRELIMINARY INFORMATION

Description

The NEC μPD9305 memory access and general bus interface chip (MAGIC) is a peripheral LSI support device for the μPD7281 image pipelined processor (ImPP). The μPD7281 is a data flow architecture processor that supports high speed image and signal processing applications. The μPD9305 chip can support from one to eight μPD7281s and also interfaces to both 8-bit and 16-bit host processors.

The μPD9305's powerful interface capabilities allow it to support basic interface operations, object program load, read/write/modify operations on image memory, and multiple μPD7281 image memory accesses.

Since the μPD7281 ImPP does not use direct addressing, the memories in a μPD7281 processor system can be seen as processing modules with unique module numbers. These separate modules must output memory access tokens containing their own unique address, data, and control signals. The modules must perform the necessary processing, and then output the result of the access as another memory access token. To do this, the multiple μPD7281 modules require external circuitry to process the memory access tokens that they output. In addition, this same circuitry is required to organize the data output from the memory into token format.

Circuitry is also needed between the host processor and the μPD7281s to organize the data from the host into token format and to return the data output from the μPD7281s into the form required by the host processor. Finally, tokens may have to be returned to other μPD7281s in token form for further processing.

The μPD9305 simplifies the above operations by keeping the data in the most convenient form. The μPD9305 replaces approximately 80 medium/small scale integrated devices with a single integrated circuit.

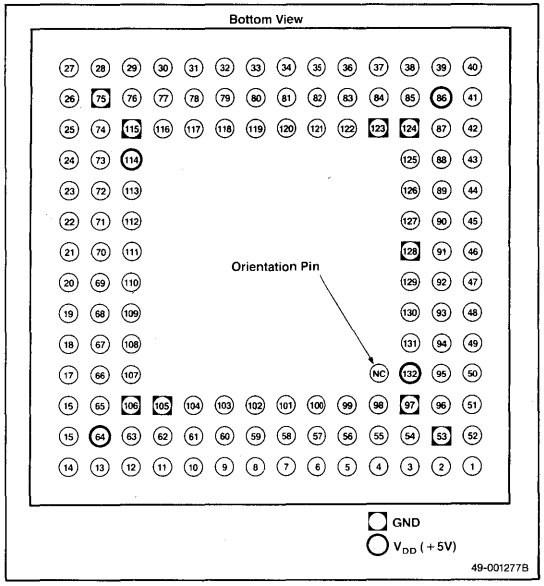
Features

- ☐ High performance image memory interface
- ☐ Reduces external circuits required for ImPP system
- ☐ Simplifies host interface
- ☐ Up to 24-bit image memory addressing
- ☐ Up to 18-bit image memory data
- ☐ Register file for memory access
- ☐ Refresh control of image memory
- ☐ Functions with separate DMA controller
- ☐ Single +5 V power supply
- ☐ CMOS technology for lower power consumption

Ordering Information

Part Number	Package Type
μPD9305R	132-pin ceramic grid array

Pin Configuration



Pin Identification

No.	Symbol	Function
1	CLK	Clock input
2-4	D_{10},D_{12}, D_{15}	Bidirectional data bus bits
5	$\overline{OACK}$	Output acknowledge input
6	$\overline{OREQ}$	Output request output
7	IDB_{14}	Input data bus bit
8	ODB_{14}	Output data bus bit
9	IDB_{11}	Input data bus bit
10, 11	ODB_{11}, ODB_8	Output data bus bits
12	IDB_9	Input data bus bit
13	ODB_5	Output data bus bit
14	IDB_8	Input data bus bit
15	ODB_4	Output data bus bit
16	IDB_7	Input data bus bit
17	ODB_2	Output data bus bit

Pin Identification (Cont)

No.	Symbol	Function
18	IDB_6	Input data bus bit
19	MN_2	Module number output
20	IDB_4	Input data bus bit
21	IMA_{22}	Image memory address output bit
22	IDB_2	Input data bus bit
23, 24	IMA_{18}, IMA_{15}	Image memory address output bits
25	IDB_0	Input data bus bit
26-28	IMA_{12}-IMA_{10}	Image memory address output bits
29	SOLBSY	Self object load busy output
30	CPURQ	CPU request output
31	DMAAEN	DMA address enable input
32-34	IMA_5,IMA_2, IMA_0	Image memory address output bits
35	$\overline{DMAAK1}$	DMA / 1 acknowledge input
36	$\overline{DMARQ1}$	DMA / 1 request output
37	IMD_{13}	Bidirectional image memory data bus bit
38	$\overline{IMAK}$	Image memory acknowledge input
39-42	IMD_{10}-IMD_7	Bidirectional image memory data bus bits
43	A_0	Address select input
44,45	IMD_3,IMD_1	Bidirectional image memory data bus bits
46	IMWR	Image memory write output
47	$\overline{WR}$	Write input
48,49	D_2,D_5	Bidirectional data bus bits
50	$\overline{CS}$	Chip select input
51,52	D_8,D_9	Bidirectional data bus bits
53	GND	Ground
54,55	D_{11},D_{14}	Bidirectional data bus bits
56	$\overline{IREQ}$	Input request input
57	$\overline{IACK}$	Input acknowledge output
58	IDB_{13}	Input data bus bit
59	ODB_{13}	Output data bus bit
60	IDB_{10}	Input data bus bit
61-63	ODB_{10},ODB_7, ODB_6	Output data bus bits
64	V_{DD}	+5 V power supply
65,66	ODB_3,ODB_1	Output data bus bits
67	IDB_5	Input data bus bit
68	MN_1	Module number output bit

Pin Indentification (Cont)

No.	Symbol	Function
69,70	IMA_{23},IMA_{21}	Image memory address output bits
71	IDB_1	Input data bus bit
72-74	IMA_{17},IMA_{14}, IMA_{13}	Image memory address output bits
75	GND	Ground
76,77	IMA_9,IMA_8	Image memory address output bits
78	INBUSY	Input to ImPP busy output
79, 80	IMA_4,IMA_1	Image memory address output bits
81	IMD_{17}	Bidirectional image memory data bus bit
82	$\overline{DMAAK2}$	DMA / 2 acknowledge input
83	$\overline{DMARQ2}$	DMA / 2 request output
84, 85	IMD_{12},IMD_{11}	Bidirectional image memory data bus bits
86	V_{DD}	+5 V power supply
87,88	IMD_6,IMD_5	Bidirectional image memory data bus bits
89	A_1	Address select input
90	IMD_0	Bidirectional image memory data bus bit
91	IMRF	Image memory refresh output
92	D_0	Bidirectional data bus bit
93	$\overline{RD}$	Read input
94-96	D_4,D_6,D_7	Bidirectional data bus bits
97	GND	Ground
98	D_{13}	Bidirectional data bus bit
99	$\overline{IPPRST}$	Image pipelined processor reset output
100	IDB_{15}	Input data bus bit
101	ODB_{15}	Output data bus bit
102	IDB_{12}	Input data bus bit
103,104	ODB_{12},ODB_9	Output data bus bits
105,106	GND	Ground
107	ODB_0	Output data bus bit
108,109	MN_3,MN_0	Module number output bits
110	IDB_3	Input data bus bit
111-113	IMA_{20},IMA_{19}, IMA_{16}	Image memory address outputs
114	V_{DD}	+5 V power supply
115	GND	Ground
116-118	IMA_7,IMA_6, IMA_3	Image memory address outputs

Pin Identification (Cont)

No.	Symbol	Function
119	$\overline{RESET}$	Reset input
120-122	IMD_{16}-IMD_{14}	Bidirectional image memory data bus bits
123,124	GND	Ground
125,126	IMD_4,IMD_2	Bidirectional image memory data bus bits
127	IMRD	Image memory read output
128	GND	Ground
129	ERR	Error output
130,131	D_1,D_3	Bidirectional data bus bits
132	V_{DD}	+5 V power supply

Pin Functions

Table 1 shows the μPD9305 pins in their particular functional groups. The paragraphs that follow table 1 describe the operation of the pins in each group.

All unused input or output pins should be pulled up to V_{DD} or down to GND through a 2K-3K ohm resistor.

Table 1. μPD9305 Pins by Function

I/O	Signal	No.
I	CLK	1
	$\overline{RESET}$	119
	Status	
O	ERR	129
	SOLBSY	29
	CPURQ	30
	INBUSY	78
	Host Interface	
I	$\overline{WR}$	47
	$\overline{RD}$	93
	$\overline{CS}$	50
	A_0	43
	A_1	89
I/O	D_0	92
	D_1	130
	D_2	48
	D_3	131
	D_4	94
	D_5	49
	D_6	95
	D_7	96
	D_8	51
	D_9	52
	D_{10}	2
	D_{11}	54
	D_{12}	3
	D_{13}	98
	D_{14}	55
	D_{15}	4
	DMA	
O	$\overline{DMARQ1}$	36
	$\overline{DMARQ2}$	83
I	$\overline{DMAAK1}$	35
	$\overline{DMAAK2}$	82
	DMAAEN	31

Table 1. μPD9305 Pins by Function (Cont)

I/O	Signal	No.
	μPD7281 Interface	
	MN_0	109
O	MN_1	68
	MN_2	19
	MN_3	108
O	$\overline{OREQ}$	6
I	$\overline{OACK}$	5
	$\overline{IREQ}$	56
O	$\overline{IACK}$	57
	$\overline{IPPRST}$	99
	ODB_0	107
	ODB_1	66
	ODB_2	17
	ODB_3	65
	ODB_4	15
	ODB_5	13
	ODB_6	63
O	ODB_7	62
	ODB_8	11
	ODB_9	104
	ODB_{10}	61
	ODB_{11}	10
	ODB_{12}	103
	ODB_{13}	59
	ODB_{14}	8
	ODB_{15}	101
	IDB_0	25
	IDB_1	71
	IDB_2	22
	IDB_3	110
	IDB_4	20
	IDB_5	67
	IDB_6	18
I	IDB_7	16
	IDB_8	11
	IDB_9	12
	IDB_{10}	60
	IDB_{11}	9
	IDB_{12}	102
	IDB_{14}	7
	IDB_{15}	100

Table 1. μPD9305 Pins by Function (Cont)

I/O	Signal	No.
	Image Memory Interface	
I	$\overline{IMAK}$	38
	IMRD	127
O	IMWR	46
	IMRF	91
	IMD_0	90
	IMD_1	45
	IMD_2	126
	IMD_3	44
	IMD_4	125
	IMD_5	88
	IMD_6	87
	IMD_7	42
I/O	IMD_8	41
	IMD_9	40
	IMD_{10}	39
	IMD_{11}	85
	IMD_{12}	84
	IMD_{13}	37
	IMD_{14}	122
	IMD_{15}	121
	IMD_{16}	120
	IMD_{17}	81

Table 1. μPD9305 Pins by Function (Cont)

I/O	Signal	No.
	Image Memory Interface	
	IMA_0	34
	IMA_1	80
	IMA_2	33
	IMA_3	118
	IMA_4	79
	IMA_5	32
	IMA_6	117
	IMA_7	116
	IMA_8	77
	IMA_9	76
	IMA_{10}	28
0	IMA_{11}	27
	IMA_{12}	26
	IMA_{13}	74
	IMA_{14}	73
	IMA_{15}	24
	IMA_{16}	113
	IMA_{17}	72
	IMA_{18}	23
	IMA_{19}	112
	IMA_{20}	111
	IMA_{21}	70
	IMA_{22}	21
	IMA_{23}	69

CLK (Clock)

CLK is the single phase master clock input. The μPD9305 clock frequency can be independent of ImPP clock frequency.

$\overline{\text{RESET}}$ (Reset)

$\overline{\text{RESET}}$ initializes the μPD9305. A reset places $\overline{\text{OREQ}}$, $\overline{\text{IACK}}$, the token I/O flip-flop, and IM access request signals at an inactive level. $\overline{\text{RESET}}$ resets the refresh address counter, refresh timer counter, and mode register to 0. $\overline{\text{RESET}}$ must be held low for a minimum of four μPD9305 or μPD7281 clock cycles, whichever is slower.

V_{DD} (Power)

V_{DD} is the single +5 volt power supply.

GND (Ground)

GND is the ground signal.

Status Signal Pin Functions

CPURQ (CPU Request)

CPURQ indicates to the host processor that the μPD9305 is ready to transfer a token to the host.

INBUSY (Input Busy)

INBUSY indicates that tokens are being input to the first ImPP from the μPD9305.

SOLBSY (Self Object Load Busy)

SOLBSY indicates that a self object load is being executed.

ERR (Error)

ERROR indicates that an error was output from the ImPPs, the host has read an invalid output token, or that the host has input a token while INBUSY was active.

Host Interface Signal Pin Functions

$\overline{\text{RD}}$ (Read)

RD reads the contents of the internal registers specified by A_1 and A_0.

$\overline{\text{WR}}$ (Write)

$\overline{\text{WR}}$ writes an input from the data bus to the internal register specified by A_1 and A_0.

$\overline{\text{CS}}$ (Chip Select)

$\overline{\text{CS}}$ enables the $\overline{\text{RD}}$ or $\overline{\text{WR}}$ control signals.

A_0, A_1 (Address)

A_0 and A_1 select the internal register for a read or write operation.

D_0-D_{15} (Data Bus)

The contents of the internal registers are read from or written to via data bus bits D_0-D_{15}.

DMA Signal Pin Functions

DMAAEN (Direct Memory Access Address Enable)

DMAAEN is used to indicate to the μPD9305 that an external DMA controller is putting DMA addresses on the address bus. During a DMA operation, DMA addresses (system memory addresses) are input to A_0 and A_1. However, these addresses have no meaning for the μPD9305 and might alter register contents. For this reason, the μPD9305 operates as if A_0 and A_1 are both reset to 0 when DMAAEN is active (high).

$\overline{\text{DMARQ1}}$ (Direct Memory Access Request 1)

$\overline{\text{DMARQ1}}$ issues a request to an external DMA controller to transfer data from the host system memory to the μPD9305.

$\overline{\text{DMARQ2}}$ (Direct Memory Access Request 2)

DMARQ2 issues a request to an external DMA controller to transfer data from the μPD9305 to the host system memory.

$\overline{\text{DMAAK1}}$ (Direct Memory Access Acknowledge 1)

$\overline{\text{DMAAK1}}$ is issued by the external DMA controller to indicate to the μPD9305 that $\overline{\text{DMARQ1}}$ has been received.

$\overline{\text{DMAAK2}}$ (Direct Memory Access Acknowledge 2)

$\overline{\text{DMAAK2}}$ is issued by the external DMA controller to indicate to the μPD9305 that $\overline{\text{DMARQ2}}$ has been received.

μPD7281 Interface Signal Pin Functions

MN_0-MN_3 (Module Number)

MN_0-MN_3 specify the module number of one ImPP. During a reset, one module number is output via MN_0-MN_3, the other via IDB_{12}-IDB_{15}. MN_0-MN_3 are three-state pins.

$\overline{\text{OREQ}}$ (Output Request)

$\overline{\text{OREQ}}$ signals to the first ImPP that the μPD9305 is ready to transfer half a token.

$\overline{\text{OACK}}$ (Output Acknowledge)

$\overline{\text{OACK}}$ signals to the μPD9305 that a half token has been accepted by the first ImPP.

$\overline{\text{IREQ}}$ (Input Request)

$\overline{\text{IREQ}}$ signals from the last ImPP that a half token is ready to be transferred from the ImPP to the μPD9305.

$\overline{\text{IACK}}$ (Input Acknowledge)

$\overline{\text{IACK}}$ indicates to the last ImPP that the μPD9305 has accepted the half token.

$\overline{\text{IPPRST}}$ (Image Pipelined Processor Reset)

IPPRST resets the ImPPs during RESET or a command reset.

ODB_0-ODB_{15} (Output Data Bus)

ODB_0-ODB_{15} transfer tokens from the μPD9305 to the first ImPP.

IDB_0-IDB_{15} (Input Data Bus)

IDB_0-IDB_{15} transfer tokens between the output of the last ImPP and the μPD9305.

Image Memory Interface Signal Pin Functions

IMRD (Image Memory Read)

IMRD requests a read of the contents of the image memory addressed by IMA_0-IMA_{23}.

IMWR (Image Memory Write)

IMWR requests a write to the image memory location addressed by IMA_0-IMA_{23}.

IMRF (Image Memory Refresh)

IMRF indicates an image memory refresh cycle.

$\overline{\text{IMAK}}$ (Image Memory Acknowledge)

$\overline{\text{IMAK}}$ indicates to the μPD9305 that an image memory read, write or refresh has been completed.

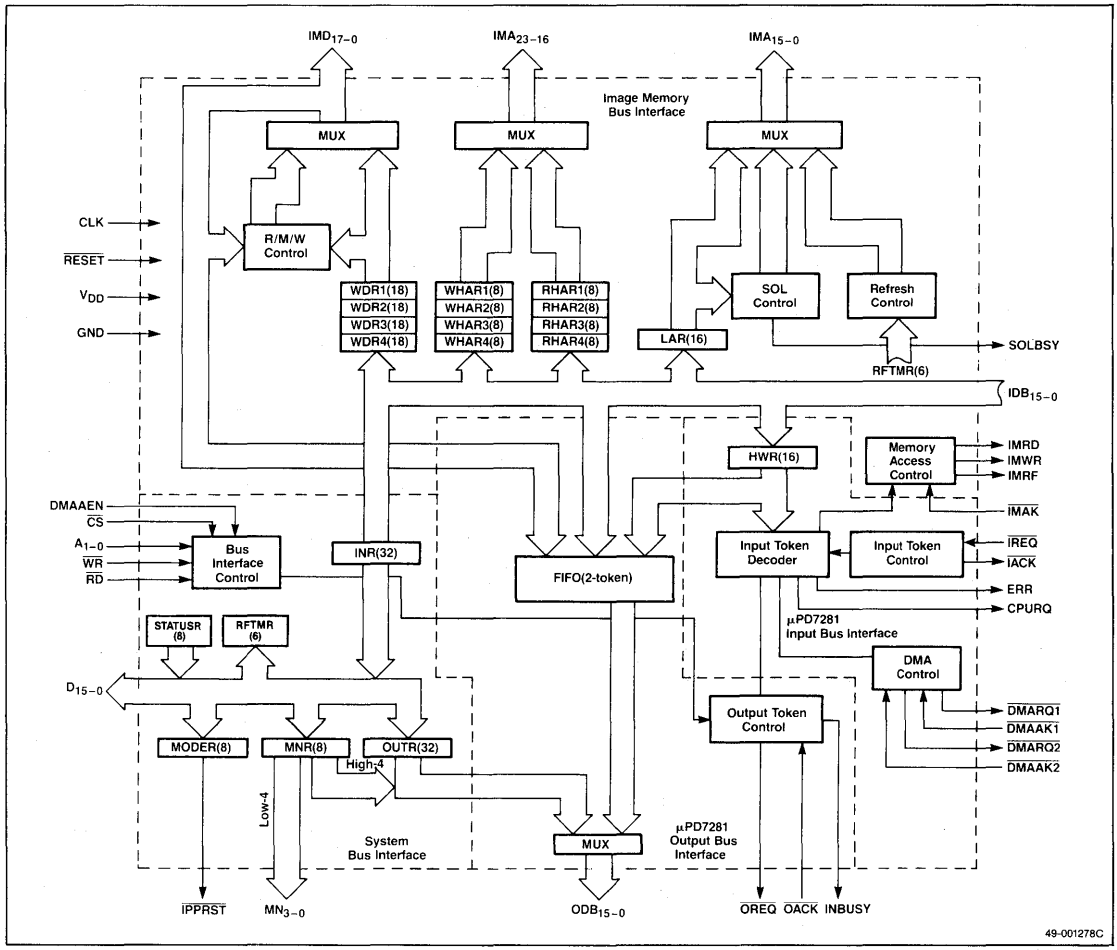
IMA_0-IMA_{23} (Image Memory Address)

IMA_0-IMA_{23} supplies the image memory address for a read or write operation or for DRAM refresh (IMA_0-IMA_9 only).

IMD_0-IMD_{17} (Image Memory Data)

IMD_0-IMD_{17} is the bidirectional data bus for transferring data to and from the image memory.

μPD9305 Block Diagram



Functional Description

The μPD9305 has the following functional units:

- μPD7281 input bus interface
- μPD7281 output bus interface
- System bus interface
- Image memory bus interface
  - —Register file
  - —R/M/W control
  - —Self object load control
  - —Image memory refresh control

μPD7281 Input Bus Interface

After the last ImPP outputs a token, the input bus interface determines whether the token should be an output token to the host CPU, to the image memory, or to the output bus interface block. The high order 16 bits of the token output from the last ImPP are latched into in the high word register (HWR) and then decoded by the input token decoder to determine the token type.

μPD7281 Output Bus Interface

The output bus interface logic transmits tokens through the multiplexer (MUX) to the first ImPP. The transmitted tokens come from the system bus interface, the μPD7281 input bus interface, or the image memory bus interface. The output bus interface uses a priority control mechanism to prevent collisions between the tokens coming from the different blocks.

System Bus Interface

The system bus interface receives a token from the host CPU for the ImPPs, sends it to the output register (OUTR), and signals the output bus interface. Conversely, it sends a token, which is output from the last ImPP, through the input register (INR) to the host CPU according to instructions from the host CPU. The host CPU can set input or output modes (MODER register), read the status register (STATUSR), set image memory refresh timing (RFTMR register), and set module numbers (MNR) for two μPD7281s.

Image Memory Bus Interface

The image memory bus interface accepts the following five types of tokens:

Token	Description
WHA	Write high address
WLA	Write low address
WD	Write data
RHA	Read high address
RLA	Read low address

Tokens have a 16-bit data value, so the address is transferred in two tokens to form the 24-bit image memory address. The lower 16-bits of the image memory address are latched in the lower address register.

The image memory bus interface also performs read/modify/write functions with the R/M/W control logic and provides a register file.

Register File. The register file is used for storing write high addresses (WHAR/four 8-bit registers), write data (WDR/four 18-bit registers), and read high addresses (RHAR/four 8-bit registers).

Read/Modify/Write (R/M/W) Control. The R/M/W control reads a word from the image memory, performs a logical operation (AND, OR, or XOR) between it and the contents of a write data register (WDR), and then writes it back to a location referenced by the WHAR (the same lower 16-bit address, but a different upper eight bits).

Self Object Load (SOL). The self object load control loads ImPP object programs stored in image memory into the ImPPs. When the SOL is given a starting address, the SOL control automatically generates the appropriate addresses to read the image memory.

Image Memory Refresh Control. The μPD9305 generates a 10-bit address and the timing for refreshing dynamic image memories. The timing is set by the RFTMR register.

Figure 1 shows the input/output token format and table 2 shows how the image memory access tokens function.

Figure 1. Input/output Token Format

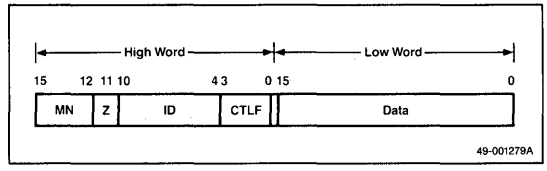


Table 2. Image Memory Access Tokens(1)

MN	Z	ID		CTLF	Data	Function	Operation
0001	-	MN′	ID′	- - - -	Image memory read low address	Image memory read (RHAR1 reference)	R
		1 1 1	- - - -	- - - -	- - - - - Image memory read high address	Read high address register (RHAR1) set (Note 2)	S
0010	-	MN′	ID′	- - - -	Image memory read low address	Image memory read (RHAR2 reference)	R
		1 1 1	- - - -	- - - -	- - - - - Image memory read high address	Read high address register (RHAR2) set (Note 2)	S
0011	-	MN′	ID′	- - - -	Image memory read low address	Image memory read (RHAR3 reference)	R
		1 1 1	- - - -	- - - -	- - - - - Image memory read high address	Read high address register (RHAR3) set (Note 2)	S
0100	-	MN′	ID′	- - - -	Image memory read low address	Image memory read (RHAR4 reference)	R
		1 1 1	- - - -	- - - -	- - - - - Image memory read high address	Read high address register (RHAR4) set (Note 2)	S
		0 0 0 0 0	DIR	- - - -	Image memory write low address	Image memory write (referencing WHAR and WDR selected by DIR)	W
		0 0 1 - -	DIR	- - - -	- - - - - Image memory write high address	Set write high address register (WHAR) selected by DIR	S
0101	-	0 1 0 - -	DIR	- - C,S	Image memory write data register	Set write data register (WDR) selected by DIR	S
		0 1 1 - -	DIR	- - - -	- - - - - Image memory read high address	Set read high address register (RHAR) selected by DIR	S
		1 0 0 MASK	DIR	- - - -	Read/write low address	Read/modify/write	RW
		1 0 1 - -	DIR	- - - -	Read/write low address	Read / modify / write (write CS bits selects mask)	RW
		0 0 - - -	DIR	- - - -	Load starting low address	Self object load	R
0110	-	0 1 - - -	DIR	- - - -	Load starting low address	Self object load MN of output token is SOLMN)	R
		1 - - - -	- -	- - - -	- - - - - - - - - - - SOLMN	Set SOLMN for self object load	S

Notes:

(1) The following definitions refer to the above table:
MN: Module number
Z: Always 0
ID: Identifier
CTLF: Control field
ID′: ID used for next circulation
MN′: MN used for next circulation (MN ≠ 111)
DIR: Specifies registers for memory image access
MASK: Specifies the modify mode
–: Do not care
S: Set
R: Read
W: Write

(2) When RHASEL of the mode register is 1, the tokens become image memory read (request) tokens

Table 3 shows module number (MN) values and the five token types (refer to figure 12).

The five token types are:
(1) Output request data to the host
(2) Image memory access data
(3) DMA request data
(4) Pass data
(5) Delete data

Table 3. MN Values and Token Types

Token Type	MN	ID	Function	Abbreviation
(1)	0 0 0 0	x x x x x x x x	μPD7281 output data to host	CPU
(2)	0 0 0 1	MN′ ID′ x x x x x x x	Image memory read1 (RHAR1 select)	IMR
		1 1 1 x x x x	RHAR1 set (Note 2)	
	0 0 1 0	MN′ ID′ x x x x x x x	Image memory read2 (RHAR2 select)	
		1 1 1 x x x x	RHAR2 set (Note 2)	
	0 0 1 1	MN′ ID′ ↔ ↔	Image memory read3 (RHAR3 select)	
		1 1 1 x x x x	RHAR3 set (Note 2)	
	0 1 0 0	MN′ ID′ ↔ ↔	Image memory read4 (RHAR4 select)	
		1 1 1 x x x x	RHAR4 set (Note 2)	
	0 1 0 1	0 0 0 0 0 DIR ↔	Image memory write	IMW
		0 0 1 x x DIR ↔	High address set for write (selected register file is DIR +1)	IMWHA
		0 1 0 x x DIR ↔	Write data set (selected register file is DIR +1)	IMWD
		0 1 1 x x DIR ↔	High address set for read (selected register file is DIR +1)	IMREA
		1 0 0 Mask DIR ↔ ↔	Read/modify/write1	RMW1
		1 0 1 x x DIR ↔	Read / modify / write2 (mask selected by CS bits of image memory write data)	RMW2
(3)	0 1 0 1	1 1 0 x x x x	DMA1 (host → μPD7281)	DMA1
		1 1 1 x x x x	DMA2 (μPD7281 → host)	DMA2
(2)	0 1 1 0	0 0 x x x DIR ↔	Self object load1	SOL1
		0 1 x x x DIR ↔	Self object load2 (rewrite MN)	SOL2
		1 x x x x x x	MN set for self object load	SOLMN
(4)	0 1 1 1		μPD7281 module number (when RHASEL=1)	PASS
	1 0 0 0 1 0 0 1 1 0 1 0 1 1 0 0 1 1 0 1 1 1 1 0		μPD7281 module numbers	
(5)	1 1 1 1		Deleted	VANISH

Notes:

(1) The following definitions refer to the above table:
MN: Module number
ID: Identifier
MN′: MN used for next circulation (MN≠111)
ID′: ID used for next circulation

(2) When RHASEL of the mode register is 1, the tokens become image memory read tokens.

Absolute Maximum Ratings

T_A = 25°C

Power supply voltage, V_{DD}	−0.5 V to 7.0 V
Input voltage, V_I	−0.5 V to 7.0 V
Output current, I_O	10 mA
Operating temperature, T_{OPT}	0°C to 70°C
Storage temperature, T_{STG}	−65°C to 150°C

***Comment:** Exposing the device to stresses above those listed in absolute maximum ratings could cause permanent damage. Do not operate the device under conditions outside the limits described in the operational sections of this specification. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Capacitance

T_A = 25°C

Parameter	Symbol	Limits Min	Limits Max	Unit	Test Conditions
Input capacitance	C_I		10	pF	f_c = 1 MHz Unmeasured pins are at 0 V.
Output capacitance	C_O		15	pF	
Input/output capacitance	C_{IO}		15	pF	

DC Characteristics

T_A = 0°C to +70°C, V_{DD} = 5 V ±10%

Parameter	Symbol	Limits Min	Limits Typ	Limits Max	Unit	Test Conditions
Input low voltage	V_{IL}	−0.5		0.8	V	
Input high voltage	V_{IH}	2.0		V_{DD}+0.5	V	
Output low voltage	V_{OL}			0.4	V	I_{OL} = 2 mA
Output high voltage	V_{OH}	V_{DD} −0.4			V	I_{OL} = −400 μA
Input leakage current	I_{LI}			±10	μA	$0 \le V_I \le V_{DD}$
Output leakage current	I_{LO}			±10	μA	$0 \le V_I \le V_{DD}$
Supply current	I_{DD}		10	100	mA	10 MHz

AC Characteristics

T_A = 0°C to +70°C, V_{DD} = 5 V ±10%

Clock Timing

Parameter	Symbol	Limits Min	Limits Max	Unit	Test Conditions
CLK cycle time	t_{CYK}	80		ns	
Clock pulse width high	t_{WKH}	30		ns	
Clock pulse width low	t_{WKL}	30		ns	
Clock rise time	t_{KR}		10	ns	
Clock fall time	t_{KF}		10	ns	

Input Timing

Parameter	Symbol	Limits Min	Limits Max	Unit	Test Conditions
Input rise time	t_{IR}	0	10	μs	
Input fall time	t_{IF}	0	10	μs	

RESET Timing

Parameter	Symbol	Limits Min	Limits Max	Unit	Test Conditions
$\overline{\text{RESET}}$ pulse width	t_{RST}	t_{CYK}		ns	μPD9305 only
$\overline{\text{RESET}}$ setup time to $\overline{\text{IPPRST}}$	t_{DRSPRL}		40	ns	
$\overline{\text{IPPRST}}$ hold time after $\overline{\text{RESET}}$ ↑	t_{DRSPRH}		50	ns	
$\overline{\text{IPPRST}}$ setup to MN_0-MN_3	t_{DMN}		60	ns	
MN_0-MN_3 float time after $\overline{\text{IPPRST}}$ ↑	t_{FMN}		50	ns	
$\overline{\text{IPPRST}}$ low until OBD_{15}-OBD_{12} active	t_{DPROD}		60	ns	
OBD_{15}-OBD_{12} float time after $\overline{\text{IPPRST}}$ ↑	t_{FPROD}		50	ns	

2

Host CPU ↔ μPD9305 Read/Write Timing

Parameter	Symbol	Limits Min	Limits Max	Unit	Test Conditions
Address setup to $\overline{WR}$ ↓, $\overline{RD}$ ↓	t_{SARW}	20		ns	
Address hold time after $\overline{WR}$ ↑, $\overline{RD}$ ↑	t_{HRWA}	20		ns	
$\overline{CS}$ setup to $\overline{WR}$ ↓ $\overline{RD}$ ↓	t_{SCRW}	0		ns	
$\overline{CS}$ hold time after $\overline{WR}$ ↑, $\overline{RD}$ ↑	t_{HRWC}	0		ns	
$\overline{WR}$, $\overline{RD}$ pulse width	t_{WRWL}	100		ns	
$\overline{RD}$ setup to data	t_{DRD}		80	ns	
Data float time after $\overline{RD}$ ↑	t_{FRD}		30	ns	
Data setup to $\overline{WR}$ ↑	t_{SDW}	20		ns	
Data hold after $\overline{WR}$ ↑	t_{HWD}	20		ns	

DMA Request Timing(1)

Parameter	Symbol	Limits Min	Limits Max	Unit	Test Conditions
$\overline{DMARQ}$ ↓ setup time to $\overline{DMAAK}$ ↓	t_{DDQDA}	20		ns	
$\overline{DMARQ}$ ↑ time from $\overline{DMAAK}$ ↓	t_{DDADQ}		50	ns	
$\overline{DMARQ}$ ↓ time from $\overline{DMAAK}$ ↑	t_{RVDQ}	50		ns	
DMAAEN ↑ setup time to ($\overline{RD}$,$\overline{WR}$) ↓	t_{SDERW}	30		ns	
DMAAEN hold time after ($\overline{RD}$,$\overline{WR}$) ↑	t_{HRWDE}	30		ns	
$\overline{DMAAK}$ low setup time to ($\overline{RD}$,$\overline{WR}$) ↓	t_{SDARW}	0		ns	
$\overline{DMAAK}$ hold time after ($\overline{RD}$,$\overline{WR}$) ↑	t_{HRWDA}	0		ns	
$\overline{DMAAK}$ pulse width	t_{WDAL}	t_{CYK}		ns	

Note:
(1) $\overline{DMAAK}$ = $\overline{DMAAK1}$ or $\overline{DMAAK2}$
$\overline{DMARQ}$ = $\overline{DMARQ1}$ or $\overline{DMARQ2}$

I/O Request/Acknowledge Timing

Parameter	Symbol	Limits Min	Limits Max	Unit	Test Conditions
$\overline{IREQ}$ ↓ setup time to $\overline{IACK}$ ↓	$t_{DIQIALI}$	15	60	ns	
$\overline{IACK}$ ↓ setup time to $\overline{IREQ}$ ↑	$t_{DIAIQHI}$	10		ns	
$\overline{IREQ}$ ↑ setup time to $\overline{IACK}$ ↑	$t_{DIQIAHI}$	20	70	ns	
$\overline{IACK}$ ↑ setup to $\overline{IREQ}$ ↓	t_{DIAIQL}	10		ns	
ID bus setup time to $\overline{IREQ}$ ↑	t_{SIDIQ}	20		ns	
ID bus hold time from $\overline{IREQ}$ ↑	t_{HIQID}	10		ns	
$\overline{OREQ}$ ↓ setup time to $\overline{OACK}$ ↓	t_{DOQOAL}	10		ns	
$\overline{OACK}$ ↓ setup time to $\overline{OREQ}$ ↑	t_{DOAOQH}	20	70	ns	
$\overline{OREQ}$ ↑ setup time to $\overline{OACK}$ ↑	t_{DOQOAH}	10		ns	
$\overline{OACK}$ ↑ setup time to $\overline{OREQ}$ ↓	t_{DOAOQL}	15	60	ns	
$\overline{OREQ}$ ↓ setup time to $\overline{ODB}$ valid	t_{DOQOD}		10	ns	
$\overline{ODB}$ float time after $\overline{OREQ}$ ↑	t_{FOQOD}	10		ns	

Note:
Pull-up resistors required on μPD9305 IDB_{15}-IDB_0 to meet t_{HIQID} timing.

Image Memory Read, Write, Refresh Timing

Parameter	Symbol	Limits Min	Limits Max	Unit	Test Conditions
IMA(1) ↑ active time from CLK ↓	t_{DKMARF}		100	ns	IM refresh
IMA active time from CLK ↓	t_{DKMAMC}		60	ns	IM read or IM write
IMA float time from IMC ↓	t_{FMCMA}	10		ns	
IMC recovery time	t_{RVMC}	$1.5t_{CYK}$		ns	
IMC ↑ delay time from CLK ↓	t_{DKMCH}		35	ns	
IMC ↓ delay time from CLK ↓	t_{DKMCL}		40	ns	
$\overline{IMAK}$ recovery time	t_{RVMK}	$1.5t_{CYK}$		ns	
$\overline{IMAK}$ setup time to CLK ↓	t_{SMKK}	10		ns	
$\overline{IMAK}$ hold time from IMC ↓	t_{HMCMK}	0		ns	
IMD setup time to CLK ↑	t_{SMDK}	20		ns	Image memory read timing
IMD hold time from IMRD ↓	t_{HMRMD}	0		ns	Image memory read timing
IMD delay time from CLK ↓	t_{DKMD}		30	ns	Image memory write timing
IMD float time from IMWR ↓	t_{FMWMD}	20		ns	Image memory write timing

Note:

(1) IMA = IMA_{23}-IMA_0

(2) IMC + IMRD, IMWR or IMRF

(3) To maximize IM access time use $\overline{IMAK}$ = $\overline{IMC}$. Then IM cycle time will be $3.k_{CYK}$

SOLBSY Timing

Parameter	Symbol	Limits Min	Limits Max	Unit	Test Conditions
SOLBSY delay time from $\overline{IACK}$ ↑	t_{DIASB}		30	ns	
SOLBSY delay time from CLK ↓	t_{DKSB}		60	ns	

CPURQ Timing

Parameter	Symbol	Limits Min	Limits Max	Unit	Test Conditions
CPURQ delay time from $\overline{IACK}$ ↑	t_{DIAPQ}		30	ns	
CPURQ delay time from $\overline{RD}$ ↑	t_{DPRQ}		60	ns	

INBUSY Timing

Parameter	Symbol	Limits Min	Limits Max	Unit	Test Conditions
INBUSY ↑ delay time from $\overline{WR}$ ↑	t_{DWIB}		70	ns	
INBUSY ↓ delay time from $\overline{OREQ}$ ↑	t_{DOQIB}		40	ns	

ERR Timing

Parameter	Symbol	Limits Min	Limits Max	Unit	Test Conditions
ERR ↑ delay time from $\overline{IACK}$ ↑	t_{DIAE}		30	ns	Error token output
ERR ↑ delay time from $\overline{WR}$ ↓	t_{DWE}		60	ns	INBUSY = 1
ERR ↑ delay time from $\overline{RD}$	t_{DRE}		60	ns	CPURQ = 0
INBUSY hold time from $\overline{WR}$ ↓	t_{HWIB}		10	ns	
CPURQ setup time to $\overline{RD}$ ↓	t_{SPQR}		10	ns	

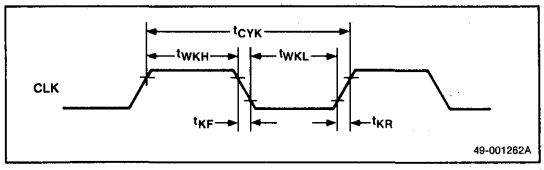
Note:

All unused input or output pins should be pulled up to V_{DD} or down to GND through a 2K-3K ohm resistor.

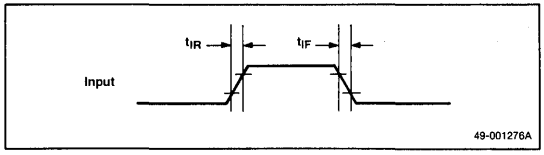
2

Timing Waveforms

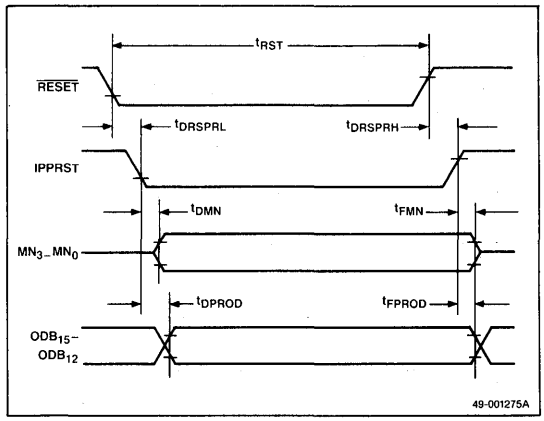
Clock Timing



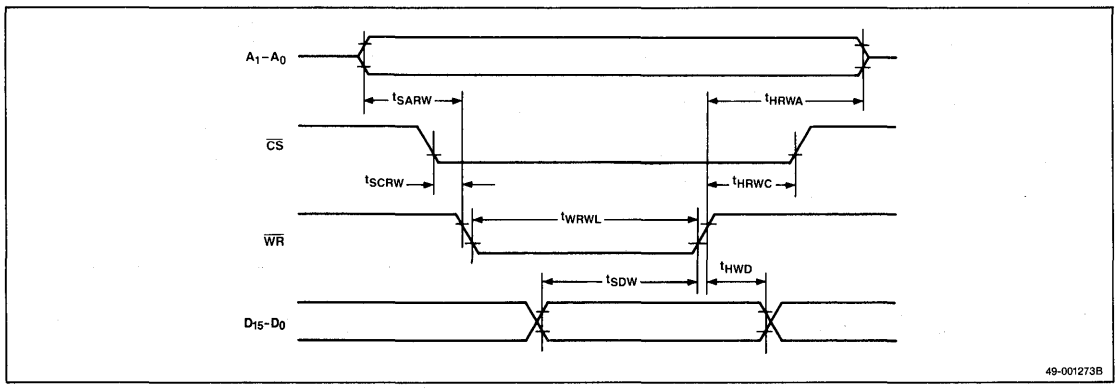
Input Timing



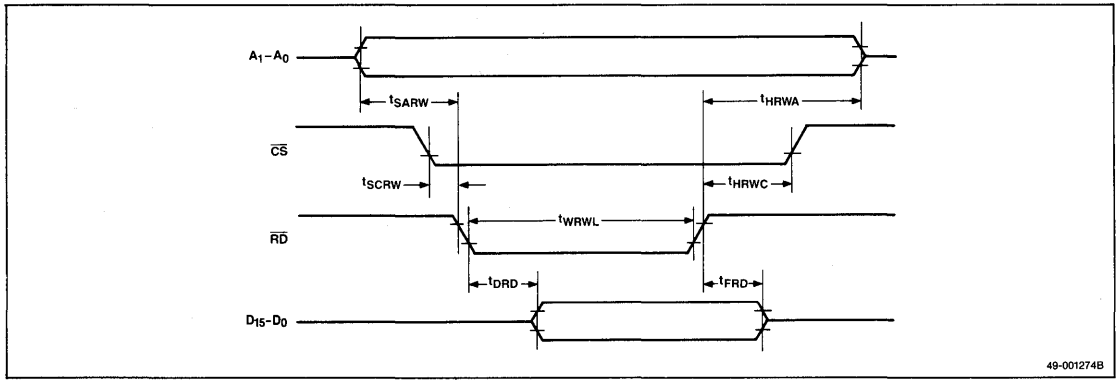
RESET Timing



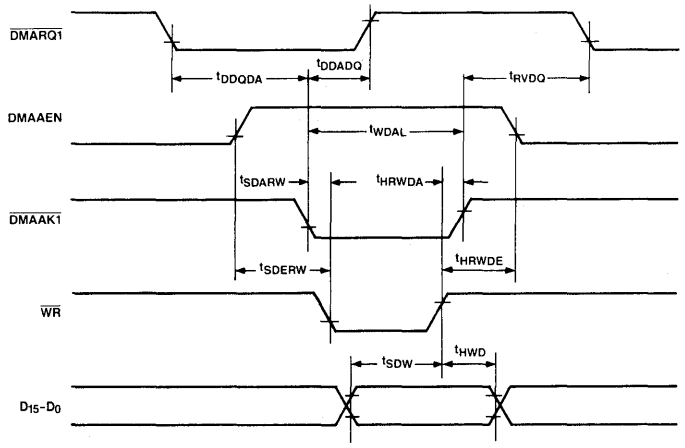
Host CPU ↔ μPD9305 Write Timing



Host CPU ↔ μPD9305 Read Timing

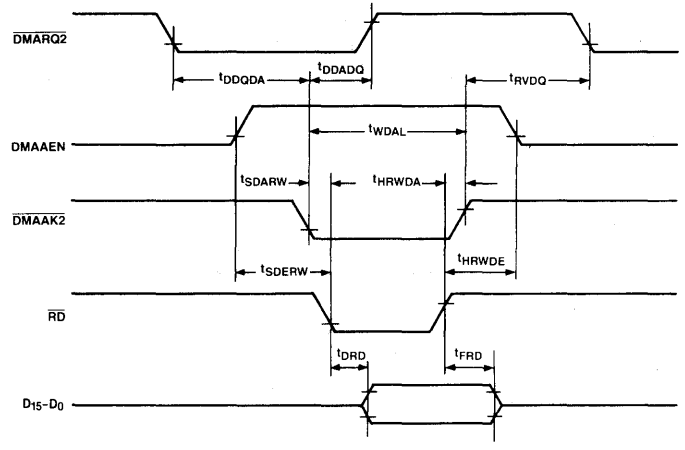


DMA1 Request Timing



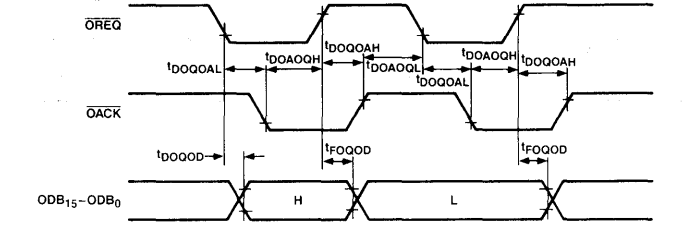
49-001271B

DMA2 Request Timing



49-001272B

I/O Request/Acknowledge Timing



49-001259B

I/O Data Bus Handshake Timing

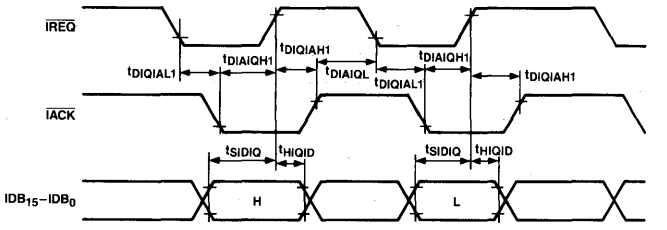


Image Memory Read Timing

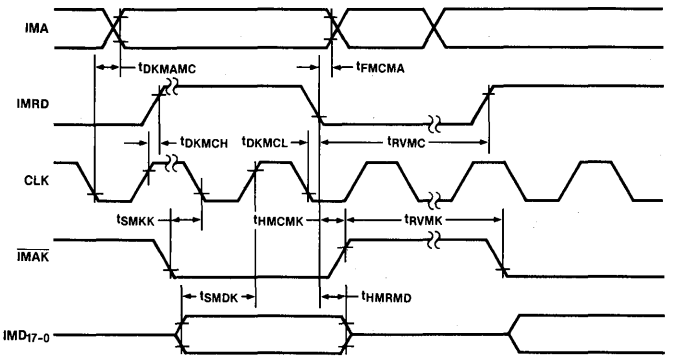


Image Memory Write Timing

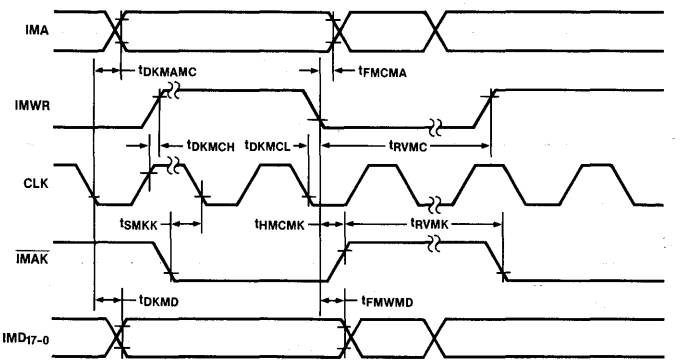
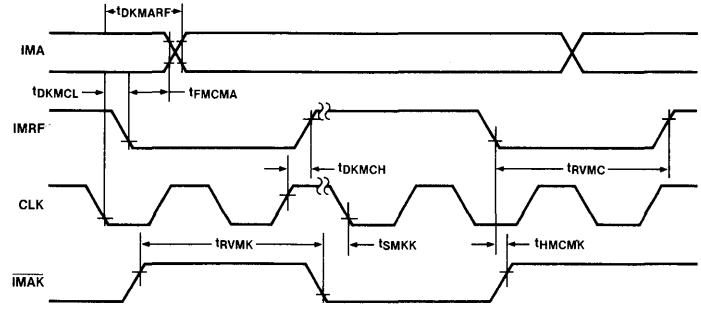
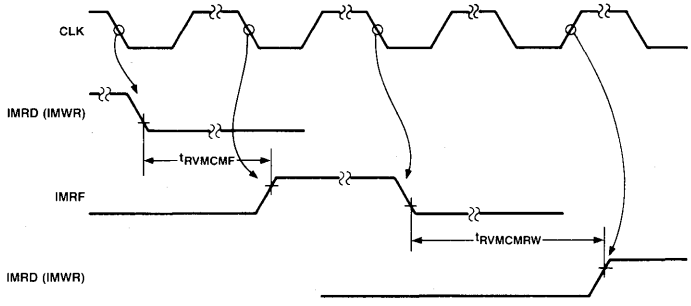


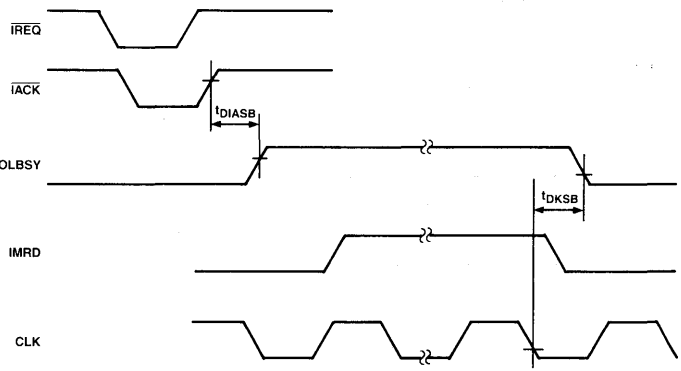
Image Memory Refresh Timing



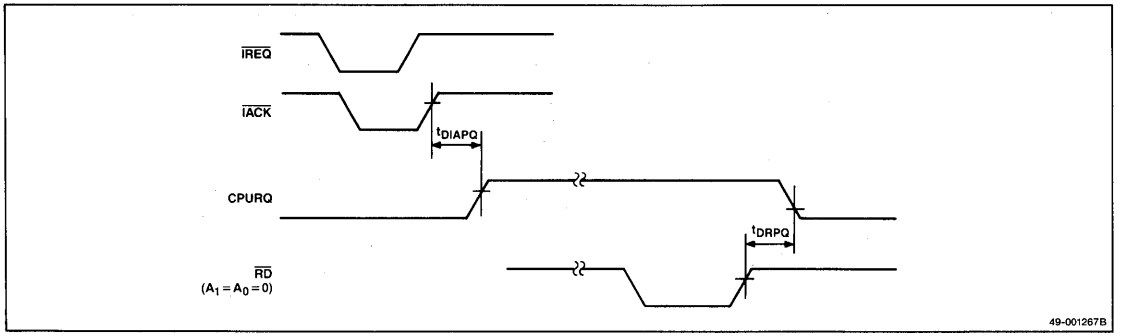
IM Command Timing



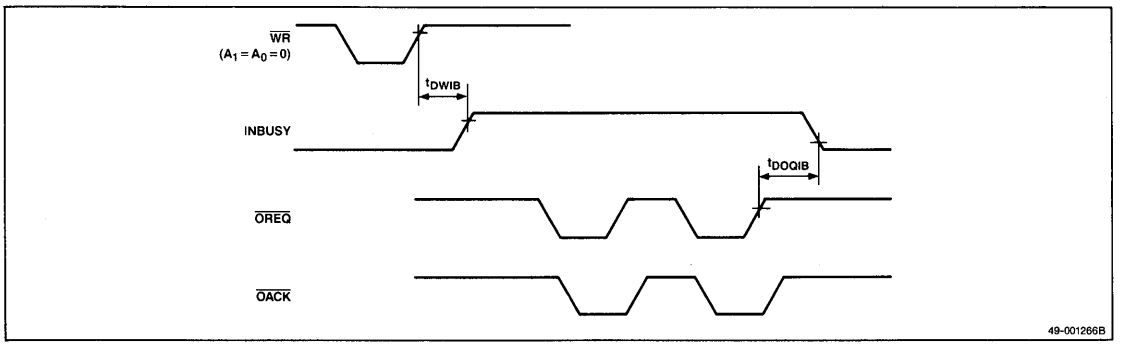
SOLBSY Timing



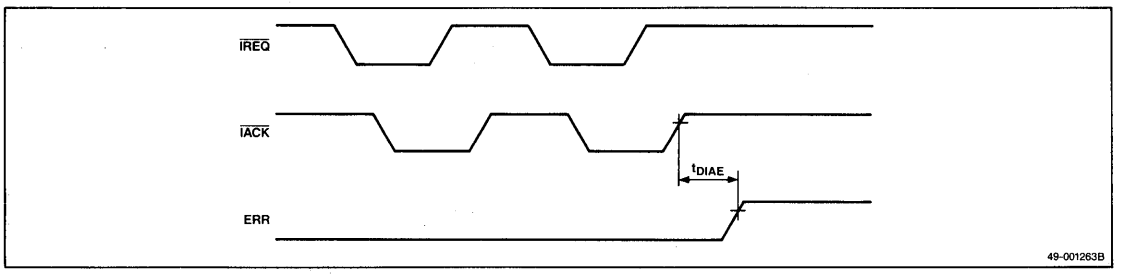
CPURQ Timing



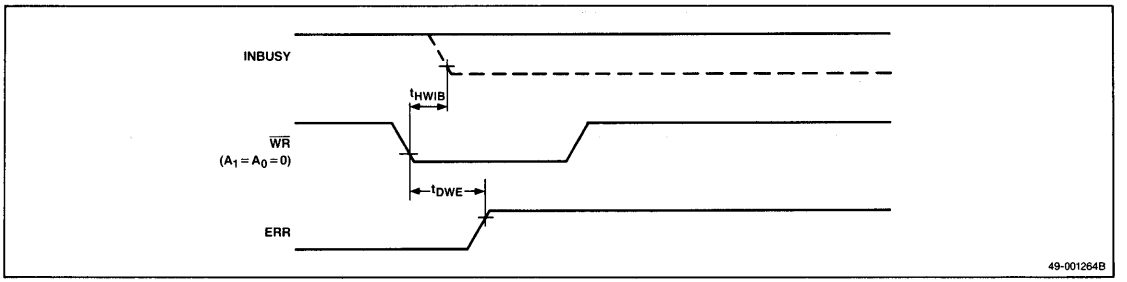
INBUSY Timing



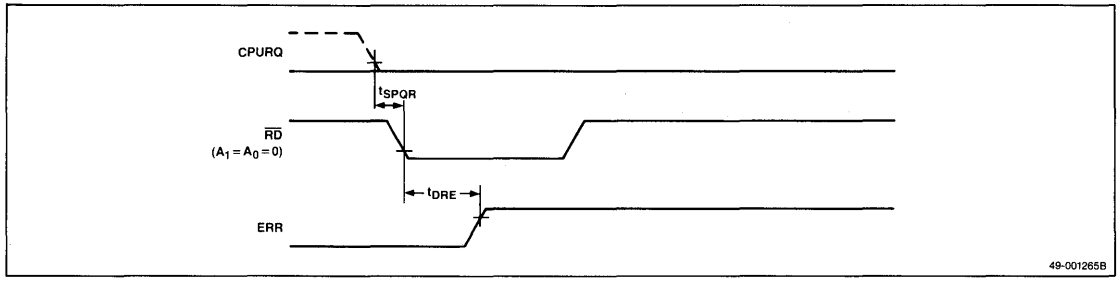
ERR Timing, Error from ImPP



ERR Timing, INBUSY



ERR Timing, CPU Request



μPD9305 Operation

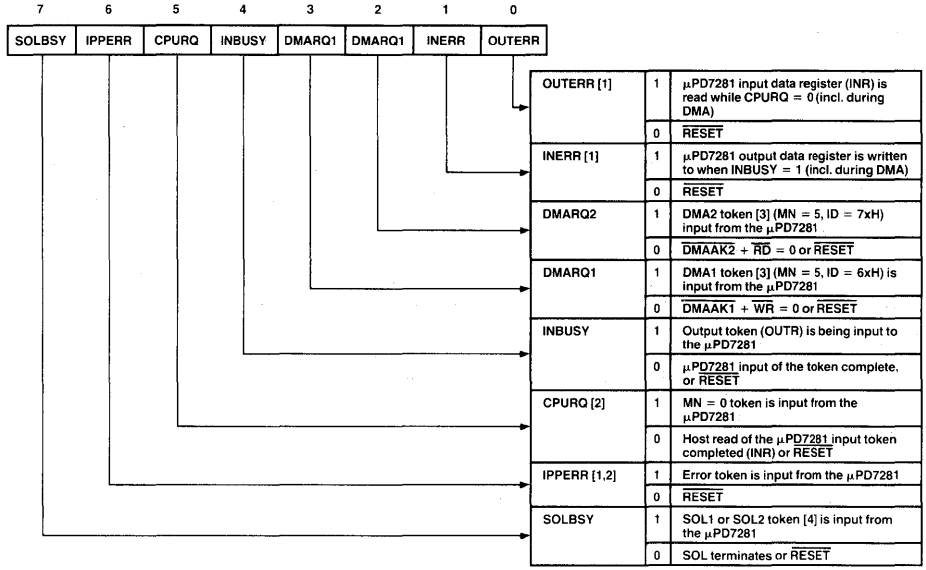
Table 4 shows how the μPD9305 uses signals $\overline{CS}$, $\overline{RD}$, $\overline{WR}$, and A_1, A_0 to read or write to I/O ports.

Table 4. I/O Ports

$\overline{CS}$	$\overline{RD}$	$\overline{WR}$	A_1	A_0	Internal I/O Ports
0	0	1	0	0	Read ImPP input data register (from ImPP)
0	0	1	0	1	Read status register
0	0	1	1	0	Command $\overline{RESET}$; data read has no meaning
0	0	1	1	1	Not used
0	1	0	0	0	Write ImPP output data register (to ImPP)
0	1	0	0	1	Write mode register
0	1	0	1	0	Write module number register
0	1	0	1	1	Write refresh timing register

Figure 2 shows the status register format.

Figure 2. Status Register Format

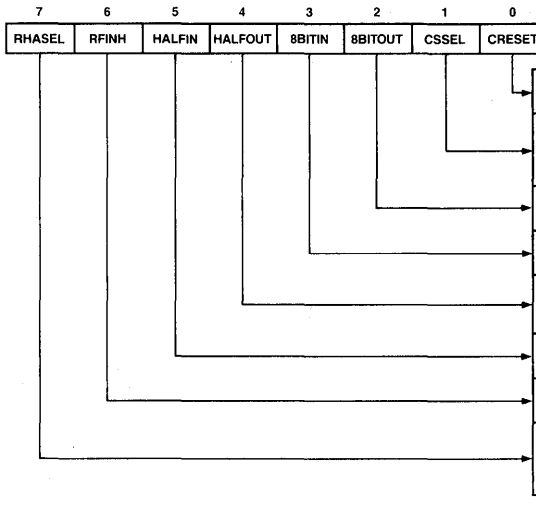


Notes: [1] IPPERR, INERR and OUTERR are ORed and output to the ERR pin.
[2] When IPPERR becomes '1', CPURQ also becomes '1' at the same time.
[3] Refer to μPD9305 User's Manual, Section 6. DMA Request Function
[4] Refer to μPD9305 User's Manual, Section 5. Image Memory/μPD7281 Interfacing

49-001280B

Figure 3 shows the mode register format.

Figure 3. Mode Register Format



CRESET	0	No operation
	1	Command reset
CSSEL	0	Image memory read data C,S bits are '00'
	1	Image memory read data C,S bits are 'CS'
8BITOUT	0	16-bit external data bus
	1	8-bit external data bus
8BITIN	0	16-bit external data bus
	1	8-bit external data bus
HALFOUT	0	High/low 32-bit output
	1	Low 16 bits output (high 16 bits ignored)
HALFIN	0	High/low 32-bit input
	1	Low 16 bits input (high 16 bits fixed)
RFINH	0	IMRF output
	1	IMRF not output
RHASEL	0	Read high address valid for MN = 1 to 4 tokens
	1	Read high address invalid for MN = 1 to 4 tokens (becomes read operation)

Note: All bits of the mode register default to 0 when an external reset is applied.

49-001281A

Figures 4-20 graphically show μPD9305 operation. For a detailed description of μPD9305 operation, refer to the *μPD9305 User's Manual.*

Figure 4. Setting Write Method for Input Data

Mode 8BITIN	Mode HALFIN	Input data write sequence (16-bit: HH HL; 16-bit: LH LL)	Use example
0	0	① [1] ②	16-bit C P U normal use
0	1	[2] Fixed data ①	Scanner →μPD7281 →Image memory
1	0	② ① ④ ③	8-bit C P U normal use
1	1	Fixed data ② ①	8-bit C P U memory →μPD7281 →Image memory

Notes: [1] Circled numbers indicate order in which data is written
[2] Fixed data should be defined beforehand

49-001283A

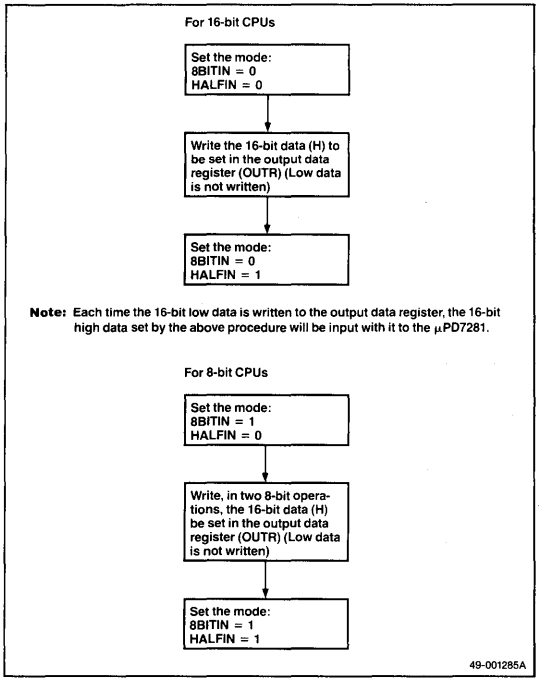
Figure 5. Setting Read Method for Output Data

Mode 8BITOUT	Mode HALFOUT	Output data read-out sequence (16-bit: HH HL; 16-bit: LH LL)	Use example
0	0	① [1] ②	16-bit C P U normal use
0	1	①	16-bit C P U image memory read
1	0	② ① ④ ③	8-bit C P U normal use
1	1	② ①	8-bit C P U image memory read

Note: [1] Circled numbers indicate order in which data is written

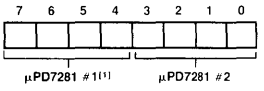
49-001284A

Figure 6. Setting Fixed (16-Bit) Data



Note: Each time the 16-bit low data is written to the output data register, the 16-bit high data set by the above procedure will be input with it to the μPD7281.

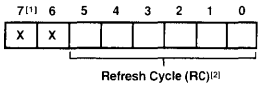
Figure 7. MN Register



Note: [1] Bits 7–4 are output on the output data bus

49-001287A

Figure 8. Refresh Timing Register

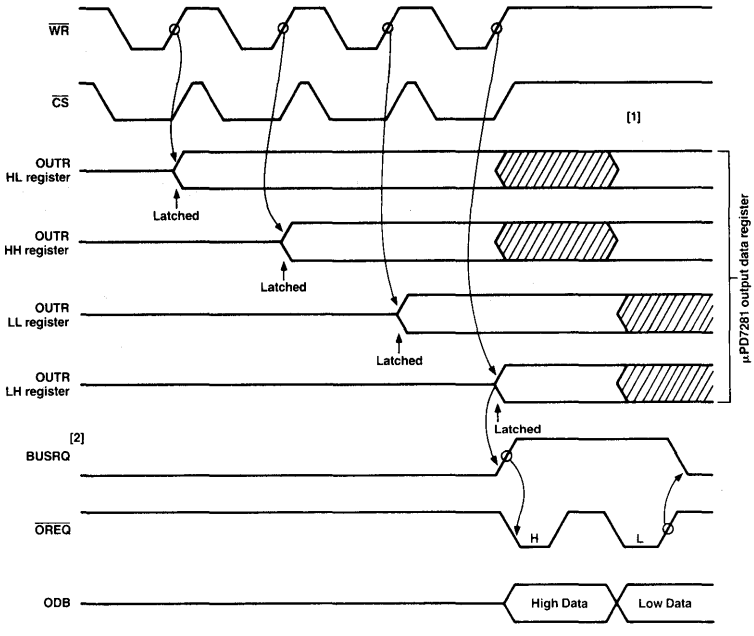


Notes: [1] X = don't care
[2] The actual refresh cycle is (RC + 1) x 8 CLK, RC = 0 to 63

49-001288A

2

Figure 9. Input Timing (Host to μPD9305 to μPD7281)



Notes: [1] ▨ = Output enable
[2] Internal signal
3. 8BIN = 1, HALFIN = 0

49-001289B

2

Figure 10. Output Timing (μPD7281 to μPD9305 to Host)

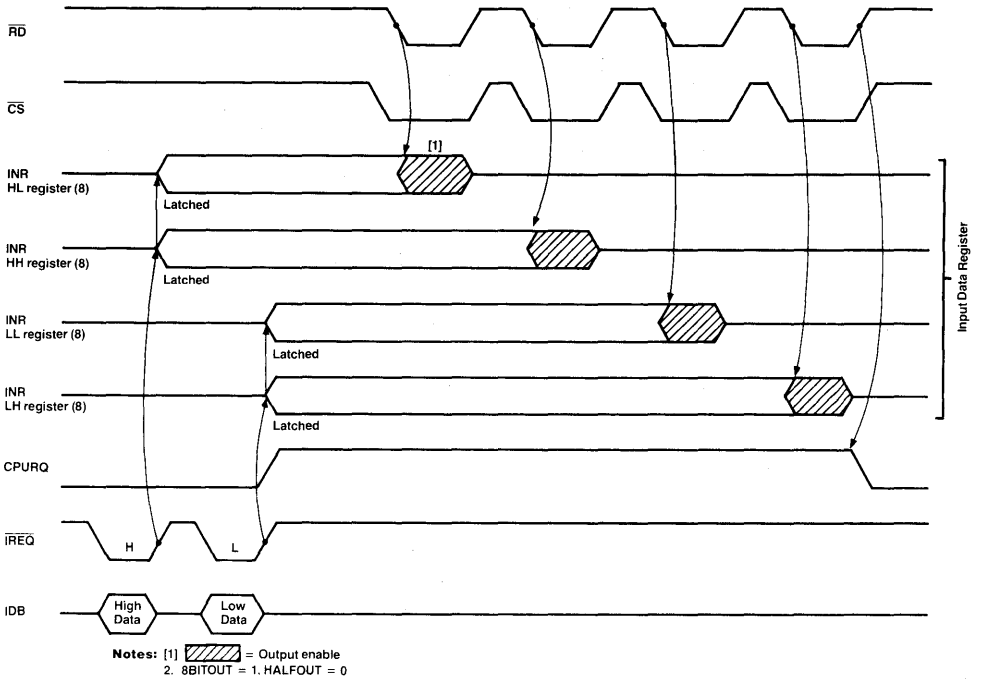


Figure 11. Output to μPD7281, Control Data Paths

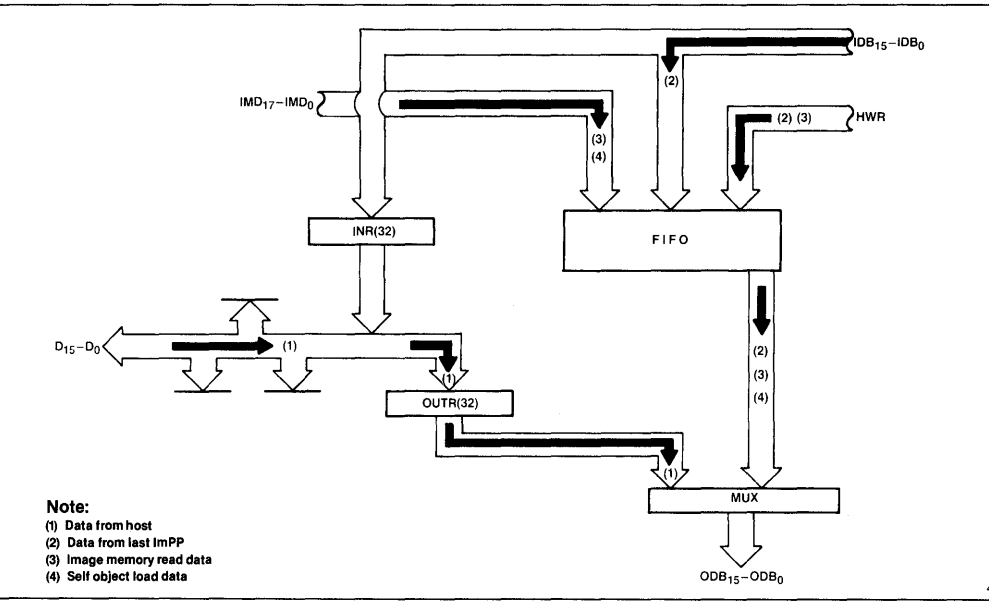


Figure 12. μPD7281, Input Control Data Flow

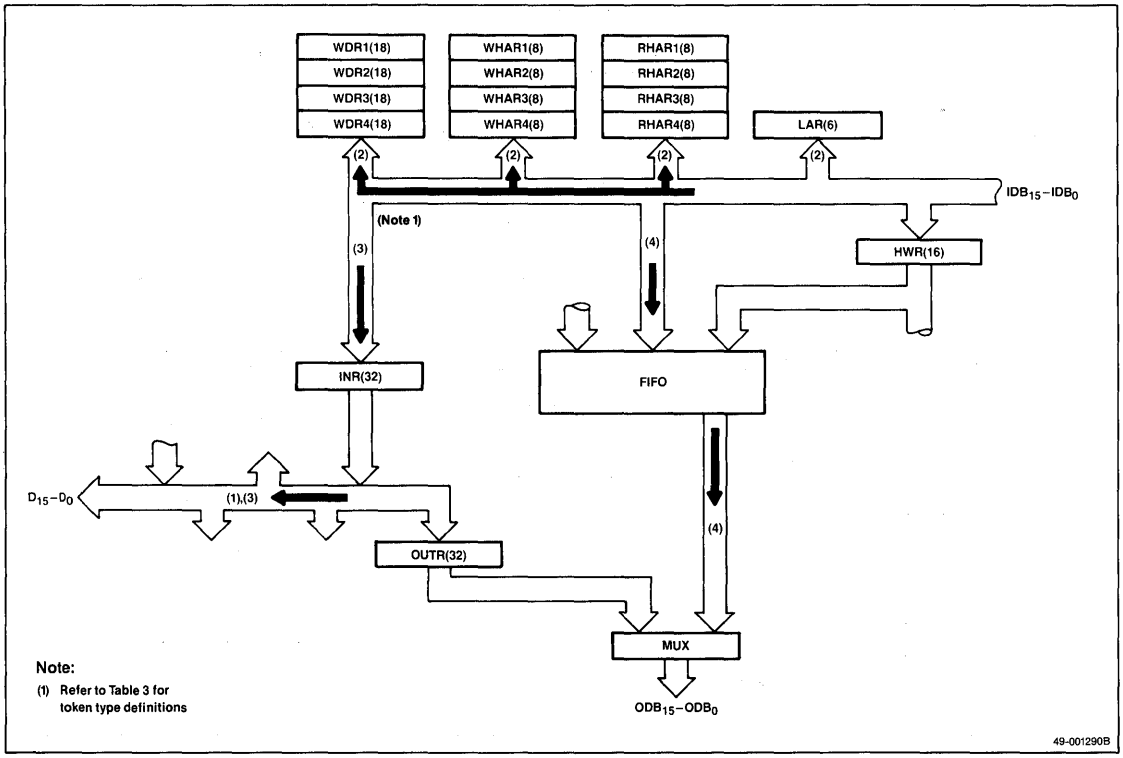


Figure 13. Image Memory Read Timing (Without Refresh Request)

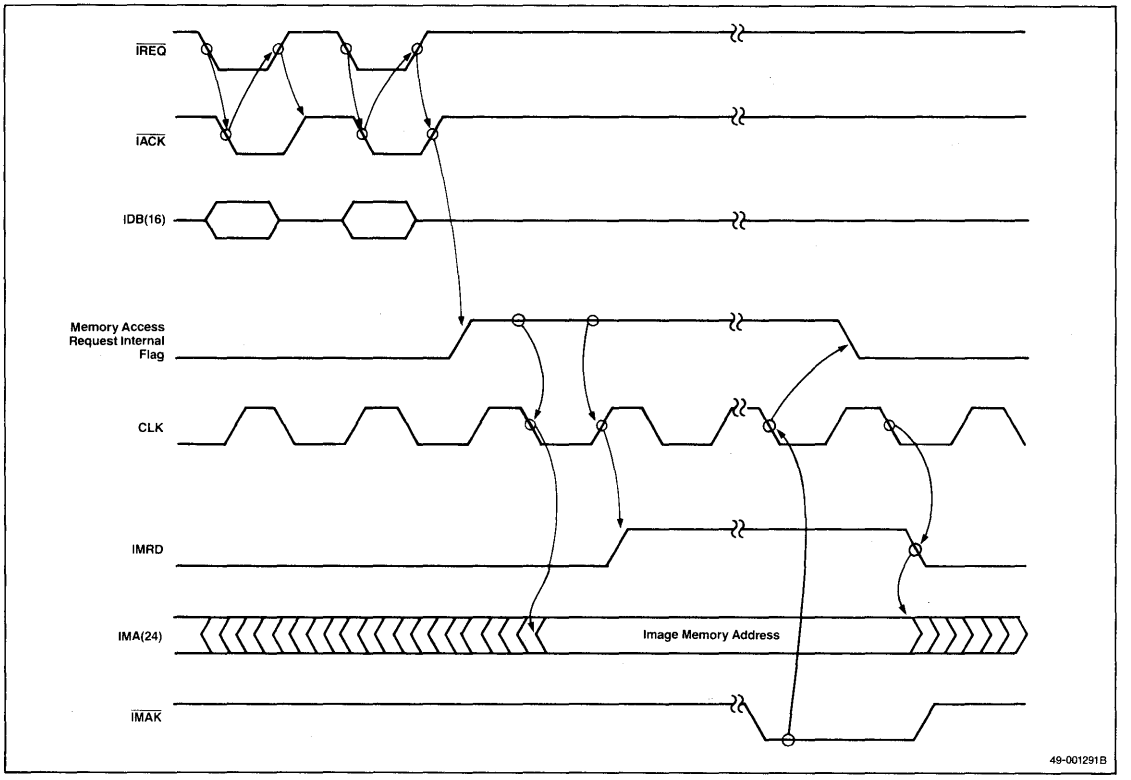


Figure 14. Image Memory Write Timing (Without Refresh Request)

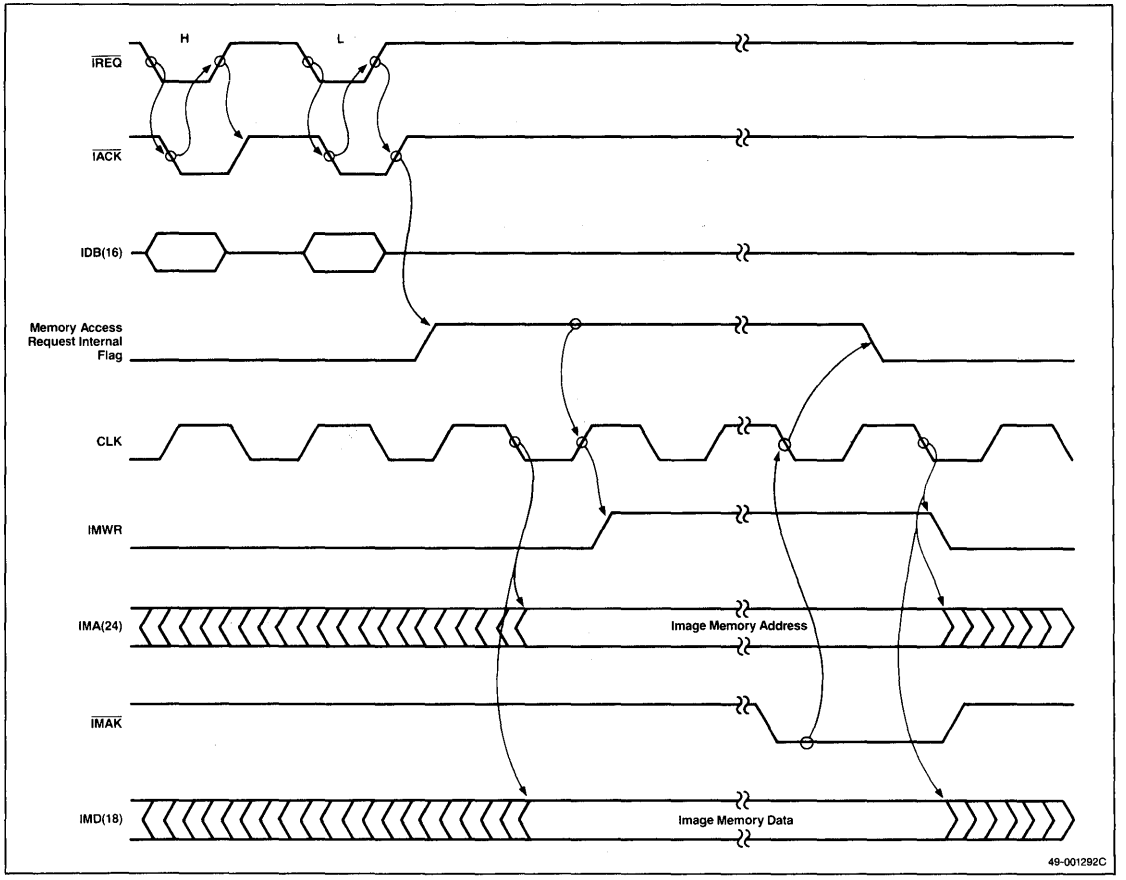


Figure 15. Image Memory Access Request Priority Control

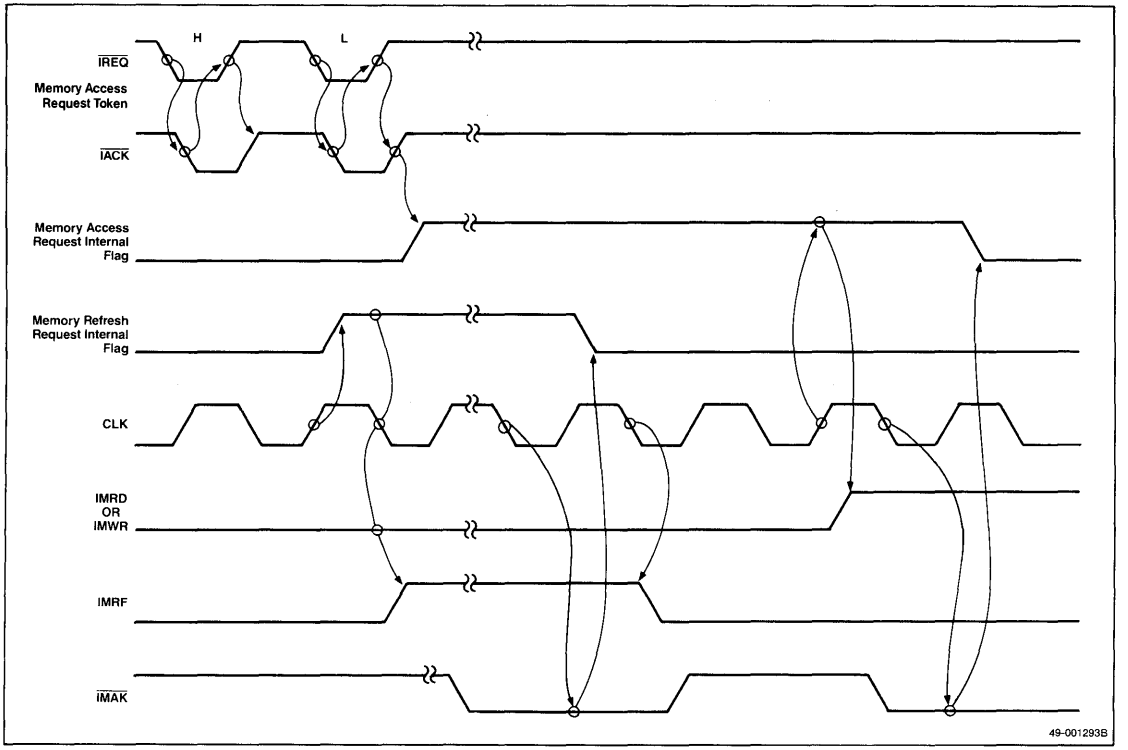


Figure 16. Read Data → μPD7281 Output Timing (Single Output)

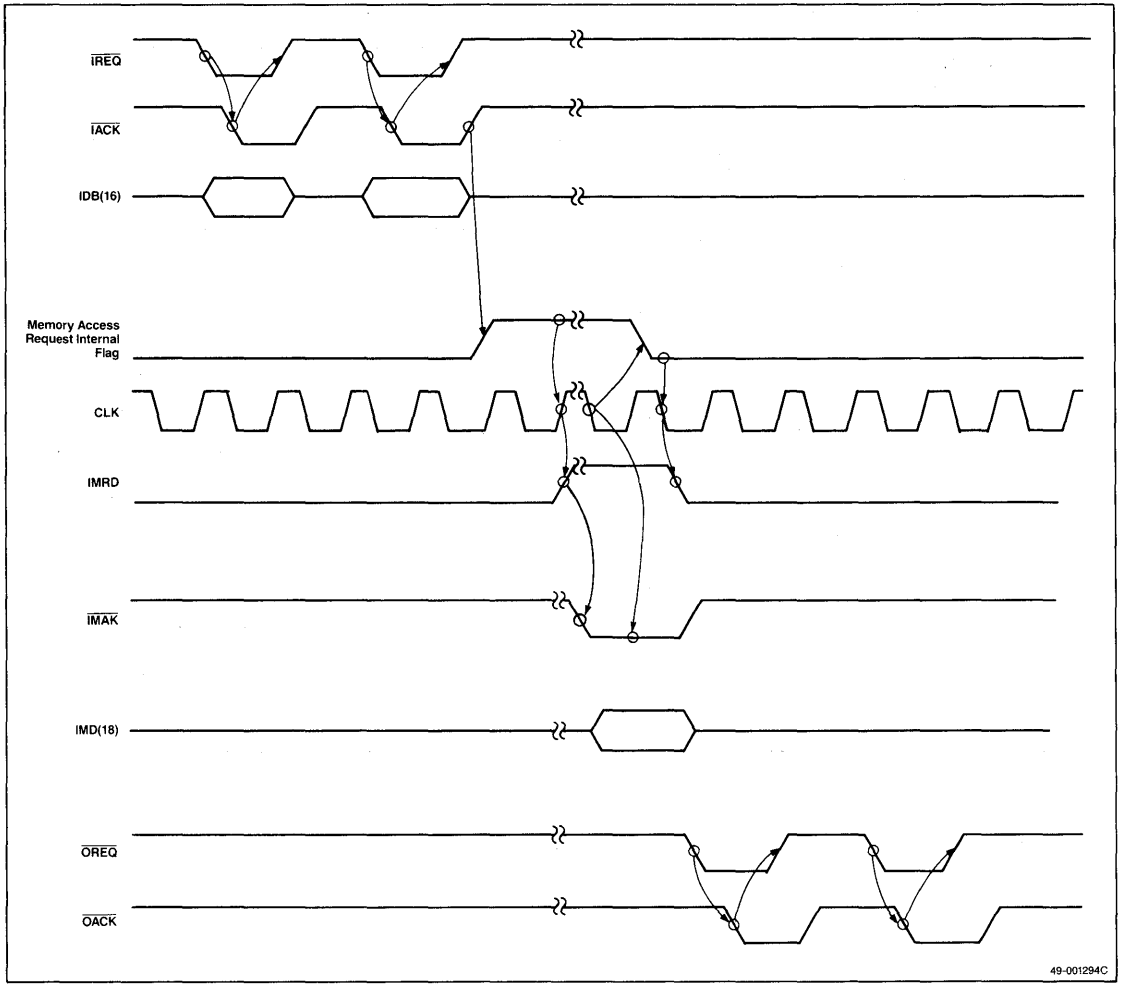


Figure 17. Read Data → μPD7281 Output Timing (Continuous Output)

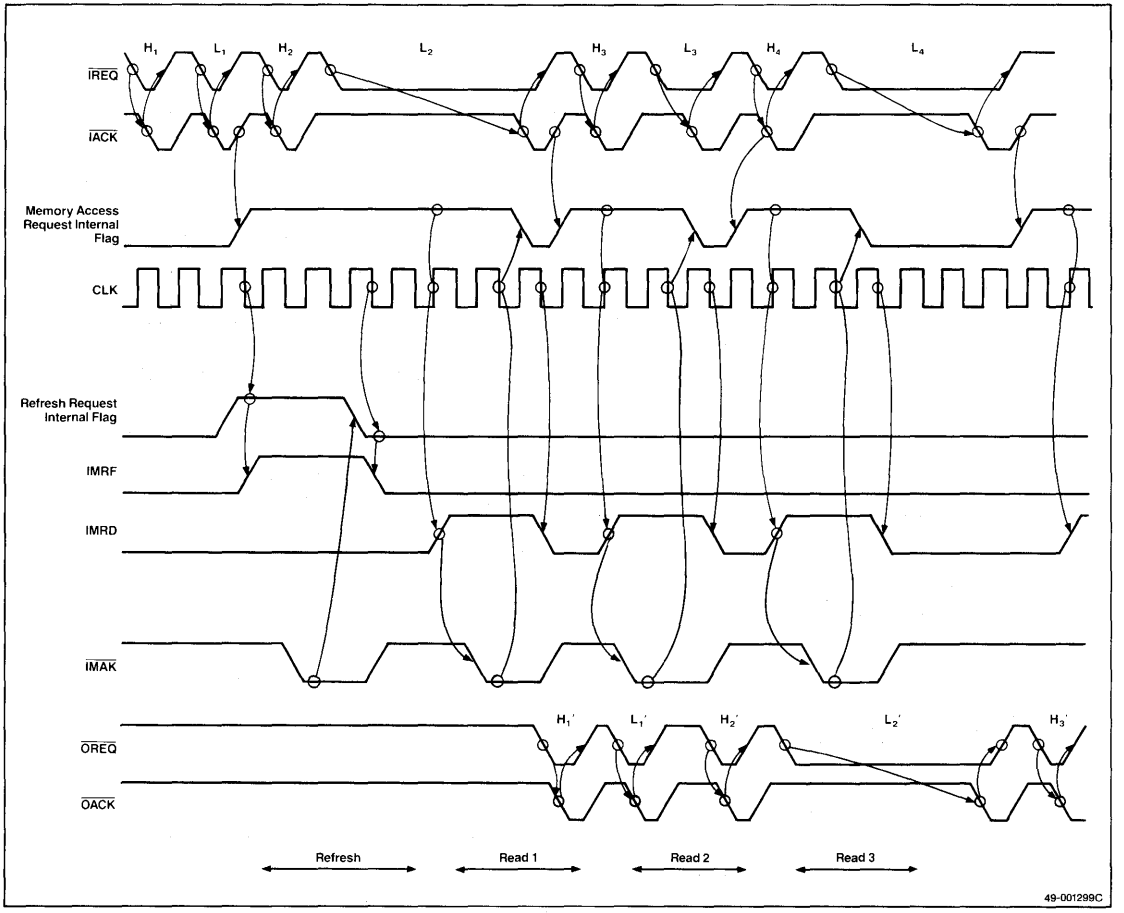
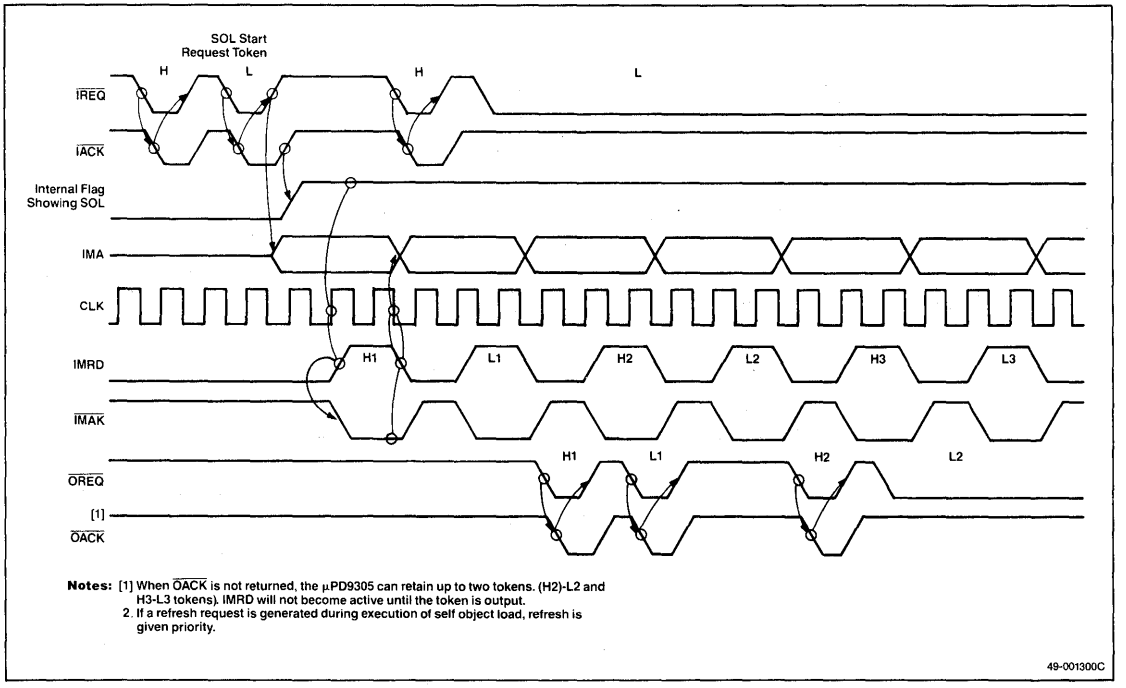


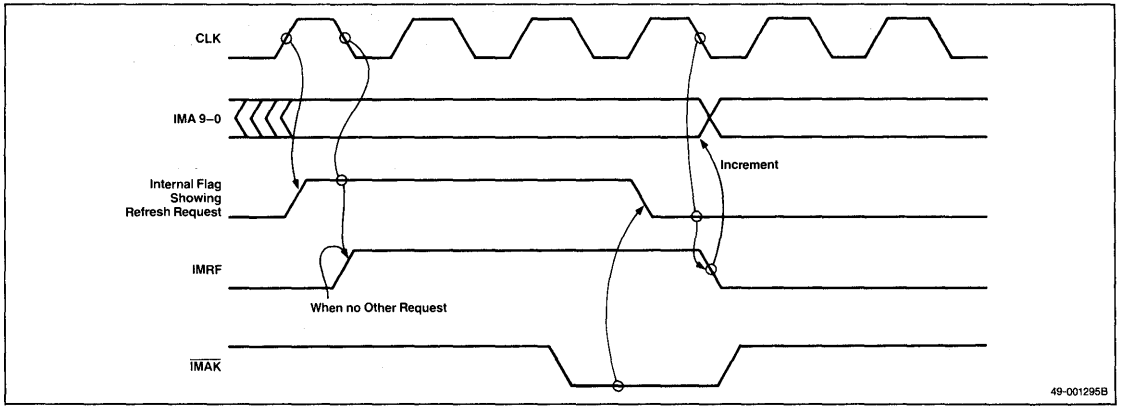
Figure 18. Self Object Load Timing



Notes: [1] When $\overline{OACK}$ is not returned, the μPD9305 can retain up to two tokens. (H2)-L2 and H3-L3 tokens). IMRD will not become active until the token is output.
2. If a refresh request is generated during execution of self object load, refresh is given priority.

49-001300C

Figure 19. Refresh Timing



49-001295B

Figure 20. Read/Modify/Write Timing

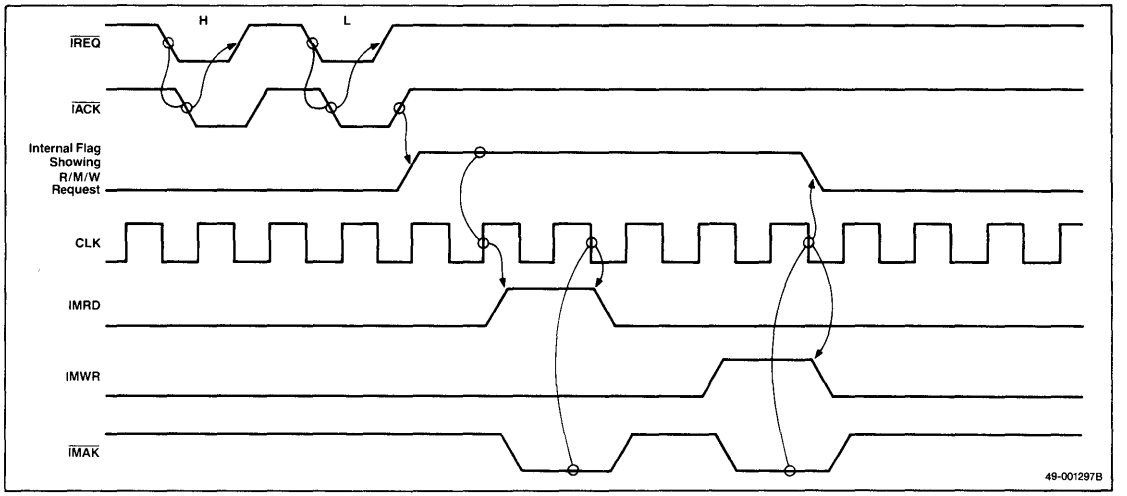


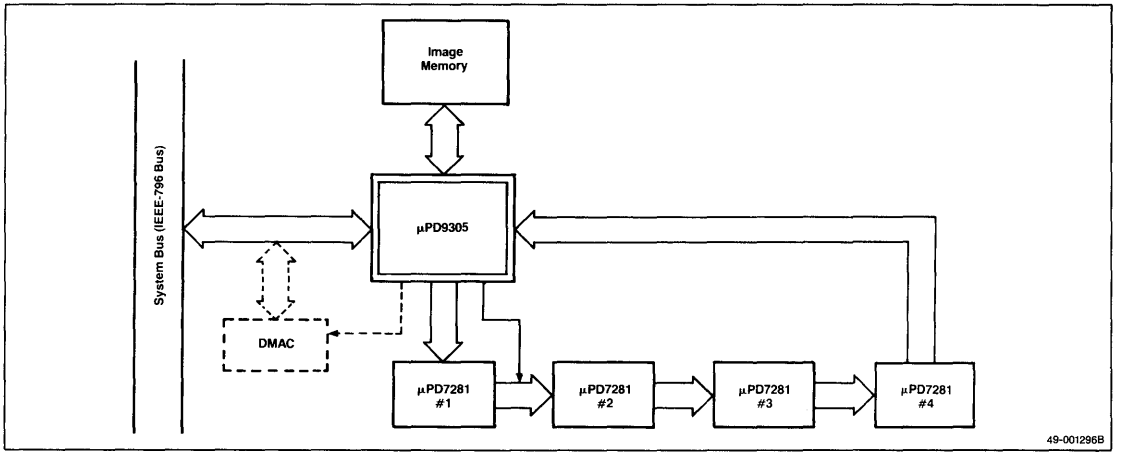
Table 5 shows the differences between command and external resets.

Table 5. Command and External Reset Differences

Item	RESET	Commmand Reset
I/O data counter; Tokens in the μPD9305; image memory access requests (except refresh); OREQ, IACK; DMA request	Cleared	Cleared
Refresh timer; refresh request; refresh address; mode register	Default values	No change
IPPRST pin	0 (active)	0 (active)

Figure 21 shows a typical system configuration using the μPD9305 with several ImPPs.

Figure 21. Typical System Configuration



NEC

SPEECH PROCESSORS

3

Section 3
Speech Processors

μPD7730/77C30
ADPCM Speech Encoder/Decoder

Description

The μPD7730/77C30 is a large scale integration (LSI) single-chip digital signal processor, which compresses and decompresses digitized speech signals. It is a speech encoder/decoder that converts pulse code modulated audio to and from adaptive differential pulse code modulation (ADPCM). The μPD7730/77C30 encodes pulse coded modulation (PCM) data into ADPCM data, and decodes ADPCM data into PCM data. The μPD7730/77C30 is ideal for office automation applications, such as voice store and forward systems, and for various telecommunication applications. It reduces voice transmission bandwidth and voice storage requirements by half (from 64 kb/s to 32 kb/s). Its robust ADPCM algorithm makes it well qualified for transmission applications and the fact that it compresses speech by half makes it suitable for store and forward applications.

The μPD7730 and μPD77C30 are functionally identical, but the power requirement for the CMOS μPD77C30 is lower than that of the NMOS μPD7730. Both devices are housed in plastic DIP; the temperature range of the low power NMOS device is −10 to +70°C and the CMOS device is −40 to +85°C.

The maximum clock (CLK) frequency for the μPD7730/77C30 is 8.33 MHz, which corresponds to a CLK cycle time of 120 ns.

The μPD7730/77C30 accepts PCM data through its serial interface. The serial interface can be connected directly to a single-chip coder/decoder (CODEC) for digital μ-law PCM input/output or to a general purpose A/D or D/A converter for linear PCM code. This programmable serial interface supports both 8-bit logarithmic (μ-law) and 16-bit linear formats. The μPD7730/77C30 interfaces to the host CPU through a standard microprocessor bus interface.

If a clock frequency of 8.33 MHz is used to encode PCM data, then the μPD7730/77C30 requires 116 μs to process each sample, thus limiting the sampling frequency to 8.59 kHz. This implies that if the sample frequency is 8.0 kHz and the CLK is 8.33 MHz, then the internal algorithm will take approximately 93% of the time between samples. Serial data being shifted in or out has the full time between samples to accomplish the transfer of the data. This is because there is an internal buffer that is separate from the shift register and the serial input is internally read at the rising edge of the sample clock, while the next value is starting to be shifted in.

When the μPD7730/77C30 operates in the sample 4-bit encode mode, it never outputs the value 00H. However, when it is in the sample 4-bit decode mode, it can accept 00H as an input value and interpret it the same as an input value of 88H.

The μPD7730/77C30 performs as a intelligent peripheral device and is controlled and programmed from the host processor. The μPD7730/77C30 offers toll quality (equivalent quality to 56 kb/s μ-law PCM) speech meeting the CCITT recommendations G.712.

Features

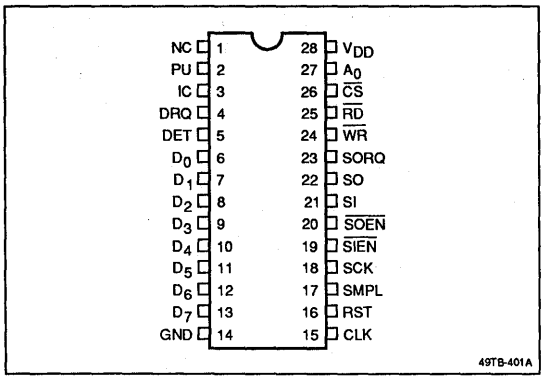
- Half-duplex ADPCM encoder or decoder
- Compression data rate
  - 32 kb/s/8 kHz sampling/4-bit data
  - 24 kb/s/8 kHz sampling/3-bit data
- Byte data (2x ADPCM data) handling
- Robust adaptation scheme for quantizer and predictors
- Selectable functions
  - Encoder /decoder operating mode
  - ADPCM data length 3 or 4 bit
  - A/D and D/A conversion μ-law or linear
- Presentable voice detection threshold
- Standard microprocessor interface to the host CPU
- Easy interface to PCM combo
- Toll quality speech at 32 kb/s (meets CCITT recommendations G.712)
- Single +5 V power supply
- Low power CMOS technology (μPD77C30) NMOS technology (μPD7730)
- Clock frequency 8.192 MHz maximum
- 28-pin plastic DIP
- 44-pin PLCC

Ordering Information

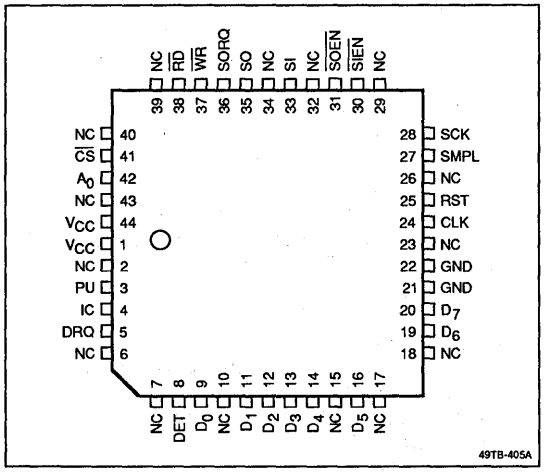
Part Number	Type	Package
μPD7730C	NMOS	28-pin plastic DIP (600 mil)
μPD77C30C	CMOS	28-pin plastic DIP (600 mil)
μPD77C30L	CMOS	44-pin PLCC

Pin Configuration

28-Pin Plastic DIP



44-Pin PLCC



Pin Identification

Symbol	I/O	Function
Host System Interface		
A_0	In	Address 0 (register select): This input selects internal registers. A high input selects the status register. A low input selects the data register.
D_7-D_0	I/O	Data Bus: This three-state bidirectional data bus interfaces with the host CPU data bus.
$\overline{CS}$	In	Chip select: This input enables the $\overline{RD}$ and $\overline{WR}$ signals.

Pin Identification (cont)

Symbol	I/O	Function
DET	Out	Signal detect: This output is asserted when the input audio signal level exceeds the threshold level specified.
DRQ	Out	Data request: This output requests data transfer between the μPD7730/77C30 and host CPU. In encoder mode, and ADPCM data read is requested. In decoder mode, and ADPCM data write is requested. (DRQ will not work unless encoder or decoder mode is specified.) The data request status can also be checked by polling the RQM bit of the status register.
$\overline{RD}$	In	Read signal: This input controls data transfer from the μPD7730/77C30 to the host CPU.
$\overline{WR}$	In	Write signal: This input controls data transfer from the host CPU to the μPD7730/77C30.
A/D-D/A Interface		
SCK	In	Serial clock: This input provides timing for transfer of serial data to/from the A/D and D/A converter.
SI	In	Serial input: Serial data input.
$\overline{SIEN}$	In	Serial input enable: This input enables data transfer on the SI pin. If not used, tie to $\overline{SOEN}$, $\overline{SIEN}$ must be asserted for the μPD7730/77C30 to recognize an operation command.
SO	Out	Serial output: Serial data output.
$\overline{SOEN}$	In	Serial output enable: This input enables data transfer on the SO pin. If not used, tie to $\overline{SIEN}$.
SORQ	Out	Serial output request: This output indicates that serial request output data is ready for transfer at the SO pin.
Circuit Control		
CLK	In	Clock: 8.192 MHz TTL clock input.
GND	In	Ground.
IC	—	Internal connection: This pin is connected internally and should be left open.
NC	—	No connection: This pin is not connected.
PU	—	Pull up: Pull this pin up to V_{DD}.
RST	In	Reset: A high input to this pin initializes the μPD7730/77C30.
SMPL	In	Sample: This input determines the rate at which the μPD7730/77C30 processes ADPCM data. This rate must equal the sampling clock of the A/D and D/A converter. SMPL must be active for the μPD7730/77C30 to recognize an operation command.
V_{DD}	In	+5-volt power supply.

Absolute Maximum Ratings

T_A = 25°C

Supply voltage, V_{DD}	
7730	−0.5 V to +7.0 V
77C30	−0.5 V to +7.0 V
Input voltage, V_I	
7730	−0.5 V to +7.0 V
77C30	−0.5 V to V_{DD} +0.5 V
Output voltage, V_O	
7730	−0.5 V to +7.0 V
77C30	−0.5 V to V_{DD} +0.5 V
Operating temperature, T_{OPT}	
7730	−10 to +70°C
77C30	−40 to +85°C
Storage temperature, T_{STG}	
7730	−65 to +150°C
77C30	−65 to +150°C

Exposure to Absolute Maximum Ratings for extended periods may affect device reliability; exceeding the ratings could cause permanent damage

Capacitance

T_A = 25°C, V_{DD} = 0 V

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
CLK, SCK capacitance	C_ϕ			20	pF	
Input capacitance	C_I			10	pF	fc = 1 MHz
Output capacitance	C_O			20	pF	

DC Characteristics

T_A = −10 to +70°C; V_{DD} = +5 V ±5%

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
Input voltage low	V_{IL}					
7730		−0.5		0.8	V	
77C30		−0.3		0.8	V	
Input voltage high	V_{IH}					
7730		2.0		V_{CC} +0.5	V	
77C30		2.2		V_{CC} +0.3	V	
CLK input voltage low	V_{OL}					
7730		−0.5		0.45	V	
77C30		−0.3		0.45	V	
CLK input voltage high	V_{OH}					
7730		3.5		V_{CC} +0.5	V	
77C30		3.5		V_{CC} +0.3	V	
Output voltage low	V_{OL}			0.45	V	I_{OL} = 2.0 mA
Output voltage high	V_{OH}	2.4			V	I_{OH} = −400 μA
Input leakage current high	I_{LIL}			−10	μA	V_I = 0 V
Input leakage current high	I_{LIH}			10	μA	V_I = V_{DD}
Output leakage current low	I_{LOL}			−10	μA	V_O = 0.47 V
Output leakage current high	I_{LOH}			10	μA	V_O = V_{DD}
Supply current	I_{DD}					
7730		180		280	mA	
77C30		24		40	mA	

3

AC Characteristics

T_A = −10 to +70°C; V_{DD} = +5 V ±5%

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
CLK cycle time	ϕ_{CY}					
7730		122		2000	ns	
77C30		120		2000	ns	
CLK pulse width	ϕ_D	60			ns	
CLK rise time	ϕ_r			10	ns	(Note 1)
CLK fall time	ϕ_f			10	ns	(Note 1)
A_0, $\overline{CS}$ set time for $\overline{RD}$	t_{AR}	0			ns	
A_0, $\overline{CS}$ hold time for $\overline{RD}$	t_{RA}	0			ns	
$\overline{RD}$ pulse width	t_{RR}	250			ns	
A_0, $\overline{CS}$ set time for $\overline{WR}$	t_{AW}	0			ns	
A_0, $\overline{CS}$ hold time for $\overline{WR}$	t_{WA}	0			ns	
$\overline{WR}$ pulse width	t_{WW}	250			ns	
Data set time for $\overline{WR}$	t_{DW}	150			ns	
Data hold time for $\overline{WR}$	t_{WD}	0			ns	
$\overline{RD}$, $\overline{WR}$ recovering time	t_{RV}	250			ns	
SCK cycle time	t_{SCY}	480		DC	ns	
SCK pulse time	t_{SCK}	230			ns	
SCK rise time	t_{rSC}			20	ns	
SCK fall time	t_{fSC}			20	ns	
$\overline{SOEN}$ set time for SCK	t_{SOC}	50		t_{SCY} −30	ns	
$\overline{SOEN}$ hold time for SCK	t_{CSO}	30		t_{SCY} −50	ns	
$\overline{SIEN}$, SI set time for SCK	t_{DC}	55		t_{SCY} −30	ns	
$\overline{SIEN}$, SI hold time for SCK	t_{CD}	30		t_{SCY} −55	ns	
$\overline{SIEN}$, $\overline{SOEN}$ pulse width high	t_{HS}	122			ϕ_{CY}	
RST pulse width	t_{RST}	4			ϕ_{CY}	
SMPL pulse width	t_{SMPL}	8			ϕ_{CY}	
Delay time between SMPL and $\overline{SIEN}$ ($\overline{SOEN}$)	t_{DX}	−1	0	1	μs	
Data access time for $\overline{RD}$	t_{RD}			150	ns	C_L = 100 pF
Data float time for $\overline{RD}$	t_{DF}	10		100	ns	C_L = 100 pF
SORQ delay	t_{DRQ}	30		150	ns	C_L = 50 pF

AC Characteristics (cont)

T_A = −10 to +70°C; V_{DD} = +5 V ±5%

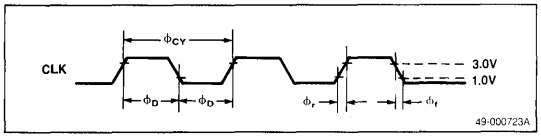
Parameter	Symbol	Min	Typ	Max	Unit	Conditions
SO delay time	t_{DCK}			150	ns	
SO delay time for SORQ	t_{DZRQ}	20		300	ns	
SO delay time for SCK	t_{DZSC}	20		300	ns	
SO delay time for $\overline{SOEN}$	t_{DZE}	20		180	ns	
SO float time for $\overline{SOEN}$	t_{HZE}	20		200	ns	
SO float time for SCK	t_{HZSC}	20		300	ns	
SO float time for SORQ	t_{HZRQ}	70		300	ns	

Notes:

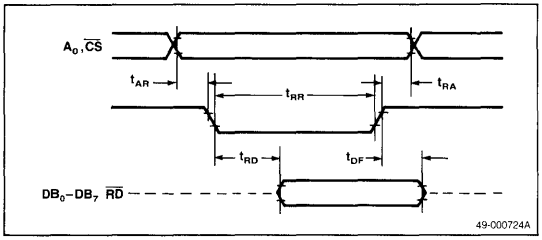
(1) AC timing measuring point voltage = 1.0 V and 3.0 V.

Timing Waveforms

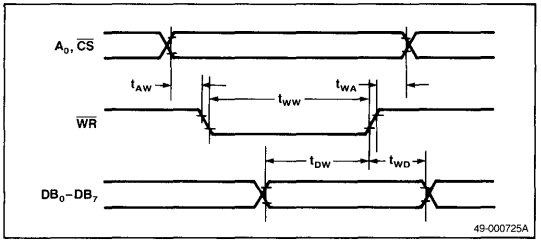
Clock



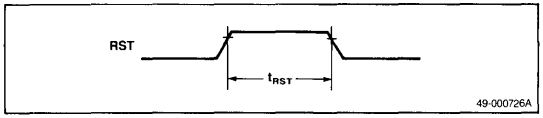
Read Operation



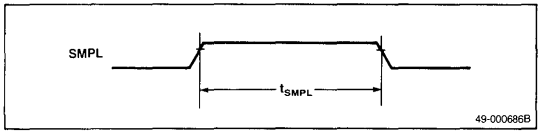
Write Operation



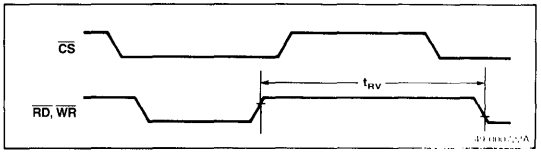
Reset



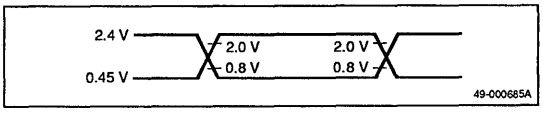
Sample



Read/Write Cycle Timing



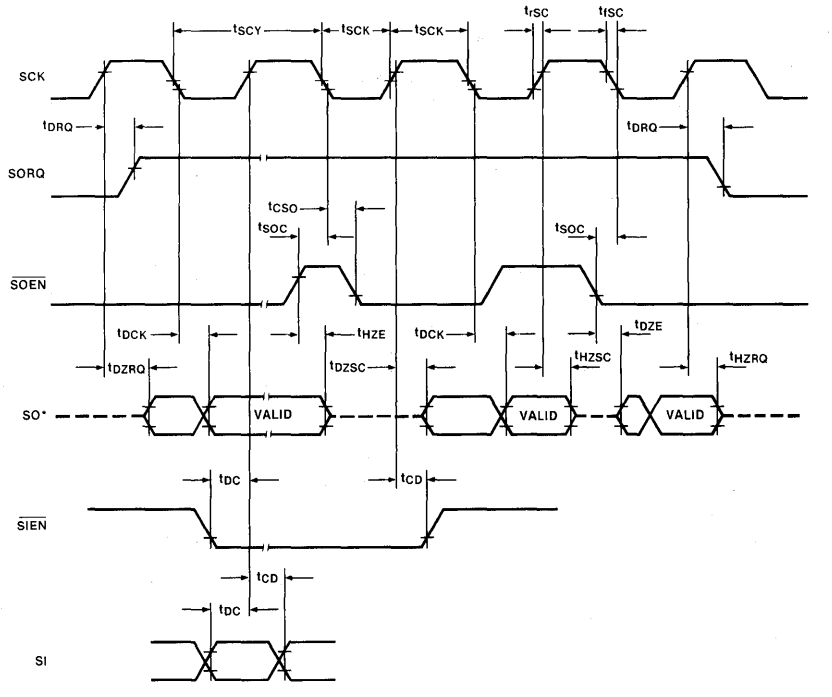
AC Waveform Measurement Points (except CLK)



3

Timing Waveforms (cont)

Serial Input/Output Timing

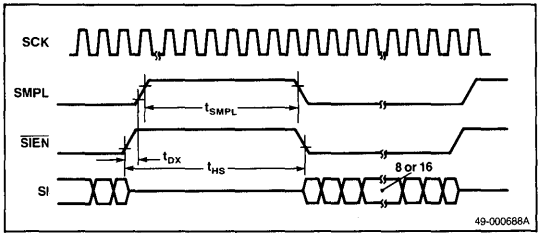


Note: In SO, the data output at the rising edge is valid, while data output at other times is invalid. Therefore, the most critical data set-up and hold times are:

Set-up time = $t_{SCK} - t_{DCK}$
Hold time = t_{HZRQ}

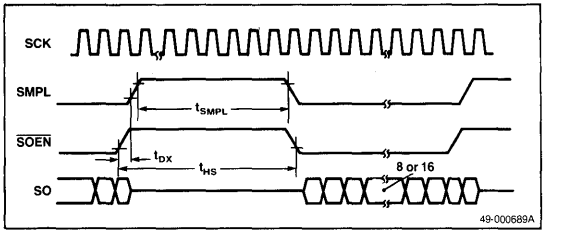
83-003663C

Serial Input Timing



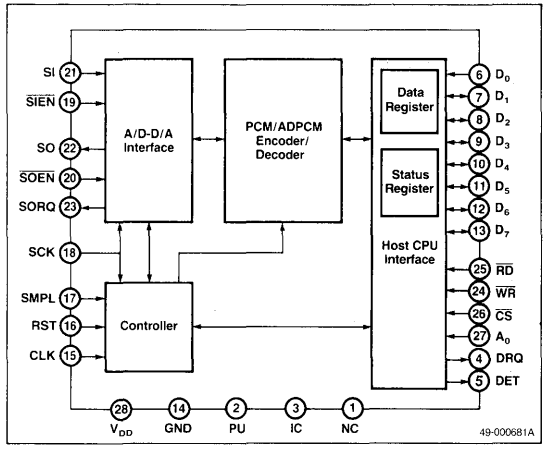
49-000688A

Serial Output Timing



49-000689A

Block Diagram



FUNCTIONAL DESCRIPTION

The μPD7730/77C30 has the following functional units:

- A/D-D/A interface
- PCM/ADPCM encoder/decoder
- Controller
- Data register
- Status register
- Host CPU interface

The ADPCM method is a medium bandwidth coding technique that represents speech waveforms. The specific ADPCM used employs a robust adaptation scheme for a quantizer and predictor to withstand transmission bit errors. Figure 1 shows the block diagram of the algorithm. The algorithm uses a backward adaptive quantizer and a fixed predictor so it never generates unstable poles in a decoder transfer function. This approach guarantees the stability of the decoder even with transmission errors.

The μPD7730/77C30 can operate in either encoder or decoder mode, and can only be set to one of the two modes at a time; it cannot handle simultaneous encoding and decoding. In encoder mode, the μPD7730/77C30 accepts either linear or μ-law PCM data from its serial voice interface, encodes it to ADPCM data format, and passes the ADPCM data through the parallel data bus to the host system. In decoder mode, the μPD7730/77C30 receives ADPCM data from the host CPU, decodes it to either linear or μ-law format, and sends it to the output port of the serial interface.

The μPD7730/77C30 has serial interfaces that can connect directly to a single-chip PCM CODEC. It interfaces easily to a host CPU through its parallel bus. With its standard microprocessor bus interface, the μPD7730/77C30 can be viewed as a complex peripheral circuit. Figure 2 shows a typical system configuration.

Figure 1. Algorithm Block Diagram

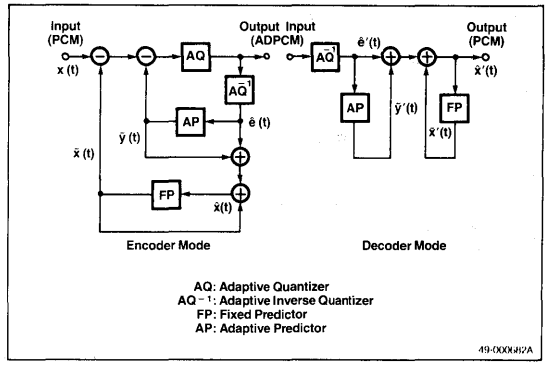
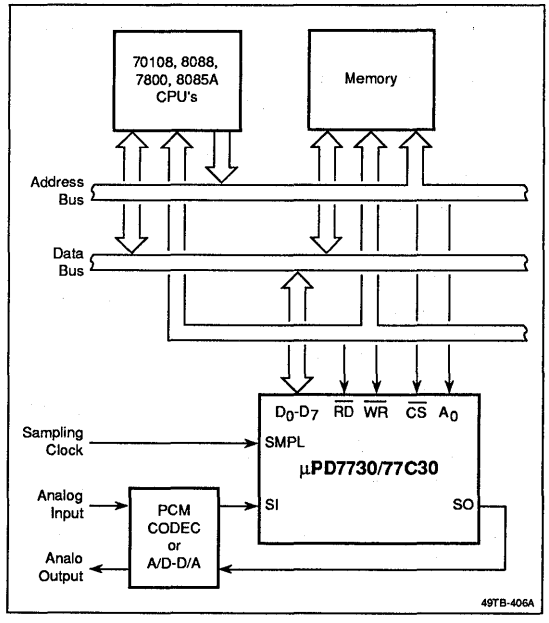


Figure 2. Typical System Configuration



OPERATIONAL DESCRIPTION

Host CPU Interface

In order to transfer ADPCM data, commands, and status, the μPD7730/77C30 interfaces with the host CPU via D_0-D_7 and through control lines $\overline{CS}$, A_0 $\overline{WR}$, and $\overline{RD}$. $\overline{CS}$ enables $\overline{RD}$ and $\overline{WR}$. A_0 selects either the data or status register. A low input to A_0 selects the data register. This read/write register handles both commands and ADPCM data transfer. A high input to A_0 selects the status register, a read-only register that the CPU reads to determine the state of the μPD7730/77C30.

Parallel I/O Operation

Table 1 shows the status of the $\overline{CS}$, A_0 $\overline{WR}$, and $\overline{RD}$ pins during parallel I/O operation. Figures 3 and 4 are timing diagrams that show the read and write operations for the host CPU interface with the μPD7730/77C30.

The RQM bit in the status register and the DRQ pin are the principal handshake signals. Their characteristics follow.

RQM characteristics:

- The μPD7730/77C30 requests a data transfer to or from a host CPU by setting the RQM signal to a high level.
- After ADPCM data has transferred, the RQM goes low at the rising edge of WR or RD pulse.
- After the threshold data has transferred, RQM goes low at the second rising edge of the WR pulse.
- Reading the status register via the data bus does not reset RQM.

DRQ characteristics:

- Except during initialization, the μPD7730/77C30 DRQ signal is high, when the status register bit RQM is set to indicate that an ADPCM data transfer to or from the host CPU is required.
- DRQ goes low after each encoding or decoding operation is completed.
- Because DRQ remains low throughout initialization it cannot be used for handshaking during initialization.
- The DRQ signal may be connected to an interrupt pin of a host CPU.

Two different approaches can be used for servicing ADPCM I/O requests by the μPD7730/77C30. The first approach is for the host CPU to repeatedly poll the status register until RQM = 1 is found. The second approach is for the DRQ pin to go high forcing an interrupt of the host CPU. In either case the host CPU then reads the data register to capture the ADPCM data.

Status Register

Figure 5 shows the format of the status register.

Operation Command

Following a power-on reset, the host CPU polls the RQM bit in the status register. When the RQM bit is set, the host CPU can send an operation command to the data register, as shown in figure 6.

Table 1. Control Line States

$\overline{CS}$	A_0	$\overline{WR}$	$\overline{RD}$	Function
1	X	X	X	No effects on internal operation.
X	X	1	1	D_0-D_7 are high impedance.
0	0	0	1	Data from D_0-D_7 is latched to the data register.
0	0	1	0	Contents of the data register are output to D_0-D_7.
0	1	0	1	Illegal operation.
0	1	1	0	Contents of the status register are output to D_0-D_7.

X = don't care.

Figure 3. ADPCM Data Read Timing

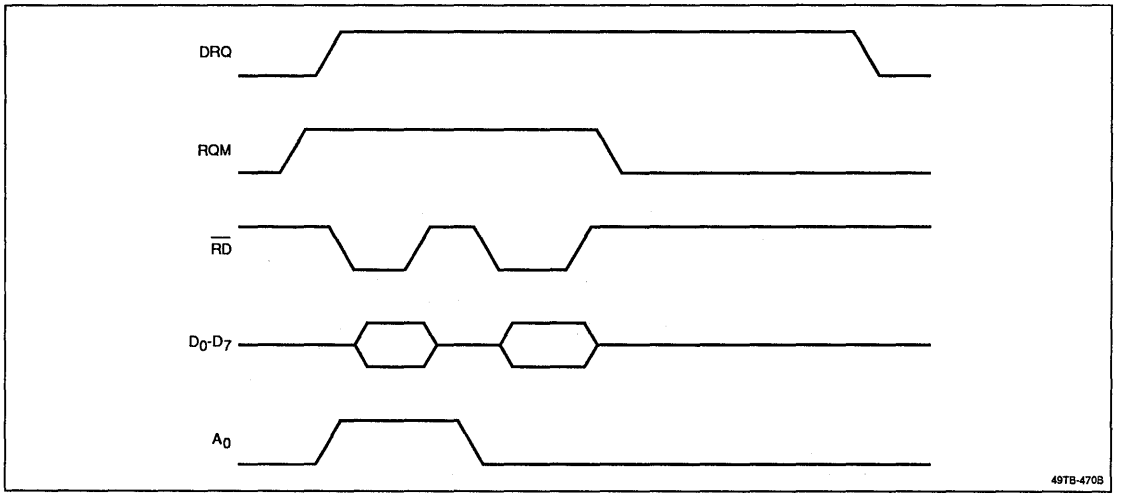


Figure 4. ADPCM Data Write Timing

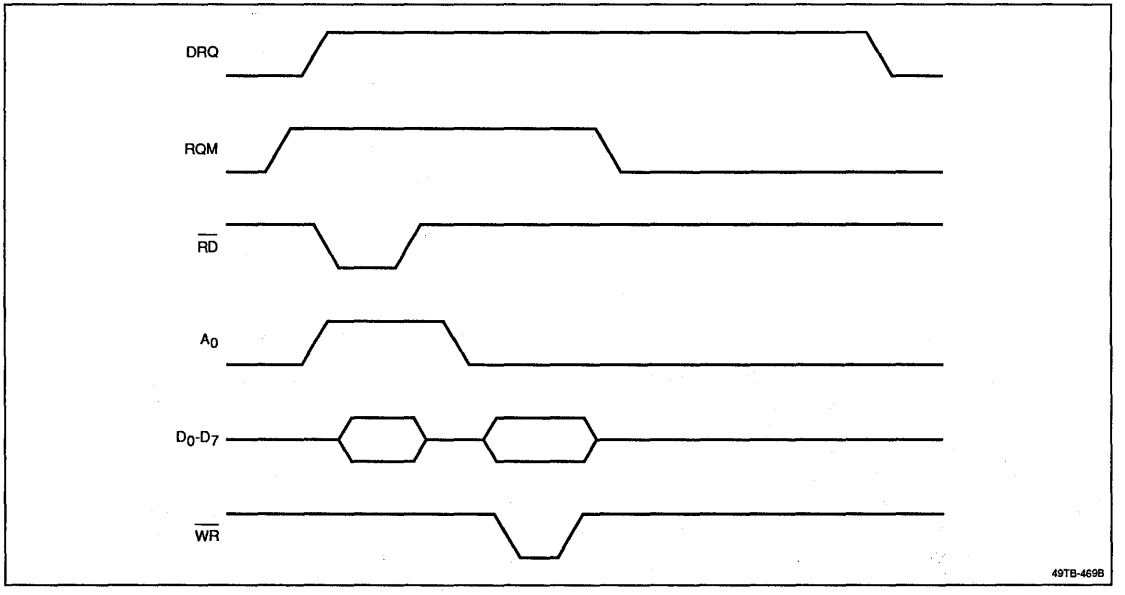
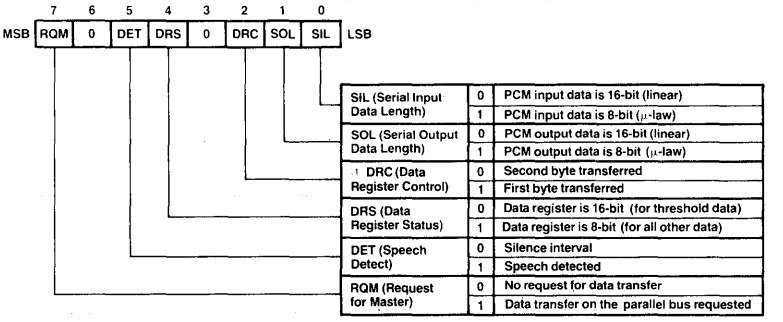


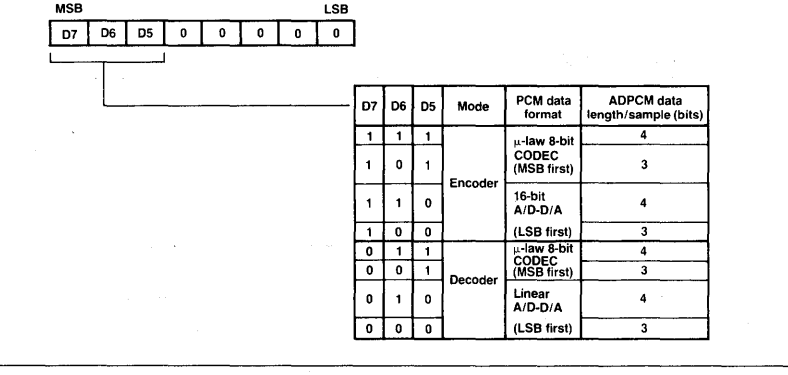
Figure 5. Status Register Format



Note: 1. DRS indicates the status of data transfers when the Data register is configured as 16-bit (DRC 0).

49-000690B

Figure 6. Operation Command



D7	D6	D5	Mode	PCM data format	ADPCM data length/sample (bits)
1	1	1	Encoder	μ-law 8-bit CODEC (MSB first)	4
1	0	1			3
1	1	0		16-bit A/D-D/A	4
1	0	0		(LSB first)	3
0	1	1	Decoder	μ-law 8-bit CODEC (MSB first)	4
0	0	1			3
0	1	0		Linear A/D-D/A	4
0	0	0		(LSB first)	3

49-000691B

Power-on and Reset

The μPD7730/77C30 operates on a single-phase, 50-50 duty cycle clock at 8 MHz. At power-on, asserting the RST pin for at least 3 clock cycles initializes the device, making it ready for an operation command from the host CPU. After the μPD7730/77C30 receives the command, it stays in the specified operational mode until the next hardware reset (high level on RST). Thus, to change the μPD7730/77C30 into different modes, reset it before writing an operation command.

Initialization and Threshold Data

See figure 7 for the initialization sequence for the encoder mode. See figure 8 for the initialization sequence for the decoder mode. During initialization signal SMPL is ignored, but the SCK and SIEN signals must be active. This is because the μPD7730/77C30 internal code checks that the serial data is being transferred in before it accepts the mode byte. Also, it is of no consequence whether or not serial input data is valid during initialization. This is true whether the μPD7730/77C30 is placed in encoder or decoder mode.

A hardware reset must be issued before a mode byte can be sent, even when the μPD7730/77C30 is being powered up. Also, to change modes (e.g., encoder to decoder mode), a hardware reset signal must be issued. In either of the above cases, the reset signal must be held active for a minimum of 3 CLK cycles to guarantee that the mode byte will be accepted. As explained below, the RQM bit of the status register should be used for data transfer handshaking, especially during initialization. The status register at a CLK frequency of 8.192 MHz is not valid until 190 μs after the trailing edge of the reset pulse, and should not be read until after that time interval.

The DRQ signal does not always follow the state of the RQM bit in the status register. In particular, the DRQ signal remains low throughout initialization. Therefore, it is essential during initialization to use the RQM bit of the status register for handshaking. The DRQ signal is intended for interrupting the host CPU so that it will transfer ADPCM data after initialization. The DRQ signal remains high until the encoding or decoding operation of the μPD7730/77C30 is complete. The RQM bit, in contrast, is intended for data transfer handshaking and is reset after each data port transfer is complete.

When the μPD7730/77C30 first enters the decoder mode the RQM bit is already set and the first byte of data sent to the μPD7730/77C30 will not be decoded properly. To avoid losing the first speech sample, a dummy first byte of ADPCM should be sent.

If the operation command places the μPD7730/77C30 in encoder mode, the next two bytes sent to the data register are the threshold data. The RQM bit establishes the data transfer signaling. In decoder mode, no threshold data is expected. The threshold data sets the level of the audio signal at which the DET pin is asserted. Figure 9 shows the format for the threshold data. Figure 10 shows how to determine the threshold data.

The μPD7730/77C30 asserts DET when the serial input audio signal exceeds the threshold level specified by the threshold data. Many silent segments exist in normal speech signals; memory storage can be used more efficiently if these segments are omitted. The host CPU can perform silent segment compression by using DET. The energy levels of 16 previous audio samples determine the state of DET. Thus DET changes at a 2 ms (16 x 8 kHz sampling) time frame. Bit 5 of the status register reflects the state of DET.

Figure 7. Encoder Mode Initialization Sequence

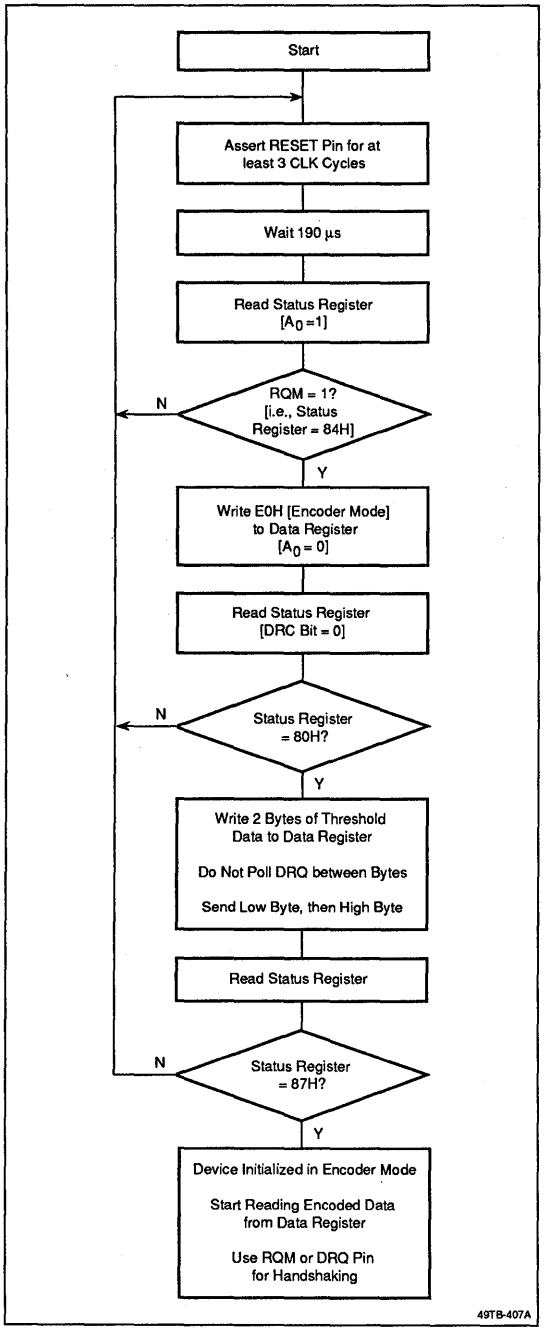


Figure 8. Decoder Mode Initialization Sequence

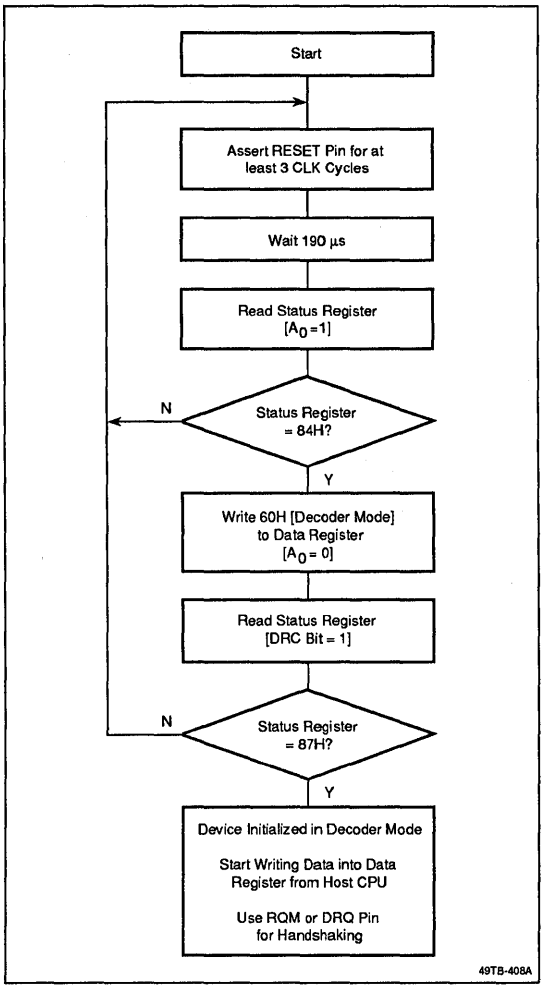


Figure 9. Threshold Data

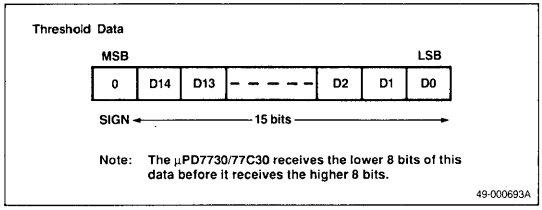
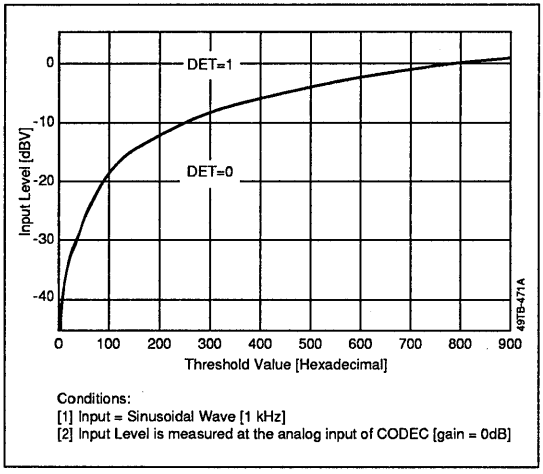


Figure 10. Typical Relationship Between Input Level and Threshold Value



ADPCM Data

In encoder mode, the μPD7730/77C30 generates one ADPCM sample (3 or 4 bits long) for each PCM sample input (8 or 16 bits long). In decoder mode, the reverse operation is performed: the μPD7730/77C30 generates one PCM sample for each ADPCM sample input. To allow efficient data transfer to and from the host CPU, two ADPCM samples are packed into one byte and transferred at the rate of 1 byte per every 2 samples. Figure 11 illustrates the ADPCM data formats for 3 bits/sample and 4 bits/samples.

The DRQ pin initiates ADPCM data transfer. In encoder mode, this pin is asserted when ADPCM data in the data register is ready to be read by the CPU. This pin is cleared after the host CPU reads the data, and is reasserted when the next byte of ADPCM data becomes available. In decoder mode, this pin serves as the data request to the host for the next byte of ADPCM data to be sent to the data register. After the host CPU writes the ADPCM data, this pin is cleared. The host CPU cannot send another byte to the μPD7730/77C30 until this pin is set again. (Note that the DRQ pin will not work until the μPD7730/77C30 is placed in encoder or decoder mode.)

The ADPCM data transfer is acknowledged by the RQM bit in the status register. The RQM Bit is set when transfer to the host is requested for ADPCM data, and is reset when the host read/write is complete.

Serial PCM Interface

The serial PCM interface can be connected directly to a CODEC. SMPL, SCK, $\overline{\text{SIEN}}$, SI, SORQ, $\overline{\text{SOENS}}$, and SO control the PCM interface.

SMPL is the sampling clock input. This signal must equal the frequency of the sampling clock of the CODEC or the A/D-D/A interface. SMPL is asserted after the completion of serial data transfers. Thus SMPL signals the μPD7730/77C30 firmware to initiate processing of the next byte of ADPCM data. SMPL is rising-edge triggered, but must be held high for at least 8 clock cycles. Since it is edge-triggered, SMPL does not need to be released until the next sampling cycle.

SCK determines the timing of the serial input and output. When the μPD7730/77C30 has data to send to the serial interface, SORQ goes high. The data is then clocked out to the SO pin serially at the falling edge of SCK, to be valid for the next rising edge. When serial data is ready to be sent to the μPD7730/77C30 $\overline{\text{SIEN}}$ is asserted externally, and data at the SI pin is clocked in at the rising edge of SCK.

Figure 12 illustrates an example of the serial interface using a combined filter and CODEC (COMBO) chip, the μPD9514. This chip provides both the low pass filtering function and the conversion from an analog signal to digital PCM μ-law representation. The timing controller provides the proper timing relationship between the COMBO and the μPD7730/77C30.

Figure 11. ADPCM Data Format

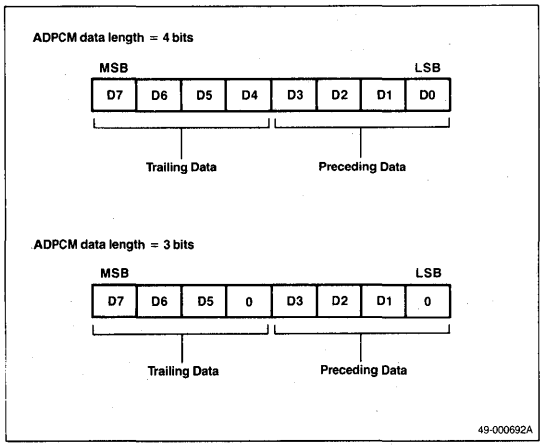
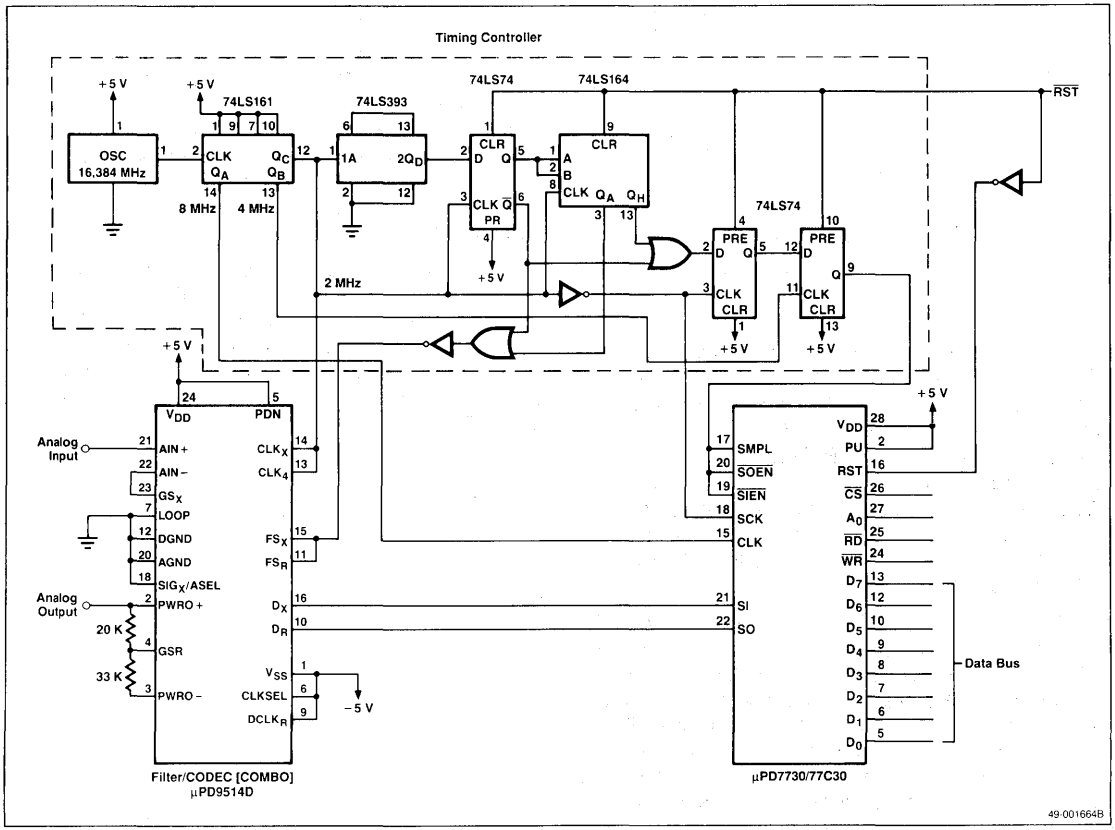


Figure 12. Serial Interface Using a COMBO



Description

The μPD7755, μPD7756, and μPD7757 are speech synthesis LSI devices that utilize the adaptive differential pulse coded modulation (ADPCM) coding method to produce high-quality, natural speech synthesis. By combining phoneme classification with the ADPCM method, the device achieves a compressed bit rate that can synthesize sound effects and melodies in addition to speech sound. A built-in speech data ROM allows synthesis of messages up to 4 seconds (μPD7755), 12 seconds (μPD7756), or 24 seconds (μPD7757) long. A wide range of operating voltages, a compact package, and a standby function permit application to the μPD7755/56/57 in a variety of speech synthesis systems, including battery-driven systems.

The μPD77P56 is one-time programmable (OTP) version of the μPD7756.

Features

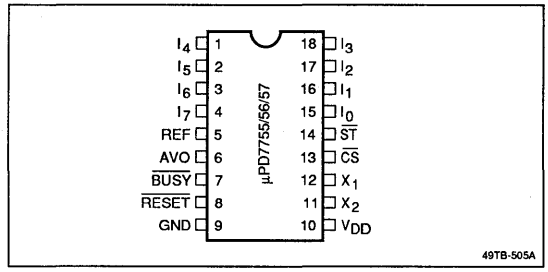
- High quality speech synthesis using ADPCM method
- Low bit rates (8K to 32K b/s) using a combination of ADPCM and phoneme methods
- D/A converter with 9-bit resolution, unipolar current waveform output
- Built-in speech data ROM
  - μPD7755: 96K bits
  - μPD7756/P56: 256K bits
  - μPD7757: 512K bits
- Standby function
- Current consumption in standby mode: 1 μA typ (V_{DD} = 3 V)
- Circuit to eliminate popcorn noise when entering or releasing standby mode
- Wide operating voltage range: 2.7 to 5.5 V
- CMOS technology
- 18-pin plastic DIP or 24-pin plastic SOP
- One-time programmable (OTP) version of μPD7756 available in a 20-pin plastic DIP or a 24-pin plastic SOP

Ordering Information

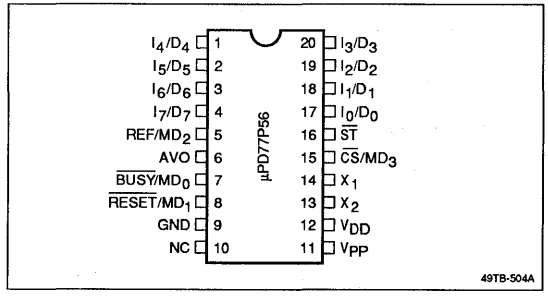
Part Number	Package Type	ROM (bits)
μPD7755C	18-pin plastic DIP (A, C outline)	96K
μPD7755G	24-pin plastic SOP	
μPD7756C	18-pin plastic DIP (A, C outline)	256K
μPD7756G	24-pin plastic SOP	256K (OTP)
μPD77P56CR	20-pin plastic DIP	256K
μPD77P56G	24-pin plastic SOP	256K (OTP)
μPD7757C	18-pin plastic DIP (SA outline)	512K
μPD7757G	24-pin plastic SOP	

Pin Configurations

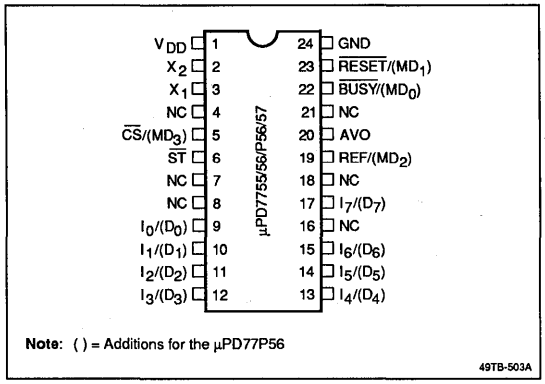
18-Pin Plastic DIP



20-Pin Plastic DIP



24-Pin Plastic SOP



Note: ( ) = Additions for the μPD77P56

49TB-503A

Pin Identification

Symbol	Name
I_0-I_7	Message select code input
REF	D/A converter reference current input
AVO	Analog voice output
$\overline{BUSY}$	Busy output
$\overline{RESET}$	Reset input
GND	Ground
V_{DD}	Power
X2, X1	Clock
$\overline{CS}$	Chip select input
$\overline{ST}$	Start input
D_0-D_7	PROM I/O data bus
MD_0-MD_3	Operation mode selection input for PROM
V_{PP}	12.5 V PROM voltage application
NC	No connection

Pin Functions

I_0-I_7 (Message Select Code)

I_0-I_7 input the message number of the message to be synthesized. The inputs are latched at the rising edge of the $\overline{ST}$ input. Unused pins should be grounded. In standby mode, these pins should be set high or low. If they are biased at or near typical CMOS switch input, they will drain excess current.

$\overline{CS}$ (Chip Select)

When the $\overline{CS}$ input goes low, $\overline{ST}$ is enabled.

$\overline{ST}$ (Start)

Setting the $\overline{ST}$ input low while $\overline{CS}$ is low will start speech synthesis of the message in the speech ROM locations addressed by the contents of I_0-I_7. If the device is in standby mode, standby mode will be released.

$\overline{BUSY}$ (Busy)

$\overline{BUSY}$ outputs the status of the μPD7755/56/57. It goes low during speech decode and output operations. When $\overline{ST}$ is received, $\overline{BUSY}$ goes low. While $\overline{BUSY}$ is low, another $\overline{ST}$ will not be accepted. In standby mode, $\overline{BUSY}$ becomes high impedance. This is an active low output.

AVO (Analog Voice Output)

AVO output synthesized speech from the D/A converter. This is a unipolar sink-load current.

$\overline{RESET}$ (Reset)

The $\overline{RESET}$ input initializes the chip. Use $\overline{RESET}$ following power-up to abort speech synthesis or to release standby mode. $\overline{RESET}$ must remain low at least 12 oscillator clocks. At power-up or when recovering from standby mode, $\overline{RESET}$ must remain low at least 12 more clocks after clock oscillation stabilizes.

X1, X2 (Clock)

Pins X1 and X2 should be connected to a 640 MHz ceramic oscillator. In standby mode, X1 goes low, and X2 goes high.

REF (D/A Converter Reference Current)

REF inputs the sink-load current that controls the D/A converter output. REF should be connected to V_{DD} via a resistor. In standby mode, REF becomes high impedance.

D_0-D_7 (Data Bus)

Eight-bit input/output data bus for PROM, when programming and verifying data.

MD_0-MD_3 (Mode Select Input)

Operation mode selection inputs for PROM, when programming and verifying data.

V_{PP} (PROM Power)

12.5 V high voltage application pin, for programming and verifying data to PROM.

GND (Ground)

Ground.

V_{DD} (Power)

+5 V power supply.

NC (No Connection)

These pins are not connected.

Operation Description

The clock pins should be connected to a ceramic oscillator at 640 MHz.

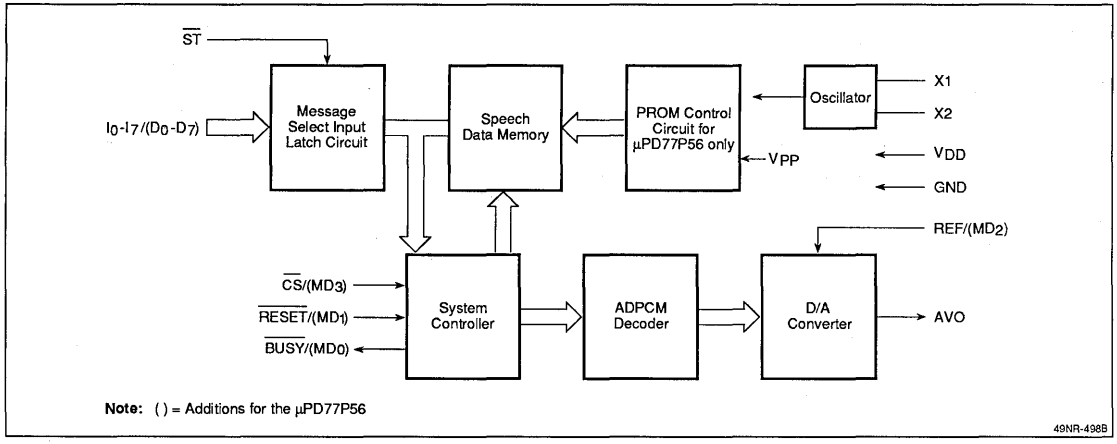
The $\overline{RESET}$ input pin is used to initialize the μPD7755/56/57. To reset, assert the pin for a minimum of 12 oscillator clock cycles.

The μDP7755/56/57 can operate with a wide range of supply voltage: 2.7 to 5.5 V. It also has a standby function; it goes to a standby mode when it has been idle (that is, when $\overline{CS}$, $\overline{ST}$, or $\overline{RESET}$ have not been asserted) for more than 3 seconds. The μDP7755/56/57 will automatically release from standby mode when $\overline{CS}$ and $\overline{ST}$ are asserted again, or when $\overline{RESET}$ is asserted.

The μDP7755/56/57 has a very simple message selection interface. A μDP7755/56/57 can store a maximum of 256 different messages and up to 12 (μDP7755), 12 (μDP7756), or 24 (μDP7757) seconds of speech. The message is selected by using the input pins I_0-I_7. The input selection is latched at the rising edge of $\overline{ST}$ when $\overline{CS}$ is asserted. When $\overline{ST}$ is asserted, $\overline{BUSY}$ will go low until the selected audio speech output is completed. While $\overline{BUSY}$ is low, a new $\overline{ST}$ will not be accepted.

The μDP7755/56/57 has an internal D/A converter that is a unipolar, current-output type with 9-bit resolution. The output current of the D/A can be controlled by the voltage applied at the REF pin.

Block Diagram



3

Absolute Maximum Ratings

T_A = 25 °C

Supply voltage, V_{DD}	−0.3 to +7.0 V
Input voltage, V_I	−0.3 to V_{DD} +0.3 V
Output voltage, V_O	−0.3 to V_{DD} +0.3 V
PROM Power voltage, V_{PP}	−0.3 to +13.5 V
PROM output current, I_O (AVO pin only)	50 mA
Operating temperature, T_{OPT}	−10 to +70 °C
Storage temperature, T_{STG}	−40 to +125 °C

Exposure to Absolute Maximum Ratings for extended periods may affect device reliability; exceeding the ratings could cause permanent damage.

Capacitance

T_A = 25 °C

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
Input capacitance	C_I			10	pF	fc = 1 MHz
Output capacitance	C_O			20	pF	

DC Characteristics

T_A = −10 to +70 °C; V_{DD} = 2.7 to 5.5 V; f_{OSC} = 640 MHz (μPD77P56 T_A = −40 to +85°C)

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
Input voltage high	V_{IH}	0.7 V_{DD}		V_{DD}	V	Common to I_0-I_7, $\overline{ST}$ $\overline{CS}$, $\overline{RESET}$
Input voltage low	V_{IL}	0		0.3 V_{DD}	V	Common to I_0-I_7, $\overline{ST}$, $\overline{CS}$, $\overline{RESET}$
Output voltage high	V_{OH}	V_{DD} −0.5		V_{DD}	V	$\overline{BUSY}$. I_{OH} = −100 μA
Output voltage low	V_{OL}	0		0.5	V	$\overline{BUSY}$. I_{OL} = 200 μA
Input leakage current	I_{LI}			3	μA	Common to I_0-I_7, $\overline{ST}$, REF $\overline{CS}$. 0 V ≤ V_{IN} ≤ V_{DD}
Output leakage current	I_{LO}			3	μA	$\overline{BUSY}$. 0 V ≤ V_O ≤ V_{DD} in standby mode
Supply current	I_{DD1}		0.8	2	mA	2.7 V ≤ V_{DD} ≤ 5.5 V
	I_{DD2}		1	20	μA	2.7 V ≤ V_{DD} ≤ 5.5 V in standby mode
	I_{DD3}		250	600	μA	2.7 V ≤ V_{DD} ≤ 3.3 V
	I_{DD4}		1	10	μA	2.7 V ≤ V_{DD} ≤ 3.3 V in standby mode
Reference input high current area (1)	I_{REF1}	140	250	440	μA	V_{DD} = 2.7 V, R_{REF} = 0 Ω
	I_{REF2}	500	760	1200	μA	V_{DD} = 5.5 V, R_{REF} = 0 Ω
Reference input low current area (1)	I_{REF3}	21	30	39	μA	V_{DD} = 2.7 V, R_{REF} = 50 kΩ
	I_{REF4}	68	78	88	μA	V_{DD} = 5.5 V, R_{REF} = 50 kΩ
D/A converter output current (1)	I_{AVO}	32	34	36	I_{REF}	2.7 V ≤ V_{DD} ≤ 5.5 V. V_{AVO} = 2.0 V, D/A input = 1FFH
D/A converter output leakage current	I_{LA}			±5	μA	0 V ≤ V_{AVO} ≤ V_{DD} in standby mode

Notes:

(1) See figure 1.

AC Characteristics

T_A = −10 to +70 °C; V_{DD} = 2.7 to 5.5 V; f_{OSC} = 640 MHz (μPD77P56 T_A = −10 to +85°C)

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
$\overline{\text{RESET}}$ pulse width	t_{RST}	18.5			μs	
$\overline{\text{ST}}$ set-up time	t_{RS}	12.5			μs	
$\overline{\text{ST}}$ pulse width	t_{CC1}	2			μs	2.7 V ≤ V_{DD} ≤ 5.5 V
	t_{CC2}	350			ns	4.5 V ≤ V_{DD} ≤ 5.5 V
Data set time	t_{DW1}	2			μs	2.7 V ≤ V_{DD} ≤ 5.5 V
	t_{DW2}	350			ns	4.5 V ≤ V_{DD} ≤ 5.5 V
Data hold time	t_{WD}	0			ns	
$\overline{\text{CS}}$ set-up time	t_{CS}	0			ns	
$\overline{\text{CS}}$ hold time	t_{SC}	0			ns	
CLK frequency	f_{OSC}	630	640	650	kHz	
$\overline{\text{BUSY}}$ output time (from $\overline{\text{ST}}$ and/or $\overline{\text{CS}}$)	t_{SBO}		6.25	10	μs	Operation mode
	t_{SBS}		4	80	ms	Standby mode, including oscillation start time (Note 1)
$\overline{\text{BUSY}}$ set time	t_{SB}		6.25	10	μs	Standby mode
Speech output start time	t_{SSO}		2.1	2.2	ms	Operation mode (from $\overline{\text{BUSY}}$)
	t_{SSS}		2.1	2.2	ms	Standby mode
D/A converter set-up time	t_{DA}		46.5	47	ms	Entering/releasing standby mode
$\overline{\text{BUSY}}$ float time	t_{BF}			15	μs	From end of speech output
$\overline{\text{BUSY}}$ output stop time	t_{RB}			9.5	μs	For $\overline{\text{RESET}}$ ↓
Standby transition time	t_{STB}		2.9	3	μs	From end of speech output

Notes:

(1) Ceramic resonators:
Kyocera Corp. KBR-640B (C1 = C2 = 150 pF).
Murata Mfg. Co. Ltd. CSB640P (C1 = C2 = 220 pF).
See figure 2.

Figure 1. Measuring Diagram for I_{REF} and I_{AVO}

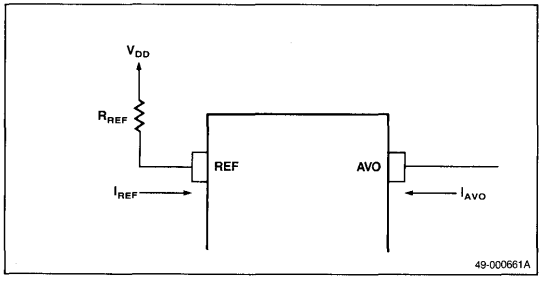
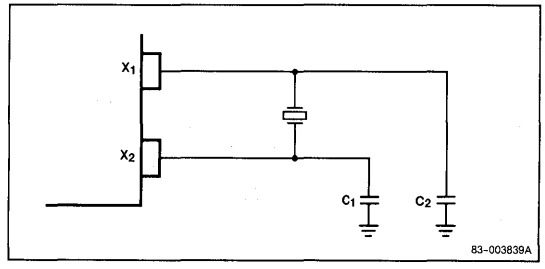
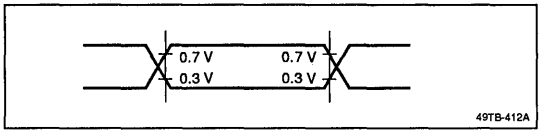


Figure 2. External Oscillator

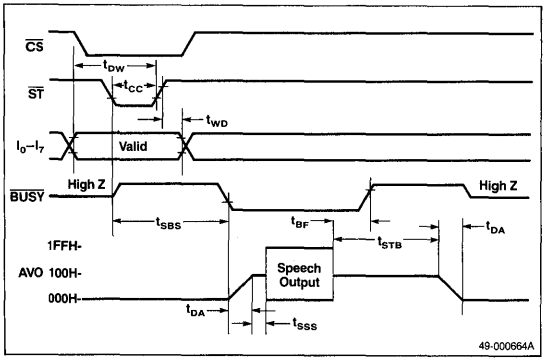


Timing Waveforms

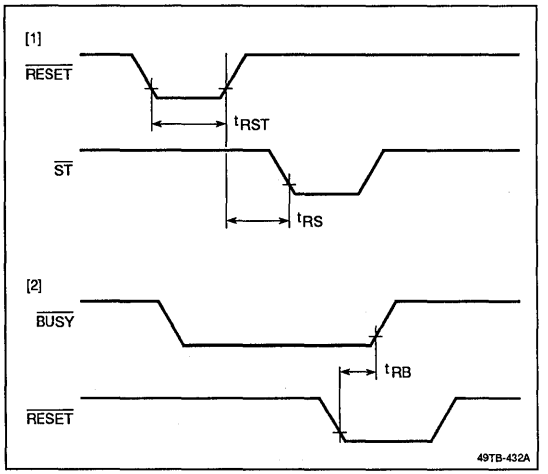
AC Waveform Measurement Points



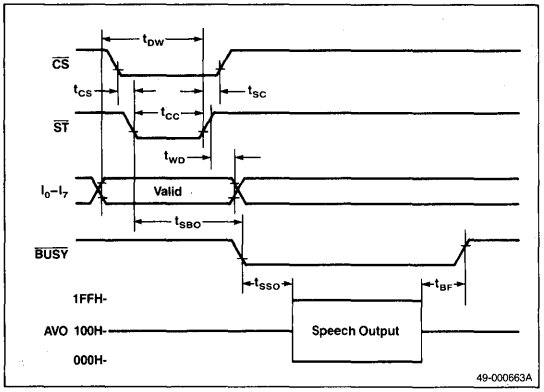
Standby Mode



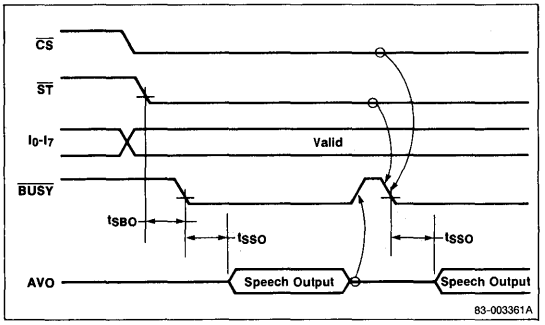
Reset Mode



Operating Mode (ST Input Pulse Mode)



Operating Mode (ST Input Hold Low Mode)



Description

The μPD7759 is a speech synthesis LSI with an external ROM that utilizes the adaptive differential pulse coded modulation (ADPCM) coding method to produce high-quality, natural speech synthesis. By combining phoneme classification with the ADPCM method, the device achieves a compressed bit rate that can synthesize sound effects and melodies in addition to speech sound. The μPD7759 can directly address up to 1M bits of external data ROM, or the host CPU can control the speech data transfer. The μPD7759 is also suitable for applications requiring small production quantities or long synthesized messages, and for emulating the μPD7755/56/57.

Features

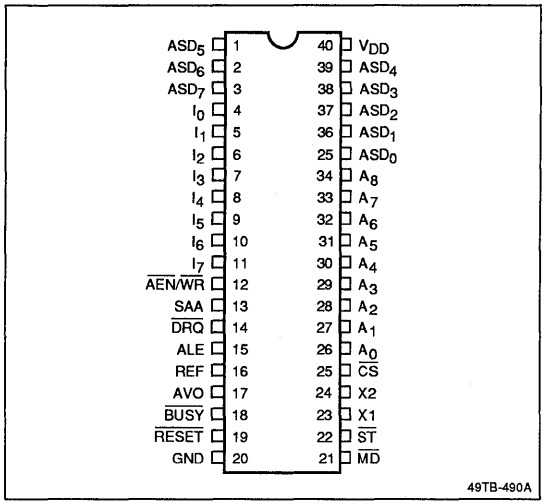
- High-quality speech synthesis using ADPCM method
- Low bit-rates (8 to 32 kb/s) realized by combined use of ADPCM and phoneme methods
- 4, 5, 6, or 8 MHz sampling frequency
- D/A converter with 9-bit resolution, unipolar current waveform output
- Up to 1M bits addressing for external data ROM
- Synthesizing time: 50 sec. typ
- Standby function
- Circuit to eliminate popcorn noise when entering or releasing standby mode
- Control signal interface; general purpose 4-or 8-bit CPU
- Wide operating voltage range; 2.7 to 5.5 V
- CMOS technology
- 40-pin plastic DIP; 52-pin plastic miniflat package

Ordering Information

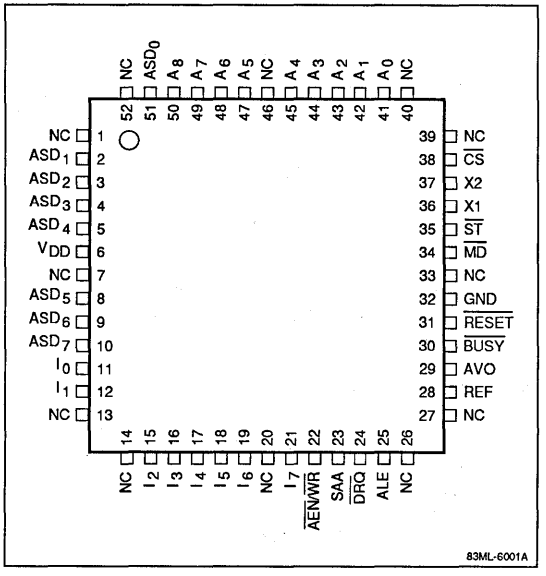
Part Number	Package Type
μPD7759C	40-pin plastic DIP
μPD7759GC	52-pin plastic miniflat

Pin Configurations

40-Pin Plastic DIP



52-Pin Plastic Miniflat Package



Pin Identification

Symbol	Function
ASD_0-ASD_7	Higher 8 bits of address output/speech data input (multiplexed)
I_0-I_7	Message select code input
$\overline{AEN}/\overline{WR}$	Address valid output/write strobe input for speech data
SAA	Directory data output address valid
$\overline{DRQ}$	Data request output
ALE	High address latch enable output
REF	D/A converter reference current input
AVO	Analog speech output
$\overline{BUSY}$	Chip busy output
$\overline{RESET}$	Reset input
GND	Ground
$\overline{MD}$	Mode select input (stand alone/slave)
$\overline{ST}$	Start synthesis strobe input
X1, X2	Ceramic resonator clock terminals
$\overline{CS}$	Chip select input
A_0-A_8	Lower 9 bits of address output for speech data
V_{DD}	+5 V power supply
NC	No connection

Pin Functions

ASD_0-ASD_7 (Address/Speech Data)

ASD_0-ASD_7 are the output lines for the higher 8 bits of the address signal and the input lines for speech data in the stand alone mode. In the slave mode, these are input lines only for speech data.

$\overline{AEN}/\overline{WR}$ (Address Enable Output/Write Signal Input)

$\overline{AEN}$ is high when the address signal is valid (stand alone mode). $\overline{WR}$ is the write input signal for speech data in the slave mode.

SAA (Start Address)

SAA indicates that the start address of a message stored in the directory of the data memory is being read out. It is ineffective in the slave mode.

ALE (Address Latch Enable)

This signal defines the higher address bit timing latched externally. It is ineffective in the slave mode.

$\overline{DRQ}$ (Data Request)

This is the data request output signal.

$\overline{MD}$ (Mode Select Input)

$\overline{MD}$ is low to specify slave mode operation. Transition between two operation modes is not accepted during synthesis or in the stand alone mode.

A_0-A_8 (Address Bus)

These are output lines for the lower 9 bits of the address bus. They are ineffective in the slave mode.

I_0-I_7 (Message Select Code)

I_0-I_7 input the message number of the message to be synthesized. The inputs are latched at the rising edge of the $\overline{ST}$ input. Unused pins should be grounded. In standby mode, these pins should be set high or low. If they are biased at or near typical CMOS switch input, they will drain excess current.

$\overline{CS}$ (Chip Select)

When the $\overline{CS}$ input goes low, $\overline{ST}$ is enabled.

$\overline{ST}$ (Start)

Setting the $\overline{ST}$ input low while $\overline{CS}$ is low will start speech synthesis of the message in the speech ROM locations addressed by the contents of I_0-I_7. If the device is in standby mode, standby mode will be released.

$\overline{BUSY}$ (Busy)

$\overline{BUSY}$ outputs the status of the μPD7759. It goes low during speech decode and output operations. When $\overline{ST}$ is received, $\overline{BUSY}$ goes low. While $\overline{BUSY}$ is low, another $\overline{ST}$ will not be accepted. In standby mode, $\overline{BUSY}$ becomes high impedance. This is an active low output.

AVO (Analog Voice Output)

AVO outputs synthesized speech from the D/A converter. This is a unipolar sink-load current.

$\overline{RESET}$ (Reset)

The $\overline{RESET}$ input initializes the chip. Use $\overline{RESET}$ following power-up to abort speech synthesis or to release standby mode. $\overline{RESET}$ must remain low at least 12 oscillator clocks. At power-up or when recovering from standby mode, $\overline{RESET}$ must remain low at least 12 more clocks after clock oscillation stabilizes.

X1, X2 (Clock)

Pins X1 and X2 should be connected to a 640-MHz ceramic oscillator. In standby mode, X1 goes low and X2 goes high.

REF (D/A Converter Reference Current)

REF inputs the sink-load current that controls the D/A converter output. REF should be connected to V_{DD} via a resistor. In standby mode, REF becomes high impedance.

GND (Ground)

Ground.

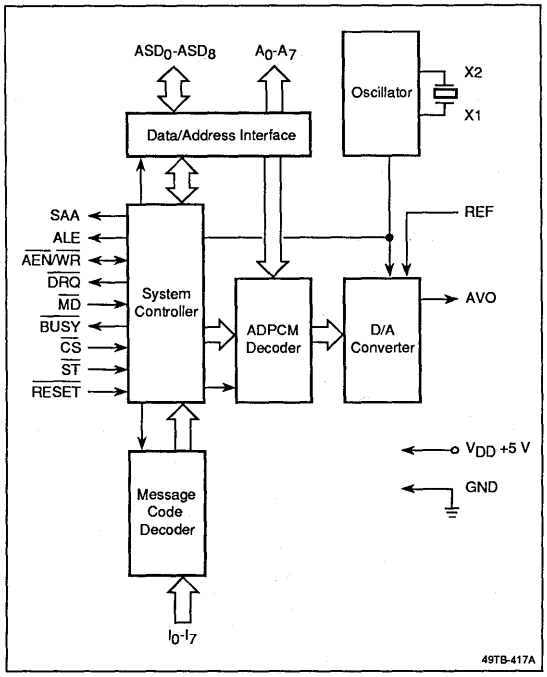
V_{DD} (Power)

+5 V power supply.

NC (No Connection)

These pins are not connected.

Block Diagram



Operation Description

The clock pins should be connected to a ceramic oscillator at 640 MHz.

The $\overline{\text{RESET}}$ input pin is used to initialize the device. To reset, assert the pin for a minimum of 12 oscillator clock cycles.

The μDP7759 can operate with a wide range of supply voltages: 2.7 to 5.5 V. It also has a standby function; it goes to a standby mode when it has been idle (that is, when $\overline{\text{CS}}$, $\overline{\text{ST}}$, or $\overline{\text{RESET}}$ have not been asserted) for more than 3 seconds. The device will automatically release from standby mode when $\overline{\text{CS}}$ and $\overline{\text{ST}}$ are asserted again, or when $\overline{\text{RESET}}$ is asserted.

The μDP7759 has a very simple message selection interface with 1 Mbit of external ROM and can store a maximum of 256 different messages and up to 50 seconds of speech. The message is selected by using the input pins I_0-I_7. The input selection is latched at the rising edge of $\overline{\text{ST}}$ when $\overline{\text{CS}}$ is asserted. When $\overline{\text{ST}}$ is asserted, $\overline{\text{BUSY}}$ will go low until the selected audio speech output is completed. While $\overline{\text{BUSY}}$ is low, a new $\overline{\text{ST}}$ will not be accepted.

The μDP7759 has an internal D/A converter that is a unipolar, current-output type with 9-bit resolution. The output current of the D/A can be controlled by the voltage applied at the REF pin.

Absolute Maximum Ratings

T_A = 25 °C

Supply voltage, V_{DD}	−0.3 to +7.0 V
Input voltage, V_I	−0.3 to V_{DD} +0.3 V
Output voltage, V_O	−0.3 to V_{DD} +0.3 V
Operating temperature, T_{OPT}	−10 to +70 °C
Storage temperature, T_{STG}	−40 to +125 °C

Exposure to Absolute Maximum Ratings for extended periods may affect device reliability; exceeding the ratings could cause permanent damage.

Capacitance

T_A = 25 °C

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
Input capacitance	C_I			10	pF	fc = 1 MHz
Output capacitance	C_O			20	pF	

DC Characteristics

T_A = −10 to +70 °C; V_{DD} = 2.7 to 5.5 V; f_{OSC} = 640 MHz

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
Input voltage high	V_{IH1}	0.7 V_{DD}		V_{DD}	V	Common to I_0-I_7, $\overline{ST}$ $\overline{CS}$, $\overline{RESET}$, $\overline{MD}$, $\overline{WR}$
	V_{IH2}	2.2		V_{DD}	V	Common to ASD_0-ASD_7, V_{DD} = 5 V ±10%
Input voltage low	V_{IL}	0		0.3 V_{DD}	V	Common to I_0-I_7, $\overline{ST}$, $\overline{CS}$, $\overline{RESET}$, $\overline{MD}$, $\overline{WR}$
	V_{IL2}	0		0.8	V	Common to ASD_0-ASD_7, V_{DD} = 5 V ±10%
Output voltage high	V_{OH}	V_{DD} −0.5		V_{DD}	V	I_{OH} = −100 μA
Output voltage low	V_{OL}	0		0.4	V	I_{OL} = 1.6 mA, V_{DD} = 5 V ±10%
Input leakage current	I_{LI}			3	μA	Common to I_0-I_7, $\overline{ST}$, $\overline{WR}$, $\overline{CS}$. $\overline{MD}$, ASD_0-ASD_7
Output leakage current	I_{LO}			3	μA	$\overline{BUSY}$. A_0-A_8
Supply current	I_{DD1}			10	mA	V_{DD} = 5 V
	I_{DD2}			20	μA	V_{DD} = 5 V in standby mode
	I_{DD3}			1	mA	2.7 V ≤ V_{DD} ≤ 3.5 V
	I_{DD4}			10	μA	2.7 V ≤ V_{DD} ≤ 3.5 V in standby mode
Reference input high current area (1)	I_{REF1}	140	250	440	μA	V_{DD} = 2.7 V, R_{REF} = 0 Ω
	I_{REF2}	500	760	1200	μA	V_{DD} = 5.5 V, R_{REF} = 0 Ω
Reference input low current area (1)	I_{REF3}	21	30	39	μA	V_{DD} = 2.7 V, R_{REF} = 50 kΩ
	I_{REF4}	68	78	88	μA	V_{DD} = 5.5 V, R_{REF} = 50 kΩ
D/A converter output current	I_{AVO}	32	34	36	I_{REF}	2.7 V ≤ V_{DD} ≤ 5.5 V. V_{AVO} = 2.0 V, D/A input = 1 FFH
D/A converter output leakage current	I_{LA}			5	μA	0 V ≤ V_{AVO} ≤ V_{DD} in standby mode

AC Characteristics

T_A = −10 to +70 °C; V_{DD} = 2.7 to 5.5 V; f_{OSC} = 640 MHz

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
Timing for all Modes						
$\overline{CS}$ set-up time	t_{CS}	0			ns	When $\overline{ST}$ ↓
$\overline{CS}$ hold time	t_{SC}	0			ns	After $\overline{ST}$ ↑
$\overline{ST}$ pulse width	t_{CC1}	350			ns	V_{DD} = 5 V ±10%
	t_{CC2}	5			μs	V_{DD} = = 2.7 to 5.5. V
Message code set-up time	t_{DW1}	350			ns	V_{DD} = 5 V ±10%
	t_{DW2}	5			μs	V_{DD} = 2.7 to 5.5 V
Message code hold time	t_{WD}	0			μs	After $\overline{ST}$ ↑
Switching Characteristics for all Modes						
$\overline{BUSY}$ rise time	t_{R1}			800	ns	C_L = 150 pF, V_{DD} = 5 V ±10%
	t_{R2}			2	μs	C_L = 150 pF, V_{DD} = 2.7 to 5.5 V ±10%
$\overline{BUSY}$ fall time	t_{F1}			800	ns	C_L = 150 pF, V_{DD} = 5 V ±10%
	t_{F2}			2	μs	C_L = 150 pF, V_{DD} = 2.7 to 5.5 V ±10%
Timing for Stand Alone Mode						
$\overline{RESET}$ pulse width	t_{RST}	18.5			μs	
$\overline{CS}$ set-up time	t_{CS}	0			ns	When $\overline{ST}$ ↓
$\overline{CS}$ hold time	t_{SC}	0			ns	After $\overline{ST}$ ↑

AC Characteristics (cont)

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
Timing for Stand Alone Mode (cont)						
$\overline{ST}$ pulse width	t_{CC1}	2			μs	2.7 V ≤ V_{DD} ≤ 5.5 V
	t_{CC2}	350			ns	4.5 V ≤ V_{DD} ≤ 5.5 V
Message code set-up time	t_{DW1}	2			μs	2.7 V ≤ V_{DD} ≤ 5.5 V
	t_{DW2}	350			ns	4.5 V ≤ V_{DD} ≤ 5.5 V
Message code hold time	t_{WD}	0			ns	After $\overline{ST}$ ↑
Speech data set-up time	t_{RD}			8	μs	When $\overline{DRQ}$ ↓
Speech data hold time	t_{RDH}			1.25	μs	After $\overline{DRQ}$ ↑
$\overline{ST}$ set-up time	t_{RS}	12.5			μs	After $\overline{RESET}$ ↑
$\overline{BUSY}$ hold time	t_{RB}			9.5	μs	After $\overline{RESET}$ ↓
Switching Characteristics for Stand Alone Mode						
$\overline{BUSY}$ output delay	t_{SBO}		6.25	10	μs	Operation mode after $\overline{ST}$ ↓
Speech output delay	t_{SSO}		2.1	2.2	ms	Operation mode after $\overline{BUSY}$
$\overline{BUSY}$ hold time	t_{BD}			15	μs	After synthesis
ALE pulse width	t_{LL}		3.13		μs	
Higher address set-up time	t_{AL}		3.13		μs	When ALE ↓
	t_{AE}		0		μs	When $\overline{AEN}$ ↑
	t_{LA}		3.13		μs	After ALE ↓
	t_{EA}		0		μs	After $\overline{AEN}$ ↑
$\overline{AEN}$ pulse width	t_{AEN}		14.1		μs	
$\overline{DRQ}$ output time	t_{LC}		3.13		μs	After ALE ↓
Pulse width timing	t_{AC}		6.25		μs	
$\overline{DRQ}$ pulse duration	t_{DCC}		7.81		μs	
ROM read cycle time	t_{MRO}		37.5		μs	
Timing for Slave Mode						
$\overline{MD}$ input timing	t_{RM}	6.2			μs	After $\overline{RESET}$ ↑
	t_{BM}	0			ns	After $\overline{BUSY}$ ↑
	t_{MD}	6.2			μs	After $\overline{MD}$ ↑
Speech data set-up time	t_{DW}	350			ns	When $\overline{WR}$ ↑; 5 V ±10%
Speech data hold time	t_{WD}	0			ns	When $\overline{WR}$ ↑; 5 V ±10%
Data write time	t_{WR}			31.7	μs	After $\overline{DRQ}$ ↓
$\overline{WR}$ pulse width	t_{CC}	350			ns	5 V ±10%
$\overline{CS}$ set-up time	t_{CW}	0			ns	When $\overline{WR}$ ↓
$\overline{CS}$ hold time	t_{WC}	0			ns	After $\overline{WR}$ ↑
$\overline{MD}$ pulse width	t_{MD2}	6.2			ns	

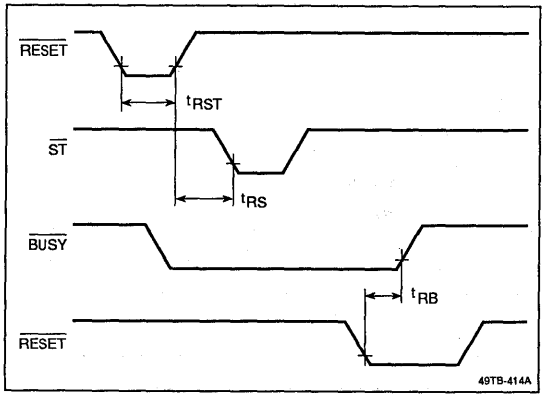
AC Characteristics (cont)

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
Switching Characteristics for Slave Mode						
$\overline{BUSY}$ output delay	t_{SBO}			9.5	μs	After $\overline{MD}$ ↓
$\overline{DRQ}$ output delay	t_{MDR}	50		70	μs	After $\overline{MD}$ ↓
Data request timing	t_{WRQ}			3	μs	After $\overline{WR}$ ↓
Timing for Standby Mode						
Pulse width standby escape signal (Note)	t_{AW}	350			ns	
Switching Characteristics for Standby Mode						
Operation mode hold time	t_{STB}		2.9	3	s	After synthesis
Activate/inactivate D/A converter time	t_{DA}		46.5	47	ms	
$\overline{BUSY}$ ↓	t_{SB}		6.25	10	μs	After L ↓ (Note)
Synthesis start time	t_{SSS}		2.1	2.2	ms	After t_{DA}
$\overline{BUSY}$ output delay	t_{SBS}		4	80	ms	After L ↓ (Note)

Note:
L = Signal to escape standby mode.
= $\overline{CS} \wedge \overline{ST}$ when operation mode is stand alone mode.
= $\overline{CS} \wedge \overline{WR}$ when operation mode is slave mode.

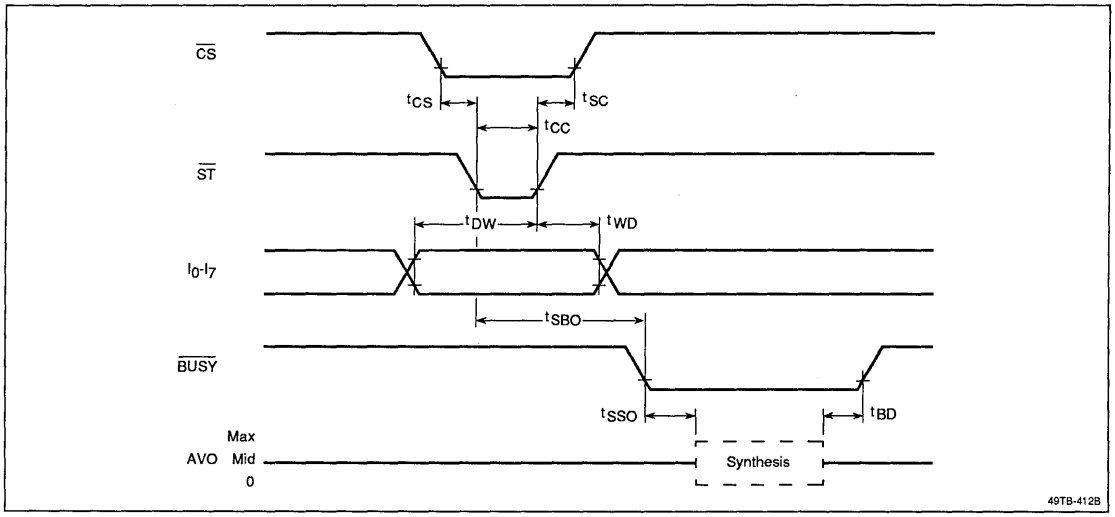
Timing Waveforms

Reset

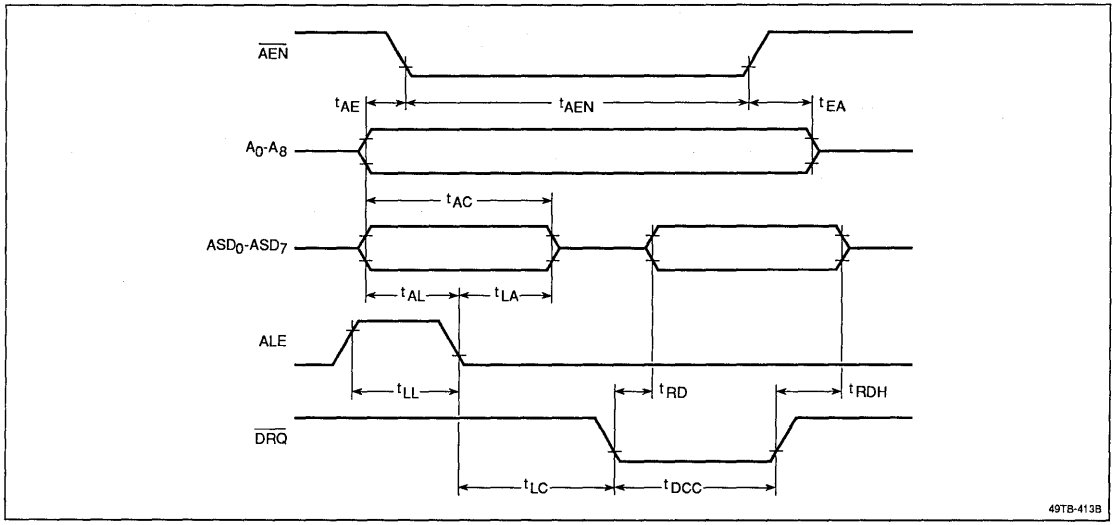


Timing Waveforms (cont)

Control Timing for Stand Alone Mode

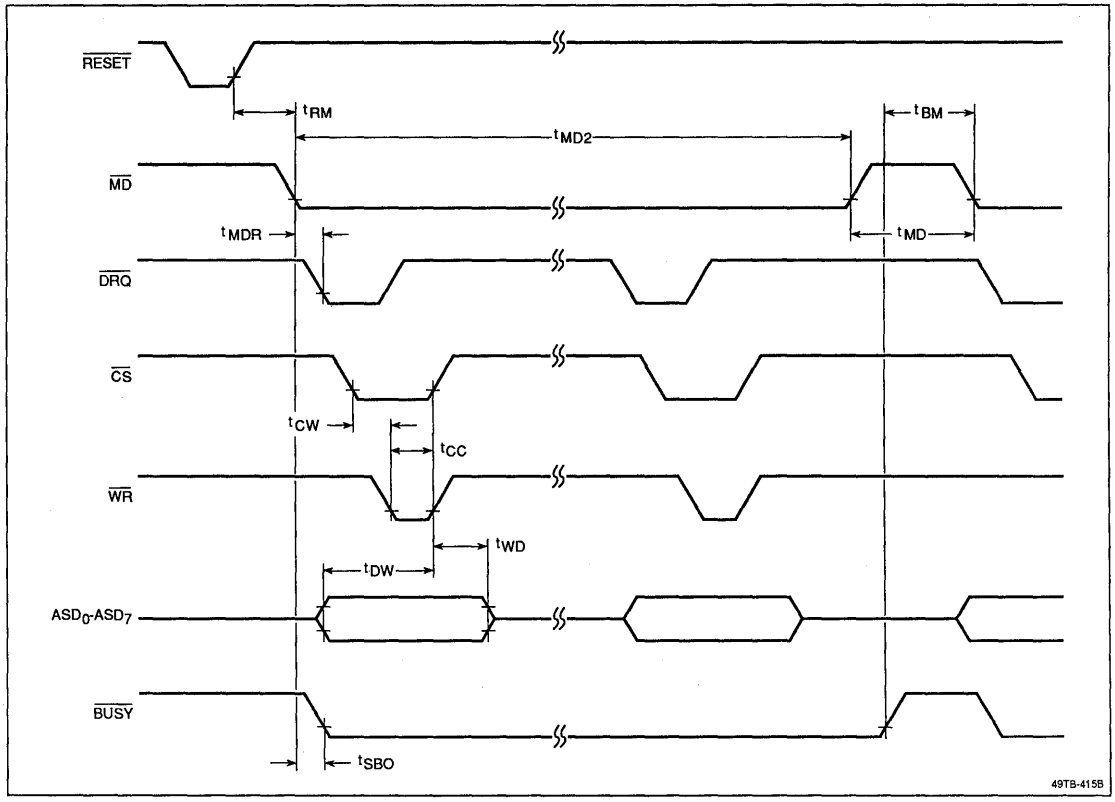


Memory Access for Timing for Stand Alone Mode



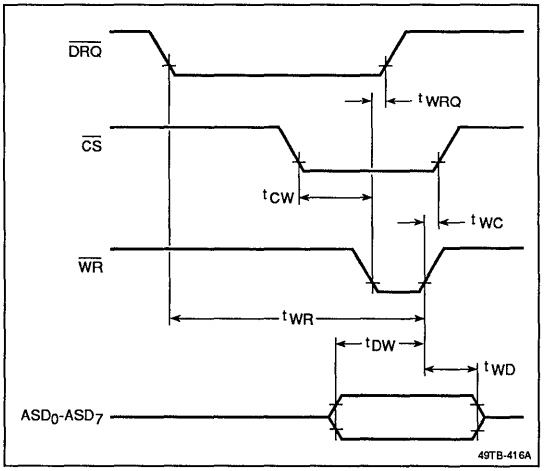
Timing Waveforms (cont)

Control Timing for Slave Mode

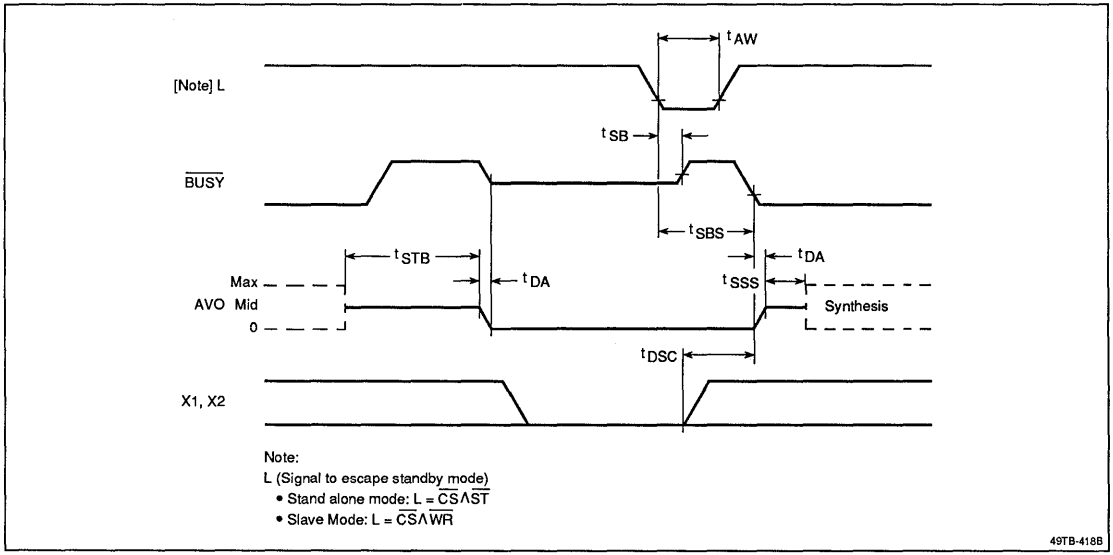


Timing Waveforms (cont)

Data Transfer for Slave Mode



Timing for Standby Mode



DEVELOPMENT TOOLS

Section 4
Development Tools

EVAKIT-7720B
μPD7720 Stand-Alone Emulator

Description

The EVAKIT-7720B is a stand-alone emulator for NEC's μPD7720A, μPD77P20 and μPD77C20A digital signal processing interfaces (SPI). The EVAKIT-7720B provides complete hardware emulation and software debug capabilites for the SPI. Real-time and single-step emulation capability, a powerful on-board system monitor, and a user-specified breakpoint create a powerful debug environment.

The EVAKIT-7720B is controlled over a serial line from a terminal or host computer system. User programs are downloaded into the instruction ROM and data ROM emulation memory through a serial line or read from an EPROM device. An on-board programmer for μPD2732 and μPD2732A EPROMs provides an easy means for submitting your final code for production. You can also use the EVAKIT-7720B to program the μPD77P20 EPROM version of the part for final system test and evaluation.

Features

- Real-time and single-step emulation capability
  - Real-time program execution at 8 MHz.
  - Real-time program execution with address breakpoint and loop counter (up to 256 loops)
  - Real-time program execution for a number of steps
  - Single-step program execution with display of address, instruction, registers and flags
- On-board emulation memory
  - Instruction ROM, data ROM and internal RAM
- Powerful system monitor
  - Display/change/initialize instruction and data ROM
  - Display/change/initialize internal RAM
  - Display/modify internal registers
  - Read/write/display/verify/blank check EPROM device
  - Upload/download/verify instruction and data ROM
  - Perform self-diagnostics
  - Reset emulation chip
- Supports two operating modes
  - External terminal controlled
  - Host computer system controlled
- Emulator controller for IBM PC®, PC/XT®, PC AT® or compatibles
- Serial interface: RS-232C, TTL, or 20 mA current loop
- EPROM programming capability (μPD2732, μPD2732A, μPD77P20)
- Requires an external power supply

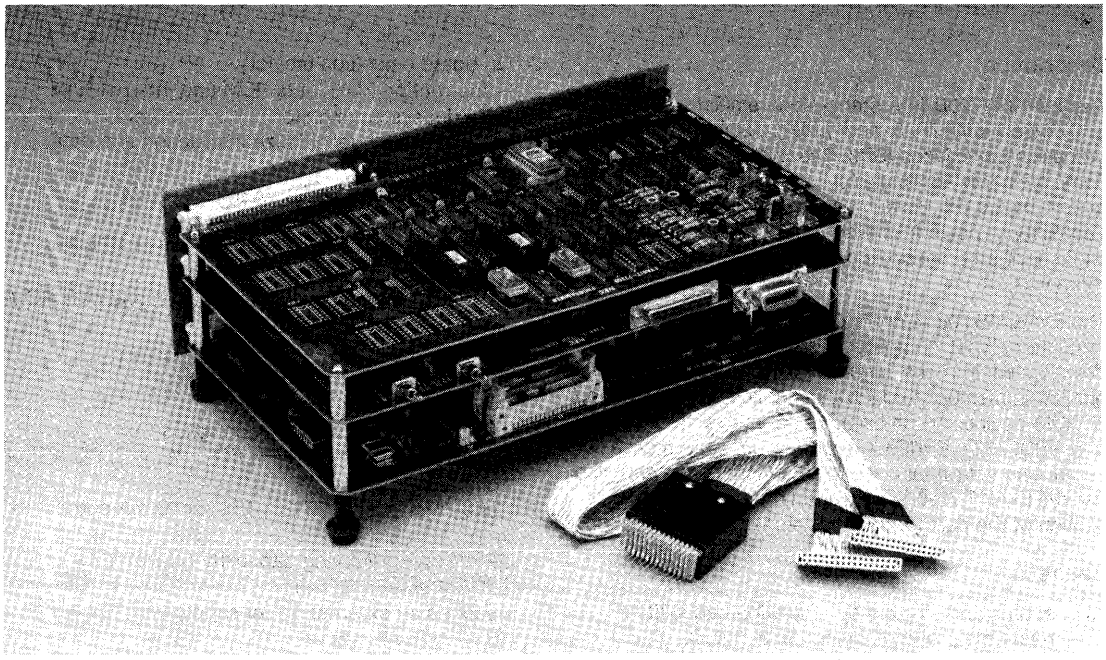
Ordering Information

Part Number	Description
EVAKIT-7720B	Stand-alone emulator for μPD7720A/P20/C20A

4

IBM PC, PC/XT, and PC AT are registered trademarks of International Business Machines Corporation.

μPD7720 Stand-Alone Emulator



Description

The ASM77 Absolute Assembler converts symbolic source code for the NEC μPD7720A/77P20/77C20A Digital Signal Processing Interfaces (SPI) into executable absolute address object code. Two separate assemblers are provided: one assembles the source program for the Instruction ROM; the other assembles the source program for the Data ROM. An object code file is produced in ASCII hexadecimal format and may be downloaded to an EPROM programmer or the NEC stand-alone emulator, the EVAKIT-7720B.

The NEC ASM77 assembler is available for operation on an MS-DOS® computer system with at least one disk drive and 128KB of installed system memory.

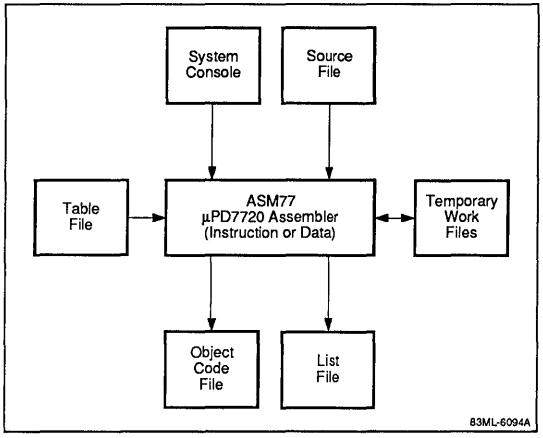
Features

- Absolute address object code output
- Free format statements
- Separate assemblers for instruction and data ROMs
- User-selectable and directable output files
- Runs under the MS-DOS operating system

Ordering Information

Part Number	Description
ASM77-D52	MS-DOS, 5.25" Double Density Disk

ASM77 Block Diagram



3

MS-DOS is a registered trademark of Microsoft Corporation

Description

The SIM77 Simulator is a software tool for analyzing program code and I/O timing for the NEC μPD7720A/77P20/77C20A Digital Signal Processing Interfaces (SPI). SIM77 simulates the operation of the SPI using your instruction and data ROM codes with specially prepared serial input, parallel input, and simulation timing files. The system console of the host computer controls simulation. SIM77 can create serial and parallel output files, display the latest trace steps, and send all console input/output to a disk file or system printer for later analysis.

SIM77 is available for operation on an MS-DOS® computer system with at least one disk drive and 128KB of installed system memory.

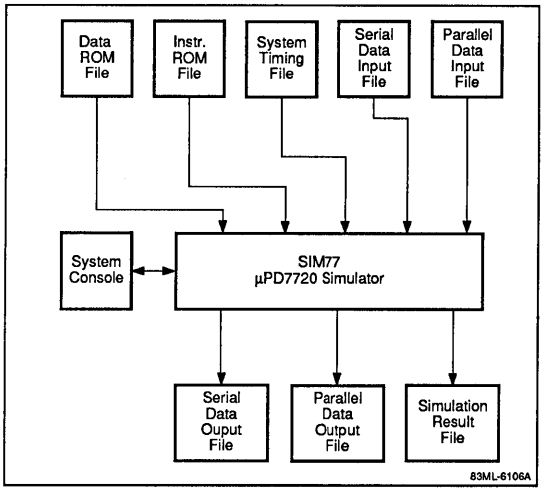
Features

- Continuous/single-step execution
- Display/modify instruction ROM, data ROM, RAM, stack, registers or flags
- User-controllable parallel data transfer direction
- User generated interrupt capability
- Sophisticated breakpoint capabilities
  — Up to eight breakpoints with loop counter
- Trace capability with start/stop conditions
- Instruction ROM disassembler
- Store all console inputs and outputs on disk
- Runs under the MS-DOS operating system

Ordering Information

Part Number	Description
SIM77-D52	MS-DOS, 5.25" Double Density Disk

SIM77 Block Diagram



4

MS-DOS is a registered trademark of Microsoft Corporation.

Description

The EVAKIT-77C25 is a stand-alone emulator for NEC's μPD77C25 and μPD77P25 digital signal processing interfaces (SPI+). The EVAKIT-77C25 provides complete hardware emulation and software debug capabilites for the SPI+. Real-time and single-step emulation capability, coupled with sophisticated breakpoint capability, real-time tracer and a powerful on-board system monitor, create a powerful debug environment. A line assembler and symbolic disassembler, full register and memory control and complete upload/download capabilities simplify the task of debugging your hardware and software.

An on-board EEPROM is available for storage of the current debug environment during EVAKIT power down. Using the freeze (FRZ) command, the current contents of the instruction and data ROM, the internal RAM, the SPI+ registers, the break registers and registered command strings are saved to the EEPROM. The Melt (MLT) command restores this information.

The EVAKIT-77C25 is controlled via serial line from a local terminal or host computer system. User programs can be uploaded from or downloaded to the Instruction and data ROM emulation memory through a serial line from a local host computer, a remote host computer system or an external EPROM programmer. NEC provides an emulator controller program for use on an IBM PC®, PC/XT®, PC AT® or compatible local host computer. To transfer data to/from a remote host computer system, the EVAKIT-77C25 can be placed into terminal emulation mode and be used as a terminal for the remote system. Data can also be read from or written to an external EPROM programmer under the control of the on-board monitor.

Features

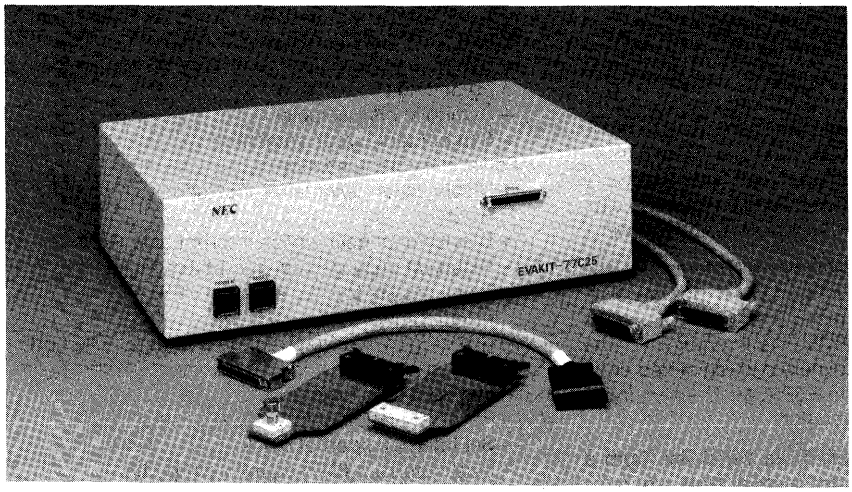
- Real-time and single-step emulation capability
  - Real-time program execution with/without breakpoint
  - Single-step program execution with trace display
- Subcommands available during real-time emulation:
  - Generate an interrupt to the emulation chip
  - Read/display status register
  - Read/write the data register
  - Reset the emulation chip
- On-board emulation memory
  - Instruction ROM: 2K x 24 bits
  - Data ROM: 1K x 16 bits
  - Data RAM: 256 x 16 bits
- Symbolic debug capability
  - Symbols may be used to specify addresses for commands
  - Symbolic disassembler
  - Symbol table clear command
- Powerful system monitor
  - Display/change/initialize instruction and data ROM
  - Display/change/initialize internal data RAM
  - Display/modify general and status registers
  - Transfer data to/from external EPROM programmer
  - Upload/download instruction and data ROM code
  - Line assembler
  - Display break registers
  - Reset emulation chip
  - Set internal/external clock
  - Mask interrupt (INT) signal from probe
- Sophisticated breakpoint capability
  - Break on address and pass count (up to 65535 passes)
  - Break on being in or out of address range
  - Breakpoints specified on command line or preset in the break address, address range and mode registers
  - Up to 37 break addresses or address ranges can be set
- Real-time program trace feature
  - Store 4092 clocks worth of information
  - Traces program counter, data bus, $\overline{RD}$, $\overline{WR}$, $\overline{CS}$, A0, DRQ, $\overline{DACK}$, RST, INT, P0, P1, SCK, SI, $\overline{SIEN}$, SO, $\overline{SOEN}$, and SORQ
  - Displays trace with/without mnemonics
  - Trace buffer pointer and search capability
- EEPROM for temporary storage of instruction and data ROM, internal RAM and registers, break registers, command strings
- On-line help facility
- Three RS-232C serial ports
  - CH1: Terminal or local host system
  - CH2: Remote host system
  - CH3: EPROM programmer
- Emulator controller for IBM PC, PC/XT, PC AT or compatibles

IBM PC, PC/XT, and PC AT are registered trademarks of International Business Machines Corporation.

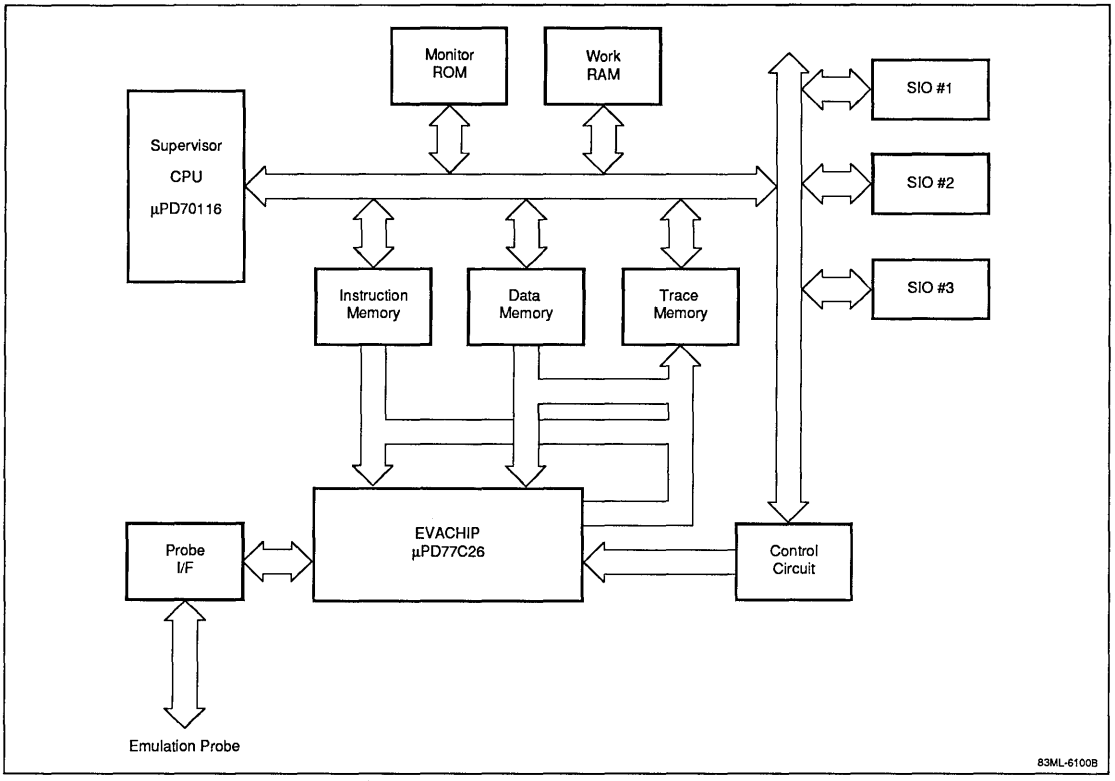
Ordering Information

Part Number	Description
EVAKIT-77C25	Stand-alone emulator for μPD77C25/P25

μPD77C25 Stand-Alone Emulator



EVAKIT-77C25 Block Diagram



RA77C25
μPD77C25 Relocatable Assembler Package

Description

The RA77C25 Relocatable Assembler package converts symbolic source code for the μPD77C25 and μPD77P25 Digital Signal Processing Interfaces (SPI+) into executable absolute address object code. The Relocatable Assembler package consists of four separate programs: an assembler (RA77C25), a linker (LK77C25), a hexadecimal format object code converter (OC77C25), and a librarian (LB77C25).

RA77C25 translates a symbolic source module file with include files into a relocatable object module. The assembler produces a relocatable object module file and a listing file that can contain the assembly list, symbol list and cross-reference list. If absolute addresses have been specified in the source module file and no relocatable segments or external variables or labels are referenced, the assembler can output an ASCII hexadecimal format object file and a symbol table file directly.

LK77C25 combines relocatable object modules, library modules when necessary, and other linker load modules and converts them into an absolute load module. The linker produces a link map and an absolute load module. OC77C25 converts an absolute object module from RA77C25 or an absolute load module from LK77C25 into an ASCII hexadecimal format object file and a symbol table file.

LB77C25 allows commonly used relocatable object modules to be stored in one file and linked into multiple programs, greatly increasing programming efficiency. When a library file is included in the input of the linker, the linker extracts only those modules required to resolve external references from the file and relocates and links them.

Features

- Absolute address object code output
- User-selectable and directable output files
- Extensive error reporting
- Macro capability
- Conditional assembly directives
- Powerful librarian
- Runs under the following operating systems:
  - MS-DOS®
  - VAX/VMS®
  - VAX/UNIX™ 4.2BSD or Ultrix™

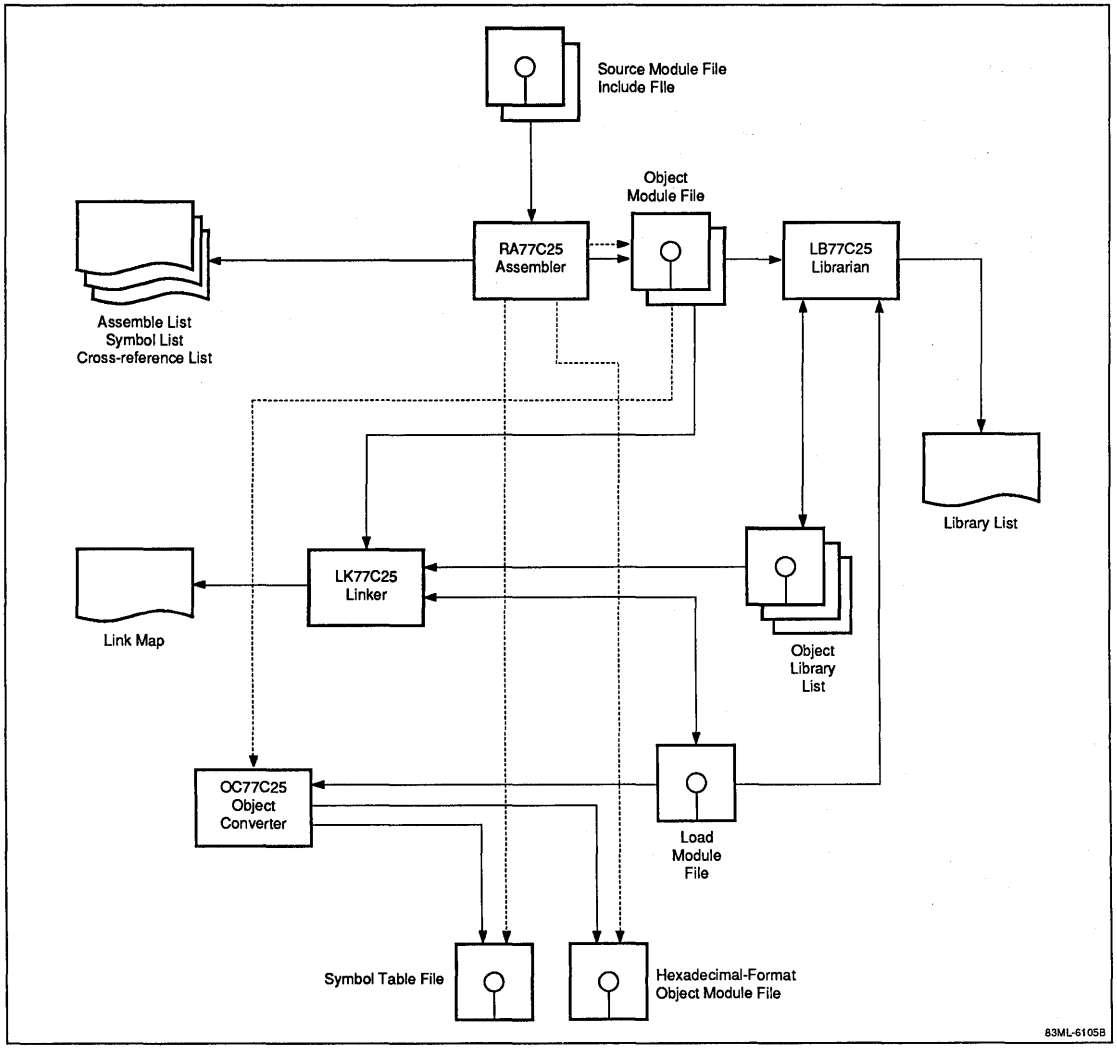
Ordering Information

Part Number	Description
RA77C25-D52	MS-DOS, 5.25" double density diskette
RA77C25-VVT1	VAX/VMS, 9-track 1600 BPI magnetic tape
RA77C25-VXT1	VAX/UNIX 4.2BSD or Ultrix, 9-track 1600BPI magnetic tape

4

MS-DOS is a registered trademark of Microsoft Corporation.
VAX and VMS are registered trademarks of Digital Equipment Corporation.
Ultrix is a trademark of Digital Equipment Corporation.
UNIX is a trademark of AT&T.

RA77C25 Block Diagram



PRELIMINARY INFORMATION

Description

The EVAKIT-77220 is a stand-alone emulator for NEC's μPD77220 and μPD77P220 24-Bit Fixed Point Advanced Signal Processor. The EVAKIT-77220 provides complete hardware emulation and software debug capabilites for the μPD77220/P220. Real-time and single-step emulation capability, coupled with sophisticated breakpoint capability, real-time tracer and a powerful on-board system monitor, create a powerful debug environment. A symbolic line assembler and disassembler, full register and memory control and complete upload/download capabilities simplify the task of debugging your hardware and software.

The EVAKIT-77220 is controlled via serial line from a local terminal or host computer system. User programs can be uploaded from or downloaded to the instruction and data ROM emulation memory through a serial line from either a local host computer, a remote host computer system or an external EPROM programmer. NEC provides an emulator controller program for use on an IBM PC®, PC/XT®, PC AT® or compatible local host computer. To transfer data to/from a remote host computer system, the EVAKIT-77220 can be placed into terminal emulation mode and be used as a terminal for the remote system. Data can also be read from or written to an external EPROM programmer under the control of the monitor.

Features

- On-board emulation memory for:
  - Instruction ROM, data ROM, internal data RAM
  - External emulation RAM: fast/slow speed
- Selectable clock: internal or external
- Real-time and single-step emulation capability
  - Real-time program execution with/without breakpoint
  - Single-step program execution with trace display
- Console I/O available during real-time emulation to:
  - Generate an INT and NMI signals to emulation chip
  - Generate HWR, P0 and P1 signals to emulation chip
  - Display HRD, P2, P3 and RQM signals from emulation chip
- Memory manipulation commands
  - Change/display/fill/move/search data in:
    Internal instruction/data ROM
    Internal data RAM
    External emulation RAM
- Register manipulation commands
  - Change/display general and status registers
  - Read/write DRS, read SI, and write SO registers
- Powerful system utilities
  - Transfer data to/from external EPROM programmer
  - Upload/download instruction/data ROM code and symbols
  - Transfer external memory contents between EVAKIT/prototype
  - Reset emulation chip
  - Specify internal/external INT, NMI, and Reset signals
  - External memory mapping: internal/user, fast/slow
- Symbolic debug capability
  - Symbols may be used to specify addresses in commands
  - Symbolic line assembler and disassembler
  - Symbol add/change/display/delete commands
- Sophisticated breakpoints for master and slave modes
  - Instruction memory address or specified instruction
  - Internal data RAM address or specified data value
  - External memory address or specified data value
  - Loop Counter Borrow
  - External break signal from probe
  - Up to 65536 passes
  - Read/write data from host system (slave mode only)
  - Breakpoints specified on command line or preset in ten logical break registers
- Real-time program trace feature
  - Store 2048 clocks worth of information
  - Trace starts with emulation or on an address
  - Traces program counter, ROM counter, loop counter borrow, internal bus, SIAK, SOAK, most external pins
  - Displays trace with/without mnemonics
  - Trace buffer pointer and search capability

4

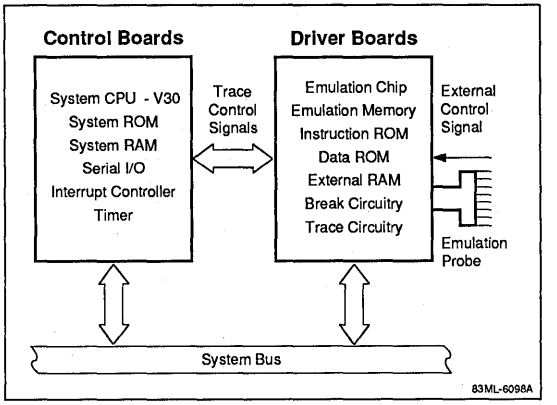
IBM PC, PC/XT, and PC AT are registered trademarks of International Business Machines Corporation.

- On-line help facility
- Automatic command execution from Macro command table
- Three RS-232C serial ports
  - CH1: Terminal or local host system
  - CH2: Remote host system
  - CH3: EPROM programmer
- Emulator controller for IBM PC, PC/XT, PC AT or compatibles

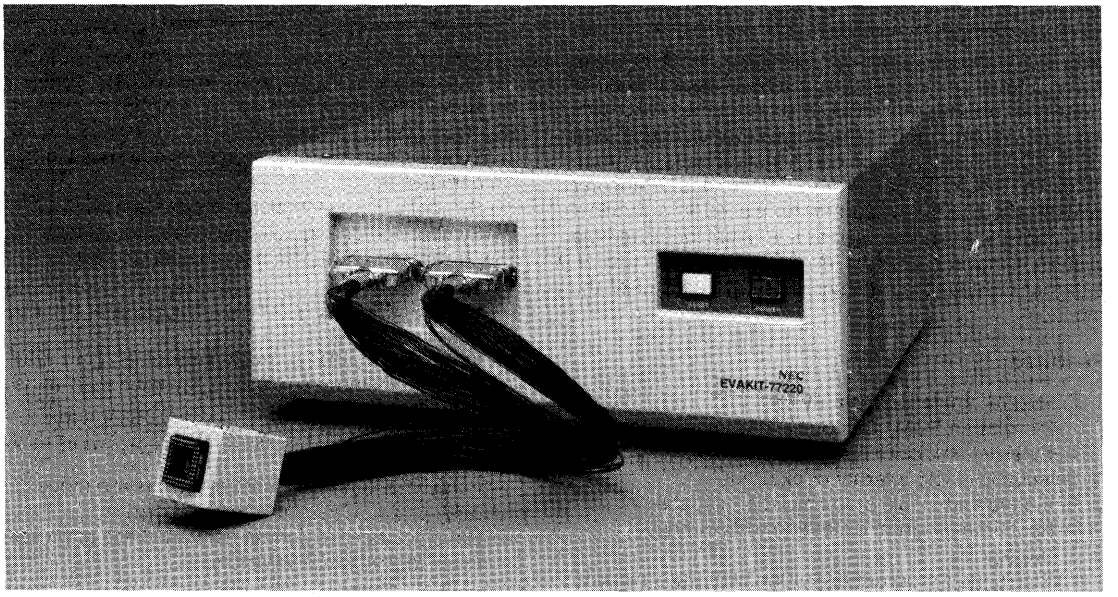
Ordering Information

Part Number	Description
EVAKIT-77220	Stand-alone emulator for μPD77220/P220

EVAKIT-77220 Block Diagram



μPD77220 Stand-Alone Emulator



Description

The EVAKIT-77230 is a stand-alone emulator for NEC's μPD77230 and μPD77P230 32-bit floating point advanced signal processor (ASP). The EVAKIT-77230 provides complete hardware emulation and software debug capabilites for the ASP. Real-time and single-step emulation capability, coupled with sophisticated breakpoint capability, real-time tracer and a powerful on-board system monitor, create a powerful debug environment. A symbolic line assembler and disassembler, full register and memory control and complete upload/download capabilities simplify the task of debugging hardware and software.

The EVAKIT-77230 is controlled via serial line from a local terminal or host computer system. User programs can be uploaded from or downloaded to the Instruction and data ROM emulation memory through a serial line from a local host computer, a remote host computer system, or an external EPROM programmer. NEC provides an emulator controller program for use on an IBM PC®, PC/XT®, PC AT® or compatible local host computer. To transfer data to/from a remote host computer system, the EVAKIT-77230 can be placed into terminal emulation mode and be used as a terminal for the remote system. Data can also be read from or written to an external EPROM programmer under the control of the monitor.

Features

- On-board emulation memory for:
  - Instruction ROM, data ROM, internal data RAM
  - External emulation RAM: fast/slow speed
- Selectable clock: 13.37/6.68/3.34 MHz internal or external
- Real-time and single-step emulation capability
  - Real-time program execution with/without breakpoint
  - Single-step program execution with trace display
- Console I/O available during real-time emulation to:
  - Generate an INT and NMI signals to emulation chip
  - Generate HWR, P0 and P1 signals to emulation chip
  - Display HRD, P2, P3 and RQM signals from emulation chip
- Memory manipulation commands
  - Change/display/fill/move/search data in:
    Internal instruction/data ROM
    Internal data RAM
    External emulation RAM
- Register manipulation commands
  - Change/display general and status registers
  - Read/write DRS, read SI, and write SO registers
- Powerful system utilities
  - Transfer data to/from external EPROM programmer
  - Upload/download instruction/data ROM code and symbols
  - Transfer external memory contents between EVAKIT/prototype
  - Reset emulation chip
  - Specify internal/external INT, NMI, and reset signals
  - External memory mapping: internal/user, fast/slow
- Symbolic debug capability
  - Symbols may be used to specify addresses in commands
  - Symbolic line assembler and disassembler
  - Symbol add/change/display/delete commands
- Sophisticated breakpoints for master and slave modes
  - Instruction memory address or specified instruction
  - Internal data RAM address or specified data value
  - External memory address or specified data value
  - Loop counter borrow
  - External break signal from probe
  - Up to 65536 passes
  - Read/write data from host system (slave mode only)
  - Breakpoints specified on command line or preset in ten logical break registers
- Real-time program trace feature
  - Store 2048 clocks worth of information
  - Trace starts with emulation or on an address
  - Traces program counter, ROM counter, loop counter borrow, internal bus, SIAK, SOAK, most external pins
  - Displays trace with/without mnemonics
  - Trace buffer pointer and search capability

4

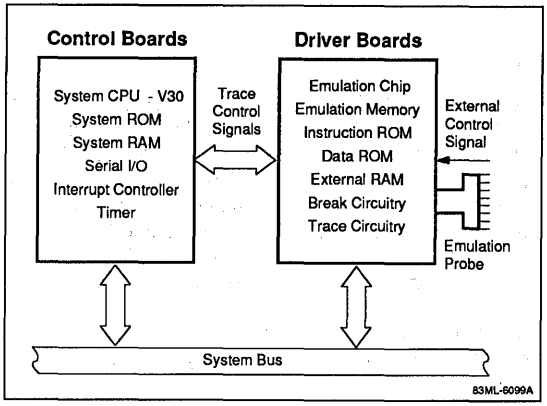
IBM PC, PC/XT, and PC AT are registered trademarks of International Business Machines Corporation.

- On-line help facility
- Automatic command execution from Macro command table
- Three RS-232C serial ports
  - CH1: Terminal or local host system
  - CH2: Remote host system
  - CH3: EPROM programmer
- Emulator controller for IBM PC, PC/XT, PC AT™ or compatibles

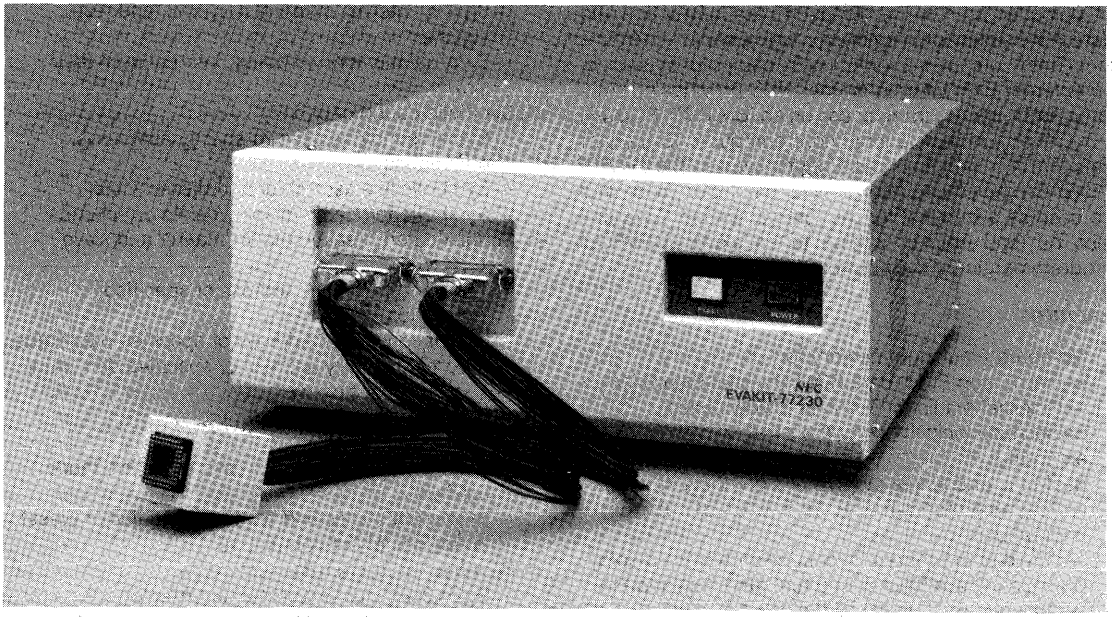
Ordering Information

Part Number	Description
EVAKIT-77230	Stand-alone emulator for μPD77230/P230

EVAKIT-77230 Block Diagram



μPD77230 Stand-Alone Emulator



DDK-77230
μPD77230 Evaluation Board

Description

The DDK-77230 Evaluation Board for the NEC μPD77230 Advanced Signal Processor (ASP) provides a low-cost hardware evaluation and development tool for high-speed digital signal processing applications. The DDK-77230 board features a preprogrammed ASP which contains built-in ROM routines for: FFTs, FIR and IIR filters; floating point math functions such as SIN, COS, LOG and EXP; Serial I/O and others. This board provides an easy-to-use hardware implementation of an ASP which will allow you to become adept at writing ASP programs.

The DDK-77230 board is a peripheral processor that occupies a single slot in an IBM PC®, PC/XT®, PC AT® or compatible. The DDK board package includes a hardware user's manual, host software drivers, ASP assembler software (RA77230), ASP programming examples and additional literature. This total package provides you with a fast, efficient means for evaluating the ASP in an application.

Features

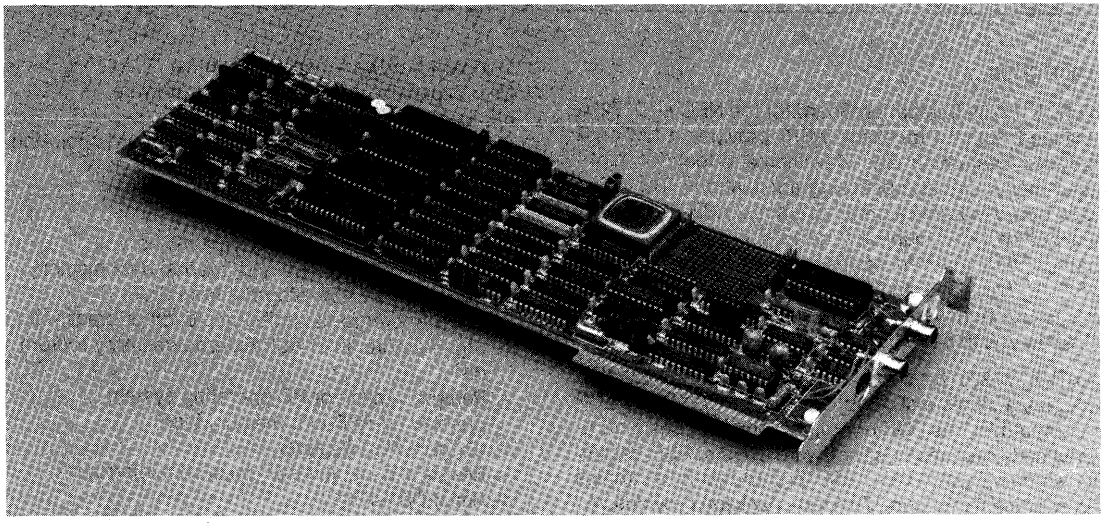
- 4K X 32 bits of high-speed external instruction memory
  - — User expandable to 16K X 32 bits
- 4K X 32 bits of low speed (450 ns) external data memory
  - — User expandable to 32K X 32 bits
- Programmable address breakpoint
- Combo/Codec and programmable timing logic for input/output of 8-bit analog data
- Onboard hardware benchmark timer
- User expansion area for specific hardware applications
- Control register to enhance/simplify board operation
- ASP/Host communication registers
- Interrupted controlled ASP status register
- Menu-driven Host program
  - — Clear/display/fill ASP external memory
  - — Display ASP registers and internal data memory
  - — Download an ASP program from disk
  - — Execute programs until completion or breakpoint
  - — Store/restore all of ASP external memory to/from disk
  - — Return to menu or PC operating system
  - — Enable last ASP ext. memory loc. (Disable Flag Register)
  - — Enable and set the ASP maskable interrupt
  - — Reset DDK-77230
  - — Write a byte from the PC to ASP register

Ordering Information

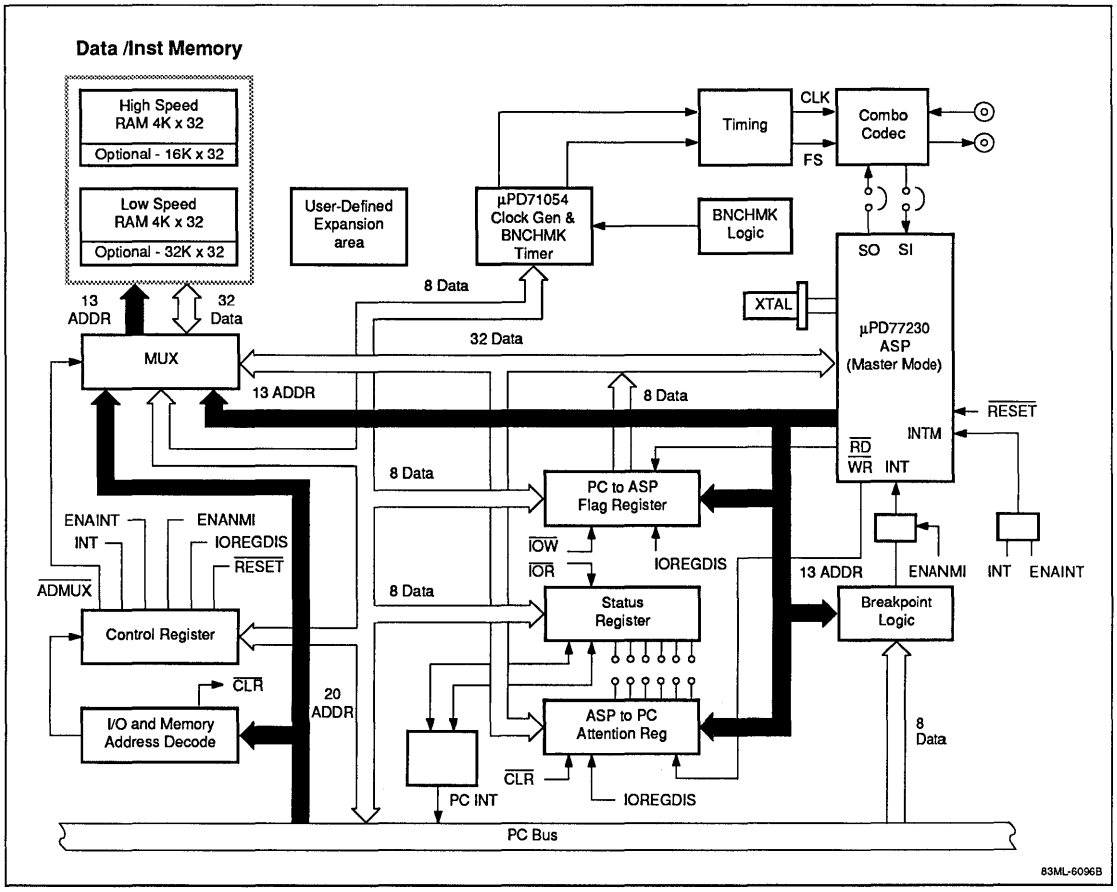
Part Number	Description
DDK-77230	Development/Evaluation Board for μPD77230 (IBM Based)

IBM PC, PC/XT, and PC AT are registered trademarks of International Business Machines Corporation.

μPD77230 Evaluation Board



DDK-77230 Block Diagram



4

RA77230
μPD77220/μPD77230 Relocatable Assembler Package

Description

The RA77230 Relocatable Assembler package converts symbolic source code for μPD77220, μPD77P220, μPD77230, and μPD77P230 Advanced Signal Processors into executable absolute address object code. The Relocatable Assembler package consists of four separate programs: an assembler (RA77230), a linker (LK77230), a hexadecimal format object code converter (OC77230), and a librarian (LB77230).

RA77230 source code modules can be written in either preassembly language or assembly language. Preassembly language allows programs to be written more simply. You do not need to consider the fields of an instruction or their combination, or pay attention to the execution timing of the μPD77220/230. The assembler optimizes the code for you. However, by using assembly language and paying close attention to the instruction fields and their combination, and the execution timing of the chips, much more efficient programs can be written. Since RA77230 can generate an assembly language source file from a preassembly language source file, you can manually optimize this code and write both simple and efficient programs.

RA77230 translates a symbolic source module file containing preassembly or assembly language source code with include files into a relocatable object module. The assembler produces a relocatable object module file, a preassembly language list, and a listing file that can contain the assembly list, symbol list, and cross-reference list.

LK77230 combines relocatable object modules, library modules, and other linker load modules and converts them into an absolute load module. The linker produces a link map and an absolute load module. OC77230 converts an absolute object module from RA77230 or an absolute load module from LK77230 into an ASCII hexadecimal format object file.

LB77230 allows commonly used relocatable object modules to be stored in one file and linked to multiple programs, greatly increasing programming efficiency. When a library file is included as input to the linker, the linker only extracts those modules required to resolve external references from the file and relocates and links them.

Features

- Assembles preassembly and assembly language source code
- Produces absolute address object code
- Supports master/slave modes
- User-selectable and directable output files
- Extensive error reporting
- Macro capability
- Conditional assembly directives
- Powerful librarian
- Runs under the following operating systems:
  - MS-DOS®
  - VAX/VMS®
  - VAX/UNIX™ 4.2BSD or Ultrix™

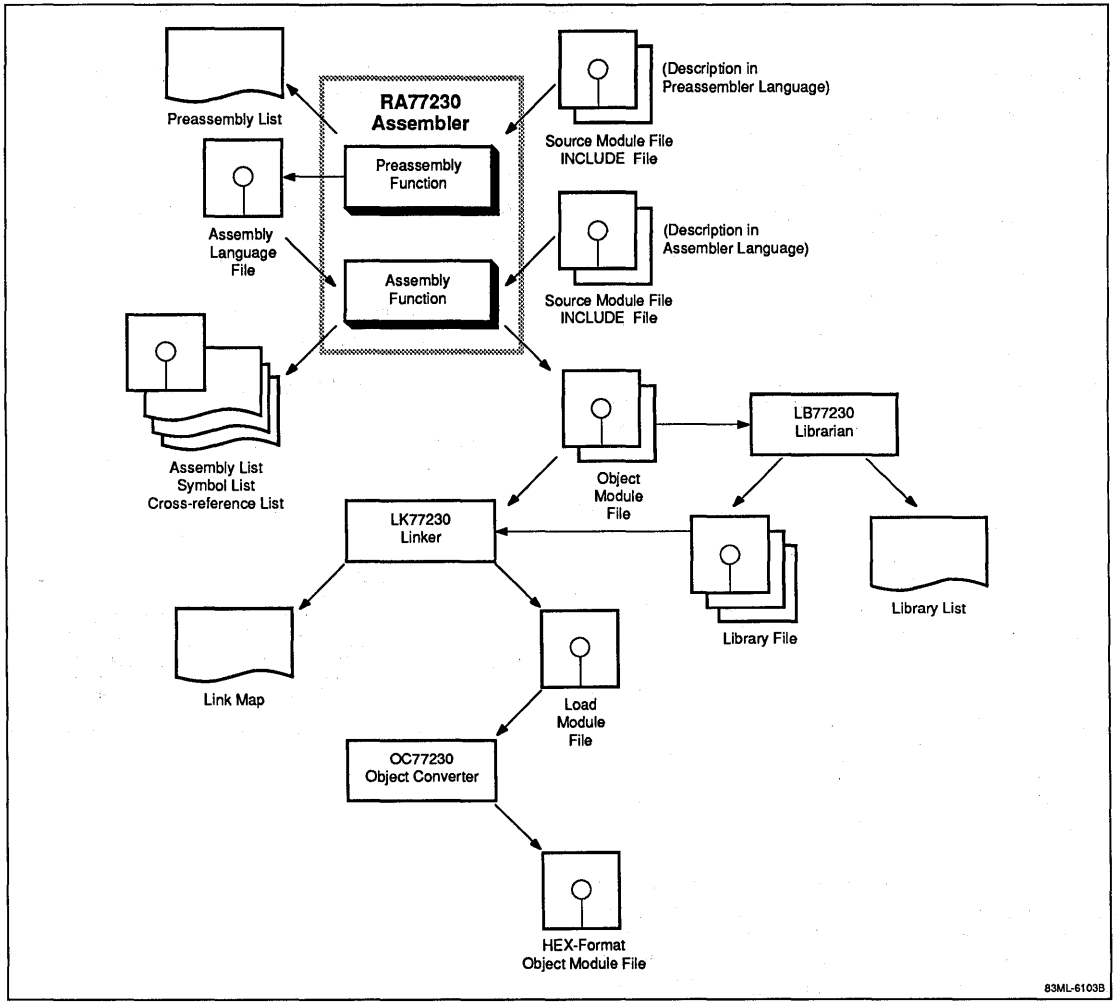
Ordering Information

Part Number	Description
RA77230-D52	MS-DOS, 5.25" double density diskette
RA77230-VVT1	VAX/VMS, 9-track 1600 BPI magnetic tape
RA77230-VXT1	VAX/UNIX 4.2BSD or Ultrix, 9-track 1600 BPI magnetic tape

4

MS-DOS is a registered trademark of Microsoft Corporation.
VAX and VMS are registered trademarks of Digital Equipment Corporation.
Ultrix is a trademark of Digital Equipment Corporation.
UNIX is a trademark of AT&T.

RA77230 Block Diagram



Description

The SM77230 Simulator is a software tool for analyzing program code and I/O timing for μPD77220, μPD77P220, μPD77230, and μPD77P230 Advanced Signal Processors. SM77230 simulates the operation of the μPD77220 or μPD77230 using your instruction and data ROM codes with specially prepared serial input/output, parallel input/output data, and timing files. Sophisticated breakpoint capability, coupled with program trace, and powerful system commands create an easy-to-use simulation environment. A disassembler, full register and memory control, on-line help facility, simulation log, and command files simplify the task of simulating your program.

SM77230 is available for operation on a VAX® computer system with a VT100-compatible terminal under the VMS®, UNIX™ 4.2BSD, or Ultrix™ operating systems.

Features

- Program simulation capability
  - From start to stop address or for a number of steps with or without breakpoints
  - Single-step with register display
- Supports symbolic debugging
- Memory manipulation commands
  - Change/display/fill/move/search data in:
    Internal instruction/data ROM
    Internal data RAM
    External emulation RAM
- Register manipulation commands
  - Change/display general and status registers
- Powerful system commands
  - Read/write instruction/data ROM code and symbols
  - Symbolic disassembler
  - Add/change/display/delete symbols
  - Reset simulation environment
  - External memory mapping in 1K blocks
  - Set value to I/O port
  - Set internal timing clock and step counter
  - Set master/slave simulation mode
  - Set timing of NMI/INT interrupts
- Sophisticated breakpoints (up to 10):
  - Read, write, or read/write of memory address/data
  - Register value
  - Ports 1 to 4 access (up to 10 passes)
  - Loop counter borrow
  - Read/write data from host system (up to tenth occurrence)
  - Execution time or clock count
  - Any instruction (up to tenth occurrence)
  - Any logical combination of these breakpoints
- Program trace displayed on console
  - Trace starts with simulation or on a breakpoint
  - Trace modes: all/registers/addresses/ports
  - Displays complete trace or just changes
- On-line help facility, calculator, and session log feature
- Automatic command execution from batch file or log file
  - Conditional execution capability
- Command entry: screen-oriented or command line
- Runs under the following operating systems:
  - VAX/VMS
  - VAX/UNIX 4.2BSD or Ultrix

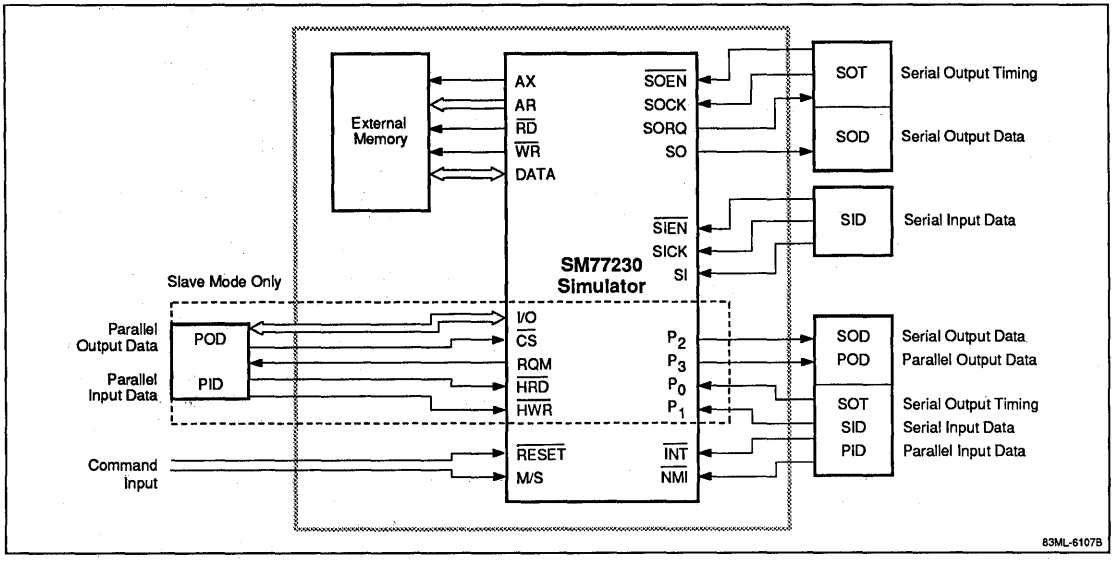
Ordering Information

Part Number	Description
SM77230-VVT1	VAX/VMS, 9-track 1600 BPI magnetic tape
SM77230-VXT1	VAX/UNIX 4.2BSD or Ultrix, 9-track 1600 BPI magnetic tape

4

VAX and VMS are registered trademarks of Digital Equipment Corporation.
Ultrix is a trademark of Digital Equipment Corporation.
UNIX is a trademark of AT&T.

SM77230 Block Diagram



PRELIMINARY INFORMATION

Description

The IE-77810 is a stand-alone in-circuit emulator for NEC's μPD77810 Modem Digital Signal Processor (MDSP). The IE-77810 provides complete hardware and software debug capabilities for the μPD77810. The IE-77810 allows you to debug either the General-Purpose Processor (GPP) or the Digital Signal Processor (DSP) software while emulating the other, to debug or emulate both the GPP and DSP together, or to debug or emulate the MDSP. Real-time emulation capability, coupled with sophisticated breakpoint capability, real-time tracer, and a powerful on-board system monitor create a powerful debug environment. A symbolic line assembler and disassembler for both the GPP and DSP, full register and memory control, and complete upload/download capabilities simplify the task of debugging hardware and software.

The IE-77810 is controlled via a serial line from a local terminal or host computer system. User programs can be uploaded from or downloaded to both the GPP or DSP emulation memory through a serial line from a local host computer, a remote host computer system, or an external EPROM programmer. NEC provides an emulator controller program for use on an IBM PC, PC/XT®, PC/AT®, or compatible local host computer. To transfer data to/from a remote host computer system, the IE-77810 can be placed into terminal emulation mode and be used as a terminal for the remote system. Data can also be read from or written to an external EPROM programmer under the control of the on-board monitor.

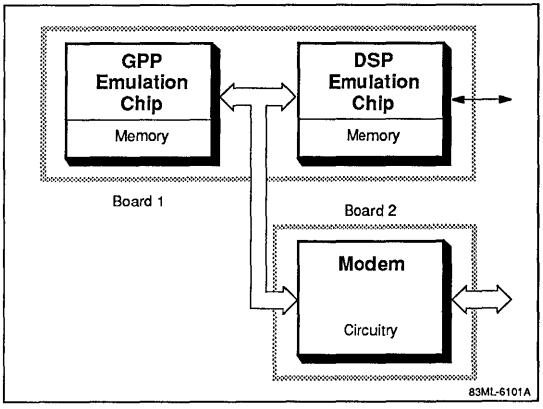
Features

- Real-time emulation for GPP, DSP, and MSDP
- Single-step emulation for GPP and DSP
- IE-77810 operation modes
  - Debug DSP, Emulate GPP
  - Debug GPP, Emulate DSP
  - Debug GPP and DSP
  - Emulate GPP and DSP
  - Debug MDSP
  - Emulate MDSP
- On-board emulation memory for GPP and DSP
- Powerful debug monitors for GPP, DSP, and MDSP
  - Transfer data to/from external EPROM programmer
  - Upload/download object code and symbol table
  - Reset emulation chip
  - For GPP and DSP only:
    Display/change/initialize emulation memory
    Display/modify general and special registers
    Symbolic line assembler and disassembler
- Sophisticated breakpoint capability for GPP, DSP, and MDSP
- Real-time program trace feature for GPP and DSP
- Automatic command execution from macro command table
- On-line help facility
- Three RS-232C serial ports
  - CH1: Terminal or local host system
  - CH2: Remote host system
  - CH3: EPROM programmer
- Emulator controller for IBM PC, PC/XT, PC AT, or compatibles

Ordering Information

Part Number	Description
IE-77810	Stand-alone in-circuit emulator for μPD77810

IE-77810 Block Diagram



IBM PC, PC/XT, and PC AT are registered trademarks of International Business Machines Corporation.

NEC Electronics Inc.

RA77810
μPD77810 Relocatable Assembler Package

PRELIMINARY INFORMATION

Description

The RA77810 Relocatable Assembler package converts symbolic source code for the μPD77810 Modem Digital Signal Processor (MDSP) into executable absolute address object code. The Relocatable Assembler package consists of five separate programs: an assembler (RA77810), a linker (LK77810), a locator (LC77810), a librarian (LB77810) and a concatenater (CN77810)

RA77810 has two assemblers: one for General Purpose Processor (GGP) and one for the Digital Signal Processor (DSP). Each assembler translates a symbolic source module file into a relocatable object module. Each assembler also produces a relocatable object module file and a listing file that can contain the assembly list, symbol list and cross-reference list.

LK77810 consists of a GPP linker and a DSP linker. LK77810 for the GPP combines relocatable object modules and other GPP linker load modules and converts them into a single relocatable load module. LK77810 for the DSP combines relocatable object modules, library modules when necessary, other DSP linker load modules, and converts them into an absolute load module. Each linker produces a link map and an absolute load module.

LC77810 is available only for the GPP. It converts a GPP relocatable object module with no external references or a GPP relocatable load module into an ASCII hexadecimal format absolute object code file.

LB77810 is available for only the DSP. It allows commonly used DSP relocatable object modules to be stored in one file and linked into multiple programs, greatly increasing programming efficiency. When a library file is included in the input of the DSP linker, the linker extracts only those modules required to resolve external references from the file and relocates and links them.

CN77810 combines a DSP absolute load module or a DSP relocatable object module and the GPP HEX file into a MSDP HEX file.

Features

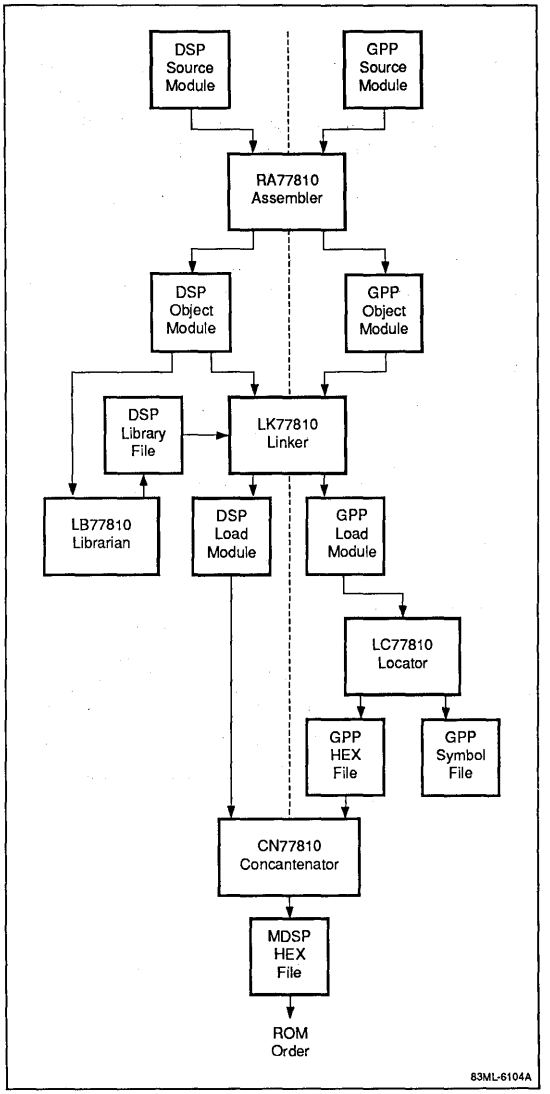
- Absolute address object code output
- User-selectable and directable output files
- Extensive error reporting
- Runs under the following operating systems:
  - MS-DOS®
  - VAX/VMS®
  - VAX/UNIX™ 4.2BSD or Ultrix™

Ordering Information

Part Number	Description
RA77810-D52	MS-DOS, 5.25" double density disk
RA77810-VVT1	VAX/VMS, 9-track 1600 BPI magnetic tape
RA77810-VXT1	VAX/UNIX 4.2BSD or Ultrix, 9-track 1600 BPI magnetic tape

MS-DOS is a registered trademark of Microsoft Corporation.
VAX, VMS and Ultrix are registered trademarks of Digital Equipment Corporation.
Ultrix is a trademark of Digital Equipment Corporation.
UNIX is a trademark of AT&T.

RA77810 Block Diagram



NV-300 System
μPD775X Family
Speech Analysis Tool

Description

The NV-300 system is a speech analysis system for use with the NEC μPD775X family of speech synthesis LSIs. The NV-300 plugs into an IBM PC AT® computer and is used to edit and encode analog original sound into the ADPCM code required for the μPD775X family. Using the NV-300 system, you can convert the original analog sound into digital data; trim and edit the digital data; play back the edited original sound data for evaluation; encode the edited original sound data into ADPCM code used by the μPD775X family; decode the ADCPM code into PCM code for further evaluation; convert the ADPCM code into HEX data for ROM/EPROM programming and final evaluation in your target hardware.

Features

- Full size IBM PC AT plug-in card
- Menu driven host software for:
  - Tape deck output level adjustments
  - A/D conversion of original sounds
  - Trimming of silent sections around original sound data
  - Editing original sound data
  - D/A conversion of edited sound for evaluation
  - Encoding edited sound into ADPCM code
  - Decoding of ADPCM data to PCM data for evaluation
  - Conversion of ADPCM data to μPD775X HEX file
- IBM PC AT or compatible host computer system:
  - EGA Color Monitor
  - EGA Card
  - At least 1MB extension RAM recommended
  - PC-DOS® or MS-DOS® operating system
- Uses I/O addresses 0220,0222,0224,0226H

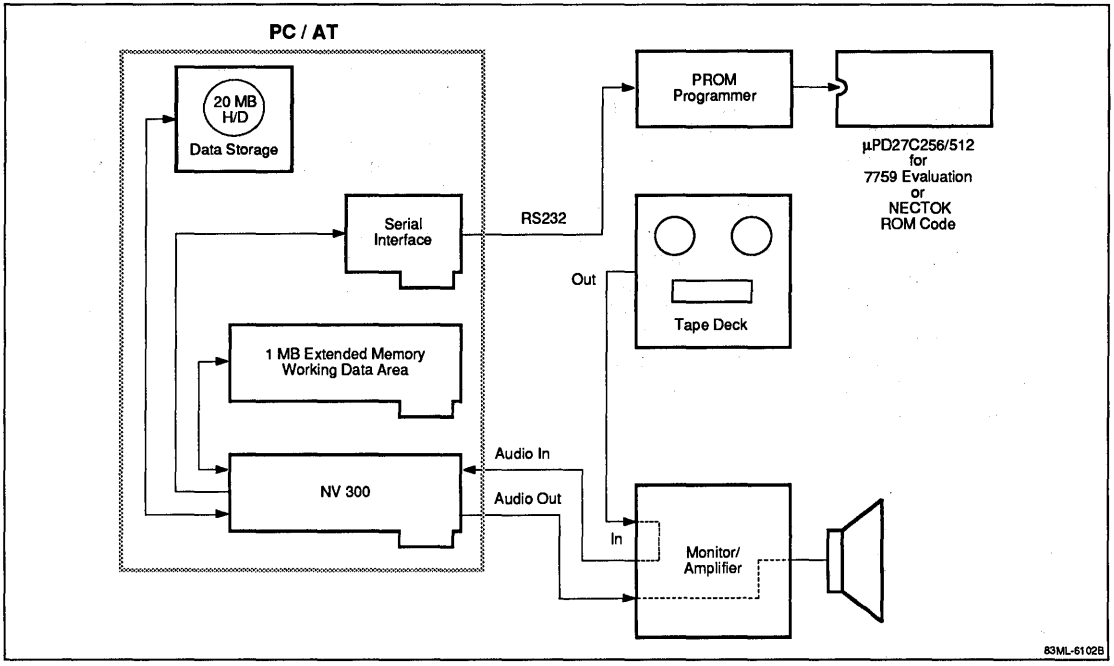
Ordering Information

Part Number	Description
NV-300	μPD775X family speech analysis system

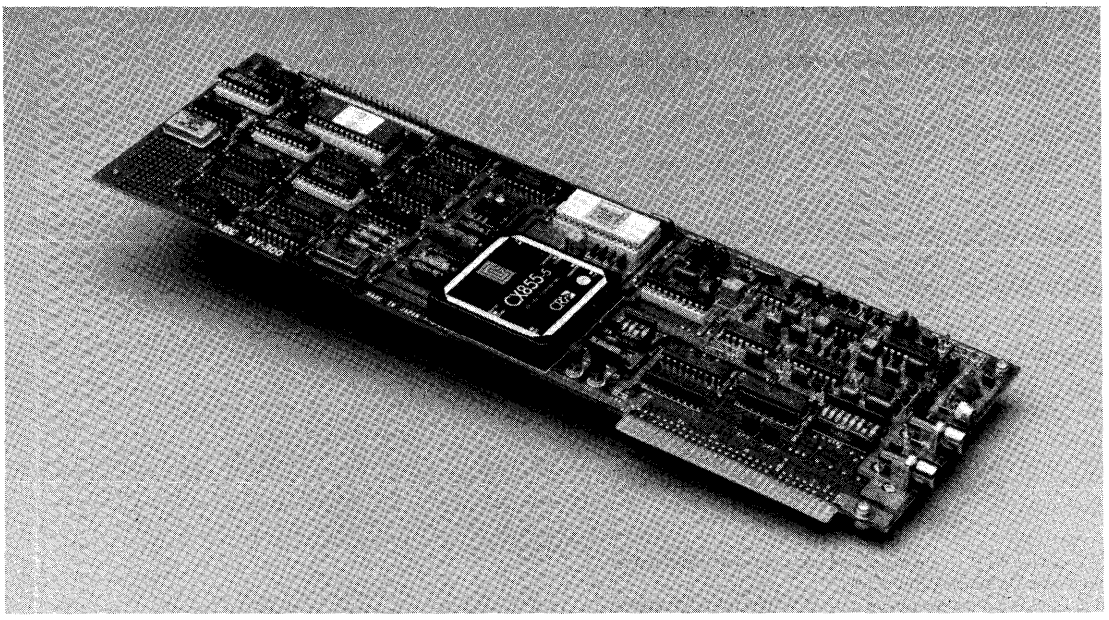
4

PC AT and PC-DOS are registered trademarks of International Business Machines Corporation.
MS-DOS is a registered trademark of Microsoft Corporation.

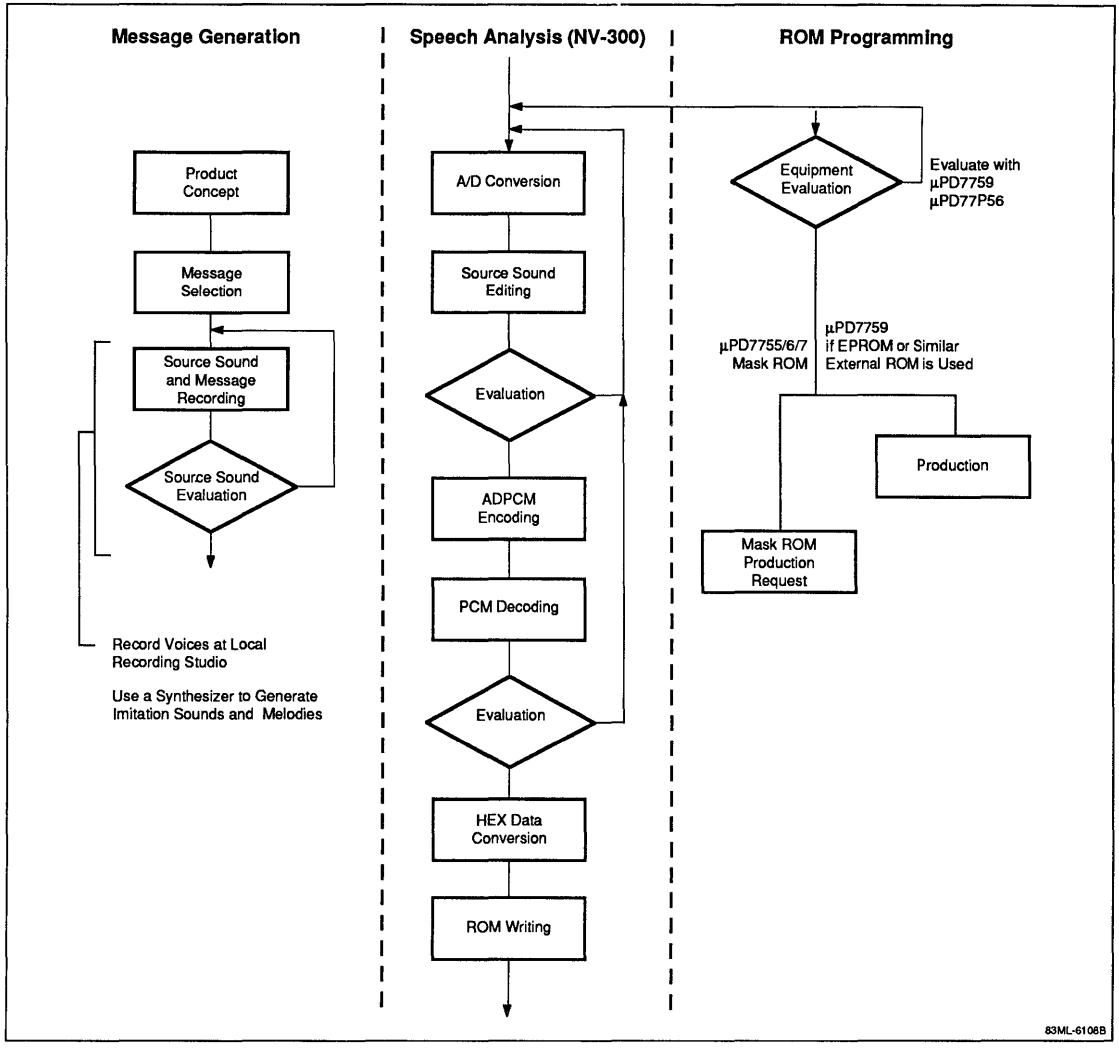
NV-300 Block Diagram



μPD775x Family Speech Analysis Tool



μPD775x Family Development Flowchart



4

NEC Electronics Inc.

EB-7759
μPD775X Demonstration and Evaluation Box

Description

The EB-7759 is demonstration and evaluation box for the μPD775X family of ADPCM speech synthesis LSI's. The EB-7759 can be used to demonstrate the speech synthesis capabilities of the μPD775X family by using NEC supplied sample messages or to evaluate the ADPCM code produced on the NV-300 speech analysis system. The EB-7759 can also be plugged into your target hardware to emulate the masked ROM parts, μPD7756/57.

The EB-7759 can be used as a stand-alone unit or may be controlled remotely via a Centronics interface from an IBM PC®, PC/XT® or PC AT® or compatibles using the supplied DBOX control software. Under remote control, concatenation of words and phrases is feasible, so that a wide variety of sentences can be built from a fixed vocabulary.

Features

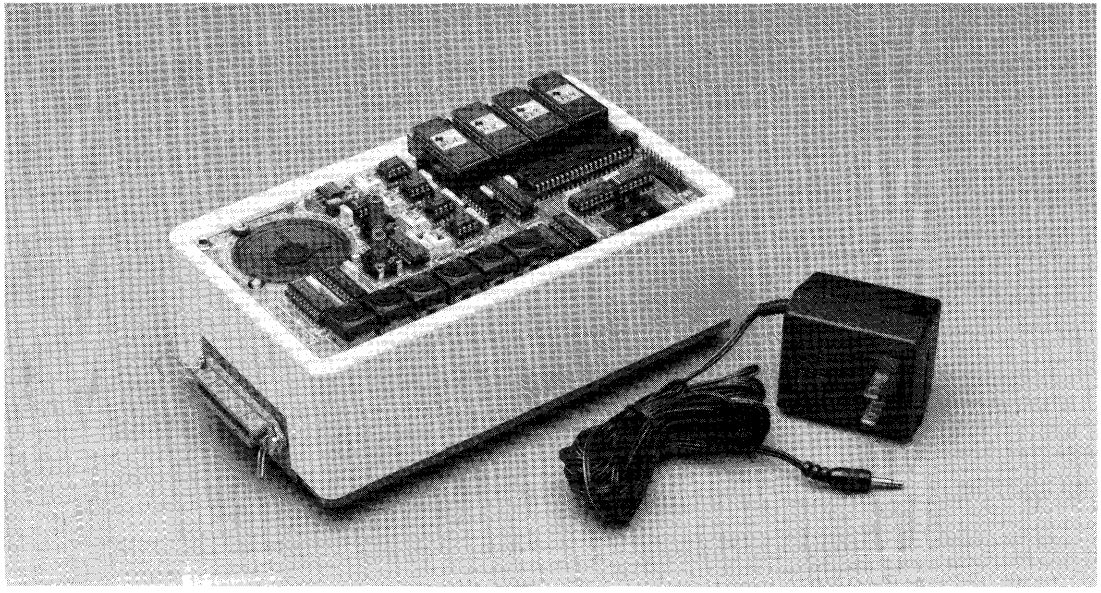
- Stand-alone demonstration and evaluation box
  - Supplied with external power supply
- Three operating modes
  - Stand-alone mode for speech evaluation
  - Remote control mode for speech evaluation allows concatenation of words and phrases
  - μPD7756/57 emulation mode
- μPD7759 ROMless ADPCM speech synthesizer
- Sockets for up to 1M bit of EPROM
  - Four 27256's
  - One 27C1000 or 27C1001
  - Sample messages provided in four 27256's
- User-selectable lowpass output filters
  - 4/5/6/8-kHz sampling rates
- 18-pin emulation probe
- IBM PC DBOX controller software
  - Windowed display
  - Allows concatenation of up to 26 recorded words/phrases with pauses of 1 to 10,000 ms
  - Allows use of labels to access messages
  - Read/store labels or phrase patterns from/to disk
  - Automatically generate multiple combinations of phrase patterns
- Complete hardware schematics provided

IBM PC, PC/XT and PC AT are registered trademarks of International Business Machines Corporation.

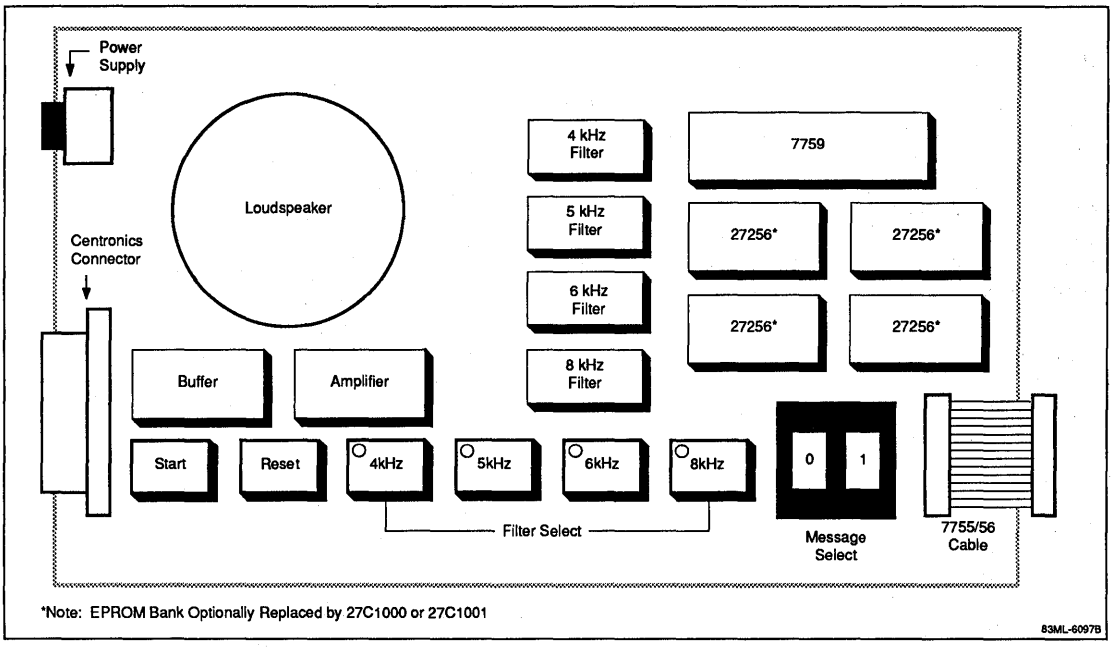
Ordering Information

Part Number	Description
EB-7759	Demonstration/Evaluation Box for μPD775X

μPD775x Demonstration and Evaluation Box



EB-7759 Block Diagram



Description

The PG-1500 series is a stand-alone EPROM programmer for programming 256-kilobit to 1-megabit EPROMs and EPROM/OTP devices for NEC's 4/8/16-bit single-chip microcomputers and digital signal processors. The system consists of the PG-1500 base programmer, interchangeable programmer adapter modules for standard EPROM devices and the μPD75XX/75XXX series 4-bit microcomputers, and a variety of programmer adapters to support the individual devices and package types. The PG-1500 can be controlled from either a remote terminal or host computer via an RS-232C serial port, or directly from the on-board keypad in stand-alone mode.

Features

- Interchangeable modules for programming:
  - 256-kilobit to 1-megabit EPROMs
  - NEC μPD75XX and μPD75XXX series 4-bit microcomputers
  - NEC μPD78XX and μPD78XXX series 8-bit microcomputers
  - NEC V-series 16-bit microcomputers
  - NEC μPD77XXX digital signal processors
- 512K-bytes data RAM
- Silicon signature read function
- PROM insertion error detection circuitry
- Address splitting for 16/32-bit microprocessors
- Memory edit function to change/confirm PG-1500 buffer
- Address/data/message display LCD
- RS-232C serial interface
- Centronics compatible parallel interface
- Power-on diagnostics
- Supports three data transfer formats
  - Intel extended hex (Note 1)
  - Extended Tektronix hex (Note 2)
  - Motorola S (Note 3)
- Two modes of operation
  - Remote controlled
  - Stand-alone

Notes:

(1) Developed by Intel Corporation.

(2) Developed by Tektronix Corporation.

(3) Developed by Motorola Inc.

Ordering Information

Part Number	Description
PG-1500	PG-1500 Series EPROM Programmer for 27XXX EPROMS, NEC 4/8/16 microcomputers, and DSP devices (Includes 027A and 04A Programming Adapter Modules)
PA-70P322L	Programmer Adapter for μPD70P322K
PA-71P301GF	Programmer Adapter for μPD71P301GF
PA-71P301GQ	Programmer Adapter for μPD71P301GQ
PA-71P301KA	Programmer Adapter for μPD71P301KA
PA-71P301KB	Programmer Adapter for μPD71P301KB
PA-71P301L	Programmer Adapter for μPD71P301L
PA-75P54CS	Programmer Adapter for μPD75P54/64CS, μPD75P54/64G
PA-75P56CS	Programmer Adapter for μPD75P56/66CS, μPD75P56/66G
PA-75P008CU	Programmer Adapter for μPD75P008CU/DU/GB
PA-75P028CW	Programmer Adapter for μPD75P028CW
PA-75P028GC	Programmer Adapter for μPD75P028GC
PA-75P108CW	Programmer Adapter for μPD75P108CW/DW/BCW, μPD75P116CW
PA-75P116GF	Programmer Adapter for μPD75P108G/BGF, μPD75P116GF
PA-75P216ACW	Programmer Adapter for μPD75P216ACW
PA-75P308GF	Programmer Adapter for μPD75P308GF, μPD75P316GF
PA-75P308K	Programmer Adapter for μPD75P308K
PA-75P328GC	Programmer Adapter for μPD75P328GC
PA-75P402CT	Programmer Adapter for μPD75P402CT
PA-75P402GB	Programmer Adapter for μPD75P402GB
PA-75P516GF	Programmer Adapter for μPD75P516GF
PA-75P516K	Programmer Adapter for μPD75P516K
PA-77P25C	Programmer Adapter for μPD77P25C/D
PA-77P56	Programmer Adapter for μPD77P56C/G
PA-77P230R	Programmer Adapter for μPD77P230R
PA-78CP14CW	Programmer Adapter for μPD78CP14CW, DW
PA-78CP14GF	Programmer Adapter for μPD78CP14GF
PA-78CP14GQ	Programmer Adapter for μPD78CP14GQ/R
PA-78CP14L	Programmer Adapter for μPD78CP14L
PA-78P214CW	Programmer Adapter for μPD78P214CW
PA-78P214GC	Programmer Adapter for μPD78P214GC
PA-78P214GJ	Programmer Adapter for μPD78P214GJ

Ordering Information (cont)

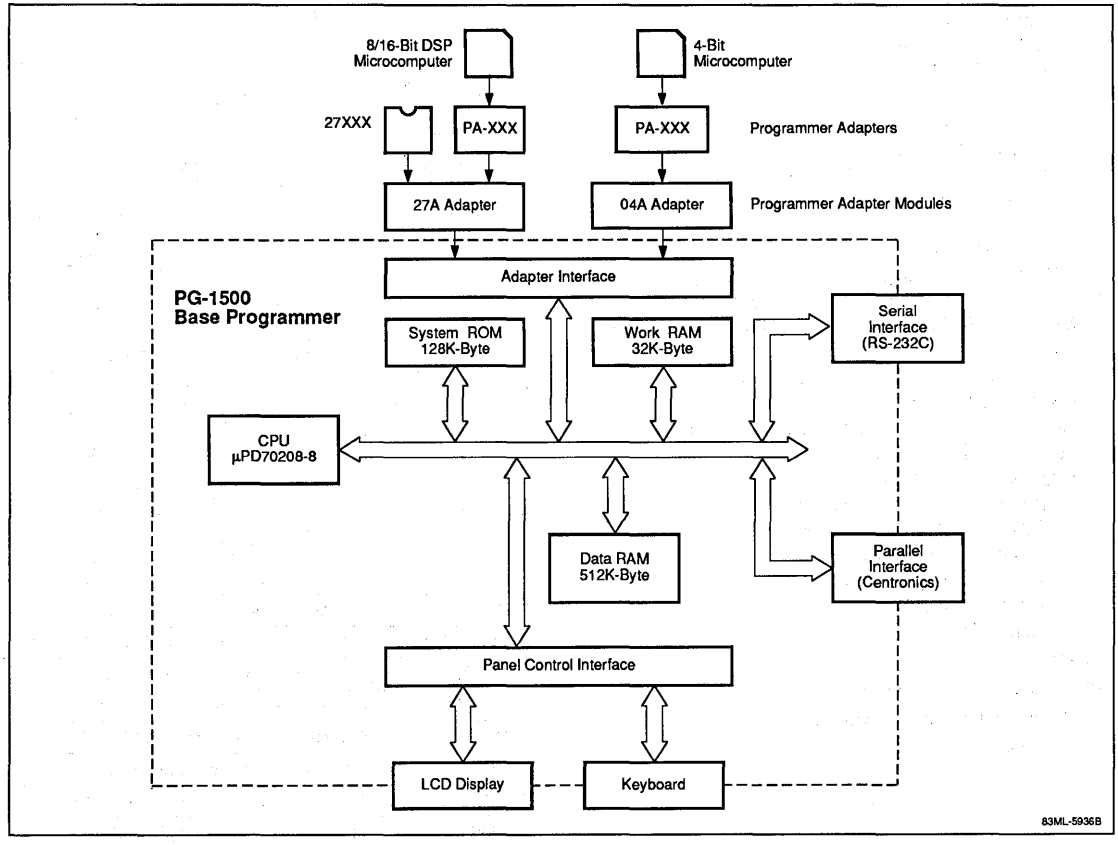
Part Number	Description
PA-78P214GQ	Programmer Adapter for μPD78P214GQ
PA-78P214L	Programmer Adapter for μPD78P214L
PA-78P224GJ	Programmer Adapter for μPD78P224GJ
PA-78P224L	Programmer Adapter for μPD78P224L
PA-78P312CW	Programmer Adapter for μPD78P312ACW/DW
PA-78P312GF	Programmer Adapter for μPD78P312AGF
PA-78P312GQ	Programmer Adapter for μPD78P312AGQ/R
PA-78P312L	Programmer Adapter for μPD78P312AL

Equipment Supplied

The PG-1500 package includes the following:

- PG-1500 EPROM Programmer Base Unit
- 027A Socket Board for 27XXX EPROMS and μPD27C256A-like devices
- 04A Interface Board for NEC μPD75XX/μPD75XXX Microcomputers
- Power Cord
- Power Ground Plug Adapter
- Spare Fuses (2)
- PG-1500 EPROM Programmer User's Manual
- Warranty Policy and Registration Card

Figure 1. PG-1500 System Block Diagram



Basic Specifications

- Power requirements:
  — 90 to 250 VAC, 50 to 60 Hz
- Environment conditions:
  — Operating temperature range: 10 to 35°C
  — Operating humidity range: 20 to 80% relative humidity
- RS-232C serial port:
  — Baud rates: 1200, 2400, 4800, 9600, 19200
  — Parity: none, even, odd
  — X-ON/X-OFF: on, off
  — Bit configuration: 7, 8
  — Stop bits: 1, 2

Architecture

The PG-1500 base unit contains an NEC μPD70208 (V40™) microprocessor with 128K bytes of monitor ROM, 32K bytes of working RAM, 512K bytes of data memory, an RS-232C serial port, a Centronics compatible parallel interface, an LCD display, and a 23-key keypad. Figure 1 shows a block diagram of the PG-1500.

The PG-1500 has two interchangeable programmer adapter modules: one for 27XXX EPROMS, NEC's 4/8/16 bit microcomputers, and DSP devices which use the μPD27C256A programming algorithm (027A board), and another for NEC's μPD75XX/75XXX 4-bit microcomputers which must be programmed in a serial fashion (04A board). These adapter modules plug directly into the top of the PG-1500 and can accept a wide variety of programmer socket adapters to support NEC's devices. Refer to the PG-1500 Programming Adapters Selection Guide for a list of all available adapters.

On power-up, the PG-1500 performs a self-diagnostic on its internal memory, its data bus, its power supply, and its reference voltages.

Operation

The PG-1500 operates in stand-alone mode from the on-board keypad, or in remote control mode from an external terminal or from a host computer via an RS-232C serial port.

Stand-Alone Mode

Table 1 lists the PG-1500 commands available in stand-alone mode.

Table 1. PG-1500 Commands in Stand-Alone Mode

Command	Function
DEVICE SELECT	Selects the EPROM to be used
DEVICE BLANK	Checks if the EPROM is blank
DEVICE COPY	Reads data from the EPROM
DEVICE PROG	Writes data into the EPROM
DEVICE VERIFY	Verifies EPROM contents against PG-1500 buffer
DEVICE CONT	Performs BLANK, PROG, VERIFY commands in sequence
EDIT CHANGE	Display/change the contents of the PG-1500 buffer
EDIT INITIAL	Initializes the PG-1500 buffer
EDIT MOVE	Moves a block of data within PG-1500 buffer
EDIT SEARCH	Searches PG-1500 buffer for 1-, 2-, or 4-byte patterns
EDIT C-SUM	Performs checksum on all data in PG-1500 buffer
FUNCTION S-IN	Inputs data from serial port in three formats
FUNCTION S-OUT	Outputs data from serial port in three formats
FUNCTION REMOTE	Sets PG-1500 to remote control mode
FUNCTION P-IN	Inputs data from parallel port in three formats
FUNCTION MODE	Sets up the RS-232C serial port parameters

The stand-alone commands fall into three groups:

- DEVICE commands associated with the device to be programmed
- EDIT commands for interacting with the PG-1500 memory buffer
- FUNCTION commands for setting up and controlling the PG-1500

The DEVICE commands are available to check if an EPROM device is blank, to copy data from the device to the PG-1500 buffer, to write the buffer data to the device, and to compare the data in the device with the data in the buffer. Blank checking, programming, and verification of the device can be performed sequentially using a single command.

To support various 16- and 32-bit microprocessors, the PG-1500 can split the data in its buffer in a variety of ways. When a data file is loaded into the PG-1500, the complete file is stored in the buffer and can be dynamically split during writing and verification. The PG-1500 supports the address splitting modes described in table 2.

Table 2. Address Splitting Modes

Mode	Description
Normal	The data is not split at all. Each byte of data in the buffer is programmed into the device.
16EVN	Each byte of data on an even address in the buffer is programmed into the device.
16ODD	Each byte of data on an odd address in the buffer is programmed into the device.
32/2E	The first two bytes of every four bytes in the buffer is programmed into the device.
32/2O	The third and fourth byte of every four bytes in the buffer is programmed into the device.
32/4E1	The first byte of every four bytes in the buffer is programmed into the device.
32/4O1	The second byte of every four bytes in the buffer is programmed into the device.
32/4E2	The third byte of every four bytes in the buffer is programmed into the device.
32/4O2	The fourth byte of every four bytes in the buffer is programmed into the device.

This method of address splitting also allows the complete original file to be recreated in the buffer when reading from a set of master EPROMs.

A silicon signature is stored in all NEC devices and contains information on the device type, start and stop addresses, and programming voltages. The PG-1500 can read the silicon signature of the particular device being programmed either manually or automatically, or the device code can be entered manually.

The EDIT commands initialize the PG-1500 buffer to a known value, move a block of data from one location to another, and change/display data at a particular address. The PG-1500 buffer can also be searched for all occurrences of any 1-, 2-, or 4-byte pattern. Finally, a checksum can be calculated for all the data contained in the buffer.

The FUNCTION commands control the setup of the RS-232C serial port, whether the PG-1500 checks for a PROM insertion error, whether the PG-1500 is operated through the serial port, and how data is input/output from the PG-1500. Data can be input to the PG-1500 through either the RS-232C serial port or the Centronics compatible parallel port in Intel Extended Hex, Extended Tektronix Hex, or Motorola S formats. Data can also be output via the RS-232C port in any of these three formats.

Remote Control Mode

Table 3 lists the PG-1500 commands available in Remote Control Mode.

Table 3. PG-1500 Commands in Remote Control Mode

Command	Function
RR	Reads data from the EPROM
RS	Selects the EPROM to be used
RV	Verifies EPROM contents against PG-1500 buffer
RW	Writes data into EPROM
RZ	Checks if EPROM is blank
MC	Change the contents of the PG-1500 buffer
MD	Displays the contents of the PG-1500 buffer
MF	Initializes the PG-1500 buffer
PI	Inputs data from parallel port (Intel Extended HEX)
PM	Inputs data from parallel port (Motorola S)
PT	Inputs data from parallel port (Extended Tektronix HEX)
LI	Inputs data from serial port (Intel Extended HEX)
LM	Inputs data from serial port (Motorola S)
LT	Inputs data from serial port (Extended Tektronix HEX)
SI	Outputs data from serial port (Intel Extended HEX)
SM	Outputs data from serial port (Motorola S)
ST	Outputs data from serial port (Extended Tektronix HEX)
??	Help command

Documentation

For further information on the operation of the PG-1500, NEC provides the following documentation:

- PG-1500 EPROM Programmer User's Manual

PACKAGE DRAWINGS

Section 5
Package Drawings

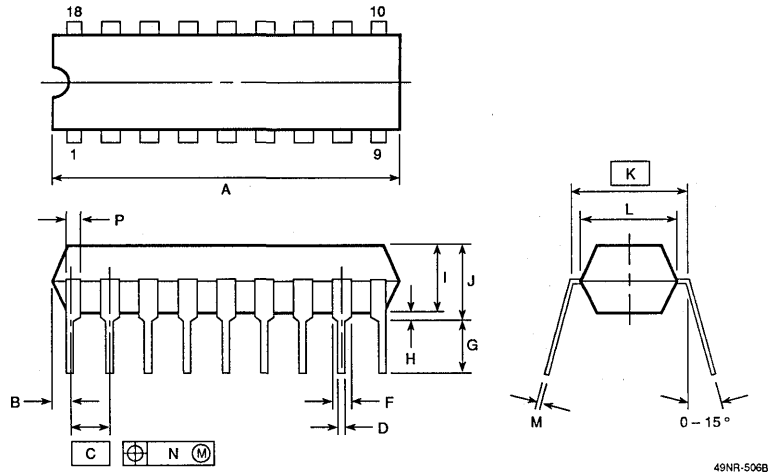
Package/Device Cross-Reference

Package	Device, μPD
18-Pin Plastic DIP (300 mil) (A, C Outline)	7755C 7756C
18-Pin Plastic DIP (300 mil) (SA Outline)	7757C
20-Pin Plastic DIP (300 mil)	77P56CR
24-Pin Plastic SOP (450 mil)	7755G 7756G 7757G 77P56G
28-Pin Plastic DIP (600 mil)	7720AC 77C20AC 77C25C 7730C 77C30C 77P25C
28-Pin Ceramic DIP (600 mil)	77P25D
28-Pin Cerdip	77P20D
28-Pin PLCC	77C20ALK
40-Pin Plastic DIP	7759C
40-Pin Ceramic DIP	7281D
44-Pin PLCC	77C20AL 77C30L 77C25L 7720AL 77P25L
52-Pin Plastic Miniflat	7759GC
68-Pin Ceramic PGA (A Outline)	77P230R
68-Pin Ceramic PGA (A-1 Outline)	77810R 77230AR 77230AR-003 77220R
68-Pin PLCC	77810L 77220L
132-Pin Ceramic PGA	9305R

18-Pin Plastic DIP (300 mil) (A,C Outline)

Item	Millimeters	Inches
A	22.86 max	.900 max
B	1.27 max	.050 max
C	2.54 (TP)	.100 (TP)
D	0.50 ±0.10	.020 $^{+.004}_{-.005}$
F	1.2 min	.047 min
G	3.5 ±0.3	.138 ±.012
H	0.51 min	.020 min
I	4.31 max	.170 max
J	5.08 max	.200 max
K *	7.62 (TP)	.300 (TP)
L	6.4	.252
M	0.25 $^{+0.10}_{-0.05}$	.010 $^{+.004}_{-.003}$
N	0.25	.010
P	1.0 min	.039 min

* Item K to center of leads when formed parallel.



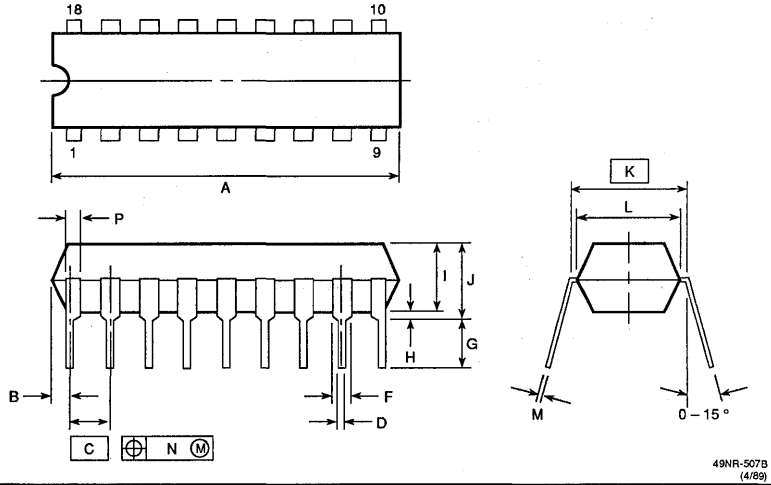
P18C-100-300A, C

49NR-506B (4/89)

18-Pin Plastic DIP (300 mil) (SA Outline)

Item	Millimeters	Inches
A	22.86 max	.900 max
B	1.27 max	.050 max
C	2.54 (TP)	.100 (TP)
D	0.50 ±0.10	.020 $^{+.004}_{-.005}$
F	1.2 min	.047 min
G	3.2 ±0.3	.126 ±.012
H	0.51 min	.020 min
I	4.31 max	.170 max
J	5.08 max	.200 max
K *	7.62 (TP)	.300 (TP)
L	6.7	.264
M	0.25 $^{+0.10}_{-0.05}$	.010 $^{+.004}_{-.003}$
N	0.25	.010
P	1.0 min	.039 min

* Item K to center of leads when formed parallel.



P18C-100-300SA

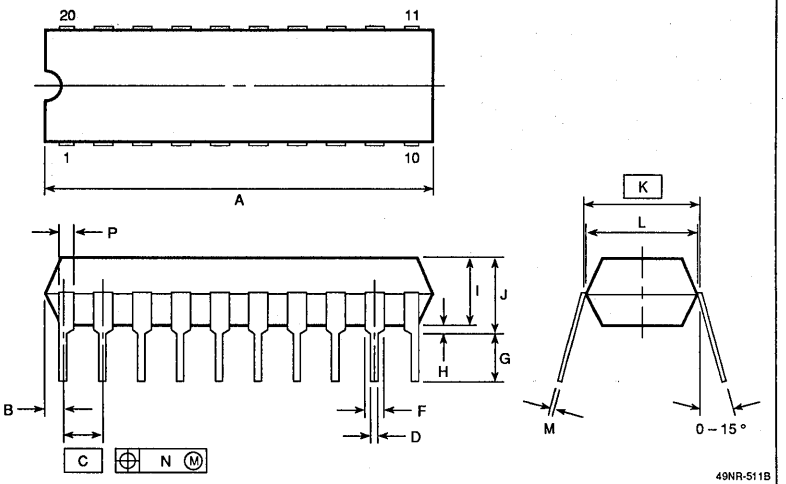
49NR-507B (4/89)

5

20-Pin Plastic DIP (300 mil)

Item	Millimeters	Inches
A	25.4 max	1.000 max
B	1.27 max	.050 max
C	2.54 (TP)	.100 (TP)
D	0.50 ±0.10	.020 $^{+.004}_{-.005}$
F	1.2 min	.047 min
G	3.2 ±0.3	.126 ±.012
H	0.51 min	.020 min
I	4.31 max	.170 max
J	5.08 max	.200 max
K *	7.62 (TP)	.300 (TP)
L	7.35	.289
M	0.25 $^{+0.10}_{-0.05}$	.010 $^{+.004}_{-.003}$
N	0.25	.010
P	1.0 min	.039 min

* Item K to center of leads when formed parallel.

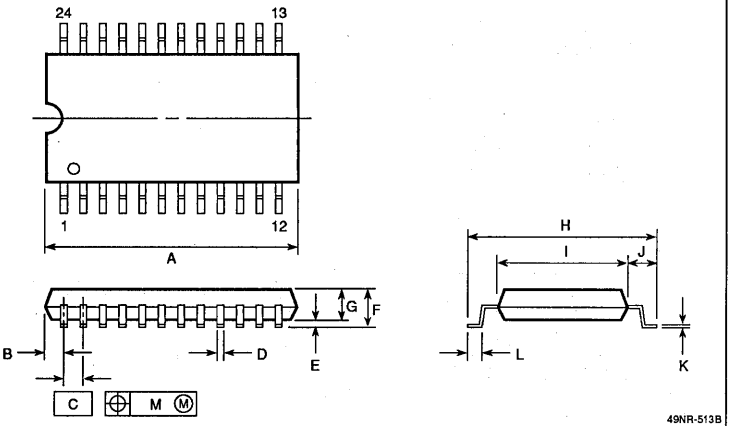


P20C-100-300WA

49NR-511B
(5/89)

24-Pin Plastic SOP (450 mil)

Item	Millimeters	Inches
A	16.51 max	.650 max
B	1.27 max	.050 max
C	1.27 (TP)	.050 (TP)
D	0.40 ±0.10	.016 $^{+.004}_{-.005}$
E	0.1 $^{+0.2}_{-0.1}$	.004 $^{+.008}_{-.004}$
F	2.5 max	.099 max
G	2.00	.079
H	12.2 ±0.3	.480 $^{+.013}_{-.012}$
I	8.4	.331
J	1.9	.075
K	0.15 $^{+0.10}_{-0.05}$	.006 $^{+.004}_{-.002}$
L	0.9 ±0.2	.035 $^{+.009}_{-.008}$
M	0.12	.005



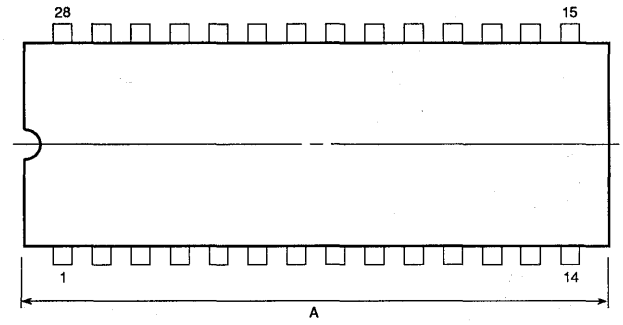
P24GM-50-450A

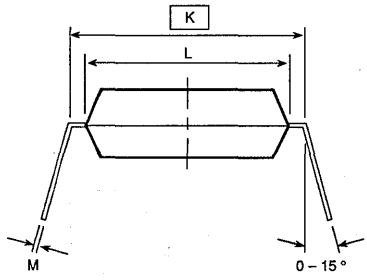
49NR-513B
(4/89)

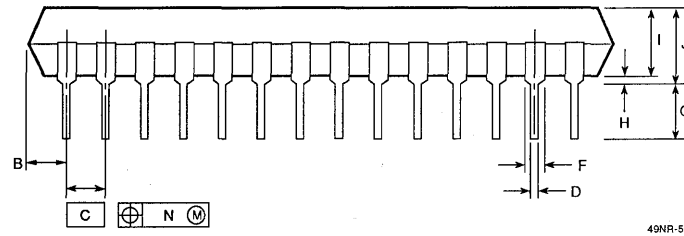
28-Pin Plastic DIP (600 mil)

Item	Millimeters	Inches
A	38.10 max	1.500 max
B	2.54 max	.100 max
C	2.54 (TP)	.100 (TP)
D	0.50 ±0.10	.020 $^{+.004}_{-.005}$
F	1.2 min	.047 min
G	3.6 ±0.3	.142 ±.012
H	0.51 min	.020 min
I	4.31 max	.170 max
J	5.72 max	.226 max
K*	15.24 (TP)	.600 (TP)
L	13.2	.520
M	0.25 $^{+0.10}_{-0.05}$	.010 $^{+.004}_{-.003}$
N	0.25	.010

* Item K to center of leads when formed parallel.







P28C-100-600A1

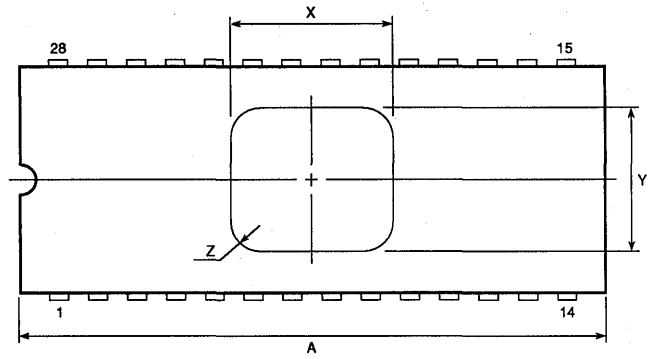
49NR-514B (5/89)

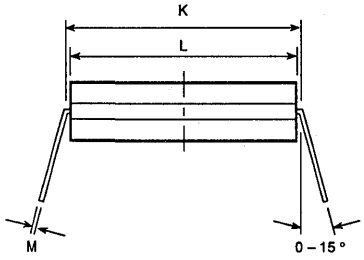
5

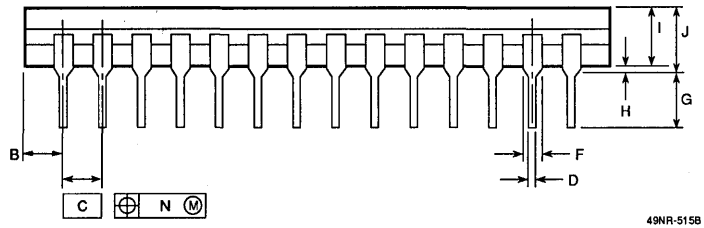
28-Pin Ceramic DIP (600 mil)

Item	Millimeters	Inches
A	38.10 max	1.500 max
B	2.54 max	.100 max
C	2.54 (TP)	.100 (TP)
D	0.50 ±0.10	.020 $^{+.004}_{-.005}$
F	1.2 min	.047 min
G	3.5 ±0.3	.138 ±.012
H	0.51 min	.020 min
I	3.80	.150
J	5.08 max	.200 max
K *	15.24 (TP)	.600 (TP)
L	14.66	.577
M	0.25 ±0.05	.010 $^{+.002}_{-.003}$
N	0.25	.010
X	10.5	.413
Y	9.2	.362
Z	2.0 rad	.079 rad

* Item K to center of leads when formed parallel.







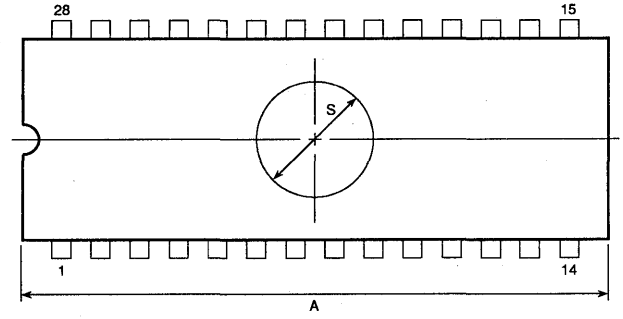
P28DW-100-600WA1

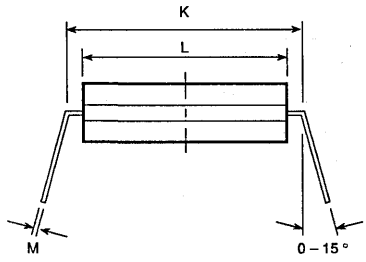
49NR-515B
(5/89)

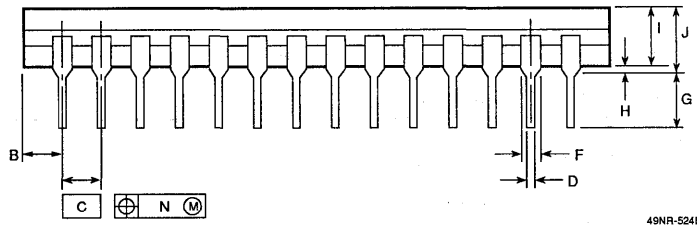
28-Pin Cerdip (600 mil)

Item	Millimeters	Inches
A	38.10 max	1.500 max
B	2.54 max	.100 max
C	2.54 (TP)	.100 (TP)
D	0.50 ±0.10	.020 $^{+.004}_{-.005}$
F	1.2 min	.047 min
G	3.5 ±0.3	.138 ±.012
H	0.51 min	.020 min
I	3.80	.150
J	5.08 max	.200 max
K *	15.24 (TP)	.600 (TP)
L	13.21	.520
M	0.25 ±0.05	.010 $^{+.002}_{-.003}$
N	0.25	.010
S	7.62 dia	.300 dia

* Item K to center of leads when formed parallel.







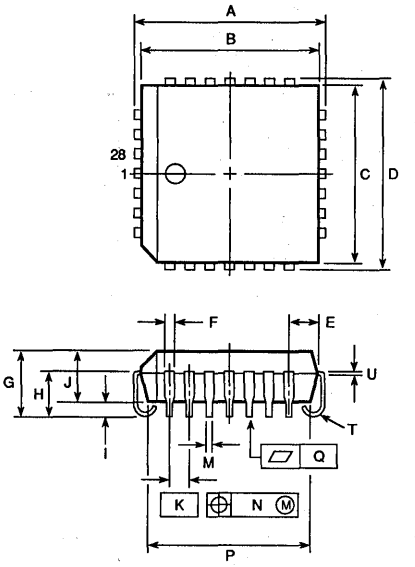
P28DW-100-600A

49NR-524B
(5/89)

5

28-Pin PLCC

Item	Millimeters	Inches
A	12.45 ±0.2	.490 ±.008
B	11.50	.453
C	11.50	.453
D	12.45 ±0.2	.490 ±.008
E	1.94 ±0.15	.076 $^{+.007}_{-.008}$
F	0.6	.024
G	4.4 ±0.2	.173 $^{+.009}_{-.008}$
H	2.8 ±0.2	.110 $^{+.009}_{-.008}$
I	0.9 min	.035 min
J	3.4	.134
K	1.27 (TP)	.050 (TP)
M	0.40 ±0.10	.016 $^{+.004}_{-.005}$
N	0.12	.005
P	10.42 ±0.20	.410 $^{+.009}_{-.008}$
Q	0.15	.006
T	0.8 rad	.031 rad
U	0.20 $^{+0.10}_{-0.05}$	.008 $^{+.004}_{-.002}$

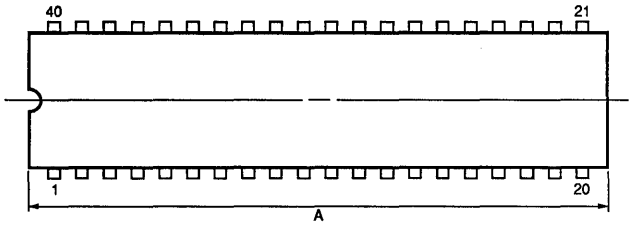


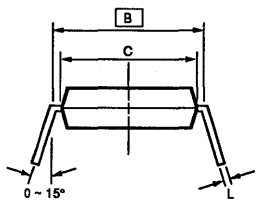
P28L-50A1

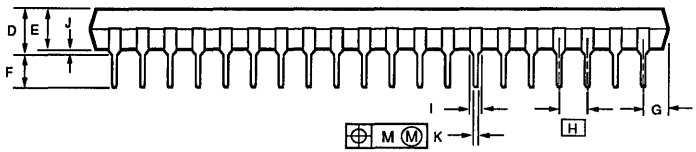
49NR-525B
(5/89)

40-Pin Plastic DIP (600 mil)

Item	Millimeters	Inches
A	53.34 max	2.100 max
B	15.24 [TP]	.600 [TP]
C	13.2	.520
D	5.72 max	.225 max
E	4.31 max	.170 max
F	3.6 ±0.3	.142 ±.012
G	2.54 max	.100 max
H	2.54 [TP]	.100 [TP]
I	1.2 min	.047 min
J	0.51 min	.020 min
K	0.50 ±0.10	.020 ±.004
L	0.25 $^{+0.10}_{-0.05}$	.010 $^{+.004}_{-.002}$
M	0.25	.010







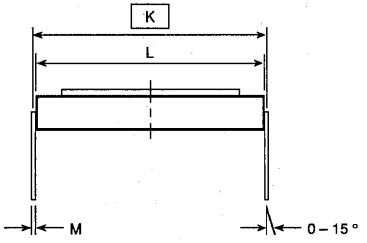
P40C-100-600A

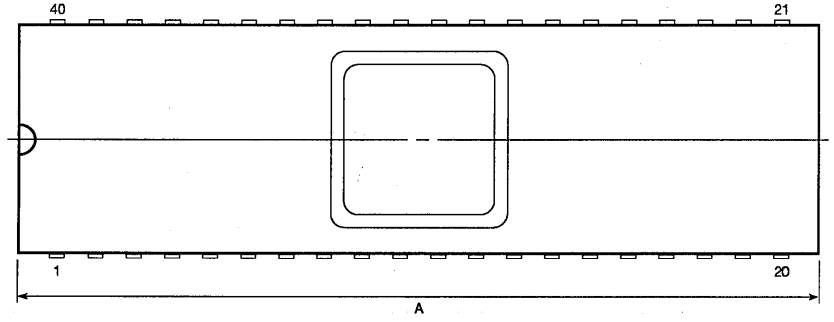
83vQ-6140B

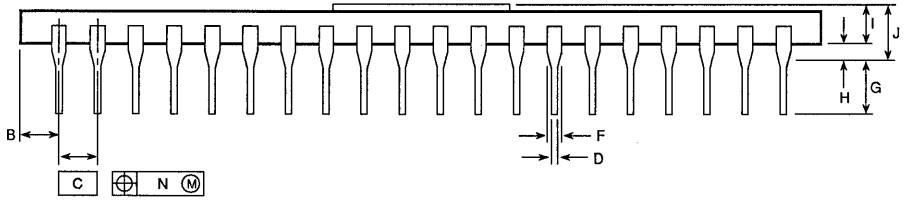
40-Pin Ceramic DIP (600 mil)

Item	Millimeters	Inches
A	53.34 max	2.100 max
B	2.54 max	.100 max
C	2.54 (TP)	.100 (TP)
D	0.46 ±0.05	.018 ±.002
F	0.92 min	.036 min
G	3.5 ±0.3	.138 ±.012
H	1.0 min	.039 min
I	2.64	.104
J	4.57 max	.180 max
K *	15.24 (TP)	.600 (TP)
L	14.93	.588
M	0.25 ±0.05	.010 $^{+.002}_{-.003}$
N	0.25	.010

* Item K to center of leads when formed parallel.







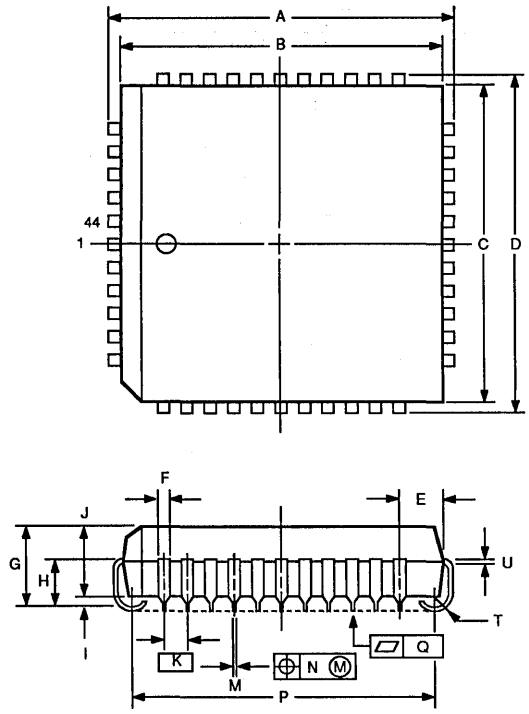
P40D-100-600A

49NR-526B
(5/89)

5

44-Pin PLCC

Item	Millimeters	Inches
A	17.5 ±0.2	.689 ±.008
B	16.58	.653
C	16.58	.653
D	17.5 ±0.2	.689 ±.008
E	1.94 ±0.15	.076 ±.006
F	0.6	.024
G	4.4 ±0.2	.173 ±.008
H	2.8 ±0.2	.110 ±.008
I	0.9 min	.035 min
J	3.4	.134
K	1.27 (TP)	.050 (TP)
M	0.40 ±0.10	.016 ±.004
N	0.12	.005
P	15.50 ±0.20	.610 ±.008
Q	0.15	.006
T	0.8 radius	.031 radius
U	$0.20^{+0.10}_{-0.05}$	$.008^{+.004}_{-.002}$

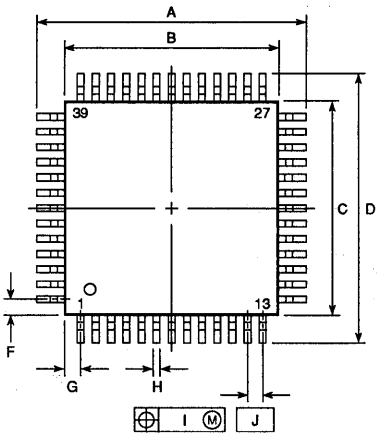


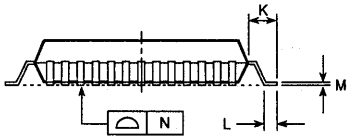
P44L-50A1

6/89
83YL-5804B

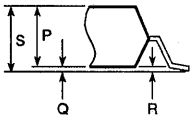
52-Pin Plastic Miniflat

Item	Millimeters	Inches
A	17.6 ±0.4	.693 ±.016
B	14.0 ±0.2	.551 $^{+.009}_{-.008}$
C	14.0 ±0.2	.551 $^{+.009}_{-.008}$
D	17.6 ±0.4	.693 ±.016
F	1.0	.039
G	1.0	.039
H	0.40 ±0.10	.016 $^{+.004}_{-.005}$
I	0.20	.008
J	1.0 (TP)	.039 (TP)
K	1.8 ±0.2	.071 $^{+.008}_{-.009}$
L	0.8 ±0.2	.031 $^{+.009}_{-.008}$
M	0.15 $^{+0.10}_{-0.05}$	.006 $^{+.004}_{-.003}$
N	0.15	.006
P	2.7	.106
Q	0.1 ±0.1	.004 ±.004
R	0.1 ±0.1	.004 ±.004
S	3.0 max	.119 max





Enlarged detail of lead end



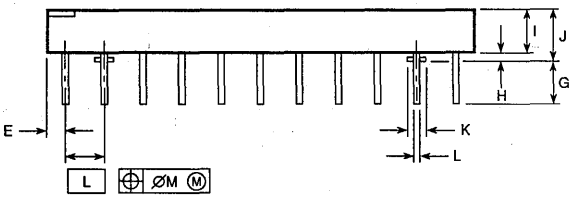
P52GC-100-3B6

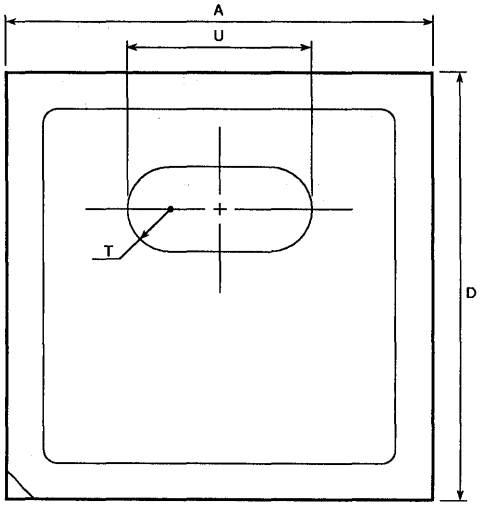
49NR-493B
(5/89)

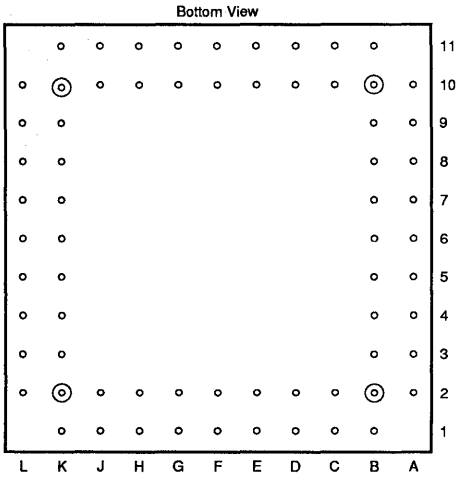
5

68-Pin Ceramic PGA (A Outline)

Item	Millimeters	Inches
A	27.94 ±0.4	1.100 $^{+.016}_{-.015}$
D	27.94 ±0.4	1.100 $^{+.016}_{-.015}$
E	1.25	.049
F	2.54 (TP)	.100 (TP)
G	2.8 ±0.3	.110 $^{+.012}_{-.011}$
H	0.5 min	.019 min
I	2.94	.116
J	4.57 max	.180 max
K	1.2 ±0.2 dia	.047 $^{+.008}_{-.007}$
L	0.46 ±0.05 dia	.018 $^{+.002}_{-.001}$
M	0.5	.020
T	3.0 rad	.118 rad
U	12.0	.472





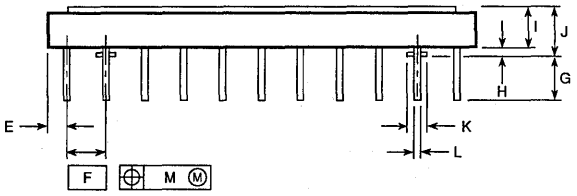


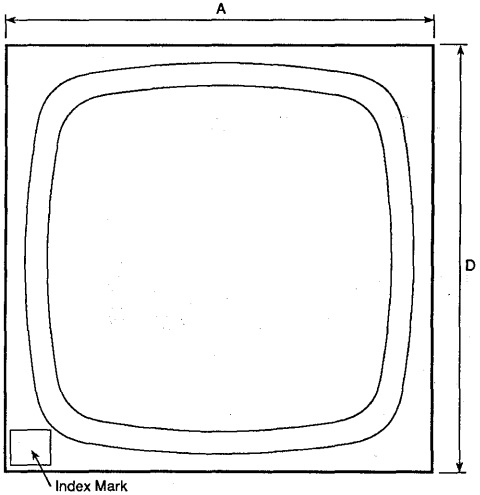
X68RW-100A

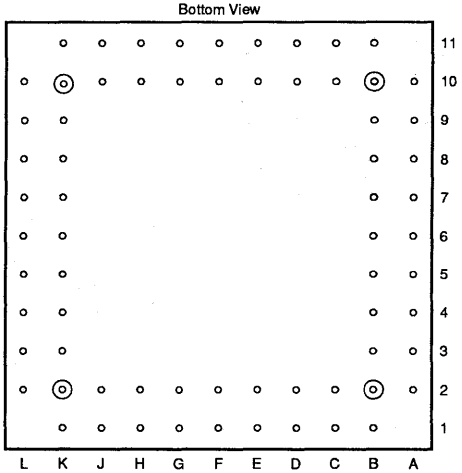
49NR-527B (5/89)

68-Pin Ceramic PGA (A-1 Outline)

Item	Millimeters	Inches
A	27.94 ±0.4	1.100 $^{+.016}_{-.015}$
D	27.94 ±0.4	1.100 $^{+.016}_{-.015}$
E	1.27	.050
F	2.54 (TP)	.100 (TP)
G	2.8 ±0.3	.110 $^{+.012}_{-.011}$
H	0.5 min	.019 min
I	2.70	.106
J	4.57 max	.180 max
K	1.2 ±0.2 dia	.047 $^{+.008}_{-.007}$
L	0.46 ±0.05 dia	.018 $^{+.002}_{-.001}$
M	0.5	.020







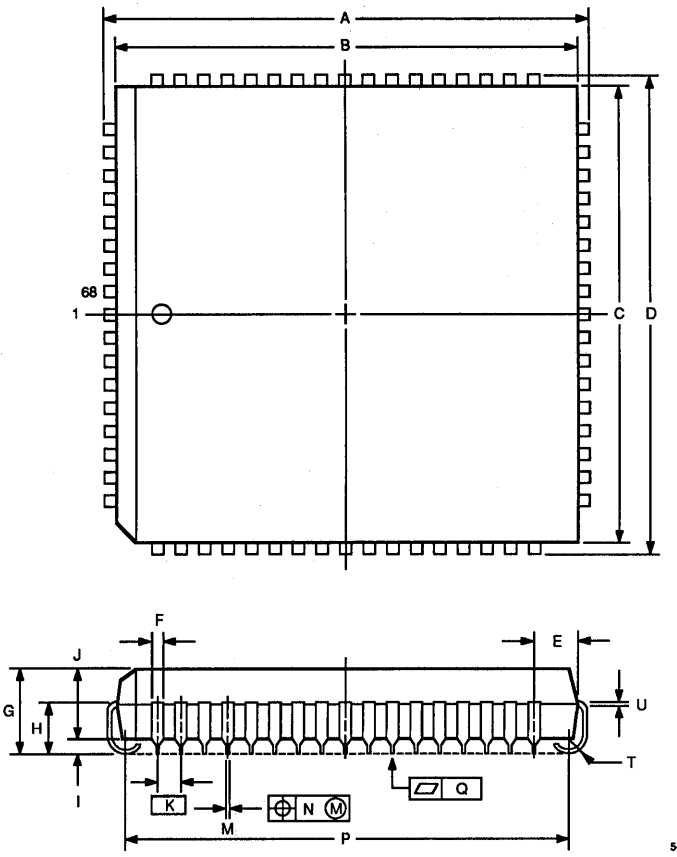
X68R-100A-1

49NR-528B
(5/89)

5

68-Pin PLCC

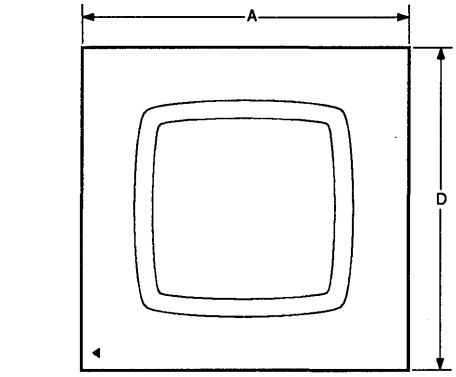
Item	Millimeters	Inches
A	25.2 ±0.2	.992 ±.008
B	24.20	.953
C	24.20	.953
D	25.2 ±0.2	.992 ±.008
E	1.94 ±0.15	$.076^{+.007}_{-.006}$
F	0.6	.024
G	4.4 ±0.2	$.173^{+.009}_{-.008}$
H	2.8 ±0.2	$.110^{+.009}_{-.008}$
I	0.9 min	.035 min
J	3.4	.134
K	1.27 (TP)	.050 (TP)
M	0.40 ±0.10	$.016^{+.004}_{-.005}$
N	0.12	.005
P	23.12 ±0.20	$.910^{+.009}_{-.008}$
Q	0.15	.006
T	0.8 radius	.031 radius
U	$0.20^{+0.10}_{-0.05}$	$.008^{+.004}_{-.002}$

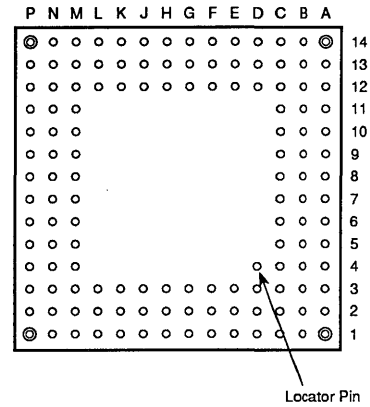


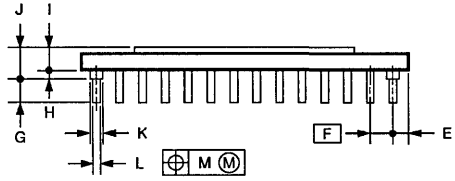
SSD-20615-1 (10-13-86)
P68L-50A1

5-03-89
83YL-5561B

132-Pin Ceramic PGA







Item	Millimeters	Inches
A	35.56 ±0.4	$1.400^{+.016}_{-.015}$
D	35.56 ±0.4	$1.400^{+.016}_{-.015}$
E	1.27	.050
F	2.54 (TP)	.100 (TP)
G	2.8 ±0.3	$.110^{+.012}_{-.011}$
H	0.5 min	.019 min
I	2.9	.114
J	4.57 max	.180 max
K	ø1.2 ±0.2	$ø.047^{+.008}_{-.007}$
L	ø0.46 ±0.05	$ø.018^{+.002}_{-.001}$
M	0.5	.020

X132R-100A

6/89
83YL-5817B

5

Notes:

Notes:

Notes:

NEC

FIELD SALES OFFICES

NORTHEAST REGION

One Natick Executive Park
Natick, MA 01760
TEL: 508-655-8833
FAX: 508-655-1605

200 Perinton Hills
Office Park
Fairport, NY 14450
TEL: 716-425-4590
FAX: 716-425-4594

2 Jefferson St.
Suite 103
Poughkeepsie, NY 12601
TEL: 914-452-4747
FAX: 914-471-2853

Two Jericho Plaza
Jericho, NY 11753
TEL: 516-932-5700
FAX: 516-932-5710

Six Neshaminy Interplex
Suite 203
Trevose, PA 19047
TEL: 215-638-8989
FAX: 215-638-1794

SOUTH CENTRAL REGION

16475 Dallas Parkway
Suite 380
Dallas, TX 75248
TEL: 214-931-0641
FAX: 214-931-1182

SOUTHEAST REGION

6625 The Corners Parkway
Suite 210
Norcross, GA 30092
TEL: 404-447-4409
FAX: 404-447-8228

901 Lake Destiny Drive
Suite 320
Maitland, FL 32751
TEL: 407-875-1145
FAX: 407-875-0962

2525 Meridian Parkway
Suite 320
Durham, NC 27713
TEL: 919-544-4132
FAX: 919-544-4109

MIDWEST REGION

1500 West Shure Drive
Suite 250
Arlington Heights, IL 60004
TEL: 312-577-9090
FAX: 312-577-2147

340 E. Big Beaver Road
Suite 210
Troy, MI 48083
TEL: 313-680-0506
FAX: 313-680-1015

1550 East 79th Street
Suite 805
Bloomington, MN 55425
TEL: 612-854-4443
FAX: 612-854-1346

Busch Corporate Center
6480 Busch Blvd., Suite 121
Columbus, OH 43229
TEL: 614-436-1778
FAX: 614-436-1769

30050 Chagrin Blvd.
Suite 320
Pepper Pike, OH 44124
TEL: 216-831-0067
FAX: 216-831-0758

NORTHWEST REGION

401 Ellis Street
P.O. Box 7241
Mountain View, CA 94039
TEL: 415-965-6200
FAX: 415-965-6683

Two Lincoln Center
10220 S.W. Greenburg Road
Suite 125
Portland, OR 97223
TEL: 503-245-1600
FAX: 503-245-3716

14001 East Iliff Avenue
Suite 411
Aurora, CO 80014
TEL: 303-755-6353
FAX: 303-755-6728

SOUTHWEST REGION

200 E. Sandpointe, Bldg. 8
Suite 460
Santa Ana, CA 92707
TEL: 714-546-0501
FAX: 714-432-8793

Encino Office Park Two
6345 Balboa Blvd.
Suite 240
Encino, CA 91316
TEL: 818-342-3111
FAX: 818-342-0842

NEC
NEC Electronics Inc.
CORPORATE HEADQUARTERS
401 Ellis Street
P.O. Box 7241
Mountain View, CA 94039
TEL 415-960-6000
TLX 3715792

For literature, call toll-free 8 a.m. to 4 p.m. Pacific time:
1-800-632-3531

DOC NO. 50052