TEXAS INSTRUMENTS

7000

TMS7000 Family Microarchitecture

MICROCOMPUTER SERIES™

User's Guide

IMPORTANT NOTICES

Texas Instruments reserves the right to make changes at any time to improve design and to supply the best possible product for the spectrum of users.

Information contained in these publications is believed to be accurate and reliable. However, responsibility is neither assumed for its use nor for any infringement of patents or rights of others that may result from its use. No license is granted by implication or otherwise under any patent or patent right of Texas Instruments or others.

TMS7000 FAMILY MICROARCHITECTURE USER'S GUIDE

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

SECTION 1

INTRODUCTION

1.1 GENERAL INFORMATION

The Texas Instruments TMS7000 Family of single-chip microcomputers is based around a microprogrammable Central Processing Unit (CPU), which can be interfaced to combinations of on-chip RAM, ROM, and I/O circuitry to provide a powerful family of single-chip microcomputers. The TMS7000 CPU implements a 64K byte logical address space and interfaces with I/O registers, timer circuit controls, and other useful on-chip functions by referencing certain addresses within the address space.

This document contains a description of the internal architecture of the TMS7000. It describes primarily the operation of CPU; the memory and on-chip I/O circuitry may vary among the TMS7000 family members, and will be described in the documentation for those individual devices. This document is intended to present information regarding the internal architecture of the TMS7000 family necessary for microcoding these devices. A symbolic microinstruction assembler called MICASM is provided for assembling microcode instruction mnemonics. This assembler is described in the TMS7000 Microassembler User's Guide.

Other information relating to the TMS7000 Family of microcomputers is contained in the following documents:

- TMS7000 8-Bit Microcomputer Data Manual (MP #008A)
- TMS7000 Assembly Language Programmer's Guide (MP #916)
- TMS7000 Microassembler User's Guide (MP #457)
- TMS7000 Microcode Development Guide (MP #458)
- TMS7000 Microprogrammer's Reference Card (MP #459)

The standard instruction set executed by the TMS7000 Family is described in the TMS7000 Assembly Language Programmer's Guide. Internally, the TMS7000 is a microprogrammed processor, whose operation is controlled by a sequence of microinstructions. The internal microprogram is stored in the CPU in the Control ROM, or CROM. Each microinstruction in the microprogram has the same format, consisting of bit fields which control the following operations:

- The gating of microregisters onto internal buses
- The operation of the internal Arithmetic Logic Unit (ALU)
- The appropriate shifting of the ALU results
- The gating of ALU results back into microregisters
- The memory controls to on-chip and off-chip memory
- The next microinstruction to be executed.

This format is called a horizontal microinstruction format, because every component of the microarchitecture is controlled by a single microinstruction word. Such an organization permits a high degree of parallelism in the operation of the microarchitecture.

1.2 INITIAL FAMILY MEMBERS

The TMS7000, TMS7020, and TMS7040 are the initial members of the TMS7000 8-bit Microcomputer Family. The TMS7020 provides 128 bytes of RAM, 2K bytes of ROM, and four 8-bit I/O ports. The TMS7040 provides the same 128 bytes of RAM but contains 4K bytes of ROM. The TMS7000 is identical to the TMS7020/7040, but it has no on-chip ROM. Throughout this document, TMS7000 will in general refer to any member of the TMS7000 family, unless explicitly stated otherwise.

1.3 TMS7000 FAMILY ADDRESS SPACE

The TMS7000 family address space is divided into multiple 256-byte pages. Addresses >0000 to >007F are utilized as a 128-byte Register File or RF, and reference the on-chip RAM. On-chip ROM is located at the top of the address space, from addresses >F800 to >FFFF for the TMS7020, and >F000 to >FFFF for the TMS7040. The last 48 bytes of memory, addresses >FFD0 to >FFFF, are reserved for trap and interrupt vectors. The TMS7000 family address space is shown in Figure 1-1.

The Peripheral File, or PF, is a special 256-byte page in the memory address space. Each location of the PF is a special control or data register. On-chip circuitry interprets PF Registers as I/O control, programmable timer, memory expansion, and other registers to control features of the chip. For example, the four I/O ports may be accessed as four registers in the PF. Accesses to the Peripheral File are recognized by the Peripheral/Memory Controller (PMC) external to the CPU. In general, all chip functions not implemented by the CPU will be implemented by the Peripheral/Memory Controller, and controlled via accesses to Peripheral File Registers.

The advantage of defining special pages for the Peripheral and Register files is that accesses to these areas may be made by specifying an offset of 8 bits, rather than a full 16-bit memory address. The Register File is located at memory addresses >0000 thru >007F and the Peripheral File is implemented in the second page of memory address space, from addresses >0100 to >01FF.

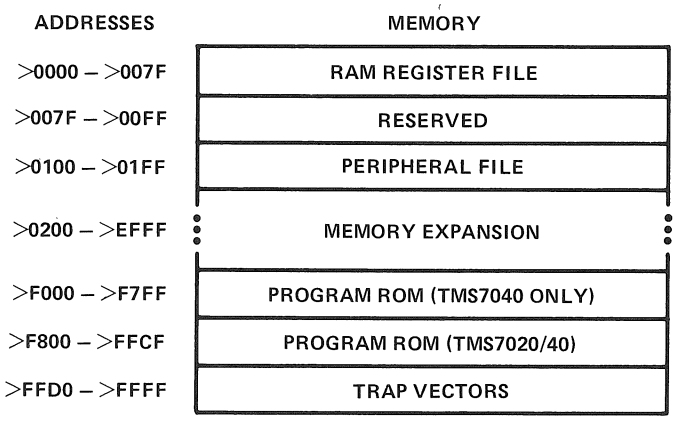


FIGURE 1-1 — TMS7000 FAMILY ADDRESS SPACE

1.4 BASIC TMS7000 ARCHITECTURE

The major components of the TMS7000 architecture are the CPU, the Peripheral/Memory Controller, and the RAM and ROM. These components and their interconnections are shown in Figure 1-2.

The Central Processing Unit (CPU) contains the internal registers, which store the operands of an instruction, and the Arithmetic Logic Unit (ALU), which operates on the internal register values. A shifter is provided to rotate the output of the ALU before its results are either stored in an internal CPU register or written to a memory location. The CPU is described in further detail in Section 3.

The Peripheral/Memory Controller (PMC) is a collection of modules which interface the CPU with the I/O ports, memory, and the interrupt inputs. The CPU is connected to the PMC via the Address Low (AL), Address High (AH), Memory Data (MD) and Control (C) Buses. The MD Bus, AL Bus, and AH Bus are also connected to the on-chip RAM and ROM memories.

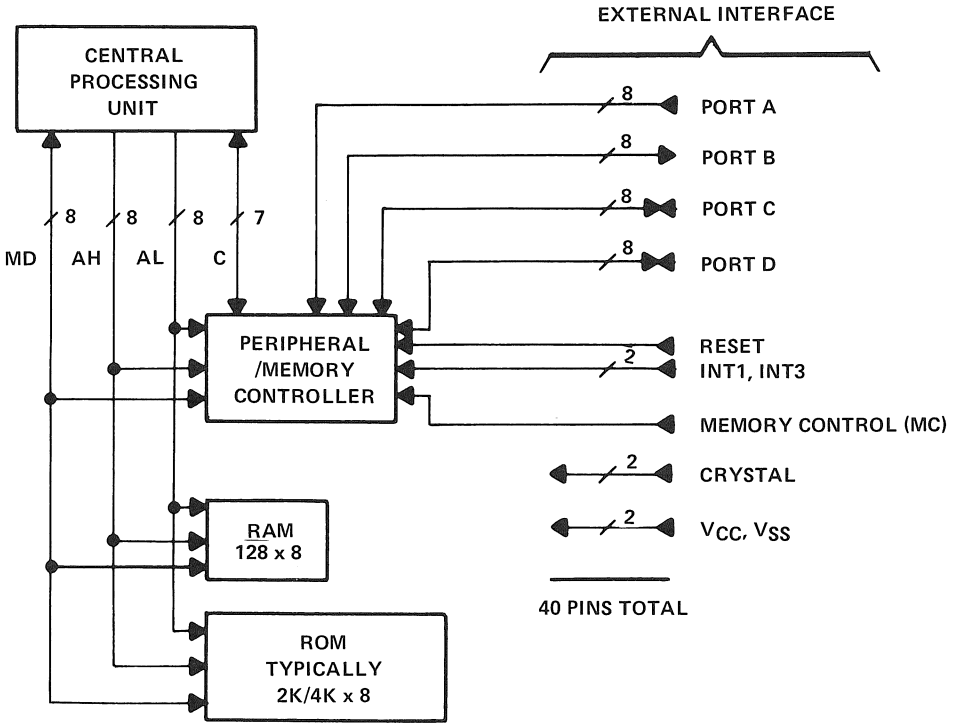


FIGURE 1-2 — TMS7000 OVERALL BLOCK DIAGRAM

The Peripheral/Memory Controller (PMC) performs many functions. It interfaces the CPU to the outside world by providing control and data registers for I/O ports, interrupts, and internal timer controls. The interface control registers appear to the CPU as addresses in the Peripheral File. In the TMS7000, the PF is implemented in the second 256 byte page of memory, at addresses >0100 to >01FF. Input/output in the TMS7000 is accomplished by reading and writing bytes in the Peripheral File implemented by the PMC. In terms of the microarchitecture, the exact functions of the Peripheral File registers are family member dependent.

The Control (C) Bus connecting the PMC and the CPU carries control information required in the interface between these two subsections of the TMS7000. The C Bus is made up of seven signals, each of which is described briefly below.

- #MEM (Memory) — set by the CPU during any memory access.
- #MEMCNT (Memory Continue) — set by the CPU during the first cycle of two cycle memory accesses.
- #WR (Write) — set to 1 by the CPU to indicate a memory write operation.
- STINT (Status Interrupt Enable) — set by the CPU to allow the PMC to assert IACT.
- IACT (Interrupt Active) — set by the PMC if a valid interrupt is active and STINT is a 1.
- RST (Reset) — set to 1 by the PMC whenever the external RESET pin is a 0.
- OTMD (O Bus to MD Bus Enable) — set by the PMC to enable the O Bus to drive the MD Bus.

Each of these signals is discussed in greater depth in later sections of this manual. Further details of interrupt control may be found in the TMS7000 8-Bit Microcomputer Data Manual (Part Number MP 008A).

SECTION 2

MICROINSTRUCTION EXECUTION

2.1 MICROINSTRUCTION FORMAT

This section describes the format of the TMS7000 microinstructions, and details the internal timing of microinstruction execution.

The CROM is organized as a 64-bit wide, 160-word memory. The current microarchitecture of the TMS7000 uses 45 bits per microinstruction to control its operation. To allow for future expansion of this architecture, however, a total of 64 microinstruction bits are reserved in the architecture definition. Table 2-1 describes the format of the TMS7000 microinstruction word.

TABLE 2-1 — MICROINSTRUCTION WORD FORMAT

BITS	FIELD	FUNCTION	SECTION
63-56	#JMPADDR(7-0)	BASE ADDRESS FOR NEXT INSTRUCTION	4.1
55-53	#JMPCNTL(2-0)	JUMP FUNCTION SELECTION	4
52	#O> PCH	GATES O BUS TO PCH REGISTER	3.6
51	#MD> T	GATES MD BUS TO T REGISTER	3.6
50	#-MD> IR	GATES MD BUS TO IR REGISTER	3.7
49-48	#LOWWRITE(1-0)	SELECTS ONE OF 3 O BUS DESTINATIONS	3.6
47	#-O> ST	GATES O BUS TO ST REGISTER	3.6
46	#MD> P	GATES MD BUS TO P BUS	3.2
45	#PCH> P	GATES PCH REGISTER TO P BUS	3.2
44	#PCL> P	GATES PCL REGISTER TO P BUS	3.2
43	#MD> N	GATES MD BUS TO N BUS	3.3
42	#T> N	GATES T REGISTER TO N BUS	3.3
41	#ST> N	GATES ST REGISTER TO N BUS	3.3
40	#BCD> N	GATES BCD CONSTANT TO N BUS	3.3
39	#IR> N	GATES IR REGISTER TO N BUS	3.3
38	#ONE> AL	GATES CONSTANT ONE TO AL BUS	3.4
37	#PAL	GATES P BUS TO AL BUS	3.4
36	#MAL> AL	GATES MAL REGISTER TO AL BUS	3.4
35	#SP> AL	GATES SP REGISTER TO AL BUS	3.4
34	#T> AH	GATES T REGISTER TO AH BUS	3.5
33	#PCH> AH	GATES PCH REGISTER TO AH BUS	3.5
32	#ONE> AH	GATES CONSTANT ONE TO AH BUS	3.5
31	#MEMCNT	FIRST ONE OF TWO CYCLE MEM. ACCESS	2.4
30	#MEM	INDICATES A MEMORY ACCESS	2.4
29	#WR	INDICATES A MEMORY WRITE	2.4
28	#-LST	UPDATES STATUS REGISTER BITS	3.11
27-24	#SHIFTCNTL(3-0)	SELECTS SHIFT/ALU CARRY FUNCTIONS	3.9
23-20	#ALUCNTL(3-0)	SELECTS ALU FUNCTION	3.8
19	#ABL	LOGICAL (VS. ARITHMETIC) ALU OP'S	3.8
18-0	Reserved		

NOTE: In multiple bit fields bit 0 is the LSB.

All 160 words of the CROM are required to implement the standard instruction set of the TMS7000. Because of this, adding other microcoded functions to the TMS7000 requires that some of the standard instructions be deleted to allow space for the new instructions.

The TMS7000 Standard Instruction Set has been divided into two instruction groups designated core and non-core instructions. Non-core instructions are those instructions which Texas Instruments will allow to be removed in order to implement other microcoded functions. Core instructions may not be removed and are provided with any TMS7000 whether further microcoding has been implemented or not. Core and Non-core instructions are described in the TMS7000 Microcode Development Guide, (Part Number MP 458).

A symbolic microprogram assembler, MICASM, is available to aid microprogram generation. MICASM accepts mnemonic names for bit fields in a microinstruction word, and builds the appropriate bit pattern. The names of each bit field in the TMS7000 microinstruction word are given in Figure 2-1. They are distinguished from other signal names by preceeding them with a '#'.

For single bit fields, if the MICASM statement contains the name of the bit, it is asserted in the assembled instruction. For high-true signals, the bit is set to 1; for low-true signals (such as #-O> ST), the bit is set to 0. For multiple-bit fields, MICASM accepts any one of a set of possible names, where each name corresponds to a bit pattern for the multi-bit field. A sample of a MICASM statement is shown in Figure 2-1.

```
.ORG ADD0                          'ADD Dual Operand Function
     Z> AH,                        'AH = 0 for Page 0 access
     MAL> AL,                      'AL = destination register #
     MD> P,                        'Source operand to P bus
     T> N,                         'Destination operand to N bus
     PADDN, ZCI, LST,              'Add them, load status register
     MW,                           'Write the result to destination
     JUNC(NEXT);                   'Jump to next microinstruction
```

FIGURE 2-1 — SAMPLE OF A MICASM STATEMENT

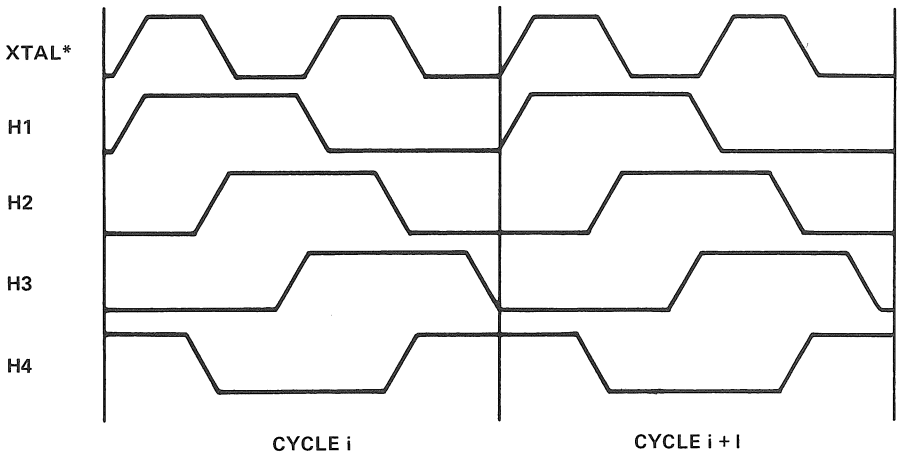
The .ORG line establishes the address of the microinstruction in the Control ROM. The remaining lines contain symbols which set bits in the current microinstruction word. The last line indicates the next microinstruction that is to be executed.

2.2 MICROINSTRUCTION CYCLE TIMING

Each microinstruction cycle has four overlapping clock phases; H1, H2, H3, and H4. H1 and H3 are non-overlapping, and H2 and H4 are non-overlapping. Microinstruction cycles begin on the rising edge of H1. Two versions of clock generator circuitry are available for the TMS7000. The first version uses the external crystal frequency directly to generate H1-H4. The second version divides the crystal frequency by two before generating the internal clock phases. Figure 2-2 shows the timing relationships of the four internal clock phases H1-H4 and the signal from the crystal oscillator.

H1-H4, the four internal clock phases, are used as data transfer signals throughout the architecture. In particular, the current microinstruction is gated out of the Control ROM during H1. Microinstruction bits required during later phases (H2, H3, H4) are appropriately sampled by the hardware.

The internal implementation of the TMS7000 uses MOS dynamic ratioless logic which allows the chip to operate with lower power requirements than with other types of MOS logic. Signal lines considered to be valid during phase HX (e.g. H1) are precharged during the non-overlapping phase of HX (e.g. H3). For this reason, timing diagrams in this document will indicate signal values only during the phase in which they are valid, with a don't care indication during the phase in which they are precharged.



***NOTE: This waveform represents the crystal oscillator output divided by two if that version of the clock generator circuitry is used.**

FIGURE 2-2 — MICROINSTRUCTION CYCLE PHASES

2.3 MEMORY CYCLE TIMING

Memory references to the on-chip Register File (RF) require one microinstruction cycle, and are called short memory cycles. All other references, i.e. to on-chip ROM, extended memory, or the Peripheral File, require two microinstruction cycles, and are called long memory cycles. Extended memory must be able to respond in this time period, since no wait states are provided in the TMS7000.

2.3.1 Short Memory References.

The timing for a read or write to the on-chip Register File is shown in Figure 2-3.

For a Register File read during cycle i, the microinstruction loaded at the initiation of cycle i asserts #MEM high and #MEMCNT low. #MEM is asserted at all times when a memory reference is active, and #MEMCNT is asserted high only during the first cycle of two-cycle (ie. long) memory cycles. #WR is set low for read operations and high for write operations. Microinstruction i also specifies the contents of the the address bus, placing a >00 on the AH (Address High) Bus and the register number on the AL (Address Low) Bus. During H2, the MD Bus is precharged and the RAM is accessed. For the duration of H4, the RAM output data on write operations and the RAM input data on read operations is on the MD Bus.

Because H4 of cycle i overlaps H1 of cycle i + 1, the data read on cycle i may be loaded into registers T or IR at the end of cycle i or gated onto the P or N Buses at the beginning of cycle i + 1. This characteristic of the MD Bus can be very useful in optimizing microcode performance.

Initial members of the TMS7000 Family implement only 128 bytes of the 256-byte Register File; attempts to write to addresses in non-existent on-chip memory will be ignored. Attempts to read non-existent memory will produce >00.

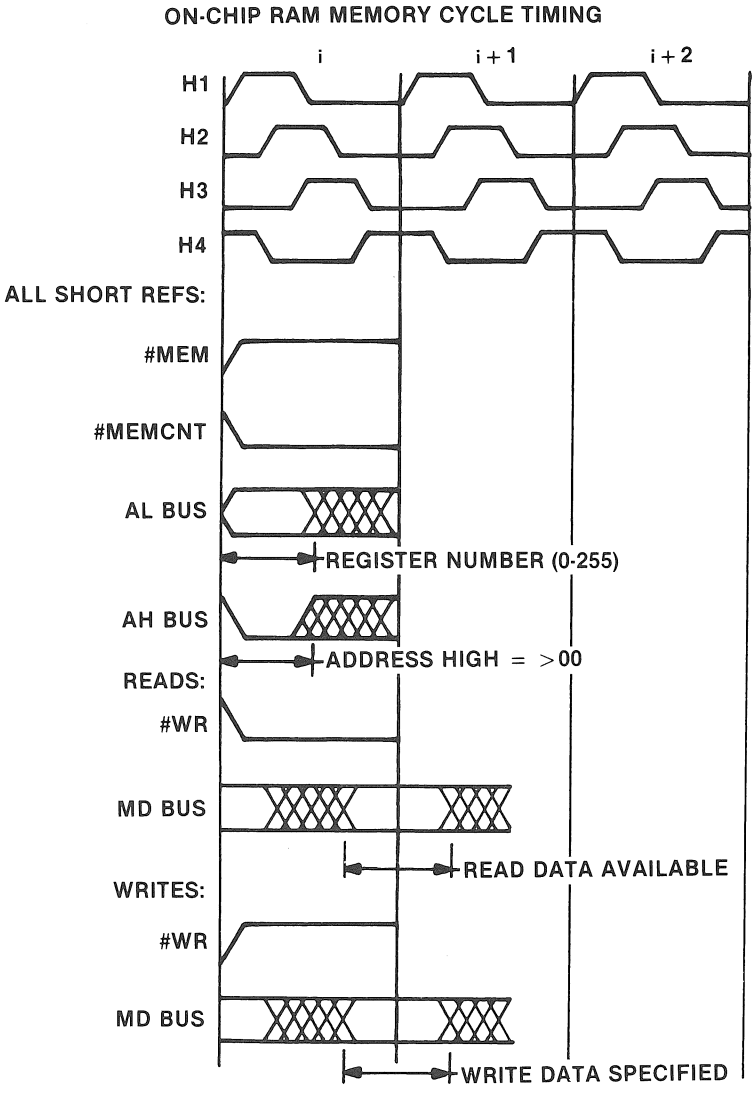


FIGURE 2-3 — ON-CHIP RAM MEMORY CYCLE TIMING

2.3.2 Long Memory References.

The timing for all long memory references is shown in Figure 2-4.

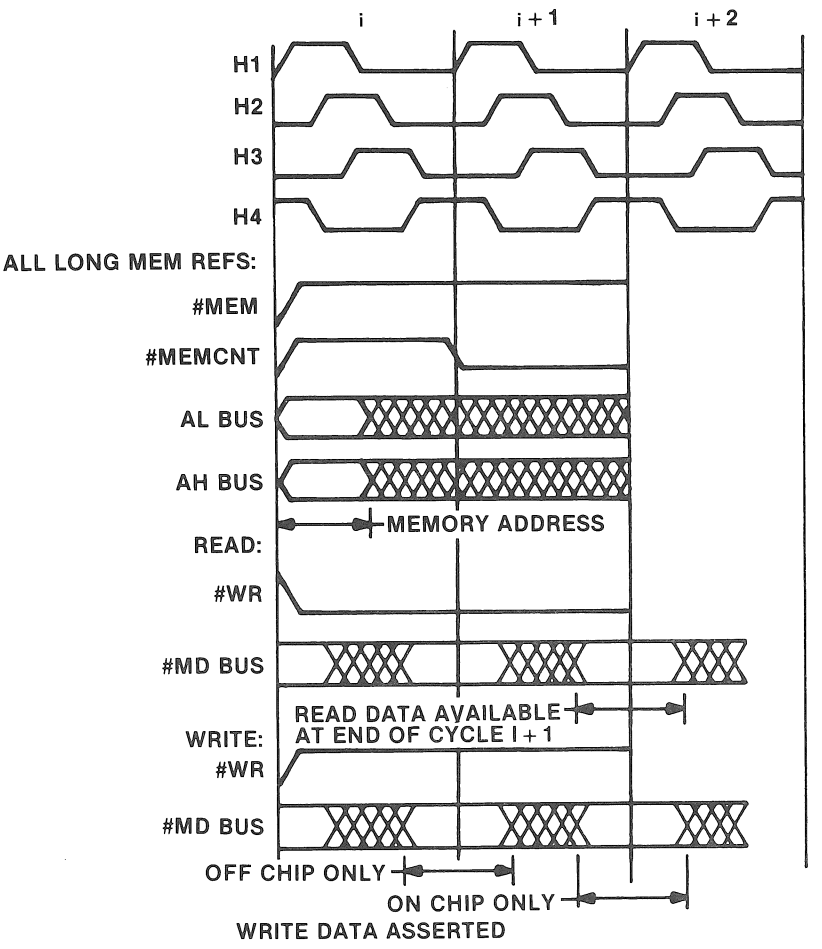


FIGURE 2-4 — LONG MEMORY CYCLE TIMING

The memory control signals #MEMCNT, #MEM, and #WR are specified in the microinstruction directly. Figure 2-4 shows these signals valid during a full microinstruction cycle because, once specified for a cycle, their state will not change during that cycle.

For all long memory references, #MEM must be asserted high for two consecutive cycles. #MEMCNT should be 1 for the first cycle, and 0 for the second cycle. #MEMCNT is asserted by specifying the MCNT symbol in the MICASM statements for the microinstruction. Various combinations of the #MEM and #WR microinstruction bits are specified by other MICASM symbols, as explained in paragraph 2.4. The 16-bit address to be accessed must be gated onto the AH and AL Buses during the first cycle. The Peripheral/Memory Controller latches the memory address, so the address need not be asserted during the second cycle. It should be noted that this feature can be used to great advantage in microcode sequences since this allows the AH and AL Buses to be used for other functions during the second microinstruction cycle. In this manner, microcode functions may be overlapped which can result in shorter, faster executing microcode.

For read cycles, #WR is set to 0 for both cycles. The result of a read appears on the MD Bus in phase H4 of the second cycle. It may either be loaded into the T or IR Registers at the end of the second cycle or loaded into the P or N Bus at the beginning of the third cycle.

For write cycles, #WR is set to 1 for both cycles. When the writes destination is an on-chip address, the write data must be valid during H4 of the second microinstruction cycle; when the writes destination is an off-chip address, the write data is required to be valid during H4 of the first microinstruction cycle. The data used in an off-chip write is latched by the PMC during the first cycle, and therefore need not be valid during the second cycle, and conversely the data in an on-chip write need not be valid during the first cycle. This can be used advantageously in certain microcoding situations. If desired, however, data may be asserted during both cycles.

2.3.3 Interrupt Vector Reads.

When an interrupt is received by the Peripheral/Memory Controller, the PMC asserts IACT on the Control Bus to the CPU, provided that STINT is a 1. The state of IACT may be tested by the CPU using an INT dispatch (see paragraph 4.2.5). If an interrupt is active, the CPU may then read an interrupt vector supplied by the PMC on the MD Bus, indicating which interrupt has occurred. The interrupt vector read requires two cycles, as shown in the timing diagram in Figure 2-5.

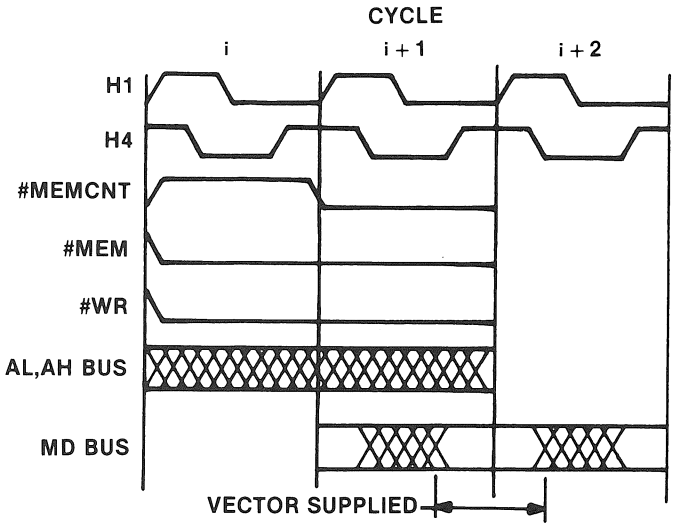


FIGURE 2-5 — INTERRUPT VECTOR READS

Notice that #MEM and #WR must be low for both cycles of the interrupt vector read. As with a long memory read, the vector is not available until the end of the second microinstruction cycle. An interrupt vector read may be coded in MICASM using the statements described in Table 2-2.

The value of the vector supplied by the PMC for each interrupt is shown in Figure 2-6. There is a distinction between the interrupt vector supplied by the PMC and the trap vector address at which the interrupt subroutine entry point address is stored.

Interrupt Level	Vector Supplied	Trap Vector Address
0 (Reset)	—	>FFFE
1	>FE	>FFFC
2	>FD	>FFFA
3	>FC	>FFF8

FIGURE 2-6 — INTERRUPT VECTOR REFERENCES

The vector supplied by the PMC is the same as the TRAPn opcode for the TMS7000 Standard Instruction Set. In order to call the interrupt handler, the microcode generates the trap vector address from the vector supplied, and reads memory at that location to get the address of the interrupt handler subroutine. It should be noted that the interrupt trap vector addresses shown in Figure 2-6 are those implemented in the currently supplied TMS7000 Standard Instruction Set Microcode. Different trap vector addresses may be implemented if additional microcode is written to handle modified interrupt servicing.

2.4 MEMORY CONTROL SIGNALS

The three memory control signals output by the CPU and interpreted by the Peripheral/Memory Controller are:

- #MEMCNT (Memory Continue) — asserted on the first cycle of a two-cycle long memory reference.
- #MEM (Memory) — asserted if the microinstruction references memory of any kind (RAM, ROM, extended, peripheral).
- #WR (Write) — 1 if a write is being performed, 0 if a read.

The interpretation of various combinations of these signals is described in Table 2-2.

The MICASM symbol or symbols listed in Table 2-2 must be used to specify the appropriate combination of memory control signals. The #MEMCNT Microinstruction Bit is set independently by the MICASM symbol MCNT. The various combinations of #MEM and #WR Microinstruction Bits are set by specifying the MICASM symbols O> MD, MR, and MW. O> MD may be specified when no memory access is desired, but the ALU Output (O) Bus contents are to be gated onto the Memory Data (MD) Bus. OTMD, the signal which enables the O Bus to drive the MD Bus, is generated by the PMC and is defined as OTMD = #WR .OR. #MEMCNT. The O> MD MICASM statement has been defined only to assert OTMD during non-memory cycles by generating a unique combination of #WR and #MEMCNT which does not occur during actual memory cycles. (O> MD should not be coded during memory accesses). Note, however that the combination of memory controls produced by O> MD is also the default and will be produced by MICASM if no memory controls are coded in a particular microinstruction cycle.

MR is specified for a memory read operation, and MW for a memory write. For long memory cycles, which require two microinstructions, MCNT is specified in the first microinstruction only. MR or MW must be specified in both microinstructions.

TABLE 2-2 — MEMORY CONTROLS

#MEMCNT (previous)	#MEMCNT (current)	#MEM	#WR	Function	OTMD	MICASM Symbol
0	0	0	0	- No Mem Reference -	0	-See Note 1-
0	0	0	1	Gate O Bus to MD Bus	1	O> MD-See Note 2
0	0	1	0	Short Memory Read	0	MR
0	0	1	1	Short Memory Write	1	MW
1	0	0	0	2nd State Int. Vector	0	INTVEC
1	0	0	1	* Illegal *	1	—
1	0	1	0	2nd State Long Read	0	MR
1	0	1	1	2nd State Long Write	1	MW
0	1	0	0	1st State Int. Vector	1	MCNT, INTVEC
0	1	0	1	* Illegal *	1	—
0	1	1	0	1st State Long Read	1	MCNT, MR
0	1	1	1	1st State Long Write	1	MCNT, MW
1	1	x	x	* Illegal *	1	—

NOTES: 1. MICASM is not capable of generating this combination of memory controls directly.

2. This combination of memory control signals is also the default combination, produced by MICASM when no memory control is specified.

SECTION 3

TMS7000 CPU INTERNAL ORGANIZATION

3.1 ORGANIZATION OF THE TMS7000 CPU

Section 3 describes the internal organization of the TMS7000 CPU. A block diagram is shown in Figure 3-1. Each of the internal registers and buses are 8 bits wide. The internal CPU Buses are used to transfer information between registers and to devices external to the CPU. Normally a bus will be used to transfer data between two particular locations during a microinstruction cycle. (Buses are precharged at various times during each microinstruction and therefore cannot be used to store data). These types of transfers of information are explained in the following descriptions of the various buses and registers within the CPU. In most cases, a bus will usually have only one source or destination; however, it may be desirable to have either multiple sources or destinations for a bus.

The case of multiple destinations of a bus is a simple extension of a single bus destination; a bus's contents are merely gated to several places simultaneously. This can be accomplished by simply including the MICASM statements for each destination, ie., MD> N and MD> P both coded in the same microinstruction cycle.

Multiple sources for a bus is more complex. Logically this may be coded in MICASM in a straightforward manner, just as multiple bus destinations are, however the result is quite different. The contents of a bus when multiple sources are specified is the logical OR of the two sources. This may be used advantageously in saving microcode in some situations with a single restriction: the TMS7000 Emulator cannot be used to debug the microcode. It should be emphasized that this technique should be used only when absolutely necessary and the Emulator may not be used to check the microcode, which can make a design very difficult to debug.

3.2 P BUS

The P Bus is one of the inputs to the Arithmetic Logic Unit, or ALU. It is called P for positive because it always contains the positive or left-hand operand; in subtract operations, the ALU always computes P-N and in add operations, P + N is computed. The P Bus is loaded from the MD Bus, the AL Bus, the PCH Register, the PCL Register or with the constant > 00 or > 01. Any of the AL Bus sources may be placed on the P Bus by gating them onto the AL Bus and asserting the #PAL microinstruction bit, connecting the P Bus to the AL Bus. A P Bus source must be coded in each microinstruction cycle. All of the possible P Bus sources are shown in Figure 3-2.

The hex representation in Figure 3-2 indicates the bits in a microinstruction that are affected when the MICASM symbol shown is specified for the P Bus source. Note that if a microinstruction requires no source on the P Bus, the MICASM symbol DC> P must be specified to indicate a don't care condition on the bus.

The P Bus is loaded at the beginning of a microinstruction cycle, on phase H1.

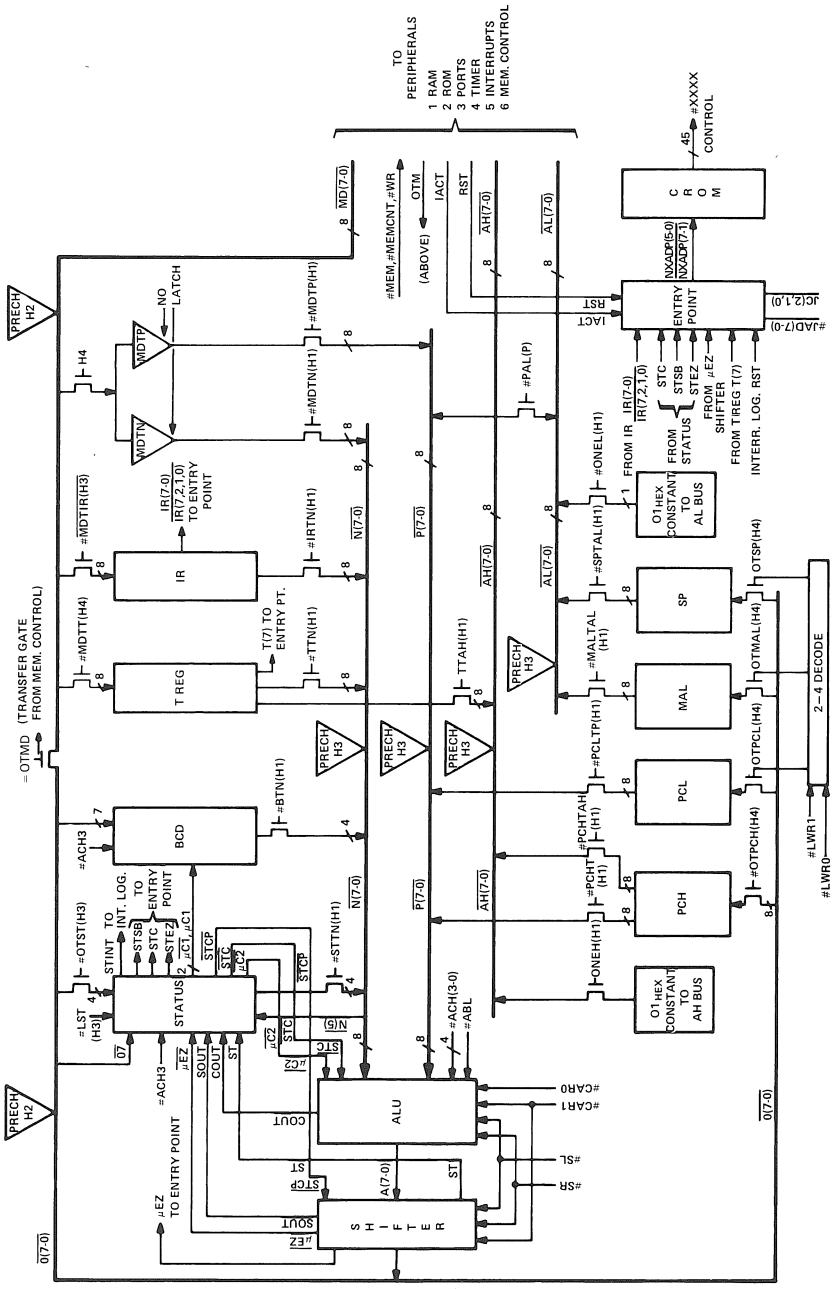


FIGURE 3-1 — CENTRAL PROCESSING UNIT DATA PATHS

P Bus Source	MICASM Symbol(s)	Hex Representation
MD Bus	MD> P	0000 4000 0000 0000
PCH Register	PCH> P	0000 2000 0000 0000
PCL Register	PCL> P	0000 1000 0000 0000
MAL Register	MAL> AL, AL> P	0000 0030 0000 0000
SP Register	SP> AL, AL> P	0000 0028 0000 0000
>01 constant	ONE> AL, AL> P	0000 0060 0000 0000
>00 constant	Z> P or DC> P	0000 0000 0000 0000

FIGURE 3-2 — P BUS SOURCES

3.3 N BUS

The N Bus is the second input to the ALU. It is called N for negative because in an ALU subtract operation, the N Bus contains the negative or right-hand operand. The N Bus is loaded from the MD Bus, the T Register, the IR Register, the Status Register, the BCD Constant Register or the constant >00. The source of the N Bus is indicated directly by a bit in the microinstruction word. If the bit is 1, the source is gated onto the N Bus. An N Bus source must be coded in each microinstruction cycle. All the possible N Bus sources are shown in Figure 3-3.

N Bus Source	MICASM Symbol(s)	Hex Representation
MD Bus	MD> N	0000 0800 0000 0000
T Register	T> N	0000 0400 0000 0000
Status Register	ST> N	0000 0200 0000 0000
BCD Constant	BCD> N	0000 0100 0000 0000
IR Register	IR> N	0000 0080 0000 0000
>00 constant	Z> N or DC> N	0000 0000 0000 0000

FIGURE 3-3 — N BUS SOURCES

If a microinstruction does not require an operand on the N Bus, the MICASM symbol DC> N must be specified to indicate a don't care condition on the bus.

The N Bus is loaded at the beginning of a microinstruction cycle, on phase H1.

3.4 AL BUS

The AL (Address Low) Bus holds the the lower 8 bits of all memory addresses. This includes references to the Register File, Peripheral File, on-chip, and extended memory. The AL Bus is loaded during phase H1, at the beginning of a microinstruction cycle. The sources of the AL Bus are the MAL Register, the SP Register, or the constant >00 or >01. The constant >01 is provided to efficiently address RAM location >01, (the B register of the standard TMS7000). This also facilitates addressing registers 16 and 17 (>10 and >11). An AL Bus source must be specified in each microinstruction cycle.

The AL Bus may also be connected to the P Bus by asserting the #PAL microinstruction bit, which can be accomplished by coding the P> AL MICASM instruction. In this manner, the AL Bus sources (MAL, SP, or the constant >00 or >01) may be gated onto the AL Bus and then onto the P Bus to be operated on by the ALU. Likewise, the P Bus sources (PCH, PCL, and MD Bus contents) may be gated onto the P Bus and then onto the AL Bus to serve as low order address lines. The MD Bus contents transferred are those present at the start of the microinstruction, i.e., those output by the previously executed microinstruction. All of the possible sources of the AL Bus are listed in Figure 3-4.

AL Bus Source	MICASM Symbol(s)	Hex Representation
MAL Register	MAL> AL	0000 0010 0000 0000
SP Register	SP> AL	0000 0008 0000 0000
PCL Register	PCL> P, P> AL	0000 1020 0000 0000
PCH Register	PCH> P, P> AL	0000 2020 0000 0000
MD Bus	MD> P, P> AL	0000 4020 0000 0000
>01 Constant	ONE> AL	0000 0040 0000 0000
>00 Constant	Z> AL or DC> AL	0000 0000 0000 0000

FIGURE 3-4 — AL BUS SOURCES

If no AL Bus source is required, the MICASM symbol DC> AL must be specified to indicate a don't care condition on the bus.

3.5 AH BUS

The 8-bit AH (Address High) Bus contains the high-order byte of the address referenced by the CPU. It is loaded during H1, at the beginning of a microinstruction cycle. It may be loaded with the contents of the PCH Register, the T Register, or the constant >00 or >01. The high byte of the program counter is intended to be stored in PCH; the T Register is intended to hold the high byte of other addresses in memory. The constant >01 is provided to efficiently access addresses in the Peripheral File (i.e., addresses of the form >01XX). An AH Bus source must be coded in each microinstruction cycle. The sources of the AH Bus are summarized in Figure 3-5.

AH Bus Source	MICASM Symbol	Hex Representation
PCH Register	PCH> AH	0000 0002 0000 0000
T Register	T> AH	0000 0004 0000 0000
>01 Constant	ONE> AH	0000 0001 0000 0000
>00 Constant	Z> AH or DC> AH	0000 0000 0000 0000

FIGURE 3-5 — AH BUS SOURCES

If no AH Bus source is required, the MICASM symbol DC> AH must be specified to indicate a don't care condition on the bus.

3.6 O BUS

The O (Output) Bus always contains the output of the ALU-Shifter combination. Its contents may be loaded onto the MD Bus, or into the Status, PCH, PCL, MAL, or SP Registers. The Status Register is loaded by the low-true microinstruction bit #-O> ST. The PCH Register is loaded by the high-true microinstruction bit #O> PCH. The load signals for the other destination registers (MAL, PCL, and SP) are encoded in the microinstruction bits #LOWWRITE(1-0), according to Figure 3-6. Note that since these bits are encoded, these three O Bus destinations are mutually exclusive; that is, only one of these destinations may be specified in a given microinstruction cycle.

#LOWWRITE 1	#LOWWRITE 0	O Bus Destination	MICASM Symbol
0	0	— none —	—
0	1	MAL Register	O> MAL
1	0	PCL	O> PCL
1	1	SP	O> SP

FIGURE 3-6 — LOWWRITE (1-0) DESCRIPTION

There is no microinstruction bit that directly loads the MD Bus from the O Bus, because the MD Bus contents are under control of the Peripheral/Memory Controller (PMC). This transfer is controlled by the OTMD signal sent from the PMC to the CPU on the C Bus. OTMD is asserted on every memory write cycle, (on-chip or extended memory), and on the first state of every long memory cycle. This is diagrammed in Table 2-2.

The O Bus is normally gated onto the MD Bus unless otherwise required in a memory cycle. Optionally, the O> MD symbol may be coded in a MICASM statement. MICASM sets up the appropriate values of the #MEM and #WR microinstruction bits so that OTMD will be asserted by the Peripheral/Memory Controller. The O Bus contents may then be loaded into the T or IR Registers from the MD Bus. Refer to paragraph 2.4 for a description of OTMD.

To write the O Bus contents to memory, the memory control signals outlined in paragraph 2.4 must be specified. The destinations of the O Bus are described in Figure 3-7.

O Bus Destination	MICASM Symbol	Microinstruction Field Hex Representation	
ST Register	O> ST	0000 0000 0000 0000	(Low True)
PCH Register	O> PCH	0010 8000 0000 0000	
PCL Register	O> PCL	0002 8000 0000 0000	}
MAL Register	O> MAL	0001 8000 0000 0000	} Only One
SP Register	O> SP	0003 8000 0000 0000	} Of Three
T Register	*[O> MD], MD> T	0008 8000 2000 0000	
IR Register	*[O> MD], MD> IR	0004 8000 2000 0000	
Short Mem Cycle	MW	0000 8000 6000 0000	
Long Mem, Cycle 1	MCNT, MW	0000 8000 E000 0000	
Long Mem, Cycle 2	MW	0000 8000 6000 0000	

* Specifying O> MD here is optional — see paragraph 2.4

FIGURE 3-7 — O BUS DESTINATIONS

The O Bus is loaded during phase H4 of the microinstruction cycle. It contains the result of the ALU and Shifter operations specified in the current microinstruction.

3.7 MD BUS

The MD (Memory Data) Bus is a bidirectional bus that transfers data to and from the CPU. Data is valid on MD during phase H4 of a microinstruction cycle, which spans two microinstructions. Thus, data may be read from the MD Bus onto the P or N Bus at the beginning of a cycle (H1), and the ALU results then loaded back onto the MD Bus at the end of the cycle (H4). It is important to note that when using data from the MD Bus during H1 of a particular microinstruction cycle, the actual data available will be the contents loaded onto the MD Bus during the end of the previous cycle.

At the end of a cycle, the MD Bus may be loaded in one of three ways:

1. The O Bus contents may be gated onto the bus.
2. The on-chip RAM or ROM may place data onto the bus.
3. The Peripheral/Memory Controller may place data onto the bus.

The MD Bus contents are controlled by the Peripheral/Memory Controller (PMC). The PMC sends the OTMD signal to the CPU to signal loading the MD Bus from the O Bus. The CPU requests use of the MD Bus by asserting combinations of the #MEM, #MEMCNT, and #WR signals, as outlined in Table 2-2. The PMC sends signals to the the on-chip ROM and RAM to control their accesses to the bus.

The timing of read and write accesses to memory is explained in paragraph 2.3. For short memory reads, data is available at the end of the microinstruction that initiated the read. This data may be loaded into the T or IR Registers during that microinstruction by specifying the MD> T or MD> IR MICASM symbols, respectively. The data may be loaded into the P or N Bus on the next microinstruction by specifying the MD> P or MD> N symbols in the MICASM statement for the next microinstruction. For short memory cycle writes, the O Bus data is placed on the MD Bus, and the MW MICASM symbol specified. For long memory reads, the desired address is placed on the AH and AL lines, and the MR and MCNT symbols specified in the first of the two cycles required. At the end of the second cycle, data is available on MD. (The memory address is latched by the PMC on the first cycle, and need not be asserted on the second cycle.) For long memory writes, the address is specified in the first cycle, and the data is placed on the MD Bus for the first and/or second cycles. Refer to Section 2 for a detailed description of short and long memory cycles.

The destinations of the MD Bus in the CPU are described in Figure 3-8.

MD Bus Destination	When Loaded	MICASM Symbol
T Register	End of Cycle	MD> T
IR Register	End of Cycle	MD> IR
P Bus	Start of Cycle	MD> P
N Bus	Start of Cycle	MD> N

FIGURE 3-8 — MD BUS DESTINATIONS

3.8 ALU OPERATION

The Arithmetic Logic Unit (ALU) accepts as inputs the values on the P and N Buses, and outputs its result to the Shifter. The ALU operation is controlled by the #ALUCNTL(3-0) and #ABL lines from the current microinstruction. The ALU operates on the values loaded on the P and N Buses during H1 of the current microinstruction and produces an 8-bit output which is input to the Shifter, and a carry bit (COUT), which is an arithmetic carry bit based on the 8-bit ALU operation. To specify the carry-in, the ALU accepts the #SHIFTCNTL(3-0) bits from the current microinstruction. An overall block diagram of the ALU appears in Figure 3-9.

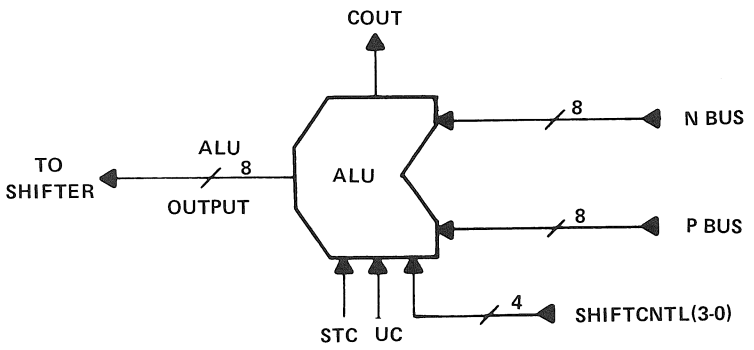


FIGURE 3-9 — ALU BLOCK DIAGRAM

The available operations of the ALU are defined in Figure 3-10.

#ALUCNTL (3-0)	#ABL	Hex Representation	MICASM Symbol	ALU Output
0000	0	0000 0000 0000 0000	PADDN	P + N + CI
0000	1	0000 0000 0008 0000	XNOR	P XNOR N
0001	1	0000 0000 0018 0000	AND	P AND N
0010	1	0000 0000 0028 0000	IPORN	(NOT P) OR N
0011	1	0000 0000 0038 0000	PASSN	N
0100	1	0000 0000 0048 0000	PORIN	P OR (NOT N)
0101	1	0000 0000 0058 0000	PASSP	P
0110	1	0000 0000 0068 0000	FF	>FF
0111	1	0000 0000 0078 0000	OR	P OR N
1000	1	0000 0000 0088 0000	NOR	P NOR N
1001	1	0000 0000 0098 0000	ZERO	>00
1010	1	0000 0000 00A8 0000	INVP	NOT P
1011	1	0000 0000 00B8 0000	IPANDN	(NOT P) AND N
1100	1	0000 0000 00C8 0000	INVN	NOT N
1101	1	0000 0000 00D8 0000	PANDIN	P AND (NOT N)
1110	1	0000 0000 00E8 0000	NAND	P NAND N
1111	0	0000 0000 00F0 0000	PSUBN	P − N − 1 + CI
1111	1	0000 0000 00F8 0000	PXORN	P XOR N

FIGURE 3-10 — ALU FUNCTIONS

The Carry-in Bit of the ALU (CI) is specified by the #SHIFTCNTL(3-0) bits of the microinstruction, which are described in full in the next section. For operations requiring no shifting of the ALU contents, the possible carry-in bits are outlined in Figure 3-11.

#SHIFTCNTL 3 2 1 0	MICASM Symbol	ALU Carry In (CI)
0 0 0 0	ZCI	0
0 0 0 1	ONECI	1
0 0 1 0	UCI	UC — Micro Carry Bit
0 0 1 1	STCI	STC — Status Carry Bit

FIGURE 3-11 — ALU CARRY IN VALUES

The Micro Carry Bit (UC) is the carry-out from the ALU operation of the immediately preceeding microinstruction. This is not the same as the Shift-out Bit (SOUT) from the Shifter operation of the previous microinstruction. The Status Carry Bit (STC) is the Carry bit of the Status Register.

The arithmetic Carry-out Bit from the ALU (COUT) is 1 if there is a carry-out during an add (PADDN) or subtract (PSUBN) operation in the ALU. For an add operation, COUT = 1 indicates there was a carry, i.e., the sum of the (unsigned) operands exceeds 255. For a subtract operation, COUT = 0 indicates there was a borrow, i.e., the P operand was lower than the N operand (unsigned). For all other operations, i.e., logical operations, COUT is set to 0. COUT is sent to the Status Register circuitry for possible loading into STC, the Status Carry Bit.

As an example of ALU operation, the following symbols appearing in a MICASM statement,

PADDN, ZCI

will cause the ALU to calculate the sum of the P and N Bus contents. To calculate the difference between the P and N Bus contents,

PSUBN,ONECI

must be specified. A 1 must be carried in since no borrow was desired. Figure 3-12 details two microcode examples. The following microinstructions read the current byte addressed by the PC, place it in the T Register, and increment the PC.

```
                                ' Read immediate byte, 1st cycle
.ORG IMMED1                     ' Define location of microinstruction
PCL> P,P> AL,                   ' Place PCL on AL Bus via P Bus
PCH> AH,                        ' Place PCH on AH Bus
Z> N,                           ' Place Zero on N Bus
PADDN,ONECI,                    ' Increment PCL by 1 (sets Micro Carry UC)
O> PCL,                         ' Place result back in PCL
MCNT,MR,                        ' 1st cycle of long read
JUNC(IMMED2);                   ' Goto next cycle

.ORG IMMED2                     ' Read immediate byte, 2nd cycle
DC> AH,DC> AL,                  ' Don't care what's on AH and AL
                                '     since address was latched on 1st cycle
PCH> P,                         ' Place PCH on P Bus
Z> N,                           ' Place Zero on N Bus
PADDN,UCI,                      ' Add micro carry from PCL increment
O> PCH,                         ' Place result back in PCH
MR,                             ' Meanwhile, continue memory read
MD> T,                          ' And place the byte read into T
JUNC(NEXT);                     ' Then goto next instruction
```

FIGURE 3-12 — A MICROCODE EXAMPLE

Notice that an increment was done in IMMED1 by using an ALU carry-in of 1. The second instruction (IMMED2) incremented the high byte of the PC only if the Micro Carry Bit (UC) generated by IMMED1 was 1.

3.9 SHIFTER OPERATION

The Shifter performs a variety of 1-bit shift operations on the output of the ALU. The #SHIFTCNTL(3-0) lines control the following ALU and Shifter characteristics:

- The ALU Carry-in Bit (CI)
- The shift direction (L or R)
- The bit shifted into the Shifter.

Figure 3-13 describes the various combinations of shift control lines.

#ShiftCntl 3 2 1 0	ALU CI	Shift Direction	Shift-In Bit	MICASM Symbol
0 0 0 0	0		—	ZCI
0 0 0 1	1	No	—	ONECI
0 0 1 0	UC	Shift	—	UCI
0 0 1 1	STC		—	STCI
0 1 0 0	1		ALU(7)	RLO
0 1 0 1	0	Shift	ALU(7)	RLZ
0 1 1 0	1	Left	STC	RLCO
0 1 1 1	0		STC	RLCZ
1 0 0 0	1		ALU(0)	RRO
1 0 0 1	0	Shift	ALU(0)	RRZ
1 0 1 0	1	Right	STC	RRCO
1 0 1 1	0		STC	RRCZ
1 1 X X	*	Invalid	*	*

FIGURE 3-13 — SHIFT/ALU CARRY-IN CONTROLS

For #SHIFTCNTL = 00XX, no shifting is performed, and the ALU Carry-in Bit CI is as described in the ALU description, above. For #SHIFTCNTL = 010X, the ALU output is rotated left, with the most significant bit, ALU(7), shifted in from the right. For #SHIFTCNTL = 011X, the ALU output is rotated left through the Status Carry Bit, STC. For #SHIFTCNTL = 100X, the ALU output is rotated right, and for #SHIFTCNTL = 101X, the output is rotated right through the carry bit. The MICASM symbols represent this, with the last character indicating the value of the ALU CI bit. #SHIFTCNTL = 11XX is an invalid command and must never be specified.

The Shift-out Bit (SOUT) shifted out in a rotate instruction is sent to the Status Register. It will be loaded as the new Status Carry Bit (STC) if the #-LST microinstruction bit is set. Operation of each of the shift instructions is diagrammed in Figure 3-14.

3.10 IR REGISTER

The Instruction Register (IR) is a register intended to hold the assembly language opcode. It is loaded from the MD Bus by specifying the MD> IR symbol in a MICASM statement. It may be loaded onto the N Bus with the IR> N MICASM symbol.

The TMS7000 Microarchitecture is designed to dispatch (branch) on various subfields of the IR contents, providing for the execution of appropriate microcode for each assembly language instruction. The IR may be considered to have two possible formats:

1. Format 0 is indicated by a 0 in IR(7), the most significant bit of the IR Register. In this format, bits IR(6-4) form a 3-bit Group field and bits IR(3-0) form a 4-bit Function field.
2. Format 1 is indicated by a 1 in IR(7). In this format, bits IR(6-3) form a 4-bit Group field and bits IR(2-0) form a 3-bit Function field.

The formats of the IR Register are diagrammed in Figure 3-15.

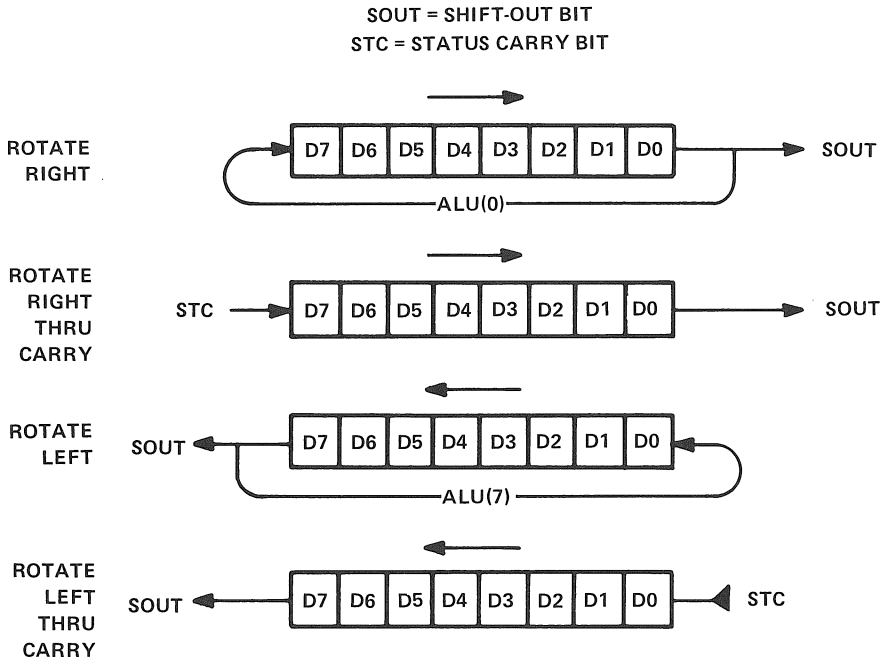


FIGURE 3-14 — SHIFTER OPERATION

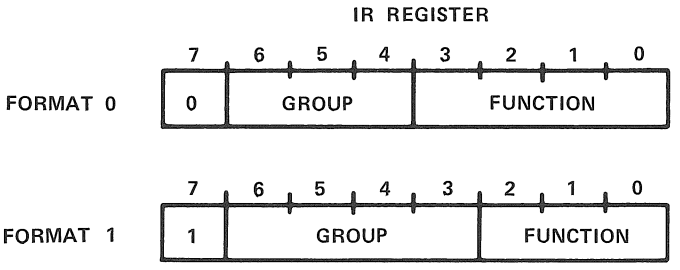


FIGURE 3-15 — IR REGISTER FORMATS

The terms group and function refer to logical subsets of assembly language opcodes. In the TMS7000 standard instruction set the Group field in an opcode indicates the addressing mode of the instruction, and the Function field indicates the arithmetic or logical operation performed on the operands. The microarchitecture is designed to allow significant sharing of microinstructions among opcodes within the same group or function. In the microcode for the standard TMS7000, for instance, all opcodes of the form >1X share microcode which fetches the A Register and a general RF Register.

The mechanisms for dispatching on the Group and Function field values in the IR are described in Section 4. Dispatching on an IR subfield may be performed on the first microinstruction after the IR is loaded. Thereafter, dispatching may be performed by microinstructions up to and including the next one that reloads the IR. If no dispatching is required, then the IR may be used as a general purpose 8-bit register.

3.11 STATUS REGISTER

The Status Register (ST) is an 8-bit register with contents indicating various conditions of the CPU. Each bit of the Status Register has a special meaning and separate circuitry devoted to it. The format of the ST Register is shown in Figure 3-16.

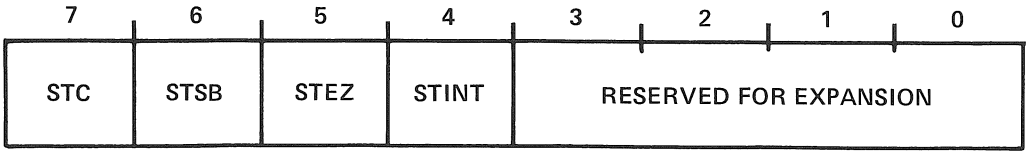


FIGURE 3-16 — STATUS REGISTER

STC is the Status Carry Bit. It holds either the carry-out of the ALU, the shift-out of the Shifter, or the decimal arithmetic carry-out. STSB is the Status Sign Bit. It contains the most significant bit of the O Bus contents. STEZ is the Status Equal to Zero Bit. It contains a 1 when all bits of the O Bus are zero. STINT is the Status Interrupt Enable Bit. Bits 3-0 of the Status Register are reserved for future expansion. These bits wil be zeros when the ST Register is loaded onto the N Bus.

The existing Status Register Bits may be modified in one of two ways:

1. All bits may be replaced by the contents of the O Bus.
2. The STC, STSB, and STEZ bits may be set according to their particular input circuitry. The STINT Bit is unaffected in this case.

The Status Register Sources are summarized in Figure 3-17.

ST Register Source	MICASM Symbol	Hex Representation
O Bus Input Circuitry	O> ST	0000 0000 1000 0000 (Asserted low)
	LST	0000 8000 0000 0000 (Asserted low)

FIGURE 3-17 — ST REGISTER SOURCES

The O Bus is gated into the Status Register if the #-O> ST Microinstruction Bit is asserted low. This may be specified by the O> ST symbol appearing in a MICASM statement. The STC, STSB, and STEZ Bits are loaded when the #-LST Microinstruction Bit is asserted low. This may be specified by the LST symbol appearing in the MICASM statement. There is no way to individually load the STC, STSB, and STEZ Bits; they must be loaded together. This feature permits an efficient implementation of the TMS7000 status logic, typically a very costly item in single-chip microarchitectures. The special circuitry defining the value of the STC, STSB, and STEZ Registers is described in the following paragraphs.

3.11.1 STC — Status Carry Bit.

When the #-LST signal is asserted by coding the LST MICASM instruction, the STC Bit will be loaded from one of three sources:

1. The ALU Arithmetic Carry-out Bit (COUT); This is the carry/borrow bit generated by the ALU on arithmetic operations. COUT is loaded if no Shifter operation is specified, i.e., #SHIFTCNTL = 00XX.
2. The Shifter Shift-out Bit (SOUT). This is the bit shifted out in Shifter operations. If a Shifter operation is specified — i.e., #SHIFTCNTL > 0011 — then SOUT is loaded into the STC Bit (whether a rotate thru carry was specified or not).
3. The BCD Decimal Carry/Borrow Out Bit (DCOUT). This is the carry bit computed by the decimal adjust hardware within the BCD Constant Register. It is loaded into the STC Status Carry Bit if the #BCD> N Bit is set, indicating a decimal adjust constant is loaded onto the N Bus.

3.11.2 STSB — Status Sign Bit.

When #-LST is asserted, the input to the STSB Bit is O(7), the most significant bit of the O Bus.

3.11.3 STEZ — Status Equal to Zero Bit.

When #-LST is asserted, the input to the STEZ Bit is the Micro Equal-to-Zero Bit, UEZ. The UEZ Bit is simply the logical NOR of all O Bus lines. That is, if all O Bus lines are 0, UEZ is set to 1. Otherwise, it is set to 0.

3.11.4 STINT — Status Interrupt Enable Bit.

The STINT Bit may only be modified by loading the O Bus contents into the Status Register. The STINT Bit corresponds to bit O(4) in this case. STINT is output from the CPU to the Peripheral/Memory Controller on the C (Control) Bus between the CPU and PMC. If STINT = 0, the PMC will not pass an interrupt to the CPU via the IACT line (also in the C Bus). If STINT = 1, the PMC will assert IACT on an interrupt. By dispatching on the IACT bit, the microcode is able to test for interrupts.

Due to propagation delays, the effect of loading STINT on IACT takes two microinstruction cycles to be asserted. Accordingly, if STINT is updated in cycle i, IACT will not be valid until cycle i + 2. Thus a JINT dispatch on IACT will not jump correctly if coded in state i + 1.

3.12 BCD CONSTANT REGISTER

The BCD Constant Register is a module which generates a correction constant for binary coded decimal arithmetic operations. Decimal numbers on the TMS7000 are represented with 2 binary coded decimal digits per byte, with the least significant digit in the least significant nibble, bits 3-0, of a byte. For example, the decimal number 78 would be represented in binary as '01111000', or >78. To perform decimal addition on two BCD Bytes X and Y, the following operations must be performed:

1. The binary sum of X and Y is computed, with the STC Bit carried in, and the result saved temporarily.
2. A decimal correction constant is computed by the BCD hardware.
3. The correction constant is added to the saved result to produce the final BCD sum.

Each of these operations requires a microinstruction cycle.

The STC Bit is added in order to permit adding multiprecision strings of BCD digits. Decimal subtraction (with borrow) is similar to the above procedure. The binary difference X-Y is first computed, and the correction constant then subtracted from the result.

Figure 3-18 indicates the decimal correction constant and decimal carry out bit generated for decimal addition and subtraction.

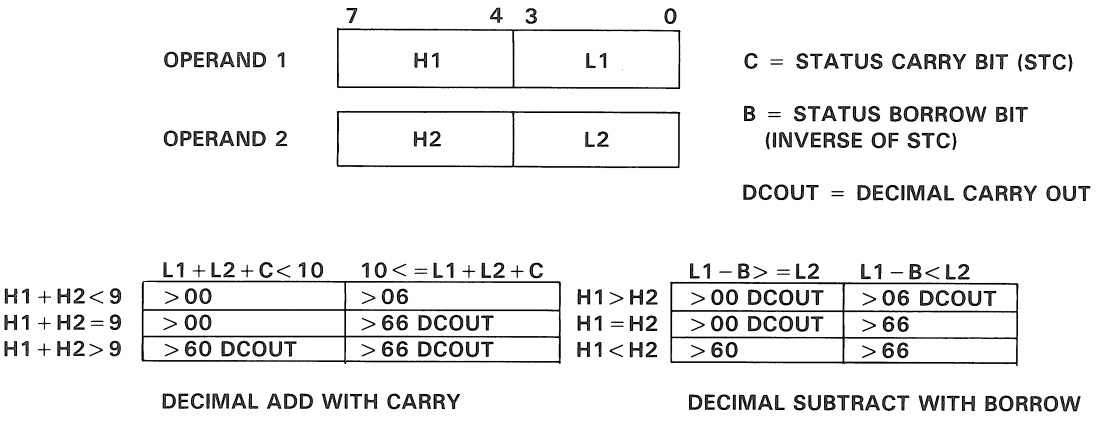


FIGURE 3-18 — BCD CORRECTION CONSTANT GENERATION

The BCD constant logic uses signals from the ALU such as the 8-bit carry (COUT), the ALU operation code #ALUCNTL(3-0), and ALU outputs on the O Bus to determine the correction constant and Decimal Carry-out Bit (DCOUT). Like the binary arithmetic carry, DCOUT is 1 if a carry is required after an addition, and 0 if a borrow is required after a subtraction. Figure 3-18 indicates the conditions in which DCOUT is 1. DCOUT is sent to the Status Register for possible loading into the STC Status Carry Bit.

Three microinstruction cycles are required to perform a decimal arithmetic operation. The timing for a decimal arithmetic operation is shown in Figure 3-19.

The first state loads the BCD operands onto the P and N Buses, and performs the appropriate ALU operation (PADDN or PSUBN) to produce the binary result. The binary result must be stored in a temporary location for use in the third state. The BCD operation diagrammed in Figure 3-19 assumes the result is stored in the RF. The second state reads this binary result from the Register File and leaves it on the MD Bus. This state allows the BCD constant hardware to determine the correction constant and Decimal Carry-out Bit, DCOUT. The third state loads the binary result onto the P Bus and the correction constant onto the N Bus and performs the appropriate ALU operation to produce the correct BCD result. The Status Register should be loaded in this state by coding an LST instruction in MICASM.

The MICASM statement shown in Figure 3-20 implement a decimal add with carry. A source operand is added to a destination operand, and the result stored in the destination operand (a register in the RF). The T Register is assumed to contain the source operand, the MD Bus contains the destination operand, and the MAL Register contains the register number of the destination operand.

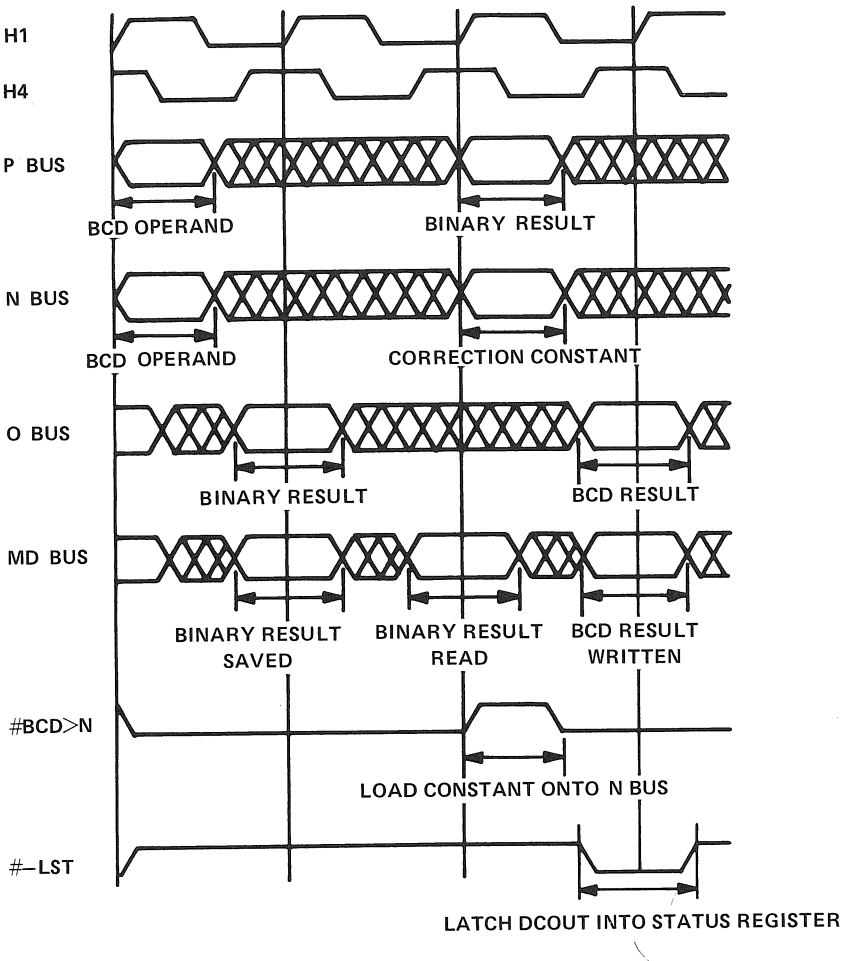


FIGURE 3-19 – BCD ARITHMETIC OPERATION TIMING

```
.ORG DAC0                    ' Decimal Add w/ Carry, first state
Z> AH,                       ' Place destination register address
MAL> AL,                     '     on address bus: AH = 0, AL = MAL
MD> P,                       ' Dest. operand to P Bus
T> N,                        ' Source operand to N Bus
PADDN,STCI,                  ' Add them, including carry from last DAC
MW,                          ' Store binary result in dest. register
JUNC(DAC1);                  ' Goto DAC1

.ORG DAC1                    ' DAC, second state
Z> AH,                       ' Read binary result back. Put dest. address
MAL> AL,                     '     on addr. bus: AH = 0, AL = MAL
DC> P, DC> N,                ' Don't cares to P and N Bus
PADDN,ZCI,                   ' Maintain ALU operation code (PADDN)
MR,                          ' Read binary result, placed on MD Bus
JUNC(DAC2);                  ' Goto DAC2

.ORG DAC2                    ' DAC, third state
Z> AH,                       ' Put destination address on Address Bus
MAL> AL,                     '     (AH = 0, AL = MAL)
MD> P,                       ' Put binary result on P Bus
BCD> N,                      ' Put BCD correction constant on N Bus
PADDN,ZCI,                   ' Add them (with no carry)
LST,                         ' Load Status register with decimal carry
MW,                          ' Store BCD result to destination register
JUNC(NEXT);                  ' Goto next microinstruction.
```

FIGURE 3-20 — MICASM STATEMENT

For a decimal subtract operation, the PADDN symbols should be replaced with PSUBN. State DAC2 should subtract the BCD constant via the MICASM symbols PSUBN,ONECI. A carry-in of 1 is needed since no borrow is required.

3.13 OTHER REGISTERS

The remaining registers implemented in the TMS7000 CPU include five storage registers and two constant registers. Two of the storage registers, the PCH and PCL, are used to hold the high and low bytes of the Program Counter. The Program Counter contents are normally essential to CPU operation, hence the PCH and PCL registers are almost never used as general purpose storage.

Two other storage registers, the Temporary or T Register and the MAL or Memory Address Low Byte Register may be paired to store the high and low bytes of a memory address, or used separately with the T Register serving as temporary storage and a memory address being generated from the MAL and a constant.

There are two constant registers used for generating the constant >01; one for each of the AH and AL Buses. Thus either of these buses may be loaded with either >00 or >01 if necessary. This capability is used for, among other things, generating RF and PF Addresses.

The SP or Stack Pointer is normally used to hold a pointer to the stack in RAM, but may be used as temporary data storage if a stack is not implemented or if the SP contents are not needed.

SECTION 4

MICROINSTRUCTION SEQUENCE CONTROL

4.1 OVERVIEW

This section describes the mechanisms used in controlling the sequence of microinstruction execution, which include generation of the next microinstruction address in both conditional and unconditional branching. Included in Section 4 is a description of dispatching capabilities which can be used to share microstates among several assembly language instructions.

Microinstructions are stored in the Control ROM, or CROM, on the TMS7000 chip. A characteristic of horizontally microprogrammed architectures like the TMS7000 is that each microinstruction indicates the address at which the next microinstruction to be executed is located. In the TMS7000, the next microaddress is specified by two fields:

1. #JMPADDR(7-0), an 8-bit field indicating a base address in CROM.
2. #JMPCNTL(2-0), a 3-bit code indicating one of 8 dispatch offsets from the base address in #JMPADDR.

If #JMPCNTL(2-0) = 000, then the #JMPADDR field is simply the address of the next microinstruction. If #JMPCNTL(2-0) is nonzero, it indicates what data will replace the low order bits of #JMPADDR, and thus form the next microaddress. This technique is called dispatching, and is extremely easy to implement in MOS technology.

All conditional branching in microcode is accomplished by means of dispatching. A base address is specified in the #JMPADDR(7-0) bits of the microinstruction. The #JMPCNTL(2-0) lines indicate what data is used to form the low order bits of the base address to generate the new microinstruction address. For example, Figure 4-1 depicts dispatching on the IR(3-0) Bits.

The dispatch field bits, like IR(3-0), actually replace the low order address bits in the #JMPADDR(7-0) field; they are not OR'ed with them. For example, suppose #JMPADDR was specified to be >11, and the #JMPCNTL(2-0) lines are set to 110, indicating a dispatch on STC, the Status Carry Bit. If STC were 0, the next microaddress would be >10.

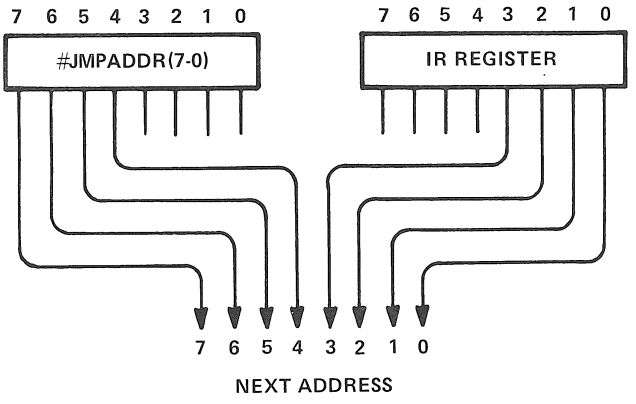


FIGURE 4-1 — MICROINSTRUCTION DISPATCH EXAMPLE

Figure 4-2 summarizes the possible dispatch fields and the MICASM code to indicate the next address.

#JMPCNTL 2 1 0	NEXT ADDRESS 7	6	5	4	3	2	1	0	MICASM Format
0 0 0	J7	J6	J5	J4	J3	J2	J1	J0	JUNC(addr)
0 0 1	J7	J6	J5	J4	IR3	IR2	IR1	IR0	IRL(baseaddr)
0 1 0	J7	J6	J5	J4	J3	J2	J1	T7	JT7 (oneaddr,zeroaddr)
0 1 1	J7	J6	J5	J4	J3	J2	J1	UEZ	JUZ(oneaddr,zeroaddr)
1 0 0	J7	J6	J5	J4	J3	J2	0	IACT	INT(oneaddr,zeroaddr)
1 0 1	J7	J6	J5	IR7	IR6	IR5	IR4	(1)	IRH(baseaddr)
1 1 0	J7	J6	J5	J4	J3	J2	J1	STC	JC(oneaddr,zeroaddr)
1 1 1	J7	J6	J5	J4	J3	J2	J1	MJMP	MJMP(oneaddr,zeroaddr)

(1) IR3 .or. (.not. IR7)

Jn	—	#JMPADDR(n)
IRn	—	IR Register bit n
T7	—	T register sign bit (bit 7)
UEZ	—	1 if 0 bus = >00, 0 otherwise
IACT	—	Interrupt Active line from PMC
STC	—	Status Carry Bit
MJMP	—	Macro jump: test Status Register bits
baseaddr	—	Base micro-address for dispatch
oneaddr	—	Next micro-address if bit 0 is 1
zeroaddr	—	Next micro-address if bit 0 is 0

FIGURE 4-2 — NEXT MICRO ADDRESS GENERATION

4.2 DISPATCH CONDITIONS

Each of the dispatch possibilities is further explained in the following sections.

4.2.1 Unconditional Branching — JUNC.

If conditional branching of the microcode is not desired, #JMPCNTL should be set to 000. The symbol

JUNC(addr)

appearing in a MICASM statement will cause the TMS7000 to branch unconditionally to the microinstruction at address addr after the current microinstruction is executed. The addr field may be a constant or, more practically, a symbol equated to the desired address of the microinstruction. The address addr is loaded into the #JMPADDR(7-0) field of the current microinstruction.

4.2.2 Function Dispatch — IRL.

When #JMPCNTL = 001, the next microinstruction is determined by the low four bits of the the IR Register. This is specified in MICASM as:

IRL(baseaddr)

The baseaddr is loaded into the #JMPADDR(7-0) field of the microinstruction. The next micro address is determined by replacing the bits 3-0 of the base address with bits 3-0 of the IR Register. To avoid confusion, it is convenient to make the base address a multiple of 16 i.e., bits baseaddr(3-0) = 0, since they will be ignored. The IRL dispatch is indicated pictorially in Figure 4-3.

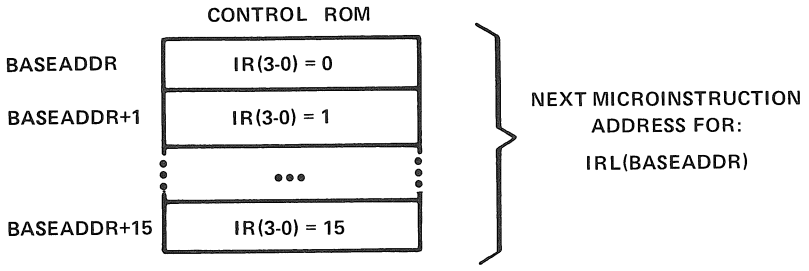


FIGURE 4-3 — IRL DISPATCH

An IRL dispatch is a dispatch on the Function field of the IR. In the TMS7000 Standard Instruction Set the Function field indicates the arithmetic operation to be performed. This is contrasted with the Group field, bits 7-4, which indicates the addressing mode of the instruction. Even though Format 1 instructions have a 3-bit Function field; IR(2-0), the IRL dispatch still performs a 16-way branch on the lower 4 bits of the IR Register. The Function dispatch for Format 1 opcodes thus depends on the value of the IR(3) Bit.

4.2.3 Test Sign Bit — JT7.

The sign bit of the contents of the T Register may be dispatched on by specifying #JMPCNTL = 010. This is indicated by

JT7(oneaddr,zeroaddr)

in a MICASM statement. The oneaddr field should be the 8-bit address of the microinstruction to be executed if T(7) is 1, and the zeroaddr field is the address of the microinstruction to be executed if T(7) is 0. This is indicated in Figure 4-4.

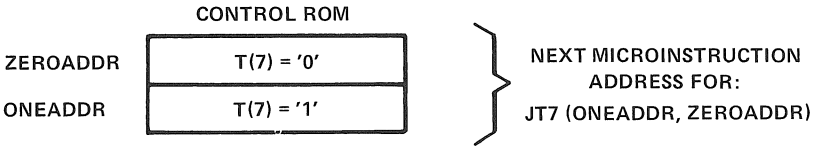


FIGURE 4-4 — JT7 DISPATCH

Typically, zeroaddr and oneaddr are MICASM labels initialized by an .EQU statement. It is required that zerroaddr be even and that oneaddr = zeroaddr + 1.

4.2.4 Test if Zero — JUZ.

The microcode may test the value on the O Bus of the immediately preceeding microinstruction by specifying #JMPCNTL = 011. This is indicated by

JUZ(oneaddr,zeroaddr)

appearing in a MICASM statement. When JUZ appears in microinstruction i, it tests the O Bus contents of the previously executed microinstruction, i-1. The entry-point logic replaces JMPADDR(0) with the UEZ Bit from the O Output Bus, which is 1 when the O Bus is all zeroes (>00) and 0 otherwise. The symbol oneaddr denotes the address to which control is transferred if the O Bus was zero, i.e., if UEZ = 1. The symbol zeroaddr denotes the address jumped to if the O Bus was nonzero, i.e., if UEZ = 0. Like the JT7 MICASM symbol, zeroaddr must be even and oneaddr must equal zeroaddr + 1. The dispatch on the UEZ Bit is depicted in Figure 4-5.

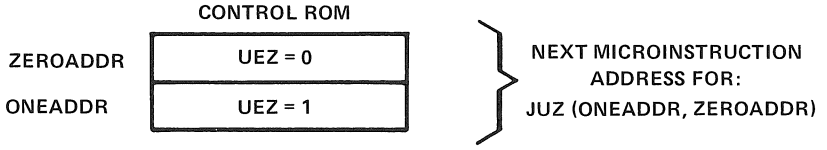


FIGURE 4-5 — JUZ DISPATCH

4.2.5 Test if Interrupt — INT.

The microcode may test for a pending interrupt by dispatching on the IACT (Interrupt Active) signal input from the Peripheral/Memory Controller. This is accomplished by specifying #JMPCNTL = 100, or in a MICASM statement by:

INT(oneaddr,zeroaddr)

As with the JT7 and JUZ instructions, oneaddr denotes the microinstruction address branch to if IACT = 1, and Zeroaddr is the address branched to if IACT = 0. Zeroaddr and oneaddr must be adjacent, as depicted in Figure 4-6.

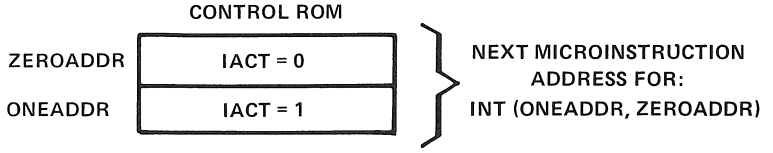


FIGURE 4-6 — INT DISPATCH

The IACT line is asserted by the Peripheral/Memory Controller (PMC) when an interrupt condition is detected. IACT can be asserted only when STINT (Status Interrupt Enable) is 1. Operation of the PMC in asserting interrupts is further explained in the TMS7000 8-Bit Microcomputer Data Manual (Part Number MP 008A).

4.2.6 Group Dispatch — IRH.

Dispatching on the Group field of the IR Register is accomplished by specifying 101 in the #JMPCNTL field. This is indicated by coding

IRH(baseaddr)

in a MICASM statement. The baseaddr field is loaded into the #JMPADDR field of the microinstruction being defined.

There are 24 groups defined, 8 in Format 0 (IR(7) = 0) and 16 in Format 1 (IR(7) = 1). The groups are numbered in Figure 4-7:

Format 0		Format 1	
IR	**Group Number**	**IR**	**Group Number**
0000XXXX	0	10000XXX	8L
0001XXXX	1	10001XXX	8H
0010XXXX	2	10010XXX	9L
0011XXXX	3	10011XXX	9H
0100XXXX	4	10100XXX	AL
0101XXXX	5	10101XXX	AH
0110XXXX	6	10110XXX	BL
0111XXXX	7	10111XXX	BH
		11000XXX	CL
		11001XXX	CH
		11010XXX	DL
		11011XXX	DH
		11100XXX	EL
		11101XXX	EH
		11110XXX	FL
		11111XXX	FH

FIGURE 4-7 — TMS7000 GROUP NUMBERS

The IRH(baseaddr) symbol performs a 24-way dispatch on the Group field. This is done by replacing the low order bits of #JMPADDR with a function of the Group number. The high nibble of the IR Register, IR(7-4), is placed in the low nibble of the next address, shifted by 1 bit. The low order bit of the next address, is defined as NEXTADRESS(0) = IR(3).OR.(.NOT.IR(7)). For Format 0 instructions, NOT IR(7) = 1, and NEXTADDRESS(0) always equals 1. Thus, the machine will jump to microaddress baseaddr + (group*2) + 1 for Format 0 group numbers. For Format 1 instructions, NOT IR(7) = 0, and NEXTADDRESS(0) equals IR(3). Thus, the machine will jump to microaddress baseaddr + (group*2) + IR(3) for Format 1 group numbers. The group names given in Figure 4-7 are the first hex digit in the two-digit hex representation of the IR Register contents. Format 1 names have an L if IR(3) = 0 and H if IR(3) = 1. The operation of the Group decode is depicted in Figure 4-8.

The CROM addresses baseaddr, baseaddr + 2, baseaddr + 4, etc., may be used for other microinstructions. The microcode for the TMS7000 Standard Instruction Set uses the IRH dispatch immediately after the assembly language instruction is loaded into the IR. Each group corresponds to an addressing mode for the instruction, and the microcode executed after the dispatch fetches the appropriate operands. Typically, a Function, or IRL, dispatch is then performed, and the microcode branches to perform the appropriate ALU function on the operands. In this manner, the operand fetch microinstructions are shared among the assembly language instructions and each instruction has its own microcode to perform the function of that instruction.

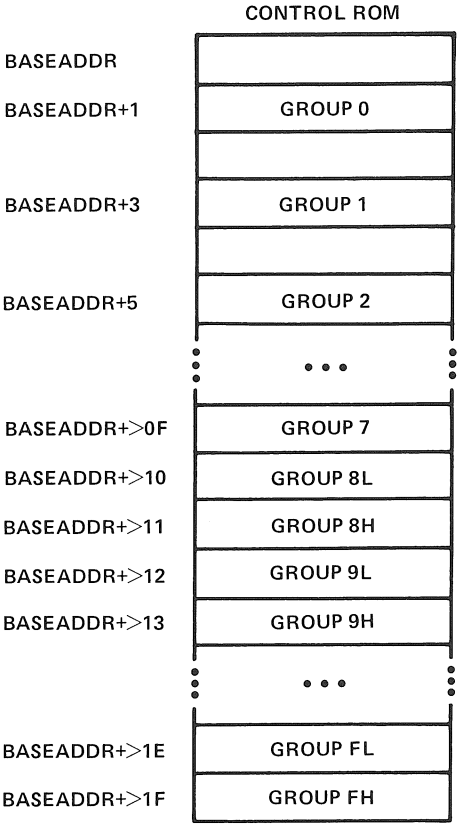


FIGURE 4-8 — IRH DISPATCH

4.2.7 Test if Carry — JC.

The microcode may test the value of the carry bit in the Status Register by performing a dispatch on the STC Bit. This is indicated by specifying #JMPCNTL(2-0) = 110, or

JC(oneaddr,zeroaddr)

appearing in a MICASM statement. The bit tested is the value of the STC (Status Carry) Bit after the execution of the immediately preceeding microinstruction, i.e., the microinstruction executed prior to the one containing the JC(oneaddr,zeroaddr) statement. The STC Bit is placed in bit 0 of #JMPADDR, and the result used as the next microinstruction address.

If the STC Bit is 1, control transfers to oneaddr, and if STC = 0, control transfers to zeroaddr. The locations zeroaddr and oneaddr must be adjacent, with zeroaddr on an even address and oneaddr on the subsequent odd address. This is diagrammed in Figure 4-9.

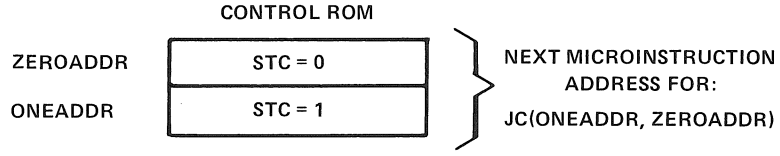


FIGURE 4-9 — JC DISPATCH

4.2.8 Test Status Register — MJMP.

The contents of the status register may be tested with the Macro Jump dispatch by specifying #JMPCNTL(2-0) = 111. This is indicated by

MJMP(oneaddr,zeroaddr)

appearing in the MICASM statement for a microinstruction. The MJMP dispatch tests eight possible conditions of the Status Register, indicated by the 3 bits in IR(2-0). If the condition is true, control transfers to oneaddr. If the condition is not true, control transfers to zeroaddr. The conditions tested are indicated in Figure 4-10.

IR(2-0)	Condition Tested			Comment
	STC	STSB	STEZ	
0 0 0	X	X	X	Unconditionally Jump
0 0 1	X	1	X	Jump if Negative
0 1 0	X	X	1	Jump if Zero
0 1 1	1	X	X	Jump if Carry
1 0 0	X	0	0	Jump if Positive
1 0 1	X	0	X	Jump if Positive or Zero
1 1 0	X	X	0	Jump if Not Zero
1 1 1	0	X	X	Jump if No Carry

FIGURE 4-10 — MACRO JUMP CONDITIONS

The X's in the Condition Tested column indicate don't care conditions.

The result of the condition test is placed in Bit 0 of #JMPADDR to form the new microinstruction address. The address oneaddr must be the odd address immediately following zeroaddr, as shown in Figure 4-11.

The MJMP dispatch is used in the microcode of the TMS7000 Standard Instruction Set to implement the conditional branch instruction.

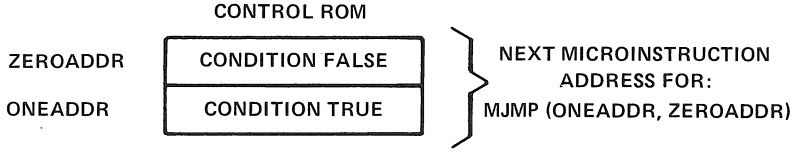


FIGURE 4-11 — MJMP DISPATCH

4.3 RESET OPERATION

When the RESET- pin is asserted externally, the PMC asserts the RST signal on the C Bus between the PMC and CPU. The entry-point logic immediately forces the next microinstruction address to be > FF. Unlike the normal interrupt facility, the microcode does not poll the RST line; rather, the microinstruction at CROM address > FF is unconditionally forced to be the next microinstruction executed.

In the TMS7000 Standard Instruction Set, the sequence of microinstructions executed upon reset fetch a subroutine entry point address at address > FFFE in memory (in the on-chip ROM) and branch to the subroutine.

INDEX

TI Sales Offices

ALABAMA: Huntsville, 500 Wynn Drive, Suite 514, Huntsville, AL 35805, (205) 837-7530.

ARIZONA: Phoenix, P.O. Box 35160, 8102 N. 23rd Ave., Suite B, Phoenix, AZ 85021, (602) 995-1007.

CALIFORNIA: El Segundo, 831 S. Douglas St., El Segundo, CA 90245, (213) 973-2571; **Irvine,** 17891 Cartwright Rd., Irvine, CA 92714, (714) 660-1200; **Sacramento,** 1900 Point West Way, Suite 171, Sacramento, CA 95815, (916) 929-1521; **San Diego,** 4333 View Ridge Ave., Suite B, San Diego, CA 92123, (714) 278-9600; **Santa Clara,** 5353 Betsy Ross Dr., Santa Clara, CA 95054, (408) 980-9000; **Woodland Hills,** 21220 Erwin St., Woodland Hills, CA 91367, (213) 704-7759.

COLORADO: Denver, 9725 E. Hampden St., Suite 301, Denver, CO 80231, (303) 695-2800.

CONNECTICUT: Wallingford, 9 Barnes Industrial Park Rd, Barnes Industrial Park, Wallingford, CT 06492, (203) 269-0074.

FLORIDA: Clearwater, 2280 U.S. Hwy 19 N., Suite 232, Clearwater, FL 33515, (813) 796-1926; **Ft. Lauderdale,** 2765 N.W. 62nd St., Ft. Lauderdale, FL 33309, (305) 973-8502; **Maitland,** 2601 Maitland Center Parkway, Maitland, FL 32751, (305) 646-9600.

GEORGIA: Atlanta, 3300 Northeast Expy., Building 9, Atlanta, GA 30341, (404) 452-4600.

ILLINOIS: Arlington Heights, 515 W. Algonquin, Arlington Heights, IL 60005, (312) 640-2934.

INDIANA: Ft. Wayne, 2020 Inwood Dr., Ft. Wayne, IN 46805 (219) 424-5174; **Indianapolis,** 2346 S. Lynhurst, Suite J-400, Indianapolis, IN 46241, (317) 248-8555.

IOWA: Cedar Rapids, 373 Collins Rd. NE, Suite 200, Cedar Rapids, IA 52402, (319) 395-9550.

MARYLAND: Baltimore, 1 Rutherford Pl., 7133 Rutherford Rd., Baltimore, MD 21207, (301) 944-8600.

MASSACHUSETTS: Waltham, 504 Totten Pond Rd., Waltham, MA 02154, (617) 890-7400.

MICHIGAN: Farmington Hills, 33737 W. 12 Mile Rd., Farmington Hills, MI 48018, (313) 553-1500.

MINNESOTA: Edina, 7625 Parklawn, Edina, MN 55435, (612) 830-1600.

MISSOURI: Kansas City, 8080 Ward Pkwy., Kansas City, MO 64114, (816) 523-2500; **St. Louis,** 11861 Westline Industrial Drive, St. Louis, MO 63141, (314) 569-7600.

NEW JERSEY: Clark, 292 Terminal Ave. West, Clark, NJ 07066, (201) 574-9800.

NEW MEXICO: Albuquerque, 5907 Alice NSE, Suite E, Albuquerque, NM 87110, (505) 265-8491.

NEW YORK: East Syracuse, 6700 Old Collamer Rd., East Syracuse, NY 13057, (315) 463-9291; **Endicott,** 112 Nanticoke Ave., P.O. Box 618, Endicott, NY 13760, (607) 754-3900; **Melville,** 1 Huntington Quadrangle, Suite 3C10, P.O. Box 2936, Melville, NY 11747, (516) 454-6600; **Poughkeepsie,** 201 South Ave., Poughkeepsie, NY 12601, (914) 473-2900; **Rochester,** 1210 Jefferson Rd., Rochester, NY 14623, (716) 424-5400.

NORTH CAROLINA: Charlotte, 8 Woodlawn Green, Woodlawn Rd., Charlotte, NC 28210, (704) 527-0930; **RALEIGH,** 3000 Highwoods Blvd., Suite 118, Raleigh, NC 27625, (919) 876-2725.

OHIO: Beachwood, 23408 Commerce Park Rd., Beachwood, OH 44122, (216) 464-6100, **Dayton,** Kingsley Bldg., 4124 Linden Ave., Dayton, OH 45432, (513) 258-3877.

OKLAHOMA: Tulsa, 3105 E. Skelly Dr., Suite 110, Tulsa, OK 74105, (918) 749-9547.

OREGON: Beaverton, 6700 SW 105th St., Suite 110, Beaverton, OR 97005, (503) 643-6758.

PENNSYLVANIA: Ft. Washington, 575 Virginia Dr., Ft. Washington, PA 19034, (215) 643-6450; Coraopolis, PA 15108, 420 Rouser Rd., 3 Airport Office PK, (412) 771-8550.

TENNESSEE: Johnson City, P.O. Drawer 1255, Erwin Hwy., Johnson City, TN 37601, (615) 461-2191.

TEXAS: Austin, 12501 Research Blvd., P.O. Box 2909, Austin, TX 78723, (512) 250-7655; **Dallas,** P.O. Box 1087, Richardson, TX 75080; **Houston,** 9100 Southwest Frwy., Suite 237, Houston, TX 77036, (713) 778-6592; **San Antonio,** 1000 Central Park South, San Antonio, TX 78232, (512) 496-1779.

UTAH: Salt Lake City, 3672 West 2100 South, Salt Lake City UT 84120, (801) 973-6310.

VIRGINIA: Fairfax, 3001 Prosperity, Fairfax, VA 22031, (703) 849-1400; **Midlothian,** 13711 Sutter's Mill Circle, Midlothian, VA 23113, (804) 744-1007.

WISCONSIN: Brookfield, 205 Bishops Way, Suite 214, Brookfield, WI 53005, (414) 784-3040.

WASHINGTON: Redmond, 2723 152nd Ave., N.E. Bldg 6, Redmond, WA 98052, (206) 881-3080.

CANADA: Ottawa, 436 McClaren St., Ottawa, Canada, K2P0M8, (613) 233-1177; **Richmond Hill,** 280 Centre St. E., Richmond Hill L4C1B1, Ontario, Canada, (416) 884-9181; **St. Laurent,** Ville St. Laurent Quebec, 9460 Trans Canada Hwy., St. Laurent, Quebec, Canada H4S1R7, (514) 334-3635. P

TI Distributors

ALABAMA: Hall-Mark (205) 837-8700.

ARIZONA: Phoenix, Kierulff (602) 243-4101; Marshall (602) 968-6181; R.V. Weatherford (602) 272-7144; Wyle (602) 249-2232; **Tucson,** Kierulff (602) 624-9986.

CALIFORNIA: Los Angeles/Orange County, Arrow (213) 701-7500, (714) 851-8961; JACO (714) 540-5600, (213) 998-2200; Kierulff (213) 725-0235, (714) 731-5711; Marshall (213) 999-5001, (213) 686-0141, (714) 556-6400; R.V. Weatherford (714) 634-9600, (213) 849-3451, (714) 623-1261; Wyle (213) 322-8100, (714) 641-1611; **San Diego,** Arrow (714) 565-4800; Kierulff (714) 278-2112; Marshall (714) 578-9600; R.V. Weatherford (714) 695-1700; Wyle (714) 565-9171; **San Francisco Bay Area,** Arrow (408) 745-6600; Kierulff (415) 968-6292; Marshall (408) 732-1100; Time (408) 734-9888; United Components (408) 496-6900; Wyle (408) 727-2500; **Santa Barbara,** R.V. Weatherford (805)465-8551.

COLORADO: Arrow (303) 758-2100; Kierulff (303) 371-6500; R.V. Weatherford (303) 428-6900; Wyle (303) 457-9953.

CONNECTICUT: Arrow (203) 265-7741; Diplomat (203) 797-9674; Kierulff (203) 265-1115; Marshall (203) 265-3822; Milgray (203) 795-0714.

FLORIDA: Ft. Lauderdale, Arrow (305) 973-8502; Diplomat (305) 971-7160; Hall-Mark (305) 971-9280; Kierulff (305) 652-6950; **Orlando,** Arrow (305) 725-1480; Diplomat (305) 725-4520; Hall-Mark (305) 855-4020; Milgray (305) 647-5747; **Tampa,** Diplomat (812) 443-4514; Kierulff (813)576-1966.

GEORGIA: Arrow (404) 449-8252 Hall-Mark (404) 447-8000; Kierulff (404) 447-5252; Marshall (404) 923-5750.

ILLINOIS: Arrow (312) 397-3440; Diplomat (312) 595-1000; Hall-Mark (312) 860-3800; Kierulff (312) 640-0200; Newark (312) 638-4411.

INDIANA: Indianapolis, Arrow (317) 243-9353; Graham (317) 634-8202; **Ft. Wayne,** Graham (219) 423-3422.

IOWA: Arrow (319) 395-7230.

KANSAS: Kansas City, Component Specialties (913) 492-3555; Hall-Mark (913) 888-4747; **Wichita,** LCOMP (316) 265-9507.

MARYLAND: Arrow (301) 247-5200; Diplomat (301) 995-1226; Hall-Mark (301) 796-9300; Kierulff (301) 247-5020; Milgray (301) 468-6400.

MASSACHUSETTS: Arrow (617) 933-8130; Diplomat (617) 429-4120; Kierulff (617) 667-8331; Marshall (617) 272-8200; Time (617) 935-8080.

MICHIGAN: Detroit, Arrow (313) 971-8200; Newark (313) 967-0600; **Grand Rapids,** Newark (616) 243-0912.

MINNESOTA: Arrow (612) 830-1800; Diplomat (612) 788-8601; Hall-Mark (612) 854-3223; Kierulff (612) 941-7500.

MISSOURI: Kansas City, LCOMP (816) 221-2400; **St. Louis,** Arrow (314) 567-6888; Hall-Mark (314) 291-5350; Kierulff (314) 739-0855.

NEW HAMPSHIRE: Arrow (603) 668-6968.

NEW JERSEY: Arrow (201) 575-5300; Diplomat (201) 785-1830; JACO (201) 778-4722; Kierulff (201) 575-6750; Marshall (201) 340-1900.

NEW MEXICO: Arrow (505) 243-4566; International Electronics (505) 345-8127.

NEW YORK: Long Island, Arrow (516) 231-1000; Diplomat (516) 454-6400; JACO (516) 273-5500; Marshall (516) 273-2424; Milgray (516) 546-5600, (800) 645-3986; **Rochester,** Arrow (716) 275-0300; Marshall (716) 235-7620; Rochester Radio Supply (716) 454-7800; **Syracuse,** Arrow (315) 652-1000; Diplomat (315) 652-5000; Marshall (607) 754-1570.

NORTH CAROLINA: Arrow (919) 876-3132, (919) 725-8711; Hall-Mark (919) 872-0712; Kierulff (919) 852-6261.

OHIO: Cincinnati, Graham (513) 772-1661; **Cleveland,** Arrow (216) 248-3990; Hall-Mark (216) 473-2907; Kierulff (216) 587-6558; **Columbus,** Hall-Mark (614) 846-1882; **Dayton,** Arrow (513) 435-5563; ESCO (513) 226-1133; Marshall (513) 236-8088.

OKLAHOMA: Component Specialties (918) 664-2820; Hall-Mark (918) 665-3200; Kierulff (918) 252-7537.

OREGON: Kierulff (503) 641-9150; Wyle (503) 640-6000.

PENNSYLVANIA: Arrow (412) 856-7000; Arrow (609) 235-1900; General Radio (609) 964-8560; Hall-Mark (609) 424-0880; Milgray (609) 983-5010.

TEXAS: Austin, Arrow (512) 835-4180; Component Specialties (512) 837-8922; Hall-Mark (512) 258-8848; Kierulff (512) 835-2090; **Dallas,** Arrow (214) 386-7500; Component Specialties (214) 357-6511; Hall-Mark (214) 341-1147; International Electronics (214) 233-9323; Kierulff (214) 343-2400; **El Paso,** International Electronics (915) 778-9761; **Houston,** Arrow (713) 491-4100; Component Specialties (713) 771-7237; Hall-Mark (713) 781-6100; Harrison Equipment (713) 879-2600; Kierulff (713) 530-7030.

UTAH: Diplomat (801) 486-4134; Kierulff (801) 973-6913; Wyle (801) 974-9953.

VIRGINIA: Arrow (04) 282-0413.

WASHINGTON: Arrow (206) 643-4800; Kierulff (206) 575-4420; United Components (206) 643-7444; Wyle (206) 453-8300.

WISCONSIN: Arrow (414) 764-6600; Hall-Mark (414) 761-3000; Kierulff (414) 784-8160.

CANADA: Calgary, Future (403) 259-6408; Varah (403) 230-1235; **Downsview,** CESCO (416) 661-0220; **Hamilton,** Varah (416) 561-9311; **Montreal,** CESCO (514) 735-5511; Future (514) 694-7710; **Ottawa,** CESCO (613) 226-6905; Future (613) 820-8313; **Quebec City,** CESCO (418) 687-4231; **Toronto,** Future (416) 663-5563; **Vancouver,** Future (604) 438-5545; Varah (604) 873-3211; **Winnipeg,** Varah (204) 633-6190. AJ

TEXAS INSTRUMENTS

Printed in U.S.A.

TI Worldwide Sales Offices

ALABAMA: Huntsville, 500 Wynn Drive, Suite 514, Huntsville, AL 35805, (205) 837-7530.

ARIZONA: Phoenix, P.O. Box 35160, 8102 N. 23rd Ave., Suite B, Phoenix, AZ 85021, (602) 995-1007.

CALIFORNIA: El Segundo, 831 S. Douglas St., El Segundo, CA 90245, (213) 973-2571; **Irvine,** 17891 Cartwright Rd., Irvine, CA 92714, (714) 660-1200; **Sacramento,** 1900 Point West Way, Suite 171, Sacramento, CA 95815, (916) 929-1521; **San Diego,** 4333 View Ridge Ave., Suite B., San Diego, CA 92123, (714) 278-9600; **Santa Clara,** 5353 Betsy Ross Dr., Santa Clara, CA 95054, (408) 980-9000; **Woodland Hills,** 21220 Erwin St., Woodland Hills, CA 91367, (213) 704-7759.

COLORADO: Denver, 9725 E. Hampden St., Suite 301, Denver, CO 80231, (303) 695-2800.

CONNECTICUT: Wallingford, 9 Barnes Industrial Park Rd., Barnes Industrial Park, Wallingford, CT 06492, (203) 269-0074.

FLORIDA: Clearwater, 2280 U.S. Hwy. 19 N., Suite 232, Clearwater, FL 33515, (813) 796-1926; **Ft. Lauderdale,** 2765 N.W. 62nd St., Ft. Lauderdale, FL 33309, (305) 973-8502; **Maitland,** 2601 Maitland Center Parkway, Maitland, FL 32751, (305) 646-9600.

GEORGIA: Atlanta, 3300 Northeast Expy., Building 9, Atlanta, GA 30341, (404) 452-4600.

ILLINOIS: Arlington Heights, 515 W. Algonquin, Arlington Heights, IL 60005, (312) 640-2934.

INDIANA: Ft. Wayne, 2020 Inwood Dr., Ft. Wayne, IN 46805, (219) 424-5174; **Indianapolis,** 2346 S. Lynhurst, Suite J-400, Indianapolis, IN 46241, (317) 248-8555.

IOWA: Cedar Rapids, 373 Collins Rd. NE, Suite 200, Cedar Rapids, IA 52402, (319) 395-9550.

MARYLAND: Baltimore, 1 Rutherford Pl., 7133 Rutherford Rd., Baltimore, MD 21207, (301) 944-8600.

MASSACHUSETTS: Waltham, 504 Totten Pond Rd., Waltham, MA 02154, (617) 890-7400.

MICHIGAN: Farmington Hills, 33737 W. 12 Mile Rd., Farmington Hills, MI 48018, (313) 553-1500.

MINNESOTA: Edina, 7625 Parklawn, Edina, MN 55435, (612) 830-1600.

MISSOURI: Kansas City, 8080 Ward Pkwy., Kansas City, MO 64114, (816) 523-2500; **St. Louis,** 11861 Westline Industrial Drive, St. Louis, MO 63141, (314) 569-7600.

NEW JERSEY: Clark, 292 Terminal Ave. West, Clark, NJ 07066, (201) 574-9800.

NEW MEXICO: Albuquerque, 5907 Alice NSE, Suite E, Albuquerque, NM 87110, (505) 265-8491.

NEW YORK: East Syracuse, 6700 Old Collamer Rd., East Syracuse, NY 13057, (315) 463-9291; **Endicott,** 112 Nanticoke Ave., P.O. Box 618, Endicott, NY 13760, (607) 754-3900; **Melville,** 1 Huntington Quadrangle, Suite 3C10, P.O. Box 2936, Melville, NY 11747, (516) 454-6600; **Poughkeepsie,** 201 South Ave., Poughkeepsie, NY 12601, (914) 473-2900; **Rochester,** 1210 Jefferson Rd., Rochester, NY 14623, (716) 424-5400.

NORTH CAROLINA: Charlotte, 8 Woodlawn Green, Woodlawn Rd., Charlotte, NC 28210, (704) 527-0930; **RALEIGH,** 3000 Highwoods Blvd., Suite 118, Raleigh, NC 27625, (919) 876-2725.

OHIO: Beachwood, 23408 Commerce Park Rd., Beachwood, OH 44122, (216) 464-6100, **Dayton,** Kingsley Bldg., 4124 Linden Ave., Dayton, OH 45432, (513) 258-3877.

OKLAHOMA: Tulsa, 3105 E. Skelly Dr., Suite 110, Tulsa, OK 74105, (918) 749-9547.

OREGON: Beaverton, 6700 SW 105th St., Suite 110, Beaverton, OR 97005, (503) 643-6758.

PENNSYLVANIA: Ft. Washington, 575 Virginia Dr., Ft. Washington, PA 19034, (215) 643-6450; Coraopolis, PA 15108, 420 Rouser Rd., 3 Airport Office PK, (412) 771-8550

TENNESSEE: Johnson City, P.O. Drawer 1255, Erwin Hwy., Johnson City, TN 37601, (615) 461-2191.

TEXAS: Austin, 12501 Research Blvd., P.O. Box 2909, Austin, TX 78723, (512) 250-7655; **Dallas,** P.O. Box 1087, Richardson, TX 75080; **Houston,** 9100 Southwest Frwy., Suite 237, Houston, TX 77036, (713) 778-6592; **San Antonio,** 1000 Central Park South, San Antonio, TX 78232, (512) 496-1779.

UTAH: Salt Lake City, 3672 West 2100 South, Salt Lake City UT 84120, (801) 973-6310.

VIRGINIA: Fairfax, 3001 Prosperity, Fairfax, VA 22031, (703) 849-1400; **Midlothian,** 13711 Sutter's Mill Circle, Midlothian, VA 23113, (804) 744-1007.

WISCONSIN: Brookfield, 205 Bishops Way, Suite 214, Brookfield, WI 53005, (414) 784-3040.

WASHINGTON: Redmond, 2723 152nd Ave., N.E. Bldg 6, Redmond, WA 98052, (206) 881-3080.

CANADA: Ottawa, 436 McClaren St., Ottawa, Canada, K2P0M8, (613) 233-1177; **Richmond Hill,** 280 Centre St. E., Richmond Hill L4C1B1, Ontario, Canada, (416) 884-9181; **St. Laurent,** Ville St. Laurent Quebec, 9460 Trans Canada Hwy., St. Laurent, Quebec, Canada H4S1R7, (514) 334-3635. P

ARGENTINA, Texas Instruments Argentina S.A.I.C.F: Km. 25.5 Ruta Panamericana Don Torcuato, C.C. 2296, 1000-Correo Central, Buenos Aires, Argentina, 748-1141.

AUSTRALIA, Texas Instruments Australia Ltd: Unit 1A, 9 Byfield St., P.O. Box 106, North Ryde, N.S.W. 2113, Sydney, Australia, 02-887-1122; 5th floor, 418 St. Kilda Road, Melbourne, 3004, Victoria, Australia, 03-267-4677.

AUSTRIA, Texas Instruments Ges. m.b.H: Rennweg 17, 1030 Vienna, Austria, 0222-724186.

BELGIUM, Texas Instruments S A: Mercure Centre, Raketstraat, Rue De La Fusee 100, 1130 Brussels, Belgium, 02-7208000.

BRAZIL, Texas Instrumentos Electronicos do Brasil Ltda: Rua Padre Pereira De Andrade, 591 Cep-05469 Sao Paulo, Brazil, 011-260-6347.

DENMARK, Texas Instruments A D: Marielundvej 46E, 2730 Herlev, Denmark, 02-917400.

FINLAND, Texas Instruments Finland OY: Fressenkatu 6, P.L. 917, 00101 Helsinki 10, Finland, 80-408300.

FRANCE, Texas Instruments France: La Boursidiere, Bat. A, R.N. 186, 92350 Le Plessis Robinson, France, 01-6302343; 31 Quai Rambaud, 69002 Lyon, France, 078-373585; 1, Av. de la Chartreuse, 38240 Meylan, France, 076-904574; 9, Place de Bretange, 35000 Rennes, France, 099-795481; 100-102 Alle de Barcelone, Residence L'Autay, 31000 Toulouse, France, 061-213032.

GERMANY, Texas Instruments Deutschland GmbH: Kurfuerstendamm 146, 1000 Berlin 31, Germany, 030-8927013; Ill Hagen 43, Frankfurter Allee 6-8, 6236 Eschborn, Germany, 06196-43074; 4300 Essen, Germany, 0201-233551; Winterhuder Weg 8, 2000 Hamburg 76, Germany; 040-2201154; Haggertystrasse 1, 8050 Freising, Germany; 08161-801; Riethorst 4, 3000 Hanover 51, Germany, 0511-648021; Arabellastrasse 13-15, 8000 Munich 81, Germany, 089-92341; Marientorgraben 3-5, 8500 Nuernberg, Germany, 0911-22877; Krefelderstrasse 11-15, 7000 Stuttgart 50, Germany, 0711-547001.

HONG KONG, Texas Instruments Asia Ltd: 902, Asian House, 1, Hennessy Rd., Hong Kong, 05-279041.

ITALY, Texas Instruments Italia Spa: Via Europa 38/44, Cologno Monzese, Milan, Italy, 02-253-2451; Via Salaria 1319, 00138 Rome, Italy, 06-6917127; Via Montebello 27, 10124 Turin, Italy, 011-832276.

JAPAN, Texas Instruments Asia Ltd: Aoyama Fuji Bldg., 4F, 3-6-12, Kita-Aoyama, Minato-ku, Tokyo, Japan 107, (03) 498-2111.

KOREA, Texas Instruments Supply Company: Room 301, Kwang Poong Bldg., 24-1 Hwayang Dong, Sungdong-Ku, Seoul, Korea, 464-1565.

MEXICO, Texas Instruments de Mexico S.A: Poniente 116 #489, Col. Industrial Vallejo, Mexico City 15, D.F., Mexico, 905-567-9200.

NETHERLANDS, Texas Instruments Holland BV: Laan Van de Helende Meesters 421 A, P.O. Box 283, 1180 AG Amstelveen, Holland, 020-473391.

NORWAY, Texas Instruments A S: Ryensvingen 15, Oslo 6, Norway, 02-689487.

PHILIPPINES, Texas Instruments Asia Ltd: 14th Floor, BA-Lepanto Building, Paseo De Roxas, Makati, Metro Manila.

PORTUGAL, Texas Instruments Equipamento Electronico LDA: Rua Eng. Frederico Ulrich, 2650 Moreira Da Maia, Douro, Portugal, 948-1003.

SINGAPORE, Texas Instruments Asia Ltd: P.O. Box 2093, 990 Bendemeer Rd., Singapore 1, Republic of Singapore, 65-2581122.

SPAIN, Texas Instruments Espana S.A: Balmes 89, 12 Barcelona 12, Spain.

SWEDEN, Texas Instruments International Trade Corporation (Sverigefilialen): Norra Hannvagen 3, Fack S-100 54 Stockholm 39, Sweden, 08-235480.

TAIWAN, Texas Instruments Taiwan Ltd: 10th floor, Fu Shing Bldg., 71 Sung-Kiang Rd., Taipei, Taiwan, Republic of China, 05-5219321.

UNITED KINGDOM, Texas Instruments Ltd: Manton Lane, Bedford, England MK417PU, 0234-67466. H

TEXAS INSTRUMENTS

Printed in U.S.A.

TEXAS INSTRUMENTS

November 1982
MP061

Post Office Box 1443 / Houston Texas 77001
Semiconductor Group

Printed in U.S.A.