ST9 FAMILY 8/16 BIT MCU

PROGRAMMING

1st EDITION

-THOMSON
OELECTRONICS

ST9 FAMILY PROGRAMMING

ST9 FAMILY
8/16 BIT MCU

PROGRAMMING

1st EDITION

MARCH 1991

USE IN LIFE SUPPORT DEVICES OR SYSTEMS MUST BE EXPRESSLY AUTHORIZED

SGS-THOMSON PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF SGS-THOMSON Microelectronics. As used herein:

1. Life support devices or systems are those which (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided with the product, can be reasonably expected to result in significant injury to the user.

2. A critical component is any component of a life support device or system whose failure to perform can reasonably be expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

GENERAL INDEX

PROGRAMMING MANUAL

INTRODUCTION

The ST9 8/16 bit microcontroller family introduces a new generation of single-chip architecture. It offers fast program execution, efficient use of memory, sophisticated interrupt handling, input/output (I/O) flexibility and bit-manipulation capabilities, with easy system expansion. Virtually all of the ST9 configuration can be tailored to the needs of the user under program control. This enables the ST9 to serve as an I/O intensive microcontroller, as an intelligent peripheral controller within a larger system, or as a memory intensive microprocessor.

Programming of the ST9 is made easy in both high level languages such as C, or directly in assembler language, by the versatility of the 14 addressing modes coupled with the comprehensive instruction set operating on bits, BCD, 8 bit bytes and 16 bit words. The availability of the Register File, giving the programmer multiple 8 and 16 bit accumulators and index pointers, the fast interrupt response time, on-chip DMA and on-chip and external memory access capabilities give the ST9 a high efficiency for real-time control applications.

The ST9 has a range of family devices made up from various memory combinations (RAM, ROM/EPROM, EEPROM), powerful peripherals such as Multifunction Timers, Analog to Digital Converters, Serial Communications Interfaces and a standard Core. The Core itself includes a Timer/Watchdog, Serial Interface, I/O ports and the 256 byte Register File.

The remainder of this section describes in more detail the ST9 features of primary interest to assembly language programmers. Please refer to the ST9 Technical Manual for detailed architectural and configuration information.

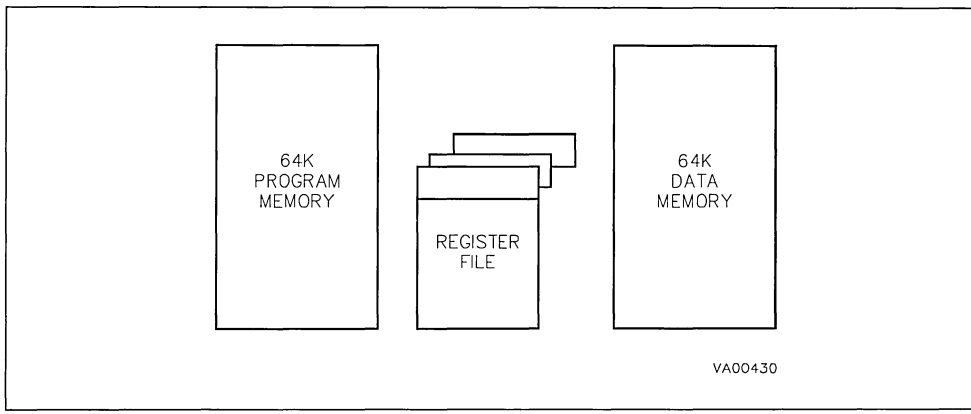
Note: This Programming Manual follows the syntax of the ST9 Software Tools (AST9 the high-level Macro Assembler running on IBM Personal Computer under MS-DOS, SUN 3 and SUN 4 under UNIX, VAX and microVAX under VMS). Register and bit names follow the recommendations of the `symbols.inc` file available as an Application Note.

ADDRESS SPACES

The ST9 has three separate address spaces:

- Program Memory for storing program instruction, with up to 64K (65536) byte for standard ST9 devices, up to 8M byte for ST9 devices with Bank Switch logic.
- the Data Memory for the storage of data, with up to 64K (65536) bytes for standard ST9 devices, up to 8M bytes for devices with Bank Switch logic.
- the Register File composed of 224 8 bit registers for all devices, plus 16 system control registers and up to 64 pages of 16 bytes for the control and status registers of the on-chip peripherals.

Figure 1. Address Spaces



MEMORY SEGMENTS

The two 64K byte memory spaces of the ST9 are addressed either directly with the 16 bit absolute memory address, or indirectly using a pair of the general purpose 8 bit registers. In addition the address may be given by an indexed mode when a short (byte) or long (word) offset is added to an indirect base word address.

Before either memory space is used, one of the two instructions SDM or SPM (Set Data Memory and Set Program Memory, respectively) should be used. There is no need to use either of these instructions again until the memory area required is to be changed. It is not necessary to use either SDM or SPM when operating with external stack pointers, where the data memory is automatically used, and when using the memory-indirect to memory-indirect post-increment addressing modes, when the memory types are specified in the instruction (ie. LDPD, Load from Data Memory to Program Memory).

An output pin (P/$\overline{D}$) can be programmed to indicate the memory space currently selected in order to be used with the external address decoding logic.

Either the Data Memory or the Program Memory can be addressed using any of the memory addressing modes.

Program Memory

The Program Memory size can be up to 64K bytes. This memory can be all external (for ROMless devices) or partially external with an internal component (ROM or EPROM devices). Access to the external Program Memory is allowed only for instruction fetches at addresses greater than the existing on-chip ROM/EPROM memory. For example, when an ST9 with 8192 bytes of on-chip Program Memory, external memory fetches are performed at addresses above location 8193, as in Figure 2.

ST9 devices with Bank Switch logic may have the Program space extended within the 64K byte addressing range by paging of the upper 32K bytes to achieve a maximum program address space of 8M bytes. The lower 32K bytes remain static and are always available for interrupt servicing, bank switching and other common program procedures.

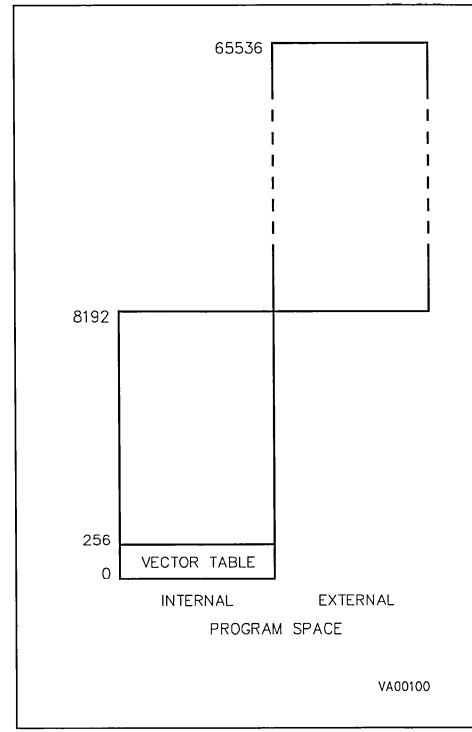
Within the Program Memory space the first 256 locations (0-255) can be used for the interrupt vector table (only locations 00h, 01h for the Reset Vector; 02h, 03h for the Divide by Zero Trap; and 04h, 05h for the Top Level priority vector are fixed). Apart from these vectors, no other part of the Program Memory has a predetermined function.

Data Memory

The Data Memory space is also of a maximum size of 64K bytes, and has exactly the same addresses and addressing modes as the Program Memory, the two spaces being distinguished by the use of the memory setting commands (SDM being relevant for setting the Data Memory). Within this space ST9 devices may include on-chip static RAM and EEPROM memory. Off-chip memory accesses will be made for address values at a higher value than the highest address of the on-chip memory component.

ST9 devices with Bank Switch logic may have the Data space extended within the 64K byte addressing range by paging of the upper 32K bytes to achieve a maximum data address space of 8M bytes (a total of 16M bytes including the Program Space). The lower 32K bytes of Data space remain static.

Figure 2. Program Memory Space



REGISTER FILE

The 256 Registers of the ST9 Register File include 224 general purpose 8 bit registers, 16 registers allocated for system functions and a paging mechanism on the top 16 registers. The pages contain the status and control registers of the on-chip peripherals which vary according to the specific device on-chip peripheral configuration.

REGISTER FILE (Continued)

The Register File is divided into 16 groups each of 16 registers which may be referred to by their hexadecimal group number; R0-R15 forming Group 0, R16-R31 forming Group 1, R160-R175 forming Group A and so on. Group E (R224-R239) is the system register group, and, as it is common to all ST9 family devices and is of specific relevance to the operation of the ST9, its functions are summarised in Figure 3.

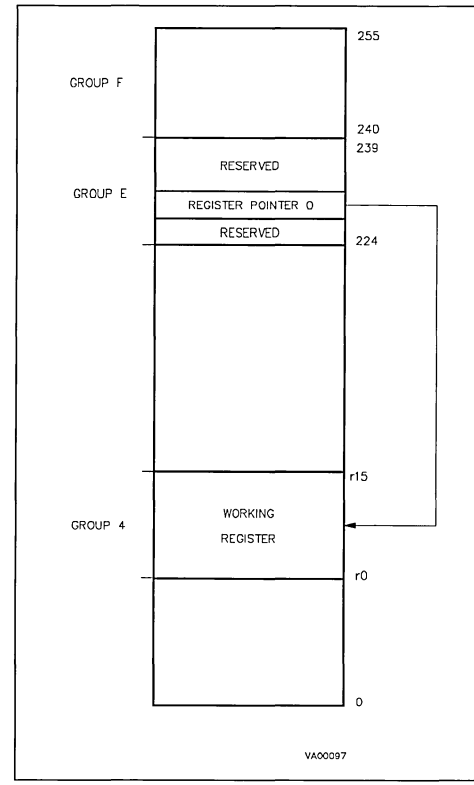
Figure 3. Group E Register Map

R239	System Stack Pointer Low (SSPLR)
	System Stack Pointer High (SSPHR)
	User Stack Pointer Low (USPLR)
	User Stack Pointer High (USPHR)
	Mode Register (MODER)
	Page Pointer (PPR)
	Register Pointer 1 (RP1R)
	Register Pointer 0 (RP0R)
	ALU Flags (FLAGR)
	Central Interrupts Control (CICR)
	Port 5 Data (P5DR)
	Port 4 Data (P4DR)
	Port 3 Data (P3DR)
	Port 2 Data (P2DR)
	Port 1 Data (P1DR)
R224	Port 0 Data (P0DR)

The ST9 Instruction Set allows direct access to all of the registers of the ST9 (see warning below). Each of the 224 general purpose registers can function as an accumulator, register address pointer or as an index register. In addition pairs of these registers may be used to provide 16 bit capability for memory addressing, indexing and arithmetic functions.

ST9 instructions can access registers directly or indirectly using an 8 bit address field. The ST9 also allows 4 bit addressing of the registers, which generally saves program bytes, and speeds program execution and task switching. In this 4 bit addressing mode, the register file is divided into 16 16 byte or 32 8 byte working register groups, each occupying contiguous register locations. Register pointers (within Group E, the system register group) address the starting location of the currently active working register group. One register (RP0) selects the base address for the 16 byte working register groups and a second (RP1) is used in conjunction with the first to select the 32 independent working register groups of 8 registers.

Figure 4. Single Working Register Bank

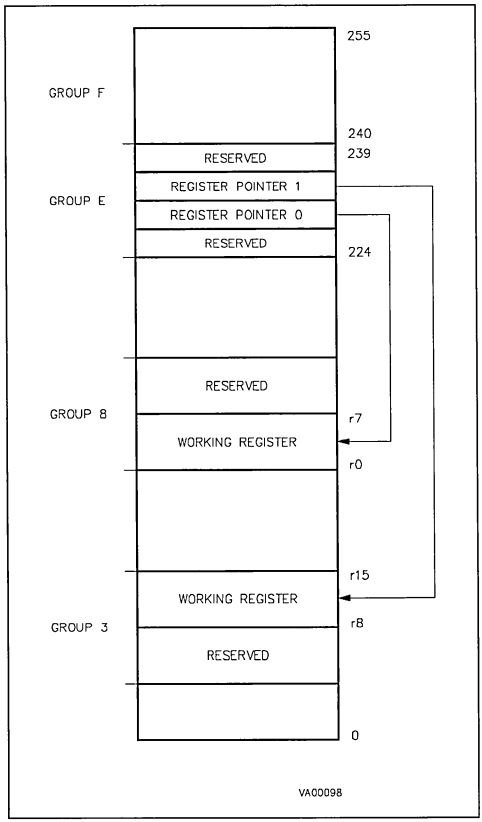


WARNING: *Working register addresses are identified in instructions with an 8 bit address field by using an upper nibble of Dh (1101b) as an escape code. As a result, Group D of the Register File can not be addressed directly but may be addressed via the working registers. It is recommended that Group D registers are used for the stacking area when the System or User stack pointers are internal.*

Changing the value of the register pointers is an easy way to save the currently active working registers (as during interrupt processing). Reserving one or more register groups for the use of interrupt-handling routines is a recommended programming practice.

REGISTER FILE (Continued)

Figure 5. Dual Working Register Bank

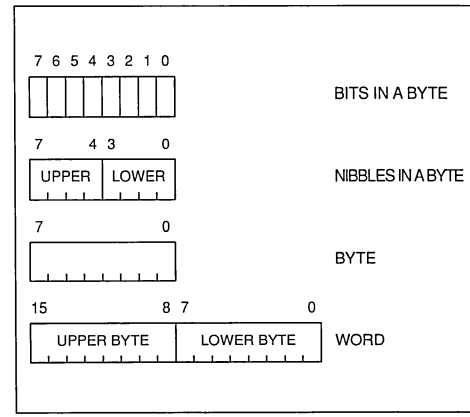


DATA LENGTHS

ST9 instructions can act on individual bits, 4 bit Binary Coded Decimal (BCD) or nibbles, 8 bit bytes, or 16 bit words.

Bits can be set, reset or tested. Nibbles are used in BCD arithmetic operations. Bytes are used for character or small integer values (in the range 0 to 255 if unsigned, or in the range -128 to +127 if signed). Words are used for larger integer values (in the range 0 to 65535 if unsigned, or in the range -32768 to +32767 if signed).

Figure 6. Data Lengths



ADDRESSING MODES

The ST9 offers a wide variety of established and new addressing modes and combinations to facilitate full and rapid access to the various address spaces while reducing program length. The available addressing modes and the special characters used in operands to identify the addressing modes are shown in Table 1.

The addressing modes available for source and destination addresses of the data for every instruction are described in detail in the Instruction Set section.

The memory addressing modes are applicable to both data and program memory spaces. Before addressing the memory, it is necessary to indicate by use of the Set Program/Data Memory instructions, `SPM` and `SDM`, in which memory space the instructions are working. This space will continue to be used until the next execution of these instructions. As each memory space is 64K bytes in size, a word address is necessary to specify memory locations.

ADDRESSING MODES (Continued)

Table 1. Addressing Modes

Addressing Mode	Notation	
Immediate Data	#N	#NN
Register Direct	r	R
Register Indirect	(r)	(R)
Register Indirect with Post-Increment	(r)+	(R)+
Register Indexed	N(r)	N(R)
Register Bit	r.b	
Memory Direct	NN	
Memory Indirect	(rr)	
Memory Indirect with Post-Increment	(rr)+	
Memory Indirect with Pre-Decrement	-(rr)	
Memory Indexed with Immediate Short Offset	N(rr)	
Memory Indexed with Immediate Long Offset	NN(rr)	
Memory Indexed with Register Offset	rr(rr)	
Memory Indirect Bit	(rr).b	

Legend: N = 8 bit Value
NN = 16 bit Value or Address
r = Working Register
R = Directly Addressed Register
( ) = Indirect Addressing
( )+ = Indirect with Post-Increment
-( ) = Indirect with Pre-Decrement
.b = Bit Number (0 to 7)

Immediate Data

Immediate Data is an addressing mode for the purposes of this discussion.

The operand value used by the instruction is the value supplied in the operand field itself. When using the immediate data addressing mode, a hashmark (#) is used to distinguish the data from an absolute address in memory.

Examples:

```
add R26,#04
```

adds 4 to the value originally contained in register R26.

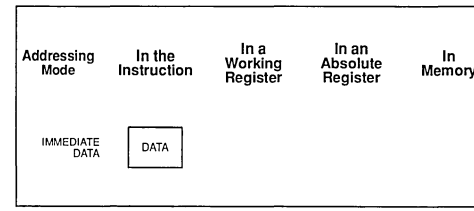
```
ldw RR42,#45017
```

loads the immediate word value 45017 (0AFD9h) into the register pair RR42 (0AFh into R42 and 0D9h into R43).

```
ldw 12355,#3467
```

loads the word value 3467 (0D8Bh) into memory locations starting at 12355 (0Dh into 12355 and 8Bh into 12356).

Figure 7. Immediate Data



Register Direct

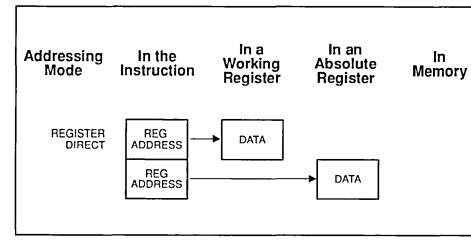
In the Register Direct addressing mode, a register can be addressed by using its absolute address in the register file. Alternatively a register can be addressed directly as a working register.

Example:

```
xch R162,r4
```

which exchanges the values in the register R162 and working register number 4.

Figure 8. Register Direct



Register Indirect

In the Register Indirect addressing mode, the address of the data does not appear in the instruction, but is located in a working register. The address of this register is located in the instruction. The indirect addressing mode is indicated by the use of parentheses.

Example: If register 200 contains 178 and working register 11 contains 86, then the instruction

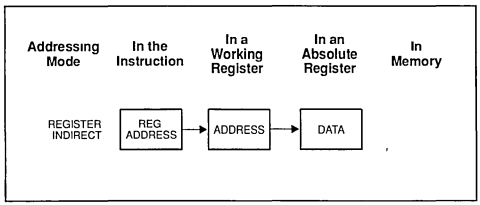
```
ld (r11),R200
```

will cause the value 178 to be loaded into register 86.

Note. the indirect address can only be contained in a working register.

ADDRESSING MODES (Continued)

Figure 9. Register Indirect



Register Indexed

To address a register using the Register Indexed mode, an offset value is used to add to an index value (which acts as a base or starting value). The offset value is the immediate value given in the instruction while the index value is given by the contents of the working register.

Example: If working register 10 contains 55 then the instruction

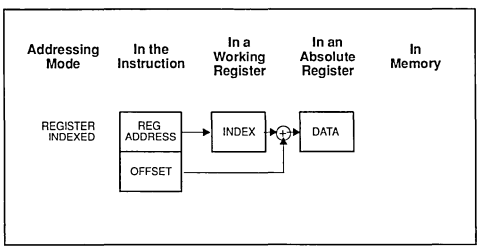
```
ld 40(r10),r18
```

will cause register 95 (i.e. 55+40) to be loaded with the content of working register 18.

The register file (256 registers) never needs an absolute value requiring more than one byte and therefore requires only a short offset and a single register to contain the index.

Note: The index value can only be contained in a working register.

Figure 10. Register Indexed



Register Indirect Post Increment

In this addressing mode, both destination and source addresses are given by the contents of the working registers which are then post-incremented. The address of the destination memory location is contained in a working register pair and the address of the register is contained in a single working register. This mode is indicated by both source and destination registers in parentheses followed by plus signs.

Example: If working register r8 contains the value 44, working register pair rr2 contains the value 2000, and register 44 contains the value 56, then by using the instruction

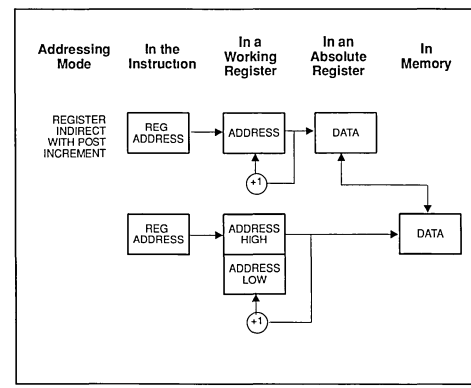
```
ld (rr2)+,(r8)+
```

the memory location 2000 will be loaded with the value 56. Immediately following this, the contents of r8 will be incremented to 45 and the contents of rr2 will be incremented to 2001.

This addressing mode is useful for moving blocks of data either from register file to memory, or from memory to register file.

Note. Only working registers may be used to contain the addresses.

Figure 11. Register Indirect Post Increment



ADDRESSING MODES (Continued)

Direct Bit

In the Direct Bit addressing mode, any bit in any working register can be addressed and potentially modified.

Examples:

```
bset r7.3
```

This instruction sets bit 3 of working register 7.

```
ld r7.3,r12.6
```

This instruction loads the value of bit 6 of working register 12 into bit 3 of working register 7.

Memory Direct

The Memory Direct addressing mode requires the specific location within the memory. This only needs the absolute address value, with no prefix or other indication necessary.

Example: If the memory location at 32184 has been assigned the label FLOW

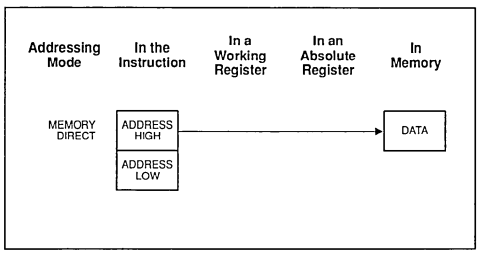
```
ld FLOW,r9
```

will enter the data in working register r9 into memory location 32184,

```
call 4308h
```

will call the subroutine at address 4308h (pushing the Program Counter onto the system stack for the return address).

Figure 12. Memory Direct



Memory Indirect

When using the Memory Indirect addressing mode to access memory, the address is contained in a pair of working registers.

Example: If the working register pair rr8 (r8,r9) contains the value 20000 then the instruction

```
ld (rr8),#34
```

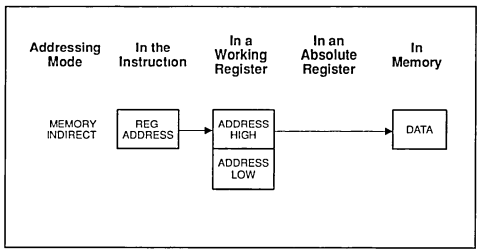
will load the value 34 to be loaded into memory location 20000.

If the data to be stored is a word, then the instruction `LDW` will automatically interpret the address as pointing to a pair of memory locations. Thus if rr8 contains 20000, then the instruction

```
ldw (rr8),#3467h
```

will cause the memory location 20000 to be loaded with the value 34h and location 20001 to be loaded with 67h.

Figure 13. Memory Indirect



Memory Indirect with Post Increment

The Memory Indirect with Post Increment addressing mode is similar to the Memory Indirect addressing mode, but, in addition, after access to the data in the currently pointed address, the value in the pointing working register pair is incremented. This mode is indicated by a plus sign following the working address pair in paretheses, e.g. (rr4)+.

Example: If working register pair rr4 (working registers r4 and r5) contains the value 3000 and memory location 3000 contains the value 88, then the instruction

```
ld R50,(rr4)+
```

will cause register 50 to be loaded with the value 88 and then the value in rr4 to be incremented to 3001. This mode is most useful in repeated situations where a number of adjacent items of data are required in succession. The use of this addressing mode saves both time and program memory space since it cuts the usual increment instruction.

ADDRESSING MODES (Continued)

Figure 14. Memory Indirect Post Increment

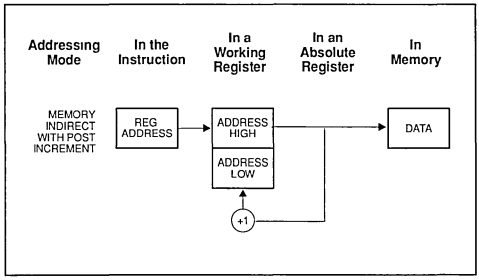
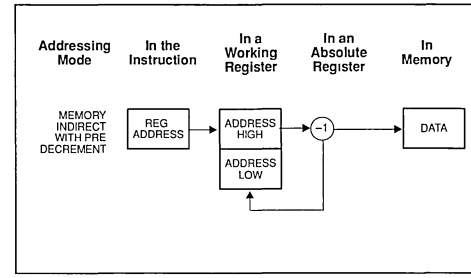


Figure 15. Memory Indirect Pre Decrement



Note. The Memory Indirect with Post Increment addressing mode may only use working registers to contain the address.

Memory Indirect with Pre Decrement

This Memory Indirect addressing mode has an automatic pre-decrement of the address contained in the pair of working registers before the action of the instruction. It is indicated by a minus sign in front of the working registers which are in parentheses, e.g. -(rr6).

Example: If working register pair rr6 contains the value 1111 and location 1110 contains the value 40, then the instruction

```
ld R56,-(rr6)
```

will cause the value in rr6 to be decremented to 1110 and then the value 40 to be loaded into register 56.

This addressing mode allows the ST9 to deal in the reverse order with data previously managed using the Memory Indirect Post-Increment mode, without resetting the pointing working registers (used with the last post-increment). The pre-decrement mode has the same benefits of time and program memory size saving as the post-increment mode.

Note. The Memory Indirect with Pre-Decrement addressing mode may only use working registers to contain the address.

Memory Indexed

There are three indexed addressing modes, each of them using an indirect address plus offset format. The index base address is given as an indirect address contained in a working register pair, while the offset can be long or short (16 or 8 bit) immediate values, or a register pair value. The address of the data required is given by the value of the working register pair indicated (the index), plus the value of the given offset. The specification of this offset which differentiates the three modes, is as follows.

Indexed with Immediate Short and Long Offset. In these indexed modes, the offset is an immediate value included in the instruction. It may be either a short (8 bit) or long (16 bit) index as required, this immediate value being added to the address given by the working register pair.

Example: If the working register pair rr6 has been assigned the label `SINE` and contains the value 8000 and memory location 8034 contains the value 254, then the instruction

```
ld R55,34(SINE)
```

will cause the value 254 to be loaded into register 55.

If working register pair rr2 contains the value 2000 and register 78 contains the value 34, then the instruction

```
ld 322(rr2),R78
```

will cause the value 34 to be loaded into memory location 2322.

ADDRESSING MODES (Continued)

Figure 16. Memory Indexed with Immediate Short Offset

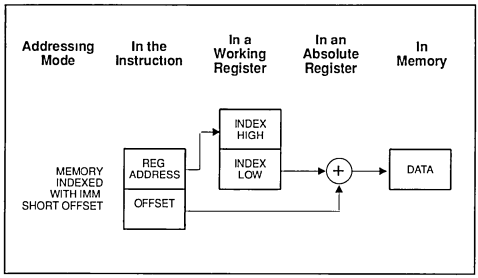
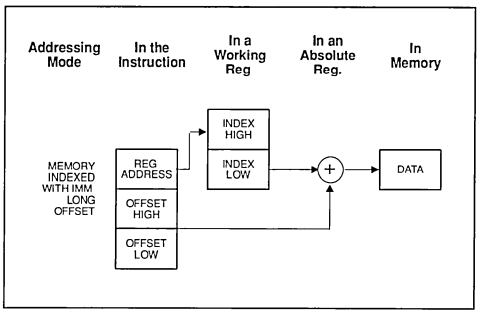


Figure 17. Memory Indexed with Immediate Long Offset



This addressing mode is useful for processing table of data, where the start address of the table may be held in the working registers and the offset to the required variable is held in the instruction.

Indexed with a Register Offset. In this addressing mode the index is supplied by one pair of working registers and the offset is supplied by a second pair of working registers. The format is rrx(rry), where x and y are in the range 0, 2, 4...12, 14.

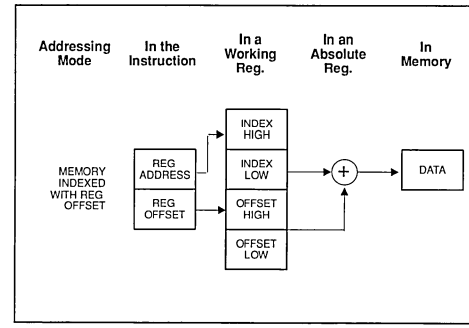
Example: If working register pair rr0 contains the value 2222 and working register pair rr4 contains 3333 while register R45 contains the value 78, then the instruction

```
ld rr4(rr0),R45
```

will cause the value 78 to be loaded into memory location 5555.

This addressing mode is useful for processing tables of data, where the start address of the table is held in working registers and the offset to the required variable is a user calculated value.

Figure 18. Memory Indexed with Register Offset



Memory Indirect Bit

In the Memory Indirect bit addressing mode, any bit of any writable Program/Data memory location can be addressed with the BTSET (Bit Test and Set) instruction.

Example:

```
btset (rr8).3
```

sets bit 3 of the memory location addressed by the working registers rr8 (r8 and r9) and indicates the original content of the bit in the Zero bit of the FLAGR register. This instruction is useful in multi-tasking applications where it may be used for semaphore flags, indicating resource allocation between tasks.

SYSTEM GROUP

The ST9 system group is common to all ST9 family devices and contains registers of major interest to programmers. The sixteen registers are located in Group E, i.e. registers 224 to 239 (0E0h to 0EFh). In the following paragraphs a brief explanation is given for each system register and for its specific function. Please refer to the ST9 Technical Manual for full information on these registers.

Stack Pointers

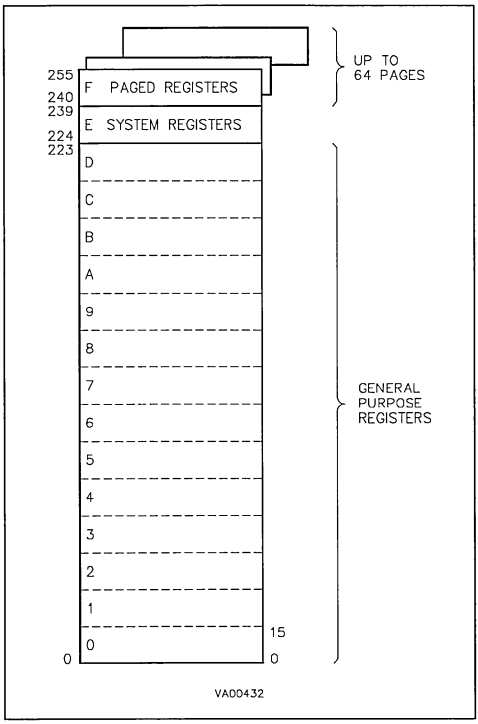
Two separate, double register stack pointers (named are System Stack Pointer and User Stack Pointer) are available to the programmer. Both stack pointers can address either the Register File or the Data Memory space for stacking area.

The Stack Pointers point to the bottom of the stack, that is, the location of the last saved value. Operation is in a Pre-Decrement mode when data is PUSHed onto the stack, and in a Post-Increment mode when data is POPed from the stack.

The System Stack Pointer (SSPR, R238:R239) is used for the storage of temporarily suspended sys-

SYSTEM GROUP (Continued)

Figure 19. Register File Map



tem and/or control registers (ie the Program Counter and the FLAG register) while interrupts are being serviced, and is used for the storage of the Program Counter following the CALLing of a subroutine.

The User Stack Pointer (USPR, R236:R237) is completely free from all interference from automatic operations and so provides for a totally user controlled stack area. Both Stack pointers may operate with both byte (`PUSH, POP`) and word (`PUSHW, POPW`) data, and are differentiated by appending a "`U`" to the instruction mnemonic for the User Stack (`PUSHU/PUSHUW, POPU/POPUW`). Calculated addresses may also be pushed onto the Stacks by the Push Effective Address instruction (`PEA, PEAU`), this instruction allows for code optimisation, for example for the ST9 ANSI C Compiler.

When the Stack Pointers are using Data Memory as the stack areas, a full word register is used as the pointer, while when operating with the stack area within the Register File (Groups 0 to 14 only, not within the system and paged groups) only an 8 bit register is required for addressing and consequently only the low byte of the word registers are used

Figure 20. Group E, the System Registers

R239	System Stack Pointer Low (SSPLR)
	System Stack Pointer High (SSPHR)
	User Stack Pointer Low (USPLR)
	User Stack Pointer High (USPHR)
	Mode Register (MODER)
	Page Pointer (PPR)
	Register Pointer 1 (RP1R)
	Register Pointer 0 (RP0R)
	ALV Flags (FLAGR)
	Central Interrupts Control (CICR)
	Port 5 Data (P5DR)
	Port 4 Data (P4DR)
	Port 3 Data (P3DR)
	Port 2 Data (P2DR)
	Port 1 Data (P1DR)
R224	Port 0 Data (P0DR)

(R239 for the System Stack and R237 for the User Stack). In this latter case the upper byte of the stack pointer registers (R238 and R236) must be considered as reserved. The Stack Pointers may be selected to point to RAM or Register File by the setting of the SSP (MODER.7) and USP (MODER.6) of the ST9 configuration register (MODER, R235) where a "1" denotes Register File operation (default at Reset and "0" causes Data Space operation.

Stacks can be located anywhere in the Register File (internal stacks) or the Data Memory (external stacks, even when using on-chip RAM memory). It is not necessary to set the Data Memory space using the instruction `SDM` as external stack instruction automatically use the Data memory.

WARNING: Care is necessary when managing stacks as there is no limit to stack sizes apart from the bottom of any address space in which the stack is placed. Consequently programmers are advised to use a stack pointer values as high as possible, particularly when using the Register File as a stacking area. In this case it is recommended that Group D is be used as it cannot otherwise be accessed directly due to the condition highlighted elsewhere in this document.

Example:

```
ld SSPL, # 223
; R223 is top register of Group D
```

SYSTEM GROUP (Continued)

Figure 21. Internal Stack Pointer

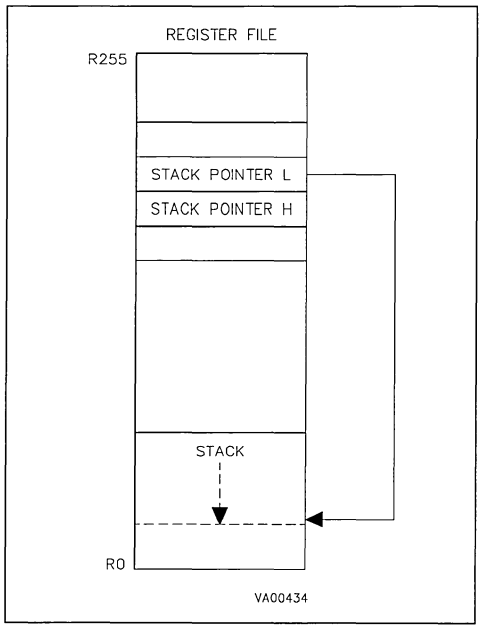
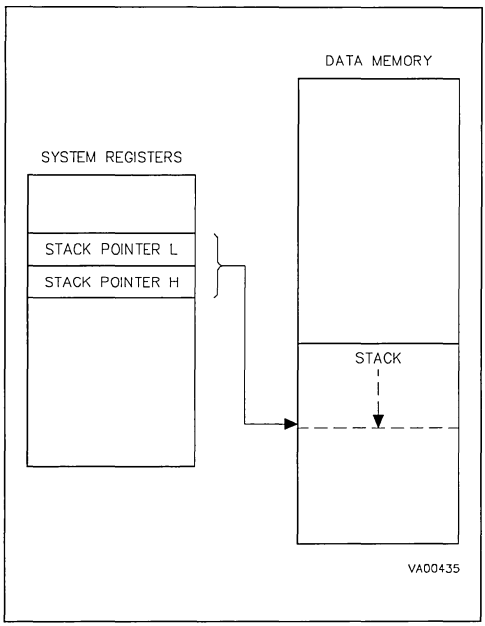


Figure 22. External Stack Pointer



Mode Register (MODER)

The ST9 Mode Register is located at register R235. It allows the programmer to select either internal or external operation of the System and User Stack Pointers, control the prescaling of the instruction cycle clock (CPUCLK) and the enable/disable the divider of the oscillator input. In addition this register allows the control of the high-impedance states for the memory interface lines.

MODER - R235 (0EBh) Sys. Reg. Read/Write

	b7							b0
Reset Value (0E0h)	1	1	1	0	0	0	0	0

b7 = **SSP:** *System Stack Pointer.* This bit selects internal (in the Register File) or external (in Data Memory) System Stack area, logical "1" for internal, and logical "0" for external. After Reset the value of this bit is "1".

b6 = **USP:** *User Stack Pointer.* Same as bit 7 for the User Stack area.

b5 = **DIV2:** *OSCIN Clock Divided by 2.* This bit controls the divide by 2 circuit which operates on the OSCIN Clock. A logical "1" value means that the OSCIN clock is internally divided by 2, and a logical "0" value means that no division of the OSCIN Clock occurs.

b4-b2 = **PRS2, PRS1, PRS0:** *Prescaling of ST9 Clock.* These bits load the prescaling module of the internal clock (INTCLK). The prescaling value selects the frequency of the ST9 clock, which can be divided by 1 to 8. Refer to the Technical Manual Clock description for more information.

b1 = **BRQEN:** *Bus Request Enable.* This bit is a software enable of an External Bus Request. When set to "1", it enables a Bus Request on the BUSREQ pin.

b0 = **HIMP:** *High Impedance Enable.* When Port 0 and/or Port 1 are programmed as multiplexed address and Data lines to interface external Program and/or Data Memory, these lines can be forced into the High Impedance state by setting the HIMP bit to "1". When this bit is reset, it has no effect on P0 and P1 lines.

If Port 1 is declared as an address and as an I/O port (example: P10 ... P14 = Address, and P15 ... P17 = I/O), the HIMP bit has no effect on the I/O lines (in this example: P15 ... P17).

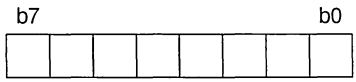
SYSTEM GROUP (Continued)

Page Pointer Register (PPR)

The ST9 Page Pointer Register is located at register R234. The top 16 registers of the Register File are paged in order that different on-chip peripheral configurations of ST9 family members will not affect the number of general purpose registers, allowing code compatibility across all devices. The ST9 can support up to 64 pages for peripheral control and status registers.

These registers are accessible via the Page Pointer Register, which is set by the Set Page Pointer Instruction (SPP). Subsequently all register access to the top group (register R240 to R255, RF0 to RFF) will refer to the selected peripheral page. Once the Page Pointer has been set, there is no need to refresh it unless a different page is required.

PPR - R234 (0EAh) Sys. Reg. Read/Write



Reset Value (undefined)	X	X	X	X	X	X	X	X

b7-b3 = **PP7-PP3:** *Page Pointer Register bits.* These bits contain the number (between 0 to 63) of the page chosen by the instruction SSP (Set Page Pointer). PP7 is the MSB of the page address. Once the page pointer has been set, there is no need to refresh it unless a different page is required.

b2-b0 = **PP2-PP0:** *Page Pointer Register bits.* These bits are fixed by hardware to zero and are not affected by any writing instruction trying to modify their value.

Note. Although the least significant two bits of PPR are hardwired to 0, using the SPP instruction will generate automatically the correct shift.

Example: If register R23 contains the value 44, the following sequence loads the third register (R242) on Page 5 with the value 44

```
spp 5
ld R242,R23
```

Page 0 is common to all ST9 devices and contains the control registers of the external interrupts, timer/watchdog, internal wait state generator and the Serial Peripheral Interface (SPI).

Figure 23. Page Pointer Configuration

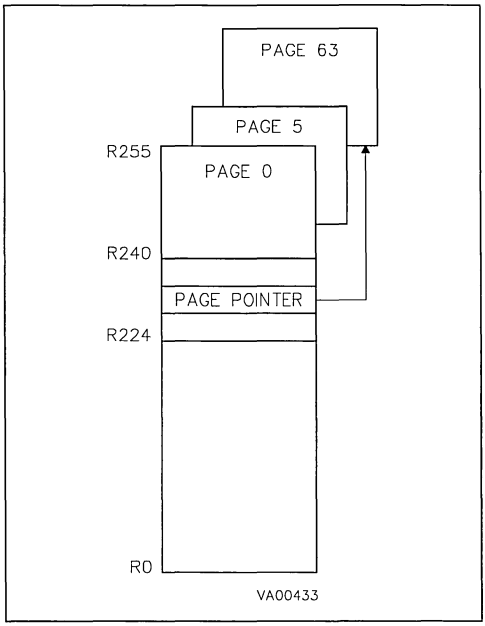


Figure 24. Page 0 Register Map

R255	Reserved
	SPI Control (SPICR)
	SPI Data (SPIDR)
	Wait Control (WCR)
	Watchdog Control (WDTCR)
	Watchdog Prescaler (WDTPR)
	Watchdog Timer Low (WDTLR)
	Watchdog Timer High (WDTLH)
	Neste Interrupt Control (NICR)
	External Interrupt Vector (EIVR)
	External Int. Priority Level (EIPLR)
	External Interrupt Mask (EIMR)
	External Interrupt Pending (EIPR)
	External Interrupt Trigger (EITR)
	EEPROM Control (EECR)
R240	Reserved

SYSTEM GROUP (Continued)

Register Pointers

Two registers, R232 and R233, are available for register pointing to allow quicker and more code efficient addressing of the Register File.

R232 (RP0) may be used as a single pointer for a 16 register working space (R233 is reserved in this case), or both R232 and R233 (RP1) may be used separately for two independent 8 register working groups. The instruction SRP, SRP0, SRP1 (the Set Register Pointer instructions) automatically inform the ST9 as to whether the Register File is to operate with a single 16-register group or two 8-register groups. There is no limitation on the order or position of these chosen register groups, other than they lie on (256 modulo 16 or 8) addresses of the register file.

Example:

```
srp #3 ; (not even modulus)
```

is equivalent to

```
srp #2 ; (2 * 256 modulo 16)
```

Figure 25. Single Register Pointer Bank

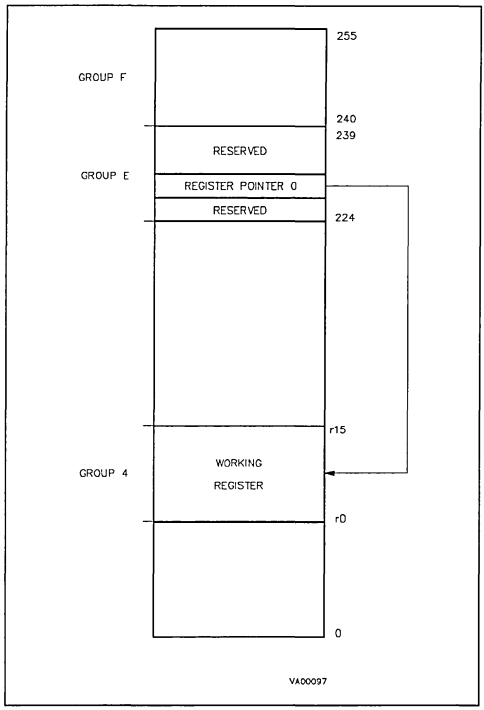
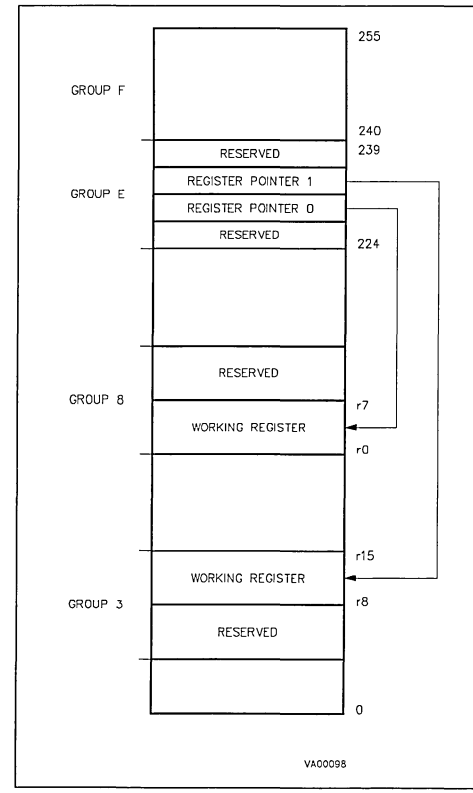


Figure 26. Dual Register Pointer Bank



The addressing of working registers involves use of the Register Pointer value plus an offset value given by the number of the addressed working register in the instruction.

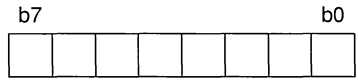
Working Registers groups of 16 registers (set by the SRP instruction) are denoted as r0 to r15 (rr0 to rr14), while the dual working register group are addressed as r0 to r7 (rr0 to rr6) for the first group of 8 registers (SRP0 instruction), and r8 to r15 (rr8 to rr15) for the second set (SRP1 instruction), although in this case the ST9 will automatically subtract 8 from the register address to give the correct offset within the second working register group.

Note. Group D can only be accessed via the Register Pointers, as the upper nibble Dh (1101b) is used in the Direct Register addressing mode to indicate the use of the working registers. It is for this reason that it is suggested that the programmer use Group D as internal stacking area (if selected).

SYSTEM GROUP (Continued)

Register Pointer 0 (RP0R)

RP0R - R232 (0E8h) Sys. Reg. Read/Write



Reset Value (undefined) X X X X X X X X

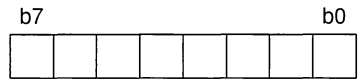
b7-b3 = **RG7-RG3:** *Register Group bits.* These bits contain the number (from 0 to 31) of the group of working registers indicated in the instruction SRP0 or SRP. When using a 16-register group, a number between 0 and 31 must be used in the SRP instruction indicating one of the two adjacent 8-register group of working registers used. RG7 is the MSB.

b2 = **RPS:** *Register Pointer Selector.* This bit is set by the instructions SRP0 and SRP1 to indicate that a double register pointing mode is used. Otherwise, the instruction SRP resets the RPS bit to zero to indicate that a single register pointing mode is used.

b1-b0 = **D1, D0.** These bits are fixed by hardware to zero and are not affected by any write instruction trying to modify their value.

Register Pointer 1 (RP1R)

RP1R - R233 (0E9h) Sys. Reg. Read/Write



Reset Value (undefined) X X X X X X X X

b7-b3 = **RG7-RG3:** *Register Group bits.* These bits contain the number (from 0 to 1) of the group of 8 working registers indicated in the instructions SRP1. Bit 7 is the MSB.

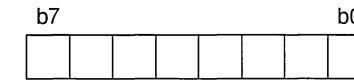
b2 = **RPS:** *Register Pointer Selector.* This bit is automatically set by the instructions SRP0 and SRP1 to indicate that a double register pointing mode is used. Otherwise the instruction SRP reset the RPS bit to zero to indicate that a single register pointing mode is used.

b1-b0 = **D1, D0.** These bits are hardware fixed to zero and are not affected by any write instruction trying to modify their value.

Flag Register (FLAGR)

The Flag Register, R231 (RE7), contains flags indicating the current status of the ST9. The ST9 Flag Register contains six bits of status information which are set or cleared by CPU operations. Four of the bits (C, Z, O and S) can be tested for use with conditional Jump instructions. Two flags (H and D) cannot be tested directly and are used for BCD arithmetic.

FLAGR - R231 (0E7h) Sys. Reg. Read/Write



Reset Value (undefined) X X X X X X X X

b7 = **C:** *Carry Flag.* When set, it indicates a carry out of the most significant bit position of the register being used as an accumulator (bit 7 for byte and bit 15 for word operations) in arithmetic operations.
The Carry Flag can be set to one by the Set Carry Flag (SCF) instruction, cleared to zero by the Reset Carry Flag (RCF) instructions, and complemented (changed to 0 if 1, and vice versa) by the Complement Carry Flag (CCF) instruction.

b6 = **Z:** *Zero Flag.* In general, the Zero Flag is set when the register being used as an accumulator is zero following an arithmetic or logical instruction.

b5 = **S:** *Sign Flag.* The Sign Flag is set to one when bit 7 or bit 15 (bit 7 for byte and bit 15 for word operations) of the register being used an accumulator contains a one (a negative number in two's complement arithmetic) following the operation specified in arithmetic, logical, Rotate or shift instructions.

b4 = **V:** *Overflow Flag.* Overflow Flag. When set, the Overflow Flag indicates that a two's complement number, in a result register, is in error, having exceeded the largest (or is less than the smallest) number that can be represented in two's complement notation.

b3 = **D:** *Decimal Adjust.* The Decimal Adjust Flag is used for BCD arithmetic. Since the algorithm for correcting BCD operations is different for addition and subtraction, this flag is used to specify which type of instruction was executed last, so that the subsequent Decimal Adjust (DA) operation can perform its function correctly. The Decimal Adjust flag cannot normally be used as a test condition by the programmer.

b2 = **H:** *Half Carry.* The Half Carry Flag is set to "1" whenever an addition generates a carry out of bit 3, or a subtraction generates a borrow into bit 3. The Half Carry flag is used by the Deci-

SYSTEM GROUP (Continued)

mal Adjust (DA) instruction to convert the binary result of a previous addition or subtraction into the correct decimal (BCD) result. As in the case of the DA flag, this flag is not normally accessed by the programmer.

b1 = **UF:** *User Flag.* The User Flag is available to the programmer, but must only be set or cleared by a Boolean instruction to prevent unwanted affects on the other flags.

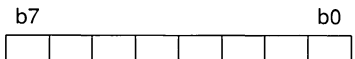
b0 = **DP :** *Data/Program Memory Flag.* This bit indicates which memory area is currently selected. Its value is affected by the Set Data Memory (SDM) and Set Program Memory (SPM) instructions. If the bit is set to "1", the ST9 is addressing the Data Memory area, when the bit is cleared to zero, the ST9 is addressing the Program area. The programmer can verify in which memory area the processor is working by inspection of this bit.

During an interrupt the Flag Register is automatically stored in the system stack area. It is recovered to its original status by the execution of the IRET instruction at the end of the interrupt service routine. This occurs for all interrupts and, when operating in nested interrupt mode, up to seven versions of the flag register may be stored.

Central Interrupt Control Register (CICR)

The Central Interrupt Control Register is at address R230 (RE6), and provides for control of the interrupt structure of the ST9, the Current priority level, the Interrupt mode selector, the Global Interrupt disable control and the Top Level Interrupt status and mask. Please refer to the Interrupt and DMA chapter of the ST9 Technical Manual for further information. In addition this register contains the Global Counter Enable of the Multi Function Timers (when available on-chip).

CICR- R230 (0E6h) Read/Write



	b7							b0
Reset Value (087h)	1	0	0	0	0	1	1	1

b7 = **GCEN:** *Global Counter Enable.* This bit is the Global Counter Enable of the 2 x 16 bit Timers of the Multifunction Timer. The GCEN bit is ANDed with the CE (Counter Enable) bit of the Timer Control Register (explained in the Multifunction Timer chapter of the Technical Manual) in order to enable the Timers when both bits are set. This bit is set after the Reset cycle.

b6 = **TLIP:** *Top Level Interrupt Pending.* This bit is automatically set when a Top Level Interrupt request is recognized. This bit can also be set by software in order to simulate a Top Level Interrupt Request.

b5 = **TLI:** *Top Level Interrupt bit.* When this bit is set, a Top Level interrupt request is acknowledged depending on the IEN bit and the TLNM bit (in Nested Interrupt Control Register). If the TLM bit is reset the top level interrupt acknowledgement depends on the TLNM alone.

b4 = **IEN:** *Enable Interrupt.* This bit, (when set), allows interrupts to be accepted. When reset no interrupts other than the TLI can be acknowledged. It is cleared by interrupt acknowledgement for concurrent mode and set by interrupt return (IRET). It can be managed by hardware and software.

b3 = **IAM:** *Interrupt Arbitration Mode.* This bit covers the selection of the two arbitration modes, the Concurrent Mode being indicated by the value "0" and the Fully Automatic Nested Mode by the value "1". This bit is under software control.

b2-b0 = **CPL2-CPL0:** *Current Priority Level bits.* These last three bits record the priority level of the interrupt presently under service (i.e. the Current Priority Level, CPL). For these priority levels 000 is the highest priority and 111 is the lowest priority. The CPL bits can be set by hardware or software and give the reference by which following interrupts are either left pending or are able to interrupt the current procedure. When the present interrupt is replaced by one of a greater priority, the current priority value is automatically stored until required, if in Nested Interrupt Mode.

I/O Port Data Registers

The data registers of I/O Port 0 to Port 5 (when available by device) are contained in registers R224 to R229 respectively. This allows direct access to these I/O ports at all times. Additional I/O Port data registers and configuration registers for all I/O Ports are to be found in the relevant Page (please refer to the specific device configuration). Each port of 8 I/O bits has three associated Control registers which determine the individual pin mode (input: TTL/CMOS thresholds; output: Open Drain/Push Pull, Bidirectional or Alternate Function). The data registers for Port 2 and Port 3 have a dual function, ST9 devices with Bank Switch Logic may also use these for the Data and Program Segment Registers.

INSTRUCTION SET

Functional Summary

The ST9 instruction set consists of 87 instruction types functionally divided into eight groups as in Table 2, they are:

- Load (two operands)
- Arithmetic & Logic (two operands)
- Arithmetic Logic and Shift (one operand)
- Stack (one operand)
- Multiply & Divide (two operands)
- Boolean (one or two operands)
- Program Control (zero to three operands)
- Miscellaneous (zero to two operands)

The wide range of instructions facilitates the full use of the register file and address spaces, reducing execution times, while the register pointers mechanism allows an unmatched code efficiency and ultrafast context switching. A particularly notable feature is the comprehensive "Any Bit, Any Register" (ABAR) addressing capability of the Boolean instructions.

The MCU can operate with a wide range of data lengths from single bit, 4-bit nibbles which can be in the form of Binary Coded Decimal (BCD) digits, 8-bit bytes and 16-bit words. The summary on Table 2 shows the instructions belonging to each group and the number of operands required for each. The source operand is "src", "dst" is the destination operand, and "cc" is a condition code.

Table 2. Instruction Set Summary

Load Instructions (Two Operands)								
Mnemonic	Operand	Instruction	Flags					
			C	Z	S	V	D	H
LD	dst, src	Load	–	–	–	–	–	–
LDW	dst, src	Load Word	–	–	–	–	–	–
Arithmetic and Logic (Two Operands)								
Mnemonic	**Operand**	**Instruction**	**Flags**					
			C	**Z**	**S**	**V**	**D**	**H**
ADD	dst, src	Add	Δ	Δ	Δ	Δ	0	Δ
ADDW	dst, src	Add Word	Δ	Δ	Δ	Δ	0	Δ
ADC	dst, src	Add with Carry	Δ	Δ	Δ	Δ	0	Δ
ADCW	dst, src	Add Word with Carry	Δ	Δ	Δ	Δ	0	Δ
SUB	dst, src	Subtract	Δ	Δ	Δ	Δ	1	Δ
SUBW	dst, src	Subtract Word	Δ	Δ	Δ	Δ	?	?
SBC	dst, src	Subtract with Carry	Δ	Δ	Δ	Δ	1	Δ
SBCW	dst, src	Subtract Word with Carry	Δ	Δ	Δ	Δ	?	?
AND	dst, src	Logical AND	–	Δ	Δ	0	–	–
ANDW	dst, src	Logical Word AND	–	Δ	Δ	0	–	–
OR	dst, src	Logical OR	–	Δ	Δ	0	–	–
ORW	dst, src	Logical Word OR	–	Δ	Δ	0	–	–

Legend: 0 = Bit set to zero
1 = Bit set to one
Δ = Bit affected
? = Bit status undefined
– = Bit not affected

INSTRUCTION SET (Continued)

Table 2. Instruction Set Summary (Continued)

Arithmetic and Logic (Two Operands) (Continued)								
Mnemonic	**Operand**	**Instruction**	**Flags**					
			C	**Z**	**S**	**V**	**D**	**H**
XOR XORW	dst, src dst, src	Logical Exclusive OR Logical Word Exclusive OR	– –	Δ Δ	Δ Δ	0 0	– –	– –
CP CPW	dst, src dst, src	Compare Compare Word	Δ Δ	Δ Δ	Δ Δ	Δ Δ	– –	– –
TM TMW	dst, src dst, src	Test Under Mask Test Word Under Mask	– –	Δ Δ	Δ Δ	0 0	– –	– –
TCM TCMW	dst, src dst, src	Test Complement Under Mask Test Word Complement Under Mask	– –	Δ Δ	Δ Δ	0 0	– –	– –
Arithmetic Logic and Shift (One Operand)								
Mnemonic	**Operand**	**Instruction**	**Flags**					
			C	**Z**	**S**	**V**	**D**	**H**
INC INCW	dst dst	Increment Increment Word	– –	Δ Δ	Δ Δ	Δ Δ	– –	– –
DEC DECW	dst dst	Decrement Decrement Word	– –	Δ Δ	Δ Δ	Δ Δ	– –	– –
SLA SLAW	dst dst	Shift Left Arithmetic Shift Word Left Arithmetic	Δ Δ	Δ Δ	Δ Δ	0 0	Δ Δ	D Δ
SRA SRAW	dst dst	Shift Right Arithmetic Shift Word Right Arithmetic	Δ Δ	Δ Δ	Δ Δ	0 0	– –	– –
RRC RRCW	dst dst	Rotate Right through Carry Rotate Word Right through Carry	Δ Δ	Δ Δ	Δ Δ	Δ Δ	– –	– –
RLC RLCW	dst dst	Rotate Left through Carry Rotate Word Left through Carry	Δ Δ	Δ Δ	Δ Δ	Δ Δ	– –	– –
ROR	dst	Rotate Right	Δ	Δ	Δ	Δ	–	–
ROL	dst	Rotate Left	Δ	Δ	Δ	Δ	–	–
CLR	dst	Clear	–	–	–	–	–	–
CPL	dst	Complement Register	–	Δ	Δ	0	–	–
SWAP	dst	Swap Nibbles	?	Δ	Δ	?	–	–
DA	dst	Decimal Adjust	Δ	Δ	Δ	?	–	–

INSTRUCTION SET (Continued)

Table 2. Instruction Set Summary (Continued)

Stack Instructions (One Operand)								
Mnemonic	**Operand**	**Instruction**	**Flags**					
			C	**Z**	**S**	**V**	**D**	**H**
PUSH PUSHW PEA	src src src	Push on System Stack Push Word on System Stack Push Effective Address on System Stack	– – –	– – –	– – –	– – –	– – –	– – –
POP POPW	dst dst	Pop from System Stack Pop Word from System Stack	– –	– –	– –	– –	– –	– –
PUSHU PUSHUW PEAU	src src src	Push on User Stack Push Word on User Stack Push Effective Address on User Stack	– – –	– – –	– – –	– – –	– – –	– – –
POPU POPUW	dst dst	Pop from User Stack Pop Word from User Stack	– –	– –	– –	– –	– –	– –
Multiply and Divide Instructions (Two Operands)								
Mnemonic	**Operand**	**Instruction**	**Flags**					
			C	**Z**	**S**	**V**	**D**	**H**
MUL	dst, src	Multiply 8x8	?	?	?	?	?	?
DIV DIVWS	dst, src	Divide 16/8 Divide Word Stepped 32/16	? ?	? ?	? ?	? ?	? ?	? ?
Boolean Instructions (Two Operands)								
Mnemonic	**Operand**	**Instruction**	**Flags**					
			C	**Z**	**S**	**V**	**D**	**H**
BLD	dst, src	Bit Load	–	–	–	–	–	–
BAND	dst, src	Bit AND	–	–	–	–	–	–
BOR	dst, src	Bit OR	–	–	–	–	–	–
BXOR	dst, src	Bit Exclusive OR	–	–	–	–	–	–
Boolean Instructions (One Operand)								
Mnemonic	**Operand**	**Instruction**	**Flags**					
			C	**Z**	**S**	**V**	**D**	**H**
BSET	dst	Bit Set	–	–	–	–	–	–
BRES	dst	Bit Reset	–	–	–	–	–	–
BCPL	dst	Bit Complement	–	–	–	–	–	–
BTSET	dst	Bit Test and Set	–	–	–	–	–	–

INSTRUCTION SET (Continued)

Table 2. Instruction Set Summary (Continued)

Program Control Instructions (Three Operands)									
Mnemonic	**Operand**	**Instruction**	**Flags**						
			C	**Z**	**S**	**V**	**D**	**H**
CPJFI	dst, src	Compare and Jump on False, Otherwise Post Increment	–	–	–	–	–	–
CPJTI	dst, src	Compare and Jump on True, Otherwise Post Increment	–	–	–	–	–	–
Program Control Instructions (Two Operands)									
Mnemonic	**Operand**	**Instruction**	**Flags**						
			C	**Z**	**S**	**V**	**D**	**H**
BTJF	dst, src	Bit Test and Jump if False	–	–	–	–	–	–
BTJT	dst, src	Bit Test and Jump if True	–	–	–	–	–	–
DJNZ	dst, src	Decrement a Working Register and Jump if Not Zero	–	–	–	–	–	–
DWJNZ	dst,src	Decrement a Register Pair and Jump if Not Zero	–	–	–	–	–	–
Program Control Instructions (One Operand)									
Mnemonic	**Operand**	**Instruction**	**Flags**						
			C	**Z**	**S**	**V**	**D**	**H**
JR	cc, dst	Jump Relative if Condition is Met	–	–	–	–	–	–
JP	cc, dst	Jump if Condition is Met	–	–	–	–	–	–
JP	dst	Unconditional Jump	–	–	–	–	–	–
CALL	dst	Unconditional Call	–	–	–	–	–	–
Program Control Instructions (No Operand)									
Mnemonic	**Operand**	**Instruction**	**Flags**						
			C	**Z**	**S**	**V**	**D**	**H**
RET		Return from Subroutine	–	–	–	–	–	–
IRET		Return from Interrupt	Δ	Δ	Δ	Δ	Δ	Δ
WFI		Stop Program Execution and Wait Next Enabled Interrupt. If a DMA request is present the CPU executes the DMA service routine and returns to WFI state.	–	–	–	–	–	–
HALT		Stop Program Execution until RESET	–	–	–	–	–	–
Miscellaneous (Two Operands)									
Mnemonic	**Operand**	**Instruction**	**Flags**						
			C	**Z**	**S**	**V**	**D**	**H**
XCH	dst, src	Exchange Registers	–	–	–	–	–	–

INSTRUCTION SET (Continued)

Table 2. Instruction Set Summary (Continued)

Miscellaneous (One Operand)								
Mnemonic	**Operand**	**Instruction**	**Flags**					
			C	**Z**	**S**	**V**	**D**	**H**
SRP	src	Set Register Pointer Long (16 Working Registers)	–	–	–	–	–	–
SRP0	src	Set Register Pointer 0 (8 LSB Working Registers)	–	–	–	–	–	–
SRP1	src	Set Register Pointer 1 (8 MSB Working Registers)	–	–	–	–	–	–
SPP	src	Set Page Pointer	–	–	–	–	–	–
EXT	src	Sign Extend	–	–	–	–	–	–
Miscellaneous (No Operand)								
Mnemonic	**Operand**	**Instruction**	**Flags**					
			C	**Z**	**S**	**V**	**D**	**H**
EI		Enable Interrupts	–	–	–	–	–	–
DI		Disable Interrupts	–	–	–	–	–	–
SCF		Set Carry Flag	Δ	–	–	–	–	–
RCF		Reset Carry Flag	Δ	–	–	–	–	–
CCF		Complement Carry Flag	Δ	–	–	–	–	–
SPM		Select Program Memory	–	–	–	–	–	–
SDM		Select Data Memory	–	–	–	–	–	–
NOP		No Operation	–	–	–	–	–	–

INSTRUCTION SET (Continued)

Processor Flags. An important aspect of any single chip microcontroller is the ability to test data and make the appropriate action based on the results. In order to provide this facility, register 231 in the Register File is used as a Flag Register. Six bits of this register are used as the following flags:

C - Carry
Z - Zero
S - Sign
V - Overflow
D - Decimal Adjust
H - Half Carry

One of the two remaining bits in the flag register is available to the user (bit 1, F1). Bit 0 is the Program/Data Memory selector bit.The flags and their description are in SYSTEM GROUP registers section.

Condition Codes. Flags C, Z, S, and V control the operation of the "conditional" Jump instructions. Table 6 shows the condition codes and the flag settings.

For example the instruction

```
JPEQ 1024
```

checks to see how the last arithmetic or logic operation has left the zero flag (Z). If the zero flag is set then the program counter is loaded with 1024 (decimal) and control is transferred to that location. Otherwise the next instruction is executed.

Note. Some of the status flags are used to indicate more than one condition, e.g. Zero and Equal. In such cases the condition codes are the same for both conditions.

Table 3. Condition Codes Summary

Mnemonic Code	Meaning	Flag Setting	Hex Value	Binary Value
F	Always False	—	0	0000
T	Always True	—	8	1000
C	Carry	C = 1	7	0111
NC	No Carry	C = 0	F	1111
Z	Zero	Z = 1	6	0110
NZ	No Zero	Z = 0	E	1110
PL	Plus	S = 0	D	1101
MI	Minus	S = 1	5	0101
OV	Overflow	V = 1	4	0100
NOV	No Overflow	V = 0	C	1100
EQ	Equal	Z = 1	6	0110
NE	Not Equal	Z = 0	E	1110
GE	Greater Then or Equal	(S xor V) = 0	9	1001
LT	Less Than	(S xor V) = 1	1	0001
GT	Greater Than	(Z or (S xor V)) = 0	A	1010
LE	Less Than or Equal	(Z or (S xor V)) = 1	2	0010
UGE	Unsigned Greater Than or Equal	C = 0	F	1111
UL	Unsigned Less Than	C = 1	7	0111
UGT	Unsigned Greater Than	(C = 0 and Z = 0) = 1	B	1011
ULE	Unsigned Less Than or Equal	(C or Z) = 1	3	0011

INSTRUCTION SET (Continued)

Notation Operands and status flags are represented by a notational shorthand in the detailed instruction description. The notation for operands (condition codes and addressing modes) and the actual operands they represent are as in Table 4.

Table 4. Notation Summary

Notation	Addressing Mode	Actual Operand/Range
cc	Condition Code	see Table 3
#N #NN	Immediate Byte Immediate Word	# data where data is a byte expression # data where data is a word expression
r	Direct Working Register	rn, where n = 0 - 15
R	Direct Register	Rn, where n = 0 - 255
rr	Direct Working Register Pair	rrn, where n is an even number in the range 0 - 14 (n = 0, 2, 4, 6....14)
RR	Direct Register Pair	RRn, where n is an even number in the range 0 - 254 (n = 0, 2, 4, 6....254)
(r)	Indirect Working Register	(rn), where n = 0 - 15
(R)	Indirect Register	(Rn), where n = 0 - 255
(r)+	Indirect Working Register Post Increment	(rn)+, where n = 0 - 15
N(r)	Indexed Register	N(rn), where n = 0 - 15; N is a one byte expression between 0 - 255
N	Memory Relative Short Address	Program label or expression in the range +127/ -128 starting from the address of the next instruction
NN	Direct Memory Long Address	Program label or expression in the range 0 - 65535 in memory area
(rr)	Indirect Pair of Working Registers Pointer	(rrn), where n is an even number in the range 0 - 14 (n = 0, 2, 4, 6....14)
(rr)+	Indirect Pair of Working Registers Pointer with Post Increment	(rrn)+, where n is an even number in the range 0 - 14 (n = 0, 2, 4, 6....14)
–(rr)	Indirect Pair of Working Registers Pointer with Pre Decrement	–(rrn), where n is an even number in the range 0 - 14 (n = 0, 2, 4, 6....14)
N(rr)	Indexed Pair of Working Registers Pointer with Short Offset	N(rrn), where n is an even number in the range 0 - 15 (n = 2, 4, 6,,....14) and N is a signed one byte expression between +127/-128
NN(rr)	Indexed Pair of Working Registers Pointer with Long Offset	NN(rr n), where n is an even number in the range 0 - 14 (n = 2, 4, 6,,....14) and NN is a word expression between 0 and 65535
N(RR)	Indexed Pair of Registers Pointer with Short Offset	N(RRn), where n is an even number in the range 0 - 254 (n = 2, 4, 6,,....254) and N is a signed one byte expression between +127/-128
NN(rr)	Indexed Pair of Registers Pointer with Long Offset	NN(RRn), where n is an even number in the range 0 - 254 (n = 2, 4, 6,,....254) and NN is a word expression between 0 and 65535
rr(rr)	Indexed Pair of Working Registers with a Pair of Working Registers used as Offset	rrn(rrx), where n and x are two even numbers in the range 0 - 14 (n = 2, 4, 6....14)

INSTRUCTION SET (Continued)

Table 4. Notation Summary (Continued)

Notation	Addressing Mode	Actual Operand/Range
r.b	Operand Inside a Direct Working Register	rn.b, where n = 0 -15 and b is a number between 0 - 7; 0 = LSB, 7 = MSB
(rr).b	Bit Pointer in a Memory Location Using a Pair of Indirect Working Registers as Address Pointer	(rrn).b, where n is an even number in the range 0 - 14 (n = 2, 4, 6....14) and b is a number between 0 - 7: 0 = LSB, 7 = MSB

Table 5. Symbols

Symbol	Meaning
dst	Destination Operand
src	Source Operand
OPC	Operation Code
XTN	Operation Code Extension
ofs	Source Offset
ofd	Destination Offset
r.b	Bit and Working Register
SSP	System Stack Pointer
USP	User Stack Pointer
PC	Program Counter
CC	Condition Code
C	Carry Flag
Z	Zero Flag
S	Sign Flag
V	Overflow Flag
D	Decimal Adjust Flag
CIC	Central Interrupt Control Register
*	Working Register Mode, 4 Bit Register value Preceded by Hex D
BTD	Destination Bit of Working Register
BTS	Source Bit of Working Register
⇐	Assignment of Result
⇔	Replaced by

ADC

Add with carry (byte) Register, Register

ADC dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst | src]	2	6	32	-	r	r
	2	6	33	-	r	(r)
[OPC] [src] [dst]	3	10	34	-	R	R
	3	10	34	-	r*	R
	3	10	34	-	R	r*
[OPC] [src] [XTN | dst]	3	10	E6	3	(r)	R
	3	10	E6	3	(r)	r*
[OPC] [XTN | src] [dst]	3	10	E7	3	R	(r)

OPERATION: dst ⇐ dst + src + C

The source byte, along with the carry flag, is added to the destination byte and the result is stored in the destination byte. The source and destination byte can be addressed either directly or indirectly.

FLAGS:
- C: Set if carry from MSB of result, otherwise cleared.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if the result is less then zero, otherwise cleared.
- V: Set if arithmetic overflow occurred, cleared otherwise.
- D: Always reset to zero.
- H: Set if carry from low-order nibble occurred.

EXAMPLE:

Instruction	HEX	Binary
ADC r8,R64	34 40 D8	0011 0100 0100 0000 1101 1000

If the carry flag is set, working register 8 contains 35 (decimal) and register 64 contains 22 (decimal), after this instruction working register 8 will contain 58.

ADC

Add with carry (byte) Register, Memory

ADC dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src	Oper
[OPC] [XTN |src,1] [dst]	3	12	72	3	R	(rr)	a
	3	12	72	3	r*	(rr)	a
	3	16	B4	3	R	(rr)+	b
	3	16	B4	3	r*	(rr)+	b
	3	16	C2	3	R	-(rr)	c
	3	16	C2	3	r*	-(rr)	c
[OPC] [ofs,1|src,0] [XTN | dst]	3	22	60	3	r	rr(rr)	a
[OPC] [XTN |src,1] [ofs] [dst]	4	24	7F	3	R	N(rr)	a
	4	24	7F	3	r*	N(rr)	a
[OPC] [XTN | dst] [src h] [src l]	4	18	C4	3	r	NN	a
[OPC] [XTN |src,0] [ofs h] [ofs l] [dst]	5	26	7F	3	R	NN(rr)	a
	5	26	7F	3	r*	NN(rr)	a

OPERATION a: dst ⇐ dst + src + C

The source byte, along with the carry flag, is added to the destination byte and the result is stored in the destination byte. The destination byte is held in the destination register. The source byte can be addressed directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst + src + C
rr ⇐ rr + 1

The source byte, along with the carry flag, is added to the destination byte and the result is stored in the destination byte. The source byte is in the memory location addressed by the source register pair, the destination byte is in the destination register. The contents of the source register pair are incremented after the ADC has been carried out.

OPERATION c: rr ⇐ rr - 1
dst ⇔ dst + src + C

The source byte, along with the carry flag, is added to the destination byte and the result is stored in the destination byte. The source byte is in the memory location addressed by the source register pair, the destination byte is in the destination register. The contents of the source register pair are decremented before the ADC is carried out.

ADC

Add with carry (byte) Register, Memory

ADC dst,src (Cont'd)

FLAGS:

C: Set if carry from MSB of result, otherwise cleared.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Always reset to zero.
H: Set if carry from low-order nibble occurred.

EXAMPLE:

Instruction	HEX	Binary
ADC r8,(rr4)+	B4 35 D8	1011 0100 0011 0101 1101 1000

If the carry flag is reset, working register 8 contains 11 (decimal), working register pair 4 contains 4200 (decimal) and memory location 4200 contains 11 (decimal), after this instruction working register 8 will contain 22 and working register pair 4 will contain 4201.

ADC

Add with carry (byte) Memory, Register

ADC dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src	Oper
[OPC] [XTN |dst,0] [src]	3	18	72	3	(rr)	R	a
	3	18	72	3	(rr)	r*	a
	3	22	B4	3	(rr)+	R	b
	3	22	B4	3	(rr)+	r*	b
	3	22	C2	3	-(rr)	R	c
	3	22	C2	3	-(rr)	r*	c
[OPC] [ofd,1|dst,1] [XTN | src]	3	24	60	3	rr(rr)	r	a
[OPC] [XTN |dst,1] [ofd]	4	26	26	3	N(rr)	R	a
[src]	4	26	26	3	N(rr)	r*	a
[OPC] [XTN | src] [dst h]	4	20	C5	3	NN	r	a
[dst l]							
[OPC] [XTN |dst,0] [ofd h]	5	28	26	3	NN(rr)	R	a
[ofd l] [src]	5	28	26	3	NN(rr)	r*	a

OPERATION a: dst ⇐ dst + src + C

The source byte, along with the carry flag, is added to the destination byte and the result is stored in the destination byte. The source byte is held in the source register. The destination byte can be addressed directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst + src + C
rr ⇔ rr + 1

The source byte, along with the carry flag, is added to the destination byte and the result is stored in the destination byte. The source byte is in the source register, the destination byte is in the memory location addressed by the destination register pair. The contents of the destination register pair are incremented after the ADC has been carried out.

OPERATION c: rr ⇐ rr - 1
dst ⇔ dst + src + C

The source byte, along with the carry flag, is added to the destination byte and the result is stored in the destination byte. The source byte is in the source register , the destination byte is in the memory location addressed by the destination register pair. The contents of the destination register pair are decremented before the ADC is carried out.

ADC

Add with carry (byte) Memory, Register

ADC dst,src (Cont'd)

FLAGS:
- C: Set if carry from MSB of result, otherwise cleared.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if the result is less then zero, otherwise cleared.
- V: Set if arithmetic overflow occurred, cleared otherwise.
- D: Always reset to zero.
- H: Set if carry from low-order nibble occurred.

EXAMPLE:

Instruction	HEX	Binary
ADC 4028,r8	C5 38 0F BC	1100 0101 0011 1000 0000 1111 1011 1100

If the carry flag is set, memory location 4028 contains 200 (decimal) and working register 8 contains 32 (decimal), after this instruction memory location 4028 will contain 233.

ADC

Add with carry (byte) Memory, Memory

ADC dst,src

INSTRUCTION FORMAT:

[OPC] [XTN |src,0] [dst,0]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	3	20	73	3	(RR)	(rr)
	3	20	73	3	(rr)*	(rr)

OPERATION: dst ⇐ dst + src + C

The source byte, along with the carry flag, is added to the destination byte and the result is stored in the destination byte. The source byte is in the memory location addressed by the source register pair, the destination byte is in the memory location addressed by the destination register pair.

FLAGS:
C: Set if carry from MSB of result, otherwise cleared.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Always reset to zero.
H: Set if carry from low-order nibble occurred.

EXAMPLE:

Instruction	HEX	Binary
ADC (rr4),(rr8)	73 38 D4	0111 0011 0010 1000 1101 0100

If the carry flag is set, working register pair 4 contains 2800 (decimal), memory location 2800 contains 46 (decimal), working register pair 8 contains 4200 (decimal) and memory location 4200 contains 45 (decimal), after this instruction memory location 2800 will contain 92.

ADC

Add with carry (byte) All, Immediate

ADC dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst] [src]	3	10	35	-	R	#N
	3	10	35	-	r*	#N
[OPC] [XTN |dst,0] [src]	3	16	F3	3	(rr)	#N
[OPC] [XTN] [src] [dst h] [dst l]	5	24	2F	31	NN	#N

OPERATION: dst ⇐ dst + src + C

The source byte, along with the carry flag, is added to the destination byte and the result is stored in the destination byte. The source byte is the immediate value in the operand, the destination byte can be in memory or in the register file.

FLAGS:
C: Set if carry from MSB of result, otherwise cleared.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Always reset to zero.
H: Set if carry from low-order nibble occurred.

EXAMPLE:

Instruction	HEX	Binary
ADC (rr8),#32	F3 38 20	1111 0011 0011 1000 0010 0000

If the carry flag is set, working register pair 8 contains 4028 (decimal) and memory location 4028 contains 74 (decimal), after this instruction memory location 4028 will contain 107.

ADCW

Add With Carry (Word) - Register, Register

ADCW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,0|src,0]	2	10	3E	-	rr	rr
[OPC] [src,0] [dst,0]	3	12	37	-	RR	RR
	3	12	37	-	rr*	RR
	3	12	37	-	RR	rr*
[OPC] [src,0] [XTN | dst]	3	14	96	3	(r)	RR
	3	14	96	3	(r)	rr*
[OPC] [XTN | src] [dst,0]	3	14	A6	3	RR	(r)
	3	14	A6	3	rr*	(r)

OPERATION: dst ⇐ dst + src + C

The source word, along with the carry flag, is added to the destination word and the result is stored in the destination word. The source and destination word can be addressed either directly or indirectly.

FLAGS:
- C: Set if carry from MSB of result, otherwise cleared.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if the result is less then zero, otherwise cleared.
- V: Set if arithmetic overflow occurred, cleared otherwise.
- D: Undefined.
- H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
ADCW (r8),RR64	96 40 38	1001 0110 0100 0000 0011 1000

If the carry flag is zero, register pair 64 contains 1102 (decimal), working register 8 contains 200 (decimal), and register pair 200 contains 2550 (decimal), after this instruction register pair 200 will hold 3652.

ADCW

Add With Carry (Word) - Register, Memory

ADCW dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [dst,0|src,1]	2	16	3E	-	rr	(rr)	a
[OPC] [XTN |src,0] [dst,0]	3	18	7E	3	RR	(rr)	a
[OPC] [XTN |src,1] [dst,0]	3	22	D5	3	RR	(rr)+	b
	3	22	D5	3	rr*	(rr)+	b
	3	24	C3	3	RR	-(rr)	c
	3	24	C3	3	rr*	-(rr)	c
[OPC] [ofs,0|src,0] [XTN |dst,0]	3	24	60	3	rr	rr(rr)	a
[OPC] [XTN |src,1] [ofs] [dst,0]	4	28	86	3	RR	N(rr)	a
	4	28	86	3	rr*	N(rr)	a
[OPC] [XTN |dst,0] [src h] [src l]	4	22	E2	3	rr	NN	a
[OPC] [XTN |src,0] [ofs h] [ofs l] [dst,0]	5	30	86	3	RR	NN(rr)	a
	5	30	86	3	rr*	NN(rr)	a

OPERATION a: dst ⇐ dst + src + C

The source word, along with the carry flag, is added to the destination word and the result is stored in the destination word. The destination word is held in the destination register. The source word can be addressed directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst + src + C
rr ⇔ rr + 2

The source word, along with the carry flag, is added to the destination word and the result is stored in the destination word. The source word is in the memory location addressed by the source register pair, the destination word is in the destination register. The contents of the source register pair are incremented after the ADD has been carried out.

OPERATION c: rr ⇐ rr - 2
dst ⇔ dst + src + C

The source word, along with the carry flag, is added to the destination word and the result is stored in the destination word. The source word is in the memory location addressed by the source register pair, the destination word is in the destination register. The contents of the source register pair are decremented before the ADD is carried out.

ADCW

Add With Carry (Word) - Register, Memory

ADCW dst,src (Cont'd)

FLAGS:

C: Set if carry from MSB of result, otherwise cleared.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Undefined.
H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
ADCW RR64,-(rr4)	C3 35 40	1100 0011 0011 0101 0100 0000

If the carry flag is set, working register pair 4 contains 1184 (decimal), register pair 64 contains 5000 (decimal) and memory pair 1182 contains 1100 (decimal), after this instruction working register pair 64 will contain 6101 and register pair 4 will contain 1182.

ADCW

Add With Carry (Word) - Memory, Register

ADCW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [dst,1|src,0]	2	30	3E	-	(rr)	rr	a
[OPC] [XTN |dst,1] [src,0]	3	32	BE	3	(rr)	RR	a
[OPC] [XTN |dst,0] [src,0]	3	34	D5	3	(rr)+	RR	b
	3	34	D5	3	(rr)+	rr*	b
	3	32	C3	3	-(rr)	RR	c
	3	32	C3	3	-(rr)	rr*	c
[OPC] [ofd,0|dst,1] [XTN |src,0]	3	36	60	3	rr(rr)	rr	a
[OPC] [XTN |dst,1] [ofd] [src,1]	4	38	86	3	N(rr)	RR	a
	4	38	86	3	N(rr)	rr*	a
[OPC] [XTN |src,1] [dst h] [dst l]	4	36	E2	3	NN	rr	a
[OPC] [XTN |dst,0] [ofd h] [. ofd l] [src,1]	5	40	86	3	NN(rr)	RR	a
	5	40	86	3	NN(rr)	rr*	a

OPERATION a: dst ⇐ dst + src + C

The source word, along with the carry flag, is added to the destination word and the result is stored in the destination word. The source word is held in the source register. The destination word can be addressed directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst + src + C
rr ⇐ rr + 2

The source word, along with the carry flag, is added to the destination word and the result is stored in the destination word. The source word is in the source register, the destination word is in the memory location addressed by the destination register pair. The contents of the destination register pair are incremented after the ADD has been carried out.

OPERATION c: rr ⇐ rr - 2
dst ⇔ dst + src + C

The source word, along with the carry flag, is added to the destination word and the result is stored in the destination word. The source word is in the source register , the destination word is in the memory location addressed by the destination register pair. The contents of the destination register pair are decremented before the ADD is carried out.

ADCW

Add With Carry (Word) - Memory, Register

ADCW dst,src (Cont'd)

FLAGS:

C: Set if carry from MSB of result, otherwise cleared.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Undefined.
H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
ADCW (rr4)+,RR64	D5 34 40	1101 0101 0011 0100 0100 0000

If the carry flag is set, register pair 64 contains 1250 (decimal), working register pair 4 contains 1064 (decimal), and memory pair 1064 contains 1750, after this instruction is carried out memory pair 1064 will contain 3001 and working register pair 4 will contain 1066.

ADCW

Add With Carry (Word) - Memory, Memory

ADCW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,1|src,1]	2	34	3E	-	(rr)	(rr)

OPERATION: dst ⇐ dst + src + C

The source word, along with the carry flag, is added to the destination word and the result is stored in the destination word. The source word is in the memory location addressed by the source register pair, the destination word is in the memory location addressed by the destination register pair.

FLAGS:
- C: Set if carry from MSB of result, otherwise cleared.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if the result is less then zero, otherwise cleared.
- V: Set if arithmetic overflow occurred, cleared otherwise.
- D: Undefined.
- H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
ADCW (rr4),(rr6)	3E 57	0011 1110 0101 0111

If the carry flag is set, working register pair 6 contains 1002 (decimal), memory pair 1002 contains 2300 (decimal), working register pair 4 contains 1060 (decimal) and memory pair 1060 contains 2700 (decimal), after this instruction memory pair 1060 will contain 5001.

ADCW

Add With Carry (Word) - All, Immediate

ADCW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src
[OPC] [dst,1] [src h] [src l]	4	14	37	-	RR	#NN
	4	14	37	-	rr*	#NN
[OPC] [XTN |dst,0] [src h] [src l]	4	34	BE	3	(rr)	#NN
[OPC] [XTN |dst,1] [ofd] [src h] [src l]	5	38	06	3	N(rr)	#NN
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src h] [src l]	6	40	06	3	NN(rr)	#NN
[OPC] [XTN] [src h] [src l] [dst h] [dst l]	6	40	36	31	NN	#NN

OPERATION: dst ⇐ dst + src + C

The source word, along with the carry flag, is added to the destination word and the result is stored in the destination word. The source word is the immediate value in the operand, the destination word can be in memory or in the register file.

FLAGS:

C: Set if carry from MSB of result, otherwise cleared.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Undefined.
H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
ADCW RR64,#4268	37 41 10 AC	0011 0111 0100 0001 0001 0000 1010 1100

If the carry flag is zero and register pair 64 contains 2000 (decimal), after this instruction has been carried out register pair 64 will contain the decimal value 6268.

ADD

Add (byte) Register, Register

ADD dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst | src]	2	6	42	-	r	r
	2	6	43	-	r	(r)
[OPC] [src] [dst]	3	10	44	-	R	R
	3	10	44	-	r*	R
	3	10	44	-	R	r*
[OPC] [src] [XTN | dst]	3	10	E6	4	(r)	R
	3	10	E6	4	(r)	r*
[OPC] [XTN | src] [dst]	3	10	E7	4	R	(r)

OPERATION: dst ⇐ dst + src

The source byte is added to the destination byte and the result is stored in the destination byte. The source and destination byte can be addressed either directly or indirectly.

FLAGS:
- C: Set if carry from MSB of result, otherwise cleared.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if the result is less then zero, otherwise cleared.
- V: Set if arithmetic overflow occurred, cleared otherwise.
- D: Always reset to zero.
- H: Set if carry from low-order nibble occurred.

EXAMPLE:

Instruction	HEX	Binary
ADD (r8),R255	E6 FF 48	1110 0110 1111 1111 0100 1000

If working register 8 contains 28 (decimal), register 28 contains 43 (decimal) and register 255 contains 21 (decimal) after this instruction register 28 will contain 64.

ADD

Add (byte) Register, Memory

ADD dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src	Oper
[OPC] [XTN |src,1] [dst]	3	12	72	4	R	(rr)	a
	3	12	72	4	r*	(rr)	a
	3	16	B4	4	R	(rr)+	b
	3	16	B4	4	r*	(rr)+	b
	3	16	C2	4	R	-(rr)	c
	3	16	C2	4	r*	-(rr)	c
[OPC] [ofs,1|src,0] [XTN | dst]	3	22	60	4	r	rr(rr)	a
[OPC] [XTN |src,1] [ofs] [dst]	4	24	7F	4	R	N(rr)	a
	4	24	7F	4	r*	N(rr)	a
[OPC] [XTN | dst] [src h] [src l]	4	18	C4	4	r	NN	a
[OPC] [XTN |src,0] [ofs h] [ofs l] [dst]	5	26	7F	4	R	NN(rr)	a
	5	26	7F	4	r*	NN(rr)	a

OPERATION a: dst ⇐ dst + src

The source byte is added to the destination byte and the result is stored in the destination byte. The destination byte is held in the destination register. The source byte can be addressed directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst + src
rr ⇔ rr + 1

The source byte is added to the destination byte and the result is stored in the destination byte. The source byte is in the memory location addressed by the source register pair, the destination byte is in the destination register. The contents of the source register pair are incremented after the ADD has been carried out.

OPERATION c: rr ⇐ rr - 1
dst ⇔ dst + src

The source byte is added to the destination byte and the result is stored in the destination byte. The source byte is in the memory location addressed by the source register pair, the destination byte is in the destination register. The contents of the source register pair are decremented before the ADD is carried out.

ADD

Add (byte) Register, Memory

ADD dst,src (Cont'd)

FLAGS:

C: Set if carry from MSB of result, otherwise cleared.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Always reset to zero.
H: Set if carry from low-order nibble occurred.

EXAMPLE:

Instruction	HEX	Binary
ADD R32,-(rr4)	C2 45 20	1100 0010 0100 0101 0010 0000

If register 32 contains 207 (decimal), working register pair 4 contains 4200 (decimal) and memory location 4199 contains 27 (decimal), after this instruction register 32 will contain 234 and working register pair 4 will contain 4199.

ADD

Add (byte) Memory, Register

ADD dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [XTN |dst,0] [src]	3	18	72	4	(rr)	R	a
	3	18	72	4	(rr)	r*	a
	3	22	B4	4	(rr)+	R	b
	3	22	B4	4	(rr)+	r*	b
	3	22	C2	4	-(rr)	R	c
	3	22	C2	4	-(rr)	r*	c
[OPC] [ofd,1|dst,1] [XTN | src]	3	24	60	4	rr(rr)	r	a
[OPC] [XTN |dst,1] [ofd]	4	26	26	4	N(rr)	R	a
[src]	4	26	26	4	N(rr)	r*	a
[OPC] [XTN | src] [dst h]	4	20	C5	4	NN	r	R
[dst l]							
[OPC] [XTN |dst,0] [ofd h]	5	28	26	4	NN(rr)	R	r*
[ofd l] [src]	5	28	26	4	NN(rr)	r*	a

OPERATION a: dst ⇐ dst + src

The source byte is added to the destination byte and the result is stored in the destination byte. The source byte is held in the source register. The destination byte can be addressed directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst + src
rr ⇔ rr + 1

The source byte is added to the destination byte and the result is stored in the destination byte. The source byte is in the source register, the destination byte is in the memory location addressed by the destination register pair. The contents of the destination register pair are incremented after the ADD has been carried out.

OPERATION c: rr ⇐ rr - 1
dst ⇔ dst + src

The source byte is added to the destination byte and the result is stored in the destination byte. The source byte is in the source register , the destination byte is in the memory location addressed by the destination register pair. The contents of the destination register pair are decremented before the ADD is carried out.

ADD

Add (byte) Memory, Register

ADD dst,src (Cont'd)

FLAGS:

C: Set if carry from MSB of result, otherwise cleared.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Always reset to zero.
H: Set if carry from low-order nibble occurred.

EXAMPLE:

Instruction	HEX	Binary
ADD 6(rr8),R255	26 49 06 FF	0010 0110 0100 1001 0000 0110 1111 1111

If working register pair 8 contains 4028 (decimal), memory location 4034 contains 110 (decimal) and register 255 contains 100 (decimal), after this instruction memory location 4034 will contain 210.

ADD

Add (byte) Memory, Memory

ADD dst,src

INSTRUCTION FORMAT:

[OPC] [XTN |src,0] [dst,0]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	3	20	73	4	(RR)	(rr)
	3	20	73	4	(rr)*	(rr)

OPERATION: dst ⇐ dst + src

The source byte is added to the destination byte and the result is stored in the destination byte. The source byte is in the memory location addressed by the source register pair, the destination byte is in the memory location addressed by the destination register pair.

FLAGS:

C: Set if carry from MSB of result, otherwise cleared.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Always reset to zero.
H: Set if carry from low-order nibble occurred.

EXAMPLE:

Instruction	HEX	Binary
ADD (rr4),(rr8)	73 48 D4	0111 0011 0100 1000 1101 0100

If working register pair 4 contains 2800 (decimal) and memory location 2800 contains 46 (decimal), working register pair 8 contains 4200 (decimal) and memory location 4200 contains 45 (decimal), after this instruction memory location 2800 will contain 91.

ADD

Add (byte) All, Immediate

ADD dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst] [src]	3	10	45	-	R	#N
	3	10	45	-	r*	#N
[OPC] [XTN |dst,0] [src]	3	16	F3	4	(rr)	#N
[OPC] [XTN] [src] [dst h] [dst l]	5	24	2F	41	NN	#N

OPERATION: dst ⇐ dst + src

The source byte is added to the destination byte and the result is stored in the destination byte. The source byte is the immediate value in the operand, the destination byte can be in memory or in the register file.

FLAGS:
- C: Set if carry from MSB of result, otherwise cleared.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if the result is less then zero, otherwise cleared.
- V: Set if arithmetic overflow occurred, cleared otherwise.
- D: Always reset to zero.
- H: Set if carry from low-order nibble occurred.

EXAMPLE:

Instruction	HEX	Binary
ADD (rr8),#32	F3 48 20	1111 0011 0100 1000 0010 0000

If working register pair 8 contains 4028 (decimal) and memory location 4028 contains 74 (decimal), after this instruction memory location 4028 will contain 106.

ADDW

Add (Word) - Register, Register

ADDW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,0|src,0]	2	10	4E	-	rr	rr
[OPC] [src,0] [dst,0]	3	12	47	-	RR	RR
	3	12	47	-	rr*	RR
	3	12	47	-	RR	rr*
[OPC] [src,0] [XTN | dst]	3	14	96	4	(r)	RR
	3	14	96	4	(r)	rr*
[OPC] [XTN | src] [dst,0]	3	14	A6	4	RR	(r)
	3	14	A6	4	rr*	(r)

OPERATION: dst ⇐ dst + src

The source word is added to the destination word and the result is stored in the destination word. The source and destination words, held in register pairs, can be addressed either directly or indirectly.

FLAGS:
- C: Set if carry from MSB of result, otherwise cleared.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if the result is less then zero, otherwise cleared.
- V: Set if arithmetic overflow occurred, cleared otherwise.
- D: Undefined.
- H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
ADDW RR64,(r8)	A6 48 40	1010 0110 0100 1000 0100 0000

If working register 8 contains 124, register pair 124 contains 1300 (decimal) and register pair 64 contains 800 (decimal) after this instruction register pair 64 will contain 2100.

ADDW

Add (Word) - Register, Memory

ADDW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [dst,0|src,1]	2	16	4E	-	rr	(rr)	a
[OPC] [XTN |src,0] [dst,0]	3	18	7E	4	RR	(rr)	a
[OPC] [XTN |src,1] [dst,0]	3	22	D5	4	RR	(rr)+	b
	3	22	D5	4	rr*	(rr)+	b
	3	24	C3	4	RR	-(rr)	c
	3	24	C3	4	rr*	-(rr)	c
[OPC] [ofs,0|src,0] [XTN |dst,0]	3	24	60	4	rr	rr(rr)	a
[OPC] [XTN |src,1] [ofs] [dst,0]	4	28	86	4	RR	N(rr)	a
	4	28	86	4	rr*	N(rr)	a
[OPC] [XTN |dst,0] [src h] [src l]	4	22	E2	4	rr	NN	a
[OPC] [XTN |src,0] [ofs h] [ofs l] [dst,0]	5	30	86	4	RR	NN(rr)	a
	5	30	86	4	rr*	NN(rr)	a

OPERATION a: dst ⇐ dst + src

The source word is added to the destination word and the result is stored in the destination word. The destination word is held in the destination register. The source word can be addressed directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst + src
rr ⇔ rr + 2

The source word is added to the destination word and the result is stored in the destination word. The source word is in the memory location addressed by the source register pair, the destination word is in the destination register. The contents of the source register pair are incremented after the add has been carried out.

OPERATION c: rr ⇐ rr - 2
dst ⇔ dst + src

The source word is added to the destination word and the result is stored in the destination word. The source word is in the memory location addressed by the source register pair, the destination word is in the destination register. The contents of the source register pair are decremented before the add is carried out.

ADDW

Add (Word) - Register, Memory

ADDW dst,src (Cont'd)

FLAGS:
- C: Set if carry from MSB of result, otherwise cleared.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if the result is less then zero, otherwise cleared.
- V: Set if arithmetic overflow occurred, cleared otherwise.
- D: Undefined.
- H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
ADDW RR64,(rr8)	7E 48 40	0111 1110 0100 1000 0100 0000

If working register pair 8 contains 1240 (decimal), memory pair 1240 contains 3000 (decimal) and register pair 64 contains 1000 (decimal), after this instruction working register pair 64 will contain 4000.

ADDW

Add (Word) - Memory, Register

ADDW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src	Oper
[OPC] [dst,1|src,0]	2	30	4E	-	(rr)	rr	a
[OPC] [XTN |dst,1] [src,0]	3	32	BE	4	(rr)	RR	a
[OPC] [XTN |dst,0] [src,0]	3	34	D5	4	(rr)+	RR	b
	3	34	D5	4	(rr)+	rr*	b
	3	32	C3	4	-(rr)	RR	c
	3	32	C3	4	-(rr)	rr*	c
[OPC] [ofd,0|dst,1] [XTN |src,0]	3	36	60	4	rr(rr)	rr	a
[OPC] [XTN |dst,1] [ofd] [src,1]	4	38	86	4	N(rr)	RR	a
	4	38	86	4	N(rr)	rr*	a
[OPC] [XTN |src,1] [dst h] [dst l]	4	36	E2	4	NN	rr	a
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src,1]	5	40	86	4	NN(rr)	RR	a
	5	40	86	4	NN(rr)	rr*	a

OPERATION a: dst ⇐ dst + src

The source word is added to the destination word and the result is stored in the destination word. The source word is held in the source register. The destination word can be addressed directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst + src
rr ⇔ rr + 2

The source word is added to the destination word and the result is stored in the destination word. The source word is in the source register, the destination word is in the memory location addressed by the destination register pair. The contents of the destination register pair are incremented after the ADD has been carried out.

OPERATION c: rr ⇐ rr - 2
dst ⇔ dst + src

The source word is added to the destination word and the result is stored in the destination word. The source word is in the source register , the destination word is in the memory location addressed by the destination register pair. The contents of the destination register pair are decremented before the ADD is carried out.

ADDW

Add (Word) - Memory, Register

ADDW dst,src (Cont'd)

FLAGS:

C: Set if carry from MSB of result, otherwise cleared.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Undefined.
H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
ADDW (rr4)+,RR64	D5 44 40	1101 0101 0100 0100 0100 0000

If register pair 64 contains 1250 (decimal), working register pair 4 contains 1064 (decimal), and memory pair 1064 contains 1750, after this instruction is carried out memory pair 1064 will contain 3000 and working register pair 4 will contain 1066.

ADDW

Add (Word) - Memory, Memory

ADDW dst,src

INSTRUCTION FORMAT:

[OPC] [dst,1|src,1]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src
	2	34	4E	-	(rr)	(rr)

OPERATION: dst ⇐ dst + src

The source word is added to the destination word and the result is stored in the destination word. The source word is in the memory location addressed by the source register pair, the destination word is in the memory location addressed by the destination register pair.

FLAGS:

C: Set if carry from MSB of result, otherwise cleared.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Undefined.
H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
ADDW (rr4),(rr6)	4E 57	0100 1110 0101 0111

If working register pair 6 contains 1002 (decimal), memory pair 1002 contains 2300 (decimal), working register pair 4 contains 1060 (decimal) and memory pair 1060 contains 2700 (decimal), after this instruction memory pair 1060 will contain 5000.

ADDW

Add (Word) - All, Immediate

ADDW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,1] [src h] [src l]	4	14	47	-	RR	#NN
	4	14	47	-	rr*	#NN
[OPC] [XTN |dst,0] [src h] [src l]	4	34	BE	4	(rr)	#NN
[OPC] [XTN |dst,1] [ofd] [src h] [src l]	5	38	06	4	N(rr)	#NN
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src h] [src l]	6	40	06	4	NN(rr)	#NN
[OPC] [XTN] [src h] [src l] [dst h] [dst l]	6	40	36	41	NN	#NN

OPERATION: dst ⇐ dst + src

The source word is added to the destination word and the result is stored in the destination word. The source word is the immediate value in the operand, the destination word can be in memory or in the register file.

FLAGS:

C: Set if carry from MSB of result, otherwise cleared.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Undefined.
H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
ADDW RR64,#4268	47 41 10 AC	0100 0111 0100 0001 0001 0000 1010 1100

If register pair 64 contains 2000 (decimal), after this instruction has been carried out register pair 64 will contain the decimal value 6268.

AND

AND (byte) Register, Register

AND dst,src

INSTRUCTION FORMAT:

[OPC] [dst | src]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src
	2	6	12	-	r	r
	2	6	13	-	r	(r)
[OPC] [src] [dst]	3	10	14	-	R	R
	3	10	14	-	r*	R
	3	10	14	-	R	r*
[OPC] [src] [XTN | dst]	3	10	E6	1	(r)	R
	3	10	E6	1	(r)	r*
[OPC] [XTN | src] [dst]	3	10	E7	1	R	(r)

OPERATION: dst ⇐ dst AND src

The contents of the source are ANDed with the destination byte and the results stored in the destination byte. The contents of the source are not affected.

FLAGS:
C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 7 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
AND r8,R64	14 40 D8	0001 0100 0100 0000 1101 1000

If working register 8 contains 11001100 and register 64 contains 10000101, after this instruction working register 8 will contain 10000100.

AND

AND (byte) Register, Memory

AND dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [XTN |src,1] [dst]	3	12	72	1	R	(rr)	a
	3	12	72	1	r*	(rr)	a
	3	16	B4	1	R	(rr)+	b
	3	16	B4	1	r*	(rr)+	b
	3	16	C2	1	R	-(rr)	c
	3	16	C2	1	r*	-(rr)	c
[OPC] [ofs,1|src,0] [XTN | dst]	3	22	60	1	r	rr(rr)	a
[OPC] [XTN |src,1] [ofs] [dst]	4	24	7F	1	R	N(rr)	a
	4	24	7F	1	r*	N(rr)	a
[OPC] [XTN | dst] [src h] [src l]	4	18	C4	1	r	NN	a
[OPC] [XTN |src,0] [ofs h] [ofs l] [dst]	5	26	7F	1	R	NN(rr)	a
	5	26	7F	1	r*	NN(rr)	a

OPERATION a: dst ⇐ dst AND src

The source byte is ANDed with the destination byte and the result stored in the destination byte. The destination register is addressed directly, the memory location is addressed either directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst AND src
rr ⇐ rr + 1

The contents of the memory location addressed by the source register pair are ANDed with the contents of the directly addressed destination register the result stored in the destination byte. The contents of the source register pair are incremented after the AND has been carried out.

OPERATION c: rr ⇐ rr - 1
dst ⇐ dst AND src

The contents of the source register pair are decremented and then the contents of the memory location addressed by the source register pair are ANDed with the contents of the directly addressed destination register. The result is stored in the destination byte.

FLAGS:
- C: Unaffected.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if result bit 7 is set, otherwise cleared.
- V: Always reset to zero.
- D: Unaffected.
- H: Unaffected.

AND

AND (byte) Register, Memory

AND dst,src (Cont'd)

EXAMPLE:

Instruction	HEX	Binary
AND r8,4028	C4 18 0F BC	1100 0100 0001 1000 0000 1111 1011 1100

If working register 8 contains 11001100 and memory location 4028 contains 10000101, after this instruction working register 8 will contain 10000100.

AND

AND (byte) Memory, Register

AND dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src	Oper
[OPC] [XTN |dst,0] [src]	3	18	72	1	(rr)	R	a
	3	18	72	1	(rr)	r*	a
	3	22	B4	1	(rr)+	R	b
	3	22	B4	1	(rr)+	r*	b
	3	22	C2	1	-(rr)	R	c
	3	22	C2	1	-(rr)	r*	c
[OPC] [ofd,1|dst,1] [XTN | src]	3	24	60	1	rr(rr)	r	a
[OPC] [XTN |dst,1] [ofd] [src]	4	26	26	1	N(rr)	R	a
	4	26	26	1	N(rr)	r*	a
[OPC] [XTN | src] [dst h] [dst l]	4	20	C5	1	NN	r	a
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src]	5	28	26	1	NN(rr)	R	a
	5	28	26	1	NN(rr)	r*	a

OPERATION a: dst ⇐ dst AND src

The source byte is ANDed with the destination byte and the result stored in the destination byte. The source registers are addressed directly, the memory location are addressed either directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst AND src
rr ⇐ rr + 1

The contents of the memory location addressed by the destination register pair (destination byte) are ANDed with the contents of the directly addressed source register the result stored in the destination byte. The contents of the destination register pair are incremented after the AND has been carried out.

OPERATION c: rr ⇐ rr - 1
dst ⇐ dst AND src

The contents of the destination register pair are decremented and then the contents of the memory location addressed by the destination register pair (destination byte) are ANDed with the contents of the directly addressed source register. The result is stored in the destination byte.

FLAGS:
C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 7 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

AND

AND (byte) Memory, Register

AND dst,src (Cont'd)

EXAMPLE:

Instruction	HEX	Binary
AND 4028,r8	C5 18 0F BC	1100 0101 0001 1000 0000 1111 1011 1100

If working register 8 contains 11001100 and memory location 4028 contains 10000101, after this instruction memory location 4028 will contain 10000100.

AND

AND (byte) Memory, Memory

AND dst,src

INSTRUCTION FORMAT:

[OPC]	[XTN |src,0]	[dst,0]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
							dst	src
			3	20	73	1	(RR)	(rr)
			3	20	73	1	(rr)*	(rr)

OPERATION: dst ⇐ dst AND src

The contents of the memory location addressed by the source register pair are ANDed with the content of the memory location addressed by the destination register pair. The source and destination addresses are for the word high order byte.

FLAGS:
- C: Unaffected.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if result bit 7 is set, otherwise cleared.
- V: Always reset to zero
- D: Unaffected.
- H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
AND (rr4),(rr8)	73 18 D4	0111 0011 0001 1000 1101 0100

If working register pair 4 contains 2800 (decimal), memory location 2800 contains 11001100, working register pair 8 contains 4200 (decimal) and memory location 4200 contains 11000011, after this instruction memory location 2800 will contain 11000000.

AND

AND (byte) All, Immediate

AND dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src
[OPC] [dst] [src]	3	10	15	-	R	#N
	3	10	15	-	r*	#N
[OPC] [XTN |dst,0] [src]	3	16	F3	1	(rr)	#N
[OPC] [XTN] [src] [dst h] [dst l]	5	24	2F	11	NN	#N

OPERATION: dst ⇐ dst AND src

The value #N is ANDed with the content of the destination register or memory location (destination byte) and stored in the destination byte.

FLAGS:
C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 7 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
AND (rr8),#32	F3 18 20	1111 0011 0001 1000 0010 0000

If working register pair 8 contains 4028 (decimal) and memory location 4028 contains 11101100, after this instruction memory location 4028 will contain 00100000.

ANDW

AND (Word) - Register, Register

ANDW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,0|src,0]	2	10	1E	-	rr	rr
[OPC] [src,0] [dst,0]	3	12	17	-	RR	RR
	3	12	17	-	rr*	RR
	3	12	17	-	RR	rr*
[OPC] [src,0] [XTN | dst]	3	14	96	1	(r)	RR
	3	14	96	1	(r)	rr*
[OPC] [XTN | src] [dst,0]	3	14	A6	1	RR	(r)
	3	14	A6	1	rr*	(r)

OPERATION: dst ⇐ dst AND src

The source word is ANDed with the destination word. The result is left in the destination. The source and destination words can be addressed either directly or indirectly.

FLAGS:
- C: Unaffected.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if result bit 15 is set, otherwise cleared.
- V: Always reset to zero.
- D: Undefined.
- H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
ANDW RR32,RR64	17 40 20	0001 0111 0100 0000 0010 0000

If register pair 64 contains 11001100/11001100B and register pair 32 contains 10101010/10101010B, after this instruction register pair 32 will contain 10001000/10001000B.

ANDW

AND (Word) - Register, Memory

ANDW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [dst,0|src,1]	2	16	1E	-	rr	(rr)	a
[OPC] [XTN |src,0] [dst,0]	3	18	7E	1	RR	(rr)	a
[OPC] [XTN |src,1] [dst,0]	3	22	D5	1	RR	(rr)+	b
	3	22	D5	1	rr*	(rr)+	b
	3	24	C3	1	RR	-(rr)	c
	3	24	C3	1	rr*	-(rr)	c
[OPC] [ofs,0|src,0] [XTN |dst,0]	3	24	60	1	rr	rr(rr)	a
[OPC] [XTN |src,1] [ofs] [dst,0]	4	28	86	1	RR	N(rr)	a
	4	28	86	1	rr*	N(rr)	a
[OPC] [XTN |dst,0] [src h] [src l]	4	22	E2	1	rr	NN	a
[OPC] [XTN |src,0] [ofs h] [ofs l] [dst,0]	5	30	86	1	RR	NN(rr)	a
	5	30	86	1	rr*	NN(rr)	a

OPERATION a: dst ⇐ dst AND src

The source word is ANDed with the destination word. The result is left in the destination word. The destination word is held in the destination register. The source word can be addressed directly, indirectly or by indexing. The contents of the source are not affected.

OPERATION b: dst ⇐ dst AND src
rr ⇐ rr + 2

The source word is ANDed with the destination word. The result is left in the destination word. The source word is in the memory location addressed by the source register pair, the destination word is in the destination register. The contents of the source register pair are incremented after the AND has been carried out.

OPERATION c: rr ⇐ rr - 2
dst ⇐ dst AND src

The source word is ANDed with the destination word. The result is left in the destination word. The source word is in the memory location addressed by the source register pair, the destination word is in the destination register. The contents of the source register pair are decremented before the AND is carried out.

ANDW

AND (Word) - Register, Memory

ANDW dst,src (Cont'd)

FLAGS:
- C: Unaffected.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if result bit 15 is set, otherwise cleared.
- V: Always reset to zero.
- D: Undefined.
- H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
ANDW RR64,(rr4)+	D5 15 40	1101 0101 0001 0101 0100 0000

If working register pair 4 contains 1184 (decimal), register pair 64 contains 10101010/10101010B and memory pair 1184 contains 11001100/11001100B, after this instruction register pair 64 will contain 10001000/10001000B and register pair 4 will contain 1186.

ANDW

AND (Word) - Memory, Register

ANDW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src	Oper
[OPC] [dst,1|src,0]	2	30	1E	-	(rr)	rr	a
[OPC] [XTN |dst,1] [src,0]	3	32	BE	1	(rr)	RR	a
[OPC] [XTN |dst,0] [src,0]	3	34	D5	1	(rr)+	RR	b
	3	34	D5	1	(rr)+	rr*	b
	3	32	C3	1	-(rr)	RR	c
	3	32	C3	1	-(rr)	rr*	c
[OPC] [ofd,0|dst,1] [XTN |src,0]	3	36	60	1	rr(rr)	rr	a
[OPC] [XTN |dst,1] [ofd] [src,1]	4	38	86	1	N(rr)	RR	a
	4	38	86	1	N(rr)	rr*	a
[OPC] [XTN |src,1] [dst h] [dst l]	4	36	E2	1	NN	rr	a
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src,1]	5	40	86	1	NN(rr)	RR	a
	5	40	86	1	NN(rr)	rr*	a

OPERATION a: dst ⇐ dst AND src

The source word is ANDed with the destination word. The result is stored in the destination word. The source word is held in the source register pair. The destination word can be addressed directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst AND src
rr ⇐ rr + 2

The source word is ANDed with the destination word. The result is stored in the destination word. The source word is in the source register pair, the destination word is in the memory location addressed by the destination register pair. The contents of the destination register pair pair are incremented after the AND has been carried out.

OPERATION c: rr ⇐ rr - 2
dst ⇐ dst AND src

The source word is ANDed with the destination word. The result is stored in the destination word. The source word is in the source register pair , the destination word is in the memory pair addressed by the destination register pair. The contents of the destination register pair pair are decremented before the AND is carried out.

FLAGS:
C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 15 is set, otherwise cleared.
V: Always reset to zero.
D: Undefined.
H: Undefined.

ANDW

AND (Word) - Memory, Register

ANDW dst,src (Cont'd)

EXAMPLE:

Instruction	HEX	Binary
ANDW (rr8),RR64	BE 19 40	1011 1110 0001 1001 0100 0000

If register pair 64 contains 11001100/11001100B, working register pair 8 contains 2000 (decimal) and memory pair 2000 contains 10101010/10101010B, after this instruction memory pair 2000 will hold 10001000/10001000B.

ANDW

AND (Word) - Memory, Memory

ANDW dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,1|src,1]	2	34	1E	-	(rr)	(rr)

OPERATION: dst ⇐ dst AND src

The source word is ANDed with the destination word. The result is stored in the destination word. The source word is in the memory pair addressed by the source register pair, the destination word is in the memory pair addressed by the destination register pair.

FLAGS:
- C: Unaffected.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if result bit 15 is set, otherwise cleared.
- V: Always reset to zero.
- D: Undefined.
- H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
ANDW (rr4),(rr6)	1E 57	0001 1110 0101 0111

If working register pair 6 contains register 1002 (decimal), memory pair 1002 contains 11001100/11001100B, working register pair 4 contains 1060 (decimal), and memory pair 1060 contains 10101010/10101010B, after this instruction memory pair 1060 will contain 10001000/10001000B.

ANDW

AND (Word) - All, Immediate

ANDW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src
[OPC] [dst,1] [src h] [src l]	4	14	17	-	RR	#NN
	4	14	17	-	rr*	#NN
[OPC] [XTN |dst,0] [src h] [src l]	4	34	BE	1	(rr)	#NN
[OPC] [XTN |dst,1] [ofd] [src h] [src l]	5	38	06	1	N(rr)	#NN
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src h] [src l]	6	40	06	1	NN(rr)	#NN
[OPC] [XTN] [src h] [src l] [dst h] [dst l]	6	40	36	11	NN	#NN

OPERATION: dst ⇐ dst AND src

The source word is ANDed with the destination word. The result is stored in the destination word. The source word is the immediate value in the operand, the destination word can be in memory or in the register file.

FLAGS:
- C: Unaffected.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if result bit 15 is set, otherwise cleared.
- V: Always reset to zero.
- D: Undefined.
- H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
ANDW RR64,#52428	17 41 CC CC	0001 0111 0100 0001 1100 1100 1100 1100

If register pair 64 contains 10101010/10101010B, after this instruction has been carried out register pair 64 will contain 10001000/10001000B.

BAND

Bit AND

BAND dst.b,src.b

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [btd,1| dst] [bts,1| src]	3	14	1F	-	r.b	r.b	a
[OPC] [btd,1| dst] [bts,0| src]	3	14	1F	-	r.b	r.!b	b

OPERATION a: dst bit ⇐ dst bit AND src bit

The selected bit in the source working register is ANDed with the selected bit of the destination working register and the result left in the destination bit. All other bits in the destination register remain unaffected. Both source and destination are directly addressed.

OPERATION b: dst bit ⇐ dst bit AND NOT src bit

The complement of the selected bit in the source working register is ANDed with the selected bit of the destination working register and the result left in the destination bit. All other bits in the destination register remain unaffected. Both source and destination are directly addressed.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
BAND r4.5,r8.2	1F B4 58	0001 1111 1011 0100 0101 1000

If bit 2 of working register 8 is 0 and bit 5 of working register 4 is 1, after this instruction bit 5 of working register 4 will be 0.

NOTE: A bit AND can use the same or different nibbles of the same register as both source and destination.

BCPL

Bit Complement

BCPL dst.b

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [btd,0| dst]	2	6	6F	-	r.b	-

OPERATION: dst bit ⇐ NOT dst bit

The selected bit in the destination working register is set to its own complement. All other bits in the destination register remain unaffected. The destination is directly addressed.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
BCPL r4.5	6F A4	0110 1111 1010 0100

If bit 5 of working register 4 was 1, after this instruction it will be 0.

BLD

Bit Load

BLD dst.b,src.b

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [bts,1| src] [btd,0| dst]	3	14	F2	-	r.b	r.b	a
[OPC] [bts,1| src] [btd,1| dst]	3	14	F2	-	r.b	r.b	b

OPERATION a: dst bit ⇐ src bit

The selected bit in the source working register is loaded into the selected bit of the destination working register. All other bits in the destination register remain unaffected. Both source and destination are directly addressed.

OPERATION b: dst bit ⇐ NOT src bit

The complement of the selected bit in the source working register is loaded into the selected bit of the destination working register. All other bits in the destination register remain unaffected. Both source and destination are directly addressed.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
BLD r4.5,r8.!2	F2 58 B4	1111 0010 0101 1000 1011 0100

If bit 2 of working register 8 is 1, after this instruction bit 5 of working register 4 will be 0.

NOTE: A bit load can use the same or different nibbles of the same register as both source and destination.

BOR

Bit OR

BOR dst.b,src.b

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [btd,1| dst] [bts,0| src]	3	14	0F	-	r.b	r.b	a
[OPC] [btd,1| dst] [bts,1| src]	3	14	0F	-	r.b	r.b	b

OPERATION a: dst bit ⇐ dst bit OR src bit

The selected bit in the source working register is ORed with the selected bit of the destination register and the result left in the destination bit. All other bits in the destination register remain unaffected. Both source and destination are directly addressed.

OPERATION b: dst bit ⇐ dst bit OR NOT src bit

The complement of the selected bit in the source working register is ORed with the selected bit of the destination working register and the result left in the destination bit. All other bits in the destination register remain unaffected. Both source and destination are directly addressed.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
BOR r4.5,r8.2	0F B4 58	0000 1111 1011 0100 0101 1000

If bit 2 of working register 8 is 1 and bit 5 of working register 4 is 0, after this instruction bit 5 of working register 4 will be 1.

NOTE: A bit OR can use the same or different nibbles of the same register as both source and destination.

BRES

Bit Reset

BRES dst.b

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [btd,0| dst]	2	6	1F	-	r.b	-

OPERATION: dst bit ⇐ 0

The selected bit in the destination working register is reset to 0. All other bits in the destination register remain unaffected. The destination is directly addressed.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
BRES r4.5	1F A4	0001 1111 1010 0100

After this instruction bit 5 of working register 4 will be 0.

BSET

Bit Set

BSET dst.b

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [btd,0| dst]	2	6	0F	-	r.b	-

OPERATION: dst bit ⇐ 1

The selected bit in the destination working register is set to 1. All other bits in the destination register remain unaffected. The destination is directly addressed.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
BSET r4.5	0F A4	0000 1111 1010 0100

After this instruction bit 5 of working register 4 will be 1.

BTJF

Bit Test And Jump If False

BTJF dst,src

INSTRUCTION FORMAT:

[OPC]	[btd,1| dst]	[src]	No. Bytes	No.Cycl		OPC (HEX)	OPC XTN	Addr Mode	
				No Jmp	Jmp			dst	src
			3	14	16	AF	-	r.b	N

OPERATION: If tested bit is zero then PC ⇐ PC + N where -128>N>127

The specified bit in the destination working register is tested and if it is found to be equal to zero, the source value N is added to the program counter and control passes to the statement whose address is now in the PC. If the tested bit is one, the instruction following BTJF is executed. N is a relative value in the range +127/-128.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
BTJF r10.2,-40	AF 5A D8	1010 1111 0101 1010 1101 1000

If bit 2 of working register 10 is zero and the program counter holds 200, after this instruction the program counter will jump to address 160.

BTJT

Bit Test And Jump If True

BTJT dst,src

INSTRUCTION FORMAT:

[OPC] [btd,0| dst] [src]	No. Bytes	No. Cycl		OPC (HEX)	OPC XTN	Addr Mode	
		No Jmp	Jmp			dst	src
	3	14	16	AF	-	r.b	N

OPERATION: If tested bit is one then PC ⇐ PC + N where -128>N>127

The specified bit in the destination working register is tested and if it is found to be equal to one, the source value N is added to the program counter and control passes to the statement whose address is now in the PC. If the tested bit is zero the instruction following BTJF is executed. N is a relative value in the range +127/-128.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
BTJF r10.2,+40	AF 2A 21	1010 1111 0100 1010 0010 1000

If bit 2 of working register 10 is a one and the program counter holds 200, after this instruction the program counter will jump to address 240.

BTSET

Bit Test and Set

BTSET dst.b

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [btd,0| dst]	2	8	F2	-	r.b	-	
	2	20	F6	-	(rr).b	-	

OPERATION: If dst.b = 0 then dst.b ⇐ 1

The selected bit in the destination is tested; if zero it is set to one and the zero flag set. The destination is addressed either by working register direct or memory indirect.

FLAGS:
C: Unaffected.
Z: Set if bit was zero, otherwise cleared.
S: Set if bit 7 is tested and set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
N: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
BTSET r4.5	F2 A4	1111 0010 1010 0100

If bit 5 of working register 4 is 0, after this instruction it is set to 1 and the zero flag is also set to 1.

BXOR

Bit XOR

BXOR dst.b,src.b

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [btd,1| dst] [bts,0| src]	3	14	6F	-	r.b	r.b	a
[OPC] [btd,1| dst] [bts,1| src]	3	14	6F	-	r.b	r.b	b

OPERATION a: dst bit ⇐ dst bit XOR src bit

The selected bit in the source working register is XORed with the selected bit of the destination register and the result left in the destination bit. All other bits in the destination register remain unaffected. Both source and destination are directly addressed.

OPERATION b: dst bit ⇐ dst bit XOR NOT src bit

The complement of the selected bit in the source working register is XORed with the selected bit of the destination working register and the result left in the destination bit. All other bits in the destination register remain unaffected. Both source and destination are directly addressed.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
BXOR r4.5,r8.2	6F B4 48	0110 1111 1011 0100 0100 1000

If bit 2 of working register 8 is 1 and bit 5 of working register 4 is 0, after this instruction bit 5 of working register 4 will be 1.

NOTE: A bit XOR can use the same or different nibbles of the same register as both source and destination.

CALL

Unconditional Call Subroutine

CALL dst

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,1]	2	16	74	-	(RR)	-
	2	16	74	-	(rr)*	-
[OPC] [dst h] [dst l]	3	18	D2	-	NN	-

OPERATION:

SSP ⇐ SSP - 2
(SP) ⇐ PC
PC ⇐ dst

The current contents of the program counter (PC) are pushed onto the top of the system stack. (The program counter value used is the address of the first instruction byte following the CALL instruction). The specified destination address is then loaded into the PC and points to the first instruction of the CALL procedure.

In direct memory addressing mode the destination is in the memory location addressed by the absolute value in the operand.

In the indirect memory addressing mode the destination is in the memory location addressed by the contents of the destination register pair.

FLAGS:

No flags affected.

EXAMPLE:

Instruction	HEX	Binary
CALL 3521H	D2 35 21	1101 0010 0011 0101 0010 0001

If the content of the program counter is 1A47 (hex) and the content of the system stack pointer is 3002 (hex) the above instruction will cause the stack pointer to be decremented to 3000 (hex), 1A4A (the address following the instruction) is stored in external data memory 3000 (hex) and 3001 (hex), and the program counter is loaded with 3521 (hex). The program counter now points to the address of the first statement in the procedure to be executed.

CCF

Complement Carry Flag

CCF

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC]	1	6	61	-	-	-

OPERATION: C ⇐ NOT C

The carry flag is complemented; if C=1 it is changed to C=0 and vice-versa.

FLAGS: C: Complemented.
No other flags affected.

EXAMPLE:

Instruction	HEX	Binary
CCF	61	0110 0001

If the carry flag is set to one, after this instruction it will be reset to zero.

CLR

Clear Register

CLR dst

INSTRUCTION FORMAT:

[OPC] [dst]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	2	6	90	-	R	-
	2	6	90	-	r*	-
	2	6	91	-	(R)	-
	2	6	91	-	(r)*	-

OPERATION: dst ⇐ 0

The contents of destination register, directly or indirectly addressed, is cleared to zero.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
CLR (R32)	91 20	1001 0001 0010 0000

If register 32 holds 142, after this instruction register 142 will contain 0.

CP

Compare (byte) Register, Register

CP dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst | src]	2	6	92	-	r	r
	2	6	93	-	r	(r)
[OPC] [src] [dst]	3	10	94	-	R	R
	3	10	94	-	r*	R
	3	10	94	-	R	r*
[OPC] [src] [XTN | dst]	3	10	E6	9	(r)	R
	3	10	E6	9	(r)	r*
[OPC] [XTN | src] [dst]	3	10	E7	9	R	(r)

OPERATION: dst - src

The source byte is compared with (subtracted from) the destination byte and the zero flag set if result is zero. The destination byte remains unaffected. The source and destination byte can be addressed either directly or indirectly.

FLAGS:
- C: Cleared if carry from MSB, otherwise set indicating a borrow.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if the result is less then zero, otherwise cleared.
- V: Set if arithmetic overflow occurred, cleared otherwise.
- D: Unaffected.
- H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
CP (r8),R255	E6 FF 98	1110 0110 1111 1111 1001 1000

If working register 8 contains 28 (decimal), register 28 contains 11001100 and register 255 contains 10000101, after this instruction the zero flag will be reset to zero.

CP

Compare (byte) Register, Memory

CP dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [XTN |src,1] [dst]	3	12	72	9	R	(rr)	a
	3	12	72	9	r*	(rr)	a
	3	16	B4	9	R	(rr)+	b
	3	16	B4	9	r*	(rr)+	b
	3	16	C2	9	R	-(rr)	c
	3	16	C2	9	r*	-(rr)	c
[OPC] [ofs,1|src,0] [XTN | dst]	3	22	60	9	r	rr(rr)	a
[OPC] [XTN |src,1] [ofs] [dst]	4	24	7F	9	R	N(rr)	a
	4	24	7F	9	r*	N(rr)	a
[OPC] [XTN | dst] [src h] [src l]	4	18	C4	9	r	NN	a
[OPC] [XTN |src,0] [ofs h] [ofs l] [dst]	5	26	7F	9	R	NN(rr)	a
	5	26	7F	9	r*	NN(rr)	a

OPERATION a: dst - src

The source byte is compared with (subtracted from) the destination byte and the zero flag set if result is zero. The destination byte remains unaffected. The destination byte is held in the destination register. The source byte can be addressed directly, indirectly or by indexing.

OPERATION b: dst - src
rr ⇐ rr + 1

The source byte is compared with (subtracted from) the destination byte and the zero flag set if result is zero. The destination byte remains unaffected. The source byte is in the memory location addressed by the source register pair, the destination byte is in the destination register. The contents of the source register pair are incremented after the CP has been carried out.

OPERATION c: rr ⇐ rr - 1
dst - src

The source byte is compared with (subtracted from) the destination byte and the zero flag set if result is zero. The destination byte remains unaffected. The source byte is in the memory location addressed by the source register pair, the destination byte is in the destination register. The contents of the source register pair are decremented before the CP is carried out.

CP

Compare (byte) Register, Memory

CP dst,src (Cont'd)

FLAGS:

C: Cleared if carry from MSB, otherwise set indicating a borrow.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
CP R32,-(rr4)	C2 95 20	1100 0010 1001 0101 0010 0000

If register 32 contains 11001100, working register pair 4 contains 4200 (decimal) and memory location 4199 contains 11001100, after this instruction the zero flag will be set to one and working register pair 4 will contain 4199.

CP

Compare (byte) Memory, Register

CP dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src	Oper
[OPC] [XTN |dst,0] [src]	3	18	72	9	(rr)	R	a
	3	18	72	9	(rr)	r*	a
	3	22	B4	9	(rr)+	R	b
	3	22	B4	9	(rr)+	r*	b
	3	22	C2	9	-(rr)	R	c
	3	22	C2	9	-(rr)	r*	c
[OPC] [ofd,1|dst,1] [XTN | src]	3	24	60	9	rr(rr)	r	a
[OPC] [XTN |dst,1] [ofd] [src]	4	26	26	9	N(rr)	R	a
	4	26	26	9	N(rr)	r*	a
[OPC] [XTN | src] [dst h] [dst l]	4	20	C5	9	NN	r	a
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src]	5	28	26	9	NN(rr)	R	a
	5	28	26	9	NN(rr)	r*	a

OPERATION a: dst - src

The source byte is compared with (subtracted from) the destination byte and the zero flag set if result is zero. The destination byte remains unaffected. The source byte is held in the source register. The destination byte can be addressed directly, indirectly or by indexing.

OPERATION b: dst - src
rr ⇐ rr + 1

The source byte is compared with (subtracted from) the destination byte and the zero flag set if result is zero. The destination byte remains unaffected. The source byte is in the source register, the destination byte is in the memory location addressed by the destination register pair. The contents of the destination register pair are incremented after the CP has been carried out.

OPERATION c: rr ⇐ rr - 1
dst - src

The source byte is compared with (subtracted from) the destination byte and the zero flag set if result is zero. The destination byte remains unaffected. The source byte is in the source register , the destination byte is in the memory location addressed by the destination register pair. The contents of the destination register pair are decremented before the CP is carried out.

CP

Compare (byte) Memory, Register

CP dst,src (Cont'd)

FLAGS:

C: Cleared if carry from MSB, otherwise set indicating a borrow.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
CP 4028,R8	C5 98 0F BC	1100 0101 1001 1000 0000 1111 1011 1100

If memory location 4028 contains 11001100 and working register 8 contains 10000101, after this instruction the zero flag will be reset to zero.

CP

Compare (byte) Memory, Memory

CP dst,src

INSTRUCTION FORMAT:

[OPC] [XTN |src,0] [dst,0]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	3	18	73	9	(RR)	(rr)
	3	18	73	9	(rr)*	(rr)

OPERATION: dst - src

The source byte is compared with (subtracted from) the destination byte and the zero flag set if result is zero. The destination byte remains unaffected. The source byte is in the memory location addressed by the source register pair, the destination byte is in the memory location addressed by the destination register pair.

FLAGS:

C: Cleared if carry from MSB, otherwise set indicating a borrow.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
CP (rr4),(rr8)	73 98 D4	0111 0011 1001 1000 1101 0100

If working register pair 4 contains 2800 (decimal), memory location 2800 contains 11001100, working register pair 8 contains 4200 (decimal) and memory location 4200 contains 11001100, after this instruction the zero flag will be set to one.

CP

Compare (byte) All, Immediate

CP dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src
[OPC] [dst] [src]	3	10	95	-	R	#N
	3	10	95	-	r*	#N
[OPC] [XTN |dst,0] [src]	3	16	F3	9	(rr)	#N
[OPC] [XTN] [src] [dst h] [dst l]	5	22	2F	91	NN	#N

OPERATION: dst - src

The source byte is compared with (subtracted from) the destination byte and the zero flag set if result is zero. The destination byte remains unaffected. The source byte is the immediate value in the operand, the destination byte can be in memory or in the register file.

FLAGS:

C: Cleared if carry from MSB, otherwise set indicating a borrow.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
CP (rr8),#32	F3 28 20	1111 0011 0010 1000 0010 0000

If working register pair 8 contains 4028 (decimal) and memory location 4028 contains 11101100, after this instruction the zero flag will be reset to zero.

CPJFI

Compare And Jump If False Otherwise Post-Increment

CPJFI dst,src,N

INSTRUCTION FORMAT:

[OPC] [src,0| dst] [PC Offset]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		PC offs.
					dst	src	
	3	22	24	9F	r	(rr)	N

OPERATION: If compare not verified jump, otherwise increment source register pair.

The source operand is compared to (subtracted from) the destination operand. If the result is different from zero the offset N (where N is in the range -128/+127) is added to the program counter and control passes to the statement whose address is now in the PC, otherwise the source pointer is incremented by one and the instruction following the CPJFI is executed.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
CPJFI r2,(rr14),+100	9F E2 64	1001 1111 1110 0010 0110 0100

If the current value of the program counter is 340 (decimal) and working register 2 contains 11001100B, working register pair 14 contains 3000 (decimal) and memory location 3000 holds 10000100B the program counter will now point at program location 440 (decimal).

NOTE : The source value must exist within the destination area (or limit checks must be included).

CPJTI

Compare And Jump If True Otherwise Post-Increment

CPJTI dst,src,N

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		PC offs.
					dst	src	
[OPC] [src,1| dst] [PC Offset]	3	22	24	9F	r	(rr)	N

OPERATION: If compare verified jump, otherwise increment source registers pair.

The source operand is compared to (subctracted from) the destination operand. If the result is zero the offset N (where N is in the range -128/+127) is added to the program counter and control passes to the statement whose address is now in the PC, otherwise the source pointer is incremented by one and the instruction following the CPJTI is executed.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
CPJTI r2,(rr14),+100	9F F2 64	1001 1111 1111 0010 0110 0100

If the current value of the program counter is 340 (decimal) and working register 2 contains 11001100B, working register pair 14 contains 3000 (decimal) and memory location 3000 holds 11001100B the program counter will now point at program location 440 (decimal).

NOTE : The source value must exist within the destination area (or limit checks must be included).

CPL

Complement Register

CPL dst

INSTRUCTION FORMAT:

[OPC] [dst]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	2	6	80	-	R	-
	2	6	80	-	r*	-
	2	6	81	-	(R)	-
	2	6	81	-	(r)*	-

OPERATION: dst ⇐ NOT dst

The contents of the destination register, directly or indirectly addressed, are ones complemented (1 becomes 0 and 0 becomes 1).

FLAGS:
- C: Unaffected.
- Z: Set if result is zero, otherwise cleared.
- S: Set if result bit 7 is set, otherwise cleared.
- V: Always reset to zero.
- D: Unaffected.
- H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
CPL R32	81 20	1000 0001 0010 0000

If register 32 contains 142 and register 142 holds 10101010B, after this instruction the contents of register 142 become 01010101B.

CPW

Compare (Word) - Register, Register

CPW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,0|src,0]	2	10	9E	-	rr	rr
[OPC] [src,0] [dst,0]	3	12	97	-	RR	RR
	3	12	97	-	rr*	RR
	3	12	97	-	RR	rr*
[OPC] [src,0] [XTN | dst]	3	14	96	9	(r)	RR
	3	14	96	9	(r)	rr*
[OPC] [XTN | src] [dst,0]	3	14	A6	9	RR	(r)
	3	14	A6	9	rr*	(r)

OPERATION: dst - src

The source word is compared with (subtracted from) the destination word and the appropriate flags set. The destination remains unaltered. The source and destination word can be addressed either directly or indirectly.

FLAGS:
- C: Cleared if carry from MSB of result, otherwise set.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if the result is less then zero, otherwise cleared.
- V: Set if arithmetic overflow occurred, cleared otherwise.
- D: Unaffected.
- H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
CPW (r8),RR64	96 40 98	1001 0110 0100 0000 1001 1000

If register pair 64 contains 11001100/11001100B, working register 8 contains 200 (decimal) and register pair 200 contains 01001000/01001000B, after this instruction the zero flag will be reset.

CPW

Compare (Word) - Register, Memory

CPW dst,src

INSTRUCTION FORMAT:

				No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
								dst	src	
[OPC]	[dst,0|src,1]			2	16	9E	-	rr	(rr)	a
[OPC]	[XTN |src,0]	[dst,0]		3	18	7E	9	RR	(rr)	a
[OPC]	[XTN |src,1]	[dst,0]		3	22	D5	9	RR	(rr)+	b
				3	22	D5	9	rr*	(rr)+	b
				3	24	C3	9	RR	-(rr)	c
				3	24	C3	9	rr*	-(rr)	c
[OPC]	[ofs,0|src,0]	[XTN |dst,0]		3	24	60	9	rr	rr(rr)	a
[OPC]	[XTN |src,1]	[ofs]		4	28	86	9	RR	N(rr)	a
[dst,0]				4	28	86	9	rr*	N(rr)	a
[OPC]	[XTN |dst,0]	[src h]		4	22	E2	9	rr	NN	a
[src 1]										
[OPC]	[XTN |src,0]	[ofs h]		5	30	86	9	RR	NN(rr)	a
[ofs 1]	[dst,0]			5	30	86	9	rr*	NN(rr)	a

OPERATION a: dst - src

The source word is compared with (subtracted from) the destination word and the appropriate flags set. The destination remains unaltered. The destination word is held in the destination register. The source word can be addressed directly, indirectly or by indexing.

OPERATION b: dst - src
rr ⇐ rr + 2

The source word is compared with (subtracted from) the destination word and appropriate flags set. The destination remains unaltered. The source word is in the memory location addressed by the source register pair, the destination word is in the destination register. The contents of the source register pair are incremented after the compare has been carried out.

OPERATION c: rr ⇐ rr - 2
dst ⇐ dst - src

The source word is compared with (subtracted from) the destination word and the appropriate flags set. The destination remains unaltered. The source word is in the memory location addressed by the source register pair, the destination word is in the destination register. The contents of the source register pair are decremented before the compare is carried out.

CPW

Compare (Word) - Register, Memory

CPW dst,src (Cont'd)

FLAGS:

C: Cleared if carry from MSB of result, otherwise set.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
CPW RR64,-(rr4)	C3 95 40	1100 0011 1001 0101 0100 0000

If working register pair 4 contains 1184 (decimal), register pair 64 contains 11001100/11001100B and memory pair 1182 contains 11001100/11001100B, after this instruction has been carried out the zero flag will be set and register pair 4 will contain 1182.

CPW

Compare (Word) - Memory, Register

CPW dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [dst,1|src,0]	2	26	9E	-	(rr)	rr	a
[OPC] [XTN |dst,1] [src,0]	3	28	BE	9	(rr)	RR	a
[OPC] [XTN |dst,0] [src,0]	3	30	D5	9	(rr)+	RR	b
	3	30	D5	9	(rr)+	rr*	b
	3	30	C3	9	-(rr)	RR	c
	3	30	C3	9	-(rr)	rr*	c
[OPC] [ofd,0|dst,1] [XTN |src,0]	3	32	60	9	rr(rr)	rr	a
[OPC] [XTN |dst,1] [ofd] [src,1]	4	34	86	9	N(rr)	RR	a
	4	34	86	9	N(rr)	rr*	a
[OPC] [XTN |src,1] [dst h] [dst l]	4	30	E2	9	NN	rr	a
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src,1]	5	36	86	9	NN(rr)	RR	a
	5	36	86	9	NN(rr)	rr*	a

OPERATION a: dst - src

The source word is compared with (subtracted from) the destination word and the appropriate flags set. the destination remains unaltered. The source word is held in the source register. The destination word can be addressed directly, indirectly or by indexing.

OPERATION b: dst - src
rr ⇐ rr + 2

The source word is compared with (subtracted from) the destination word and the appropriate flags set. The destination remains unaltered. The source word is in the source register, the destination word is in the memory location addressed by the destination register pair. The contents of the destination register pair are incremented after the compare has been carried out.

OPERATION c: rr ⇐ rr - 2
dst - src

The source word is compared with (subtracted from) the destination word and the appropriate flags set. The destination remains unaltered. The source word is in the source register , the destination word is in the memory location addressed by the destination register pair. The contents of the destination register pair are decremented before the compare is carried out.

CPW

Compare (Word) - Memory, Register

CPW dst,src (Cont'd)

FLAGS:

C: Cleared if carry from MSB of result, otherwise set.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
CPW (rr4)+,RR64	D5 94 40	1101 0101 1001 0100 0100 0000

If register pair 64 contains 11001100/11001100B, working register pair 4 contains 1064 (decimal) and memory pair 1064 contains 01001000/01001000B, after this instruction has been carried out the zero flag will be reset and working register pair 4 will contain 1066.

CPW

Compare (Word) - Memory, Memory

CPW dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,1|src,1]	2	30	9E	-	(rr)	(rr)

OPERATION: dst - src

The source word is compared with (subtracted from) the destination word and the appropriate flags set. The destination remains unaltered. The source word is in the memory location addressed by the source register pair, the destination word is in the memory location addressed by the destination register pair.

FLAGS:
- C: Cleared if carry from MSB of result, otherwise set.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if the result is less then zero, otherwise cleared.
- V: Set if arithmetic overflow occurred, cleared otherwise.
- D: Unaffected.
- H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
CPW (rr4),(rr6)	9E 57	1001 1110 0101 0111

If working register pair 6 contains 1002 (decimal), memory pair 1002 contains 11001100/11001100B, working register pair 4 contains 1060 (decimal) and memory pair 1060 contains 11001100/11001100B, after this instruction the zero flag will be set.

CPW

Compare (word) - All, Immediate

CPW dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,1] [src h] [src l]	4	14	97	-	RR	#NN
	4	14	97	-	rr*	#NN
[OPC] [XTN |dst,0] [src h] [src l]	4	30	BE	9	(rr)	#NN
[OPC] [XTN |dst,1] [ofd] [src h] [src l]	5	34	06	9	N(rr)	#NN
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src h] [src l]	6	36	06	9	NN(rr)	#NN
[OPC] [XTN] [src h] [src l] [dst h] [dst l]	6	36	36	91	NN	#NN

OPERATION: dst - src

The source word is compared with (subtracted from) the destination word and the appropriate flags set. The destination remains unaltered. The source word is the immediate value in the operand, the destination word can be in memory or in the register file.

FLAGS:
- C: Cleared if carry from MSB of result, otherwise set.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if the result is less then zero, otherwise cleared.
- V: Set if arithmetic overflow occurred, cleared otherwise.
- D: Unaffected.
- H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
CPW RR64,#52428	97 41 CC CC	1001 0111 0100 0001 1100 1100 1100 1100

If register pair 64 contains 01001000/01001000B, after this instruction has been carried out the zero flag will be reset to zero.

DA

Decimal Adjust

DA dst

INSTRUCTION FORMAT:

[OPC] [src]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	2	6	70	-	R	-
	2	6	70	-	r*	-
	2	6	71	-	(R)	-
	2	6	71	-	(r)*	-

OPERATION: dst ⇐ DA dst

After an addition (ADD, ADC) or subtraction (SUB, SBC), this instruction adds a number, determined by the binary result of the previous arithmetic operation, in order to convert the contents of the destination register into two 4-bit BCD digits. The following table indicates the operation performed:

Instruction	Carry before DA	Bits 4-7 value (Hex)	H Flag before DA	Bits 0-3 value (Hex)	Number added to byte	Carry after DA
	0	0-9	0	0-9	00	0
	0	0-8	0	A-F	06	0
	0	0-9	1	0-3	06	0
ADD	0	A-F	0	0-9	60	1
ADC	0	9-F	0	A-F	66	1
	0	A-F	1	0-3	66	1
	1	0-2	0	0-9	60	1
	1	0-2	0	A-F	66	1
	0	0-9	0	0-9	00	0
SUB	0	0-8	1	6-F	FA	0
SBC	1	7-F	0	0-9	A0	1
	1	6-F	1	6-F	9A	1

DA

Decimal Adjust

DA dst (Cont'd)

FLAGS:

C: Set if carry from MSB, otherwise cleared.(see table above)
Z: Set if result is zero, otherwise cleared.
S: Set if result bit 7 is set, otherwise clered.
V: Undefined.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
DA R32	70 20	0111 0000 0010 0000

If addition is performed using the BCD values 15 and 27, the result should be 42. The sum is incorrect, however, in the destination location when using standard binary arithmetic.

```
0001    0101                          0011    1100
0010    0111                          0000    0110
-----   -----                         -----   -----
0011    1100   =3CH                   0100    0010   =42H
```

The DA statement adjusts this result so that the correct BCD representation is obtained.

DEC

Decrement Register

DEC dst

INSTRUCTION FORMAT:

[OPC] [dst]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	2	6	40	-	R	-
	2	6	40	-	r*	-
	2	6	41	-	(R)	-
	2	6	41	-	(r)*	-

OPERATION: dst ⇐ dst - 1

The content of destination register, directly or indirectly addressed, is decremented by 1.

FLAGS:
C: Unaffected.
Z: Set if result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
DEC (r2)	41 D2	0100 0001 1101 0010

If working register 2 holds 122 and register 122 contains 100 (decimal), after this instruction is executed register 122 will contain 99.

DECW

Decrement Word Register

DECW dst

INSTRUCTION FORMAT:

[OPC] [dst,0]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	2	8	CF	-	RR	-
	2	8	CF	-	rr*	-

OPERATION: dst ⇐ dst - 1

The destination register content is decremented by 1.

FLAGS:
- C: Unaffected.
- Z: Set if result is zero, otherwise cleared.
- S: Set if result is negative, otherwise cleared.
- V: Set if arithmetic overflow occured, otherwise cleared.
- D: Unaffected.
- H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
DECW rr2	CF D2	1100 1111 1101 0010

If working register pair 2 holds 2000 (decimal), after this instruction is executed it will contain 1999 (decimal).

DI

Disable Interrupts

DI

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC]	1	6	10	-	-	-

OPERATION: CIC.4 ⇐ 0

Bit 4 of the Central Interrupt Control register (R230) is reset to zero. All interrupts except NMI are then disabled.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
DI	10	0001 0000

After this instruction all interrupts (except NMI) are disabled.

NOTE: The NMI (Not Maskable Interrupt) can be disabled only with a general chip reset.

DIV

Divide (16/8)

DIV dst,src

INSTRUCTION FORMAT:

[OPC] [dst,0| src]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	2	28/20	5F	-	rr	r

OPERATION: dst/src: dst (low) ⇐- result
dst (high) ⇐- remainder

The contents of the destination register pair are divided by the contents of the source register. The result is left in the destination register low byte and the remainder in the destination register high byte. This operation takes 28 clock cycles.
If the dividend high byte is greather than the divider, this operation takes 20 clock cycles.

```
Input    rr dst high (even address) = dividend high
         rr dst low (odd address)   = dividend low
         rr src                     = divisor

Output   rr dst high                = remainder
         rr dst low                 = result

The src byte holds the unmodified divisor.
```

FLAGS:
C: Set to one.
Z: Set if result is zero, otherwise reset.
S: Set if reminder is zero, otherwise reset.
V: Undefined.
D: Always set to one.
H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
DIV rr8,r6	5F 68	0101 1111 0110 1000

If working register 6 contains 30 (decimal) and working register pair 8 contains 500 (decimal), after this instruction working register 7 will contain 16 (decimal) and working register 8 will contain 20 (decimal).

DIV

Divide (16/8)

DIV dst,src (Cont'd)

NOTE 1: If the dividend high is greater than the divisor the instruction is not carried out, the carry flag is reset to zero and the decimal adjust flag is set to one. All other flags are unaffected. This control control takes 20 clock cycles and both destination and source register remain unmodified.

NOTE 2: If the divisor is zero, a trap is generated simulating an interrupt. The current Program Counter and flag register are saved on the system stack and then the PC is set to the contents of memory locations 0002 and 0003 of the Program memory which contains the Divide-by-zero trap vector. This trap procedure takes 38 clock cycles.

Location 0002 ⇒ Interrupt Vector Pointer High
Location 0003 ⇒ Interrupt Vector Pointer Low

The "divide by zero attempted" subroutine should be written by the user.

DIVWS

Divide Word Stepped (32/16)

DIVWS dsth, dstl, src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		
					dst	src	src
[OPC] [src,0] [dsh,0|dsl,0]	3	28	56	-	rr	rr	RR

OPERATION:

When executed 16 times and then followed by a RLCW on the destination low working register pair, this instruction carries out a 32 bit by 16 bit divide and leaves the result in the destination low working register pair and the remainder in the destination high working register pair. No automatic controls are carried out on the relationship between divisor and dividend before this instruction is carried out, nor is the divisor checked for zero, these should be supplied by the user.

FLAGS: All undefined.

EXAMPLE:

Instruction	HEX	Binary
DIVWS rr6,rr8,RR10	56 0A 68	0101 0110 0000 1010 0110 1000

Working register pair 6 will contain the 16 high order bits of the dividend, working register pair 8 will contain the 16 low order bits of the dividend and register pair 10 will contain the 16 bit divisor. After this instruction working register pair 8 will contain the result and working register pair 6 the remainder. See subroutine example.

NOTE: A typical example of a subroutine using the DIVWS instruction is shown below.

DIVWS

Divide Word Stepped (32/16)

DIVWS dsth, dstl, src (Cont'd)

DIVSTEP SUBROUTINE EXAMPLE

This subroutine first checks that divisor is less than the dividend and that the divisor is greater than zero before carrying out the division.

```
d_len   = r9
dvsr    = RR10
dvd_hi  = rr6
dvd_low = rr8
;
;inputs:   RR10 = 16 bit divisor
;          rr6  = 32 bit dividend high
;          rr8  = 32 bit dividend low
;outputs:  RR10 = unmodified divisor
;          rr6  = remainder
;          rr8  = result
;
DIVSTEP:
       cpw dvd_hi,dvsr      ;check that dividend  high divisor
       jrnc Out             ;if not leave subroutine
       cpw dvsr,#0000h      ;check divisor zero
       jrnz Defloop         ;if true start divide
Out:   ret
Defloop:
       pushu d_len          ;set 16 bit step divide loop
       ld d_len,#16

Loop:  divws dvd_hi,dvd_low,dvsr
                            ;carry out divws
       djnz d_len,Loop      ;16 times
       rlcw dvd_low
       popu d_len
       ret
```

DJNZ

Decrement And Jump If Not Zero

DJNZ dst,N

INSTRUCTION FORMAT:

[dst | OPC] [PC Offset]	No. Bytes	No. Cycl		OPC (HEX)	OPC XTN	Addr Mode		PC offs.
		No Jmp	Jmp			dst	src	
	2	10	12	A	-	r	-	N

OPERATION: r ⇐ r - 1
If r not equal to 0 then PC ⇐ PC + N

The destination working register being used as a counter is decremented. If the contents of the register are not zero after decrementing, the offset N (where N is in the range -128/+127) is added to the program counter. The original value of the program counter is taken to be the address of the instruction byte following the DJNZ instruction. When the working register counter reaches zero, control falls through to the statement following the DJNZ statement.

FLAGS: No flags affected.

EXAMPLE: DJNZ is typically used to control a "loop" of instructions. In the following example 12 bytes are moved from one area in the register file to another. The steps involved are:

```
;load 12 into the counter (working register 6)
;set up the loop to perform the moves
;end the loop with djnz

pointer1 = oldbuf-1
pointer2 = oldbuf-1

       ld r6,#12              ;load counter

Loop:  ld r9,pointer1(r6)     ;move one byte to
       ld r9,pointer2(r6),r9;new location
       djnz r6,Loop           ;decrement and loop until
                              ;counter = 0
```

NOTE : Due to the ST9 architecture, the DJNZ instruction should not be used with working registers in group E or F.

DWJNZ

Decrement Word And Jump If Not Zero

DWJNZ dst,N

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		PC offs.
					dst	src	
[OPC] [dst,0] [PC Offset]	3	14	16	C6	-	RR	N
	3	14	16	C6	-	rr*	N

OPERATION: rr ⇐ rr - 1
If rr not equal to 0 then PC ⇐ PC + N

The destination working register being used as a counter is decremented. If the contents of the register are not zero after decrementing, the offset N (where N is in the range -128/+127) is added to the program counter. The original value of the program counter is taken to be the address of the instruction byte following the DWJNZ instruction. When the working register counter reaches zero, control falls through to the statement following the DWJNZ statement.

FLAGS: No flags affected.

EXAMPLE: DJNZ is typically used to control a "loop" of instructions. In the following example 300 bytes are moved from one area in the register file to another. The steps involved are:

```
;load 300 into the counter (working register pair 6)
;set up the loop to perform the moves
;end the loop with dwjnz

pointer1 = oldbuf-2 pointer2 = oldbuf-2

       ld rr6,#300          ;load counter
Loop:  ld r9, pointer1(rr6) ;move one to byte
       ld pointer2(rr6),r9  ;new location
       dwjnz rr6,Loop       ;decrement and loop until
                            ;counter = 0
```

NOTE : Due to the ST9 architecture, the DWJNZ instruction cannot be used with working registers in groups D, E or F.

EI

Enable Global Interrupts

EI

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC]	1	6	00	-	-	-

OPERATION: CIC.4 ⇐ 1

Bit 4 of the Central Interrupt Control register (R230) is set to one. All interrupts except NMI are then enabled.

FLAGS: No flag affected.

EXAMPLE:

Instruction	HEX	Binary
EI	00	0000 0000

After this instruction all interrupts (except NMI) are enabled.

NOTE: The NMI (Not Maskable Interrupt) must be separately enabled (see Technical Manual).

EXT

Sign Extend

EXT dst

INSTRUCTION FORMAT:

[OPC] [dst,1]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	2	10	C6	-	RR	-
	2	10	C6	-	rr*	-

OPERATION: dst(n) MSB ⇐ dst(7) LSB; where n=8,..,15

This instruction extends to the MSB register the sign bit (bit 7) of the LSB register. If bit 7 of the LSB is 1, all bits of the MSB register will be set to 1, if bit 7 of the LSB is 0, all bits of the MSB are reset. The destination is directly addressed.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
EXT RR10	C6 0B	1100 0110 0000 1011

If bit 7 of register R11 is 1, after this instruction all bits in register R10 will be 1.

HALT

Halt

HALT

INSTRUCTION FORMAT:

[OPC] [XTN]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	2	6	BF	01	-	-

OPERATION: Stops program execution until next system reset.

FLAGS: No flags Affected.

EXAMPLE:

Instruction	HEX	Binary
HALT	BF 01	1011 1111 0000 0001

When the program encounters this instruction it is halted until a reset is executed.

INC

Increment Register

INC dst

INSTRUCTION FORMAT:

[OPC] [dst]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	2	6	50	-	R	-
	2	6	50	-	r*	-
	2	6	51	-	(R)	-
	2	6	51	-	(r)*	-

OPERATION: dst ⇐ dst + 1

The content of the destination register, directly or indirectly addressed, is incremented by 1.

FLAGS:
- C: Unaffected.
- Z: Set if result is zero, otherwise cleared.
- S: Set if result is negative, otherwise cleared.
- V: Set if arithmetic overflow occured, otherwise cleared.
- D: Unaffected.
- H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
INC (R32)	51 20	0101 0001 0010 0000

If register 32 holds 142 and register 142 contains 95 (decimal), after this instruction register 142 will contain 96.

INCW

Register Increment Word

INCW dst

INSTRUCTION FORMAT:

[OPC] [dst,0]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	2	8	DF	-	RR	-
	2	8	DF	-	rr*	-

OPERATION: dst ⇐ dst + 1

The destination register pair content is incremented by 1.

FLAGS:

C: Unaffected.
Z: Set if result is zero, otherwise cleared.
S: Set if result is negative, otherwise cleared.
V: Set if arithmetic overflow occured, otherwise cleared.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
INCW RR32	DF 20	1101 1111 0010 0000

If register pair 32 contains 4000 (decimal) after this instruction it will contain 4001 (decimal).

IRET

Interrupt Return

IRET

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC]	1	16	D3	-	-	-

OPERATION:

FLAGS ⇐ (SSP)
SSP ⇐ SSP + 1
PC ⇐ (SSP)
SSP ⇐ SSP + 2
CIC.4 ⇐ 1

Issued at the end of an interrupt service routine, this instruction restores the flag register and the program counter. It also re-enables any interrupts that are potentially enabled.

FLAGS: All flags are restored to original setting (before interrupt occurred).

EXAMPLE:

Instruction	HEX	Binary
IRET	D3	1101 0011

This instruction causes the program to resume execution exactly at the point it left when an interrupt service routine was initiated. All flags are set to the status they had when the interrupt service routine was started.

JP

Unconditional Jump

JP dst

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,0]	2	8	D4	-	(RR)	-
	2	8	D4	-	(rr)*	-
[OPC] [dst h] [dst l]	3	10	8D	-	NN	-

OPERATION: PC ⇐ dst

The unconditional jump simply replaces the contents of the program counter with the destination contents. Control then passes to the statement addressed by the program counter.

The destination operand can be in a directly or indirectly addressed program memory location.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
JP 1024	8D 04 00	1000 1101 0000 0100 0000 0000

The instruction replaces the content of the program counter with 1024 (decimal) and transfers program control to that location.

JPcc

Conditional Jump

JPcc dst

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[cc | OPC] [dst h] [dst l]	3	10	D	-	NN	-

OPERATION: If cc is true, PC ⇐ dst

The conditional jump transfers program control to the designated location if the condition code specified by "cc" is true. The destination operand is a directly addressed program memory location.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
JPEQ 1024	6D 04 00	0110 1101 0000 0100 0000 0000

If the result of the last mathematic or logic operation left the zero flag set, then the program counter is loaded with 1024 (decimal) and control is transferred to that location.

JRcc

Conditional Jump Relative

JRcc dst

INSTRUCTION FORMAT:

[cc | OPC] [dst]	No. Bytes	No. Cycl		OPC (HEX)	OPC XTN	Addr Mode	
		No Jmp	Jmp			dst	src
	2	12	10	B	-	-	N

OPERATION: If cc is true, PC ⇐ PC + dst

The conditional jump adds the immediate data to the program counter and control is transferred to the new location if the condition code specified by "cc" is true. The range of the relative address is +127/-128, and the original value of the program counter is taken to be the address of the first instruction byte following the JRcc statement.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
JREQ 24	6B 18	0110 1011 0001 1000

If the result of the last mathematic or logic operation left the zero flag set then the program counter is loaded with the present value plus 24 and control is transferred to that location.

LD

Load (byte) Register, Register

LD dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src
[dst | OPC] [src]	2	6	8	-	r	R
	2	6	8	-	r	r*
[src | OPC] [dst]	2	6	9	-	R	r
[OPC] [dst | src]	2	6	E5	-	(r)	r
	2	6	E4	-	r	(r)
[OPC] [src] [XTN | dst]	3	10	E6	F	(r)	R
	3	10	E6	F	(r)	r*
[OPC] [XTN | src] [dst]	3	10	E7	F	R	(r)
[OPC] [src | dst] [ofd]	3	10	B2	-	N(r)	r
[OPC] [dst | src] [ofs]	3	10	B3	-	r	N(r)
[OPC] [src] [dst]	3	10	F4	-	R	R

OPERATION: dst ⇐ src

The contents of the source are loaded into the destination. The contents of the source are not affected. The source and destination can both be addressed directly, indirectly or by indexing.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
LD r8,72(r5)	B3 85 48	1011 0011 1000 0101 0100 1000

If register 5 contains 183 (decimal) and register 255 (i.e. 183+72) contains 131 (decimal), after this instruction working register 8 will contain 131.

LD

Load (byte) Register, Memory

LD dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [dst |src,0]	2	10	B5	-	r	(rr)	a
[OPC] [dst |src,1]	2	14	D7	-	(r)+	(rr)+	d
[OPC] [XTN |src,1] [dst]	3	16	B4	F	R	(rr)+	b
	3	16	B4	F	r*	(rr)+	b
	3	16	C2	F	R	-(rr)	c
	3	16	C2	F	r*	-(rr)	c
	3	12	72	F	R	(rr)	a
[OPC] [ofs,1|src,0] [XTN | dst]	3	22	60	F	r	rr(rr)	a
[OPC] [XTN |src,1] [ofs] [dst]	4	24	7F	F	R	N(rr)	a
	4	24	7F	F	r*	N(rr)	a
[OPC] [XTN | dst] [src h] [src l]	4	18	C4	F	r	NN	a
[OPC] [XTN |src,0] [ofs h] [ofs l] [dst]	5	26	7F	F	R	NN(rr)	a
	5	26	7F	F	r*	NN(rr)	a

OPERATION a: dst ⇐ src

The destination register will be loaded with the contents of the memory location addressed either directly, indirectly or by indexing.

OPERATION b: dst ⇐ src
rr ⇐ rr + 1

The contents of the memory location addressed by the source register pair are loaded into the directly addressed destination register. The contents of the source register pair are incremented after the load has been carried out.

OPERATION c: rr ⇐ rr - 1
dst ⇐ src

The contents of the source register pair are decremented and then the contents of the memory location addressed by the source register pair are loaded into the directly addressed destination register.

OPERATION d: dst ⇐ src
r ⇐ r + 1
rr ⇐ rr + 1

The contents of the memory location addressed by the source register pair are loaded into the register addressed by the destination register. The source and destination register are incremented after the load has been carried out.

LD

Load (byte) Register, Memory

LD dst,src (Cont'd)

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
LD (r4)+,(rr6)+	D7 47	1101 0111 0100 0111

If working register 4 contains 100 (decimal), working register pair 6 contains 1242 (decimal) and memory location 1242 contains 132, after this instruction register 100 will contain 132, working register 4 will contain 101 and working register 6 will contain 1243.

LD

Load (byte) Memory, Register

LD dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src	Oper
[OPC] [src |dst,0]	2	18	D7	-	(rr)+	(r)+	d
[OPC] [src |dst,1]	2	10	B5	-	(rr)	(r)	a
[OPC] [XTN |dst,0] [src]	3	18	B4	F	(rr)+	R	b
	3	18	B4	F	(rr)+	r*	b
	3	18	C2	F	-(rr)	R	c
	3	18	C2	F	-(rr)	r*	c
	3	14	72	F	(rr)	R	a
[OPC] [ofd,1|dst,1] [XTN | src]	3	22	60	F	rr(rr)	r	a
[OPC] [XTN |dst,1] [ofd]	4	24	26	F	N(rr)	R	a
[src]	4	24	26	F	N(rr)	r*	a
[OPC] [XTN | src] [dst h]	4	18	C5	F	NN	r	a
[dst l]							
[OPC] [XTN |dst,0] [ofd h]	5	26	26	F	NN(rr)	R	a
[ofd l] [src]	5	26	26	F	NN(rr)	r*	a

OPERATION a: dst ⇐ src

The data in the source register is loaded into the memory location addressed either directly, indirectly or by indexing.

OPERATION b: dst ⇐ src
rr ⇐ rr + 1

The memory location addressed by the destination register pair is loaded with the contents of the directly addressed source register. The contents of the source register pair are incremented after the load has been carried out.

OPERATION c: rr ⇐ rr - 1
dst ⇐ src

The contents of the destination register pair are decremented and then the memory location addressed by the the destination register pair is loaded with the contents of the directly addressed source register.

OPERATION d: dst ⇐ src
r ⇐ r + 1
rr ⇐ rr + 1

The memory location addressed by the destination register pair is loaded with the contents of the register addressed by the source register. The source and destination register are incremented after the load has been carried out.

LD

Load (byte) Memory, Register

LD dst,src (Cont'd)

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
LD (rr4)+,(r6)+	D7 64	1101 0111 0110 0100

If working register pair 4 contains 1000 (decimal), working register 6 contains 242 (decimal) and register 242 contains 132, after this instruction memory location 1000 will contain 132, working register pair 4 will contain 1101 and working register 6 will contain 243.

LD

Load (Byte) Memory, Memory

LD dst,src

INSTRUCTION FORMAT:

[OPC] [XTN |src,0] [dst,0]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	3	16	73	F	(RR)	(rr)
	3	16	73	F	(rr)*	(rr)

OPERATION: dst ⇐ src

The contents of the memory location addressed by the source register pair are loaded into the memory location addressed by the destination register pair.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
LD (rr4),(rr6)	73 F6 D4	0111 0011 1111 0110 1101 0100

If working register pair 4 contains 1000 (decimal), working register pair 6 contains 1242 (decimal) and memory location 1242 contains 132, after this instruction memory location 1000 will contain 132.

LD

Load (Byte) All, Immediate

LD dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[dst | OPC] [src]	2	6	C	-	r	#N
[OPC] [dst] [src]	3	10	F5	-	R	#N
[OPC] [XTN |dst,0] [src]	3	12	F3	F	(rr)	#N
[OPC] [XTN] [src] [dst h] [dst l]	5	20	2F	F1	NN	#N

OPERATION: dst ⇐ src

The value #N is loaded into the destination register or memory location.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
LD r8,#242	8C F2	1000 1100 1111 0010

After this instruction has been carried out working register 8 contains the decimal value 242.

LDPP LDDP LDPD LDDD

Load (Byte) Data/Program Memory, Data/Program Memory

LDPP dst,src LDDP dst,src LDPD dst,src LDDD dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
LDPP: [OPC] [dst,0|src,0]	2	16	D6	-	(rr)+	(rr)+
LDDP: [OPC] [dst,1|src,0]	2	16	D6	-	(rr)+	(rr)+
LDPD: [OPC] [dst,0|src,1]	2	16	D6	-	(rr)+	(rr)+
LDDD: [OPC] [dst,1|src,1]	2	16	D6	-	(rr)+	(rr)+

OPERATION:

dst ⇐ src
rrd ⇐ rrd + 1
rrs ⇐ rrs + 1

The data in the indirectly addressed memory source byte is loaded into the indirectly addressed memory destination byte. The contents of the working register pairs used to address both source and destination are incremented after the instruction has been carried out. Source and destination can be both in the data memory, both in the program memory or one can be in the data memory while the other is in the program memory.

FLAGS:

No flags affected.

EXAMPLE:

Instruction	HEX	Binary
LDDP (rr8)+,(rr12)+	D6 9D	1101 0110 1001 1101

If working register pair 8 contains 1131 (decimal), working register pair 12 contains 2400 (decimal) and the memory location 2400 contains 100 (decimal), after this instruction memory location 1131 will contain 100, working register pair 8 will contain 1132 and working register pair 12 will contain 2401.

LDW

Load (Word) Register, Register

LDW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,0|src,0]	2	10	E3	-	rr	rr
[OPC] [src,0] [XTN | dst]	3	10	96	F	(r)	RR
	3	10	96	F	(r)	rr*
[OPC] [XTN | src] [dst,0]	3	10	A6	F	RR	(r)
	3	10	A6	F	rr*	(r)
[OPC] [src,1| dst] [ofd]	3	14	DE	-	N(r)	rr
[OPC] [dst,0| src] [ofs]	3	16	DE	-	rr	N(r)
[OPC] [src,0] [dst,0]	3	10	EF	-	RR	RR
	3	10	EF	-	rr*	RR
	3	10	EF	-	RR	rr*

OPERATION: dst ⇐ src

The contents of the source are loaded into the destination. The contents of the source are not affected. The source and destination can be addressed directly, indirectly or by indexing.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
LDW rr8,RR254	EF FE D8	1110 1111 1111 1110 1101 1000

If register pair 254 contains 3F C1 (hex), after this instruction the working register pair 8 will contains 3F C1 (hex).

LDW

Load (Word) Register, Memory

LDW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src	Oper
[OPC] [dst,0|src,1]	2	16	E3	-	rr	(rr)	a
[OPC] [XTN |src,1] [dst,0]	3	20	D5	F	RR	(rr)+	b
	3	20	D5	F	rr*	(rr)+	b
	3	22	C3	F	RR	-(rr)	c
	3	22	C3	F	rr*	-(rr)	c
[OPC] [XTN |src,0] [dst,0]	3	18	7E	F	RR	(rr)	a
[OPC] [ofs,0|src,0] [XTN |dst,0]	3	24	60	F	rr	rr(rr)	a
[OPC] [XTN |src,1] [ofs] [dst,0]	4	28	86	F	RR	N(rr)	a
	4	28	86	F	rr*	N(rr)	a
[OPC] [XTN |dst,0] [src h] [src l]	4	22	E2	F	rr	NN	a
[OPC] [XTN |src,0] [ofs h] [ofs l] [dst,0]	5	30	86	F	RR	NN(rr)	a
	5	30	86	F	rr*	NN(rr)	a

OPERATION a: dst ⇐ src

In the destination register pair will be loaded the contents of the memory location addressed either directly, indirectly or by indexing.

OPERATION b: dst ⇐ src
rr ⇐ rr + 2

The word in the memory pair addressed by the source register pair is loaded into the destination register pair The source address is for the word high order byte. The contents of the source register pair are incremented by two after the load has been carried out.

OPERATION c: rr ⇐ rr - 2
dst ⇐ src

The contents of the source register pair are decremented twice and then the word in the memory pair adressed by the source register pair is loaded into the destination register pair. The source address is for the word high order byte.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
LD rr8,(rr4)+	D5 F5 D8	1101 0101 1111 0101 1101 1000

If working register 4 contains 2400 (decimal) and memory pair 2400 contains 56 ED (Hex), after this instruction working register pair 8 will contain 56 ED and working register pair 4 will contain 2402.

LDW

Load (Word) Memory, Register

LDW dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src	Oper
[OPC] [dst,1|src,0]	2	18	E3	-	(rr)	rr	a
[OPC] [XTN |dst,0] [src,0]	3	24	D5	F	(rr)+	RR	b
	3	24	D5	F	(rr)+	rr*	b
	3	26	C3	F	-(rr)	RR	c
	3	26	C3	F	-(rr)	rr*	c
[OPC] [XTN |dst,1] [src,0]	3	20	BE	F	(rr)	RR	a
[OPC] [ofd,0|dst,1] [XTN |src,0]	3	24	60	F	rr(rr)	rr	a
[OPC] [XTN |dst,1] [ofd] [src,1]	4	26	86	F	N(rr)	RR	a
	4	26	86	F	N(rr)	rr*	a
[OPC] [XTN |src,1] [dst h] [dst l]	4	22	E2	F	NN	rr	a
[OPC] [XTN |src,0] [ofd h] [ofd l] [src,1]	5	28	86	F	NN(rr)	RR	a
	5	28	86	F	NN(rr)	rr*	a

OPERATION a: dst ⇐ src

The contents of the source register pair are loaded into the memory pair addressed either directly, indirectly or by indexing. The destination address is for the word high order byte.

OPERATION b: dst ⇐ src
rr ⇐ rr + 2

The contents of the source register pair are loaded into the memory pair addressed by the contents of the destination register pair. The destination address is for the word high order byte. The contents of the destination register pair are incremented twice after the load has been carried out.

OPERATION c: rr ⇐ rr - 2
dst ⇐ src

The contents of the destination register pair are decremented twice and then the contents of the source register pair are loaded into the memory pair addressed by the contents of the destination register pair. The destination address is for the word high order byte.

LDW

Load (Word) Memory, Register

LDW dst,src (Cont'd)

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
LDW (rr4)+,RR64	D5 F4 40	1101 0101 1111 0100 0100 0000

If working register pair 4 contains 1024 (decimal) and register pair 64 contains 8F E3 (Hex), after this instruction memory pair 1024 will contain 8F E3 and register pair 4 will contain 1026.

LDW

Load (Word) Memory, Memory

LDW dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,1|src,1]	2	22	E3	-	(rr)	(rr)

OPERATION: dst ⇐ src

The contents of the memory pair addressed by the source register pair are loaded into the memory pair addressed by the destination register pair. The source and destination addresses are for the word high order byte.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
LDW (rr4),(rr6)	E3 57	1110 0011 0101 0111

If working register pair 4 contains 1024 (decimal), working register pair 6 contains 2042 (decimal) and memory pair 2042 contains CB ED (Hex), after this instruction memory pair 1024 will contain CB ED.

LDW

Load (Word) All, Immediate

LDW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src
[OPC] [dst,0] [src h] [src l]	4	12	BF	-	RR	#NN
	4	12	BF	-	rr*	#NN
[OPC] [XTN |dst,0] [src h] [src l]	4	22	BE	F	(rr)	#NN
[OPC] [XTN |dst,1] [ofd] [src h] [src l]	5	28	06	F	N(rr)	#NN
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src h] [src l]	6	30	06	F	NN(rr)	#NN
[OPC] [XTN] [src h] [src l] [dst h] [dst l]	6	26	36	F1	NN	#NN

OPERATION: dst ⇐ src

The value #NN is loaded into the destination register pair or memory pair.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
LDW RR100,#4268	BF 64 10 AC	1011 1111 0110 0100 0001 0000 1010 1100

After this instruction has been carried out register pair 100 contains the decimal value 4268 (10 AC Hex.).

MUL

Multiply (8x8)

MUL dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,0| src]	2	22	4F	-	rr	r

OPERATION: dst ⇐ dst(low) * src

The contents of the source register are multiplied by the low order byte of the destination register pair. The 16 bit result is left in the destination register pair.

Input
rr dst high (even address) = don't care
rr dst low (odd address) = first operand
rr src = second operand

Output
rr dst high = LSB of the result
rr dst low = MSB of the result

The src byte holds the unmodified second operand.

FLAGS:
C: Contains a copy of result bit 0.
Z: Contains a copy of result bit 15.
S: Set if result MSB is zero, otherwise reset to zero.
V: Set if result LSB is zero, otherwise reset.
D: Always reset to zero.
H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
MUL rr6,r8	4F 68	0100 1111 0110 1000

If working register 7 contains 35 and working register 8 contains 220, after this instruction working register pair 6 will contain 7700 (decimal), i.e. working register 6 will contain 1E (Hex) and register 7 will contain 14 (Hex).

NOP

No Operation

NOP

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC]	1	6	FF	-	-	-

OPERATION: No Operation is carried out. This instruction is often used in timing or delay loops.

FLAGS: No flags affected.

OR

OR (byte) Register, Register

OR dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src
[OPC] [dst | src]	2	6	02	-	r	r
	2	6	03	-	r	(r)
[OPC] [src] [dst]	3	10	04	-	R	R
	3	10	04	-	r*	R
	3	10	04	-	R	r*
[OPC] [src] [XTN | dst]	3	10	E6	0	(r)	R
	3	10	E6	0	(r)	r*
[OPC] [XTN | src] [dst]	3	10	E7	0	R	(r)

OPERATION: dst ⇐ dst OR src

The contents of the source are ORed with the destination byte and the results stored in the destination byte. The contents of the source are not affected.

FLAGS:
C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 7 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
OR r8,R64	04 40 D8	0000 0100 0100 0000 1101 1000

If working register 8 contains 11001100 and register 64 contains 10000101, after this instruction working register 8 will contain 11001101.

OR

OR (byte) Register, Memory

OR dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [XTN |src,1] [dst]	3	12	72	0	R	(rr)	a
	3	12	72	0	r*	(rr)	a
	3	16	B4	0	R	(rr)+	b
	3	16	B4	0	r*	(rr)+	b
	3	16	C2	0	R	-(rr)	c
	3	16	C2	0	r*	-(rr)	c
[OPC] [ofs,1|src,0] [XTN | dst]	3	22	60	0	r	rr(rr)	a
[OPC] [XTN |src,1] [ofs] [dst]	4	24	7F	0	R	N(rr)	a
	4	24	7F	0	r*	N(rr)	a
[OPC] [XTN | dst] [src h] [src l]	4	18	C4	0	r	NN	a
[OPC] [XTN |src,0] [ofs h] [ofs l] [dst]	5	26	7F	0	R	NN(rr)	a
	5	26	7F	0	r*	NN(rr)	a

OPERATION a: dst ⇐ dst OR src

The source byte is ORed with the destination byte and the result stored in the destination byte. The destination register is addressed directly, the memory location (source byte) addressed either directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst OR src
rr ⇐ rr + 1

The contents of the memory location addressed by the source register pair are ORed with the contents of the directly addressed destination register the result stored in the destination register. The contents of the source register pair are incremented after the OR has been carried out.

OPERATION c: rr ⇐ rr - 1
dst ⇐ dst OR src

The contents of the source register pair are decremented and then the contents of the memory location addressed by the source register pair are ORed with the contents of the directly addressed destination register. The result is stored in the destination register.

OR

OR (byte) Register, Memory

OR dst,src (Cont'd)

FLAGS:

C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 7 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
OR r8,4028	C4 08 0F BC	1100 0100 0000 1000 0000 1111 1011 1100

If working register 8 contains 11001100 and memory location 4028 contains 10000101, after this instruction working register 8 will contain 11001101.

OR

OR (byte) Memory, Register

OR dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [XTN |dst,0] [src]	3	18	72	0	(rr)	R	a
	3	18	72	0	(rr)	r*	a
	3	22	B4	0	(rr)+	R	b
	3	22	B4	0	(rr)+	r*	b
	3	22	C2	0	-(rr)	R	c
	3	22	C2	0	-(rr)	r*	c
[OPC] [ofd,1|dst,1] [XTN | src]	3	24	60	0	rr(rr)	r	a
[OPC] [XTN |dst,1] [ofd]	4	26	26	0	N(rr)	R	a
[src]	4	26	26	0	N(rr)	r*	a
[OPC] [XTN | src] [dst h]	4	20	C5	0	NN	r	a
[dst l]							
[OPC] [XTN |dst,0] [ofd h]	5	28	26	0	NN(rr)	R	a
[ofd l] [src]	5	28	26	0	NN(rr)	r*	a

OPERATION a: dst ⇐ dst OR src

The source byte is ORed with the destination byte and the result stored in the destination byte. The source registers are addressed directly, the memory location are addressed either directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst OR src
rr ⇐ rr + 1

The contents of the memory location addressed by the destination register pair (destination byte) are ORed with the contents of the directly addressed source register the result stored in the destination byte. The contents of the destination register pair are incremented after the OR has been carried out.

OPERATION c: rr ⇐ rr - 1
dst ⇐ dst OR src

The contents of the destination register pair are decremented and then the contents of the memory location addressed by the destination register pair (destination byte) are ORed with the contents of the directly addressed source register. The result is stored in the destination byte.

OR

OR (byte) Memory, Register

OR dst,src (Cont'd)

FLAGS:

C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 7 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
OR 4028,r8	C5 08 0F BC	1100 0101 0000 1000 0000 1111 1011 1100

If working register 8 contains 11001100 and memory location 4028 contains 10000101, after this instruction memory location 4028 will contain 11001101.

OR

OR (byte) Memory, Memory

OR dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [XTN |src,0] [dst,0]	3	20	73	0	(RR)	(rr)
	3	20	73	0	(rr*)	(rr)

OPERATION: dst ⇐ dst OR src

The contents of the memory location addressed by the source register pair are ORed with the content of the memory location addressed by the destination register pair. The source and destination addresses are for the word high order byte.

FLAGS:
C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 7 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
OR (rr4),(rr8)	73 08 D4	0111 0011 0000 1000 1101 0100

If working register pair 4 contains 2800 (decimal), memory location 2800 contains 11001100, working register pair 8 contains 4200 (decimal) and memory location 4200 contains 00001100, after this instruction memory location 2800 will contain 11001100.

OR

OR (byte) All, Immediate

OR dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst] [src]	3	10	05	-	R	#N
	3	10	05	-	r*	#N
[OPC] [XTN |dst,0] [src]	3	16	F3	0	(rr)	#N
[OPC] [XTN] [src] [dst h] [dst l]	5	24	2F	01	NN	#N

OPERATION: dst ⇐ dst OR src

The value #N is ORed with the content of the destination register or memory location.

FLAGS:
C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 7 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
OR (rr8),#32	F3 18 20	1111 0011 0001 1000 0010 0000

If working register pair 8 contains 4028 (decimal) and memory location 4028 contains 11101101, after this instruction memory location 4028 will contain 11101101.

ORW

OR (Word) - Register, Register

ORW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,0|src,0]	2	10	0E	-	rr	rr
[OPC] [src,0] [dst,0]	3	12	07	-	RR	RR
	3	12	07	-	rr*	RR
	3	12	07	-	RR	rr*
[OPC] [src,0] [XTN | dst]	3	14	96	0	(r)	RR
	3	14	96	0	(r)	rr*
[OPC] [XTN | src] [dst,0]	3	14	A6	0	RR	(r)
	3	14	A6	0	rr*	(r)

OPERATION: dst ⇐ dst OR src

The source word is ORed with the destination word and the result is stored in the destination word. The source and destination word can be addressed either directly or indirectly.

FLAGS:
- C: Unaffected.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if result bit 15 is set, otherwise cleared.
- V: Always reset to zero.
- D: Unaffected.
- H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
ORW (r8),RR64	96 40 08	1001 0110 0100 0000 0000 1000

If register pair 64 contains 11001100/11001100B, working register 8 contains 200 (decimal) and register pair 200 contains 10101010/10101010B, after this instruction register pair 200 will hold 11101110/11101110B.

ORW

OR (Word) - Register, Memory

ORW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src	Oper
[OPC] [dst,0|src,1]	2	16	0E	-	rr	(rr)	a
[OPC] [XTN |src,0] [dst,0]	3	18	7E	0	RR	(rr)	a
[OPC] [XTN |src,1] [dst,0]	3	22	D5	0	RR	(rr)+	b
	3	22	D5	0	rr*	(rr)+	b
	3	24	C3	0	RR	-(rr)	c
	3	24	C3	0	rr*	-(rr)	c
[OPC] [ofs,0|src,0] [XTN |dst,0]	3	24	60	0	rr	rr(rr)	a
[OPC] [XTN |src,1] [ofs] [dst,0]	4	28	86	0	RR	N(rr)	a
	4	28	86	0	rr*	N(rr)	a
[OPC] [XTN |dst,0] [src h] [src l]	4	22	E2	0	rr	NN	a
[OPC] [XTN |src,0] [ofs h] [ofs l] [dst,0]	5	30	86	0	RR	NN(rr)	a
	5	30	86	0	rr*	NN(rr)	a

OPERATION a: dst ⇐ dst OR src

The source word is ORed with the destination word and the result is stored in the destination word. The destination word is held in the destination register. The source word can be addressed directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst OR src
rr ⇐ rr + 2

The source word is ORed with the destination word and the result is stored in the destination word. The source word is in the memory location addressed by the source register pair, the destination word is in the destination register. The contents of the source register pair are incremented after the OR has been carried out.

OPERATION c: rr ⇐ rr - 2
dst ⇐ dst OR src

The source word is ORed with the destination word and the result is stored in the destination word. The source word is in the memory location addressed by the source register pair, the destination word is in the destination register. The contents of the source register pair are decremented before the OR is carried out.

ORW

OR (Word) - Register, Memory

ORW dst,src (Cont'd)

FLAGS:
- C: Unaffected.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if result bit 15 is set, otherwise cleared.
- V: Always reset to zero.
- D: Unaffected.
- H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
ORW RR64,-(rr4)	C3 05 40	1100 0011 0000 0101 0100 0000

If working register pair 4 contains 1184 (decimal), register pair 64 contains 10101010/10101010B and memory pair 1182 contains 11001100/11001100B, after this instruction register pair 64 will contain 11101110/11101110B and register pair 4 will contain 1182.

ORW

OR (Word) - Memory, Register

ORW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [dst,1|src,0]	2	30	0E	-	(rr)	rr	a
[OPC] [XTN |dst,1] [src,0]	3	32	BE	0	(rr)	RR	a
[OPC] [XTN |dst,0] [src,0]	3	34	D5	0	(rr)+	RR	b
	3	34	D5	0	(rr)+	rr*	b
	3	32	C3	0	-(rr)	RR	c
	3	32	C3	0	-(rr)	rr*	c
[OPC] [ofd,0|dst,1] [XTN |src,0]	3	36	60	0	rr(rr)	rr	a
[OPC] [XTN |dst,1] [ofd] [src,1]	4	38	86	0	N(rr)	RR	a
	4	38	86	0	N(rr)	rr*	a
[OPC] [XTN |src,1] [dst h] [dst l]	4	36	E2	0	NN	rr	a
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src,1]	5	40	86	0	NN(rr)	RR	a
	5	40	86	0	NN(rr)	rr*	a

OPERATION a: dst ⇐ dst OR src

The source word is ORed with the destination word and the result is stored in the destination word. The source word is held in the source register. The destination word can be addressed directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst OR src
rr ⇐ rr + 2

The source word is ORed with the destination word and the result is stored in the destination word. The source word is in the source register, the destination word is in the memory location addressed by the destination register pair. The contents of the destination register pair are incremented after the OR has been carried out.

OPERATION c: rr ⇐ rr - 2
dst ⇐ dst OR src

The source word is ORed with the destination word and the result is stored in the destination word. The source word is in the source register , the destination word is in the memory location addressed by the destination register pair. The contents of the destination register pair are decremented before the OR is carried out.

ORW

OR (Word) - Memory, Register

ORW dst,src (Cont'd)

FLAGS:

C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 15 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
ORW (rr4)+,RR64	D5 04 40	1101 0101 0000 0100 0100 0000

If register pair 64 contains 11001100/1101100B, working register pair 4 contains 1064 (decimal) and memory pair 1064 contains 10101010/10101010B, after this instruction has been carried out memory pair 1064 will contain 11101110/11101110B and working register pair 4 will contain 1066.

ORW

OR (Word) - Memory, Memory

ORW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,1|src,1]	2	34	0E	-	(rr)	(rr)

OPERATION: dst ⇐ dst OR src

The source word is ORed with the destination word and the result is stored in the destination word. The source word is in the memory location addressed by the source register pair, the destination word is in the memory location addressed by the destination register pair.

FLAGS:
- C: Unaffected.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if result bit 15 is set, otherwise cleared.
- V: Always reset to zero.
- D: Unaffected.
- H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
ORW (rr4),(rr6)	0E 57	0000 1110 0101 0111

If working register pair 6 contains 1002 (decimal), memory pair 1002 contains 11001100/11001100B, working register pair 4 contains 1060 (decimal) and memory pair 1060 contains 10101010/10101010B, after this instruction memory pair 1060 will contain 11101110/11101110B.

ORW

OR (Word) - All, Immediate

ORW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,1] [src h] [src l]	4	14	07	-	RR	#NN
	4	14	07	-	rr*	#NN
[OPC] [XTN |dst,0] [src h] [src l]	4	34	BE	0	(rr)	#NN
[OPC] [XTN |dst,1] [ofd] [src h] [src l]	5	38	06	0	N(rr)	#NN
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src h] [src l]	6	40	06	0	NN(rr)	#NN
[OPC] [XTN] [src h] [src l] [dst h] [dst l]	6	40	36	01	NN	#NN

OPERATION: dst ⇐ dst OR src

The source word is ORed with the destination word and the result is stored in the destination word. The source word is the immediate value in the operand, the destination word can be in memory or in the register file.

FLAGS:
- C: Unaffected.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if result bit 15 is set, otherwise cleared.
- V: Always reset to zero.
- D: Unaffected.
- H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
ORW RR64,#52428	07 41 CC CC	0000 0111 0100 0001 1100 1100 1100 1100

If register pair 64 contains 10101010/10101010B, after this instruction has been carried out register pair 64 will contain 11101110/11101110B.

RCF

Reset Carry Flag

RCF

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC]	1	6	11	-	-	-

OPERATION: $C \Leftarrow 0$

The carry flag is reset to zero, regardless of its previous content.

FLAGS: C: reset to zero.
No other flags affected.

EXAMPLE:

Instruction	HEX	Binary
RCF	11	0001 0001

Regardless of its prior condition, after this instruction the carry flag will be reset to zero.

RET

Return From Subroutine

RET

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC]	1	12	46	-	-	-

OPERATION: PC ⇐ (SSP)
SSP ⇐ SSP - 2
This instruction is normally used to return to the previously executed procedure at the end of procedure entered by a CALL statement. The contents of the location addressed by the system stack pointer are popped into the program counter. The next statement executed is that addressed by the new content of the PC.

FLAGS: No flags affected.

EXAMPLE : If the program counter contains 35B4 (hex), the system stack pointer contains 2000 (hex), external data memory location 2000 (hex) contains 18 (hex), and location 2001 (hex) contains 85 (hex), then the instruction:

```
RET
```

leaves the value 2002 (hex) in the system stack pointer and 18B5 (hex), the addressed of the next instruction, in the program counter.

RLC

Rotate Left Through Carry

RLC dst

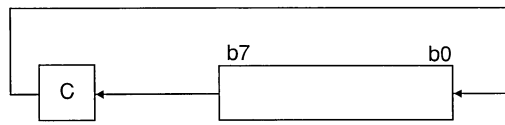
INSTRUCTION FORMAT:

[OPC] [dst]

No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
				dst	src
2	6	80	-	R	-
2	6	80	-	r*	-
2	6	81	-	(R)	-
2	6	81	-	(r)*	-

OPERATION: dst(0) ⇐ C
C ⇐ dst(7)
dst(n+1) ⇐ dst(n) Where n=0-6

The contents of the destination register are shifted one place to the left with bit 7 shifted into the carry flag and the carry flag shifted into bit 0. The destination register can be directly or indirectly addressed.



FLAGS:
C: Set if carry from MSB (bit 7 was 1).
Z: Set if the result is zero, otherwise cleared.
S: Set if the result bit 7 is set, otherwise cleared.
V: Set if result bit 7 is changed, otherwise cleared.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
RLC (r2)	B1 D2	1011 0001 1101 0010

If the carry flag is zero, working register 2 contains 155 (decimal) and register 155 contains 11001100B, after this instruction register 155 will contain 10011000B and the carry flag will be set to 1.

RLCW

Rotate Left Through Carry Word

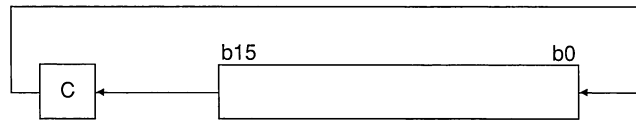
RLCW dst

INSTRUCTION FORMAT:

[OPC] [dst,0]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	2	8	8F	-	RR	-
	2	8	8F	-	rr*	-

OPERATION: dst(0) ⇐ C
C ⇐ dst(15)
dst(n+1) ⇐ dst(n) where n=0-14

The contents of the destination register pair are shifted one place to the left with bit 15 shifted into the carry flag and the carry flag shifted into bit 0.



FLAGS:
C: Set if carry from MSB bit 15 was 1.
Z: Undefined.
S: Set if the result bit 15 is set, otherwise cleared.
V: Set if result bit 15 is changed, otherwise cleared.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
LCW rr2	8F D2	1000 1111 1101 0010

If the carry flag is zero, and working register pair 2 contains 11001100/11001100B, after this instruction it will 10011001/10011000B and the carry flag will be set to 1.

ROL

Rotate Left Byte

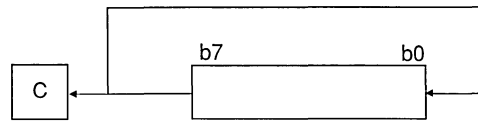
ROL dst

INSTRUCTION FORMAT:

[OPC] [dst]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	2	6	A0	-	R	-
	2	6	A0	-	r*	-
	2	6	A1	-	(R)	-
	2	6	A1	-	(r)*	-

OPERATION: C ⇐ dst(7
dst(0) ⇐ dst(7
dst(n+1) ⇐ dst(n) Where n=0-6

The contents of the destination register are shifted one place to the left with bit 7 shifted into bit 1 and into the carry flag. The destination register can be directly or indirectly addressed.



FLAGS:
C: Set if carry from MSB (bit 7 was 1).
Z: Set if the result is zero, otherwise cleared.
S: Set if the result bit 7 is set, otherwise cleared.
V: Set if result bit 7 is changed, otherwise cleared.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
ROL (r2)	A1 D2	1010 0001 1101 0010

If working register 2 contains 146 (decimal) and register 146 contains 11001100B, after this instruction register 146 will contain 10011001B and the carry flag will be set to 1.

ROR

Rotate Right Byte

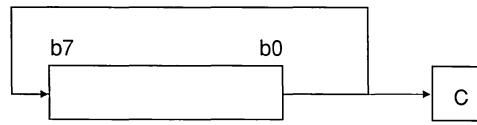
ROR dst

INSTRUCTION FORMAT:

[OPC] [dst]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	2	6	C0	-	R	-
	2	6	C0	-	r*	-
	2	6	C1	-	(R)	-
	2	6	C1	-	(r)*	-

OPERATION: C ⇐ dst(0)
dst(7) ⇐ dst(0)
dst(n) ⇐ dst(n+1) Where n=0-6

The contents of the destination register are shifted one place to the right with bit 0 shifted into bit 7 and into the carry flag. The destination register can be directly or indirectly addressed.



FLAGS:
C: Set if carry from LSB (bit 0 was 1).
Z: Set if the result is zero, otherwise cleared.
S: Set if the result bit 7 is set, otherwise cleared.
V: Set if result bit 7 is changed, otherwise cleared.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
ROR R32	C0 20	1100 0000 0010 0000

If the carry flag is set to one and register 32 contains 11001100B, after this instruction register 32 will contain 01100110B and the carry flag will be reset to zero.

RRC

Rotate Right Through Carry Byte

RRC dst

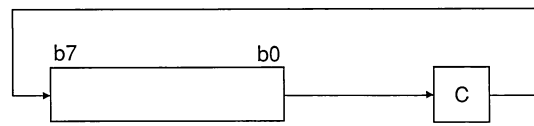
INSTRUCTION FORMAT:

[OPC] [dst]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	2	6	D0	-	R	-
	2	6	D0	-	r*	-
	2	6	D1	-	(R)	-
	2	6	D1	-	(r)*	-

OPERATION:

dst(7) ⇐ C
C ⇐ dst(0)
dst(n) ⇐ dst(n+1) Where n=0-6

The contents of the destination register are shifted one place to the right with bit 0 shifted into the carry flag and the carry flag shifted into bit 7. The destination register can be directly or indirectly addressed.



FLAGS:

C: Set if carry from LSB (bit 0 was 1).
Z: Set if the result is zero, otherwise cleared.
S: Set if the result bit 7 is set, otherwise cleared.
V: Set if result bit 7 is changed, otherwise cleared.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
RRC (R32)	D1 20	1101 0001 0010 0000

If the carry flag is zero, register 32 contains 155 and register 155 contains 00110011B, after this instruction register 155 will contain 00011001B and the carry flag will be set to 1.

RRCW

Rotate Right Through Carry Word

RRCW dst

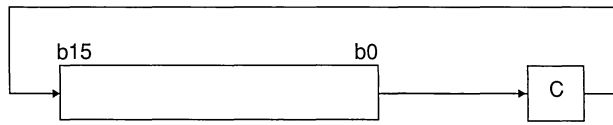
INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,0]	2	8	36	-	RR	-
	2	8	36	-	rr*	-

OPERATION: dst(15) ⇐ C
C ⇐ dst(0)
dst(n) ⇐ dst(n+1) where n=0-14

The contents of the destination register pair are shifted one place to the right with bit 0 shifted into the carry flag and the carry flag shifted into bit 15.

FLAGS: C: Set if carry from LSB (bit 0 was 1).



Z: Undefined.
S: Set if the result bit 15 is set, otherwise cleared.
V: Set if result bit 15 is changed, cleared otherwise.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
RRCW R32	36 20	0011 0110 0010 0000

If the carry flag is set and register 32 pair contains 11001100/11001100B, after this instruction register 32 will contain 11100110/01100110B and the zero flag will be reset to 0.

PEA

Push Effective Address on System Stack

PEA src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [XTN] [src,0] [ofs]	4	20	8F	01	-	N(RR)
	4	20	8F	01	-	NN(rr)*
[OPC] [XTN] [src,1] [ofs l] [ofs h]	5	26	8F	01	-	NN(RR)
	5	26	8F	01	-	NN(rr)*

OPERATION: SSP ⇐ SSP - 2
(SSP) ⇐ RR + "a" (Where "a" is the immediate value N or NN)
The present value of the SSP is decremented by 2 and the content of RR summed with the offset is pushed onto the system stack.

FLAGS: No flag affected.

EXAMPLE:

Instruction	HEX	Binary
PEA 16(RR32)	8F 01 20 10	1000 1111 0000 0001 0010 0000 0001 0000

The content of register pair RR32 is 1024, to this value is added the immediate value 16 and the result is pushed into the stack location pointed by the pre-decremented system stack pointer.

PEAU

Push Effective Address on User Stack

PEAU src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [XTN] [src,0] [ofs]	4	20	8F	03	-	N(RR)
	4	20	8F	03	-	N(rr)*
[OPC] [XTN] [src,1] [ofs l] [ofs h]	5	26	8F	03	-	NN(RR)
	5	26	8F	03	-	NN(rr)*

OPERATION: USP ⇐ USP - 2
(USP) ⇐ RR + "a" (Where "a" is the immediate value N or NN)
The present value of the USP is decremented by 2 and the contents of RR summed with the offset is pushed into the user stack.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
PEAU 16(RR32)	8F 03 20 10	1000 1111 0000 0011 0010 0000 0001 0000

The content of register pair RR32 is 1024, to this value is added the immediate value 16 and the result is pushed into the stack location pointed by the pre-decremented user stack pointer.

POP

Pop Byte from System Stack

POP dst

INSTRUCTION FORMAT:

[OPC] [dst]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	2	10	76	-	R	-
	2	10	76	-	r*	-
	2	10	77	-	(R)	-
	2	10	77	-	(r)*	-

OPERATION: dst ⇐ (SSP)
SSP ⇐ SSP + 1

The contents of the system stack addressed by the system stack pointer are loaded into the destination location and then the system stack pointer is incremented automatically by one.

FLAGS: No flags affected

EXAMPLE:

Instruction	HEX	Binary
POP (r2)	77 D2	0111 0111 1101 0010

If the system stack pointer contains 2000 (decimal), working register 2 contains 52 (decimal) and system stack location 2000 contains 124 (decimal), after this instruction register 52 will contain 124 and the system stack pointer will contain 2001.

POPU

Pop Byte from User Stack

POPU dst

INSTRUCTION FORMAT:

[OPC] [dst]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	2	10	20	-	R	-
	2	10	20	-	r*	-
	2	10	21	-	(R)	-
	2	10	21	-	(r)*	-

OPERATION: dst ⇐ (USP)
USP ⇐ USP + 1

The contents of the user stack addressed by the user stack pointer are loaded into the destination location and then the user stack pointer is increment automatically by one.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
POPU (r2)	21 D2	0010 0001 1101 0010

If the user stack pointer contains 2000 (decimal), working register 2 contains 52 (decimal) and user stack location 2000 contains 124 (decimal), after this instruction register 52 will contain 124 and the user stack pointer will contain 2001.

POPUW

Pop Word from User Stack

POPUW dst

INSTRUCTION FORMAT:

[OPC] [dst,0]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	2	14	B7	-	RR	-
	2	14	B7	-	rr*	-

OPERATION: dst ⇐ (USP)
USP ⇐ USP + 2

The contents of the user stack addressed by the user stack pointer are loaded into the destination register pair and the user stack pointer is automatically increment by two.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
POPUW rr2	B7 D2	1011 0111 1101 0010

If the user stack pointer contains 2000 (decimal), user stack location 2000 contains 11 (hex) and user stack location 2001 contains 24 (hex), after this instruction working register 2 will contain 11 (hex), working register 3 will contain 24 (hex) and the user stack pointer will contain 2002.

POPW

Pop Word from System Stack

POPW dst

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,0]	2	14	75	-	RR	-
	2	14	75	-	rr*	-

OPERATION: dst ⇐ (SSP)
SSP ⇐ SSP + 2

The contents of the system stack pointer are loaded into the destination register pair and the system stack pointer is automatically incremented by two.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
POPW rr2	75 D2	0111 0101 1101 0010

If the system stack pointer contains 2000 (decimal), system stack location 2000 contains 11 (hex), system stack location 2001 contains 24 (hex), after this instruction working register 2 will contain 11 (hex), working register 3 will contain 24 (hex) and the system stack pointer will contain 2002.

PUSH

Push Byte on System Stack

PUSH src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src
[OPC] [src]	2	10	66	-	-	R
	2	10	66	-	-	r*
	2	10	F7	-	-	(R)
	2	10	F7	-	-	(r)*
[OPC] [XTN] [src]	3	16	8F	F1	-	#N

OPERATION: SSP ⇐ SSP - 1
(SSP) ⇐ src
The system stack pointer is decremented automatically by one and then the operand loaded into the location addressed by the decremented system stack pointer.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
PUSH (R32)	F7 20	1111 0111 0010 0000

If the system stack pointer contains 2000 (decimal), register 32 contains 100 and register 100 contains 60 (decimal), after this instruction system stack pointer location 1999 will contain 60.

PUSHU

Push Byte on User Stack

PUSHU src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [src]	2	10	30	-	-	R
	2	10	30	-	-	r*
	2	10	31	-	-	(R)
	2	10	31	-	-	(r)*
[OPC] [XTN] [src]	3	16	8F	F3	-	#N

OPERATION: USP ⇐ USP - 1
(USP) ⇐ src

The user stack pointer is decremented automatically by one and then the contents of the source operand loaded into the location addressed by the decremented user stack pointer.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
PUSHU #20	8F F3 14	1000 1111 1111 0011 0001 0100

If the user stack pointer contains 2000 (decimal), after this instruction user stack pointer location 1999 will contain 20.

PUSHUW

Push Word on User Stack

PUSHUW src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [src,0]	2	12	B6	-	-	RR
	2	12	B6	-	-	rr*
[OPC] [XTN] [src h] [src l]	4	20	8F	C3	-	#NN

OPERATION: USP ⇐ USP - 2
(USP) ⇐ src

The user stack pointer is automatically decremented by two and then the contents of the source operand is loaded into the user stack.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
PUSHUW RR32	B6 20	1011 0110 0010 0000

If the stack pointer contains 2000 (decimal) and register pair 32 contains 6000 (hex), after this instruction the user stack pointer will contain 1998, user stack location 1999 will contain 00 (hex) and user stack location 1998 will contain 60 (hex).

NOTE: See also PEAUW instruction.

PUSHW

Push Word on System Stack

PUSHW src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [src,0]	2	12	74	-	-	RR
	2	12	74	-	-	rr*
[OPC] [XTN] [src h] [src l]	4	20	8F	C1	-	#NN

OPERATION: SSP ⇐ SSP - 2
(SSP) ⇐ src
The system stack pointer is automatically decremented by two and then the contents of the source register pair is loaded into the system stack.

FLAGS: No flag affected.

EXAMPLE:

Instruction	HEX	Binary
PUSHW RR32	74 20	0111 0100 0010 0000

If the system stack pointer contains 2000 (decimal) and register pair 32 contains 6000 (hex), aftert this instruction the system stack pointer will contain 1998, system stack location 1999 will contain 00 (hex) and system stack location 1998 will contain 60 (hex).

NOTE: See also PEAW instruction.

SBC

Subtract with carry (byte) Register, Register

SBC dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst | src]	2	6	22	-	r	r
	2	6	23	-	r	(r)
[OPC] [src] [dst]	3	10	24	-	R	R
	3	10	24	-	r*	R
	3	10	24	-	R	r*
[OPC] [src] [XTN | dst]	3	10	E6	-	(r)	R
	3	10	E6	2	(r)	r*
[OPC] [XTN | src] [dst]	3	10	E7	2	R	(r)

OPERATION: dst ⇐ dst - src - C

The source byte, along with the carry, is subtracted from the destination byte and the result is stored in the destination byte. The source and destination byte can be addressed either directly or indirectly.

FLAGS:
- C: Cleared if carry from MSB of result, set otherwise indicating a borrow.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if the result is less then zero, otherwise cleared.
- V: Set if arithmetic overflow occurred, cleared otherwise.
- D: Always reset to one.
- H: Set if carry from low-order nibble occurred.

EXAMPLE:

Instruction	HEX	Binary
SBC r8,(r4)	23 84	0010 0011 1000 0100

If the carry flag is reset, working register 8 contains 100 (decimal), working register 4 contains 200 (decimal) and register 200 contains 25 (decimal), after this instruction working register 8 will contain 75.

SBC

Subtract with carry (byte) Register, Memory

SBC dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src	Oper
[OPC] [XTN |src,1] [dst]	3	12	72	2	R	(rr)	a
	3	12	72	2	r*	(rr)	a
	3	16	B4	2	R	(rr)+	b
	3	16	B4	2	r*	(rr)+	b
	3	16	C2	2	R	-(rr)	c
	3	16	C2	2	r*	-(rr)	c
[OPC] [ofs,1|src,0] [XTN | dst]	3	22	60	2	r	rr(rr)	a
[OPC] [XTN |src,1] [ofs] [dst]	4	24	7F	2	R	N(rr)	a
	4	24	7F	2	r*	N(rr)	a
[OPC] [XTN | dst] [src h] [src l]	4	18	C4	2	r	NN	a
[OPC] [XTN |src,0] [ofs h] [ofs l] [dst]	5	26	7F	2	R	NN(rr)	a
	5	26	7F	2	r*	NN(rr)	a

OPERATION a: dst ⇐ dst - src - C

The source byte, along with the carry, is subtracted from the destination byte and the result is stored in the destination byte. The destination byte is held in the destination register. The source byte can be addressed directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst - src - C
rr ⇐ rr + 1

The source byte, along with the carry, is subtracted from the destination byte and the result stored in the destination byte. The source byte is in the memory location addressed by the source register pair, the destination byte is in the destination register. The contents of the source register pair are incremented after the SBC has been carried out.

OPERATION c: rr ⇐ rr - 1
dst ⇐ dst - src - C

The source byte, along with the carry, is subtracted from the destination byte and the result is stored in the destination byte. The source byte is in the memory location addressed by the source register pair, the destination byte is in the destination register. The contents of the source register pair are decremented before the SBC is carried out.

SBC

Subtract with carry (byte) Register, Memory

SBC dst,src (Cont'd)

FLAGS:

C: Cleared if carry from MSB of result, set otherwise indicating a borrow.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Always reset to one.
H: Set if carry from low-order nibble occurred.

EXAMPLE:

Instruction	HEX	Binary
SBC r8,6(rr4)	7F 25 06 D8	0111 1111 0010 0101 0000 0110 1101 1000

If the carry flag is set, working register 8 contains 110 (decimal), working register pair 4 contain 4200 (decimal)

SBC

Subtract with carry (byte) Memory, Register

SBC dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src	Oper
[OPC] [XTN |dst,0] [src]	3	18	72	2	(rr)	R	a
	3	18	72	2	(rr)	r*	a
	3	22	B4	2	(rr)+	R	b
	3	22	B4	2	(rr)+	r*	b
	3	22	C2	2	-(rr)	R	c
	3	22	C2	2	-(rr)	r*	c
[OPC] [ofd,1|dst,1] [XTN | src]	3	24	60	2	rr(rr)	r	a
[OPC] [XTN |dst,1] [ofd] [src]	4	26	26	2	N(rr)	R	a
	4	26	26	2	N(rr)	r*	a
[OPC] [XTN | src] [dst h] [dst l]	4	20	C5	2	NN	r	a
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src]	5	28	26	2	NN(rr)	R	a
	5	28	26	2	NN(rr)	r*	a

OPERATION a: dst ⇐ dst - src - C

The source byte, along with the carry, is subtracted from destination byte and the result is stored in the destination byte. The source byte is held in the source register. The destination byte can be addressed directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst - src - C
rr ⇐ rr + 1

The source byte, along with the carry, is subtracted from the destination byte and the result is stored in the destination byte. The source byte is in the source register, the destination byte is in the memory location addressed by the destination register pair. The contents of the destination register pair are incremented after the SBC has been carried out.

OPERATION c: rr ⇐ rr - 1
dst ⇐ dst - src - C

The source byte, along with the carry, is subtracted from the destination byte and the result is stored in the destination byte. The source byte is in the source register , the destination byte is in the memory location addressed by the destination register pair. The contents of the destination register pair are decremented before the SBC is carried out.

SBC

Subtract with carry (byte) Memory, Register

SBC dst,src (Cont'd)

FLAGS:

C: Cleared if carry from MSB of result, set otherwise indicating a borrow.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Always reset to one.
H: Set if carry from low-order nibble occurred.

EXAMPLE:

Instruction	HEX	Binary
SBC (rr8)+,R255	B4 28 FF	1011 0100 0010 1000 1111 1111

If the carry flag is set, working register pair 8 contains 4028 (decimal) memory location 4028 contains 110 (decimal) and register 255 contains 101 (decimal), after this instruction memory location 4028 will contain 8 and working register pair 8 will contain 4029.

SBC

Subtract with carry (byte) Memory, Memory

SBC dst,src

INSTRUCTION FORMAT:

[OPC] [XTN |src,0] [dst,0]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	3	20	73	2	(RR)	(rr)
	3	20	73	2	(rr)*	(rr)

OPERATION: dst ⇐ dst - src - C

The source byte, along with the carry, is subtracted from the destination byte and the result is stored in the destination byte. The source byte is in the memory location addressed by the source register pair, the destination byte is in the memory location addressed by the destination register pair.

FLAGS:
C: Cleared if carry from MSB of result, set otherwise indicating a borrow.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Always reset to one.
H: Set if carry from low-order nibble occurred.

EXAMPLE:

Instruction	HEX	Binary
SBC (rr4),(rr8)	73 28 D4	0111 0011 0010 1000 1101 0100

If the carry flag is set, working register pair 4 contains 2800 (decimal), memory location 2800 contains 46 (decimal), working register pair 8 contains 4200 (deciamal) and memory location 4200 contains 45 (decimal), after this instruction memory location 2800 will contain 0.

SBC

Subtract with carry (byte) All, Immediate

SBC dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst] [src]	3	10	25	-	R	#N
	3	10	25	-	r*	#N
[OPC] [XTN |dst,0] [src]	3	16	F3	2	(rr)	#N
[OPC] [XTN] [src] [dst h] [dst l]	5	24	2F	21	NN	#N

OPERATION: dst ⇐ dst - src - C

The source byte, along with the carry, is subtracted from the destination byte and the result is stored in the destination byte. The source byte is the immediate value in the operand, the destination byte can be in memory or in the register file.

FLAGS:
C: Cleared if carry from MSB of result, set otherwise indicating a borrow.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Always reset to one.
H: Set if carry from low-order nibble occurred.

EXAMPLE:

Instruction	HEX	Binary
SBC (rr8),#32	F3 28 20	1111 0011 0010 1000 0010 0000

If the carry flag is set, working register pair 8 contains 4028 (decimal) and memory location 4028 contains 74 (decimal), after this instruction memory location 4028 will contain 41.

SBCW

Subtract With Carry (Word) - Register, Register

SBCW dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,0|src,0]	2	10	2E	-	rr	rr
[OPC] [src,0] [dst,0]	3	12	27	-	RR	RR
	3	12	27	-	rr*	RR
	3	12	27	-	RR	rr*
[OPC] [src,0] [XTN | dst]	3	14	96	2	(r)	RR
	3	14	96	2	(r)	rr*
[OPC] [XTN | src] [dst,0]	3	14	A6	2	RR	(r)
	3	14	A6	2	rr*	(r)

OPERATION: dst ⇐ dst - src - C

The source word, along with the carry flag, is subtracted from the destination word and the result is stored in the destination word. The source and destination word can be addressed either directly or indirectly.

FLAGS:
- C: Cleared if carry from MSB of result, otherwise set indicating borrow.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if the result is less then zero, otherwise cleared.
- V: Set if arithmetic overflow occurred, cleared otherwise.
- D: Undefined.
- H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
SBCW (r8),RR64	96 40 28	1001 0110 0100 0000 0010 1000

If the carry flag is set, register pair 64 contains 1102 (decimal), working register 8 contains 200 (decimal) and register pair 200 contains 2550 (decimal), after this instruction register pair 200 will hold 1447.

SBCW

Subtract With Carry (Word) - Register, Memory

SBCW dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [dst,0|src,1]	2	16	2E	-	rr	(rr)	a
[OPC] [XTN |src,0] [dst,0]	3	18	7E	2	RR	(rr)	a
[OPC] [XTN |src,1] [dst,0]	3	22	D5	2	RR	(rr)+	b
	3	22	D5	2	rr*	(rr)+	b
	3	24	C3	2	RR	-(rr)	c
	3	24	C3	2	rr*	-(rr)	c
[OPC] [ofs,0|src,0] [XTN |dst,0]	3	24	60	2	rr	rr(rr)	a
[OPC] [XTN |src,1] [ofs] [dst,0]	4	28	86	2	RR	N(rr)	a
	4	28	86	2	rr*	N(rr)	a
[OPC] [XTN |dst,0] [src h] [src l]	4	22	E2	2	rr	NN	a
[OPC] [XTN |src,0] [ofs h] [ofs l] [dst,0]	5	30	86	2	RR	NN(rr)	a
	5	30	86	2	rr*	NN(rr)	a

OPERATION a: dst ⇐ dst - src - C

The source word, along with the carry flag, is subtracted from the destination word and the result is stored in the destination word. The destination word is held in the destination register. The source word can be addressed directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst - src - C
rr ⇐ rr + 2

The source word, along with the carry flag, is subtracted from the destination word and the result is stored in the destination word. The source word is in the memory location addressed by the source register pair, the destination word is in the destination register. The contents of the source register pair are incremented after the subtraction has been carried out.

OPERATION c: rr ⇐ rr - 2
dst ⇐ dst - src - C

The source word, along with the carry flag, is subtracted from the destination word and the result is stored in the destination word. The source word is in the memory location addressed by the source register pair, the destination word is in the destination register. The contents of the source register pair are decremented before the subtraction is carried out.

SBCW

Subtract With Carry (Word) - Register, Memory

SBCW dst,src (Cont'd)

FLAGS:

C: Cleared if carry from MSB of result, otherwise set indicating borrow.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Undefined.
H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
SBCW RR64,-(rr4)	C3 25 40	1100 0011 0010 0101 0100 0000

If the carry flag is set, working register pair 8 contains 1184 (decimal), register pair 64 contains 5000 (decimal) and memory pair 1182 contains 1100 (decimal), after this instruction register pair 64 will contain 3899 and register pair 4 will contain 1182.

SBCW

Subtract With Carry (Word) - Memory, Register

SBCW dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src	Oper
[OPC] [dst,1|src,0]	2	30	2E	-	(rr)	rr	a
[OPC] [XTN |dst,1] [src,0]	3	32	BE	2	(rr)	RR	a
[OPC] [XTN |dst,0] [src,0]	3	34	D5	2	(rr)+	RR	b
	3	34	D5	2	(rr)+	rr*	b
	3	32	C3	2	-(rr)	RR	c
	3	32	C3	2	-(rr)	rr*	c
[OPC] [ofd,0|dst,1] [XTN |src,0]	3	36	60	2	rr(rr)	rr	a
[OPC] [XTN |dst,1] [ofd] [src,1]	4	38	86	2	N(rr)	RR	a
	4	38	86	2	N(rr)	rr*	a
[OPC] [XTN |src,1] [dst h] [dst l]	4	36	E2	2	NN	rr	a
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src,1]	5	40	86	2	NN(rr)	RR	a
	5	40	86	2	NN(rr)	rr*	a

OPERATION a: dst ⇐ dst - src - C

The source word, along with the carry flag, is subtracted from the destination word and the result is stored in the destination word. The source word is held in the source register. The destination word can be addressed directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst - src - C
rr ⇐ rr + 2

The source word, along with the carry flag, is subtracted from the destination word and the result is stored in the destination word. The source word is in the source register, the destination word is in the memory location addressed by the destination register pair. The contents of the destination register pair are incremented after the subtraction has been carried out.

OPERATION c: rr ⇐ rr - 2
dst ⇐ dst - src - C

The source word, along with the carry flag, is subtracted from the destination word and the result is stored in the destination word. The source word is in the source register , the destination word is in the memory location addressed by the destination register pair. The contents of the destination register pair are decremented before the subtraction is carried out.

SBCW

Subtract With Carry (Word) - Memory, Register

SBCW dst,src (Cont'd)

FLAGS:

C: Cleared if carry from MSB of result, otherwise set indicating borrow.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Undefined.
H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
SBCW (rr4)+,RR64	D5 24 40	1101 0101 0010 0100 0100 0000

If the carry flag is set, register pair 64 contains 1250 (decimal), working register pair 4 contains 1064 (decimal) and memory pair 1064 contains 11750, after this instruction has been carried out memory pair 1064 will contain 499 and workig register pair 4 will contain 1066.

SBCW

Subtract With Carry (Word) - Memory, Memory

SBCW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,1|src,1]	2	34	2E	-	(rr)	(rr)

OPERATION: dst ⇐ dst - src - C

The source word, along with the carry flag, is subtracted from the destination word and the result is stored in the destination word. The source word is in the memory location addressed by the source register pair, the destination word is in the memory location addressed by the destination register pair.

FLAGS:
C: Cleared if carry from MSB of result, otherwise set indicating borrow.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Undefined.
H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
SBCW (rr4),(rr6)	2E 57	0010 1110 0101 0111

If the carry flag is zero, working register pair 6 contains 1002 (decimal), memory pair 1002 contains 2300 (decimal), working register pair 4 contains 1060 (decimal) and memory pair 1060 contains 2700 (decimal), after this instruction memory pair 1060 will contain 400.

SBCW

Subtract With Carry (Word) - All, Immediate

SBCW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,1] [src h] [src l]	4	14	27	-	RR	#NN
	4	14	27	-	rr*	#NN
[OPC] [XTN |dst,0] [src h] [src l]	4	34	BE	2	(rr)	#NN
[OPC] [XTN |dst,1] [ofd] [src h] [src l]	5	38	06	2	N(rr)	#NN
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src h] [src l]	6	40	06	2	NN(rr)	#NN
[OPC] [XTN] [src h] [src l] [dst h] [dst l]	6	40	36	21	NN	#NN

OPERATION: dst ⇐ dst - src - C

The source word, along with the carry flag, is subtracted from the destination word and the result is stored in the destination word. The source word is the immediate value in the operand, the destination word can be in memory or in the register file.

FLAGS:
- C: Cleared if carry from MSB of result, otherwise set indicating borrow.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if the result is less then zero, otherwise cleared.
- V: Set if arithmetic overflow occurred, cleared otherwise.
- D: Undefined.
- H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
SBCW RR64,#4268	27 41 10 AC	0010 0111 0100 0001 0001 0000 1010 1100

If the carry flag is zero, register pair 64 contains 5000 (decimal), after this instruction has been carried out register pair 64 will contain the decimal value 732.

SCF

Set Carry Flag

SCF

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC]	1	6	01	-	-	-

OPERATION: C ⇐ 1
The carry flag is set to 1.

FLAGS: C: Set to one.
No other flags affected.

EXAMPLE:

Instruction	HEX	Binary
SCF	01	0000 0001

Regardless of its prior condition, after this instruction the carry flag will be set to one.

SDM

Set Data Memory

SDM

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC]	1	6	FE	-	-	-

OPERATION: Set Data Memory.

This instruction selects the data memory space. After this instruction all instructions that address memory are related to the data space. This instruction sets to one bit 0 of the Flag Register R231.

FLAGS: No flags affected.

SLA

Shift Left Arithmetic Byte

SLA dst

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst | dst]	2	6	42	-	r	-
[OPC] [dst] [dst]	3	10	44	-	R	-
[OPC] [XTN |dst,0] [dst,0]	3	20	73	4	(rr)	-

OPERATION:

dst C ⇐ dst(7)
dst(0) ⇐ 0
dst(n+1) ⇐ dst(n) where n=0-6

The content of the destination register are shifted one place to the left with the most significant bit shifted into the carry flag and a zero shifted into bit 0. The destination register can be directly or indirectly addressed.

FLAGS:

C: Set if MSB set, otherwise cleared.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Always reset to zero.
H: Set if carry from low-order nibble occurred.

EXAMPLE:

Instruction	HEX	Binary
SLA r6	44 66	0100 0100 0110 0110

If working register 6 contains A4 hex , after this instruction the carry bit will be set and working register 8 will contain 68 hex.

NOTE:

This instruction is logically and functionally equivalent to the ADD dst, dst operation and is encoded in an assembler macro function.

SLAW

Shift Left Arithmetic Word

SLAW dst

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst | dst]	2	10	4E	-	rr	-
[OPC] [dst,0] [dst,0]	3	12	47		RR	-
[OPC] [dst,1] [dst,1]	2	32	4E	-	(rr)	-

OPERATION:

dst C ⇐ dst(15)
dst(0) ⇐ 0
dst(n+1) ⇐ dst(n) where n=0-14

The content of the destination register are shifted one place to the left with the most significant bit shifted into the carry flag and a zero shifted into bit 0. The destination register can be directly or indirectly addressed.

FLAGS:

C: Set if MSB set, otherwise cleared.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Undefined
H: Undefined

EXAMPLE:

Instruction	HEX	Binary
SLAW RR4	47 08 08	0100 0111 0000 1000 0000 1000

If working register pair 4 contains A438 hex , after this instruction the carry bit will be set and working register pair 4 will contain 6870 hex.

NOTE:

This intruction is logically and functionally equivalent to the ADD dst, dst operation and is encoded in an assembler macro function.

SPM

Set Program Memory

SPM

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC]	1	6	EE	-	-	-

OPERATION: Set Program Memory.

This instruction selects the program memory space. After this instruction all instructions that address memory are related to the program space. This instruction sets to zero bit 0 of the Flag Register R231.

FLAGS: No flags affected.

SPP

Set Page Pointer

SPP src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [src ,1,0]	2	6	C7	-	-	N

OPERATION: Set to N the Page Pointer Register (R234), where $0 \leq N \leq 63$

This instruction selects one of the 64 pages available to be used for the storage of control information relevant to particular peripherals. Each page is composed of 16 registers based on the top group (F) of the register file. After selecting a page any address on the top group (R240-R255) will be referred to the selected page.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
SPP #5	C7 16	1100 0111 0001 0110

This instruction will select page 5 of paged registers. Then operations addressing group F of the register file are related to page 5.

SRA

Shift Right Arithmetic Byte

SRA dst

INSTRUCTION FORMAT:

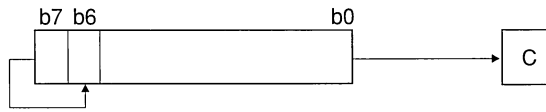
[OPC] [dst]

No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
				dst	src
2	6	E0	-	R	-
2	6	E0	-	r*	-
2	6	E1	-	(R)	-
2	6	E1	-	(r)*	-

OPERATION: dst(7) ⇐ dst(7)
C ⇐ dst(0)
dst(n) ⇐ dst(n+1) Where n=0-6

The contents of the destination register are shifted one place to the right with the bit 0 shifted into the carry flag. Bit 7 (the sign bit) is unchanged but its value is also carried into bit position 6. The destination register can be directly or indirectly addressed.

FLAGS: C: Set if carry from MSB of result, otherwise cleared.



Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if result bit 7 is changed, otherwise cleared.
D: Always reset to zero.
H: Set if carry from low-order nibble occurred.

EXAMPLE:

Instruction	HEX	Binary
SRA (r2)	E1 D2	1110 0001 1101 0010

If the carry flag is one, working register 2 contains 137 (decimal) and register 137 contains 11001100, after this instruction register 137 will contain 11100110 and the carry flag will be zero.

SRAW

Shift Right Arithmetic Word

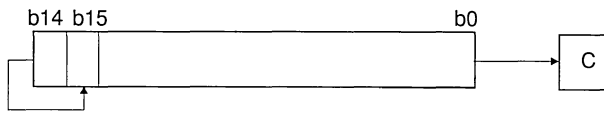
SRAW dst

INSTRUCTION FORMAT:

[OPC] [dst,0]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	2	8	2F	-	RR	-
	2	8	2F	-	rr*	-

OPERATION: dst(15) ⇐ dst(15)
C ⇐ dst(0)
dst(n) ⇐ dst(n+1) where n=0-14

The contents of the destination register pair are shifted one place to the right with bit 0 shifted into the carry flag. Bit 15 (the sign bit) is unchanged but its value is also carried into bit position 14.



FLAGS:
C: Set if carry from LSB (bit 0 was 1).
Z: Undefined.
S: Set if the result is negative, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
SRAW rr2	2F D2	0010 1111 1101 0010

If the carry flag is one, working register pair 2 contains 11001100/11001100B, after this instruction working register pair 2 will contain 11100110/01100110B and the carry flag will be zero.

SRP

Set Register Pointer

SRP src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [src,0,0,0]	2	6	C7	-	-	N

OPERATION: Set Register Pointer

This instruction selects one pair of the thirty-two groups of 8 registers available in the register file. See the section dealing with register pointing possibilities in chapter one for further informations. The pair will always start from the lowest even number equal or lower to the number given in the instruction.

When this instruction is followed by a SRP1 instruction, that is when the mode is changed to the twin working register groups, an 8 register group is selected, equivalent to the SRP0 instruction.

After having selected the window pair every absolutely addressed register that refers to group D (R208-R223) will be referenced to the working window pair.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
SRP #3	C7 18	1100 0111 0001 1000

The first instruction will select the second pair of windows available in the register file (R16-R31). The second instruction therefore will load in the third register of the pair of windows the immediate value, that is it will load in register R19 the value 10 (decimal). The register R213 in the third instruction is equivalent to r5. After this instruction register R21 will contain 20 (decimal).

SRP0

Set Register Pointer 0

SRP0 src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [src 1,0,0]	2	6	C7	-	-	#N

OPERATION: Set Register Pointer 0

This instruction activates the twin register mode and therefore register pointer 0 will refer to one of the thirty-two available groups in the register file.

In particular, after having selected the appropriate window every register between R208 and R215 will be equivalent to working registers r0-r7 and therefore will refer to the window pointed to by RP0.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
SRP0 #3	C7 1C	1100 0111 0001 1100

This instruction will select the window R24-R31. The second instruction will therefore load in the third register of the selected window the immediate data, that is register R27 will contain 10 (decimal). The third instruction will load in the sixth working register, that is R29, the value 20 (decimal).

SRP1

Set Register Pointer 1

SRP1 src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [src 1,0,1]	2	6	C7	-	-	#N

OPERATION: Set Register Pointer 1

This instruction activates the twin register mode and therefore register pointer 1 will refer to one of the thirty-two available in the register file.

In particular after having selected the appropiate window every register between R216 and R223 will be equivalent to working registers r8-r15 and therefore will refer to the window pointed to by RP1.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
SRP #3	SRP1 #2	C7 15

The first instruction will select the window pair R16-R31. With the second instruction the mode will be changed to the twin register groups and register RP0 will point to R24-R31 while register RP1 will point to R16-R23. The first load instruction will therefore refer to register pointer zero since the value of the short register is between 0-7 and will place the value 10 (decimal) into R19. The second load refers to register pointer one since the value of the short register is between 8-15 and will place 20 (decimal) into R26.

SUB

Subtract (byte) Register, Register

SUB dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst | src]	2	6	52	-	r	r
	2	6	53	-	r	(r)
[OPC] [src] [dst]	3	10	54	-	R	R
	3	10	54	-	r*	R
	3	10	54	-	R	r*
[OPC] [src] [XTN | dst]	3	10	E6	5	(r)	R
	3	10	E6	5	(r)	r*
[OPC] [XTN | src] [dst]	3	10	E7	5	R	(r)

OPERATION: dst ⇐ dst - src

The source byte is subtracted from the destination byte and the result is stored in the destination byte. The source and destination byte can be addressed either directly or indirectly.

FLAGS:

C: Cleared if carry from MSB of result, set otherwise indicating a borrow.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Always reset to one.
H: Set if carry from low-order nibble occurred.

EXAMPLE:

Instruction	HEX	Binary
SUB (r8),R255	E6 FF 58	1110 0110 1111 1111 0101 1000

If working register 8 contains 28 (decimal), register 28 contains 43 (decimal) and register 255 contains 21 (decimal), after this instruction register 28 will contain 22.

SUB

Subtract (byte) Register, Memory

SUB dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src	Oper
[OPC] [XTN |src,1] [dst]	3	12	72	5	R	(rr)	a
	3	12	72	5	r*	(rr)	a
	3	16	B4	5	R	(rr)+	b
	3	16	B4	5	r*	(rr)+	b
	3	16	C2	5	R	-(rr)	c
	3	16	C2	5	r*	-(rr)	c
[OPC] [ofs,1|src,0] [XTN | dst]	3	22	60	5	r	rr(rr)	a
[OPC] [XTN |src,1] [ofs] [dst]	4	24	7F	5	R	N(rr)	a
	4	24	7F	5	r*	N(rr)	a
[OPC] [XTN | dst] [src h] [src l]	4	18	C4	5	r	NN	a
[OPC] [XTN |src,0] [ofs h] [ofs l] [dst]	5	26	7F	5	R	NN(rr)	a
	5	26	7F	5	r*	NN(rr)	a

OPERATION a: dst ⇐ dst - src

The source byte is subtracted from the destination byte and the result is stored in the destination byte. The destination byte is held in the destination register. The source byte can be addressed directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst - src
rr ⇐ rr + 1

The source byte is subtracted from the destination byte and the result is stored in the destination byte. The source byte is in the memory location addressed by the source register pair, the destination byte is in the destination register. The contents of the source register pair are incremented after the SUB has been carried out.

OPERATION c: rr ⇐ rr - 1
dst ⇐ dst - src

The source byte is subtracted from the destination byte and the result is stored in the destination byte. The source byte is in the memory location addressed by the source register pair, the destination byte is in the destination register. The contents of the source register pair are decremented before the SUB is carried out.

SUB

Subtract (byte) Register, Memory

SUB dst,src (Cont'd)

FLAGS:

C: Cleared if carry from MSB of result, set otherwise indicating a borrow.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Always reset to one.
H: Set if carry from low-order nibble occurred.

EXAMPLE:

Instruction	HEX	Binary
SUB r8,(rr4)	72 55 D8	0111 0010 0101 0101 1101 1000

If working register 8 contains 213 (decimal), working register pair 4 contain 4200 (decimal) and memory location 4200 contains 25 (decimal), after this instruction register 8 will contain 188.

SUB

Subtract (byte) Memory, Register

SUB dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [XTN |dst,0] [src]	3	18	72	5	(rr)	R	a
	3	18	72	5	(rr)	r*	a
	3	22	B4	5	(rr)+	R	b
	3	22	B4	5	(rr)+	r*	b
	3	22	C2	5	-(rr)	R	c
	3	22	C2	5	-(rr)	r*	c
[OPC] [ofd,1|dst,1] [XTN | src]	3	24	60	5	rr(rr)	r	a
[OPC] [XTN |dst,1] [ofd]	4	26	26	5	N(rr)	R	a
[src]	4	26	26	5	N(rr)	r*	a
[OPC] [XTN | src] [dst h]	4	20	C5	5	NN	r	a
[dst l]							
[OPC] [XTN |dst,0] [ofd h]	5	28	26	5	NN(rr)	R	a
[ofd l] [src]	5	28	26	5	NN(rr)	r*	a

OPERATION a: dst ⇐ dst - src

The source byte is subtracted from destination byte and the result is stored in the destination byte. The source byte is held in the source register. The destination byte can be addressed directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst - src
rr ⇐ rr + 1

The source byte is subtracted from the destination byte and the result is stored in the destination byte. The source byte is in the source register, the destination byte is in the memory location addressed by the destination register pair. The contents of the destination register pair are incremented after the SUB has been carried out.

OPERATION c: rr ⇐ rr - 1
dst ⇐ dst - src

The source byte is subtracted from the destination byte and the result is stored in the destination byte. The source byte is in the source register , the destination byte is in the memory location addressed by the destination register pair. The contents of the destination register pair are decremented before the SUB is carried out.

SUB

Subtract (byte) Memory, Register

SUB dst,src (Cont'd)

FLAGS:

C: Cleared if carry from MSB of result, set otherwise indicating a borrow.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Always reset to one.
H: Set if carry from low-order nibble occurred.

EXAMPLE:

Instruction	HEX	Binary
SUB (rr8),R255	72 58 FF	0111 0010 0101 1000 1111 1111

If working register pair 8 contains 4028 (decimal) memory location 4028 contains 144 (decimal) and register 255 contains 22 (decimal), after this instruction memory location 4028 will contain 122.

SUB

Subtract (byte) Memory, Memory

SUB dst,src

INSTRUCTION FORMAT:

[OPC] [XTN |src,0] [dst,0]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	3	20	73	5	(RR)	(rr)
	3	20	73	5	(rr)*	(rr)

OPERATION: dst ⇐ dst - src

The source byte is subtracted from the destination byte and the result is stored in the destination byte. The source byte is in the memory location addressed by the source register pair, the destination byte is in the memory location addressed by the destination register pair.

FLAGS:

C: Cleared if carry from MSB of result, set otherwise indicating a borrow.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Always reset to one.
H: Set if carry from low-order nibble occurred.

EXAMPLE:

Instruction	HEX	Binary
SUB (rr4),(rr8)	73 58 D4	0111 0011 0101 1000 1101 0100

If working register pair 4 contains 2800 (decimal), memory location 2800 contains 46 (decimal), working register pair 8 contains 4200 (deciamal) and memory location 4200 contains 45 (decimal), after this instruction memory location 2800 will contain 1.

SUB

Subtract (byte) All, Immediate

SUB dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src
[OPC] [dst] [src]	3	10	55	-	R	#N
	3	10	55	-	r*	#N
[OPC] [XTN |dst,0] [src]	3	16	F3	5	(rr)	#N
[OPC] [XTN] [src] [dst h] [dst l]	5	24	2F	51	NN	#N

OPERATION: dst ⇐ dst - src

The source byte is subtracted from the destination byte and the result is stored in the destination byte. The source byte is the immediate value in the operand, the destination byte can be in memory or in the register file.

FLAGS:
- C: Cleared if carry from MSB of result, set otherwise indicating a borrow.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if the result is less then zero, otherwise cleared.
- V: Set if arithmetic overflow occurred, cleared otherwise.
- D: Always reset to one.
- H: Set if carry from low-order nibble occurred.

EXAMPLE:

Instruction	HEX	Binary
SUB (rr8),#32	F3 58 20	1111 0011 0101 1000 0010 0000

If working register pair 8 contains 4028 (decimal) and memory location 4028 contains 74 (decimal), after this instruction memory location 4028 will contain 42.

SUBW

Subtract (Word) - Register, Register

SUBW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,0|src,0]	2	10	5E	-	rr	rr
[OPC] [src,0] [dst,0]	3	12	57	-	RR	RR
	3	12	57	-	rr*	RR
	3	12	57	-	RR	rr*
[OPC] [src,0] [XTN | dst]	3	14	96	5	(r)	RR
	3	14	96	5	(r)	rr*
[OPC] [XTN | src] [dst,0]	3	14	A6	5	RR	(r)
	3	14	A6	5	rr*	(r)

OPERATION: dst ⇐ dst - src

The source word, along with the carry flag, is subtracted from the destination word and the result is stored in the destination word. The source and destination words can be addressed either directly or indirectly.

FLAGS:
- C: Cleared if carry from MSB of result, otherwise set indicating borrow.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if the result is less then zero, otherwise cleared.
- V: Set if arithmetic overflow occurred, cleared otherwise.
- D: Undefined.
- H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
SUBW (r8),RR64	96 40 58	1001 0110 0100 0000 0101 1000

If register pair 64 contains 1102 (decimal), working register 8 contains 200 (decimal) and register pair 200 contains 2550 (decimal), after this instruction register pair 200 will hold 1448.

SUBW

Subtract (Word) - Register, Memory

SUBW dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [dst,0|src,1]	2	16	5E	-	rr	(rr)	a
[OPC] [XTN |src,0] [dst,0]	3	18	7E	5	RR	(rr)	a
[OPC] [XTN |src,1] [dst,0]	3	22	D5	5	RR	(rr)+	b
	3	22	D5	5	rr*	(rr)+	b
	3	24	C3	5	RR	-(rr)	c
	3	24	C3	5	rr*	-(rr)	c
[OPC] [ofs,0|src,0] [XTN |dst,0]	3	24	60	5	rr	rr(rr)	a
[OPC] [XTN |src,1] [ofs] [dst,0]	4	28	86	5	RR	N(rr)	a
	4	28	86	5	rr*	N(rr)	a
[OPC] [XTN |dst,0] [src h] [src l]	4	22	E2	5	rr	NN	a
[OPC] [XTN |src,0] [ofs h] [ofs l] [dst,0]	5	30	86	5	RR	NN(rr)	a
	5	30	86	5	rr*	NN(rr)	a

OPERATION a: dst ⇐ dst - src

The source word, along with the carry flag, is subtracted from the destination word and the result is stored in the destination word. The destination word is held in the destination register. The source word can be addressed directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst - src
rr ⇐ rr + 2

The source word, along with the carry flag, is subtracted from the destination word and the result is stored in the destination word. The source word is in the memory location addressed by the source register pair, the destination word is in the destination register. The contents of the source register pair are incremented after the subtraction has been carried out.

OPERATION c: rr ⇐ rr - 2
dst ⇐ dst - src

The source word, along with the carry flag, is subtracted from the destination word and the result is stored in the destination word. The source word is in the memory location addressed by the source register pair, the destination word is in the destination register. The contents of the source register pair are decremented before the subtraction is carried out.

SUBW

Subtract (Word) - Register, Memory

SUBW dst,src (Cont'd)

FLAGS:

C: Cleared if carry from MSB of result, otherwise set indicating borrow.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Undefined.
H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
SUBW RR64,-(rr4)	C3 55 40	1100 0011 0101 0101 0100 0000

If working register pair 8 contains 1184 (decimal), register pair 64 contains 5000 (decimal) and memory pair 1182 contains 1100 (decimal), after this instruction register pair 64 will contain 3900 and register pair 4 will contain 1182.

SUBW

Subtract (Word) - Memory, Register

SUBW dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [dst,1|src,0]	2	30	5E	-	(rr)	rr	a
[OPC] [XTN |dst,1] [src,0]	3	32	BE	5	(rr)	RR	a
[OPC] [XTN |dst,0] [src,0]	3	34	D5	5	(rr)+	RR	b
	3	34	D5	5	(rr)+	rr*	b
	3	32	C3	5	-(rr)	RR	c
	3	32	C3	5	-(rr)	rr*	c
[OPC] [ofd,0|dst,1] [XTN |src,0]	3	36	60	5	rr(rr)	rr	a
[OPC] [XTN |dst,1] [ofd] [src,1]	4	38	86	5	N(rr)	RR	a
	4	38	86	5	N(rr)	rr*	a
[OPC] [XTN |src,1] [dst h] [dst l]	4	36	E2	5	NN	rr	a
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src,1]	5	40	86	5	NN(rr)	RR	a
	5	40	86	5	NN(rr)	rr*	a

OPERATION a: dst ⇐ dst - src

The source word is subtracted from the destination word and the result is stored in the destination word. The source word is held in the source register. The destination word can be addressed directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst - src
rr ⇐ rr + 2

The source word is subtracted from the destination word and the result is stored in the destination word. The source word is in the source register, the destination word is in the memory location addressed by the destination register pair. The contents of the destination register pair are incremented after the subtraction has been carried out.

OPERATION c: rr ⇐ rr - 2
dst ⇐ dst - src

The source word, along with the carry flag, is subtracted from the destination word and the result is stored in the destination word. The source word is in the source register , the destination word is in the memory location addressed by the destination register pair. The contents of the destination register pair are decremented before the subtraction is carried out.

SUBW

Subtract (Word) - Memory, Register

SUBW dst,src (Cont'd)

FLAGS:

C: Cleared if carry from MSB of result, otherwise set indicating a borrow.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Undefined.
H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
SUBW (rr4)+,RR64	D5 54 40	1101 0101 0101 0100 0100 0000

If register pair 64 contains 1250 (decimal), working register pair 4 contains 1064 (decimal) and memory pair 1064 contains 11750, after this instruction has been carried out memory pair 1064 will contain 500 and workig register pair 4 will contain 1066.

SUBW

Subtract (Word) - Memory, Memory

SUBW dst,src

INSTRUCTION FORMAT:

[OPC] [dst,1|src,1]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src
	2	34	5E	-	(rr)	(rr)

OPERATION: dst ⇐ dst - src

The source word is subtracted from the destination word and the result is stored in the destination word. The source word is in the memory location addressed by the source register pair, the destination word is in the memory location addressed by the destination register pair.

FLAGS:
- C: Cleared if carry from MSB of result, otherwise set indicating borrow.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if the result is less then zero, otherwise cleared.
- V: Set if arithmetic overflow occurred, cleared otherwise.
- D: Undefined.
- H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
SUBW (rr4),(rr6)	5E 57	0101 1110 0101 0111

If working register pair 6 contains 1002 (decimal), memory pair 1002 contains 2300 (decimal), working register pair 4 contains 1060 (decimal) and memory pair 1060 contains 2700 (decimal), after this instruction memory pair 1060 will contain 400.

SUBW

Subtract (Word) - All, Immediate

SUBW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src
[OPC] [dst,1] [src h] [src l]	4	14	57	-	RR	#NN
	4	14	57	-	rr*	#NN
[OPC] [XTN |dst,0] [src h] [src l]	4	34	BE	5	(rr)	#NN
[OPC] [XTN |dst,1] [ofd] [src h] [src l]	5	38	06	5	N(rr)	#NN
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src h] [src l]	6	40	06	5	NN(rr)	#NN
[OPC] [XTN] [src h] [src l] [dst h] [dst l]	6	40	36	51	NN	#NN

OPERATION: dst ⇐ dst - src

The source word is subtracted from the destination word and the result is stored in the destination word. The source word is the immediate value in the operand, the destination word can be in memory or in the register file.

FLAGS:

C: Cleared if carry from MSB of result, otherwise set indicating borrow.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result is less then zero, otherwise cleared.
V: Set if arithmetic overflow occurred, cleared otherwise.
D: Undefined.
H: Undefined.

EXAMPLE:

Instruction	HEX	Binary
SUBW RR64,#4268	57 41 10 AC	0011 0111 0100 0001 0001 0000 1010 1100

If register pair 64 contains 5000 (decimal), after this instruction has been carried out register pair 64 will contain the decimal value 732.

SWAP

Swap Nibbles

SWAP dst

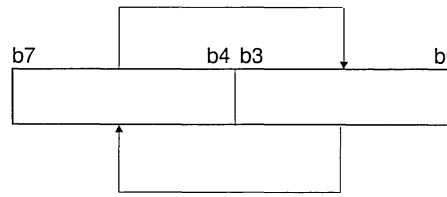
INSTRUCTION FORMAT:

```
[    OPC    ] [    dst    ]
```

No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
				dst	src
2	8	F0	-	R	-
2	8	F0	-	r*	-
2	8	F1	-	(R)	-
2	8	F1	-	(r)*	-

OPERATION: dst(0-3) ⇐- dst(4-7)

The upper and lower nibbles of the destination register are swapped. The destination register can be directly or indirectly addressed.



FLAGS:

C: Undefined.
Z: Set if the result is zero, otherwise cleared.
S: Set if the result bit 7 is set, otherwise cleared.
V: Undefined.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
SWAP R32	F0 20	1111 0000 0010 0000

If register 32 contains 11100111B, after this instruction the contents become 01111110B.

TCM

Test complement under mask (byte) Register, Register

TCM dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst | src]	2	6	82	-	r	r
	2	6	83	-	r	(r)
[OPC] [src] [dst]	3	10	84	-	R	R
	3	10	84	-	r*	R
	3	10	84	-	R	r*
[OPC] [src] [XTN | dst]	3	10	E6	8	(r)	R
	3	10	E6	8	(r)	r*
[OPC] [XTN | src] [dst]	3	10	E7	8	R	(r)

OPERATION: NOT dst AND src

Selected bits in the destination byte are tested for a logical one value. The bits to be tested are selected by setting to one the corresponding bits in the source byte (mask). TCM instruction complements the destination byte, which is then ANDed with the source byte. The zero flag can then be checked to determine the result. The destination byte remains unaltered by this instruction. The source byte is held in the source register and the destination byte in the destination register.

FLAGS:
C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 7 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
TCM r8,R64	84 40 D8	1000 0100 0100 0000 1101 1000

If working register 8 contains 11001100 and register 64 contains 10000101, after this instruction the zero flag will be reset to zero.

TCM

Test complement under mask (byte) Register, Memory

TCM dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [XTN |src,1] [dst]	3	12	72	8	R	(rr)	a
	3	12	72	8	r*	(rr)	a
	3	16	B4	8	R	(rr)+	b
	3	16	B4	8	r*	(rr)+	b
	3	16	C2	8	R	-(rr)	c
	3	16	C2	8	r*	-(rr)	c
[OPC] [ofs,1|src,0] [XTN | dst]	3	22	60	8	r	rr(rr)	a
[OPC] [XTN |src,1] [ofs] [dst]	4	24	7F	8	R	N(rr)	a
	4	24	7F	8	r*	N(rr)	a
[OPC] [XTN | dst] [src h] [src l]	4	18	C4	8	r	NN	a
[OPC] [XTN |src,0] [ofs h] [ofs l] [dst]	5	26	7F	8	R	NN(rr)	a
	5	26	7F	8	r*	NN(rr)	a

OPERATION a: NOT dst AND src

Selected bits in the destination byte are tested for a logical one value. The bits to be tested are selected by setting to one the corresponding bits in the source byte (mask). TCM instruction complements the destination byte, which is then ANDed with the source byte. The zero flag can then be checked to determine the result. The destination byte remains unaltered by this instruction. The source byte is held in the source memory location and the destination byte in the destination register. The destination register is addressed directly, the memory location is addressed either directly, indirectly or by indexing.

OPERATION b: NOT dst AND src
rr ⇐ rr + 1

As operation 'a' (indirect memory addressing only), but the contents of the destination register pair are incremented after the TCM has been carried out.

OPERATION c: rr ⇐ rr - 1 NOT dst AND src

As operation 'a' (indirect memory addressing only), but the contents of the destination register pair are decremented before the TCM is carried out.

TCM

Test complement under mask (byte) Register, Memory

TCM dst,src (Cont'd)

FLAGS:

C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 7 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
TCM r8,4028	C4 88 0F BC	1100 0100 1000 1000 0000 1111 1011 1100

If working register 8 contains 11001100 and memory location 4028 contains 10000101, after this instruction the zero flag will be reset to zero.

TCM

Test complement under mask (byte) Memory, Register

TCM dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [XTN |dst,0] [src]	3	18	72	8	(rr)	R	a
	3	18	72	8	(rr)	r*	a
	3	22	B4	8	(rr)+	R	b
	3	22	B4	8	(rr)+	r*	b
	3	22	C2	8	-(rr)	R	c
	3	22	C2	8	-(rr)	r*	c
[OPC] [ofd,1|dst,1] [XTN | src]	3	24	60	8	rr(rr)	r	a
[OPC] [XTN |dst,1] [ofd] [src]	4	26	26	8	N(rr)	R	a
	4	26	26	8	N(rr)	r*	a
[OPC] [XTN | src] [dst h] [dst l]	4	20	C5	8	NN	r	a
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src]	5	28	26	8	NN(rr)	R	a
	5	28	26	8	NN(rr)	r*	a

OPERATION a: NOT dst AND src

Selected bits in the destination byte are tested for a logical one value. The bits to be tested are selected by setting to one the corresponding bits in the source byte (mask). TCM instruction complements the destination byte, which is then ANDed with the source byte. The zero flag can then be checked to determine the results. The destination byte remains unaltered by this instruction. The source byte is held in the source memory location and the destination byte in the destination register. The source register is addressed directly, the memory location is addressed either directly, indirectly or by indexing.

OPERATION b: NOT dst AND src
rr ⇐ rr + 1

As operation 'a' (indirect memory addressing only), but the contents of the source register pair are incremented after the TCM has been carried out.

OPERATION c: rr ⇐ rr - 1
NOT dst AND src

As operation 'a' (indirect memory addressing only), but the contents of the source register pair are decremented before the TCM is carried out.

TCM

Test complement under mask (byte) Memory, Register

TCM dst,src (Cont'd)

FLAGS:

C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 7 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
TCM 4028,r8	C5 88 0F BC	1100 0101 1000 1000 0000 1111 1011 1100

If memory location 4028 contains 11001100 and working register 8 contains 10000101, after this instruction the zero flag will be reset to zero.

TCM

Test complement under mask (byte) Memory, Memory

TCM dst,src

INSTRUCTION FORMAT:

[OPC] [XTN |src,0] [dst,0]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	3	18	73	8	(RR)	(rr)
	3	18	73	8	(rr)*	(rr)

OPERATION: NOT dst AND src

Selected bits in the destination byte are tested for a logical one value. The bits to be tested are selected by setting to one the corresponding bits in the source byte (mask). TCM instruction complements the destination byte, which is then ANDed with the source byte. The zero flag can then be checked to determine the results. The destination byte remains unaltered by this instruction. The source byte is in the memory location addressed by the source register pair, the destination byte is in the memory location addressed by the destination register pair.

FLAGS:

C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 7 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
TCM (rr4),(rr8)	73 88 D4	0111 0011 1000 1000 1101 0100

If working register pair 4 contains 2800 (decimal), memory location 2800 contains 11001100, working register pair 8 contain 4200 (decimal) and memory location 4200 contains 11001100, after this instruction the zero flag will be set to one.

TCM

Test complement under mask (byte) All, Immediate

TCM dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst] [src]	3	10	85	-	R	#N
	3	10	85	-	r*	#N
[OPC] [XTN |dst,0] [src]	3	16	F3	8	(rr)	#N
[OPC] [XTN] [src] [dst h] [dst l]	5	22	2F	81	NN	#N

OPERATION: NOT dst AND src

Selected bits in the destination byte are tested for a logical one value. The bits to be tested are selected by setting to one the corresponding bits in the source byte (mask). TCM instruction complements the destination byte, which is then ANDed with the source byte. The zero flag can then be checked to determine the results. The destination byte remains unaltered by this instruction. The source byte is the immediate value in the operand, the destination byte can be in memory or in the register file.

FLAGS:

C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 7 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
TCM (rr8),#32	F3 88 20	1111 0011 1000 1000 0010 0000

If working register pair 8 contains 4028 (decimal) and memory location 4028 contains 11101100, after this instruction the zero flag will be set to one.

TCMW

Test Complement Under Mask (Word) - Register, Register

TCMW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,0|src,0]	2	10	8E	-	rr	rr
[OPC] [src,0] [dst,0]	3	12	87	-	RR	RR
	3	12	87	-	rr*	RR
	3	12	87	-	RR	rr*
[OPC] [src,0] [XTN | dst]	3	14	96	8	(r)	RR
	3	14	96	8	(r)	rr*
[OPC] [XTN | src] [dst,0]	3	14	A6	8	RR	(r)
	3	14	A6	8	rr*	(r)

OPERATION: NOT dst AND src

Selected bits in the destination word are tested for a logical one value. The bits to be tested are selected by setting to one the corresponding bit in the source word (mask). The TCMW instruction complements the destination word, which is then ANDed with source word. The zero flag can then be checked to determine the result. The destination word remains unaltered by this instruction. The source and the destination word can be addressed either directly or indirectly.

FLAGS:

C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 15 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
TCMW (r8),RR64	96 40 88	1001 0110 0100 0000 1000 1000

If register pair 64 contains 11001100/11001100B, working register 8 contains 200 (decimal) and register pair 200 contains 01001000/01001000B, after this instruction the zero flag will be reset to zero.

TCMW

Test Complement Under Mask (Word) - Register, Memory

TCMW dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [dst,0|src,1]	2	16	8E	-	rr	(rr)	a
[OPC] [XTN |src,0] [dst,0]	3	18	7E	8	RR	(rr)	a
[OPC] [XTN |src,1] [dst,0]	3	22	D5	8	RR	(rr)+	b
	3	22	D5	8	rr*	(rr)+	b
	3	24	C3	8	RR	-(rr)	c
	3	24	C3	8	rr*	-(rr)	c
[OPC] [ofs,0|src,0] [XTN |dst,0]	3	24	60	8	rr	rr(rr)	a
[OPC] [XTN |src,1] [ofs] [dst,0]	4	28	86	8	RR	N(rr)	a
	4	28	86	8	rr*	N(rr)	a
[OPC] [XTN |dst,0] [src h] [src l]	4	28	E2	8	rr	NN	a
[OPC] [XTN |src,0] [ofs h] [ofs l] [dst,0]	5	30	86	8	RR	NN(rr)	a
	5	30	86	8	rr*	NN(rr)	a

OPERATION a: NOT dst AND src

Selected bits in the destination word are tested for a logical one value. The bits to be tested are selected by setting to one the corresponding bit in the source word (mask). The TCMW instruction complements the destination word, which is then ANDed with source word. The zero flag can then be checked to determine the result. The destination word remains unaltered by this instruction.

The source word is held in the source memory pair and the destination word in the destination register pair. The destination register pair is addressed directly, the memory pair is addressed either directly, indirectly or by indexing.

OPERATION b: NOT dst AND src
rr ⇐ rr + 2

As operation 'a' (indirect memory addressing only), but the contents of the destination register pair are incremented after the TCMW has been carried out.

OPERATION c: rr ⇐ rr - 2
NOT dst AND src

As operation 'a' (indirect memory addressing only), but the contents of the destination register pair are decremented before the TCMW is carried out.

TCMW

Test Complement Under Mask (Word) - Register, Memory

TCMW dst,src (Cont'd)

FLAGS:
- C: Unaffected.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if result bit 15 is set, otherwise cleared.
- V: Always reset to zero.
- D: Unaffected.
- H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
TCMW RR64,-(rr4)	C3 85 40	1100 0011 0011 0101 0100 0000

If working register pair 4 contains 1184 (decimal), register pair 64 contains 11001100/11001100B and memory pair 1182 contains 11001100/11001100B, after this instruction the zero flag will be set and register pair 4 will contain 1182.

TCMW

Test Complement Under Mask (Word) - Memory, Register

TCMW dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [dst,1|src,0]	2	26	8E	-	(rr)	rr	a
[OPC] [XTN |dst,1] [src,0]	3	28	BE	8	(rr)	RR	a
[OPC] [XTN |dst,0] [src,0]	3	30	D5	8	(rr)+	RR	b
	3	30	D5	8	(rr)+	rr*	b
	3	30	C3	8	-(rr)	RR	c
	3	30	C3	8	-(rr)	rr*	c
[OPC] [ofd,0|dst,1] [XTN |src,0]	3	32	60	8	rr(rr)	rr	a
[OPC] [XTN |dst,1] [ofd] [src,1]	4	34	86	8	N(rr)	RR	a
	4	34	86	8	N(rr)	rr*	a
[OPC] [XTN |src,1] [dst h] [dst l]	4	30	E2	8	NN	rr	a
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src,1]	5	36	86	8	NN(rr)	RR	a
	5	36	86	8	NN(rr)	rr*	a

OPERATION a: NOT dst AND src

Selected bits in the destination word are tested for a logical one value. The bits to be tested are selected by setting to one the corresponding bit in the source word (mask). The TCMW instruction complements the destination word, which is then ANDed with the source word. The zero flag can then be checked to determine the result. The destination word remains unaltered by this instruction.

The source word is held in the source register pair and the destination word in the destination memory pair. The source register pair is addressed directly, the memory pair is addressed either directly, indirectly or by indexing.

OPERATION b: NOT dst AND src
rr ⇐ rr + 2

As operation 'a' (indirect memory addressing only), but the contents of the source register pair are incremented after the TCMW has been carried out.

OPERATION c: rr ⇐ rr - 2
NOT dst AND src

As operation 'a' (indirect memory addressing only), but the contents of the source register pair are decremented before the TCMW is carried out.

TCMW

Test Complement Under Mask (Word) - Memory, Register

TCMW dst,src (Cont'd)

FLAGS:
C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 15 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
TCMW (rr8),RR64	BE 89 40	1011 1110 1000 1001 0100 0000

If register pair 64 contains 11001100/11001100B, working register pair 8 contains 2000 (decimal) and memory pair 2000 contains 11001100/11001100B, after this instruction the zero flag will be set.

TCMW

Test Complement Under Mask (Word) - Memory, Memory

TCMW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,1|src,1]	2	30	8E	-	(rr)	(rr)

OPERATION: NOT dst AND src

Selected bits in the destination word are tested for a logical one value. The bits to be tested are selected by setting to one the corresponding bit in the source word (mask). The TCMW instruction complements the destination word, which is then ANDed with source word. The zero flag can then be checked to determine the result. The destination word remains unaltered by this instruction.

The source word is held in the source memory pair and the destination word in the destination register pair. The source word is in the memory pair addressed by the contents of the source register pair. the destination word is in the memory pair addressed by the contents of the destination register pair.

FLAGS:

C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 15 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
TCMW (rr4),(rr6)	8E 57	1000 1110 0101 0111

If working register pair 6 contains 1002 (decimal), memory location pair 1002 contains 11001100/11001100B, working register pair 4 contains 1060 (decimal) and memory pair 1060 contains 11001100/11001100B, after this instruction the zero flag wil be set.

TCMW

Test Complement Under Mask (Word) - All, Immediate

TCMW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,1] [src h] [src l]	4	14	87	-	RR	#NN
	4	14	87	-	rr*	#NN
[OPC] [XTN |dst,0] [src h] [src l]	4	30	BE	8	(rr)	#NN
[OPC] [XTN |dst,1] [ofd] [src h] [src l]	5	34	06	8	N(rr)	#NN
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src h] [src l]	6	36	06	8	NN(rr)	#NN
[OPC] [XTN] [src h] [src l] [dst h] [dst l]	6	36	36	81	NN	#NN

OPERATION: NOT dst AND src

Selected bits in the destination word are tested for a logical one value. The bits to be tested are selected by setting to one the corresponding bit in the source word (mask). The TCMW instruction complements the destination word, which is then ANDed with source word. The zero flag can then be checked to determine the result. The destination word remains unaltered by this instruction.

The source word is the immediate value held in the operand. The destination word can be in memory or register file.

FLAGS:

C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 15 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
TCMW RR64,#52428	87 41 CC CC	0011 0111 0100 0001 1100 1100 1100 1100

If register pair 64 contains 01001000/01001000B, after this instruction has been carried out the zero flag will be reset.

TM

TM Test under mask (byte) Register, Register

TM dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst | src]	2	6	A2	-	r	r
	2	6	A3	-	r	(r)
[OPC] [src] [dst]	3	10	A4	-	R	R
	3	10	A4	-	r*	R
	3	10	A4	-	R	r*
[OPC] [src] [XTN | dst]	3	10	E6	A	(r)	R
	3	10	E6	A	(r)	r*
[OPC] [XTN | src] [dst]	3	10	E7	A	R	(r)

OPERATION: dst AND src

Selected bits in the destination byte are tested for a logical zero value. The bits to be tested are selected by setting to one the corresponding bits in the source byte (mask) which is then ANDed with the destination byte. The zero flag can then be checked to determine the results. The destination byte remains unaltered by this instruction. The source byte is held in the source register and the destination byte in the destination register.

FLAGS:

C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 7 is set, otherwise cleared.
V: Always reset
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
TM r8,R64	A4 40 D8	1010 0100 0100 0000 1101 1000

If working register 8 contains 01001100 and register 64 contains 00110011, after this instruction the zero flag will be reset to one.

TM

TM Test under mask (byte) Register, Memory

TM dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [XTN |src,1] [dst]	3	12	72	A	R	(rr)	a
	3	12	72	A	r*	(rr)	a
	3	16	B4	A	R	(rr)+	b
	3	16	B4	A	r*	(rr)+	b
	3	16	C2	A	R	-(rr)	c
	3	16	C2	A	r*	-(rr)	c
[OPC] [ofs,1|src,0] [XTN | dst]	3	22	60	A	r	rr(rr)	a
[OPC] [XTN |src,1] [ofs] [dst]	4	24	7F	A	R	N(rr)	a
	4	24	7F	A	r*	N(rr)	a
[OPC] [XTN | dst] [src h] [src l]	4	18	C4	A	r	NN	a
[OPC] [XTN |src,0] [ofs h] [ofs l] [dst]	5	26	7F	A	R	NN(rr)	a
	5	26	7F	A	r*	NN(rr)	a

OPERATION a: dst AND src

Selected bits in the destination byte are tested for a logical zero value. The bits to be tested are selected by setting to one the corresponding bits in the source byte (mask) which is then ANDed with the destination byte. The zero flag can then be checked to determine the results. The destination byte remains unaltered by this instruction. The source byte is held in the source memory location and the destination byte in the destination register. The destination register is addressed directly, the memory location is addressed either directly, indirectly or by indexing.

OPERATION b: dst AND src
rr ⇐ rr + 1

As operation 'a' (indirect memory addressing only), but the contents of the destination register pair are incremented after the TM has been carried out.

OPERATION c: rr ⇐ rr - 1
dst AND src

As operation 'a' (indirect memory addressing only), but before the TM is carried out the contents of the destination register pair are decremented.

TM

TM Test under mask (byte) Register, Memory

TM dst,src (Cont'd)

FLAGS:
- C: Unaffected.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if result bit 7 is set, otherwise cleared.
- V: Always reset
- D: Unaffected.
- H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
TM r8,4028	C4 A8 0F BC	1100 0100 1010 1000 0000 1111 1011 1100

If working register 8 contains 11001100 and memory location 4028 contains 10000101, after this instruction the zero flag will be set.

TM

Test under mask (byte) Memory, Register

TM dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src	Oper
[OPC] [XTN |dst,0] [src]	3	18	72	A	(rr)	R	a
	3	18	72	A	(rr)	r*	a
	3	22	B4	A	(rr)+	R	b
	3	22	B4	A	(rr)+	r*	b
	3	22	C2	A	-(rr)	R	c
	3	22	C2	A	-(rr)	r*	c
[OPC] [ofd,1|dst,1] [XTN | src]	3	24	60	A	rr(rr)	r	a
[OPC] [XTN |dst,1] [ofd] [src]	4	26	26	A	N(rr)	R	a
	4	26	26	A	N(rr)	r*	a
[OPC] [XTN | src] [dst h] [dst l]	4	20	C5	A	NN	r	a
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src]	5	28	26	A	NN(rr)	R	a
	5	28	26	A	NN(rr)	r*	a

OPERATION a: dst AND src

Selected bits in the destination byte are tested for a logical zero value. The bits to be tested are selected by setting to one the corresponding bits in the source byte (mask) which is then ANDed with the destination byte. The zero flag can then be checked to determine the results. The destination byte remains unaltered by this instruction. The source byte is held in the source memory location and the destination byte in the destination register. The source register is addressed directly, the memory location is addressed either directly, indirectly or by indexing.

OPERATION b: dst AND src
rr ⇐ rr + 1

As operation 'a' (indirect memory addressing only), but the contents of the source register pair are incremented after the TM has been carried out.

OPERATION c: rr ⇐ rr - 1
dst AND src

As operation 'a' (indirect memory addressing only), but before the TM is carried out the contents of the source register pair are decremented.

TM

Test under mask (byte) Memory, Register

TM dst,src (Cont'd)

FLAGS:
C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 7 is set, otherwise cleared.
V: Always reset
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
TM 4028,r8	C5 A8 0F BC	1100 0101 1010 1000 0000 1111 1011 1100

If working register 8 contains 11001100 and memory location 4028 contains 10000101, after this instruction the zero flag will be reset to zero.

TM

Test under mask (byte) Memory, Memory

TM dst,src

INSTRUCTION FORMAT:

[OPC] [XTN |src,0] [dst,0]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	3	18	73	A	(RR)	(rr)
	3	18	73	A	(rr)*	(rr)

OPERATION: dst AND src

Selected bits in the destination byte are tested for a logical zero value. The bits to be tested are selected by setting to one the corresponding bits in the source byte (mask) which is then ANDed with the destination byte. The zero flag can then be checked to determine the results. The destination byte remains unaltered by this instruction. The source byte is in the memory location addressed by the source register pair, the destination byte is in the memory location addressed by the destination register pair.

FLAGS:

C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 7 is set, otherwise cleared.
V: Always reset
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
TM (rr4),(rr8)	73 A8 D4	0111 0011 1010 1000 1101 0100

If working register pair 4 contains 2800 (decimal), memory location 2800 contains 11001100, working register pair 8 contains 4200 (decimal) and memory location 4200contains 00110011, after this instruction the zero flag will be set to one.

TM

Test under mask (byte) All, Immediate

TM dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst] [src]	3	10	A5	-	R	#N
	3	10	A5	-	r*	#N
[OPC] [XTN |dst,0] [src]	3	16	F3	A	(rr)	#N
[OPC] [XTN] [src] [dst h] [dst l]	5	22	2F	A1	NN	#N

OPERATION: dst AND src

Selected bits in the destination byte are tested for a logical zero value. The bits to be tested are selected by setting to one the corresponding bits in the source byte (mask) which is then ANDed with the destination byte. The zero flag can then be checked to determine the results. The destination byte remains unaltered by this instruction. The source byte is the immediate value in the operand, the destination byte can be in memory or in the register file.

FLAGS:

C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 7 is set, otherwise cleared.
V: Always reset
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
TM (rr8),#32	F3 A8 20	1111 0011 1010 1000 0010 0000

If working register pair 8 contains 4028 (decimal) and memory location 4028 contains 11101100, after this instruction the zero flag will be reset to zero.

TMW

Test Under Mask (Word) - Register, Register

TMW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode dst	Addr Mode src
[OPC] [dst,0|src,0]	2	10	AE	-	rr	rr
[OPC] [src,0] [dst,0]	3	12	A7	-	RR	RR
	3	12	A7	-	rr*	RR
	3	12	A7	-	RR	rr*
[OPC] [src,0] [XTN | dst]	3	14	96	A	(r)	RR
	3	14	96	A	(r)	rr*
[OPC] [XTN | src] [dst,0]	3	14	A6	A	RR	(r)
	3	14	A6	A	rr*	(r)

OPERATION: dst AND src

Selected bits in the destination word are tested for a logical zero value. The bits to be tested are selected by setting to one the corresponding bits in the source word (mask) which is then ANDed with the destination word. The zero flag can then be checked to determine the result. The destination word remains unaltered by this instruction. The source and the destination word can be addressed either directly or indirectly.

FLAGS:

C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 15 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
TMW (r8),RR64	96 40 A8	1001 0110 0100 0000 1010 1000

If register pair 64 contains 11001100/11001100B, working register 8 contains 200 (decimal) and register pair 200 contains 00110011/00110011B, after this instruction the zero flag will be reset to zero.

TMW

Test Under Mask (Word) - Register, Memory

TMW dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [dst,0|src,1]	2	16	AE	-	rr	(rr)	a
[OPC] [XTN |src,0] [dst,0]	3	18	7E	A	RR	(rr)	a
[OPC] [XTN |src,1] [dst,0]	3	22	D5	A	RR	(rr)+	b
	3	22	D5	A	rr*	(rr)+	b
	3	24	C3	A	RR	-(rr)	c
	3	24	C3	A	rr*	-(rr)	c
[OPC] [ofs,0|src,0] [XTN |dst,0]	3	24	60	A	rr	rr(rr)	a
[OPC] [XTN |src,1] [ofs] [dst,0]	4	28	86	A	RR	N(rr)	a
	4	28	86	A	rr*	N(rr)	a
[OPC] [XTN |dst,0] [src h] [src l]	4	22	E2	A	rr	NN	a
[OPC] [XTN |src,0] [ofs h] [ofs l] [dst,0]	5	30	86	A	RR	NN(rr)	a
	5	30	86	A	rr*	NN(rr)	a

OPERATION a: dst AND src

Selected bits in the destination word are tested for a logical zero value. The bits to be tested are selected by setting to one the corresponding bits in the source word (mask) which is then ANDed with the destination word. The zero flag can then be checked to determine the result. The destination word remains unaltered by this instruction. The source word is held in the source memory pair and the destination word in the destination register pair. The destination register pair is addressed directly, the memory pair is addressed either directly, indirectly or by indexing.

OPERATION b: dst AND src
rr ⇐ rr + 2

As operation 'a' (indirect memory addressing only), but the contents of the destination register pair are incremented after the TMW has been carried out.

OPERATION c: rr ⇐ rr - 2
dst AND src

As operation 'a' (indirect memory addressing only), but the contents of the destination register pair are decremented before the TMW is carried out.

TMW

Test Under Mask (Word) - Register, Memory

TMW dst,src (Cont'd)

FLAGS:

C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 15 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
TMW RR64,-(rr4)	C3 A5 40	1100 0011 1010 0101 0100 0000

If working register pair 4 contains 1184 (decimal), register pair 64 contains 11001100/11001100B and memory pair 1182 contains 11001100/11001100B, after this instruction the zero flag will be set and register pair 4 will contain 1182.

TMW

Test Under Mask (Word) - Memory, Register

TMW dst,src

INSTRUCTION FORMAT:

				No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
								dst	src	
[OPC]	[dst,1|src,0]			2	26	AE	-	(rr)	rr	a
[OPC]	[XTN |dst,1]	[src,0]		3	28	BE	A	(rr)	RR	a
[OPC]	[XTN |dst,0]	[src,0]		3	30	D5	A	(rr)+	RR	b
				3	30	D5	A	(rr)+	rr*	b
				3	30	C3	A	-(rr)	RR	c
				3	30	C3	A	-(rr)	rr*	c
[OPC]	[ofd,0|dst,1]	[XTN |src,0]		3	32	60	A	rr(rr)	rr	a
[OPC] [src,1]	[XTN |dst,1]	[ofd]		4	34	86	A	N(rr)	RR	a
				4	34	86	A	N(rr)	rr*	a
[OPC] [dst 1]	[XTN |src,1]	[dst h]		4	30	E2	A	NN	rr	a
[OPC] [ofd 1]	[XTN |dst,0]	[ofd h]	[src,1]	5	36	86	A	NN(rr)	RR	a
				5	36	86	A	NN(rr)	rr*	a

OPERATION a: dst AND src

Selected bits in the destination word are tested for a logical zero value. The bits to be tested are selected by setting to one the corresponding bits in the source word (mask) which is then ANDed with the destination word. The zero flag can then be checked to determine the result. The destination word remains unaltered by this instruction.

The source word is held in the source register pair and the destination word in the destination memory pair. The source register pair is addressed directly, the memory pair is addressed either directly, indirectly or by indexing.

OPERATION b: dst AND src
rr ⇐ rr + 2

As operation 'a' (indirect memory addressing only), but the contents of the source register pair are incremented after the TMW has been carried out.

OPERATION c: rr ⇐ rr - 2
dst AND src

As operation 'a' (indirect memory addressing only), but the contents of the source register pair are decremented before the TMW is carried out.

TMW

Test Under Mask (Word) - Memory, Register

TMW dst,src (Cont'd)

FLAGS:
C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 15 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
TMW (rr8),RR64	BE A9 40	1011 1110 1010 1001 0100 0000

If register pair 64 contains 11001100/11001100B, working register pair 8 contains 2000 (decimal) and memory pair 2000 contains 11001100/11001100B, after this instruction the zero flag will be set.

TMW

Test Under Mask (Word) - Memory, Memory

TMW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,1|src,1]	2	30	AE	-	(rr)	(rr)

OPERATION: dst AND src

Selected bits in the destination word are tested for a logical zero value. The bits to be tested are selected by setting to one the corresponding bits in the source word (mask) which is then ANDed with the destination word. The zero flag can then be checked to determine the result. The destination word remains unaltered by this instruction.

The source word is held in the source memory pair and the destination word in the destination register pair. The source word is in the memory pair addressed by the contents of the source register pair. the destination word is in the memory pair addressed by the contents of the destination register pair.

FLAGS:

C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 15 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
TMW (rr4),(rr6)	AE 57	1010 1110 0101 0111

If working register pair 6 contains 1002 (decimal), memory location pair 1002 contains 11001100/11001100B, working register pair 4 contains 1060 (decimal) and memory pair 1060 contains 11001100/11001100B, after this instruction the zero flag wil be set.

TMW

Test Under Mask (Word) - All, Immediate

TMW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst,1] [src h] [src l]	4	14	A7	-	RR	#NN
	4	14	A7	-	rr*	#NN
[OPC] [XTN |dst,0] [src h] [src l]	4	30	BE	A	(rr)	#NN
[OPC] [XTN |dst,1] [ofd] [src h] [src l]	5	34	06	A	N(rr)	#NN
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src h] [src l]	6	36	06	A	NN(rr)	#NN
[OPC] [XTN] [src h] [src l] [dst h] [dst l]	6	36	36	A1	NN	#NN

OPERATION: dst AND src

Selected bits in the destination word are tested for a logical zero value. The bits to be tested are selected by setting to one the corresponding bits in the source word (mask) which is then ANDed with the destination word. The zero flag can then be checked to determine the result. The destination word remains unaltered by this instruction.

The source word is the immediate value held in the operand. the destination word can be in memory or register file.

FLAGS:
- C: Unaffected.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if result bit 15 is set, otherwise cleared.
- V: Always reset to zero.
- D: Unaffected.
- H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
TMW RR64,#52428	A7 41 CC CC	0011 0111 0100 0001 1100 1100 1100 1100

If register pair 64 contains 01001000/01001000B, after this instruction has been carried out the zero flag will be reset.

WFI

Wait For Interrupt

WFI

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [XTN]	2	18	EF	01	-	-

OPERATION : This instruction suspends program operation until an interrupt is acknowledged, although DMA requests are still serviced.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
WFI	EF 01	1110 1111 0000 0001

The program is suspended until an interrupt occurs.

XCH

Exchange Registers

XCH dst,src

INSTRUCTION FORMAT:

[OPC] [src] [dst]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	3	12	16	-	R	R
	3	12	16	-	R	r*
	3	12	16	-	r*	R
	3	12	16	-	r*	r*

OPERATION: dst ⇐ src

The contents of the destination register are loaded into the source register and the contents of the source register loaded into the destination register.

FLAGS: No flags affected.

EXAMPLE:

Instruction	HEX	Binary
XCH r2,r4	16 D4 D2	0001 0110 1101 0100 1101 0010

If working register 2 contains 26 (decimal) and working register 4 contains 100 (decimal), after this instruction register 2 will contain 100 and register 4 will contain 26.

XOR

Exclusive OR (byte) Register, Register

XOR dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
[OPC] [dst | src]	2	6	62	-	r	r
	2	6	63	-	r	(r)
[OPC] [src] [dst]	3	10	64	-	R	R
	3	10	64	-	r*	R
	3	10	64	-	R	r*
[OPC] [src] [XTN | dst]	3	10	E6	6	(r)	R
	3	10	E6	6	(r)	r*
[OPC] [XTN | src] [dst]	3	10	E7	6	R	(r)

OPERATION: dst ⇐ dst XOR src

The contents of the source are XORed with the destination byte and the results stored in the destination byte. The contents of the source are not affected.

FLAGS:
C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 7 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
XOR r8,R64	64 40 D8	0110 0100 0100 0000 1101 1000

If working register 8 contains 11001100 and register 64 contains 10000101, after this instruction working register 8 will contain 01001001.

XOR

Exclusive OR (byte) Register, Memory

XOR dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [XTN |src,1] [dst]	3	12	72	6	R	(rr)	a
	3	12	72	6	r*	(rr)	a
	3	16	B4	6	R	(rr)+	b
	3	16	B4	6	r*	(rr)+	b
	3	16	C2	6	R	-(rr)	c
	3	16	C2	6	r*	-(rr)	c
[OPC] [ofs,1|src,0] [XTN | dst]	3	22	60	6	r	rr(rr)	a
[OPC] [XTN |src,1] [ofs] [dst]	4	24	7F	6	R	N(rr)	a
	4	24	7F	6	r*	N(rr)	a
[OPC] [XTN | dst] [src h] [src l]	4	18	C4	6	r	NN	a
[OPC] [XTN |src,0] [ofs h] [ofs l] [dst]	5	26	7F	6	R	NN(rr)	a
	5	26	7F	6	r*	NN(rr)	a

OPERATION a: dst ⇐ dst XOR src

The source byte is XORed with the destination byte and the result stored in the destination byte. The destination register is addressed directly, the memory location is addressed either directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst XOR src
rr ⇐ rr + 1

The contents of the memory location addressed by the source register pair are XORed with the contents of the directly addressed destination register the result stored in the destination byte. The contents of the source register pair are incremented after the XOR has been carried out.

OPERATION c: rr ⇐ rr - 1
dst ⇐ dst XOR src

The contents of the source register pair are decremented and then the contents of the memory location addressed by the source register pair are XORed with the contents of the directly addressed destination register. The result is stored in the destination byte.

XOR

Exclusive OR (byte) Register, Memory

XOR dst,src (Cont'd)

FLAGS:

C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 7 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
XOR r8,(rr4)	72 65 D8	0111 0010 0110 0101 1101 1000

If working register 8 contains 11001100, working register pair 4 contains 4200 (decimal) and memory location 4200.

XOR

Exclusive OR (byte) Memory, Register

XOR dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode		Oper
					dst	src	
[OPC] [XTN |dst,0] [src]	3	18	72	6	(rr)	R	a
	3	18	72	6	(rr)	r*	a
	3	22	B4	6	(rr)+	R	b
	3	22	B4	6	(rr)+	r*	b
	3	22	C2	6	-(rr)	R	c
	3	22	C2	6	-(rr)	r*	c
[OPC] [ofs,d|dst,1] [XTN | src]	3	24	60	6	rr(rr)	r	a
[OPC] [XTN |dst,1] [ofd] [src]	4	26	26	6	N(rr)	R	a
	4	26	26	6	N(rr)	r*	a
[OPC] [XTN | src] [dst h] [dst l]	4	20	C5	6	NN	r	a
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src]	5	28	26	6	NN(rr)	R	a
	5	28	26	6	NN(rr)	r*	a

OPERATION a: dst ⇐ dst XOR src

The source byte is XORed with the destination byte and the result stored in the destination byte. The source registers are addressed directly, the memory location are addressed either directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst XOR src
rr ⇐ rr + 1

The contents of the memory location addressed by the destination register pair (destination byte) are XORed with the contents of the directly addressed source register the result stored in the destination byte. The contents of the destination register pair are incremented after the XOR has been carried out.

OPERATION c: rr ⇐ rr - 1
dst ⇐ dst XOR src

The contents of the destination register pair are decremented and then the contents of the memory location addressed by the destination register pair (destination byte) are XORed with the contents of the directly addressed source register. The result is stored in the destination byte.

XOR

Exclusive OR (byte) Memory, Register

XOR dst,src (Cont'd)

FLAGS:

C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 7 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
XOR 4028,r8	C5 18 0F BC	1100 0101 0001 1000 0000 1111 1011 1100

If memory location 4028 contains 11001100 and working register 8 contains 10000101, after this instruction memory.

XOR

Exclusive OR (byte) Memory, Memory

XOR dst,src

INSTRUCTION FORMAT:

[OPC] [XTN |src,0] [dst,0]	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Addr Mode	
					dst	src
	3	20	73	6	(RR)	(rr)
	3	20	73	6	(rr)*	(rr)

OPERATION : dst ⇐ dst XOR src

The contents of the memory addressed by the source register pair are XORed with the content of the memory location addressed by the destination register pair. The source and destination addresses are for the word high order byte.

FLAGS:

C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 7 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
XOR (rr4),(rr8)	73 68 D4	0111 0011 0110 1000 1101 0100

If working register pair 4 contains 2800 (decimal), memory location 2800 contains 11001100, working register pair 8 contains 4200 (decimal) and memory location 4200 contains 1100011, after this instruction memory location 2800 will contain 00000000.

XOR

Exclusive OR (byte) All, Immediate

XOR dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Adress Mode	
					dst	src
[OPC] [dst] [src]	3	10	65	-	R	#N
	3	10	65	-	r*	#N
[OPC] [XTN |dst,0] [src]	3	16	F3	6	(rr)	#N
[OPC] [XTN] [src] [dst h] [dst l]	5	24	2F	61	NN	#N

OPERATION: dst ⇐ dst XOR src

The value #N is XORed with the content of the destination register or memory location (destination byte) and stored in the destination byte.

FLAGS:

C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 7 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
XOR (rr8),#32	F3 68 20	1111 0011 0110 1000 0010 0000

If working register pair 8 contains 4028 (decimal) and memory location 4028 contains 11001100, after this instruction memory location 4028 will contain 00100000.

XORW

Exclusive OR (Word) - Register, Register

XORW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Adress Mode	
					dst	src
[OPC] [dst,0|src,0]	2	10	6E	-	rr	rr
[OPC] [src,0] [dst,0]	3	12	67	-	RR	RR
	3	12	67	-	rr*	RR
	3	12	67	-	RR	rr*
[OPC] [src,0] [XTN | dst]	3	14	96	6	(r)	RR
	3	14	96	6	(r)	rr*
[OPC] [XTN | src] [dst,0]	3	14	A6	6	RR	(r)
	3	14	A6	6	rr*	(r)

OPERATION: dst ⇐ dst XOR src

The source word is XORed with the destination word and the result is stored in the destination word. The source and destination word can be addressed either directly or indirectly.

FLAGS:
- C: Unaffected.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if result bit 15 is set, otherwise cleared.
- V: Always reset to zero.
- D: Unaffected.
- H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
XORW (r8),RR64	96 40 68	1001 0110 0100 0000 0110 1000

If register pair 64 contains 11001100/11001100B, working register 8 contains 200 (decimal) and register pair 200 contains 10101010/10101010B, after this instruction register pair 200 will hold 01100110/01100110B.

XORW

Exclusive OR (Word) - Register, Memory

XORW dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Adress Mode		Oper
					dst	src	
[OPC] [dst,0|src,1]	2	16	6E	-	rr	(rr)	a
[OPC] [XTN |src,0] [dst,0]	3	18	7E	6	RR	(rr)	a
[OPC] [XTN |src,1] [dst,0]	3	22	D5	6	RR	(rr)+	b
	3	22	D5	6	rr*	(rr)+	b
	3	24	C3	6	RR	-(rr)	c
	3	24	C3	6	rr*	-(rr)	c
[OPC] [ofs,0|src,0] [XTN |dst,0]	3	24	60	6	rr	rr(rr)	a
[OPC] [XTN |src,1] [ofs]	4	28	86	6	RR	N(rr)	a
[dst,0]	4	28	86	6	rr*	N(rr)	a
[OPC] [XTN |dst,0] [src h]	4	22	E2	6	rr	NN	a
[src l]							
[OPC] [XTN |src,0] [ofs h]	5	30	86	6	RR	NN(rr)	a
[ofs l] [dst,0]	5	30	86	6	rr*	NN(rr)	a

OPERATION a: dst ⇐ dst XOR src

The source word is XORed with the destination word and the result is stored in the destination word. The destination word is held in the destination register. The source word can be addressed directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst XOR src
rr ⇐ rr + 2

The source word is XORed with the destination word and the result is stored in the destination word. The source word is in the memory location addressed by the source register pair, the destination word is in the destination register. The contents of the source register pair are incremented after the XOR has been carried out.

OPERATION c: rr ⇐ rr - 2
dst ⇐ dst XOR src

The source word is XORed with the destination word and the result is stored in the destination word. The source word is in the memory location addressed by the source register pair, the destination word is in the destination register. The contents of the source register pair are decremented before the XOR is carried out.

XORW

Exclusive OR (Word) - Register, Memory

XORW dst,src (Cont'd)

FLAGS:
C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 15 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
XORW RR64,-(rr4)	C3 65 40	1100 0011 0110 0101 0100 0000

If working register pair 4 contains 1184 (decimal), register pair 64 contains 10101010/10101010B and memory pair 1182 contains 11001100/11001100B, after this instruction register pair 64 will contain 01100110/01100110B and register pair 4 will contain 1182.

XORW

Exclusive OR (Word) - Memory, Register

XORW dst,src

INSTRUCTION FORMAT:	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Adress Mode		Oper
					dst	src	
[OPC] [dst,1|src,0]	2	28	6E	-	(rr)	rr	a
[OPC] [XTN |dst,1] [src,0]	3	30	BE	6	(rr)	RR	a
[OPC] [XTN |dst,0] [src,0]	3	32	D5	6	(rr)+	RR	b
	3	32	D5	6	(rr)+	rr*	b
	3	32	C3	6	-(rr)	RR	c
	3	32	C3	6	-(rr)	rr*	c
[OPC] [ofd,0|dst,1] [XTN |src,0]	3	34	60	6	rr(rr)	rr	a
[OPC] [XTN |dst,1] [ofd] [src,1]	4	36	86	6	N(rr)	RR	a
	4	36	86	6	N(rr)	rr*	a
[OPC] [XTN |src,1] [dst h] [dst l]	4	32	E2	6	NN	rr	a
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src,1]	5	38	86	6	NN(rr)	RR	a
	5	38	86	6	NN(rr)	rr*	a

OPERATION a: dst ⇐ dst XOR src

The source word is XORed with the destination word and the result is stored in the destination word. The source word is held in the source register. The destination word can be addressed directly, indirectly or by indexing.

OPERATION b: dst ⇐ dst XOR src
rr ⇐ rr + 2

The source word is XORed with the destination word and the result is stored in the destination word. The source word is in the source register, the destination word is in the memory location addressed by the destination register pair. The contents of the destination register pair are incremented after the XOR has been carried out.

OPERATION c: rr ⇐ rr - 2
dst ⇐ dst XOR src

The source word is XORed with the destination word and the result is stored in the destination word. The source word is in the source register , the destination word is in the memory location addressed by the destination register pair. The contents of the destination register pair are decremented before the XOR is carried out.

XORW

Exclusive OR (Word) - Memory, Register

XORW dst,src (Cont'd)

FLAGS:
C: Unaffected.
Z: Set if the result is zero, otherwise cleared.
S: Set if result bit 15 is set, otherwise cleared.
V: Always reset to zero.
D: Unaffected.
H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
XORW (rr4)+,RR64	D5 64 40	1101 0101 0110 0100 0100 0000

If register pair 64 contains 11001100/11001100B, working register pair 4 contains 1064 (decimal) and memory pair 1064 contains 10101010/10101010B, after this instruction is carried out memory pair 1064 will contain 01100110/01100110B and working register pair 4 will contain 1066.

XORW

Exclusive OR (Word) - Memory, Memory

XORW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Adress Mode	
					dst	src
[OPC] [dst,1|src,1]	2	32	6E	-	(rr)	(rr)

OPERATION: dst ⇐ dst XOR src

The source word is XORed with the destination word and the result is stored in the destination word. The source word is in the memory location addressed by the source register pair, the destination word is in the memory location addressed by the destination register pair.

FLAGS:
- C: Unaffected.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if result bit 15 is set, otherwise cleared.
- V: Always reset to zero.
- D: Unaffected.
- H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
XORW (rr4),(rr6)	6E 57	0110 1110 0101 0111

If working register pair 6 contains 1002 (decimal), memory pair 1002 contains 11001100/11001100B, working register pair 4 contains 1060 (decimal) and memory pair 1060 contains 10101010/10101010B, after this instruction memory pair 1060 will contains 01100110/01100110B.

XORW

Exclusive OR (Word) - All, Immediate

XORW dst,src

INSTRUCTION FORMAT:

	No. Bytes	No. Cycl	OPC (HEX)	OPC XTN	Adress Mode	
					dst	src
[OPC] [dst,1] [src h] [src l]	4	14	67	-	RR	#NN
	4	14	67	-	rr*	#NN
[OPC] [XTN |dst,0] [src h] [src l]	4	32	BE	6	(rr)	#NN
[OPC] [XTN |dst,1] [ofd] [src h] [src l]	5	36	06	6	N(rr)	#NN
[OPC] [XTN |dst,0] [ofd h] [ofd l] [src h] [src l]	6	38	06	6	NN(rr)	#NN
[OPC] [XTN] [src h] [src l] [dst h] [dst l]	6	38	36	61	NN	#NN

OPERATION: dst ⇐ dst XOR src

The source word is XORed with the destination word and the result is stored in the destination word. The source word is the immediate value in the operand, the destination word can be in memory or in the register file.

FLAGS:
- C: Unaffected.
- Z: Set if the result is zero, otherwise cleared.
- S: Set if result bit 15 is set, otherwise cleared.
- V: Always reset to zero.
- D: Unaffected.
- H: Unaffected.

EXAMPLE:

Instruction	HEX	Binary
XORW RR64,#52428	67 41 CC CC	0110 0111 0100 0001 1100 1100 1100 1100

If register pair 64 contains 10101010/10101010B, after this instruction has benn carried out register pair 64 will contains 01100110/01100110B.

Appendix A

ASCII Character Set

Hex	Dec	Char	Hex	Dec	Char	Hex	Dec	Char	Hex	Dec	Char
00	0	NUL	20	32	SP	40	64	@	60	96	'
01	1	SOH	21	33	!	41	65	A	61	97	a
02	2	STX	22	34	"	42	66	B	62	98	b
03	3	ETX	23	35	#	43	67	C	63	99	c
04	4	EOT	24	36	$	44	68	D	64	100	d
05	5	ENQ	25	37	%	45	69	E	65	101	e
06	6	ACK	26	38	&	46	70	F	66	102	f
07	7	BEL	27	39	'	47	71	G	67	103	g
08	8	BS	28	40	(	48	72	H	68	104	h
09	9	HT	29	41	)	49	73	I	69	105	i
0A	10	LF	2A	42	*	4A	74	J	6A	106	j
0B	11	VT	2B	43	+	4B	75	K	6B	107	k
0C	12	FF	2C	44	,	4C	76	L	6C	108	l
0D	13	CR	2D	45	-	4D	77	M	6D	109	m
0E	14	S0	2E	46	.	4E	78	N	6E	110	n
0F	15	SI	2F	47	/	4F	79	O	6F	111	o
10	16	DLE	30	48	0	50	80	P	70	112	p
11	17	DCI	31	49	1	51	81	Q	71	113	q
12	18	DC2	32	50	2	52	82	R	72	114	r
13	19	DC3	33	51	3	53	83	S	73	115	s
14	20	DC4	34	52	4	54	84	T	74	116	t
15	21	NAK	35	53	5	55	85	U	75	117	u
16	22	SYN	36	54	6	56	86	V	76	118	v
17	23	ETB	37	55	7	57	87	W	77	119	w
18	24	CAN	38	56	8	58	88	X	78	120	x
19	25	EM	39	57	9	59	89	Y	79	121	y
1A	26	SUB	3A	58	:	5A	90	Z	7A	122	z
1B	27	ESC	3B	59	;	5B	91	[	7B	123	{
1C	28	FS	3C	60	<	5C	92	\	7C	124	|
1D	29	GS	3D	61	=	5D	93	]	7D	125	}
1E	30	RS	3E	62	>	5E	94		7E	126	~
1F	31	US	3F	63	?	5F	95	_	7F	127	DEL

SALES OFFICES

EUROPE

DENMARK

2730 HERLEV
Herlev Torv, 4
Tel (45-42) 94 85 33
Telex 35411
Telefax (45-42) 948694

FINLAND

LOHJA SF-08150
Karjalankatu, 2
Tel 12 155 11
Telefax 12 155 66

FRANCE

94253 GENTILLY Cedex
7 - avenue Gallieni - BP 93
Tel (33-1) 47 40 75 75
Telex 632570 STMHQ
Telefax (33-1) 47 40 79 10

67000 STRASBOURG
20, Place des Halles
Tel (33) 88 75 50 66
Telex 870001F
Telefax (33) 88 22 29 32

GERMANY

6000 FRANKFURT
Gutleutstrasse 322
Tel (49-69) 237492
Telex 176997 689
Telefax (49-69) 231957
Teletex 6997689=STVBP

8011 GRASBRUNN
Bretonischer Ring 4
Neukeferloh Technopark
Tel (49-89) 46006-0
Telex 528211
Telefax (49-89) 4605454
Teletex 897107=STDISTR

3000 HANNOVER 1
Eckenerstrasse 5
Tel. (49-511) 634191
Telex 175118418
Teletex 5118418 csfbeh
Telefax (49-511) 633552

8500 NÜRNBERG 20
Erlenstegenstrasse, 72
Tel (49-911) 59893-0
Telex 626243
Telefax (49-911) 5980701

5200 SIEGBURG
Frankfurter Str 22a
Tel (49-2241) 660 84-86
Telex 889510
Telefax (49-2241) 67584

7000 STUTTGART
Oberer Kirchhaldenweg 135
Tel (49-711) 692041
Telex 721718
Telefax (49-711) 691408

ITALY

20090 ASSAGO (MI)
Vle Milanofiori - Strada 4 - Palazzo A/4/A
Tel (39-2) 89213 1 (10 linee)
Telex 330131 - 330141 SGSAGR
Telefax (39-2) 8250449

40033 CASALECCHIO DI RENO (BO)
Via R Fucini, 12
Tel (39-51) 591914
Telex 512442
Telefax (39-51) 591305

00161 ROMA
Via A Torlonia, 15
Tel (39-6) 8443341
Telex 620653 SGSATE I
Telefax (39-6) 8444474

NETHERLANDS

5652 AR EINDHOVEN
Meerenakkerweg 1
Tel (31-40) 550015
Telex 51186
Telefax (31-40) 528835

SPAIN

08021 BARCELONA
Calle Platon, 6 4th Floor, 5th Door
Tel (34-3) 4143300-4143361
Telefax (34-3) 2021461

28027 MADRID
Calle Albacete, 5
Tel (34-1) 4051615
Telex 27060 TCCEE
Telefax (34-1) 4031134

SWEDEN

S-16421 KISTA
Borgarfjordsgatan, 13 - Box 1094
Tel (46-8) 7939220
Telex 12078 THSWS
Telefax (46-8) 7504950

SWITZERLAND

1218 GRAND-SACONNEX (GENEVA)
Chemin Francois-Lehmann, 18/A
Tel (41-22) 7986462
Telex 415493 STM CH
Telefax (41-22) 7984869

UNITED KINGDOM and EIRE

MARLOW, BUCKS
Planar House, Parkway
Globe Park
Tel.. (44-628) 890800
Telex 847458
Telefax (44-628) 890391

AMERICAS

BRAZIL

05413 SÃO PAULO
R Henrique Schaumann 286-CJ33
Tel (55-11) 883-5455
Telex (391)11-37988 "UMBR BR"
Telefax 11-551-128-22367

CANADA

BRAMPTON, ONTARIO
341 Main St North
Tel (416) 455-0505
Telefax 416-455-2606

U.S.A.

NORTH & SOUTH AMERICAN
MARKETING HEADQUARTERS
1000 East Bell Road
Phoenix, AZ 85022
(1)-(602) 867-6100

SALES COVERAGE BY STATE

ALABAMA
Huntsville - (205) 533-5995

ARIZONA
Phoenix - (602) 867-6340

CALIFORNIA
Santa Ana - (714) 957-6018
San Jose - (408) 452-8585

COLORADO
Boulder (303) 449-9000

ILLINOIS
Schaumburg - (708) 517-1890

INDIANA
Kokomo - (317) 459-4700

MASSACHUSETTS
Lincoln - (617) 259-0300

MICHIGAN
Livonia - (313) 462-4030

NEW JERSEY
Voorhees - (609) 772-6222

NEW YORK
Poughkeepsie - (914) 454-8813

NORTH CAROLINA
Raleigh - (919) 787-6555

TEXAS
Carrollton - (214) 466-8844

FOR RF AND MICROWAVE POWER TRANSISTORS CONTACT THE FOLLOWING REGIONAL OFFICE IN THE U.S.A.

PENNSYLVANIA
Montgomeryville - (215) 362-8500

ASIA / PACIFIC

AUSTRALIA

NSW 2027 EDGECLIFF
Suite 211, Edgecliff centre
203-233, New South Head Road
Tel (61-2) 327 39 22
Telex 071 126911 TCAUS
Telefax (61-2) 327 61 76

HONG KONG

WANCHAI
22nd Floor - Hopewell centre
183 Queen's Road East
Tel (852-5) 8615788
Telex 60955 ESGIES HX
Telefax (852-5) 8656589

INDIA

NEW DELHI 110001
LiasonOffice
62, Upper Ground Floor
World Trade Centre
Barakhamba Lane
Tel 3715191
Telex 031-66816 STMI IN
Telefax 3715192

MALAYSIA

PULAU PINANG 10400
4th Floor - Suite 4-03
Bangunan FOP-123D Jalan Anson
Tel (04) 379735
Telefax (04) 379816

KOREA

SEOUL 121
8th floor Shinwon Building
823-14, Yuksam-Dong
Kang-Nam-Gu
Tel (82-2) 553-0399
Telex SGSKOR K29998
Telefax (82-2) 552-1051

SINGAPORE

SINGAPORE 2056
28 Ang Mo Kio - Industrial Park 2
Tel (65) 4821411
Telex RS 55201 ESGIES
Telefax (65) 4820240

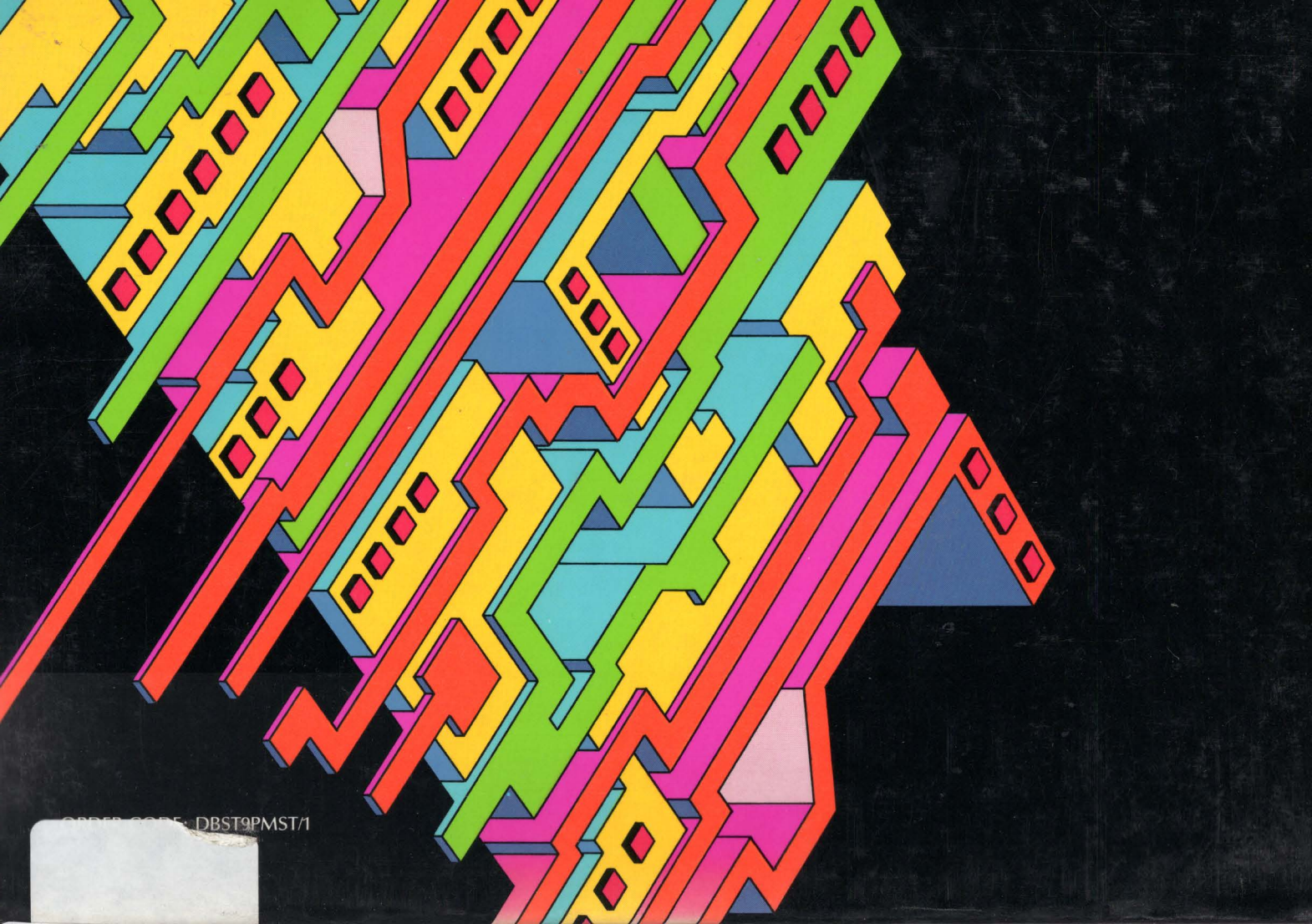
TAIWAN

TAIPEI
12th Floor
571, Tun Hua South Road
Tel (886-2) 755-4111
Telex 10310 ESGIE TW
Telefax (886-2) 755-4008

JAPAN

TOKYO 108
Nisseki - Takanawa Bld 4F
2-18-10 Takanawa
Minato-Ku
Tel (81-3) 3280-4121
Telefax (81-3) 3280-4131

Printed in Italy - by Grafiche Mambretti - Inverigo



ORDER CODE: DBST9PMST/1