Zilog - Z8038018FSC Datasheet

Welcome to E-XFL.COM

Understanding Embedded - Microprocessors

Embedded microprocessors are specialized computing chips designed to perform specific tasks within an embedded system. Unlike general-purpose microprocessors found in personal computers, embedded microprocessors are tailored for dedicated functions within larger systems, offering optimized performance, efficiency, and reliability. These microprocessors are integral to the operation of countless electronic devices, providing the computational power necessary for controlling processes, handling data, and managing communications.

Applications of Embedded - Microprocessors

Embedded microprocessors are utilized across a broad spectrum of applications, making them indispensable in

Details	
Product Status	Obsolete
Core Processor	Z380
Number of Cores/Bus Width	1 Core, 32-Bit
Speed	18MHz
Co-Processors/DSP	-
RAM Controllers	DRAM
Graphics Acceleration	No
Display & Interface Controllers	-
Ethernet	-
SATA	-
USB	-
Voltage - I/O	5.0V
Operating Temperature	0°C ~ 70°C (TA)
Security Features	-
Package / Case	100-QFP
Supplier Device Package	100-QFP
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8038018fsc

Revision History

Each instance in Revision History reflects a change to this document from its previous revision. For more details, refer to the corresponding pages and appropriate links in the table below.

Date	Revision Level	Description	Page No
July 2008	02	Updated format to the latest PS template	All
March 2001	01	Original Issue	All

GENERAL DESCRIPTION

The Z380 Microprocessor is an integrated high-performance microprocessor with fast and efficient throughput and increased memory addressing capabilities. The Z380 offers a continuing growth path for present Z80-or Z180-based designs, while maintaining Z80® CPU and Z180 MPU object-code compatibility. The Z380 MPU enhancements include an improved 280 CPU, expanded 4-Gbyte space and flexible bus interface timing.

An enhanced version of the Z80 CPU is key to the Z380 MPU. The basic addressing modes of the Z80 microprocessor have been augmented as follows: Stack Pointer Relative loads and stores, 16-bit and 24-bit indexed offsets, and more flexible Indirect Register addressing, with all of the addressing modes allowing access to the entire 32-bit address space. Additions made to the instruction set, include a full complement of 16-bit arithmetic and logical operations, 16-bit I/O operations, multiply and divide, plus a complete set of register-to-register loads and exchanges.

The expanded basic register file of the Z80 MPU microprocessor includes alternate register versions of the IX and IY registers. There are four sets of this basic Z80 microprocessor register file present in the Z380 MPU, along with the necessary resources to manage switching between the different register sets. All of the register-pairs and index registers in the basic Z80 microprocessor register file are expanded to 32 bits.

The Z380 MPU expands the basic 64 Kbyte Z80 and Z180 address space to a full 4 Gbyte (32-bit) address space. This address space is linear and completely accessible to the user program. The I/O address space is similarly expanded to a full 4 Gbyte (32-bit) range and 16-bit I/O, and both simple and block move are added.

Some features that have traditionally been handled by external peripheral devices have been incorporated in the design of the Z380 microprocessor. The on-chip peripherals reduce system chip count and reduce interconnection on the external bus. The Z380 MPU contains a refresh controller for DRAMs that employs a /CAS-before-/RAS refresh cycle at a programmable rate and burst size.

Six programmable memory-chip selects are available, along with programmable wait-state generators for each chip-select address range.

The Z380 MPU provides flexible bus interface timing, with separate control signals and timing for memory and I/O. The memory bus control signals provide timing references suitable for direct interface to DRAM, static RAM, EPROM, or ROM. Full control of the memory bus timing is possible because the /WAIT signal is sampled three times during a memory transaction, allowing complete user control of edge-to-edge timing between the reference signals provided by the Z380 MPU. The I/O bus control signals allow direct interface to members of the Z80 family of peripherals, the Z8000 family of peripherals, or the Z8500 series of peripherals. Figure 1 shows the Z380 block diagram; Figure 2 shows the pin assignments.

/EV Evaluation Mode (input, active Low). This input should be left unconnected for normal operation. When it is driven to logic 0, the Z380 MPU conditions itself in the reset mode and tri-states all of its output pin drivers.

/HALT Halt Status (output, active Low, tri-state). If the Z380 MPU standby mode option is not selected, a Sleep instruction is executed no different than a Halt instruction, and the one HALT signal goes active to indicate the CPU's HALT state. If the standby mode option is selected, this signal goes active only at the Halt instruction execution.

/STNBY Standby Status (output, active Low, tri-state). If the Z380 MPU standby mode is selected, executing a sleep instruction stops clocking within the Z380 MPU and at BUSCLK and IOCLK after which this signal is asserted. The Z380 MPU is then in the low power standby mode, with all operations suspended.

/INT3-0 Interrupt Requests (inputs, active Low). These signals are four asynchronous maskable interrupt inputs.

IOCLK I/O Clock (output, active High, tri-state). This signal is a program controlled divided-down version of BUSCLK. The division factor can be two, four, six or eight with I/O transactions and interrupt-acknowledge transactions occurring relative to IOCLK.

/INTAK Interrupt Acknowledge Status (output, active Low, tri-state). This signal is used to distinguish between I/O and interrupt acknowledge transactions. This signal is High during I/O read and I/O write transactions and Low during interrupt acknowledge transactions.

/IORQ Input/Output Request (output, active Low, tri-state). This signal is active during all I/O read and write transactions and interrupt acknowledge transactions.

/M1 Machine Cycle One (output, active Low, tri-state). This signal is active during interrupt acknowledge and RETI transactions.

/IORD Input, Output Read Strobe (output, active Low, tri-state). This signal is used strobe data from the peripherals during I/O read transactions. In addition, /IORD is active during the special RETI transaction and the I/O heartbeat cycle in the Z80 protocol case.

/IOWR Input/Output Write Strobe (output, active Low, tri-state). This signal is used to strobe data into the peripherals during I/O write transactions.

/LMCS Low Memory Chip Select (output, active Low, tri-state). This signal is activated during a memory read or memory write transaction when accessing the lower portion of the linear address space within the first 16 Mbytes, but only if this chip select function is enabled.

/MCS3-/MCS0 Mid-range Memory Chip Selects (output, active Low, tri-state). These signals are individually active during memory read or write transactions when accessing the mid-range portions of the linear address space within the first 16 Mbytes. These signals can be individually enabled or disabled.

/MRD Memory Read (output, active Low, tri-state). This signal indicates that the addressed memory location should place its data on the data bus as specified by the /

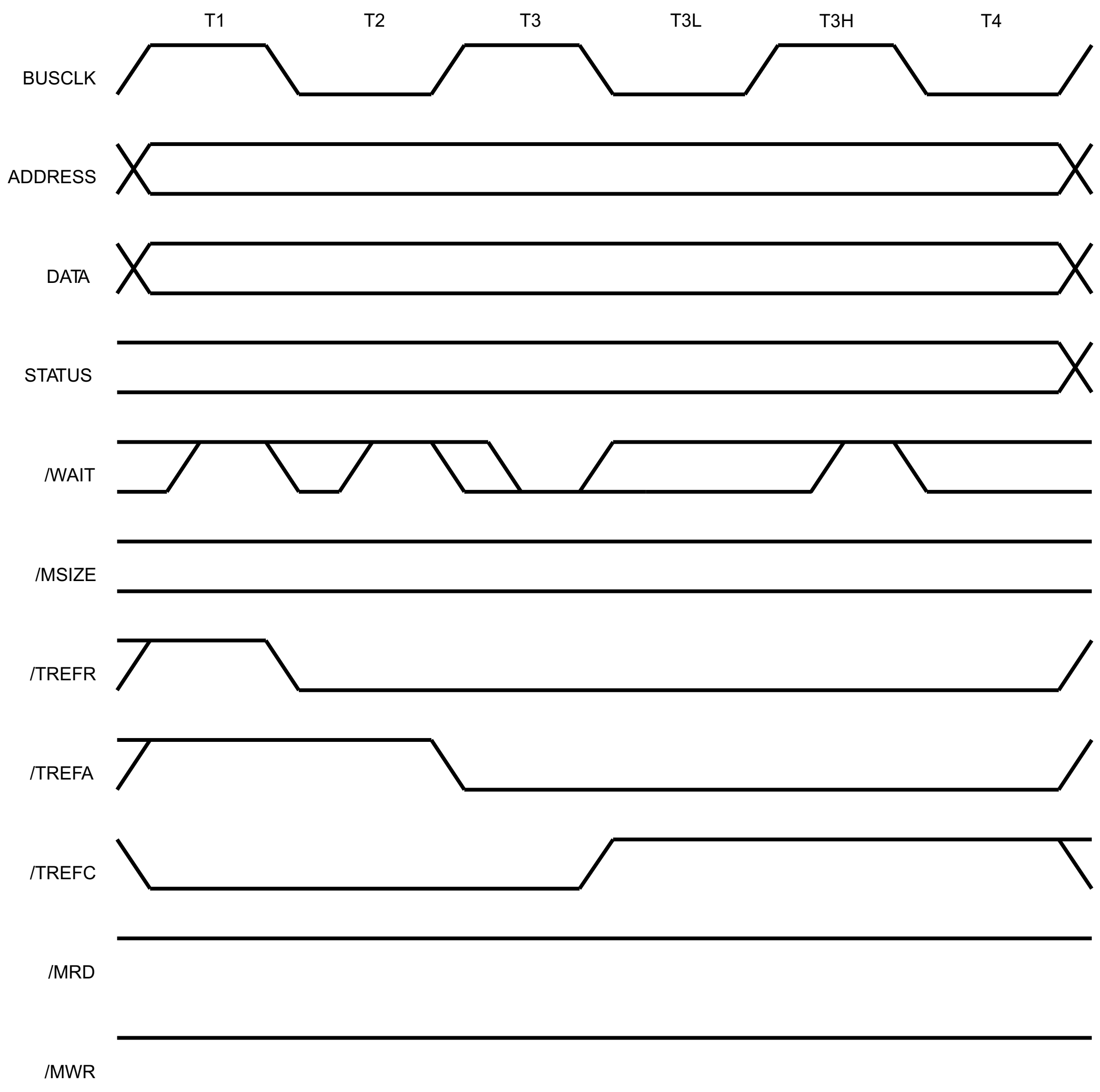

Figure 15. Refresh Cycle, T3 Wait

I/O Transactions

I/O transactions move data to or from an external peripheral when the Z380 MPU performs an I/O access. All I/O transactions occur referenced to the IOCLK signal, when it is a divided-down version of the BUSCLKsignal. BUSCLK may be divided by a factor of from two to eight to form the IOCLK, under program control. An example of this division is shown, for the four possible divisors, in Figure 16. Note that the IOCLK divider is synchronized (i.e., starts with a known timing relationship) at the trailing edge of /RESET. This is discussed in the Reset Section.

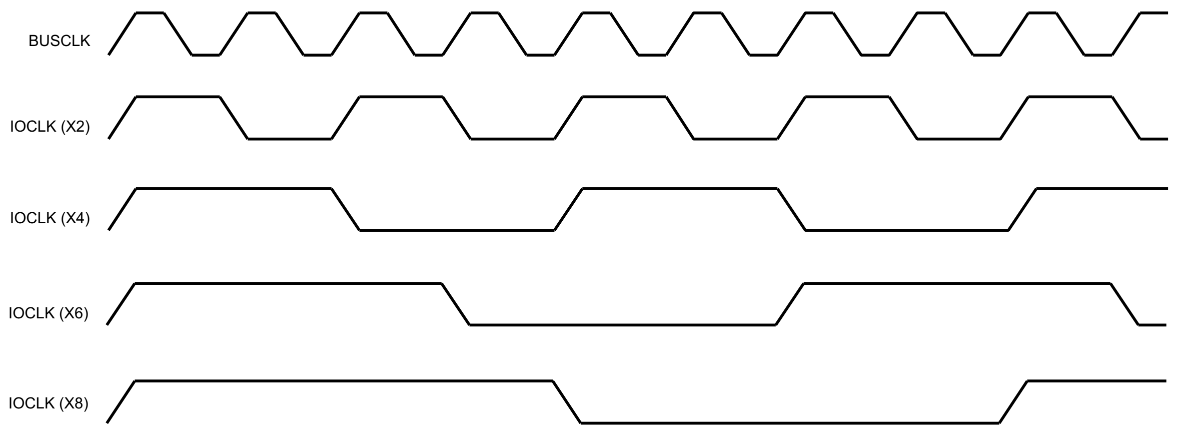

Figure 16. IOCLK Timing

EXTERNAL INTERFACE (Continued)

The Z380 MPU is unique in that it employs separate control signals for accessing the memory and I/O. This allows the two interfaces to be optimized independent of one another. The I/O bus control signals allow direct connection to members of the Z80 family of peripherals or the Z8500 family of peripherals.

Note that because all I/O bus transactions start on a rising edge of IOCLK, there may be up to n BUSCLK cycles of latency between the execution unit request for the transaction and the transaction actually starting, where n is the programmed clock divisor for IOCLK. This implies that the lowest possible divisor should always be used for IOCLK.

All I/O transactions are four IOCLK cycles long unless extended by Wait states. Wait states may be inserted between the third and fourth IOCLK cycles in an I/O transaction and are one IOCLK cycle per wait state. The external /WAIT input is sampled only after internally-generated wait states are inserted.

Requests

A request can be initiated by a device that does not have control of the bus. Two types of request can occur: Bus request and Interrupt request. When an interrupt or bus request is made, it is answered by the CPU according to its type. For an interrupt request, the CPU initiates an interrupt acknowledge transaction and for bus requests, the CPU enters the bus disconnect state, relinquishes the bus, and activates an Acknowledge signal.

BUS Requests

To generate transactions on the bus, a potential bus master (such as a DMA controller) must gain control of the bus by making a bus request. A bus request is initiated by driving /BREQ Low. Several bus requesters may be wired-OR to the /BREQ pin; priorities are resolved externally to the CPU, usually by a priority daisy chain.

The asynchronous /BREQ signal generates an internal /BUSREQ, which is synchronous. If the /BREQ is active at the beginning of any transaction, the internal /BUSREQ causes the /BACK signal to be asserted after the current transaction is completed. The Z380 MPU then enters the Bus Disconnect state and gives up control of the bus. All Z380 MPU control signals, except /BACK, /MI and /INTAK are tri-stated. Note that release of the bus may be inhibited under program control to allow the Z380 MPU exclusive access to a shared resource; this is controlled by the SETC LCK and RESC LCK instructions. Entry into the Bus Disconnect state is shown in Figure 26. The Z380 MPU regains control of the bus after /BREQ is deasserted. This is shown in Figure 27.

Interrupt Requests

The Z380 MPU supports two types of interrupt requests, maskable /INT3-INT0 and non-maskable (/NMI). The interrupt request line of a device that is capable of generating an interrupt can be tied to either /NMI or one of the maskable interrupt request lines, and several devices can be connected to one interrupt request line with the devices arranged in a priority daisy chain. However, because of the need for Z80 family peripheral devices to see the RETI instruction, only one daisy chain of Z80-family peripherals can be used. The Z380 MPU handles maskable and nonmaskable interrupt requests somewhat differently, as follows:

Any High-to-Low transition on the /NMI input is asynchronously edge-detected, and the internal NMI latch is set. At the beginning of the last clock cycle in the last internal machine cycle of any instruction, the maskable interrupts are sampled along with the state of the NMI latch.

If an enabled maskable interrupt is requested, at the next possible time (the next rising edge of IOCLK) an interrupt acknowledge transaction is generated to fetch the interrupt-vector from the interrupting device.For a nonmaskable interrupt, no interrupt acknowledge transaction is generated; the NMI service routine always starts at address 00000066H.

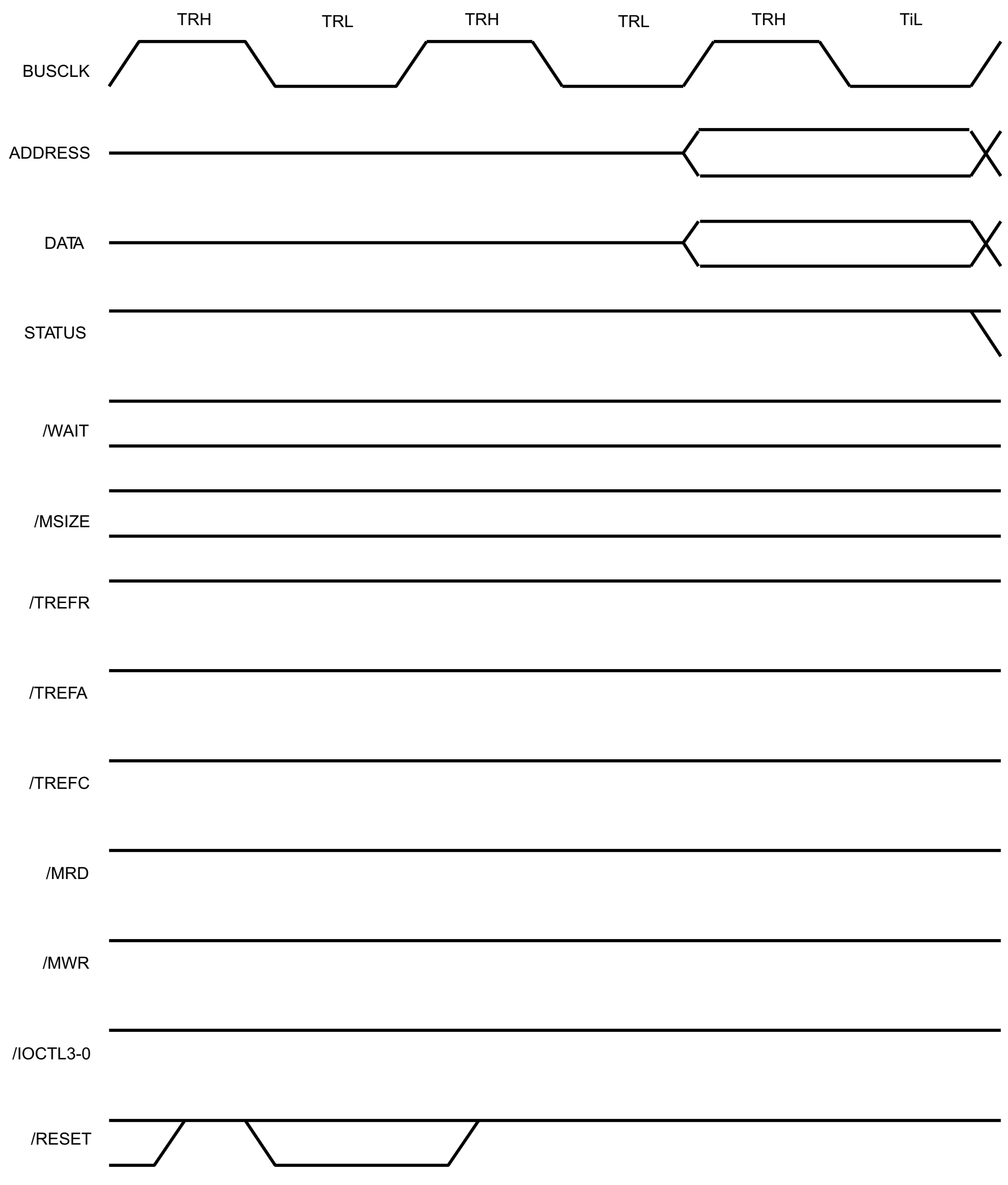

Figure 31. Reset Exit

Each register set includes the primary registers A, F, B, C, D, E, H, L, IX, and IY, as well as the alternate registers A’, F’, B’, C’, D’, E’, H’, L’, IX’, and IY’. These byte registers can be paired B with C, D with E, H with L, B’ with C’, D’ with E’ and H’ with L’ to form word registers. These word registers are extended to 32 bits with the z extension to the register. This register extension is only accessible when using the register as a 32-bit register (the Long Word mode) or when swapping between the most-significant and least-significant word of a 32-bit register. Whenever an instruction refers to a word register, the implicit size is controlled by the Word or Long Word mode. Also included are the R, I and SP registers, as well as the PC.

CPU Control Register Space

The CPU control register space consists of the 32-bit Select Register (SR), Figure 34. The SR may be accessed as a whole or the upper three bytes of the SR may be accessed individually as the YSR, XSR, and DSR. In addition, these upper three bytes can be loaded with the same byte value. The SR may also be PUSHed and POPed and is cleared to all zeros on Reset.

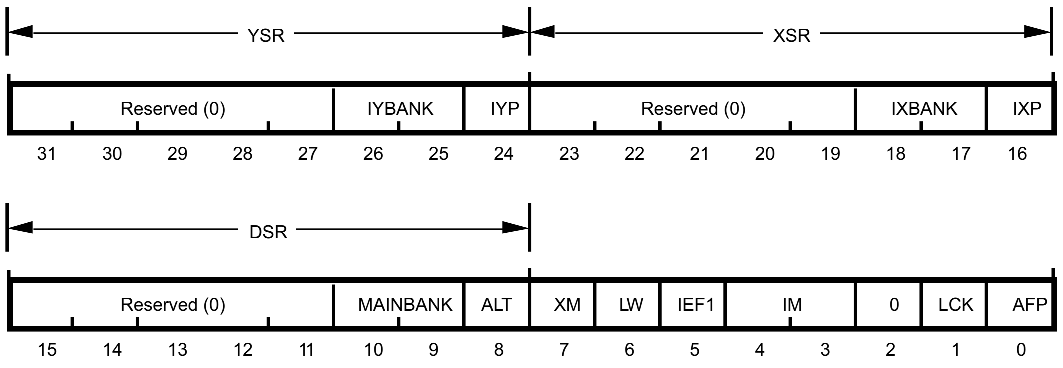

Figure 34. Select Register

IYBANK (IY Bank Select). This 2-bit field selects the register set to be used for the IY and IY' registers. This field can be set independently of the register set selection for the other Z380 CPU registers. Reset selects Bank 0 for IY and IY'.

IYP (IYPrime Register Select). This bit controls and reports whether IY or IY' is the currently active register. IY is selected when this bit is cleared and IY' is selected when this bit is set. Reset clears this bit and selects IY.

IXBANK (IX Bank Select). This 2-bit field selects the register set to be used for the IX and IX' registers. This field can be set independently of the register set selection for the other Z380 CPU registers. Reset selects Bank 0 for IX and IX'.

INSTRUCTION SET

The Z380 CPU's instruction set is a superset of the Z80 CPU's; the Z380 CPU is opcode compatible with the Z80 CPU. Thus a Z80 program can be executed on a Z380 MPU without modification. The instruction set is divided into seventeen groups by function:

The instructions are divided into the following categories.

- 8-bit load group
- 16/32 bit load group
- Push/Pop group
- Exchanges, block transfers, and searches
- 8-bit arithmetic and logic operations
- General purpose arithmetic and CPU control
- Decoder Directive Instructions
- 16/32 bit arithmetic operations
- Multiply/Divide Instruction group
- 8-bit Rotates and shifts
- 16-bit Rotates and shifts
- 8-bit bit set, reset, and test operations
- Jumps
- Calls, returns, and restarts
- 8-bit input and output operations for External I/O address space
- 8-bit input and output operations for Internal I/O address space
- 16-bit input and output operations

Instruction Set

The following is a summary of the Z380 instruction set which shows the assembly language mnemonic, the operation, the flag status, and gives comments on each instructions.

Note: Mnemonic and object code assignment for newly added instructions (instructions in ***Italic*** face) are preliminary and subject to change without notice.

The Z380 Technical Manual will contain significantly more details for programming use. A list of instructions, as well as encoding is included in Appendix A of this document.

16/32 BIT LOAD GROUP

	Symbolic	Flags					P/				Opcode			# of	Execute	
Mnemonic	**Operation**	**S**	**Z**	**x**	**H**	**x**	**V**	**N**	**C**	**76**	**543**	**210**	**HEX**	**Bytes**	**Time**	**Notes**
LD dd,nn	dd ← nn	•	•	x	•	x	•	•	•	00	dd0	001		3	2	L1,I
											⟵ n ⟶					
											⟵ n ⟶					
LD XY,nn	XY ← nn	•	•	x	•	x	•	•	•	11	y11	101		4	2	L1,I
										00	100	001	21			
											⟵ n ⟶					
											⟵ n ⟶					
LD HL,(nn)	H ← (nn+1)	•	•	x	•	x	•	•	•	00	101	010	2A	3	3+r	L1,I
	L ← (nn)										⟵ n ⟶					
											⟵ n ⟶					
LD dd,(nn)	ddh ← (nn+1)	•	•	x	•	x	•	•	•	11	101	101	ED	4	3+r	L1,I
	ddl ← (nn)									01	dd1	011				
											⟵ n ⟶					
											⟵ n ⟶					
LD XY,(nn)	XYU ← (nn+1)	•	•	x	•	x	•	•	•	11	y11	101		4	3+r	L1,I
	XYL ← (nn)									00	101	010	2A			
											⟵ n ⟶					
											⟵ n ⟶					
LD (nn),HL	(nn+1) ← H	•	•	x	•	x	•	•	•	00	100	010	22	3	4+w	L1,I
	(nn) ← L										⟵ n ⟶					
											⟵ n ⟶					
LD (nn),dd	(nn+1) ← ddh	•	•	x	•	x	•	•	•	11	101	101	ED	4	4+w	L1,I
	(nn) ← ddl									01	dd0	011				
											⟵ n ⟶					
											⟵ n ⟶					
LD (nn),XY	(nn+1) ← XYU	•	•	x	•	x	•	•	•	11	y11	101		4	4+w	L1,I
	(nn) ← XYL									00	100	010	22			
											⟵ n ⟶					
											⟵ n ⟶					
LD W(pp),nn	(pp+1) ← nh	•	•	x	•	x	•	•	•	11	101	101	ED	4	3+w	L1,I
	(pp) ← nl									00	pp0	110				
											⟵ n ⟶					
											⟵ n ⟶					
LD pp,(uu)	pph ← (uu+1)	•	•	x	•	x	•	•	•	11	011	101	DD	2	2+r	L1
	ppl ← (uu)									00	pp1	1uu				
LD (pp),uu	(pp+1) ← uuh	•	•	x	•	x	•	•	•	11	111	101	FD	2	3+w	L1
	(pp) ← uul									00	pp1	1uu				
LD SP,HL	SP ← HL	•	•	x	•	x	•	•	•	11	111	001	F9	1	2	L1
LD SP,XY	SP ← XY	•	•	x	•	x	•	•	•	11	y11	101		2	2	L1
										11	111	001	F9			
LD pp,UU	pp ← UU	•	•	x	•	x	•	•	•	11	UU1	101		2	2	L1
										00	pp0	010				
LD XY,pp	XY ← pp	•	•	x	•	x	•	•	•	11	y11	101		2	2	L1
										00	pp0	111				
LD IX,IY	IX ← IY	•	•	x	•	x	•	•	•	11	011	101	DD	2	2	L1
										00	100	111	27			

Mnemonic	Symbolic Operation	S	Z	x	H	x	P/V	N	C	76	543	210	HEX	# of Bytes	Execute Time	Notes
LD IY,IX	IY ← IX	•	•	x	•	x	•	•	•	11	111	101	FD	2	2	L1
										00	100	111	27			
LD pp,XY	pp ← XY	•	•	x	•	x	•	•	•	11	y11	101		2	2	L1
										00	pp1	011				
LD (pp),XY	(pp+1) ← XYU	•	•	x	•	x	•	•	•	11	y11	101		2	3+w	L1
	(pp) ← XYL									00	pp0	001				
LD XY,(pp)	XYU ← (pp+1)	•	•	x	•	x	•	•	•	11	y11	101		2	2+r	L1
	XYL ← (pp)									00	pp0	011				
LD pp,(XY+d)	pph ← (XY+d)h	•	•	x	•	x	•	•	•	11	y11	101		4	4+r	L1,I
	ppl ← (XY+d)l									11	001	011	CB			
										←	d	→				
										00	pp0	011				
LD IX,(IY+d)	IXU ← (IY+d)h	•	•	x	•	x	•	•	•	11	111	101	FD	4	4+r	L1,I
	IXL ← (IY+d)l									11	001	011	CB			
										←	d	→				
										00	100	011	23			
LD IY,(IX+d)	IYU ← (IX+d)h	•	•	x	•	x	•	•	•	11	011	101	DD	4	4+r	L1,I
	IYL ← (IX+d)l									11	001	011	CB			
										←	d	→				
										00	100	011	23			
LD pp,(SP+d)	pph ← (SP+d)h	•	•	x	•	x	•	•	•	11	011	101	DD	4	4+r	L1,I
	ppl ← (SP+d)l									11	001	011	CB			
										←	d	→				
										00	pp0	001				
LD XY,(SP+d)	XYU ← (SP+d)h	•	•	x	•	x	•	•	•	11	y11	101		4	4+r	L1, I
	XYL ← (SP+d)l									11	001	011	CB			
										←	d	→				
										00	100	001	21			
LD (XY+d),pp	(XY+d)h ← pph	•	•	x	•	x	•	•	•	11	y11	101		4	5+w	L1, I
	(XY+d)l ← ppl									11	001	011	CB			
										←	d	→				
										00	pp1	011				
LD (IX+d),IY	(IX+d)h ← IYU	•	•	x	•	x	•	•	•	11	011	101	DD	4	5+w	L1, I
	(IX+d)l ← IYL									11	001	011	CB			
										←	d	→				
										00	101	011	2B			
LD (IY+d),IX	(IY+d)h ← IXU	•	•	x	•	x	•	•	•	11	111	101	FD	4	5+w	L1, I
	(IY+d)l ← IXL									11	001	011	CB			
										←	d	→				
										00	101	011	2B			

Mnemonic	Symbolic Operation	S	Z	x	H	x	P/V	N	C	76	543	210	HEX	# of Bytes	Execute Time	Notes
LDDW	(DE) ← (HL)	•	•	x	0	x	V	0	•	11	101	101	ED	1	3+r+w	N,L8(4)
	(DE+1) ← (HL+1)						(1)			11	101	000	E8			
	DE ← DE-2															
	HL ← HL-2															
	BC(15-0) ← BC(15-0)-2															
LDDRW	(DE) ← (HL)	•	•	x	0	x	0	0	•	11	101	101	ED	1	(3+r+w)n	N,L8(4)
	(DE+1) ← (HL+1)						(2)			11	111	000	F8			
	DE ← DE-2															
	HL ← HL-2															
	BC(15-0) ← BC(15-0)-2															
	Repeat until BC = 0															

r	Reg
000	B
001	C
010	D
011	E
100	H
101	L
111	A

pp	Regs
00	BC
00	DE
11	HL

y	XY
0	IX
1	IY

Notes:

Instructions in ***Italic*** face are Z380 new instructions, instructions with **underline** are Z180 original instructions.

L7: In Long Word mode, this instruction exchanges in 32-bits;
src(31-0) ↔ dst(31-0)

L8: In Long Word mode, this instruction transfers in 2 words and BC modified by 4 instead of 2

N: In Native mode, this instruction uses addresses modulo 65536.

(1): P/V flag is 0 if the result of BC-1 = 0, otherwise P/V = 1.

(2): P/V flag is 0 only at completion of instruction.

(3): Z Flag is 1 if A = (HL), otherwise Z = 0

(4): Source, Destination address, count value must be even numbers.

Mnemonic	Symbolic Operation	S	Z	x	H	x	P/V	N	C	Opcode 76	543	210	HEX	# of Bytes	Execute Time	Notes
TST r	A AND r	↕	↕	x	1	x	P	0	0	11	101	101	ED	2	2	
										00	r	100				
TST n	A AND n	↕	↕	x	1	x	P	0	0	11	101	101	ED	3	2	
										01	100	100	64			
										⟵	n	⟶				
TST (HL)	A AND (HL)	↕	↕	x	1	x	P	0	0	11	101	101	ED	2	2+r	
										00	110	100	34			

r	Reg
000	B
001	C
010	D
011	E
100	H
101	L
111	A

y	XY
0	IX
1	IY

Notes:
Instructions in ***Italic*** face are Z380 new instructions, instructions with **underline** are Z180 original instructions.
I: This instruction may be used with DDIR Immediate instructions.
(5): Two cycles to execute for Accumulator, three cycles to execute for any other registers.

Mnemonic	Symbolic Operation	Flags S Z x H x P/V N C	Opcode 76 543 210	HEX	# of Bytes	Execute Time	Notes
CALL nn	(SP-1) ← PCh (SP-2) ← PCl SP ← SP-2 PC ← nn	• • x • x • • •	11 001 101 ⟵ n ⟶ ⟵ n ⟶	CD	3	4+w	X3, I
CALL cc,nn	If condition cc is false continue otherwise same as CALL nn	• • x • x • • •	11 cc 100 ⟵ n ⟶ ⟵ n ⟶		3	2/4+w	X3, I
CALR e	(SP-1) ← PCh (SP-2) ← PCl SP ← SP-2 PC ← PC + e	• • x • x • • •	11 101 101 11 001 101 ⟵ e-3 ⟶	ED CD	3	4+w	N,X3,(11)
CALR cc,e	If condition cc is false continue otherwise same as CALR e	• • x • x • • •	11 101 101 11 cc 100 ⟵ e-3 ⟶	ED	3	2/4+w	N,X3,(11)
CALR ee	(SP-1) ← PCh (SP-2) ← PCl SP ← SP-2 PC ← PC + ee	• • x • x • • •	11 011 101 11 001 101 ← (ee-4)L → ← (ee-4)H →	DD CD	4	4+w	N,X3,(8)
CALR cc,ee	If condition cc is false continue otherwise same as CALR ee	• • x • x • • •	11 011 101 11 cc 100 ← (ee-4)L → ← (ee-4)H →	DD	4	2/4+w	N,X3,(8)
CALR eee	(SP-1) ← PCh (SP-2) ← PCl SP ← SP-2 PC ← PC + eee	• • x • x • • •	11 111 101 11 001 101 ← (eee-5)L → ← (eee-5)M → ← (eee-5)H →	FD CD	5	4+w	N,X3,(9)
CALR cc,eee	If condition cc is false continue otherwise same as CALR eee	• • x • x • • •	11 111 101 11 cc 100 ← (eee-5)L → ← (eee-5)M → ← (eee-5)H →	FD	5	2/4+w	N,X3,(9)
RET	PCL ← (SP) PCH ← (SP + 1) SP ← SP+2	• • x • x • • •	11 001 001	C9	1	2+r	N, X4
RET cc	If condition cc is false continue otherwise same as RET	• • x • x • • •	11 cc 000		1	2/2+r	N, X4
RETI	Return from Interrupt	• • x • x • • •	11 101 101 01 001 101	ED 4D	2	2+r	N, X4

	Symbolic	Flags					P/			Opcode				# of	Execute	
Mnemonic	**Operation**	**S**	**Z**	**x**	**H**	**x**	**V**	**N**	**C**	**76**	**543**	**210**	**HEX**	**Bytes**	**Time**	**Notes**
IN A,(n)	A ← (n)	•	•	x	•	x	•	•	•	11	011	011	DB	2	3+i	
										⟵	n	⟶				
IN r,(C)	r ← (C)	↕	↕	x	0	x	P	0	•	11	101	101	ED	2		
										01	r	000				
INA A,(nn)	A ← (nn)	•	•	x	•	x	•	•	•	11	101	101	ED	2	3+i	I
										11	011	011	DB			
										⟵	n	⟶				
										⟵	n	⟶				
INI	(HL) ← (C)	•	↕	x	•	x	•	1	•	11	101	101	ED	2	2+i+w	
	B ← B - 1		(1)							10	100	010	A2			
	HL ← HL + 1															
INIR	(HL) ← (C)	•	1	x	•	x	•	1	•	11	101	101	ED	2	(2+i+w)	
	B ← B-1		(2)							10	110	010	B2			
	HL ← HL + 1															
	Repeat until B = 0															
IND	(HL) ← (C)	•	↕	x	•	x	•	1	•	11	101	101	ED	2	2+i+w	
	B ← B - 1		(1)							10	101	010	AA			
	HL ← HL - 1															
INDR	(HL) ← (C)	•	1	x	•	x	•	1	•	11	101	101	ED	2	(2+i+w)n	
	B ← B-1		(2)							10	111	010	BA			
	HL ← HL - 1															
	Repeat until B = 0															
OUT (n),A	(n) ← A	•	•	x	•	x	•	•	•	11	010	011	D3	2	3+o	
										⟵	n	⟶				
OUT (C),r	(C) ← r	•	•	x	•	x	•	•	•	11	101	101	ED	2	3+o	
										01	r	001				
OUT (C),n	(C) ← r	•	•	x	•	x	•	•	•	11	101	101	ED	3	3+o	
										01	110	001	71			
										⟵	n	⟶				
OUTA (nn),A	(nn) ← A	•	•	x	•	x	•	•	•	11	101	101	ED	4	2+o	I
										11	010	011	D3			
										⟵	n	⟶				
										⟵	n	⟶				

Mnemonic	Symbolic Operation	S	Z	x	H	x	P/V	N	C	76	543	210	HEX	# of Bytes	Execute Time	Notes
OUTDW	BC(15-0) ← BC(15-0) - 1 (DE) ← (HL) HL ← HL - 2	•	↕ (1)	x	•	x	•	1	•	11 11	101 101	101 011	ED EB	2	2+r+o	
OTDRW	BC(15-0) ← BC(15-0) - 1 (DE) ← (HL) HL ← HL - 2 Repeat until B = 0	•	1 (2)	x	•	x	•	1	•	11 11	101 111	101 011	ED FB	2	2+r+o	

ppp	Reg
000	BC
010	DE
111	HL

Notes:
Instructions in ***Italic*** face are Z380 new instructions, instructions with **underline** are Z180 original instructions.
I: This instruction may be used with DDIR Immediate instructions.
N: In Native mode, this instruction uses addresses modulo 65536.
(1) If the result of B-1 is zero, the Z flag is set; otherwise it is reset.
(2) Z flag is set upon instruction completion only.

	Address Bus			
I/O Instruction	**A31-A24**	**A23-A16**	**A15-A8**	**A7-A0**
IN A, (n)	00000000	00000000	Contents of A reg	n
IN dst,(C)	BC31-BC24	BC23-BC16	BC15-BC8	BC7-BC0
INA(W) dst,(mn)	00000000	00000000	m	n
DDIR IB INA(W) dst,(lmn)	00000000	l	m	n
DDIR IW INA(W) dst,(klmn)	k	l	m	n
Block Input	BBC31-BC24	BC23-BC16	BC15-BC8	BC7-BC0
OUT (n),A	00000000	00000000	Contents of A reg	n
OUT (C),dst	BC31-BC24	BC23-BC16	BC15-BC8	BC7-BC0
OUTA(W) (mn),dst	00000000	00000000	m	n
DDIR IB OUTA(W) (lmn),dst	00000000	l	m	n
DDIR IW OUTA(W) (klmn),dst	k	l	m	n
Block output	BC31-BC24	BC23-BC16	BC15-BC8	BC7-BC0

Interrupt Control

The Z380 MPU's flags and registers associated with interrupt processing are listed in Table 4. As discussed in the CPU Architecture section, some of the registers reside in the on-chip I/O address space and can be accessed only with reserved on-chip I/O instructions.

Table 3. Interrupt Flags and Registers

Names	Mnemonics	Access Methods
Interrupt Enable Flags	IEF1, IEF2	EI and DI instructions
Interrupt Register	I	LD I,A and LD A,I instructions
Interrupt Register Extension	Iz	LD I,HL and LD HL,I instructions (accessing both Iz and I)
Interrupt Enable Register	IER	On-chip I/O instructions, addr 00000017H, EI and DI instructions
Assigned Vectors Base Register	AVBR	On-chip I/O instructions, addr 00000018H
Trap and Break Register	TRPBK	On-chip I/O instructions, addr 00000019H

IEF1, IEF2

IEF1 controls the overall enabling and disabling of all on-chip peripheral and external maskable interrupt requests. If IEF1 is at logic 0, all such interrupts are disabled. The purpose of IEF2 is to correctly manage the occurrence of /NMI. When /NMI is acknowledged, the state of IEF1 is copied to IEF2 and then IEF1 is cleared to logic 0. At the end of the /NMI interrupt service routine, execution of the Return From Nonmaskable Interrupt instruction, RETN, automatically copies the state of IEF2 back to IEF1. This is a means to restore the interrupt enable condition existing before the occurrence of /NMI. Table 5 summarizes the states of IEF1 and IEF2 resulting from various operations.

Table 4. Operation Effects on IEF1 and IEF2

Operation	IEF1	IEF2	Comments
/RESET	0	0	Inhibits all interrupts except Trap and /NMI.
Trap	0	0	Disables interrupt nesting.
/NMI	0	IEF1	IEF1 value copied to IEF2, then IEF1 is cleared.
RETN	IEF2	NC	Returns from /NMI service routine.
/INT3-/INT0	0	0	Disables interrupt nesting.
RETI	NC	NC	Returns from service routine, Z80 I/O device.
RET	NC	NC	Returns from service routine, non-Z80 I/O device.
EI	1	1	
DI	0	0	
LD A,I or LD R,I	NC	NC	IEF2 value is copied to P/V Flag.
LD HL,I	NC	NC	

Note:
NC = No Change

I, I Extend

The 8-bit Interrupt Register and the 16-bit Interrupt Register Extension are cleared during reset.

Nonmaskable Interrupt

The nonmaskable interrupt input /NMI is edge sensitive, with the Z380 MPU internally latching the occurrence of its falling edge. When the latched version of /NMI is recognized, the following operations are performed.

1. The interrupted PC (Program Counter) value is pushed onto the stack.

2. The state of IEF1 is copied to IEF2, then IEF1 is cleared.

3. The Z380 MPU commences to fetch and execute instructions from address 00000066H.

Interrupt Mode 0 Response For Maskable Interrupt /INT0

During the interrupt acknowledge transaction, the external I/O device being acknowledged is expected to output a vector onto the lower portion of the data bus, D7-D0. The Z380 MPU interprets the vector as an instruction opcode, which is usually one of the single-byte Restart (RST) instructions that pushes the interrupted PC (Program Counter) value onto the stack and resumes execution at a fixed memory location. However, the Z380 MPU will generate multiple transactions to capture vectors that form a multi-byte instruction. IEF1 and IEF2 are reset to logic 0's, disabling all further maskable interrupt requests. Note that unlike the other interrupt responses, the PC is not automatically PUSHed onto the stack. Note also that a trap occurs if an undefined opcode is supplied by the I/O device as a vector.

Interrupt Mode 1 Response For Maskable Interrupt /INT0

An interrupt acknowledge transaction is generated, during which the data bus contents are ignored by the Z380 MPU. The interrupted PC value is PUSHed onto the stack. IEF1 and IEF2 are reset to logic 0's so as to disable further maskable interrupt requests. Instruction fetching and execution restarts at memory location 00000038H.

Interrupt Mode 2 Response For Maskable interrupt /INT0

During the interrupt acknowledge transaction, the external I/O device being acknowledged is expected to output a vector onto the lower portion of the data bus, D7-D0. The interrupted PC value is PUSHed onto the stack and IEF1 and IEF2 are reset to logic 0's so as to disable further maskable interrupt requests. The Z380 MPU then reads an entry from a table residing in memory and loads it into the PC to resume execution. The address of the table entry is composed of the I Extend contents as A31-A16, the I Register contents as A15-A8 and the vector supplied by the I/O device as A7-A0. Note that the table entry is effectively the starting address of the interrupt service routine designed for the I/O device being acknowledged. The table, composed of starting addresses for all the interrupt mode 2 service routines, can be referred to as the interrupt mode two vector table. Each table entry should be word-sized if the Z380 MPU is in the Native Mode and longword-sized if in the Extended Mode, in either case it is even-aligned (least significant byte with address A0 = 0).

Interrupt Mode 3 Response For Maskable Interrupt /INT0

Interrupt mode 3 is similar to mode 2 except that a 16-bit vector is expected to be placed on the data bus D15-D0 by the I/O device during the interrupt acknowledge transaction. The interrupted PC is PUSHed onto the stack. IEF1 and IEF2 are reset to logic 0's so as to disable further maskable interrupt requests. The starting address of the service routine is fetched and loaded into the PC to resume execution from the memory location with an address composed of the I Extend contents as A31-A16 and the vector supplied by the I/O device as A15-A0. Again the starting address of the service routine is word-sized if the Z380 MPU is in the Native Mode and longword-sized if in the Extend Mode, in either case even-aligned.

BA23-BA14 *(Base Address 23-14).* In chip select scheme 1, the address signals A23-A16 of a memory transaction are compared with BA23-BA16 for a match, for those bits programmed for address matching in the Mid-range Memory Chip Select Register 1. The contents of this register have no effects in chip select scheme 2. Note that in order for one of /MCS3-/MCS0 to go active in a memory transaction in chip select scheme 1, the ENM1 bit in the Memory Selects Master Enable Register (described later) has to be at logic 1, all the address signals A31-A24 at logic 0s, and for those bits programmed for address matching, A23-A14 matching BA23-BA14. For the intended usage to maintain the mid-range memory area as a single block, MA23-MA14 (in that order) should be programmed for address matching with contiguous 1s followed by contiguous 0s. Note also that /MCS3-/MCS0 can be individually enabled to go active during refresh transactions, independent of the value programmed into the Memory Selects Master Enable Register.

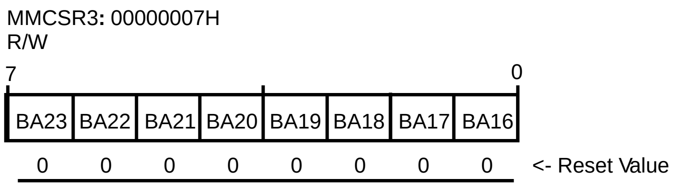

Figure 39. Mid-range Memory Chip Select Register 3

Lower Memory Wait Register

T1W2-T1W0 *(T1 Waits).* This binary field defines up to seven T1 wait states to be inserted in transactions accessing the lower memory area.

T2W1-T2W0 *(T2 Wait States).* This binary field defines up to three T2 wait states to be inserted in transactions accessing the lower memory area.

T3W2-T3W0 *(T3 Waits).* This binary field defines up to seven T3 wait states to be inserted in transactions accessing the lower memory area.

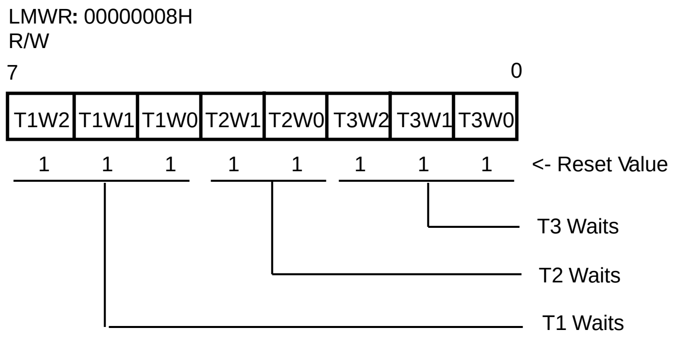

Figure 40. Lower Memory Waits Register

Upper Memory Wait Register

T1W2-T1W0 *(T1 Waits).* This binary field defines up to seven T1 wait states to be inserted in transactions accessing the upper memory area.

T2W1-T2W0 *(T2 Waits).* This binary field defines up to three T2 wait states to be inserted in transactions accessing the upper memory area.

T3W2-T3W0 *(T3 Waits).* This binary field defines up to seven T3 wait states to be inserted in transactions accessing the upper memory area.

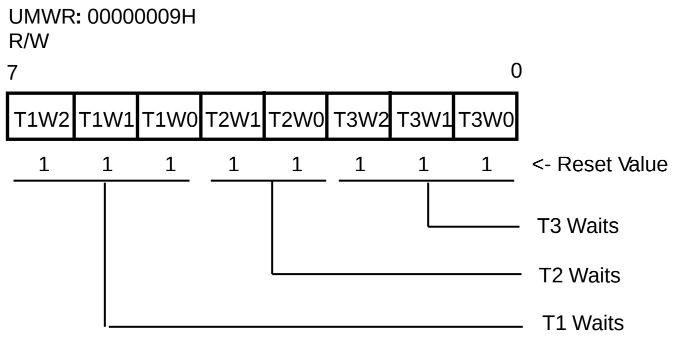

Figure 41. Upper Memory Waits Register

	no esc	ED esc	DD esc	FD esc	CB esc	ED-CB	DD-CB	FD-CB
CA	JP Z,nn	-	**LDCTL SR,n**	-	SET 1,D	-	-	-
CB	escape	escape	escape	escape	SET 1,E	-	-	-
CC	CALL Z,nn	**CALR Z,e**	**CALR Z,ee**	**CALR Z,eee**	SET 1,H	-	-	-
CD	CALL nn	**CALR e**	**CALR ee**	**CALR eee**	SET 1,L	-	-	-
CE	ADC A,n	-	**ADCW (IX+d)**	**ADCW (IY+d)**	SET 1,(HL)	-	SET 1,(IX+d)	SET 1,(IY+d)
CF	RST 1	**BTEST**	**MTEST**	-	SET 1,A	-	-	-
D0	RET NC	**LDCTL A,DSR**	**LDCTL A,XSR**	**LDCTL A,YSR**	SET 2,B	-	-	-
D1	POP DE	-	-	-	SET 2,C	-	-	-
D2	JP NC,nn	-	-	-	SET 2,D	-	-	-
D3	OUT (n),A	**OUTA (nn),A**	-	**OUTAW (nn),HL**	SET 2,E	-	-	-
D4	CALL NC,nn	**CALR NC,e**	**CALR NC,ee**	**CALR NC,eee**	SET 2,H	-	-	-
D5	PUSH DE	-	-	-	SET 2,L	-	-	-
D6	SUB n	**SUB HL,(nn) ****	**SUBW (IX+d)**	**SUBW (IY+d)**	SET 2,(HL)	-	SET 2,(IX+d)	SET 2,(IY+d)
D7	RST 2	-	-	-	SET 2,A	-	-	-
D8	RET C	**LDCTL DSR,A**	**LDCTL XSR,A**	**LDCTL YSR,A**	SET 3,B	-	-	-
D9	EXX	**EXALL**	**EXXX**	**EXXY**	SET 3,C	-	-	-
DA	JP C,nn	**LDCTL DSR,n**	**LDCTL XSR,n**	**LDCTL YSR,n**	SET 3,D	-	-	-
DB	IN A,(n)	**INA A,(nn)**	-	**INAW HL,(nn)**	SET 3,E	-	-	-
DC	CALL C,nn	**CALR C,e**	**CALR C,ee**	**CALR C,eee**	SET 3,H	-	-	-
DD	escape	reserved	reserved	reserved	SET 3,L	-	-	-
DE	SBC A,n	-	**SBCW (IX+d)**	**SBCW (IY+d)**	SET 3,(HL)	-	SET 3,(IX+d)	SET 3,(IY+d)
DF	RST 3	-	-	-	SET 3,A	-	-	-
E0	RET PO	**LDIW**	-	-	SET 4,B	-	-	-
E1	POP HL	-	POP IX	POP IY	SET 4,C	-	-	-
E2	JP PO,nn	**INIW**	-	-	SET 4,D	-	-	-
E3	EX (SP),HL	**OUTIW**	EX (SP),IX	EX (SP),IY	SET 4,E	-	-	-
E4	CALL PO,nn	**CALR PO,e**	**CALR PO,ee**	**CALR PO,eee**	SET 4,H	-	-	-
E5	PUSH HL	-	PUSH IX	PUSH IY	SET 4,L	-	-	-
E6	AND n	-	**ANDW (IX+d)**	**ANDW (IY+d)**	SET 4,(HL)	-	SET 4,(IX+d)	SET 4,(IY+d)
E7	RST 4	-	-	-	SET 4,A	-	-	-
E8	RET PE	**LDDW**	-	-	SET 5,B	-	-	-
E9	JP (HL)	-	JP (IX)	JP (IY)	SET 5,C	-	-	-
EA	JP PE,nn	**INDW**	-	-	SET 5,D	-	-	-
EB	EX DE,HL	**OUTDW**	-	-	SET 5,E	-	-	-
EC	CALL PE,nn	**CALR PE,e**	**CALR PE,ee**	**CALR PE,eee**	SET 5,H	-	-	-
ED	escape	reserved	reserved	reserved	SET 5,L	-	-	-
EE	XOR n	-	**XORW (IX+d)**	**XORW (IY+d)**	SET 5,(HL)	-	SET 5,(IX+d)	SET 5,(IY+d)
EF	RST 5	-	-	-	SET 5,A	-	-	-
F0	RET P	**LDIRW**	-	-	SET 6,B	-	-	-
F1	POP AF	-	-	-	SET 6,C	-	-	-
F2	JP P,nn	**INIRW**	-	-	SET 6,D	-	-	-
F3	DI	**OTIRW**	**DI n**	-	SET 6,E	-	-	-
F4	CALL P,nn	**CALR P,e**	**CALR P,ee**	**CALR P,eee**	SET 6,H	-	-	-
F5	PUSH AF	-	-	**PUSH nn**	SET 6,L	-	-	-
F6	OR n	-	**ORW (IX+d)**	**ORW (IY+d)**	SET 6,(HL)	-	SET 6,(IX+d)	SET 6,(IY+d)
F7	RST 6	**SETC LCK**	**SETC LW**	**SETC XM**	SET 6,A	-	-	-
F8	RET M	**LDDRW**	-	-	SET 7,B	-	-	-
F9	LD SP,HL	-	LD SP,IX	LD SP,IY	SET 7,C	-	-	-
FA	JP M,nn	**INDRW**	-	-	SET 7,D	-	-	-
FB	EI	**OTDRW**	**EI n**	-	SET 7,E	-	-	-
FC	CALL M,nn	**CALR M,e**	**CALR M,ee**	**CALR M,eee**	SET 7,H	-	-	-
FD	escape	reserved	reserved	reserved	SET 7,L	-	-	-
FE	CP n	-	**CPW (IX+d)**	**CPW (IY+d)**	SET 7,(HL)	-	SET 7,(IX+d)	SET 7,(IY+d)
FF	RST 7	**RESC LCK**	**RESC LW**	**-**	SET 7,A	-	-	-