به نام خدا

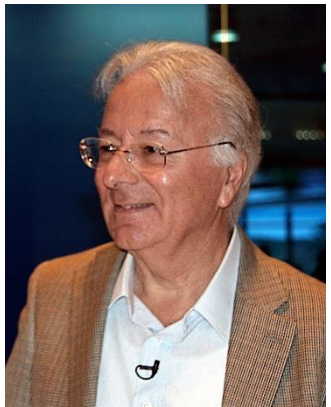
آشنایی با نحوه عملکرد µpها

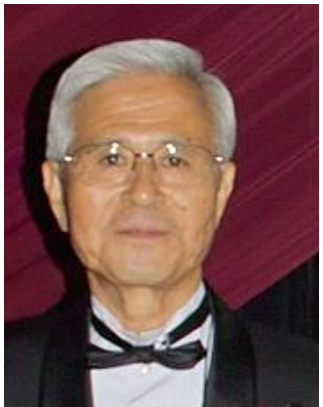
مطالعه موردی Z80

Dr. Aref Karimiafshar
A.karimiafshar@ec.iut.ac.ir

میکروپروسسور Z80

- By Zilog
- 1976
- Enhancement of the Intel 8080
- Used mainly at embedded systems
  - Also, desktop and home computers
    - from the 1970s to the mid-1980s.
- Designed by
  - Federico Faggin and Masatoshi Shima





At Intel led the work on the Intel 4004, the 8080 and several other ICs

میکروپروسسور Z80

- 8-bit microprocessor
- Instruction set is binary compatible with Intel 8080
  - most 8080 code would run unmodified on the Z80
- groups instructions into the following categories:
  - Load and Exchange
  - Block Transfer and Search
  - Arithmetic and Logical
  - Rotate and Shift
  - Bit Manipulation (set, reset, test)
  - Jump, Call and Return
  - Input/Output
  - Basic CPU Control

No explicit multiply instructions are available in the original Z80

میکروپروسسور Z80

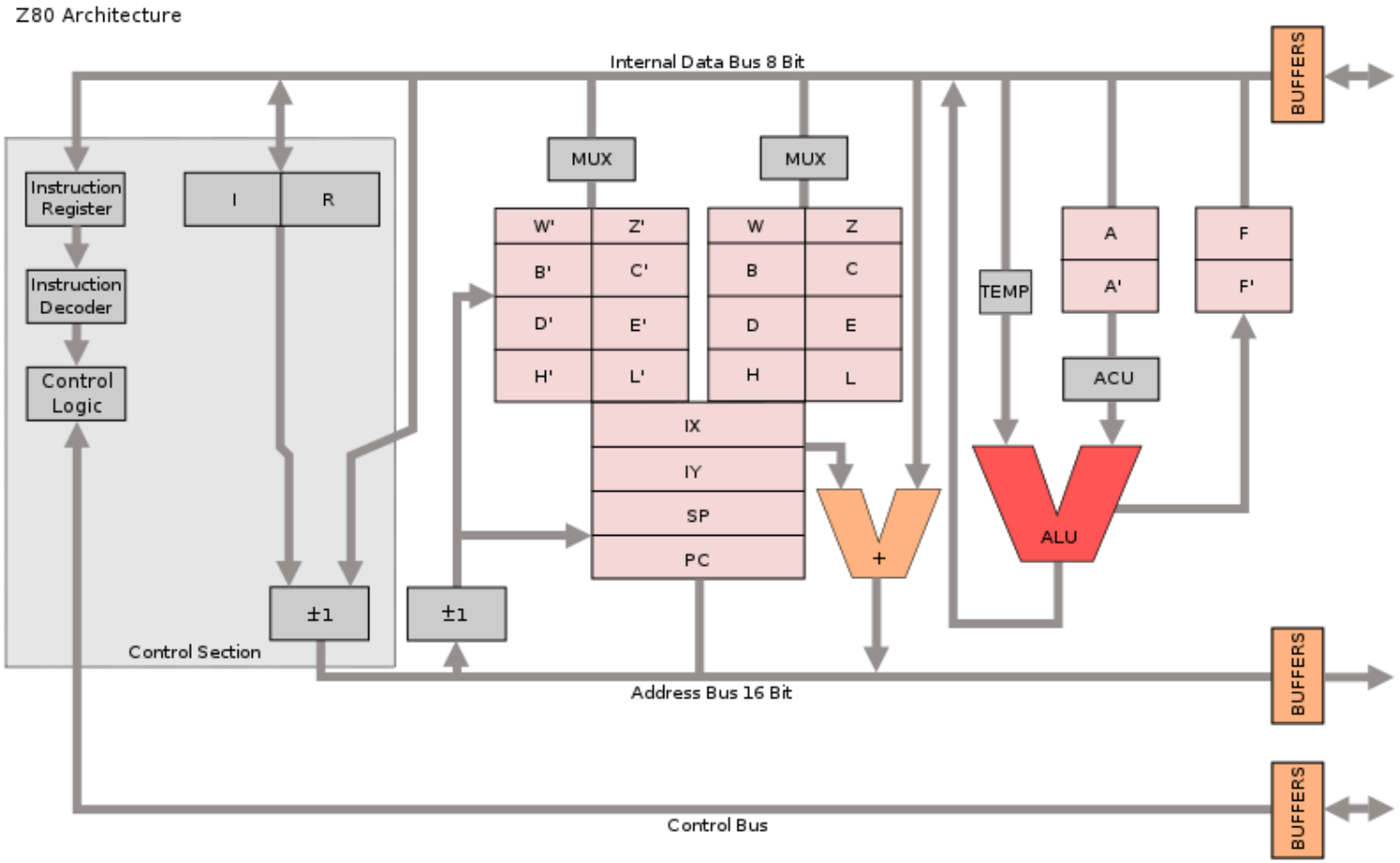
Hitachi HD64180

Zilog Z180

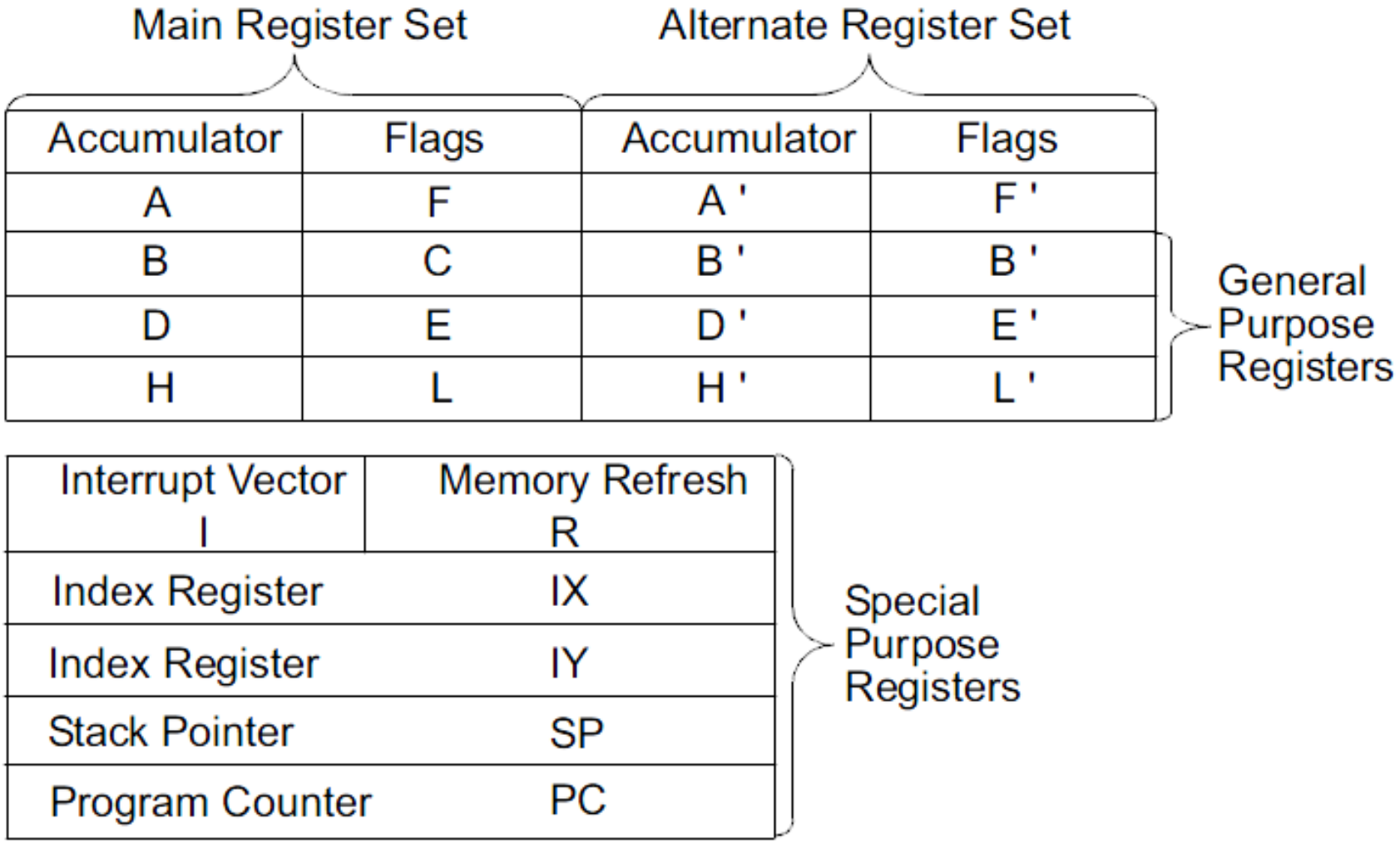
Toshiba TMPZ84C015

Zilog Z280

معماری Z80

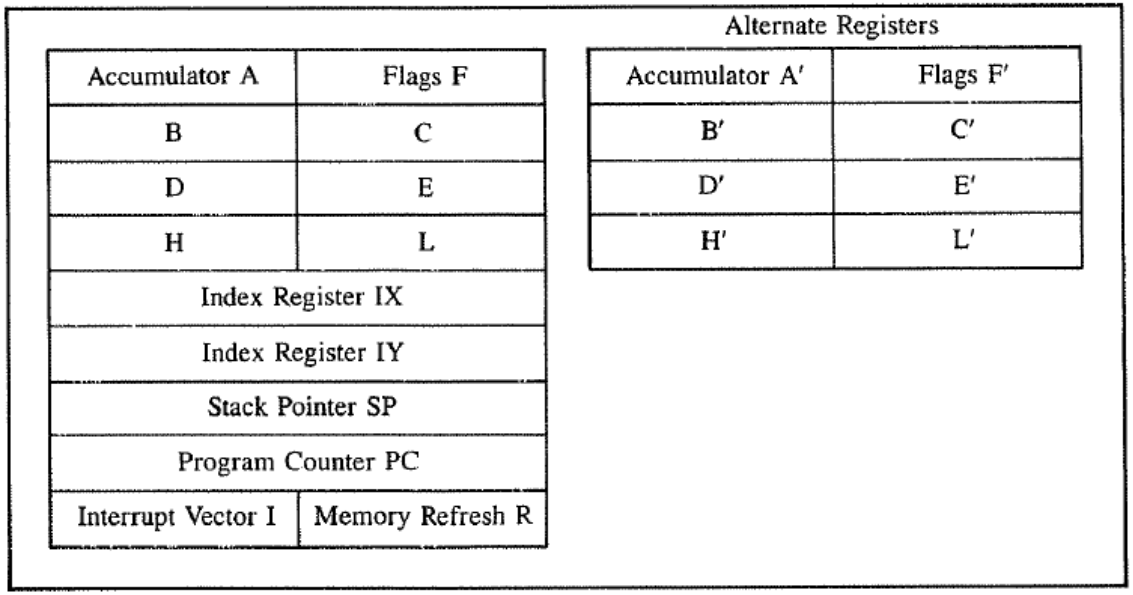

Z80 Architecture
Internal Data Bus 8 Bit
BUFFERS
Instruction Register
Instruction Decoder
Control Logic
I
R
MUX
MUX
W'
Z'
B'
C'
D'
E'
H'
L'
W
Z
B
C
D
E
H
L
IX
IY
SP
PC
TEMP
A
A'
F
F'
ACU
ALU
+
±1
±1
Control Section
Address Bus 16 Bit
Control Bus

معماری Z80

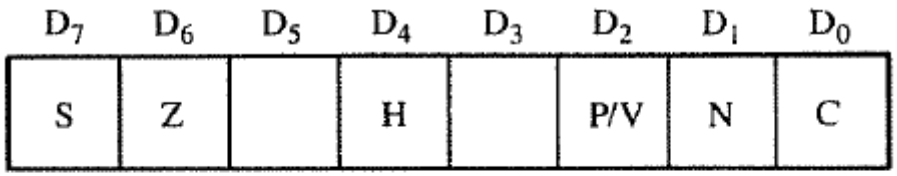

Main Register Set
Alternate Register Set
Accumulator
Flags
Accumulator
Flags
A
F
A '
F '
B
C
B '
B '
D
E
D '
E '
H
L
H '
L '
General Purpose Registers
Interrupt Vector I
Memory Refresh R
Index Register
IX
Index Register
IY
Stack Pointer
SP
Program Counter
PC
Special Purpose Registers

Accumulator

The accumulator is an 8-bit register that is part of the Arithmetic/Logic unit (ALU) and is also identified as register A. This register is used to store 8-bit data and to perform arithmetic and logic operations. The result of an operation performed in the ALU is also stored in the accumulator. For example, in an 8-bit addition, the instruction ADD always assumes that one of the numbers is the byte in the accumulator, and the result of the addition is stored in the accumulator by replacing the previous byte.



Flag Registers



S = Sign

Z = Zero

H = Half-Carry

P/V = Parity/Overflow

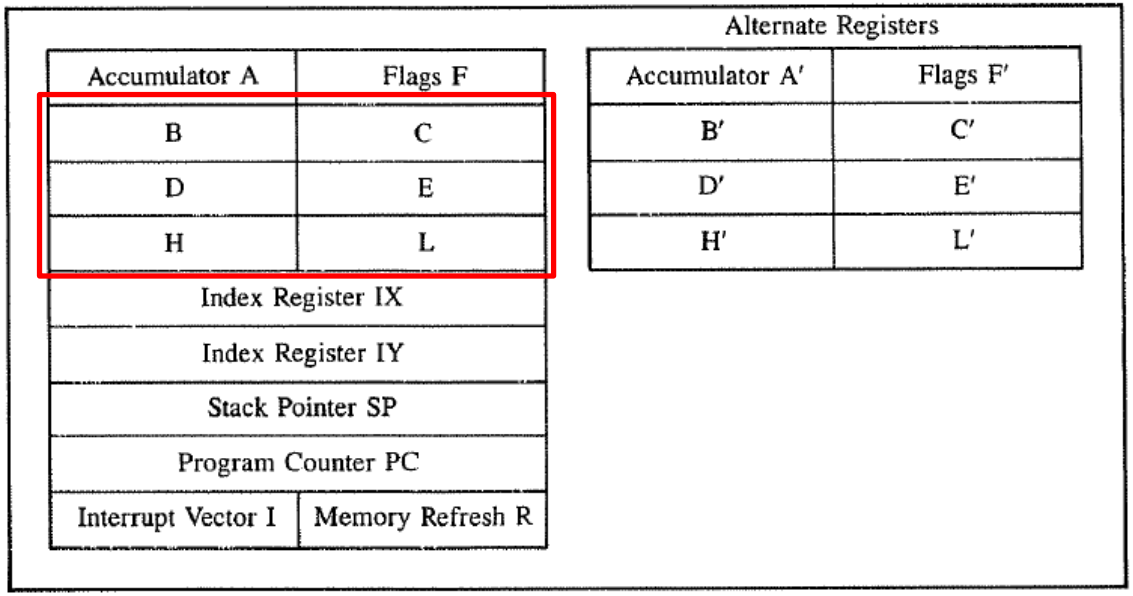
N = Add/Subtract

C = Carry

General Purpose Registers

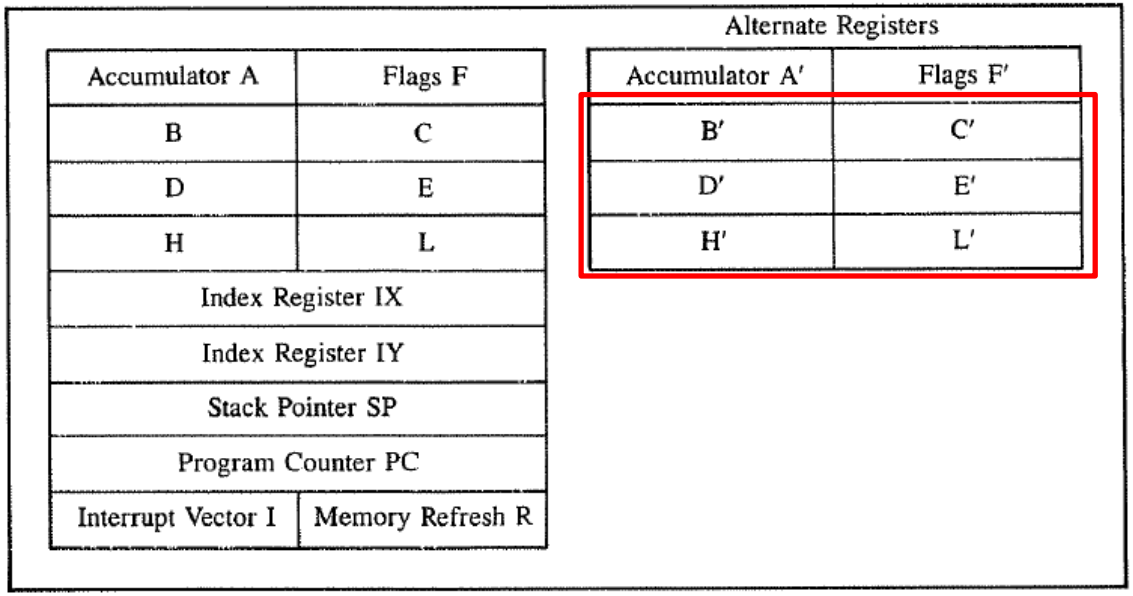
The Z80 microprocessor has six programmable general-purpose registers named B, C, D, E, H, and L, as shown in Figure 3.1. These are 8-bit registers used for storing data during the program execution. They can be combined as register pairs—BC, DE, HL—to perform 16-bit operations or to hold memory addresses.

The programmer can use these registers to *load* or copy data. For example, the instruction LD B, C copies the data from register C into register B. Conceptually, these registers can be viewed as memory locations, except that they are built inside the microprocessor and identified by specific names. Some microprocessors do not have this type of register; instead, they use memory as their registers.



Alternate Registers

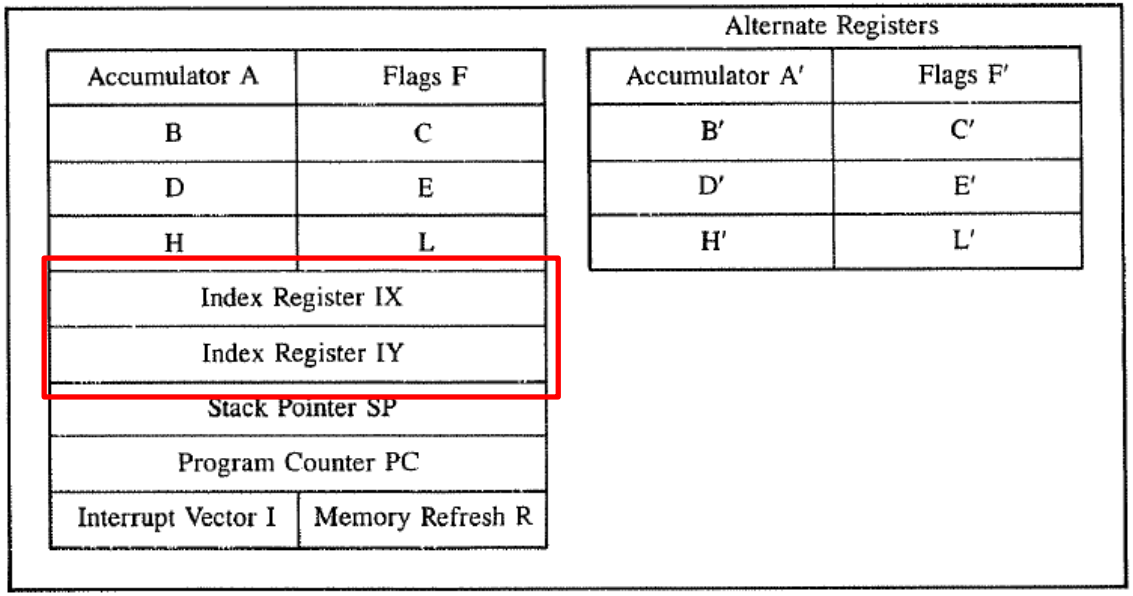
In addition to the general-purpose registers, the Z80 includes a similar set of six **alternate registers** designated as B′, C′, D′, E′, H′, and L′. These are 8-bit registers used for exchanging data with the general-purpose registers. They are not directly available to the programmer, except through the exchange instructions.



Index Registers

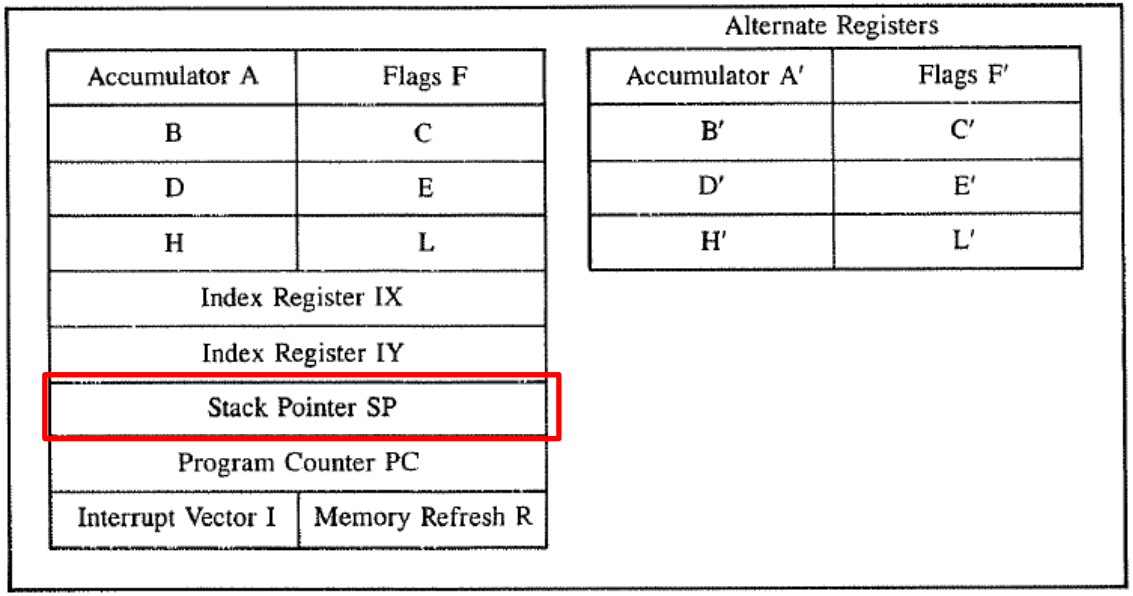
The Z80 has two 16-bit **index registers** called IX and IY. Each register is used to specify a memory address by the 16-bit address it holds and a displacement count. For example, if the IX register holds 2050_H, a higher memory address such as 2060_H can be specified by adding the displacement count of 10_H. Similarly, a lower memory address such as 2040_H can be specified by adding the negative of 10_H in 2's complement.

In addition to the index registers, the HL pair is frequently used as a memory pointer. Similarly, the BC and DE pairs can be used also as memory pointers in a limited way. However, no displacement byte can be added to the contents of these pairs.



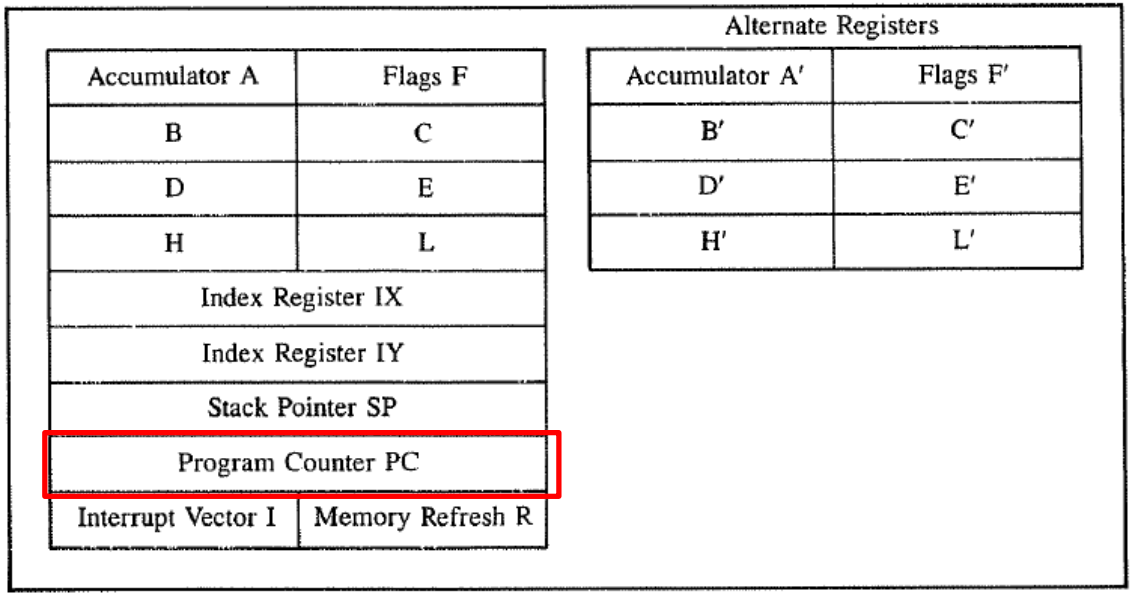
Stack Pointer (SP)

The stack pointer is also a 16-bit register used to point to the memory location called the **stack.** The stack is a defined area of memory locations in R/W memory, and the beginning of the stack is defined by loading a 16-bit address into the stack pointer.



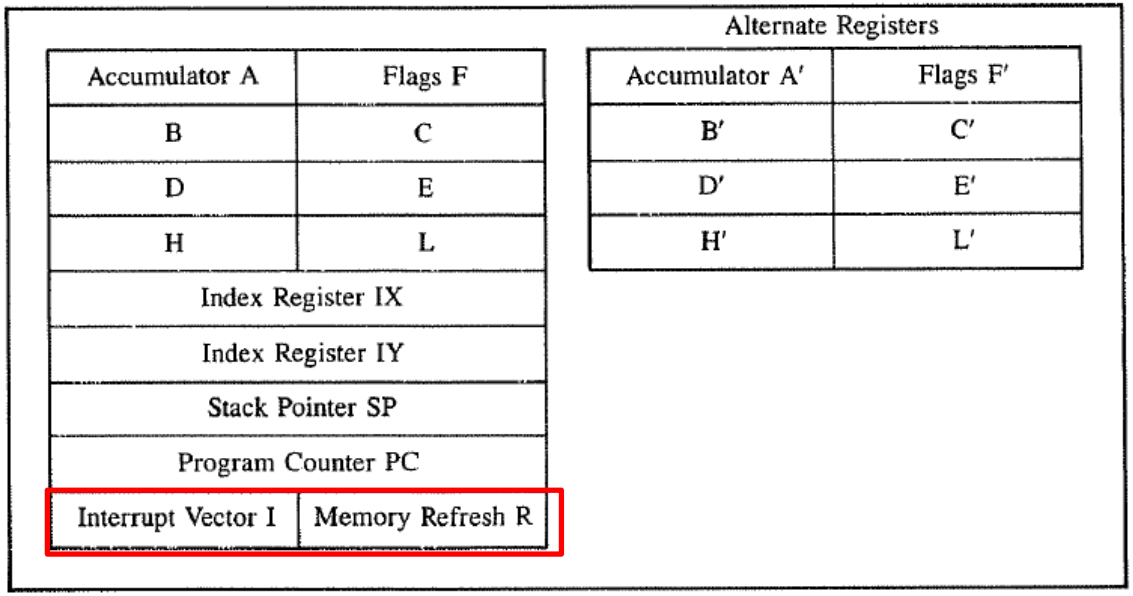
Program Counter (PC)

This register functions as a 16-bit counter. The microprocessor uses this register to sequence the execution of instructions. The program counter points to the memory address from which the next byte is to be fetched, and when the microprocessor places an address on the address bus to fetch the byte from memory, it then increments the program counter by one to point to the next memory location.



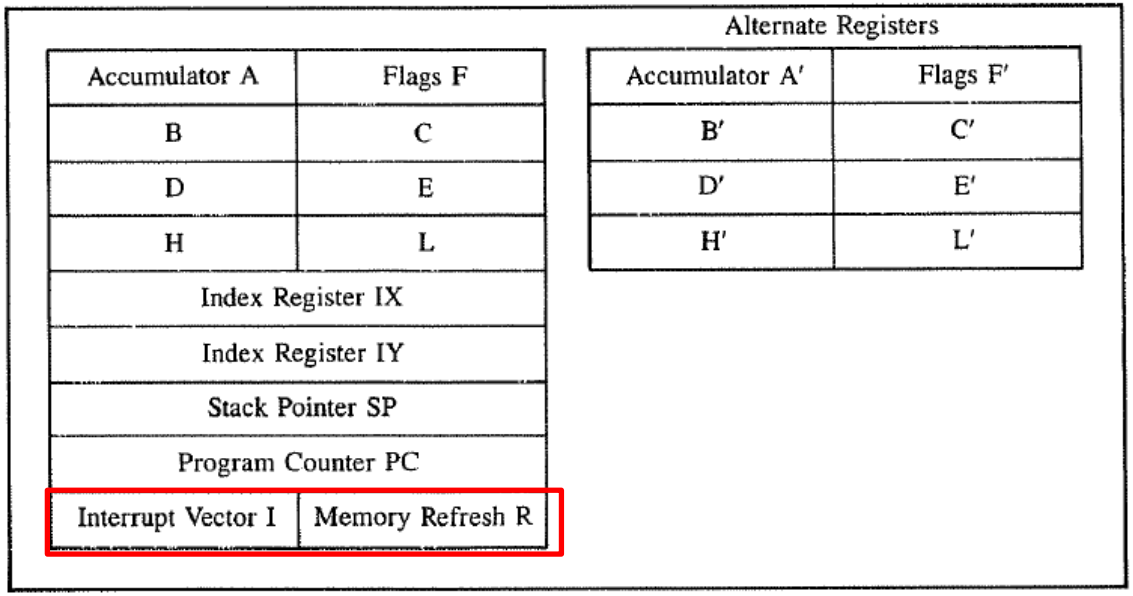
Interrupt Vector Register (I)

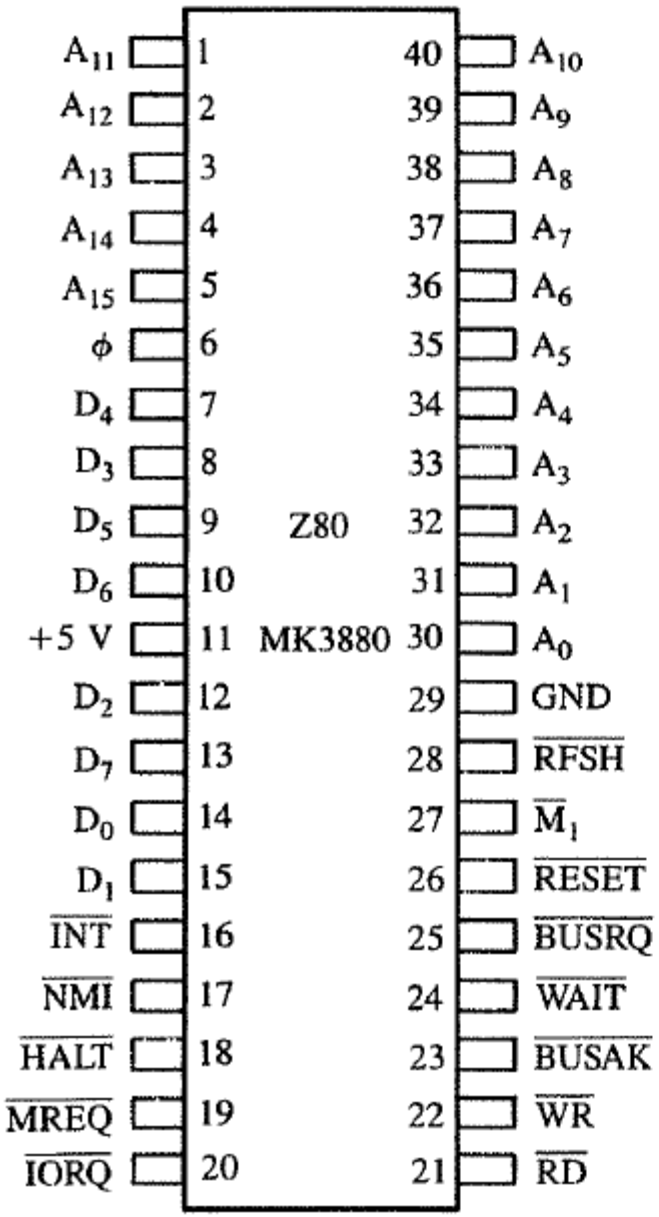
This is an 8-bit register used in the interrupt process. When an external device interrupts the microprocessor with a request to do something else, the microprocessor should be directed to a 16-bit address in memory where it can find what to do next. The I register is used to store the high-order eight bits of the 16-bit address; the low-order eight bits must be supplied by the interrupting device.

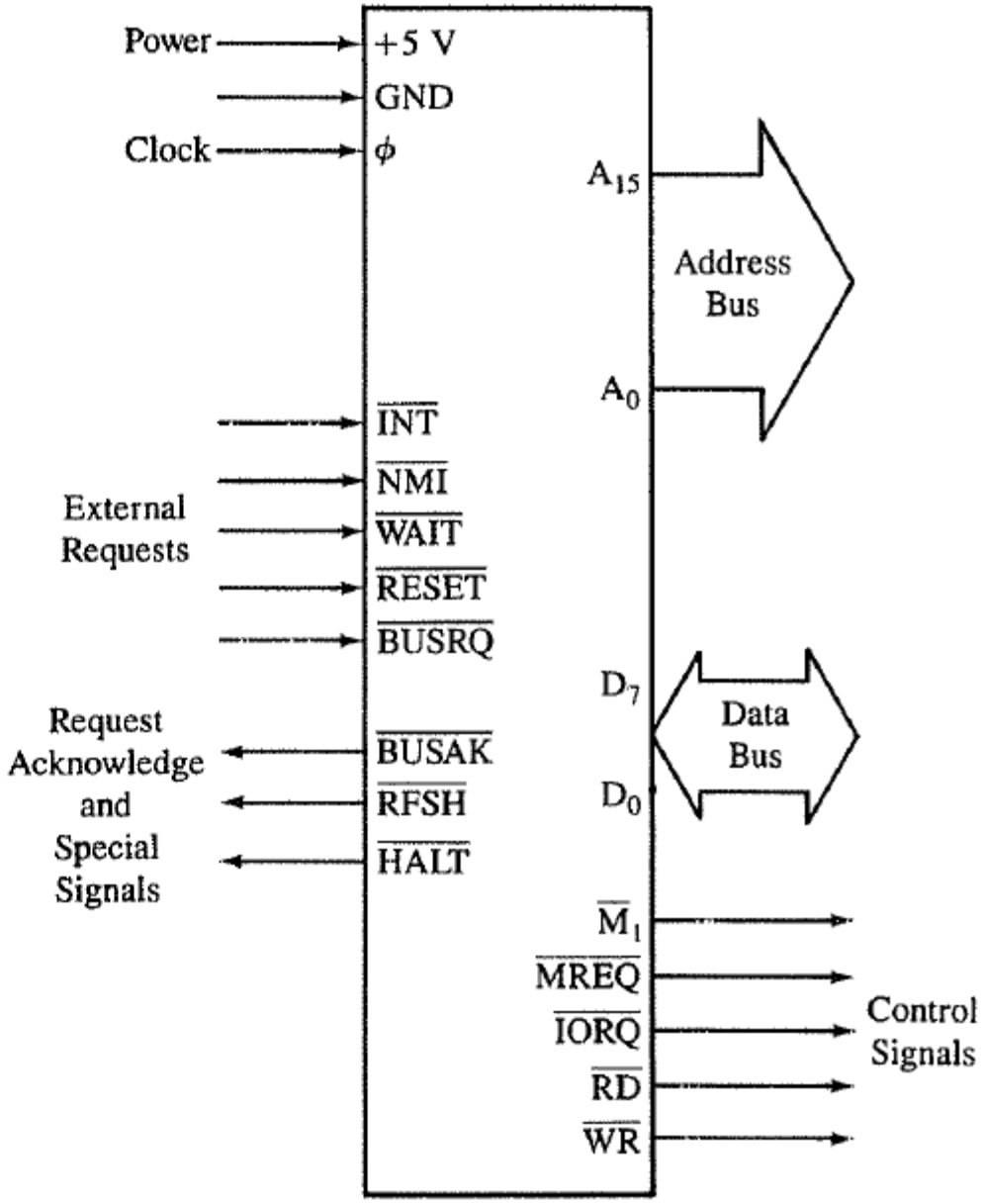


Memory Refresh Register (R)

The memory refresh register (R) is also an 8-bit register which is used as a 7-bit counter to provide an address of memory cells to be refreshed in dynamic memory.




A11 1
A12 2
A13 3
A14 4
A15 5
φ 6
D4 7
D3 8
D5 9
D6 10
+5 V 11
D2 12
D7 13
D0 14
D1 15
INT 16
NMI 17
HALT 18
MREQ 19
IORQ 20
40 A10
39 A9
38 A8
37 A7
36 A6
35 A5
34 A4
33 A3
32 A2
31 A1
30 A0
29 GND
28 RFSH
27 M1
26 RESET
25 BUSRQ
24 WAIT
23 BUSAK
22 WR
21 RD
Z80
MK3880


Power
Clock
+5 V
GND
φ
A15
A0
Address Bus
INT
NMI
WAIT
RESET
BUSRQ
External Requests
D7
D0
Data Bus
Request Acknowledge and Special Signals
BUSAK
RFSH
HALT
M1
MREQ
IORQ
RD
WR
Control Signals

Control Signals

- $\overline{M_1}$—Machine Cycle One: This is an active low signal indicating that an opcode is being fetched from memory. This signal is also used in an interrupt operation to generate an interrupt acknowledge signal.
- $\overline{MREQ}$—Memory Request: This is an active low tri-state line. This signal indicates that the address bus holds a valid address for a memory read or write operation.
- $\overline{IORQ}$—I/O Request: This is an active low tri-state line. This signal indicates that the low-order address bus (A_7–A_0) holds a valid address for an I/O read or write operation. This signal is also generated for an interrupt operation.
- $\overline{RD}$—Read: This is an active low tri-state line. This signal indicates that the microprocessor is ready to read data from memory or an I/O device. This signal should be used in conjunction with $\overline{MREQ}$ for the Memory Read ($\overline{MEMRD}$) operation and with IORQ for the I/O Read ($\overline{IORD}$) operation.
- $\overline{WR}$—Write: This is an active low tri-state line. This signal indicates that the microprocessor has already placed a data byte on the data bus and is ready to write into memory or an I/O device. This signal should be used in conjunction with $\overline{MREQ}$ for the Memory Write ($\overline{MEMWR}$) operation and with $\overline{IORQ}$ for the I/O Write ($\overline{IOWR}$) operation.

External Requests

- $\overline{\text{RESET}}$—Reset: This is an active low signal used to reset the microprocessor. When $\overline{\text{RESET}}$ is activated, the program counter (PC), the interrupt register (I), and the memory refresh register (R) are all cleared to 0. During the reset time, the address bus and the data bus are in high impedance state, and all control signals become inactive. This signal also disables interrupt and refresh. The $\overline{\text{RESET}}$ signal can be initiated by an external key or switch and must be active at least for three clock periods to complete the reset operation.
- $\overline{\text{INT}}$—Interrupt Request: This is an active low signal, initiated by an I/O device to interrupt the microprocessor operation. When the microprocessor accepts the interrupt request, it acknowledges by activating the $\overline{\text{IORQ}}$ signal during the M_1 cycle. The $\overline{\text{INT}}$ signal is maskable, meaning it can be disabled through a software instruction.
- $\overline{\text{NMI}}$—Nonmaskable Interrupt: This is a nonmaskable interrupt; it cannot be disabled. It is activated by a negative edge-triggered signal from an external source. This signal is used primarily for implementing emergency procedures. There is no signal or pin to acknowledge this signal; it is accepted provided the Bus Request signal is inactive.

External Requests

- $\overline{\text{BUSRQ}}$—Bus Request: This is an active low signal initiated by external I/O devices such as the DMA (Direct Memory Access) controller. An I/O device can send a low signal to $\overline{\text{BUSRQ}}$ to request the use of the address bus, the data bus, and the control signals. The external device can use the buses, and when its operations are complete, it returns the control to the microprocessor. This signal is used primarily for the direct
- $\overline{\text{WAIT}}$—Wait: This is an active low signal and can be used by memory or I/O devices to add clock cycles to extend the Z80 operations. This signal is used when the response time of memory or I/O devices is slower than that of the Z80. When this signal goes low, it indicates to the microprocessor that the addressed memory or I/O device is not yet ready for data transfer. As long as this signal is low, the Z80 keeps adding cycles to its operation.

Bus Acknowledge and Special Signals

- $\overline{\text{BUSAK}}$—Bus Acknowledge: This is an active low output signal initiated by the Z80 in response to the Bus Request signal. This signal indicates to the requesting device that the address bus, the data bus, and the control signals ($\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{MREQ}}$, and $\overline{\text{IORQ}}$) have entered into the high impedance state and can be used by the requesting device.
- $\overline{\text{HALT}}$—Halt: This is an active low output signal used to indicate that the MPU has executed the HALT instruction.
- $\overline{\text{RFSH}}$—Refresh: This is an active low signal indicating that the address bus A_6-A_0 (low-order seven bits) holds a refresh address of dynamic memory; it should be used in conjunction with $\overline{\text{MREQ}}$ to refresh memory contents.

Power and Frequency Signals

- ϕ—Clock: This pin is used to connect a single phase frequency source. The Z80 does not include a clock circuit on its chip; the circuit must be built separately.
- +5 V and GND—These pins are for a power supply and ground reference; the Z80 requires one +5 V power source.

Instructions

- 158 different instructions
  - Each instruction
    - Operation code
    - Operand
- Instruction
  - 1-Byte
  - Multi-byte
- No direct relationship
  - Between number of byte of instruction and number of operations

Instruction

OUT (10H), A

- Byte 1: OUT → This is the opcode to output data.
- Byte 2: (10H*), A → This is the operand to specify that the byte should be sent from the accumulator to Port 10_H.

But the Z80 has to perform three operations: (1) read Byte 1 from memory, (2) read Byte 2 from memory, (3) send data to port 10_H.

Memory Address	Machine Code	Mnemonics	Memory Contents
2050	D3	OUT (10H), A ; 2050	1 1 0 1 0 0 1 1 = $D3_H$
2051	10	; 2051	0 0 0 0 1 0 1 0 = 10_H

Instruction

instructions are divided into a few basic operations called machine cycles,

machine cycles are divided into precise *system clock periods*.

external communication functions

1. Memory Read and Write.
2. I/O Read and Write.
3. Request Acknowledge.

Instruction cycle is defined as the time required to complete the execution of an instruction. The Z80 instruction cycle consists of one to six machine cycles or one to six operations.

Machine cycle is defined as the time required to complete one operation of accessing memory, accessing I/O, or acknowledging an external request. This cycle may consist of three to six T-states.

T-state is defined as one subdivision of the operation performed in one clock period. These subdivisions are internal states synchronized with the system clock, and each T-state is precisely equal to one clock period. The terms T-state and clock period are often used synonymously.

Instruction

The Z80 Machine Cycles and Control Signals

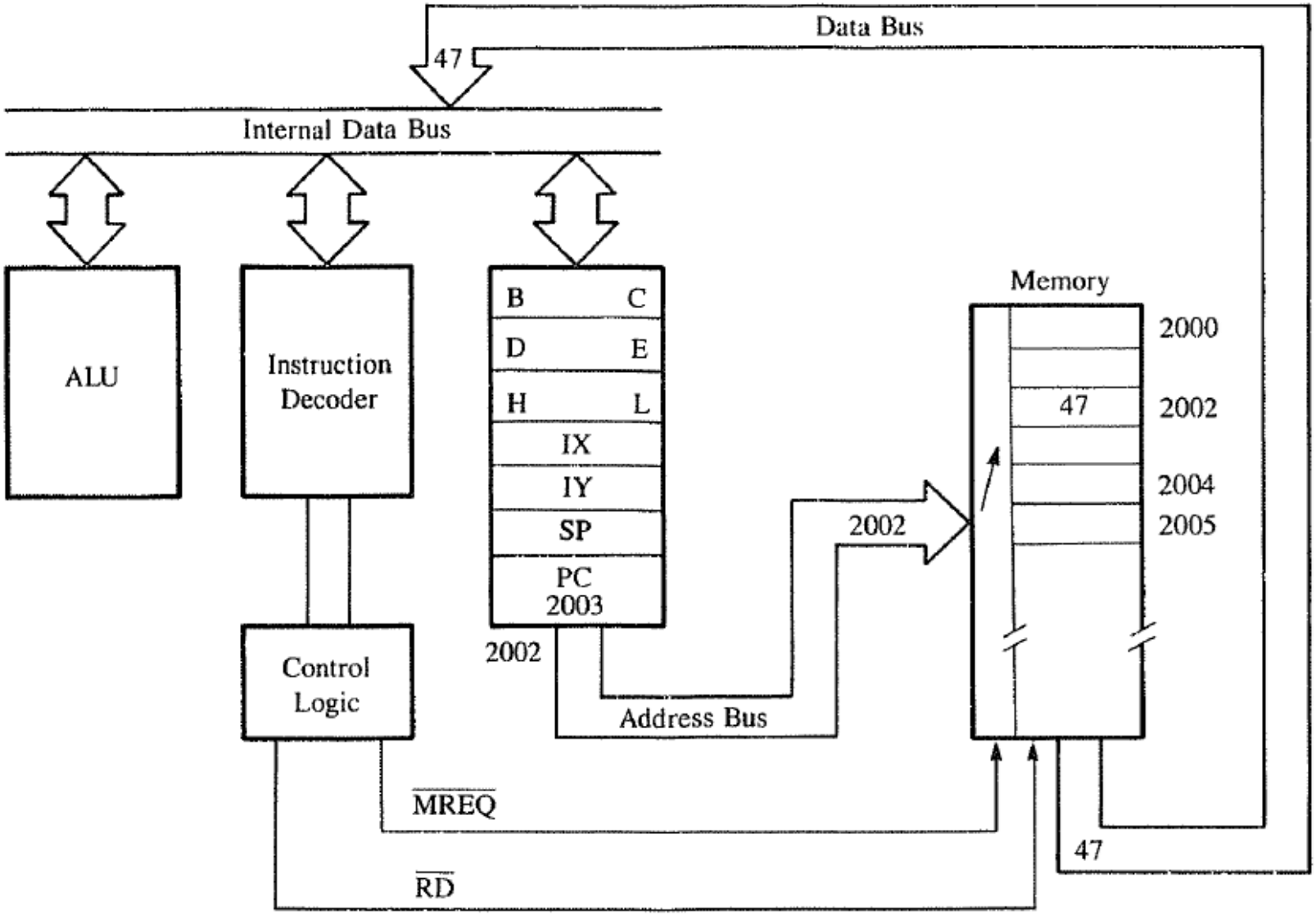
Machine Cycle	$\overline{M_1}$	$\overline{MREQ}$	$\overline{IORQ}$	$\overline{RD}$	$\overline{WR}$
Opcode Fetch ($\overline{M_1}$)	0	0	1	0	1
Memory Read	1	0	1	0	1
Memory Write	1	0	1	1	0
I/O Read	1	1	0	0	1
I/O Write	1	1	0	1	0
Interrupt Acknowledge	0	1	0	1	1
Non-maskable Interrupt	0	0	1	0	1
Bus Acknowledge ($\overline{BUSAK} = 0$)	1	Z	Z	Z	Z

NOTE: Logic 0 = Active, Logic 1 = Inactive, Z = High Impedance

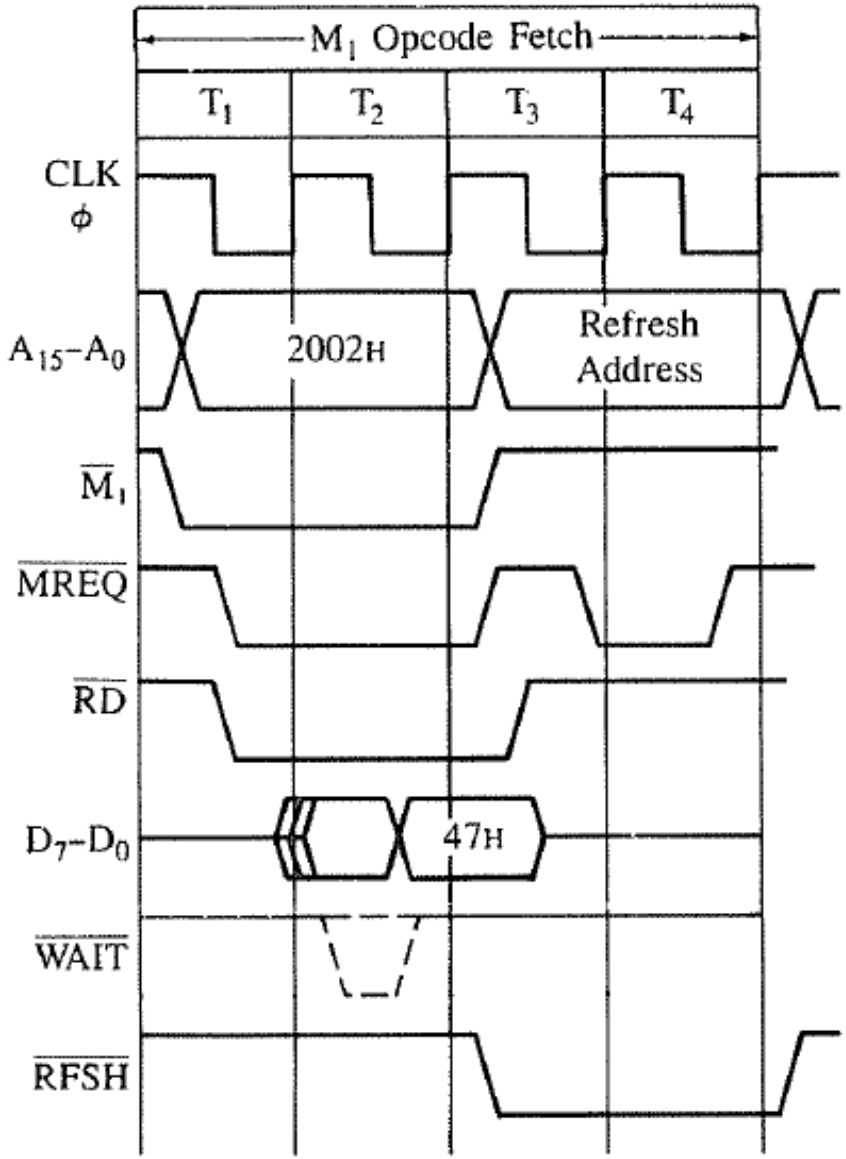
Opcode Fetch Machine Cycle

1. The Z80 places the contents of the program counter (2002_H) on the address bus, and increments the program counter to the next address, 2003_H. The program counter always points to next byte to be executed.
2. The address is decoded by the external decoding circuit and the register 2002_H is identified.
3. The Z80 sends the control signals ($\overline{MREQ}$ and $\overline{RD}$) to enable the memory output buffer.
4. The contents of the memory register (opcode 47_H) are placed on the data bus and brought into the instruction decoder of the microprocessor.
5. The Z80 decodes the opcode and executes the instruction, meaning it copies the contents of the accumulator into register B.

Opcode Fetch Machine Cycle



Opcode Fetch Machine Cycle

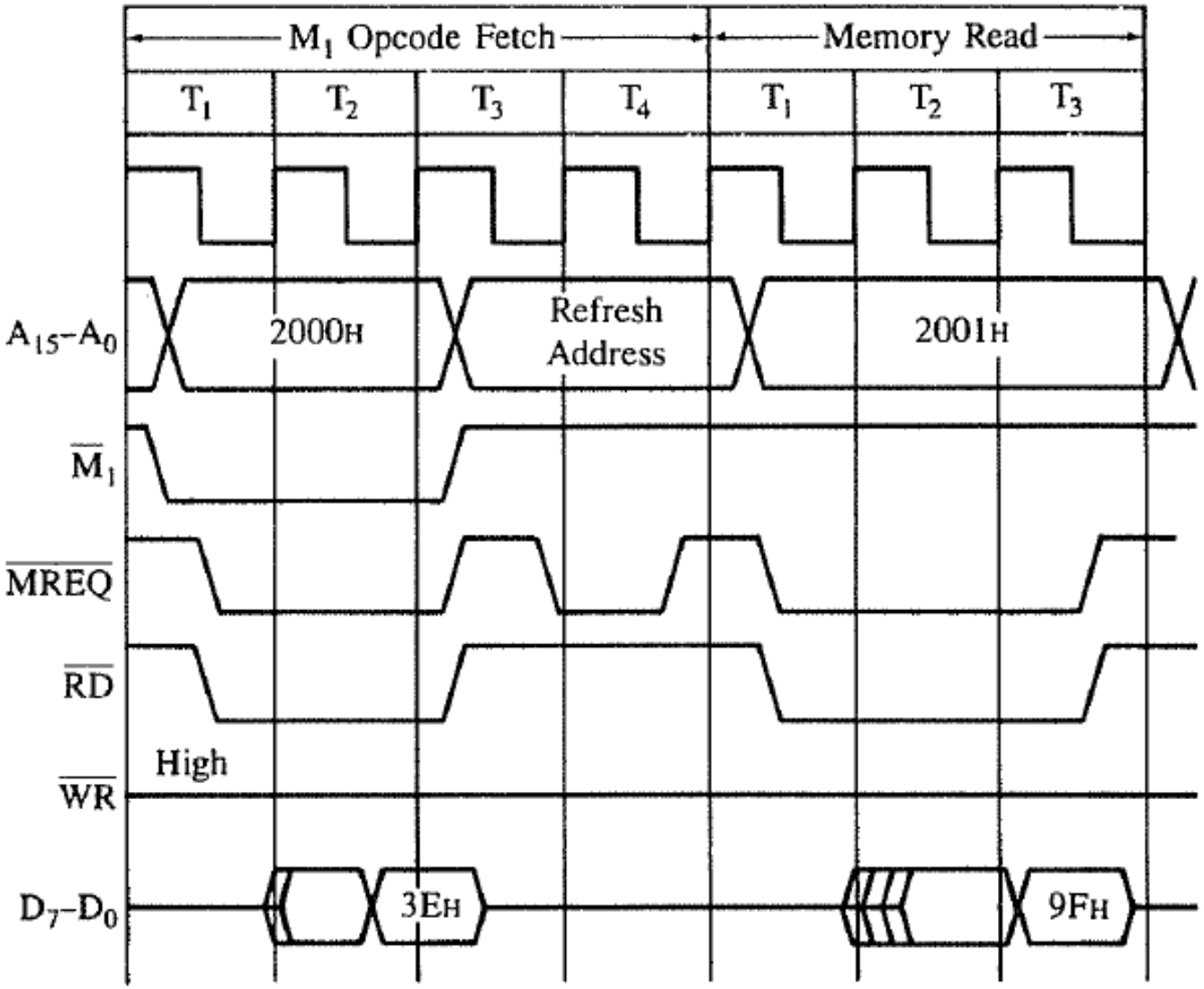


Memory Read Machine Cycle

Address	Machine Code			Instruction	Comment
2000_H	0 0 1 1 1 1 1 0	→	3E	LD A, 9FH	;Load 9FH in the accumulator
2001_H	1 0 0 1 1 1 1 1	→	9F		

1. The first machine cycle (Opcode Fetch) is identical in bus timings with the machine cycle illustrated in Example 3.2, except for the bus contents. The address bus contains 2000_H and the data bus contains the opcode $3E_H$. When the Z80 decodes the opcode during the T_3 state, it realizes that a second byte must be read.
2. After the completion of the Opcode Fetch cycle, the Z80 places the address 2001_H on the address bus and increments the program counter to the next address, 2002_H. To differentiate the second cycle from the Opcode Fetch cycle, the M_1 signal remains inactive (high).
3. After the falling edge of T_1 of the Memory Read cycle, the control signals $\overline{MREQ}$ and $\overline{RD}$ are asserted. These signals along with the memory address are used to identify the register 2001_H and enable the memory chip.
4. After the leading edge of T_3, the Z80 activates the data bus as an input bus; memory places the data byte $9F_H$ on the data bus, and the Z80 reads and stores the byte in the accumulator during T_3.
5. After the falling edge of T_3, both control signals become inactive (high), and at the end of T_3, the next machine cycle begins.

Memory Read Machine Cycle

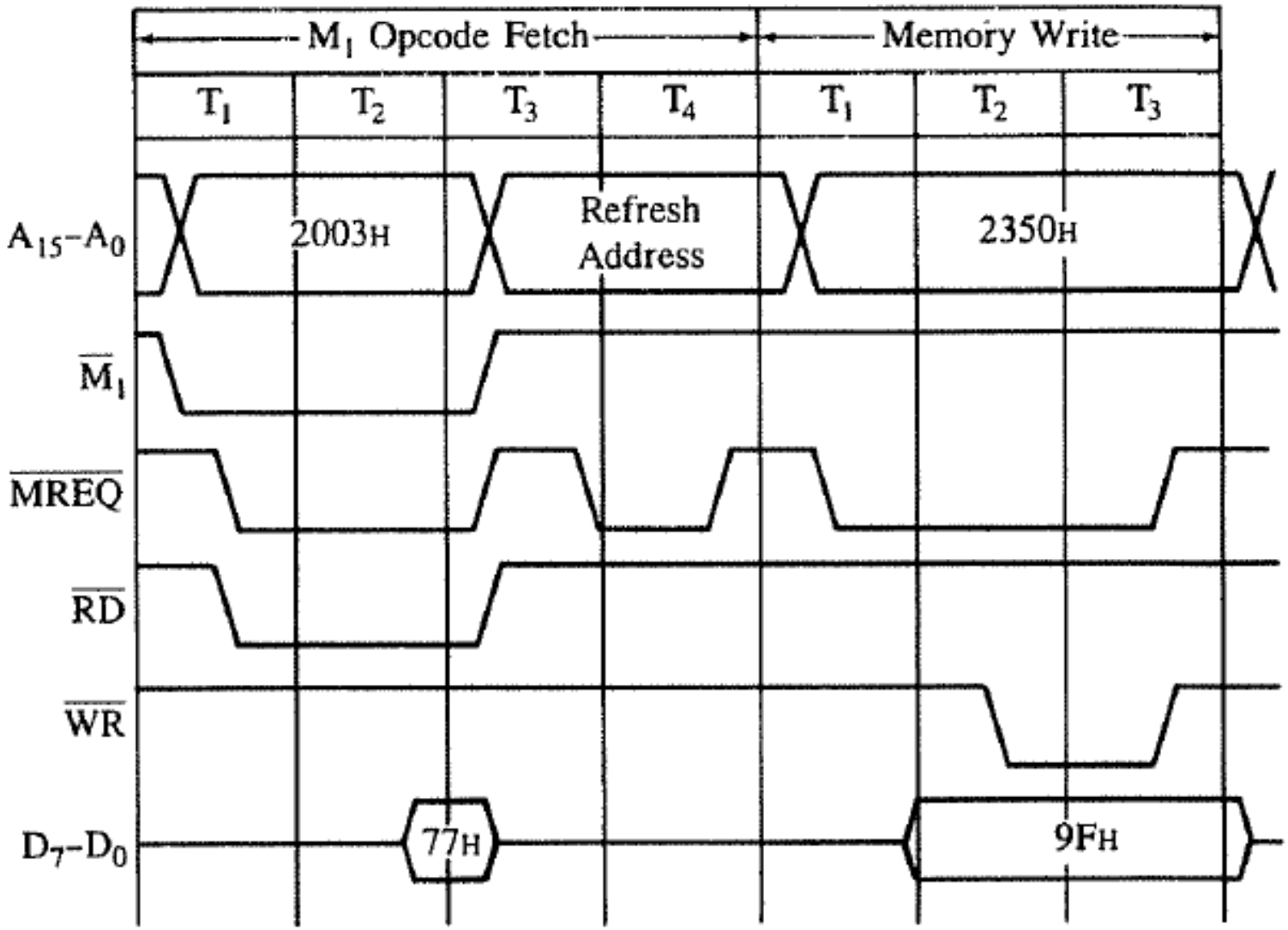


Memory Write Machine Cycle

Instruction: LD (HL), A ;Copy contents of the accumulator into memory location, the address of which is stored in HL register.

1. In the Opcode Fetch machine cycle, the Z80 places the address 2003_H on the address bus and gets the code 77_H by using the control signals $\overline{MREQ}$ and $\overline{RD}$ as in the previous examples. The program counter is also incremented to the next address, 2004_H.
2. During the T_3 and T_4 states, the Z80 decodes the machine code 77_H and prepares for the memory write operation.
3. At the beginning of the next machine cycle (Memory Write), it places the contents (2350_H) of the HL register on the address bus. At the falling edge of T_1, $\overline{MREQ}$ goes low and the data byte $9F_H$ from the accumulator is placed on the data bus.
4. After allowing one T-state (after $\overline{MREQ}$) to stabilize the address, the Z80 asserts the control signal Write ($\overline{WR}$), which is used to write the data byte at the address shown on the address bus.
5. After the falling edge of T_3, both control signals become inactive, and one-half T-state later, the data bus goes into high impedance state.

Memory Write Machine Cycle



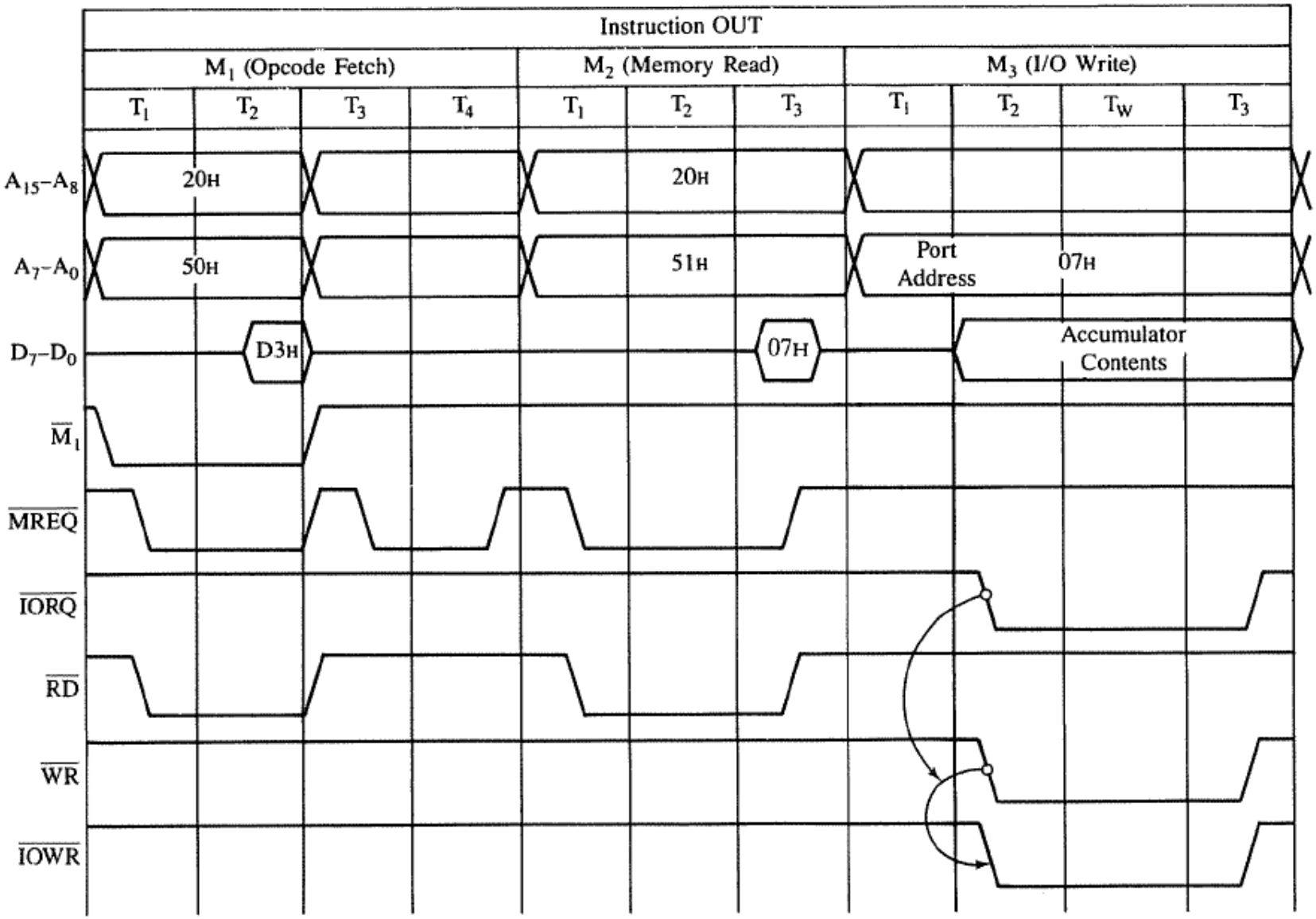
Out Instruction

Memory Address	Machine Code	Mnemonics	Memory Contents
2050	D3	OUT (07H), A ; 2050	1 1 0 1 0 0 1 1 = $D3_H$
2051	07	; 2051	0 0 0 0 0 1 1 1 = 07_H

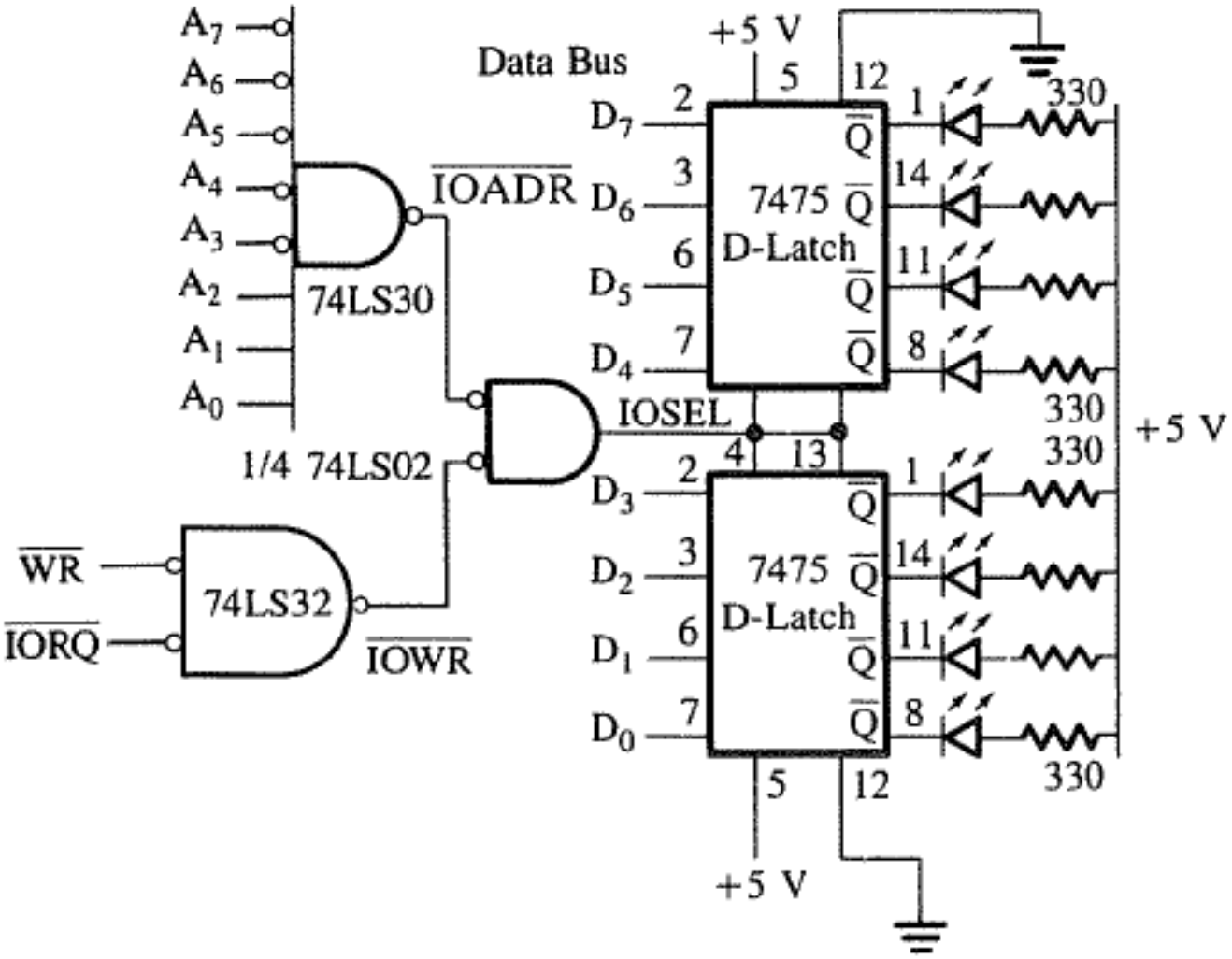
The first two machine cycles are similar as before

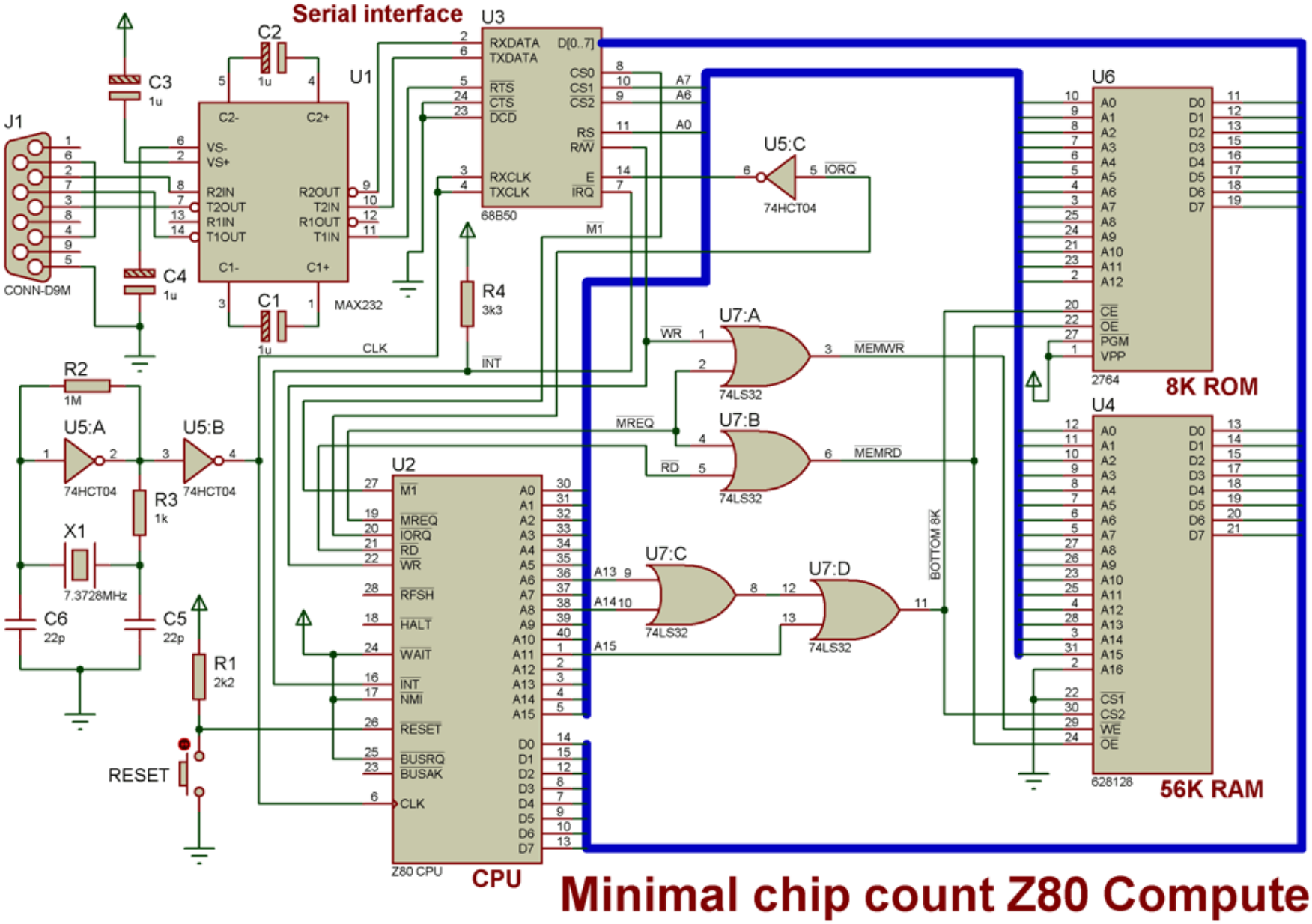
1. The Z80 places the port address 07_H on the low-order address bus and the contents of the accumulator on the data bus.
2. During T_2, it asserts the $\overline{\text{IORQ}}$ and $\overline{\text{WR}}$ control signals; the assertion of $\overline{\text{IORQ}}$ indicates that it is an I/O operation.
3. The Z80 automatically inserts a single Wait state T_W after T2 to allow sufficient response time for an I/O device; this Wait state is added regardless of the WAIT signal status.
4. During T_3, the control signals $\overline{\text{IORQ}}$ and $\overline{\text{WR}}$ become inactive.

Out Instruction



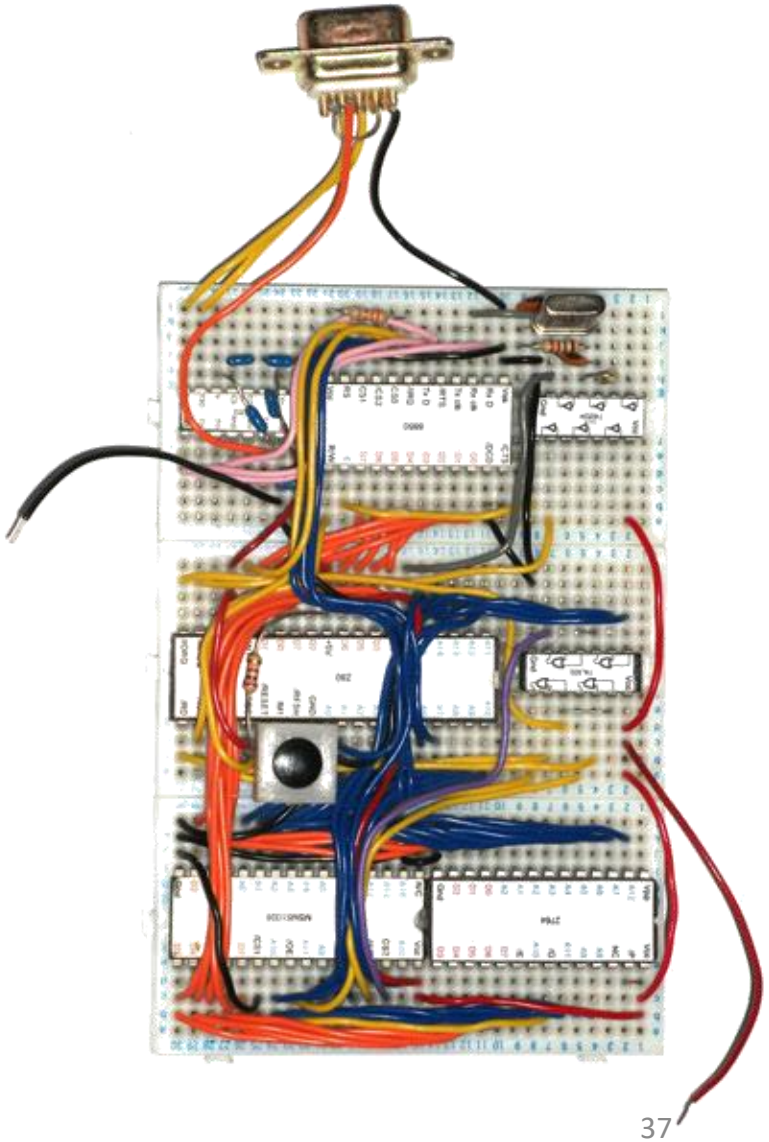
Interfacing LED Output Port



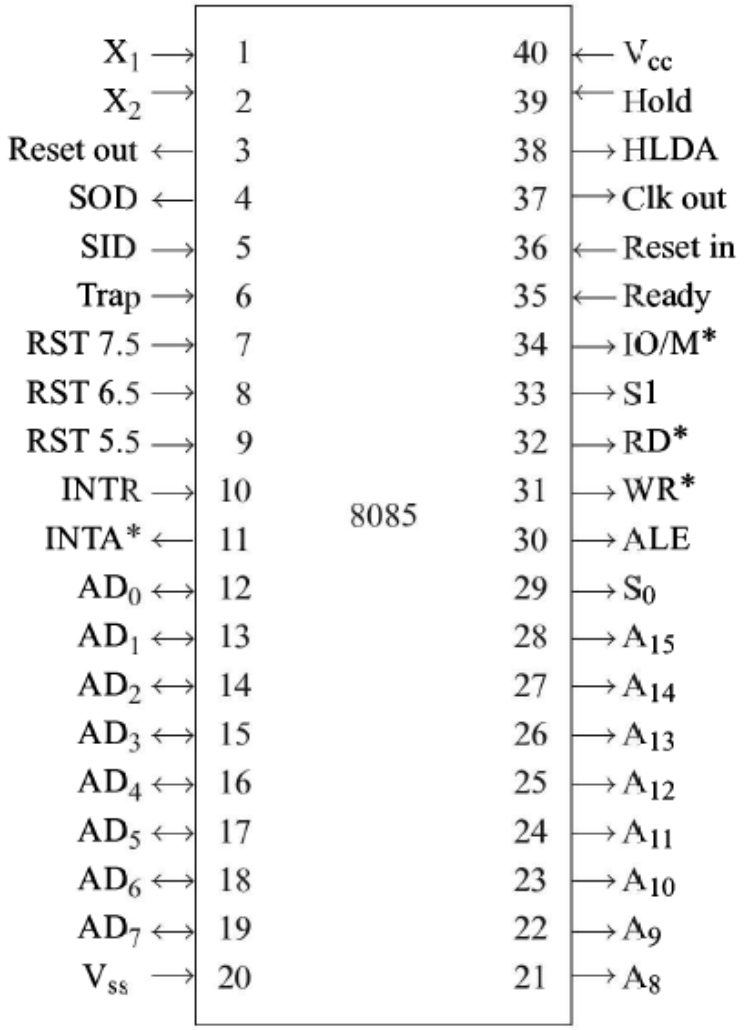

Serial interface
U3
U1
C2
C3
1u
J1
CONN-D9M
C4
1u
C1
1u
MAX232
C2-
C2+
VS-
VS+
R2IN
T2OUT
R1IN
T1OUT
C1-
C1+
R2OUT
T2IN
R1OUT
T1IN
RXDATA
TXDATA
RTS
CTS
DCD
RXCLK
TXCLK
D[0..7]
CS0
CS1
CS2
RS
R/W
E
IRQ
68B50
R4
3k3
INT
M1
CLK
U5:C
74HCT04
IORQ
U7:A
74LS32
WR
MEMWR
U7:B
MREQ
RD
MEMRD
U6
2764
8K ROM
U4
628128
56K RAM
BOTTOM 8K
R2
1M
U5:A
U5:B
74HCT04
74HCT04
R3
1k
X1
7.3728MHz
C6
22p
C5
22p
R1
2k2
RESET
U2
Z80 CPU
CPU
M1
MREQ
IORQ
RD
WR
RFSH
HALT
WAIT
INT
NMI
RESET
BUSRQ
BUSAK
CLK
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
D0
D1
D2
D3
D4
D5
D6
D7
U7:C
U7:D
74LS32
74LS32
A13
A14
A15
CE
OE
PGM
VPP
WE
CS1
CS2
Minimal chip count Z80 Computer

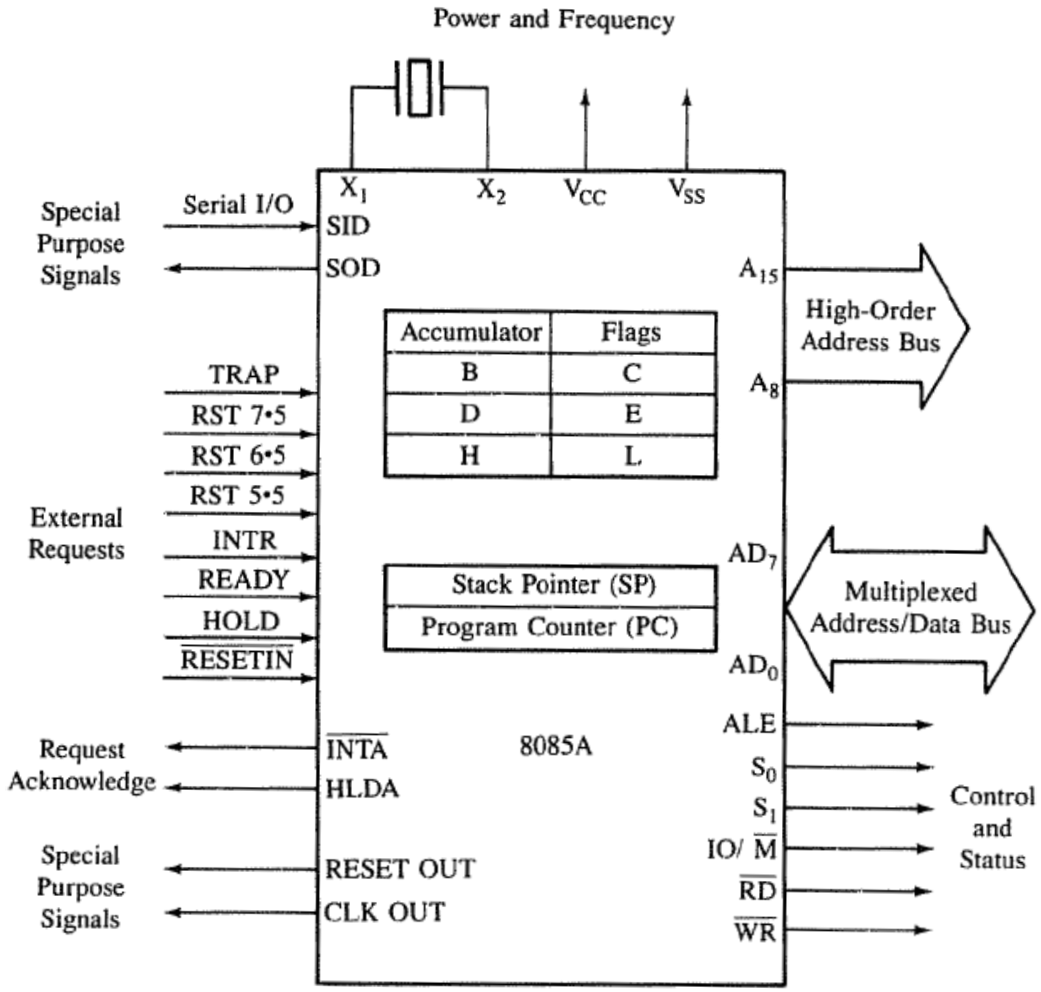
```
Z80 SBC By Grant Searle

Memory top?
Z80 BASIC Ver 4.7b
Copyright (C) 1978 by Microsoft
56958 Bytes free
Ok
10 FOR A=0 TO 6.2 STEP 0.2
20 PRINT TAB(40+SIN(A)*20);"*"
30 NEXT A
RUN
```

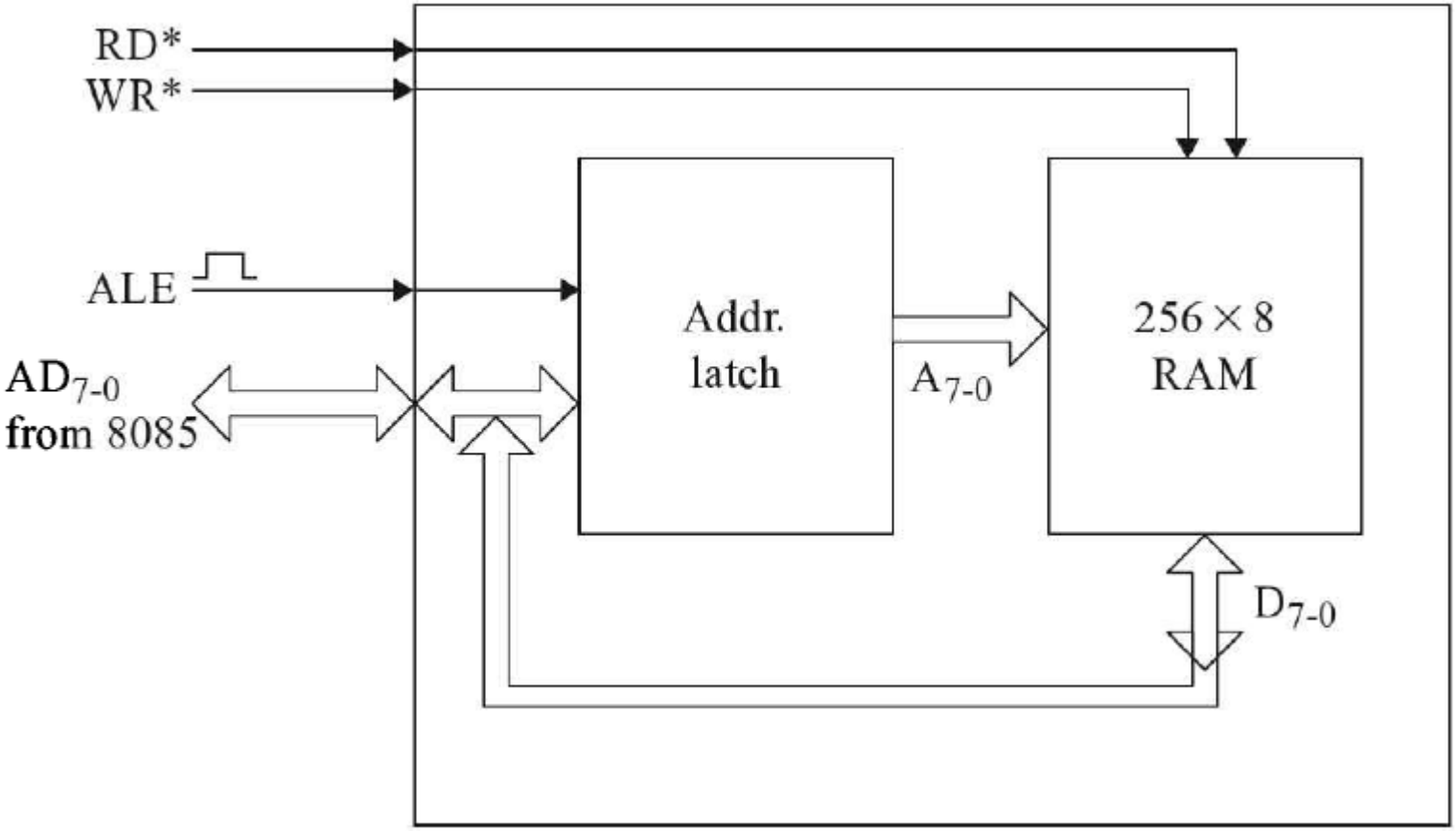


Intel 8085


X1 → 1
X2 → 2
Reset out ← 3
SOD ← 4
SID → 5
Trap → 6
RST 7.5 → 7
RST 6.5 → 8
RST 5.5 → 9
INTR → 10
INTA* ← 11
AD0 ↔ 12
AD1 ↔ 13
AD2 ↔ 14
AD3 ↔ 15
AD4 ↔ 16
AD5 ↔ 17
AD6 ↔ 18
AD7 ↔ 19
Vss → 20
8085
40 ← Vcc
39 ← Hold
38 → HLDA
37 → Clk out
36 ← Reset in
35 ← Ready
34 → IO/M*
33 → S1
32 → RD*
31 → WR*
30 → ALE
29 → S0
28 → A15
27 → A14
26 → A13
25 → A12
24 → A11
23 → A10
22 → A9
21 → A8

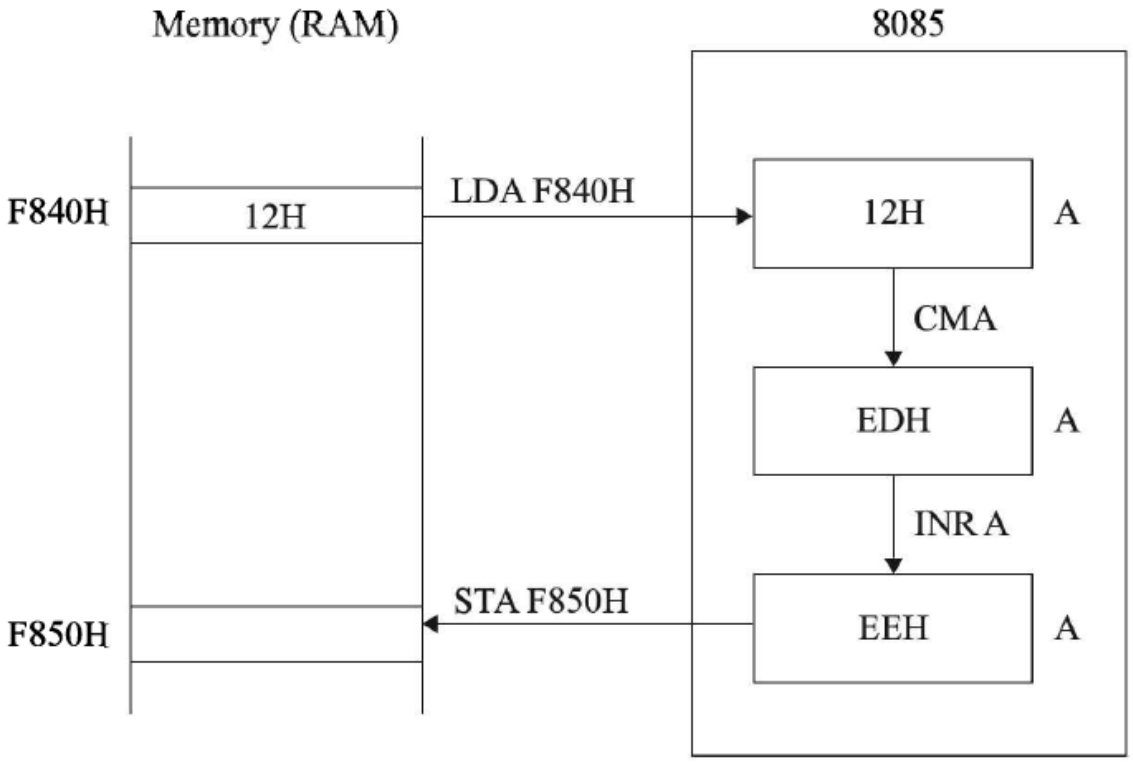

Power and Frequency
X1
X2
VCC
VSS
Special Purpose Signals
Serial I/O
SID
SOD
TRAP
RST 7•5
RST 6•5
RST 5•5
External Requests
INTR
READY
HOLD
RESETIN
Request Acknowledge
INTA
HLDA
Special Purpose Signals
RESET OUT
CLK OUT
Accumulator
Flags
B
C
D
E
H
L
Stack Pointer (SP)
Program Counter (PC)
8085A
A15
A8
High-Order Address Bus
AD7
AD0
Multiplexed Address/Data Bus
ALE
S0
S1
IO/ M
RD
WR
Control and Status

Address Latch



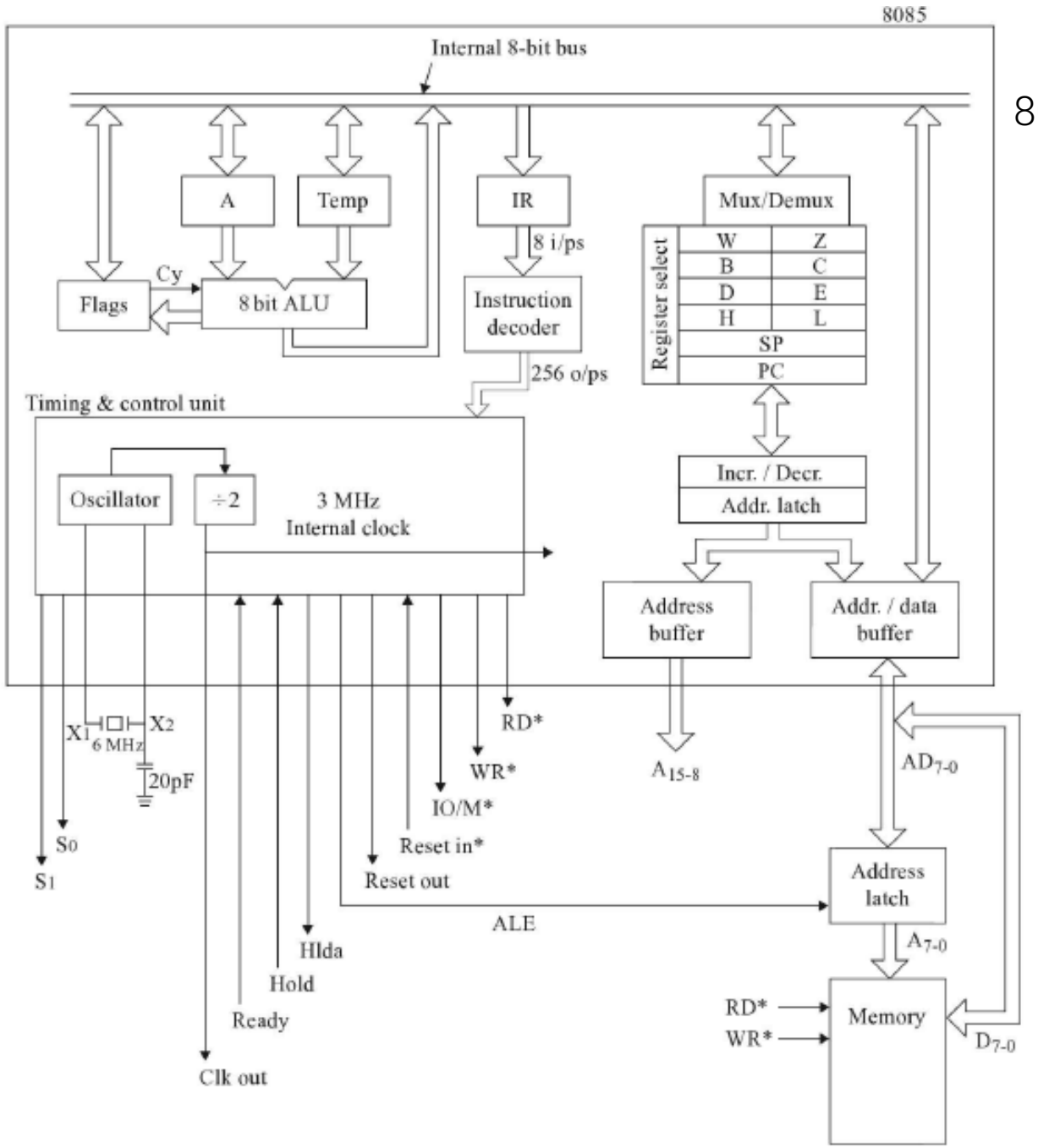
حل یک مسئله ساده با 8085

Let us say, we have some 8-bit number in a symbolic memory location X. We want to compute its 2's complement and store the result in symbolic memory location Y, overwriting the previous contents of memory location Y.




```
LDA F840H   ; (A) = 12H
CMA         ; (A) = EDH
INR A       ; (A) = EEH
STA F850H   ; (F850) = EEH
HLT
```

معماری داخلی 8085



سیگنال‌های وضعیت

Status signals IO/M, S1 and S0:* An instruction cycle (to be discussed later) requires one to five machine cycles, depending on the instruction. Each machine cycle performs a specific operation, like read from memory. A machine cycle needs a fixed number of clock cycles, the minimum being three clock cycles. The first clock cycle is termed as T1 (T for time period), the second clock cycle as T2, etc.

The type of machine cycle that 8085 is going to execute is indicated by the status signals IO/M*, S1, and S0. This is shown in the following table. These status signals are emitted by the 8085 during T1 of a machine cycle.

IO/M*	***S1***	***S0***	***Machine cycle***
0	0	1	Memory write (MW)
0	1	0	Memory read (MR)*
0	1	1	Opcode fetch (OF)
1	0	1	I/O write (IOW)
1	1	0	I/O read (IOR)
1	1	1	Interrupt acknowledge (INA)**

سیگنال‌های کنترل

Control signals RD, WR*, and INTA*:* During T1, the status signals and address are sent out. Only after the address has become stable, the control signals are emitted by the 8085 during T2 of a machine cycle. The values of the control signals for the various machine cycles are shown in the following table. INTA* signal will be discussed in detail in the chapter on 8085 interrupts.

Machine cycle	***RD****	***WR****	***INTA****
OF, MR, IOR	0	1	1
MW, IOW	1	0	1
INA	1	1	0
BI	1	1	1

Opcode fetch

Execution of 'MOV A, B' instruction.

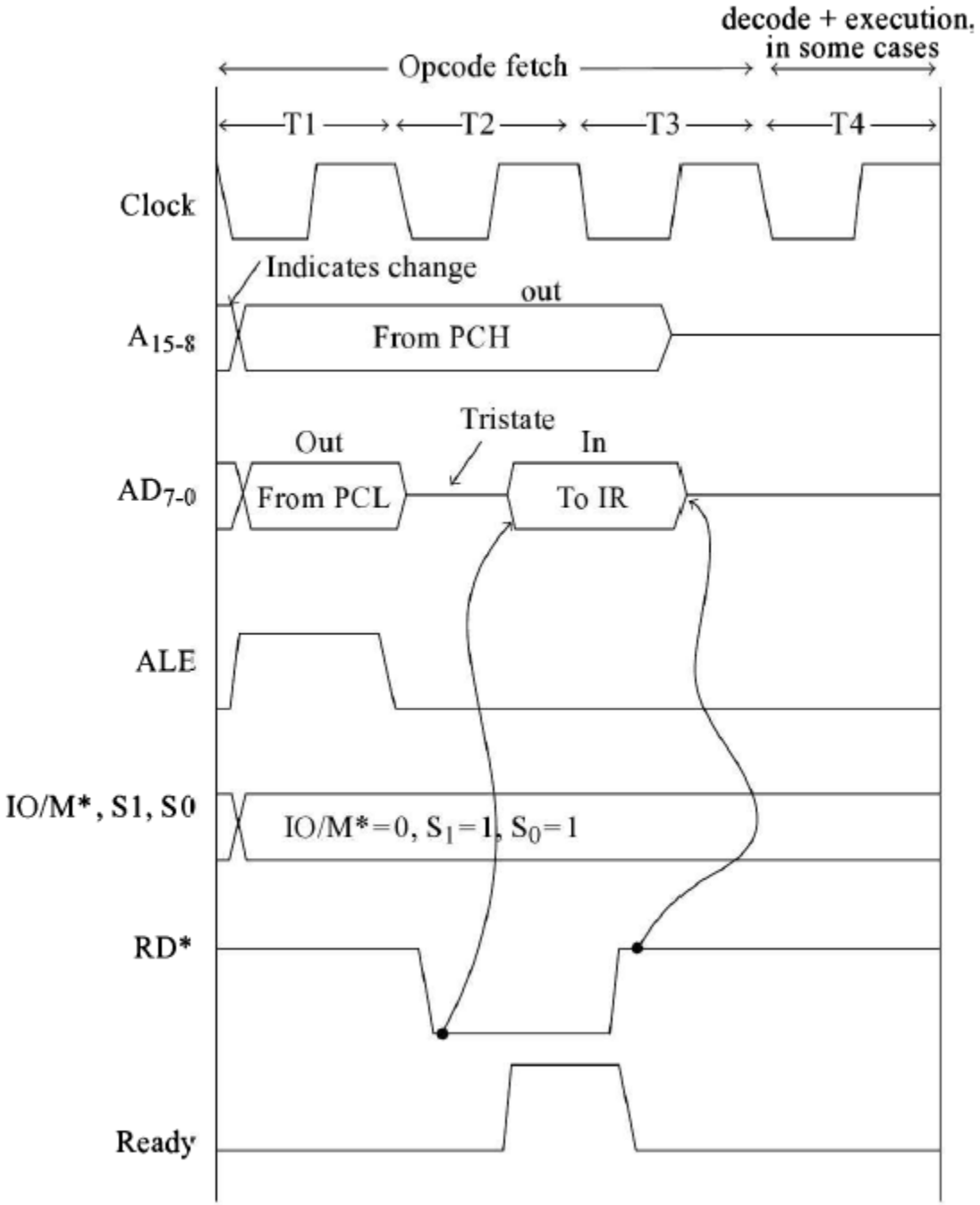
Address information is sent out on AD_{7-0} only during T1. In T2, the 8085 stops sending out the address and tristates AD_{7-0}. It expects that memory chip should now drive the AD_{7-0} pins with the opcode.

Logic 1 value is sent out on ALE pin during T1. During this clock period, LS byte of address is sent out on AD_{7-0}. The address latch does latching of this LS byte of address, when ALE is made 0 before the end of T1. Only during the beginning of T2, RD* will be made 0. This enables the memory to output the opcode after the access time of the memory chip.

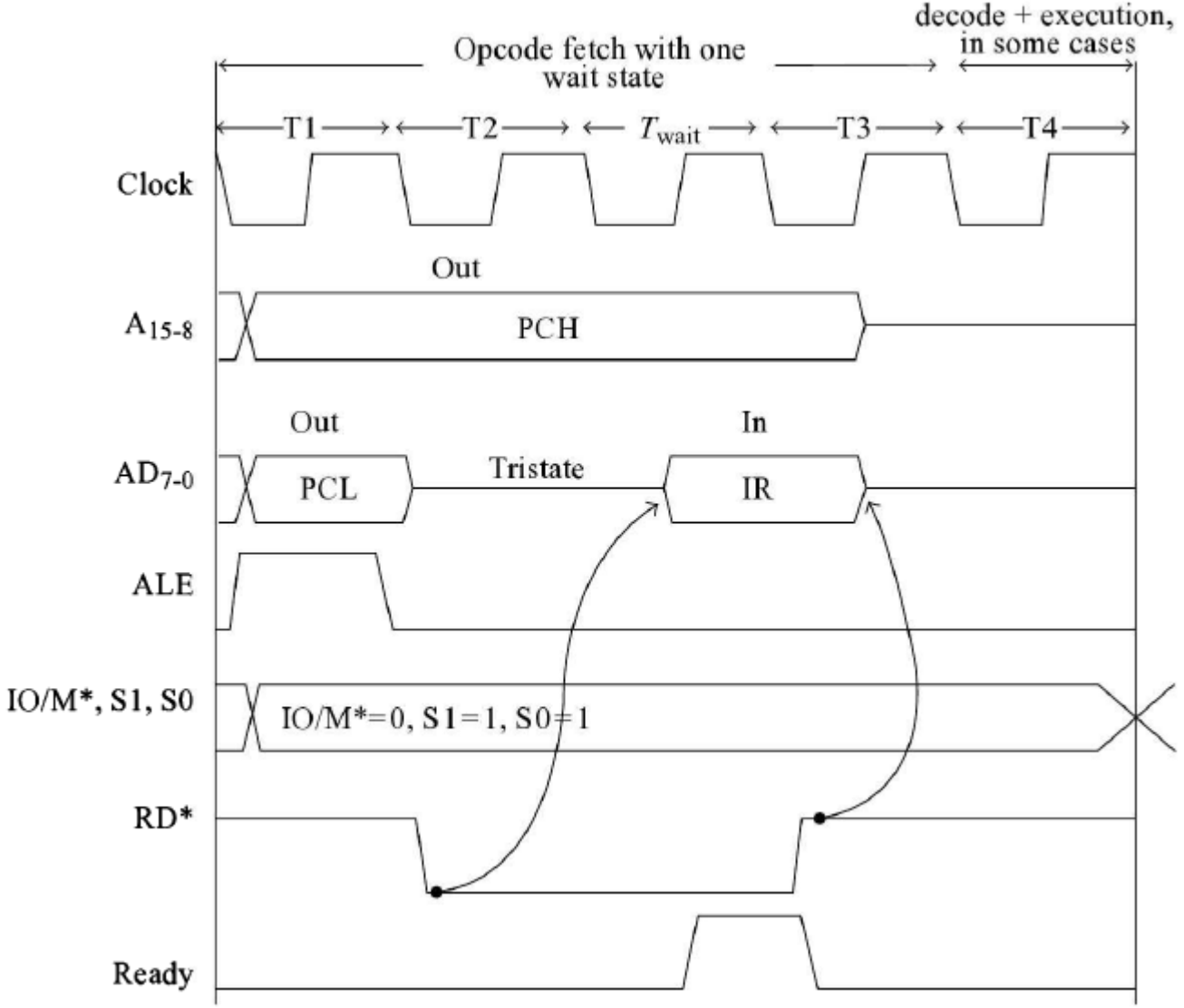
At the end of T2, the 8085 will sense Ready input. If it is 0, the 8085 understands that memory is not yet ready to supply the opcode. In such a case, the next clock cycle is not termed as T3. It will be termed as T_{wait}. At the end of T_{wait} Ready input will be sensed once again. If it is still logic 0, it enters another T_{wait} state once again. The 8085 goes to T3 state only when Ready signal is at logic 1 when sensed by the 8085 at the end of T2 or T_{wait}.

Thus in three clock cycles (if there are no wait states), the opcode is received in IR register. The decoding is done in T4 state. The control unit understands that the opcode corresponds to the 1-byte instruction whose mnemonic is 'MOV A, B'. Then it generates control signals such that B register contents are moved to A the register, via the multiplexer and the internal bus. This decoding and execution is completed in T4.

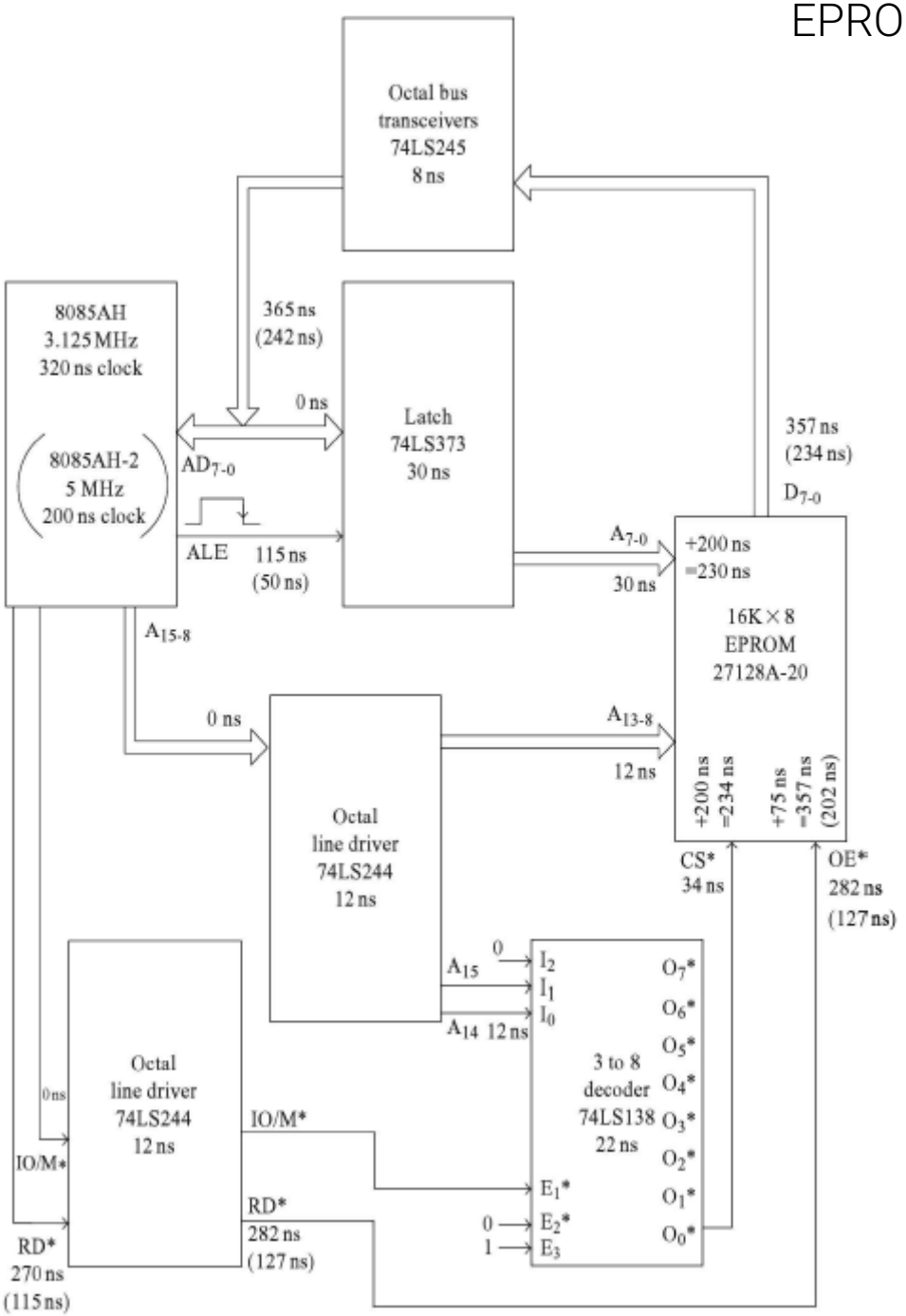
زمانبندی بارگذاری یک کد دستور در 8085



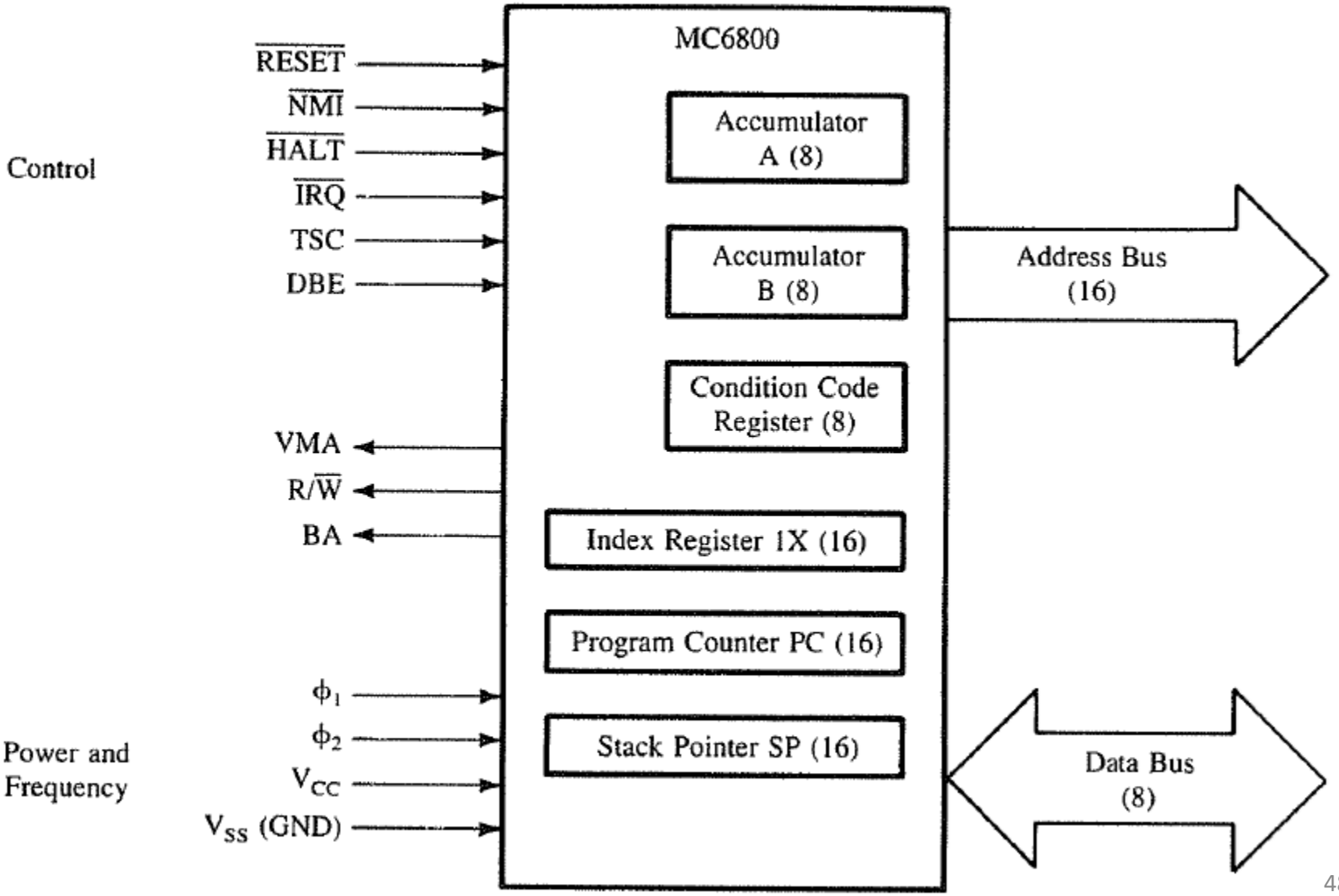
زمانبندی بارگذاری یک کد دستور در 8085 در حالتی که داده از سوی حافظه با تاخیر آماده شود!



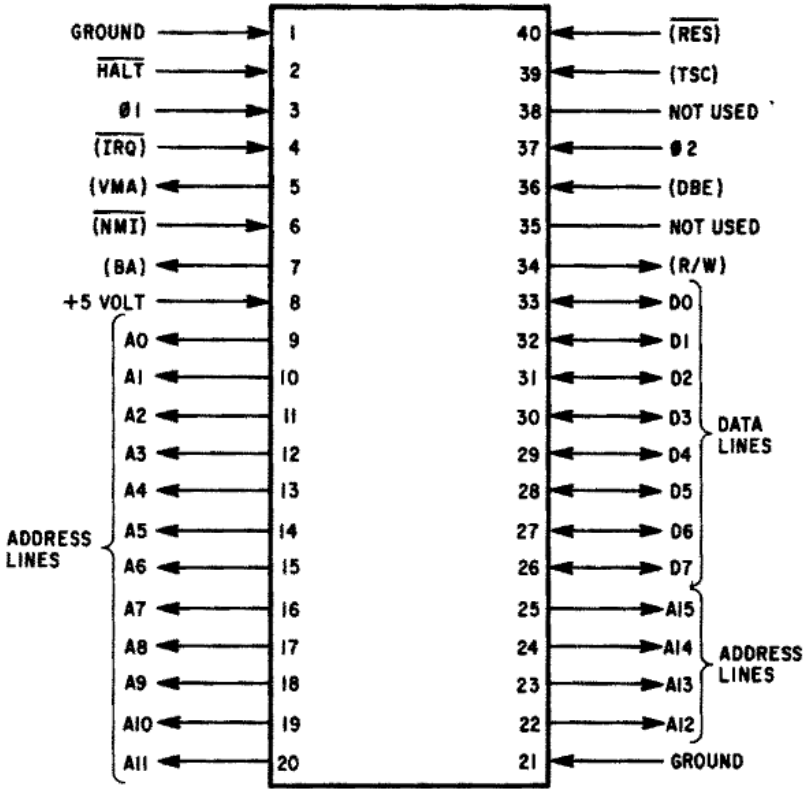
تاخیرهای دستیابی به EPROM مرتبط با یک برد 8085



Motorola MC6800

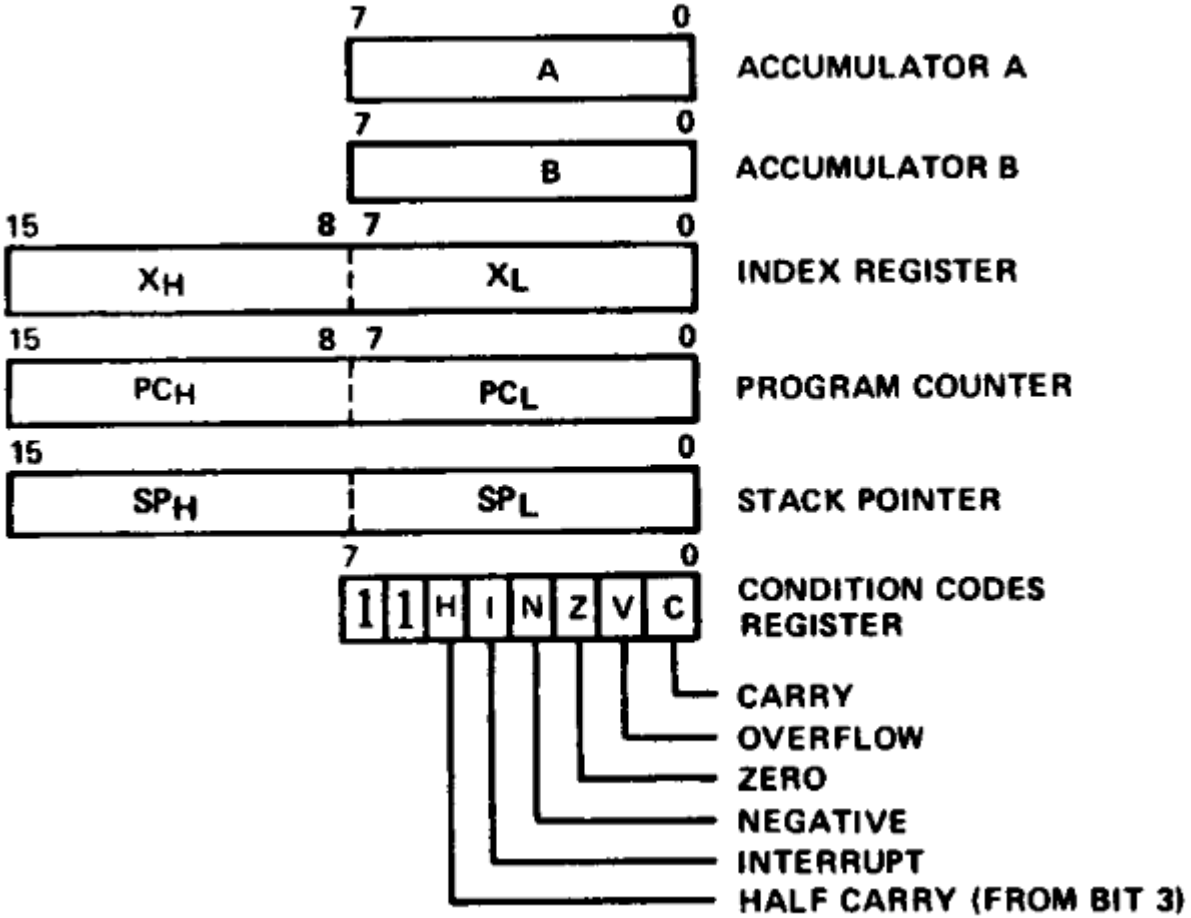


Motorola MC6800

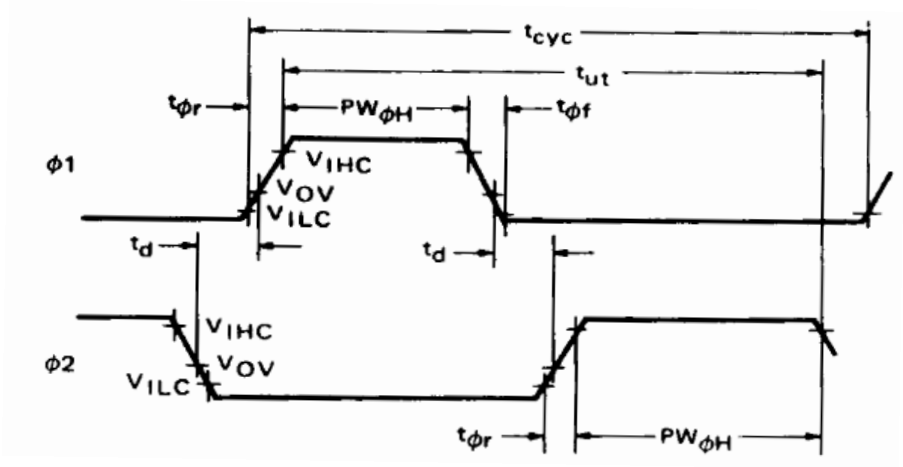


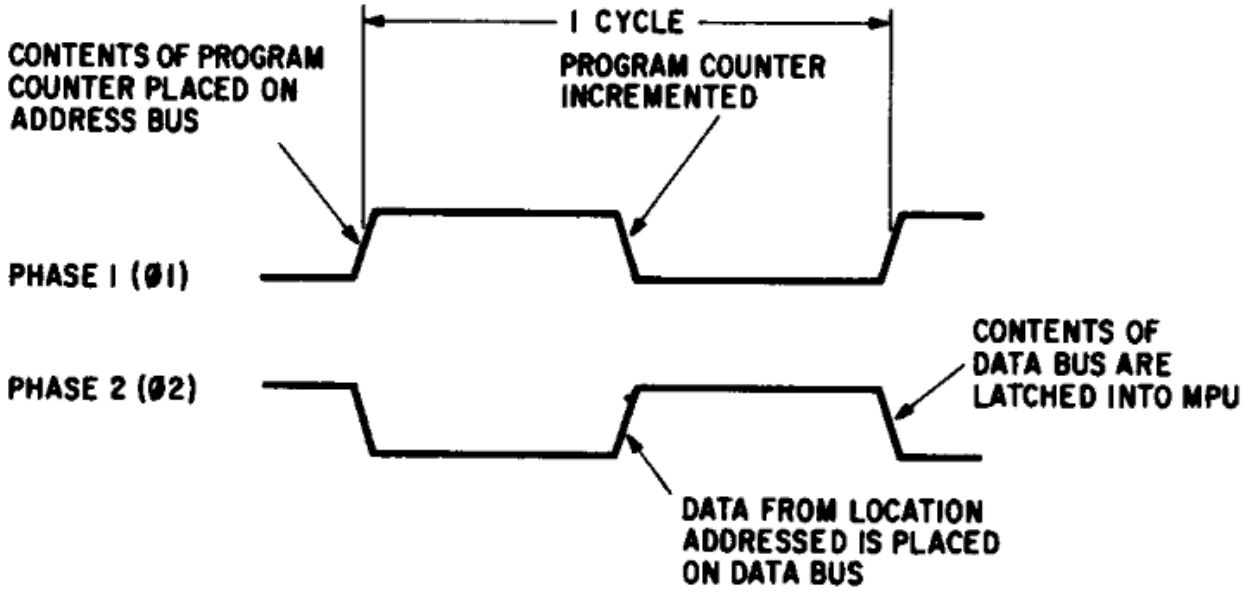
Pin No.	*Signal Description*	*Signal Name*	*Signal Type*	*Three State*
1	Ground	GND	Input	No
2	Halt	$\overline{\text{HALT}}$	Input	No
3	Phase 1	$\phi 1$	Input	No
4	Interrupt Request	$\overline{\text{IRQ}}$	Input	No
5	Valid Memory Address	VMA	Output	No
6	Nonmaskable Interrupt	$\overline{\text{NMI}}$	Input	No
7	Bus Available	BA	Output	No
8	Power	+5	Input	No
9–20, 22–25	Address Lines	A0–A15	Output	Yes
21	Ground	GND	Input	No
26–33	Data Lines	D0-D7	Input/Output	Yes
34	Read/Write	R/W	Output	Yes
35,38	Not Used			
36	Data Bus Enable	DBE	Input	No
37	Phase 2	$\phi 2$	Input	No
39	Three-State Control	TSC	Input	No
40	Reset	$\overline{\text{RES}}$	Input	No

Motorola MC6800



Motorola MC6800





Motorola MC6800

Address	*Contents*	*Description*
100	86	Load A Immediate
101	25	Data
102	D6	Load B Direct
103	35	Address
104	B7	STA A Extended
105	40	Address
106	02	

1. Addresses 100 and 101 (two cycles):

Cycle 1: (a) The contents of the P counter (100) are placed on the address bus.
(b) The P counter is then incremented by 1 (to 101).
(c) The contents of address 100 (86) are placed on the data bus, latched into the MPU instruction register (IR), and decoded as a load A Immediate instruction (R/W and VMA are "1").

Cycle 2: (a) The contents of the P counter (101) are placed on the address bus.
(b) The P counter is then incremented by 1 (to 102).
(c) The contents of address 101 (25) are placed on the data bus and then latched into the A accumulator (R/W and VMA are "1").

پایان

موفق و پیروز باشید