

Advance Ethernet

ARPAN CHAVDA

09BCE006

Department of Computer Science & Engineering
Institute of Technology
Nirma University
Ahmedabad 382 481
Gujarat, India.

Email: 09BCE006@nirmauni.ac.in

BHAVIN CHAUHAN

09BCE005

Department of Computer Science & Engineering
Institute of Technology
Nirma University
Ahmedabad 382 481
Gujarat, India.

Email: 09BCE005@nirmauni.ac.in

I. INTRODUCTION

- The Intel 8080 was the second 8-bit microprocessor designed and manufactured by Intel and was released in April 1974. It was an extended and enhanced variant of the earlier 8008 design, although without binary compatibility.
- The initial specified clock frequency limit was 2 MHz and with common instructions having execution times of 4,5,7,10 or 11 cycles this meant a few hundred thousand instructions per second.
- The 8080 has sometimes been labeled "the first truly usable microprocessor", despite the fact that earlier microprocessors were used for calculators and other applications.

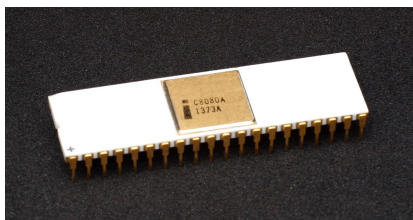


Fig. 1. Intel 8080 Chip

II. ARCHITECTURE

Memory

- Program, data and stack memories occupy the same memory space. The total addressable memory size is 64 KB.
- Program memory - program can be located anywhere in memory. Jump, branch and call instructions use 16-bit addresses, i.e. they can be used to jump/branch anywhere within 64 KB. All jump/branch instructions use absolute addressing.
- Data memory - the processor always uses 16-bit addresses so that data can be placed anywhere.
- Stack memory is limited only by the size of memory. Stack grows downward.

Registers

- Accumulator or A register is an 8-bit register used for arithmetic, logic, I/O and load/store operations.
- Flag is an 8-bit register containing 5 1-bit flags:
- General registers:
 - 8-bit B and 8-bit C registers can be used as one 16-bit BC register pair. When used as a pair the C register contains low-order byte. Some instructions may use BC register as a data pointer.
 - 8-bit D and 8-bit E registers can be used as one 16-bit DE register pair. When used as a pair the E register contains low-order byte. Some instructions may use DE register as a data pointer.
 - 8-bit H and 8-bit L registers can be used as one 16-bit HL register pair. When used as a pair the L register contains low-order byte. HL register usually contains a data pointer used to reference memory addresses.
- Stack pointer is a 16 bit register. This register is always incremented/decremented by 2.
- Program counter is a 16-bit register.

Input/Output Ports

- 256 Input ports and 256 Output Ports

Addressing modes

- Register-references the data in a register or in a register pair.
- Register indirect-instruction specifies register pair containing address, where the data is located.
- Direct.
- Immediate - 8 or 16-bit data.

Instruction Set: 8080 instruction set consists of the following instructions:

- Data moving instructions.
- Arithmetic - add, subtract, increment and decrement.
- Logic - AND, OR, XOR and rotate.
- Control transfer - conditional, unconditional, call subroutine, return from subroutine and restarts.
- Input/Output instructions.
- Other - setting/clearing flag bits, enabling/disabling interrupts, stack operations, etc.

Intel 8080 Architecture

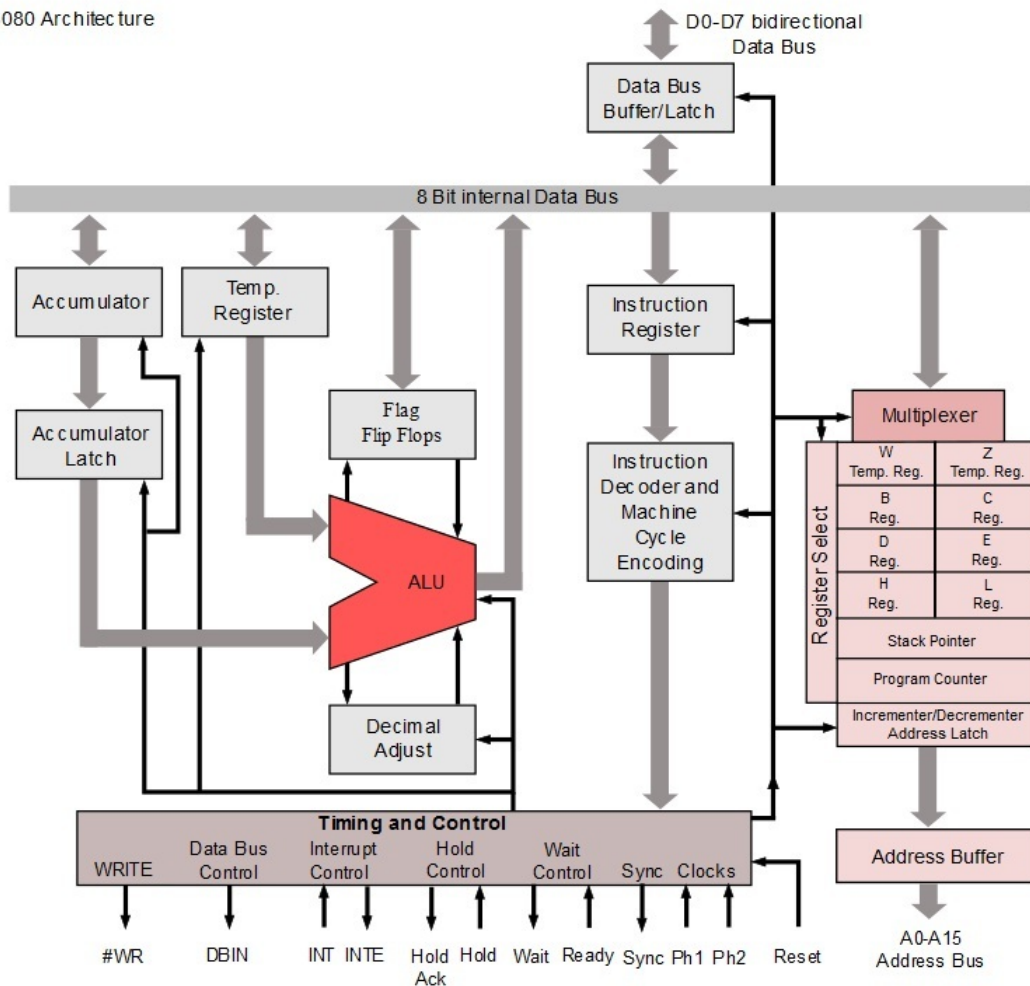


Fig. 2. Intel 8080 Architecture

Mnemonic	Instruction Code [1]								Operations Description	Clock Cycles [2]
	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀		
MOVE, LOAD, AND STORE										
MOV r ₁ , r ₂	0	1	D	D	D	S	S	S	Move register to register	5
MOV M, r	0	1	1	1	0	S	S	S	Move register to memory	7
MOV r, M	0	1	D	D	D	1	1	0	Move memory to register	7
MVI r	0	0	D	D	D	1	1	0	Move immediate register	7
MVI M	0	0	1	1	0	1	1	0	Move immediate memory	10
LXI B	0	0	0	0	0	0	0	1	Load immediate register Pair B & C	10
LXI D	0	0	0	1	0	0	0	1	Load immediate register Pair D & E	10
LXI H	0	0	1	0	0	0	0	1	Load immediate register Pair H & L	10
STAX B	0	0	0	0	0	0	1	0	Store A indirect	7
STAX D	0	0	0	1	0	0	1	0	Store A indirect	7
LDAX B	0	0	0	0	1	0	1	0	Load A indirect	7
LDAX D	0	0	0	1	1	0	1	0	Load A indirect	7
STA	0	0	1	1	0	0	1	0	Store A direct	13
LDA	0	0	1	1	1	0	1	0	Load A direct	13
SHLD	0	0	1	0	0	0	1	0	Store H & L direct	16
LHLD	0	0	1	0	1	0	1	0	Load H & L direct	16
XCHG	1	1	1	0	1	0	1	1	Exchange D & E, H & L Registers	4
STACK OPS										
PUSH B	1	1	0	0	0	1	0	1	Push register Pair B & C on stack	11
PUSH D	1	1	0	1	0	1	0	1	Push register Pair D & E on stack	11
PUSH H	1	1	1	0	0	1	0	1	Push register Pair H & L on stack	11
PUSH PSW	1	1	1	1	0	1	0	1	Push A and Flags on stack	11
POP B	1	1	0	0	0	0	0	1	Pop register Pair B & C off stack	10
POP D	1	1	0	1	0	0	0	1	Pop register Pair D & E off stack	10
POP H	1	1	1	0	0	0	0	1	Pop register Pair H & L off stack	10
POP PSW	1	1	1	1	0	0	0	1	Pop A and Flags off stack	10
XTHL	1	1	1	0	0	0	1	1	Exchange top of stack, H & L	18
SPHL	1	1	1	1	1	0	0	1	H & L to stack pointer	5
LXI SP	0	0	1	1	0	0	0	1	Load immediate stack pointer	10
INX SP	0	0	1	1	0	0	1	1	Increment stack pointer	5
DCX SP	0	0	1	1	1	0	1	1	Decrement stack pointer	5
JUMP										
JMP	1	1	0	0	0	0	1	1	Jump unconditional	10
JC	1	1	0	1	1	0	1	0	Jump on carry	10
JNC	1	1	0	1	0	0	1	0	Jump on no carry	10
JZ	1	1	0	0	1	0	1	0	Jump on zero	10
JNZ	1	1	0	0	0	0	1	0	Jump on no zero	10
JP	1	1	1	1	0	0	1	0	Jump on positive	10
JM	1	1	1	1	1	0	1	0	Jump on minus	10
JPE	1	1	1	0	1	0	1	0	Jump on parity even	10

Mnemonic	Instruction Code [1]								Operations Description	Clock Cycles [2]
	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀		
JPO	1	1	1	0	0	0	1	0	Jump on parity odd	10
PCHL	1	1	1	0	1	0	0	1	H & L to program counter	5
CALL										
CALL	1	1	0	0	1	1	0	1	Call unconditional	17
CC	1	1	0	1	1	1	0	0	Call on carry	11/17
CNC	1	1	0	1	0	1	0	0	Call on no carry	11/17
CZ	1	1	0	0	1	1	0	0	Call on zero	11/17
CNZ	1	1	0	0	0	1	0	0	Call on no zero	11/17
CP	1	1	1	1	0	1	0	0	Call on positive	11/17
CM	1	1	1	1	1	1	0	0	Call on minus	11/17
CPE	1	1	1	0	1	1	0	0	Call on parity even	11/17
CPO	1	1	1	0	0	1	0	0	Call on parity odd	11/17
RETURN										
RET	1	1	0	0	1	0	0	1	Return	10
RC	1	1	0	1	1	0	0	0	Return on carry	5/11
RNC	1	1	0	1	0	0	0	0	Return on no carry	5/11
RZ	1	1	0	0	1	0	0	0	Return on zero	5/11
RNZ	1	1	0	0	0	0	0	0	Return on no zero	5/11
RP	1	1	1	1	0	0	0	0	Return on positive	5/11
RM	1	1	1	1	1	0	0	0	Return on minus	5/11
RPE	1	1	1	0	1	0	0	0	Return on parity even	5/11
RPO	1	1	1	0	0	0	0	0	Return on parity odd	5/11
RESTART										
RST	1	1	A	A	A	1	1	1	Restart	11
INCREMENT AND DECREMENT										
INR r	0	0	D	D	D	1	0	0	Increment register	5
DCR r	0	0	D	D	D	1	0	1	Decrement register	5
INR M	0	0	1	1	0	1	0	0	Increment memory	10
DCR M	0	0	1	1	0	1	0	1	Decrement memory	10
INX B	0	0	0	0	0	0	1	1	Increment B & C registers	5
INX D	0	0	0	1	0	0	1	1	Increment D & E registers	5
INX H	0	0	1	0	0	0	1	1	Increment H & L registers	5
DCX B	0	0	0	0	1	0	1	1	Decrement B & C	5
DCX D	0	0	0	1	1	0	1	1	Decrement D & E	5
DCX H	0	0	1	0	1	0	1	1	Decrement H & L	5
ADD										
ADD r	1	0	0	0	0	S	S	S	Add register to A	4
ADC r	1	0	0	0	1	S	S	S	Add register to A with carry	4
ADD M	1	0	0	0	0	1	1	0	Add memory to A	7
ADC M	1	0	0	0	1	1	1	0	Add memory to A with carry	7
ADI	1	1	0	0	0	1	1	0	Add immediate to A	7
ACI	1	1	0	0	1	1	1	0	Add immediate to A with carry	7
DAD B	0	0	0	0	1	0	0	1	Add B & C to H & L	10
DAD D	0	0	0	1	1	0	0	1	Add D & E to H & L	10
DAD H	0	0	1	0	1	0	0	1	Add H & L to H & L	10
DAD SP	0	0	1	1	1	0	0	1	Add stack pointer to H & L	10

Fig. 3. Instructions-1

A.

Mnemonic	Instruction Code [1]								Operations Description	Clock Cycles [2]
	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀		
SUBTRACT										
SUB r	1	0	0	1	0	S	S	S	Subtract register from A	4
SBB r	1	0	0	1	1	S	S	S	Subtract register from A with borrow	4
SUB M	1	0	0	1	0	1	1	0	Subtract memory from A	7
SBB M	1	0	0	1	1	1	1	0	Subtract memory from A with borrow	7
SUI	1	1	0	1	0	1	1	0	Subtract immediate from A	7
SBI	1	1	0	1	1	1	1	0	Subtract immediate from A with borrow	7
LOGICAL										
ANA r	1	0	1	0	0	S	S	S	And register with A	4
XRA r	1	0	1	0	1	S	S	S	Exclusive Or register with A	4
ORA r	1	0	1	1	0	S	S	S	Or register with A	4
CMP r	1	0	1	1	1	S	S	S	Compare register with A	4
ANA M	1	0	1	0	0	1	1	0	And memory with A	7
XRA M	1	0	1	0	1	1	1	0	Exclusive Or memory with A	7
ORA M	1	0	1	1	0	1	1	0	Or memory with A	7
CMP M	1	0	1	1	1	1	1	0	Compare memory with A	7
ANI	1	1	1	0	0	1	1	0	And immediate with A	7
XRI	1	1	1	0	1	1	1	0	Exclusive Or immediate with A	7
ORI	1	1	1	1	0	1	1	0	Or immediate with A	7
CPI	1	1	1	1	1	1	1	0	Compare immediate with A	7

Mnemonic	Instruction Code [1]								Operations Description	Clock Cycles [2]
	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀		
ROTATE										
RLC	0	0	0	0	0	1	1	1	Rotate A left	4
RRC	0	0	0	0	1	1	1	1	Rotate A right	4
RAL	0	0	0	1	0	1	1	1	Rotate A left through carry	4
RAR	0	0	0	1	1	1	1	1	Rotate A right through carry	4
SPECIALS										
CMA	0	0	1	0	1	1	1	1	Complement A	4
STC	0	0	1	1	0	1	1	1	Set carry	4
CMC	0	0	1	1	1	1	1	1	Complement carry	4
DAA	0	0	1	0	0	1	1	1	Decimal adjust A	4
INPUT/OUTPUT										
IN	1	1	0	1	1	0	1	1	Input	10
OUT	1	1	0	1	0	0	1	1	Output	10
CONTROL										
EI	1	1	1	1	1	0	1	1	Enable interrupts	4
DI	1	1	1	1	0	0	1	1	Disable interrupt	4
NOP	0	0	0	0	0	0	0	0	No-operation	4
HLT	0	1	1	1	0	1	1	0	Halt	7

NOTES:

1. DDD or SSS: B=000, C=001, D=010, E=011, H=100, L=101, Memory=110, A=111.

2. Two possible cycle times (6/12) indicate instruction cycles dependent on condition flags.

*All mnemonics copyright ©Intel Corporation 1977

Fig. 4. Instructions-2

THE PROCESSOR STATUS REGISTER (FLAGS)



Fig. 5. Flag Register

- 1) Sign - set if the most significant bit of the result is set.
- 2) Zero - set if the result is zero.
- 3) Auxiliary carry - set if there was a carry out from bit 3 to bit 4 of the result.
- 4) Parity - set if the parity (the number of set bits in the result) is even.
- 5) Carry - set if there was a carry during addition, or borrow during subtraction/comparison.

III. PIN DIAGRAM

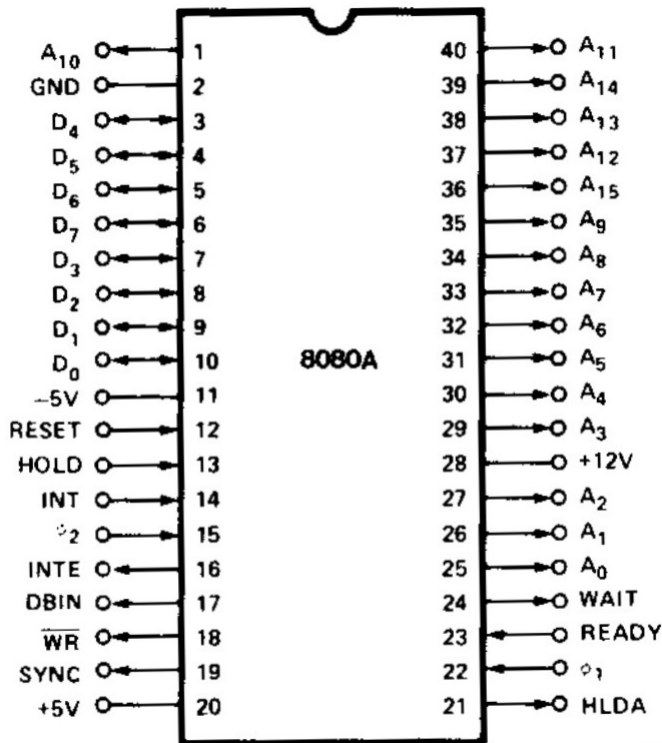


Fig. 6. Pin Diagram of 8080A

Symbol	Type	Name and Function
A ₁₅ -A ₀	O	Address Bus: The address bus provides the address to memory (up to 64K 8-bit words) or denotes the I/O device number for up to 256 input and 256 output devices. A ₀ is the least significant address bit.
D ₇ -D ₀	I/O	Data Bus: The data bus provides bi-directional communication between the CPU, memory, and I/O devices for instructions and data transfers. Also, during the first clock cycle of each machine cycle, the 8080A outputs a status word on the data bus that describes the current machine cycle. D ₀ is the least significant bit.
SYNC	O	Synchronizing Signal: The SYNC pin provides a signal to indicate the beginning of each machine cycle.
DBIN	O	Data Bus In: The DBIN signal indicates to external circuits that the data bus is in the input mode. This signal should be used to enable the gating of data onto the 8080A data bus from memory or I/O.
READY	I	Ready: The READY signal indicates to the 8080A that valid memory or input data is available on the 8080A data bus. This signal is used to synchronize the CPU with slower memory or I/O devices. If after sending an address out the 8080A does not receive a READY input, the 8080A will enter a WAIT state for as long as the READY line is low. READY can also be used to single step the CPU.
WAIT	O	Wait: The WAIT signal acknowledges that the CPU is in a WAIT state.
WR	O	Write: The WR signal is used for memory WRITE or I/O output control. The data on the data bus is stable while the WR signal is active low (WR = 0).
HOLD	I	Hold: The HOLD signal requests the CPU to enter the HOLD state. The HOLD state allows an external device to gain control of the 8080A address and data bus as soon as the 8080A has completed its use of these busses for the current machine cycle. It is recognized under the following conditions: <ul style="list-style-type: none"> the CPU is in the HALT state. the CPU is in the T2 or TW state and the READY signal is active. As a result of entering the HOLD state the CPU ADDRESS BUS (A₁₅-A₀) and DATA BUS (D₇-D₀) will be in their high impedance state. The CPU acknowledges its state with the HOLD ACKNOWLEDGE (HLDA) pin.
HLDA	O	Hold Acknowledge: The HLDA signal appears in response to the HOLD signal and indicates that the data and address bus will go to the high impedance state. The HLDA signal begins at: <ul style="list-style-type: none"> T3 for READ memory or input. The Clock Period following T3 for WRITE memory or OUTPUT operation. In either case, the HLDA signal appears after the rising edge of phi ₂ .
INTE	O	Interrupt Enable: Indicates the content of the internal interrupt enable flip/flop. This flip/flop may be set or reset by the Enable and Disable Interrupt instructions and inhibits interrupts from being accepted by the CPU when it is reset. It is automatically reset (disabling further interrupts) at time T1 of the instruction fetch cycle (M1) when an interrupt is accepted and is also reset by the RESET signal.
INT	I	Interrupt Request: The CPU recognizes an interrupt request on this line at the end of the current instruction or while halted. If the CPU is in the HOLD state or if the Interrupt Enable flip/flop is reset it will not honor the request.
RESET ¹	I	Reset: While the RESET signal is activated, the content of the program counter is cleared. After RESET, the program will start at location 0 in memory. The INTE and HLDA flip/flops are also reset. Note that the flags, accumulator, stack pointer, and registers are not cleared.
V _{SS}		Ground: Reference.
V _{DD}		Power: +12 ± 5% Volts.
V _{CC}		Power: +5 ± 5% Volts.
V _{BB}		Power: -5 ± 5% Volts.
phi ₁ , phi ₂		Clock Phases: 2 externally supplied clock phases. (non TTL compatible)

Fig. 7. Pin Functions of 8080A

IV. THE INDUSTRIAL IMPACT

- The 8080 was used in many early microcomputers, such as the MITS Altair 8800 Computer, Processor Technology SOL-20 Terminal Computer and IMSAI 8080 Microcomputer, forming the basis for machines running the CP/M operating system.
- The 8080 was actually designed for just about any application except a complete computer system. Hewlett Packard developed the HP 2640 series of smart terminals around the 8080. The HP 2647 was a terminal which ran BASIC on the 8080.
- In addition, several early arcade video games were built around the 8080 microprocessor. Space Invaders was perhaps the most popular such title.

V. REFERENCE

- Wikipedia, The free Encyclopedia
- MICROCOMPUTERS AND MICROPROCESSORS(2nd EDITION) BY JOHN UFFENBECK
- Intel 8080 Microprocessor Datasheet