Design of basic microprocessor architectural Concepts

Chapter 1

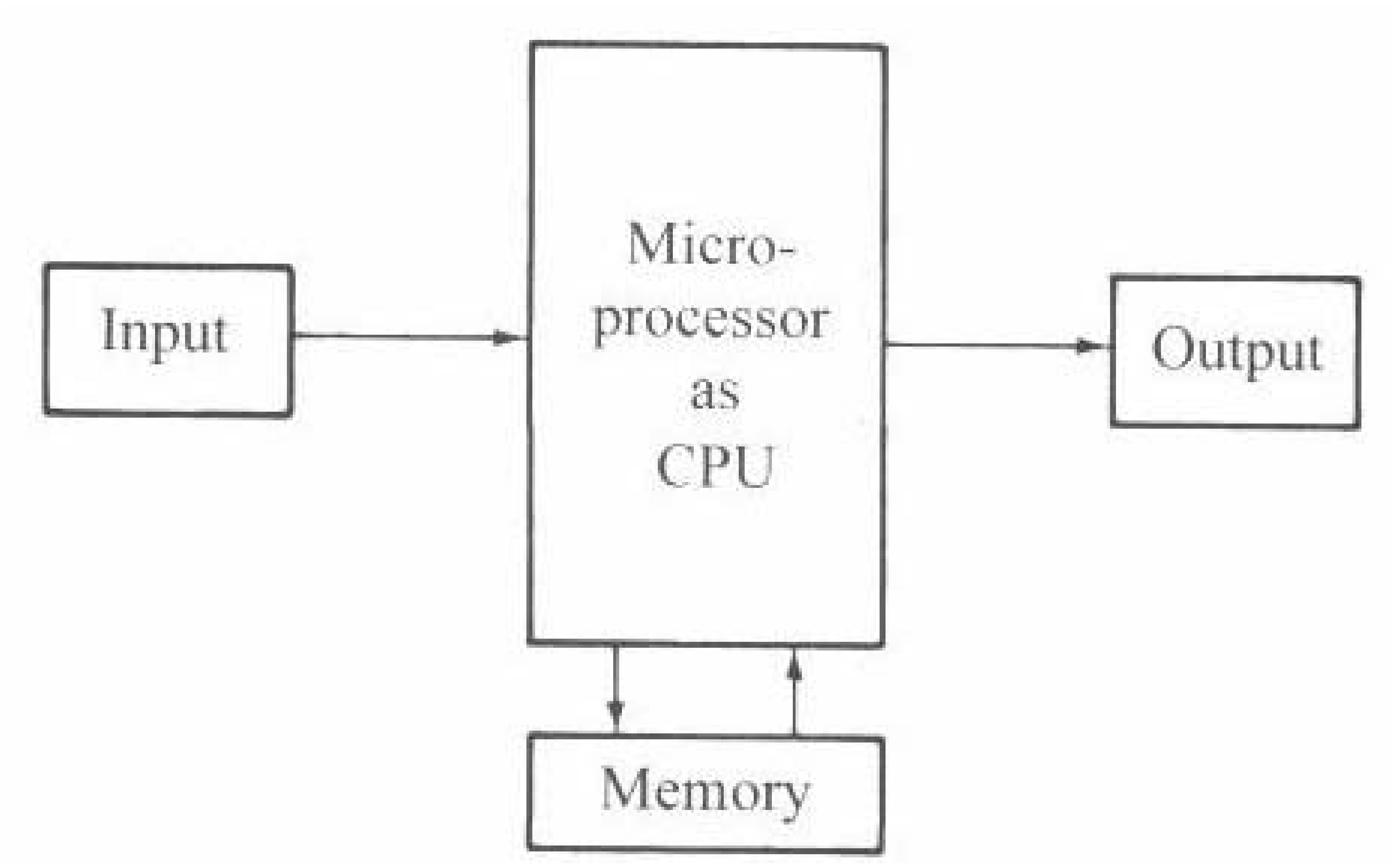
Design of basic microprocessor architectural Concepts

- Microprocessor architecture, word Lengths, addressable memory
- Microprocessor’s speed architectural characteristics
- Registers, instruction, memory addressing architecture, ALU,GPR’s Control logic & internal data bus.

MICROPROCESSOR

- Is a programmable integrated device that has computing and decision- making capability similar to that of central processing unit of a computer.
- It is a multipurpose, programmable, clock driven, register based electronic device that reads binary information from a storage device called memory, accepts binary data as input and processes data according to those instructions and provides results as output.

A Programmable machine



The 8-Bit Microprocessor

- 8085 was a newer version of 8080, again introduced by Intel in 1977.
- The 8085 addresses the same amount of memory, executed about the same number of instructions.
- However it was a little faster than the 8080 in that sense that an addition instruction which took about 2 μs on an 8080 took only 1.3 μs on an 8085.
- The main advantage of an 8085 is its built-in clock and system controller which was external components on an 8080 based system.

Organization of a Microprocessor-Based System

The functions of various components:

- **The microprocessor**
  - reads instructions from memory.
  - communicates with all peripherals (memory and 1/Os) using the system bus.
  - controls the timing of information flow.
  - performs the computing tasks specified in a program.
- **The memory**
  - stores binary information, called instructions and data.
  - provides the instructions and data to the microprocessor on request.
  - stores results and data for the microprocessor.
- **The input device**
  - enters data and instructions under the control of a program such as program.
- **The output device**
  - accepts data from the microprocessor as specified in a program.
- **The bus**
  - carries bits between the microprocessor and memory and I/Os.

Introduction to Microprocessor

- The microprocessor communicates and operates in the binary numbers 0 and 1, called bits.
- Each microprocessor has a fixed set of instructions in the form of binary patterns called a machine language.
- It is difficult for humans to communicate in the language of 0s and 1s.
- Therefore, the binary instructions are given abbreviated names, called mnenomics, which form the assembly language for a given microprocessor.

Applications

- In reprogrammable systems, such as microcomputers, the microprocessor is used for computing and data processing. These systems include:
  - general-purpose microprocessors capable of handling large data, mass storage devices (such as disks and CD-ROMs), and peripherals such as printers
  - a personal computer (PC) is a typical illustration.

Applications

- In embedded systems, the microprocessor is a part of a final product and is not available for reprogramming to the end user. Example:
  - copying machine
  - washing machine.
  - Air-conditioner etc.

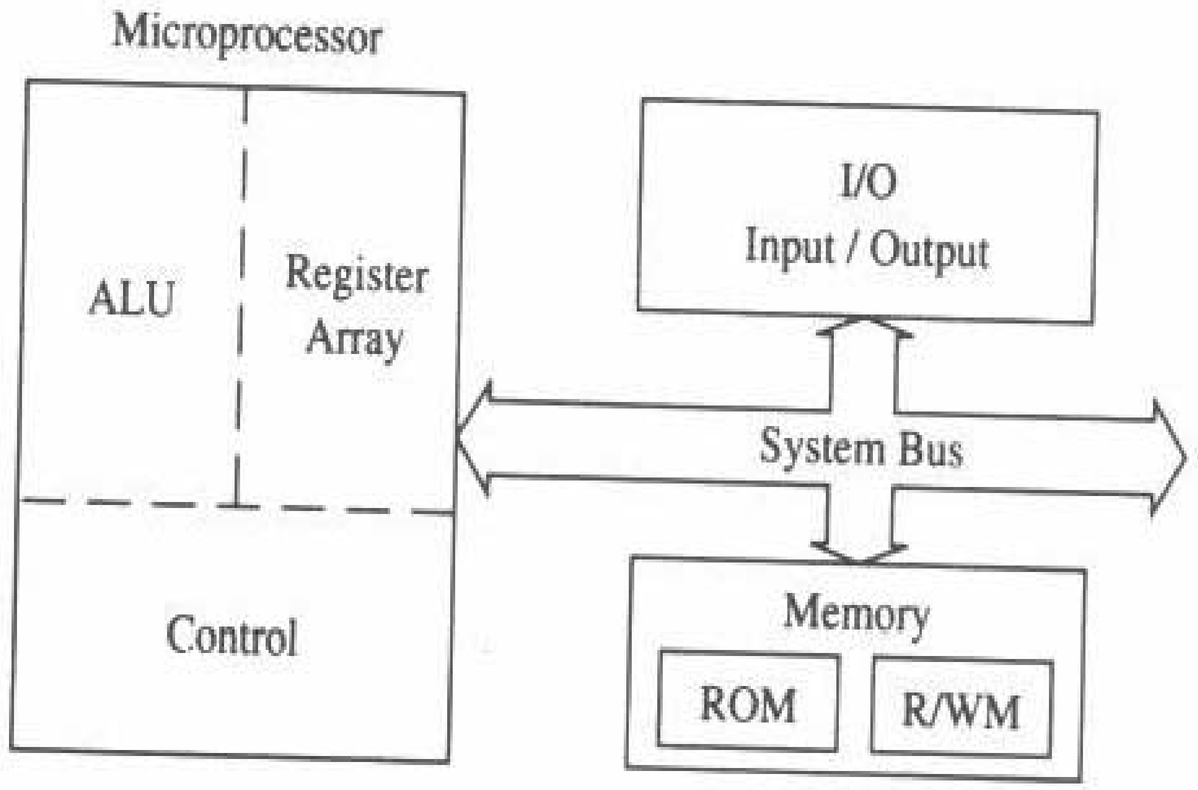
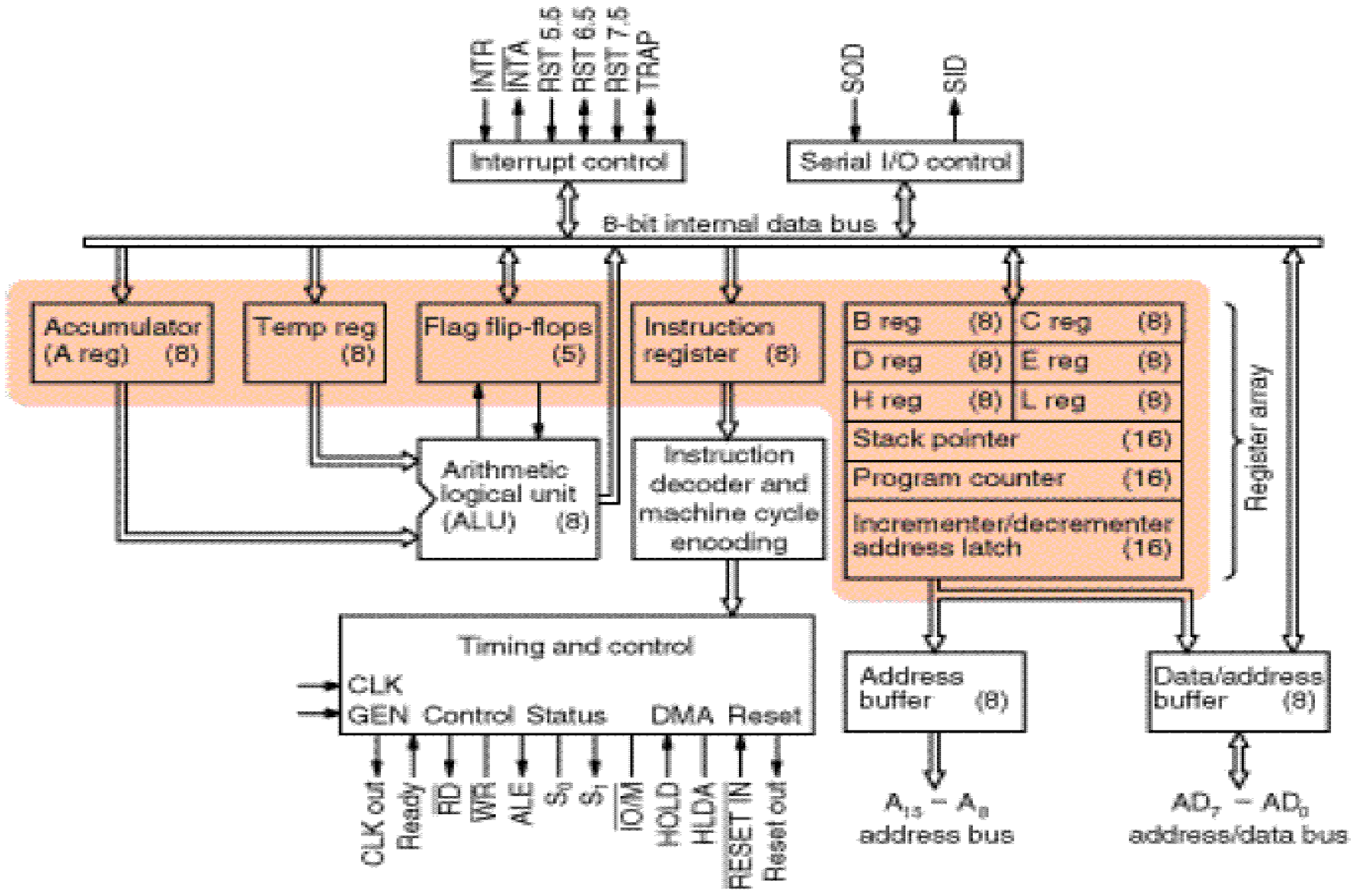


FIGURE 1.3
Microprocessor-Based System with Bus Architecture

Speed ARCHITECTURAL characteristics



Register Group

- Temporary registers (W,Z):These are not available for user. These are loaded only when there is an operation being performed.
- General purpose:There are six general purpose registers in 8085 namely B,C,D,E,H,L.These are used for various data manipulations.
- Special purpose :There are two special purpose registers in 8085:

1. SP :Stack Pointer.
2. PC:Program Counter.

Registers and GPR's controlregiser

Stack Pointer: This is a temporary storage memory 16 bit register. Since there are only 6 general purpose registers, there is a need to reuse them .

- Whenever stack is to be used previous values are PUSHED on stack and then after the program is over these values are POPED back.

Program Counter: It is 16 bit register used to point the location from which the next instruction is to be fetched.

- When a single byte instruction is executed PC is automatically incremented by 1.
- Upon reset PC contents are set to 0000H and next instruction is fetched onwards.

INSTRUCTION REGISTER,DECODER & CONTROL

- Instruction register:When an instruction is fetched , it is executed in instruction register.This register takes the Opcode value only.
- Instruction decoder: It decodes the instruction from instruction register and then to control block.
- Timing and control:This is the control section of μP.It accepts clock input .

Internal data Bus

- The 8085 is an 8-bit general purpose microprocessor that can address 64K Byte of memory.
- It has 40 pins and uses +5V for power. It can run at a maximum frequency of 3 MHz.
  - The pins on the chip can be grouped into 6 groups:
    - Address Bus.
    - Data Bus.
    - Control and Status Signals.
    - Power supply and frequency.
    - Externally Initiated Signals.
    - Serial I/O ports.

The Address and Data Busses

- The address bus has 8 signal lines A8 – A15 which are unidirectional.
- The other 8 address bits are multiplexed (time shared) with the 8 data bits.
  - So, the bits AD0 – AD7 are bi-directional and serve as A0 – A7 and D0 – D7 at the same time.
    - During the execution of the instruction, these lines carry the address bits during the early part, then during the late parts of the execution, they carry the 8 data bits.
  - In order to separate the address from the data, we can use a latch to save the value before the function of the bits changes.

The Control and Status Signals

- There are 4 main control and status signals. These are:
  - ALE: Address Latch Enable. This signal is a pulse that become 1 when the AD0 – AD7 lines have an address on them. It becomes 0 after that. This signal can be used to enable a latch to save the address bits from the AD lines.
  - RD: Read. Active low.
  - WR: Write. Active low.
  - IO/M: This signal specifies whether the operation is a memory operation (IO/M=0) or an I/O operation (IO/M=1).
  - S1 and S0 : Status signals to specify the kind of operation being performed .Usually un-used in small systems.

Frequency Control Signals

- There are 3 important pins in the frequency control group.
  - X0 and X1 are the inputs from the crystal or clock generating circuit.
    - The frequency is internally divided by 2.
      - So, to run the microprocessor at 3 MHz, a clock running at 6 MHz should be connected to the X0 and X1 pins.
  - CLK (OUT): An output clock pin to drive the clock of the rest of the system.
- We will discuss the rest of the control signals as we get to them.

Microprocessor Communication and Bus Timing

- To understand how the microprocessor operates and uses these different signals, we should study the process of communication between the microprocessor and memory during a memory read or write operation.
- Lets look at timing and the data flow of an instruction fetch operation. (Example 3.1)

Steps For Fetching an Instruction

- Lets assume that we are trying to fetch the instruction at memory location 2005. That means that the program counter is now set to that value.
  - The following is the sequence of operations:
    - The program counter places the address value on the address bus and the controller issues a RD signal.
    - The memory’s address decoder gets the value and determines which memory location is being accessed.
    - The value in the memory location is placed on the data bus.
    - The value on the data bus is read into the instruction decoder inside the microprocessor.
    - After decoding the instruction, the control unit issues the proper control signals to perform the operation.

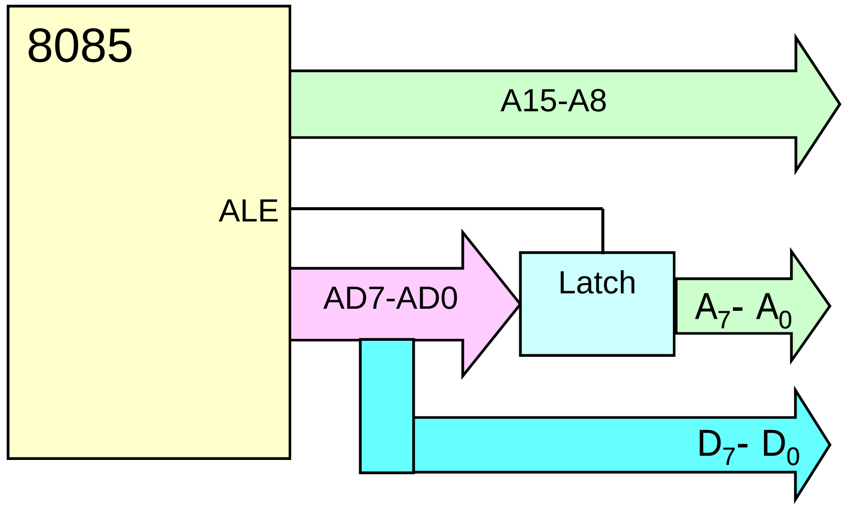
Timing Signals For Fetching an Instruction

- Now, lets look at the exact timing of this sequence of events as that is extremely important. (figure 3.3)
  - At T1 , the high order 8 address bits (20H) are placed on the address lines A8 – A15 and the low order bits are placed on AD7–AD0. The ALE signal goes high to indicate that AD0 – AD8 are carrying an address. At exactly the same time, the IO/M signal goes low to indicate a memory operation.
  - At the beginning of the T2 cycle, the low order 8 address bits are removed from AD7– AD0 and the controller sends the Read (RD) signal to the memory. The signal remains low (active) for two clock periods to allow for slow devices. During T2 , memory places the data from the memory location on the lines AD7– AD0 .
  - During T3 the RD signal is Disabled (goes high). This turns off the output Tri-state buffers in the memory. That makes the AD7– AD0 lines go to high impedence mode.

Demultiplexing AD7-AD0

- From the above description, it becomes obvious that the AD7– AD0 lines are serving a dual purpose and that they need to be demultiplexed to get all the information.
- The high order bits of the address remain on the bus for three clock periods. However, the low order bits remain for only one clock period and they would be lost if they are not saved externally. Also, notice that the low order bits of the address disappear when they are needed most.
- To make sure we have the entire address for the full three clock cycles, we will use an external latch to save the value of AD7– AD0 when it is carrying the address bits. We use the ALE signal to enable this latch.

Demultiplexing AD7-AD0



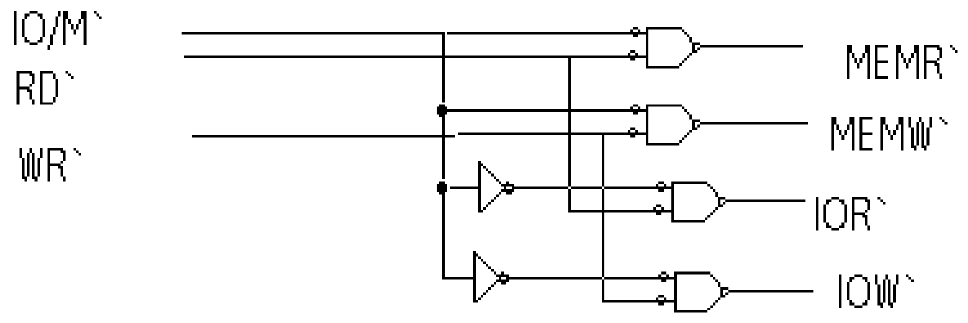
– Given that ALE operates as a pulse during T1, we will be able to latch the address. Then when ALE goes low, the address is saved and the AD7– AD0 lines can be used for their purpose as the bi-directional data lines.

Cycles and States

- From the above discussion, we can define terms that will become handy later on:
  - T- State: One subdivision of an operation. A T-state lasts for one clock period.
    - An instruction's execution length is usually measured in a number of T-states. (clock cycles).
  - Machine Cycle: The time required to complete one operation of accessing memory, I/O, or acknowledging an external request.
    - This cycle may consist of 3 to 6 T-states.
  - Instruction Cycle: The time required to complete the execution of an instruction.
    - In the 8085, an instruction cycle may consist of 1 to 6 machine cycles.

Generating Control Signals

- The 8085 generates a single RD signal. However, the signal needs to be used with both memory and I/O. So, it must be combined with the IO/M signal to generate different control signals for the memory and I/O.
  - Keeping in mind the operation of the IO/M signal we can use the following circuitry to generate the right set of signals:



The ALU

- In addition to the arithmetic & logic circuits, the ALU includes the accumulator, which is part of every arithmetic & logic operation.

- Also, the ALU includes a temporary register used for holding data temporarily during the execution of the operation. This temporary register is not accessible by the programmer.

The Flags register

- There is also the flags register whose bits are affected by the arithmetic & logic operations.
  - S-sign flag
    - The sign flag is set if bit D7 of the accumulator is set after an arithmetic or logic operation.
  - Z-zero flag
    - Set if the result of the ALU operation is 0. Otherwise is reset. This flag is affected by operations on the accumulator as well as other registers. (DCR B).
  - AC-Auxiliary Carry
    - This flag is set when a carry is generated from bit D3 and passed to D4 . This flag is used only internally for BCD operations. (Section 10.5 describes BCD addition including the DAA instruction).
  - P-Parity flag
    - After an ALU operation if the result has an even # of 1's the p-flag is set. Otherwise it is cleared. So, the flag can be used to indicate even parity.
  - CY-carry flag
    - Discussed earlier

Machine cycles

- The 8085 executes several types of instructions with each requiring a different number of operations of different types. However, the operations can be grouped into a small set.
- The three main types are:
    - Memory Read and Write.
    - I/O Read and Write.
    - Request Acknowledge.
- These can be further divided into various operations (machine cycles).

Opcode Fetch Machine Cycle

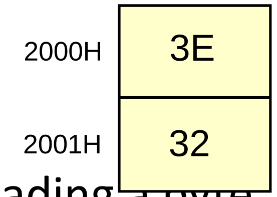
- The first step of executing any instruction is the Opcode fetch cycle.
  - In this cycle, the microprocessor brings in the instruction's Opcode from memory.
    - To differentiate this machine cycle from the very similar "memory read" cycle, the control & status signals are set as follows:
      - IO/M=0, s0 and s1 are both 1.
  - This machine cycle has four T-states.
    - The 8085 uses the first 3 T-states to fetch the opcode.
    - T4 is used to decode and execute it.
  - It is also possible for an instruction to have 6 T-states in an opcode fetch machine cycle.

Memory Read Machine Cycle

- The memory read machine cycle is exactly the same as the opcode fetch except:
  - It only has 3 T-states
  - The s0 signal is set to 0 instead.

The Memory Read Machine Cycle

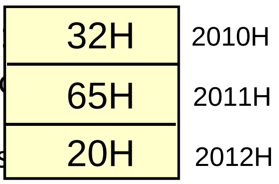
- To understand the memory read machine cycle, let’s study the execution of the following instruction:
  - MVI A, 32
- In memory, this instruction looks like:
  - The first byte 3EH represents the opcode for loading a byte into the accumulator (MVI A), the second byte is the data to be loaded.



- The 8085 needs to read these two bytes from memory before it can execute the instruction. Therefore, it will need at least two machine cycles.
    - The first machine cycle is the opcode fetch discussed earlier.
    - The second machine cycle is the Memory Read Cycle.
    - Figure 3.10 page 83.

Machine Cycles vs. Number of bytes in the instruction

- Machine cycles and instruction length, do not have a direct relationship.
  - To illustrate lets look at the machine cycles needed to execute the following instruction.
    - STA 2065H
    - This is a 3-byte instruction requiring 4 machine cycles and 13 T-states.
    - The machine code will be stored in memory as shown to the right
    - This instruction requires the following 4 machine cycles
      - Opcode fetch to fetch the opcode (32H) from location 2010H, deco determine that 2 more bytes are needed (4 T-states).
      - Memory read to read the low order byte of the address (65H) (3 T-s
      - Memory read to read the high order byte of the address (20H) (3 T-states).
      - A memory write to write the contents of the accumulator into the memory location.



The Memory Write Operation

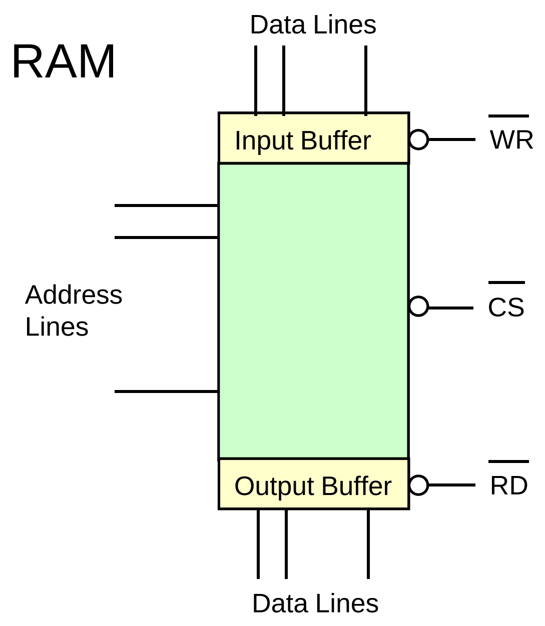
- In a memory write operation:
  - The 8085 places the address (2065H) on the address bus
  - Identifies the operation as a memory write (IO/M=0, s1=0, s0=1).
  - Places the contents of the accumulator on the data bus and asserts the signal WR.
  - During the last T-state, the contents of the data bus are saved into the memory location.

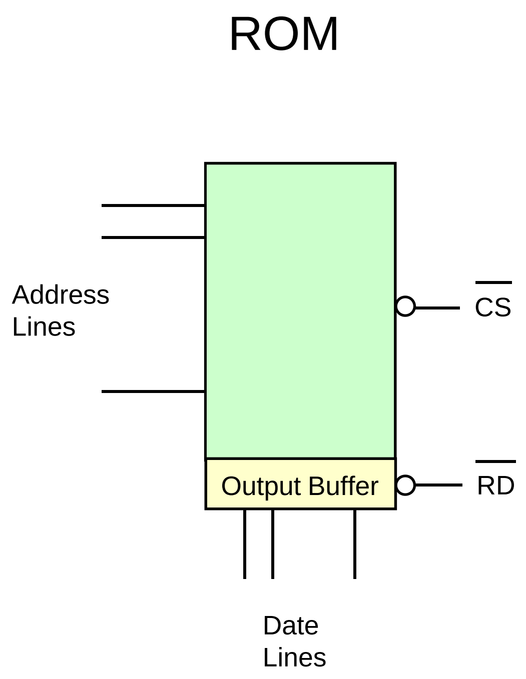
Memory interfacing

- There needs to be a lot of interaction between the microprocessor and the memory for the exchange of information during program execution.
  - Memory has its requirements on control signals and their timing.
  - The microprocessor has its requirements as well.

- The interfacing operation is simply the matching of these requirements.

Memory structure & its requirements





- The process of interfacing the above two chips is the same.
  - However, the ROM does not have a WR signal.

Interfacing Memory

- Accessing memory can be summarized into the following three steps:
  - Select the chip.
  - Identify the memory register.
  - Enable the appropriate buffer.

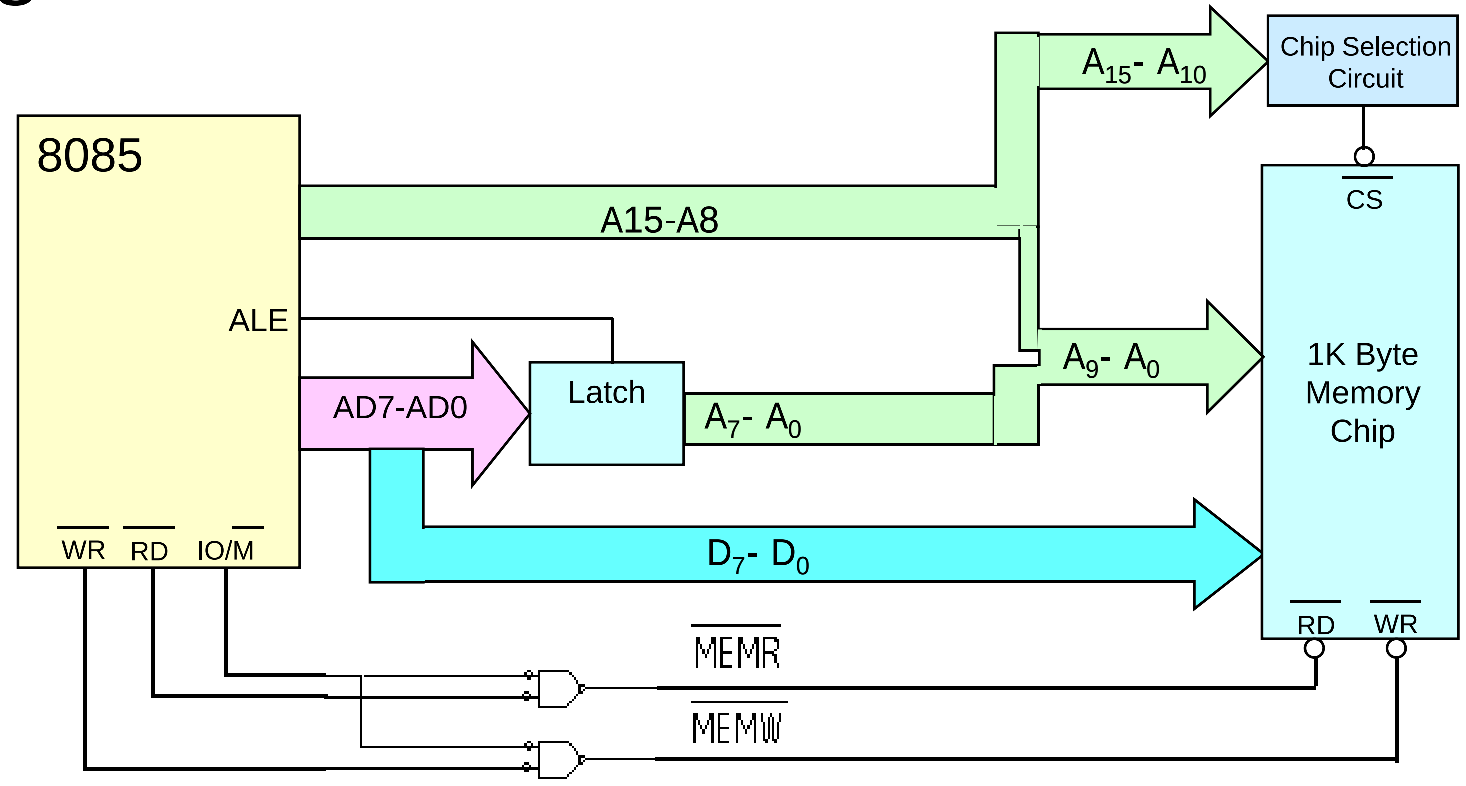
- Translating this to microprocessor domain:
  - The microprocessor places a 16-bit address on the address bus.
  - Part of the address bus will select the chip and the other part will go through the address decoder to select the register.
  - The signals IO/M and RD combined indicate that a memory read operation is in progress. The MEMR signal can be used to enable the RD line on the memory chip.

Address decoding

- The result of address decoding is the identification of a register for a given address.
  - A large part of the address bus is usually connected directly to the address inputs of the memory chip.
  - This portion is decoded internally within the chip.
  - What concerns us is the other part that must be decoded externally to select the chip.
  - This can be done either using logic gates or a decoder.

The Overall Picture

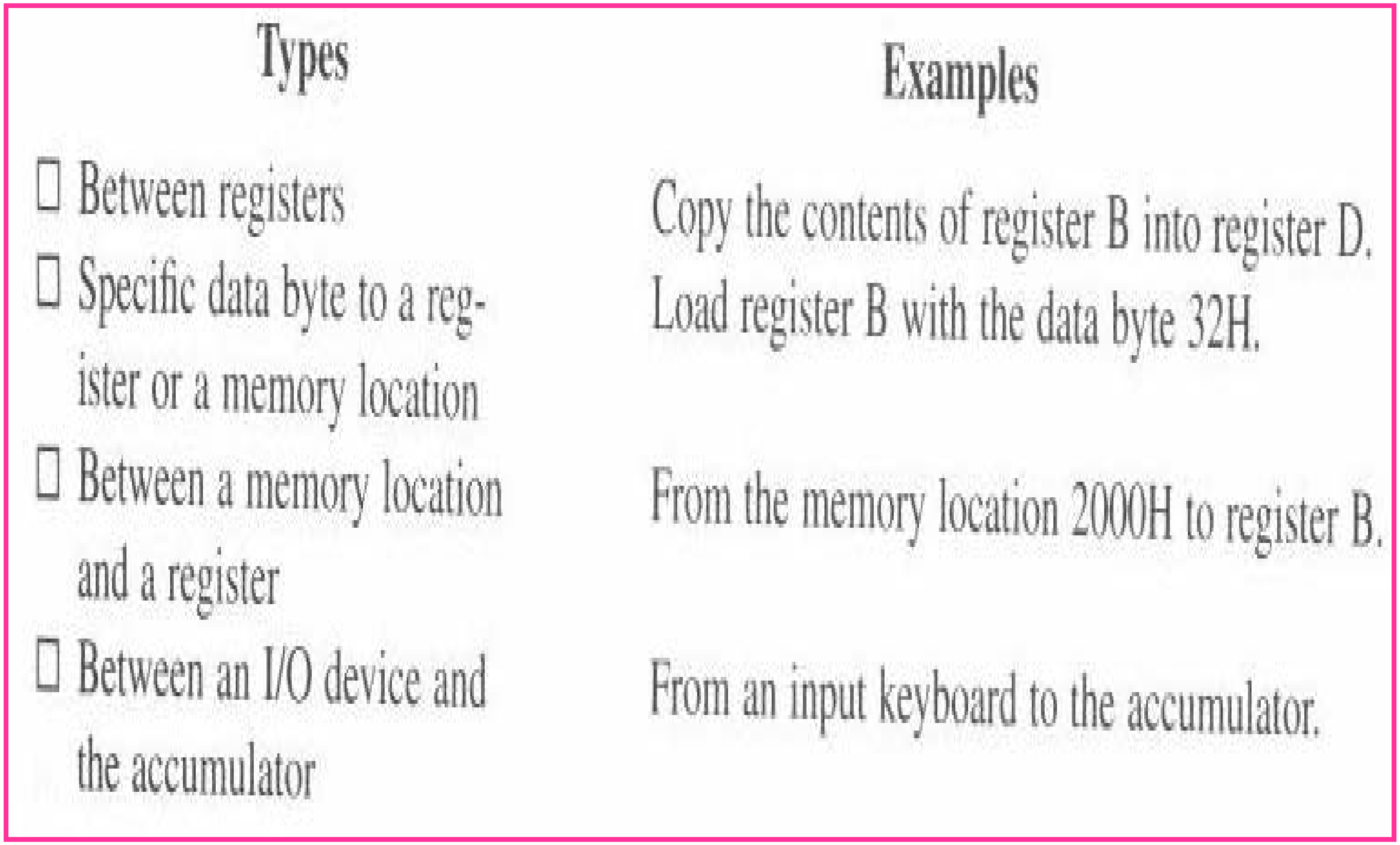
- Putting all of the concepts together, we get:



Instruction Set

- Can be classified into the following five functional categories:
  - data transfer (copy) operations,
  - arithmetic operations,
  - logical operations,
  - branching operations, and
  - machine-control operations.

Data Transfer (copy) Instructions



Types	Examples
☐ Between registers	Copy the contents of register B into register D.
☐ Specific data byte to a register or a memory location	Load register B with the data byte 32H.
☐ Between a memory location and a register	From the memory location 2000H to register B.
☐ Between an I/O device and the accumulator	From an input keyboard to the accumulator.

Data Transfer (Copy) Instructions

- These instruction perform the following 6 operations:
  - Load an 8-bit register
  - Copy from register to register
  - Copy between I/O and accumulator
  - Load 16-bit number in a register pair
  - Copy between register and memory
  - Copy between registers and stack memory

Arithmetic Instructions

- Add
- Subtract
- Increment (Add 1)
- Decrement (Subtract 1)

ARITHMETIC GROUP

ADD R (ADD register content with Acc and result in A).

Example:

ADD C. (ADD the content of C with A).

Suppose the Data at C register is 10H.

Initially	After execution
. C= 10H ,A=10H	A=20H,C=10H.

Flags Affected :All flags are modified.

Addressing mode: Register

Example

Instruction: **ADD B**

Register contents before instruction			Register contents after instruction:	
A	9A h		*A*	23 h
B	89 h		*B*	89 h
Flag	80 h		*Flag*	10 h

```
1 0 0 1   1 0 1 0
1 0 0 0   1 0 0 1
0 0 1 0   0 0 1 1
```

Note: All flags are modified to reflect the result of the addition.

Flag: S=0, Z=0, AC=1 , P=0 and C=1,

Logic and Bit Manipulation Instr.

- AND
- OR
- X-OR (Exclusive OR)
- Compare
- Rotate Bits

LOGICAL GROUP

Write a program to reset last 4 bits of the number 32H

Store result at C200H.

```
MVI A, 32H          A=32H
ANI F0H             00110010 AND 1111000
                     =00110000=30H
STA C200H.           C200=30H
RST1                  Stop
```

BRANCH GROUP

- JUMP
- CALL

- (conditional and unconditional)

BRANCH GROUP

JMP address(Unconditional jump to address)

Example:

JMP C200H.

- After this instruction the Program Counter is loaded with this location and starts executing and the contents of PC are loaded on Stack.

Flags Affected :No Flags are affected.

Addressing mode:Immediate.

CALL address(Unconditional CALL from address)

Example:

CALL C200H.

- After this instruction the Program Counter is loaded with this location and starts executing and the contents of PC are loaded on Stack.

Flags Affected :No Flags are affected.

Addressing mode:Immediate

BRANCH GROUP

Conditional Jump Instructions.

- JC (Jump if Carry flag is set)
- JNC (Jump if Carry flag is reset)
- JZ (Jump if zero flag set)
- JNZ (Jump if zero flag is reset)
- JPE (Jump if parity flag is set)
- JPO (Jump if parity odd or P flag is reset)
- JP (Jump if sign flag reset)
- JM (Jump if sign flag is set or minus)

BRANCH GROUP

Conditional Call Instructions.

- CC (Call if Carry flag is set)
- CNC (Call if Carry flag is reset)
- CZ (Call if zero flag set)
- CNZ (Call if zero flag is reset)
- CPE (Call if parity flag is set)
- CPO (Call if parity odd or P flag is reset)
- CP (Call if sign flag reset)
- CM (Call if sign flag is set or minus)

BRANCH GROUP

RET (Return from subroutine)

Example:

MOV A,C

RET

- After this instruction the Program Counter POPS PUSHED contents from stack and starts executing from that address .

Flags Affected :No Flags are affected.

Addressing mode:Register indirect .

BRANCH GROUP

RST (Restart instruction)

Example:

MOV A,C

RST 1.

- After this instruction the Program Counter goes to address 0008H and starts executing from that address .

Flags Affected :No Flags are affected.

Addressing mode:Register indirect.

BRANCH GROUP

The addresses of the respective RST commands are:

Instruction	Address
RST 0	0000H
RST 1	0008H
RST 2	0010H
RST 3	0018H
RST 4	0020H
RST 5	0028H
RST 6	0030H
RST 7	0038H

STACK AND MACHINE CONTROL

PUSH Rp.(Push register pair contents on stack).

Example:LXI SP FFFFH.

PUSH H. (Move the content of HL pair on Stack).

- Suppose at HL pair the data is H= 20H,L= 30H & SP is initialized at FFFFH

Initially	After execution
H=20H,L=30H	H=20H,L=30H.
SP=FFFF H	FFFD=30H,FFFE=20H

Flags Affected :No flags affected.

Addressing mode: Register indirect.

STACK AND MACHINE CONTROL

POP Rp.(Pop register pair contents from stack).

Example:POP D(POP the content of DE pair from Stack).

- Suppose at DE pair the data is H= 20H,L= 30H SP was initialized at FFFFH

Initially	After execution
D=20H,E=30H	D=10H,E=80H.
FFFD=80H,FFFE=10H	

Flags Affected :No flags affected.

Addressing mode: Register indirect

STACK AND MACHINE CONTROL

XTHL (Exchange HL register pair contents with top of stack).

Example:XTHL(Exchange top with HL pair).

- Suppose at HL pair the data is H= 20H,L= 30H & SP =FFFFH

& at locations FFFF=10H and at FFFE= 80H.

Initially	After execution
H=20H,L=30H	H=10H,L=80H.
SP=FFFF =10H,FFFE=80H	FFFD=20H,FFFE=30H

Flags Affected :No flags affected.

Addressing mode: Register indirect.

Common Errors

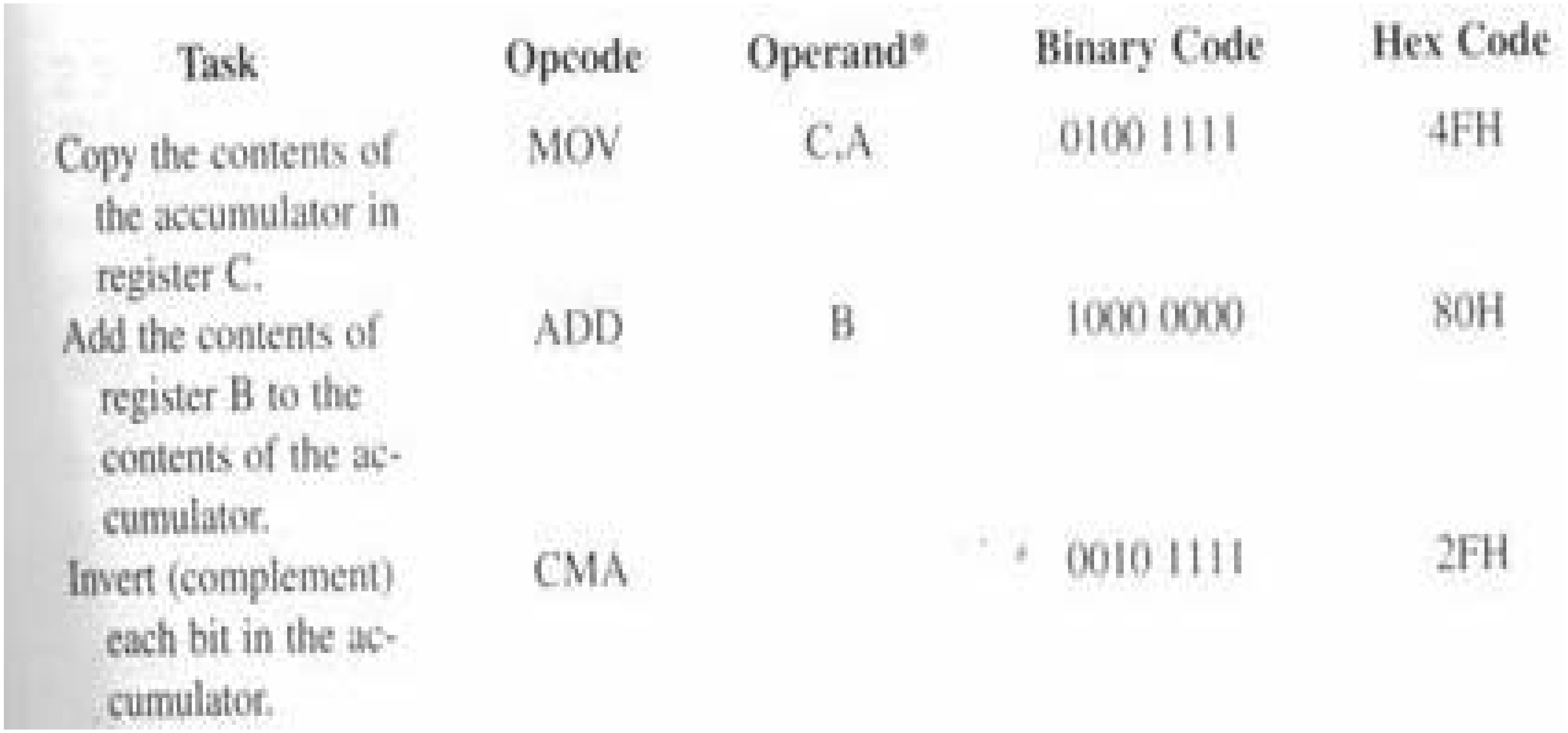
- LDA 205lH: Not entering the code of the 16-bit address in reversed order.
- Forgetting to enter the code for the operand, such as 205lH.
- MOV B, A: Assuming that this copies from B to A.
- Incrementing the address in decimal, from 2039H to 2040H.
- HLT: Not terminating a program.
- Confusing the entering of Hex code in memory as executing a program.

Instruction, Data Format, And Storage

- In the 8085, “byte” and “word” are synonymous because it is an 8-bit microprocessor.
- However, instructions are commonly referred to in terms of bytes rather than words.
  - 1-byte instructions
  - 2-byte instructions
  - 3-byte instructions

1-byte instructions

- It includes the opcode and the operand in the same byte. For example:



Task	Opcode	Operand*	Binary Code	Hex Code
Copy the contents of the accumulator in register C.	MOV	C,A	0100 1111	4FH
Add the contents of register B to the contents of the accumulator.	ADD	B	1000 0000	80H
Invert (complement) each bit in the accumulator.	CMA		0010 1111	2FH

2-byte instructions

- The first byte specifies the operation code and the second byte is the data
- For example:

MVI A,20H. Transfer immediate data 20H to accumulator.

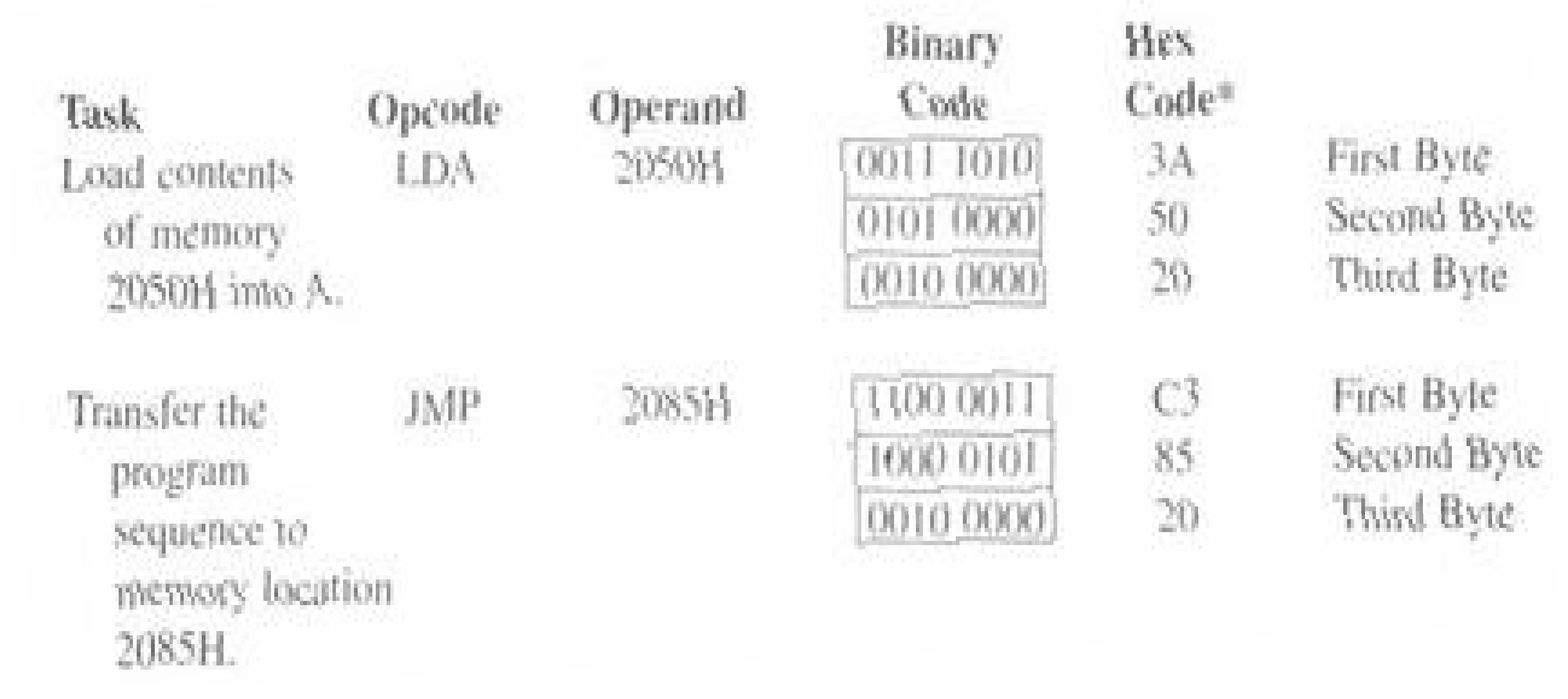
Number of bytes: 2

3E,06H

1st byte is opcode.
2nd byte 8 bit data .

3-byte instructions

- The first byte specifies the operation code and the following two bytes specify the 16-bit address.
- The second byte is the low-order address and the third byte is the high-order address.
- For example:



Task	Opcode	Operand	Binary Code	Hex Code*	
Load contents of memory 2050H into A.	LDA	2050H	0011 1010	3A	First Byte
			0101 0000	50	Second Byte
			0010 0000	20	Third Byte
Transfer the program sequence to memory location 2085H.	JMP	2085H	1100 0011	C3	First Byte
			1000 0101	85	Second Byte
			0010 0000	20	Third Byte