Free Study Material from All Lab Experiments

Digital Systems & Applications
Chapter - 9, 10, 11, 12
9.Shift Register 10.Counters 11.Computer Organization
12. Intel 8085 Microprocessor Architecture

Support us by Donating
at the link "DONATIONS" given on the Main Menu

Even the smallest contribution of you
will Help us keep Running

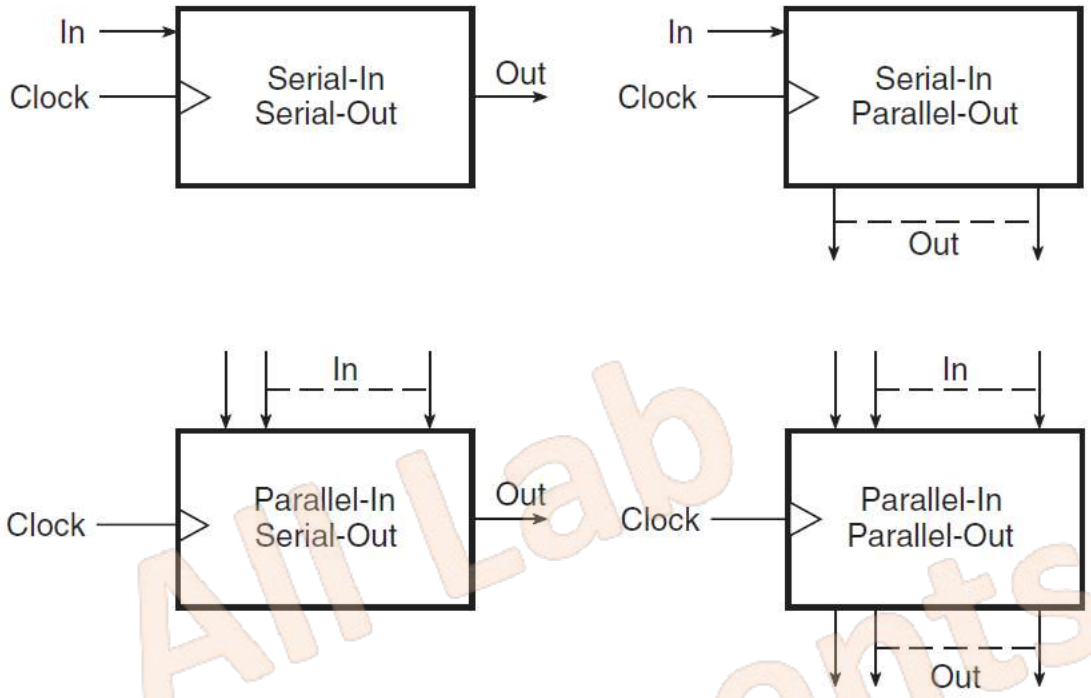
Chapter – 9

Shift Registers

Shift registers: Serial-in-Serial-out, Serial-in-Parallel-out, Parallel-in-Serial-out and Parallel-in-Parallel-out Shift Registers (only up to 4 bits). **(2 Lectures)**

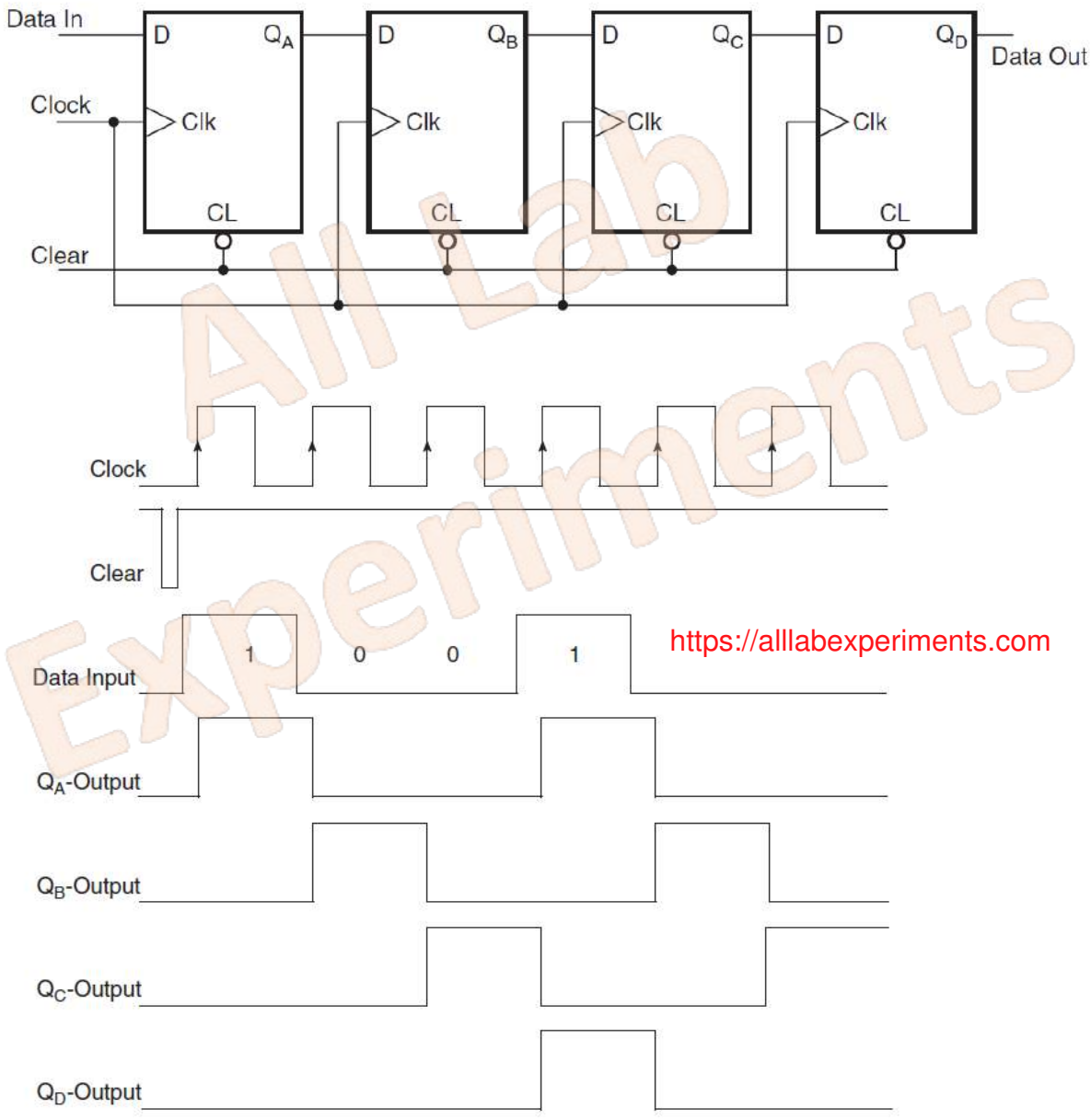
Que1: Draw block diagram for different type of shift registers.

Ans:



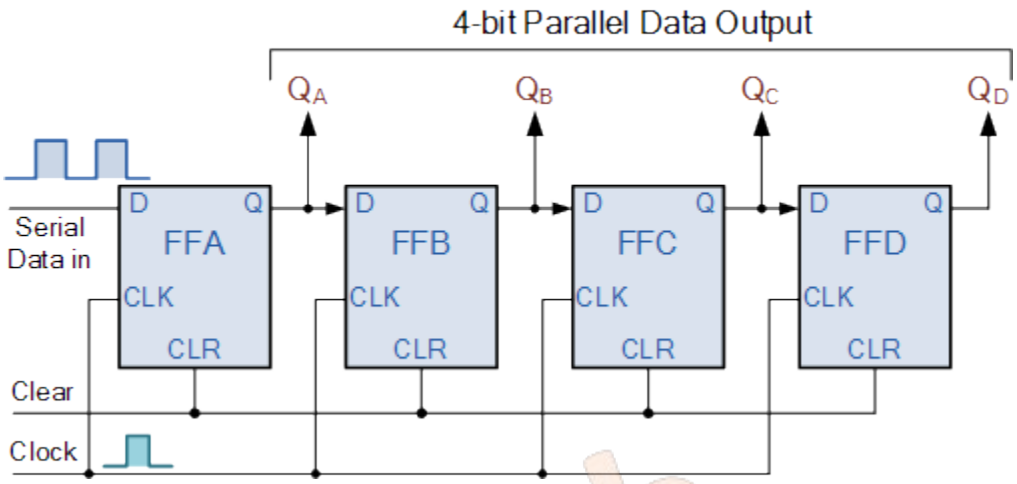
Que2: Design a 4-bit Serial-in-Serial-out shift register using D flip flops and draw its waveform.

Ans: Figure below shows the basic four-bit serial-in serial-out shift register implemented using D flip-flops. The circuit functions as follows. A reset applied to the CLEAR input of all the flip-flops resets their Q outputs to 0s. Refer to the timing waveforms of Fig. 11.36. The waveforms shown include the clock pulse train, the waveform representing the data to be loaded onto the shift register and the Q outputs of different flip-flops. The flip-flops shown respond to the LOW-to-HIGH transition of the clock pulses as indicated by their logic symbols.



Que3: Design a 4-bit serial-in-Parallel-out shift register.

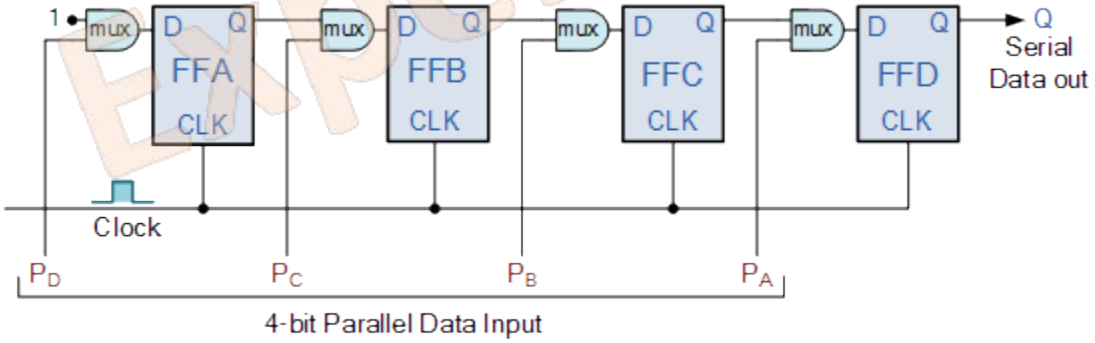
Ans: In order to shift the data out in parallel, it is simply necessary to have all the data bits available as outputs at the same time. This is easily accomplished by connecting the output of each flip-flop to an output pin. For instance, an 4-bit shift register would have four output lines- one for each flip-flop in the register.



Que4: Explain the working of a 4-bit Parallel-in-Serial-out shift register.

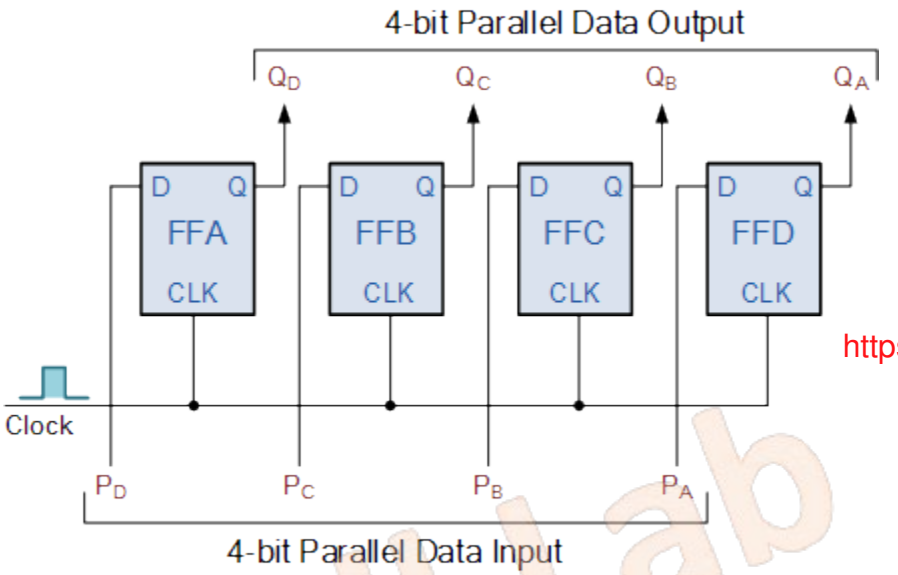
Ans: The Parallel-in to Serial-out shift register acts in the opposite way to the serial-in to parallel-out one above. The data is loaded into the register in a parallel format in which all the data bits enter their inputs simultaneously, to the parallel input pins P_A to P_D of the register. The data is then read out sequentially in the normal shift-right mode from the register at Q representing the data present at P_A to P_D.

This data is outputted one bit at a time on each clock cycle in a serial format. It is important to note that with this type of data register a clock pulse is not required to parallel load the register as it is already present, but four clock pulses are required to unload the data.



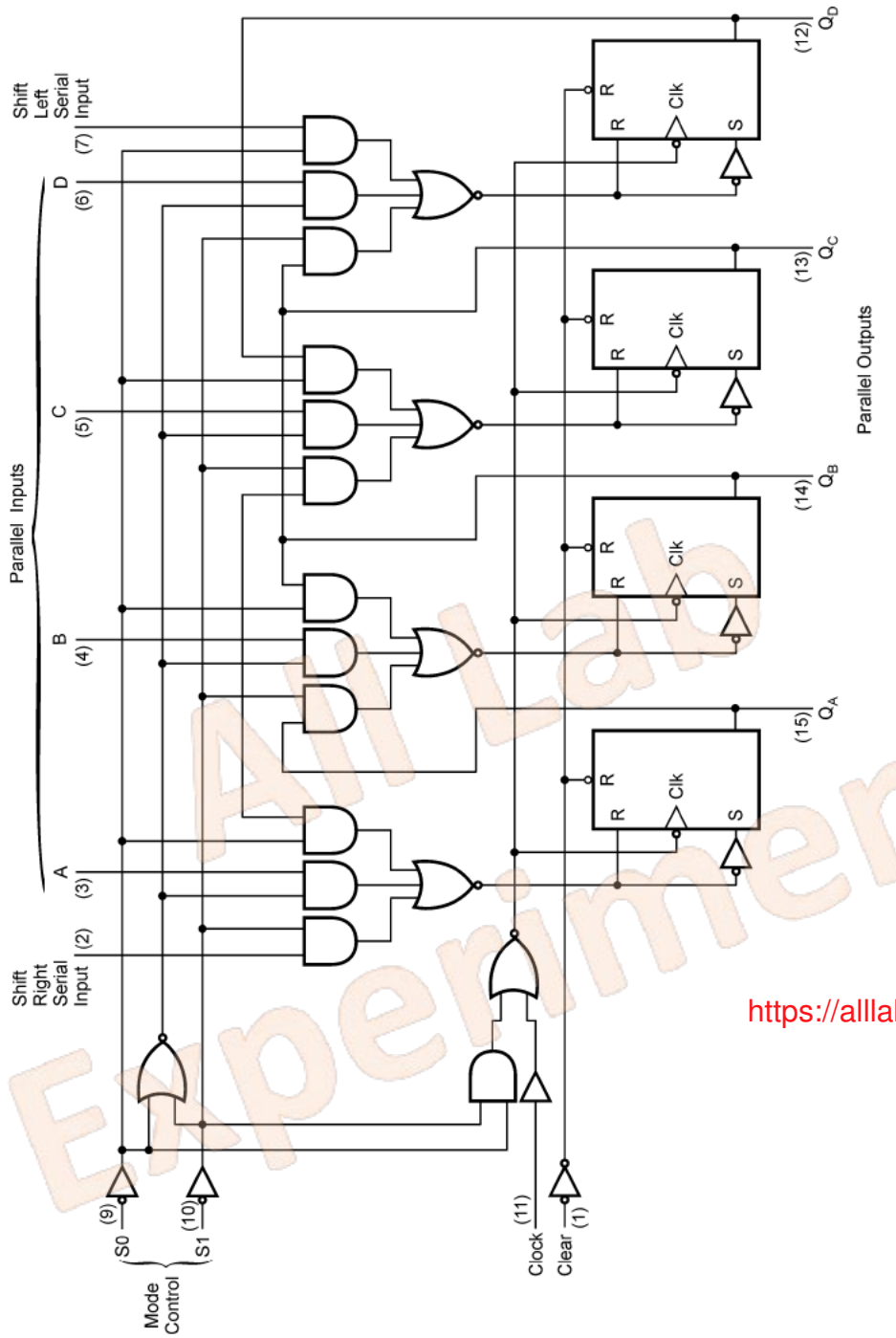
Que5: Design a 4-bit Parallel-in to Parallel-out (PIPO) Shift Register

Ans: This type of shift register also acts as a temporary storage device or as a time delay device similar to the Serial-in-Serial-out configuration above. The data is presented in a parallel format to the parallel input pins P_A to P_D and then transferred together directly to their respective output pins Q_A to Q_A by the same clock pulse. Then one clock pulse loads and unloads the register. This arrangement for parallel loading and unloading is shown below.



Que6: Design and explain a four-bit universal shift register.

Ans: A universal shift register can be made to function as any of the four types of register. That is, it has serial/parallel data input and output capability, which means that it can function as serial-in serial-out, serial-in parallel-out, parallel-in serial out and parallel-in parallel-out shift registers. the device offers four modes of operation, namely (a) inhibit clock, (b) shift right, (c) shift left and (d) parallel load. Clocking of the device is inhibited when both the mode control inputs S1 and S0 are in the logic LOW state. shift right and shift left operations are accomplished synchronously with LOW-to-HIGH transition of the clock with S1 LOW and S0 HIGH (for shift right) and S1 HIGH and S0 LOW (for shift left). Serial data are entered in the case of shift right and shift left operations at the corresponding data input terminals. Parallel loading is also accomplished synchronously with LOW-to-HIGH clock transitions by applying four bits of data and then driving the mode control inputs S1 and S0 to the logic HIGH state. Data are loaded into corresponding flipflops and appear at the outputs with LOW-to-HIGH clock transition. Serial data flow is inhibited during parallel loading.



Chapter – 10

Counters

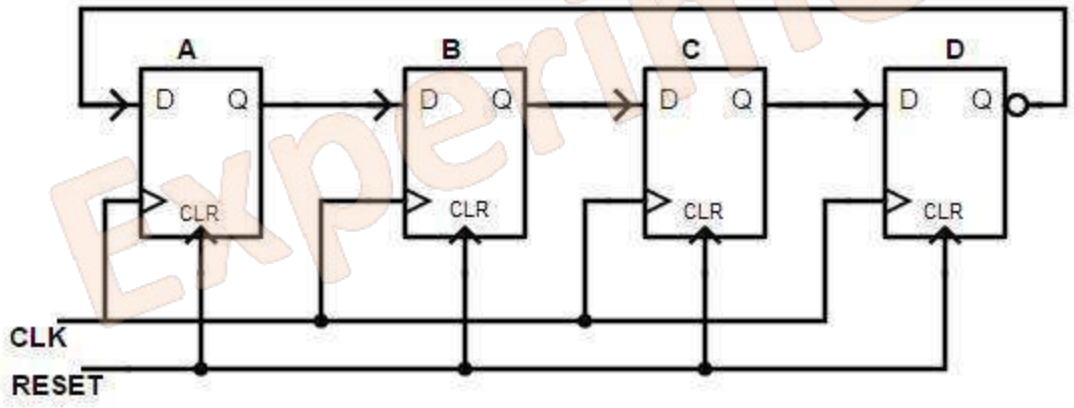
Counters(4 bits): Ring Counter. Asynchronous counters, Decade Counter. Synchronous Counter. **(4 Lectures)**

Que1: What is a ring counter and design a 4- bit ring counter.

Ans: The ring counter is a cascaded connection of flip flops, in which the output of last flip flop is connected to input of first flip flop. In ring counter if the output of any stage is 1, then its reminder is 0. The Ring counters transfers the same output throughout the circuit.

That means if the output of the first flip flop is 1, then this is transferred to its next stage i.e. 2nd flip flop. By transferring the output to its next stage, the output of first flip flop becomes 0. And this process continues for all the stages of a ring counter. If we use n flip flops in the ring counter, the '1' is circulated for every n clock cycles.

The circuit diagram of the ring counter is shown below.



Here we design the ring counter by using D flip flop. This is a Mod 4 ring counter which has 4 D flip flops connected in series. The clock signal is applied to clock input of each flip flop, simultaneously and the RESET pulse is applied to the CLR inputs of all the flip flops.

Operation of Ring Counter

Initially, all the flip flops in ring counter are reset to 0 by applying CLEAR signal. Before applying the clock pulse, we apply the PRESET pulse to the flip flops which assigns the value '1' to the

ring counter circuit. For each clock signal, the data circulates among all the 4 flip flop stages of ring counter.This 4 staged ring counter is called Mod 4 ring counter or 4 bit ring counter. To circulate the data correctly in the ring counter, we must load the counter with required values like all 0's or all 1's.

Truth table of ring counter

The truth table of the 4 -bit ring counter is explained below.

Q_0	Q_1	Q_2	Q_3
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

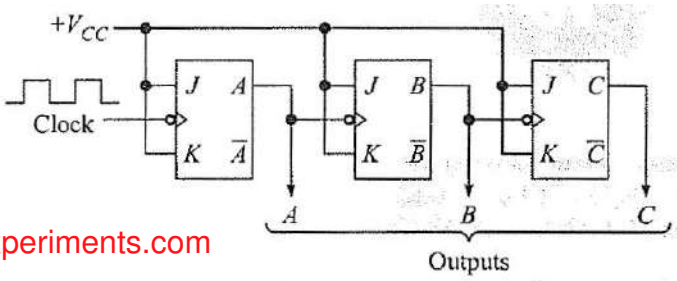
When CLEAR input CLR = 0, then all flip flops are set to 1. When CLEAR input CLR = 1, the ring counter starts its operation. For one clock signal, the counter starts its operation. On next clock signal, the counter again resets to 0000. Ring counter has 4 sequences: 0001, 0010, 0100, 1000, 000.

Que2: What is an Asynchronous counter.

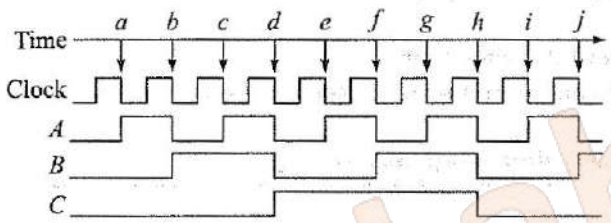
Ans: When the output of a flip-flop is used as the clock input for the next flip-flop, we call the counter a ripple counter, or asynchronous counter. The A flip-flop must change state before it can trigger the B flip-flop, and the B flip-flop has to change state before it can trigger the C flip-flop and so on. The triggers move through the flip-flops like a ripple in water.

Que3: Design a three- bit binary ripple counter and draw the output waveform.

Ans: A binary ripple counter can be constructed using clocked JK flip-flops. Figure 10.1 shows three negative edge- triggered, JK flip-flops connected in cascade. The system clock, a square wave, drives flip-flop A. The output of A drives B, and the output of B drives flip-flop C. All the J and K inputs are tied to +Vcc· This means that each flip-flop will change state (toggle) with a negative transition at its clock input.



(a) Three-bit binary ripple counter



(b) Waveforms

Negative clock transitions	C	B	A	State or count
---	0	0	0	0
a	0	0	1	1
b	0	1	0	2
c	0	1	1	3
d	1	0	0	4
e	1	0	1	5
f	1	1	0	6
g	1	1	1	7
h	0	0	0	0

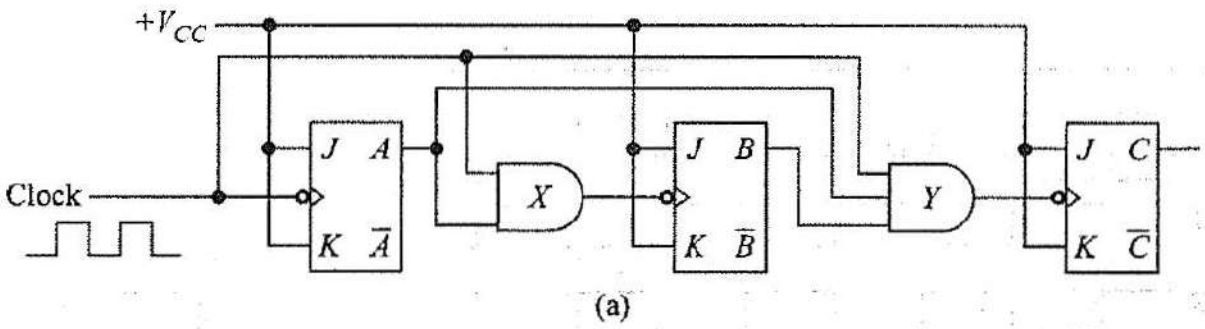
(c) Truth table

Que4: What is the limitation of an Asynchronous counter and how does synchronous counter overcome that.

Ans: The ripple counter is the simplest to build, but there is a limit to its highest operating frequency. As previously discussed, each flip-flop has a delay time. In a ripple counter these delay times are additive, and the total "settling" time for the counter is approximately the delay time times the total number of flip-flops. Furthermore, there is the possibility of glitches occurring at the output of decoding gates used with a ripple counter. The first problem fully and the second problem, to some extent can be overcome by use of a synchronous parallel counter. The main difference here is that every flip-flop is triggered in synchronism with the clock.

Que5: Design a Mod-8 binary counter with parallel clock input and draw the output waveform.

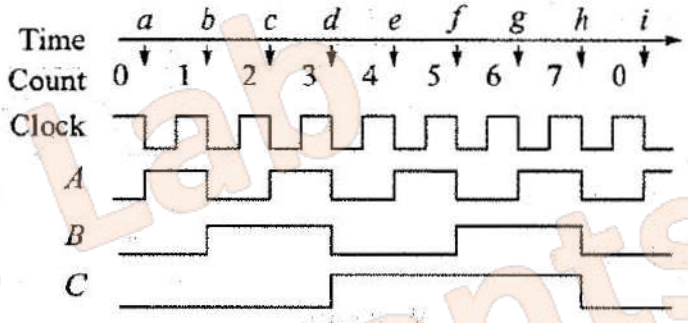
Ans: The basic idea here is to keep the *J* and *K* inputs of each flip-flop high, such that the flip-flop will toggle with any clock NT at its clock input. We then use AND gates to gate every second clock to flip-flop *B,* every fourth clock to flip-flop C, and so on. This logic configuration is often referred to as "steering logic" since the clock pulses are gated or steered to each individual flip-flop. Figure below shows the implementation of Mod-8 binary counter using parallel clock input.



(a)

C	B	A	Count
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7
0	0	0	0

(b)



(c)

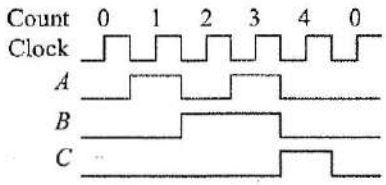
Que6: Construct and explain the working 0f a Mod-5 counter and write down the truth table.

Ans: The three-flip-flop counter shown in Fig. below has a natural count of 8, but it is connected in such a way that it will skip over three counts. It will, in fact, advance one count at a time, through a strict binary sequence, beginning with 000 and ending with 100; therefore, it is a mod-5 counter. Let's see how it works. The waveforms show that flip-flop *A* changes state each time the clock goes negative, except during the transition from count 4 to count 0. Thus, flip-flop *A* should be triggered by the clock and must have an inhibit during count 4-that is, some signal must be provided during the transition from count 4 to count 0. Notice that *C* is high during all counts except count 4. If *C* is connected to the *J* input of flip-flop *A,* we will have the desired inhibit signal. This is true since the *J* and *K* inputs to flip-flop *A* are both true for all counts except count 4; thus the flip-flop triggers each time the clock goes negative. However, during count 4, the *J* side is
low and the next time the clock goes negative the flip-flop will be prevented from being set. The connections which cause flip-flop *A* to progress through the desired sequence are shown in Fig. below. The desired waveforms (Fig. b) show that flip-flop *B* must change state each time *A* goes negative. Thus the clock input of flip-flop *B* will be driven by *A* (Fig. c).
If flip-flop *C* is triggered by the clock while the *J* input is held low and the *K* input high, every clock pulse will reset it. Now, if the *J* input is high only during count 3, *C* will be high during count 4 and low during all other counts. The necessary levels for the *J* input can be obtained by

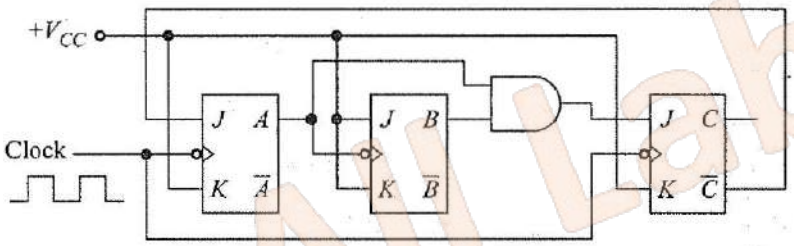
ANDing flip-flops *A* and *B.* Since *A* and *Bare* both high only during count 3, the *J* input to flip-flop *C* is high only during count 3. Thus, when the clock goes negative during the transition from count 3 to count 4, flip-flop *C* will be set. At all other times, the *J* input to flip-flop *C* is low and is held in the reset state. The complete mod-5 counter is shown in Fig below.

C	B	A	Count
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
0	0	0	0

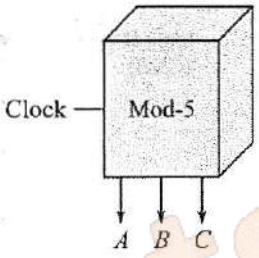
(a)



(b)



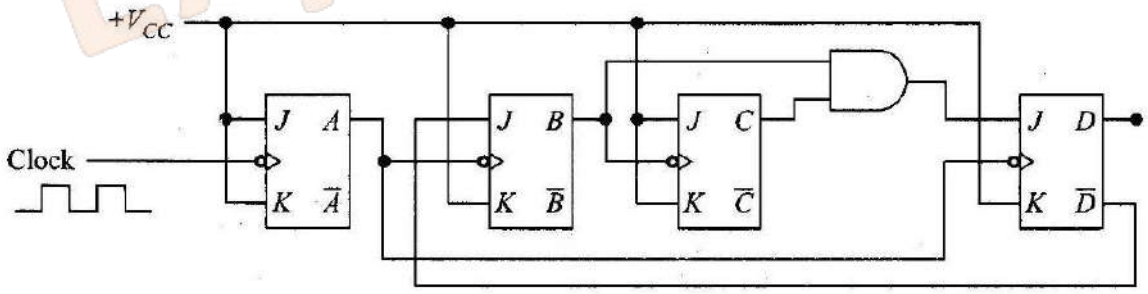
(c)

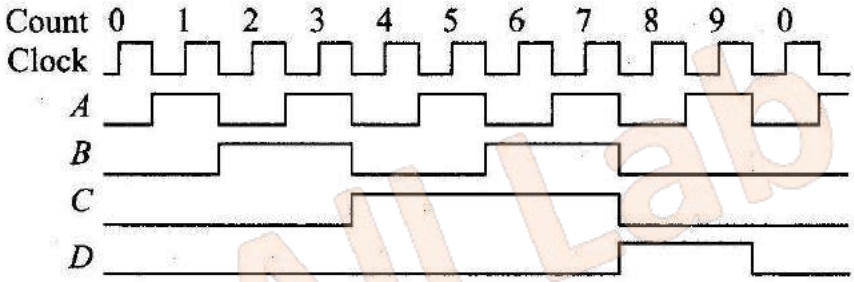


(d) Logic block

Que7: Draw the circuit, waveform and truth Table of a decade counter.

Ans: A decade up counter can be designed using J-K flip Flops. There are ten states in a decade counter. Four flip flops are required for this purpose. The remaining six states are not used states.





D	C	B	A	Count
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
0	0	0	0	0

Chapter – 11

Computer organization

Computer Organization: Input/Output Devices. Data storage (idea of RAM and ROM). Computer memory. Memory organization and addressing. Memory Interfacing. Memory Map. **(6 Lectures)**

Que1: What is a memory?

Ans. A memory is a device that stores information in electrical, magnetic or optical form. A Micro-controller based system, which operates on digital logic, holds binary information. Semiconductor memories are used in micro-controller based system.

Que2: In how many categories memories can be classified? Give examples and distinguish between them.

Ans. It can be categorised into two ways:

- Primary memory or main memory or working memory.
- Secondary memory or auxiliary memory or mass storage.

RAM and ROM comprise the primary memory while magnetic tapes, magnetic disks, floppy disks or compact disks (CDs) are examples of secondary memory.
Distinction between the two types of memories are:

S. No.	Primary memory		Secondary memory
1.	Can store less amount of data.	1.	Can store huge amount of data.
2.	Faster speed of operation.	2.	Slower speed of operation.
3.	Can be volatile/non-volatile in nature.	3.	Always non-volatile in nature.
4.	Program/data used by the programmer are resident in the main memory.	4.	Not used for such purpose.
5.	Can be directly accessed by CPU.	5.	Cannot be directly accessed by CPU but can be accessed through I/O ports or in a serial format using hardware/software.
6.	If the CPU has n address lines, a maximum of 2^n main memory locations can be accessed.	6.	No such relation exists.
7.	Less costly.	7.	Costlier than main memory.

Que3: Make a comparison between RAM and ROM

Ans: Comparison Chart

BASIS FOR COMPARISON	RAM	ROM
Basic	It is a read-write memory.	It is read only memory.
Use	Used to store the data that has to be currently processed by CPU temporarily.	It stores the instructions required during bootstrap of the computer.
Volatility	It is a volatile memory.	It is a nonvolatile memory.
Stands for	Random Access Memory.	Read Only Memory.
Modification	Data in ROM can be modified.	Data in ROM can not be modified.
Capacity	RAM sizes from 64 MB to 4GB.	ROM is comparatively smaller than RAM.
Cost	RAM is a costlier memory.	ROM is comparatively cheaper than RAM.

BASIS FOR COMPARISON	RAM	ROM
Type	Types of RAM are static RAM and dynamic RAM.	Types of ROM are PROM, EPROM, EEPROM.

Que4: Explain Static and Dynamic RAM.

Ans: There are two kinds of RAM, **Static RAM** and **Dynamic RAM**. **Static RAM** is one which requires the constant flow of the power to retain the data inside it. It is **faster** and **more expensive** than DRAM. It is used as a **cache memory** for the computer. **Dynamic RAM** needs to be refreshed to retain the data it holds. It is **slower** and **cheaper** than static RAM.

Que5: Distinguish between between EPROM and EEPROM.

Ans: Comparison Chart

BASIS FOR COMPARISON	EPROM	EEPROM
Basic	Ultraviolet Light is used to erase the content of EPROM.	EEPROM contents are erased using electronic signal.
Appearance	EPROM has a transparent quartz crystal window at the top.	EEPROM are totally encased in an opaque plastic case.

BASIS FOR COMPARISON	EPROM	EEPROM
Erased and Reprogrammed	EPROM chip has to be removed from the computer circuit to erase and reprogram the computer BIOS.	EEPROM chip can be erased and reprogrammed in the computer circuit to erase and reprogram the content of computer BIOS.
Technology	EPROM is an older technology.	EEPROM is a modern version over EPROM.

Que6: A semiconductor memory is specified as 4 K × 8. Indicate the number of words, word size and total capacity of this memory.

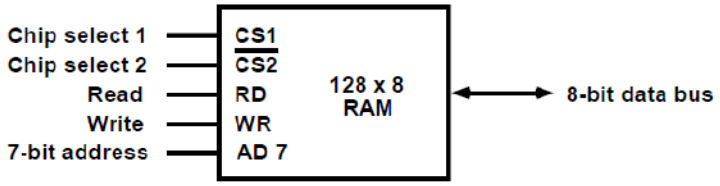
Ans. Total number of words or memory locations that can be accessed is
= 4 K = 4 × 1024 = 4096
Word size = 8
and total capacity of the memory is
Data inputs = 4096 × 8 = 32768

Que7: Diagrammatically show a 128 × 8 memory, indicating the relevant pins.

Ans:

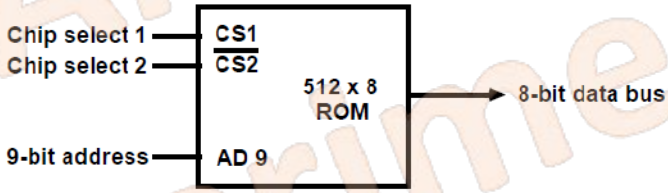
RAM and ROM Chips

Typical RAM chip



CS1	$\overline{CS2}$	RD	WR	Memory function	State of data bus
0	0	x	x	Inhibit	High-impedence
0	1	x	x	Inhibit	High-impedence
1	0	0	0	Inhibit	High-impedence
1	0	0	1	Write	Input data to RAM
1	0	1	x	Read	Output data from RAM
1	1	x	x	Inhibit	High-impedence

Typical ROM chip



Que8: Explain how the entire addressable memory space of 8085 can be conceived of the pages of a book?

Ans:

The entire addressable memory space is 2^{16} = 65,536 lines, because 8085 has 16 address lines. This is arbitrarily divided into 256 pages (0 to 255 pages) with each page consisting of 256 lines (0 to 255 lines), as shown in Fig. 8.4

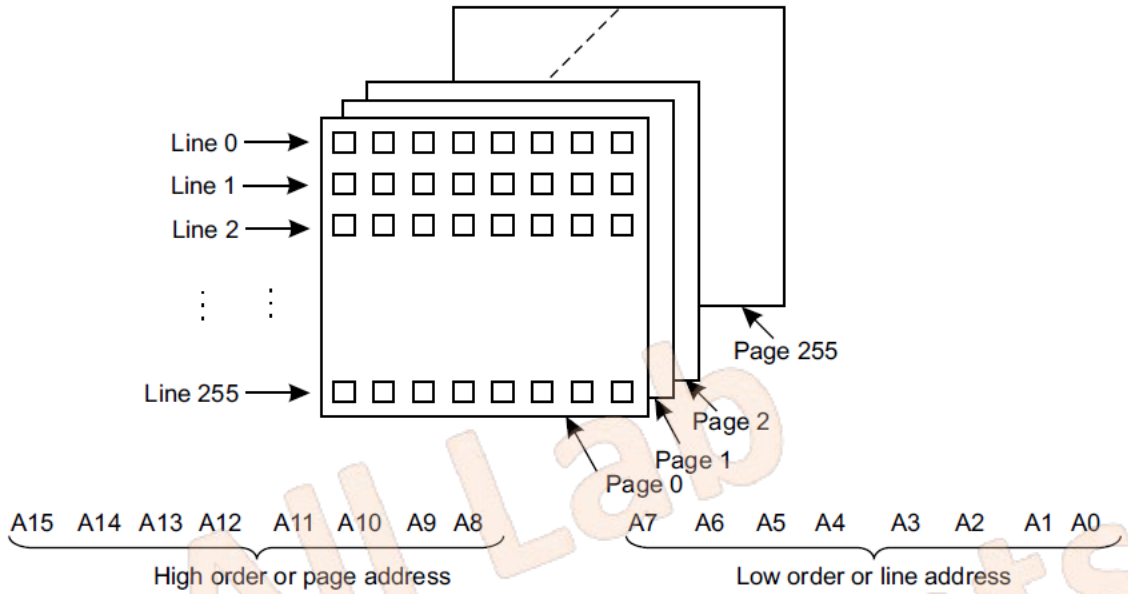


Fig. 8.4: Arranging total memory space of 8085 into pages and lines

The sixteen address lines can be divided into higher ($A_{15} - A_8$) and lower ($A_7 - A_0$) eight bits. On each page there are 256 lines defined by the lower byte ($A_7 - A_0$), while each page of the book is defined by the different combinations of the upper byte ($A_{15} - A_8$).

The following figure shows an arbitrary but convenient way of showing the total memory space, with each block consisting of 4 K lines i.e., 16 pages.

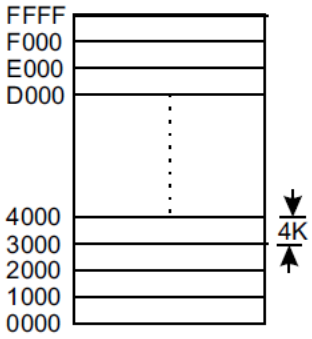
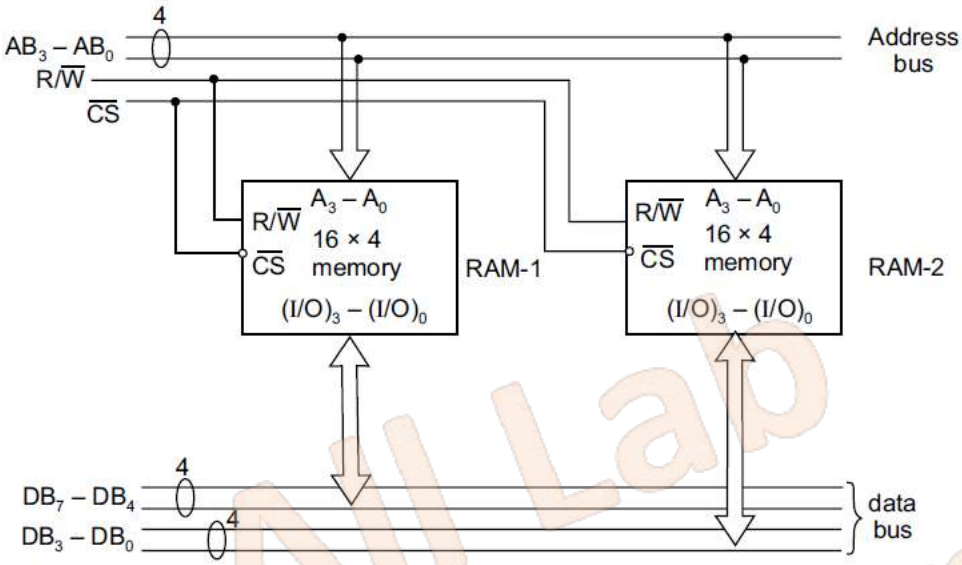


Fig. 8.5: Total memory space of 8085

Que9: Design a memory having size 16 × 8 from 16 × 4 memory modules.

Ans:

A 16 × 8 memory module indicates that it can address 16 different memory addresses, each address location can store 1 byte of data/instruction. The design is given below:



The address lines $AB_3 - AB_0$ can address 16 different addresses (from 0000 to 1111) and connected to the two RAMs as shown. Chip selection is on the basis of $\overline{CS}$ signal while R/$\overline{W}$ signal governs of whether reading (from memory) or writing (into the memory) is to be done.

Once a particular address has been selected (by $AB_3 - AB_0$ lines), $DB_7 - DB_4$ stores the upper nibble of data in the left RAM (RAM-1) while $DB_3 - DB_0$ stores the lower nibble of the data in the right RAM (RAM-2).

Que: Explain briefly Memory interfacing to 8085 using an example. Support by Donating

Ans: Memory interfacing to 8085

https://alllabexperiments.com

8085 has 16 bit address bus; hence it can access 2^{16} no. of memory locations, which is equal to 64KB memory. For any microprocessor memory is required to store program as well as data. Since microprocessor doesn't have on-chip memory, we need to connect it externally. So it requires addressing mechanism. The following are the steps involved in interfacing memory with 8085 processor.

1. First decide the size of memory requires to be interfaced. Depending on this we can say how many address lines are required for it. For example if you want to interface 4KB (2^{12}) memory it requires 12 address lines. Remaining address lines can be used in address decoding.
2. Depending on the size of memory required and given address range, construct address decoding circuitry. This address decoding circuitry can be implemented with NAND gates and/or decoders or using PAL (when board size is a constraint).
3. Connect data bus of memory to processor data bus.
4. Generate the control signals required for memory using IO/M', WR', RD' signals of 8085 processor.

Example:

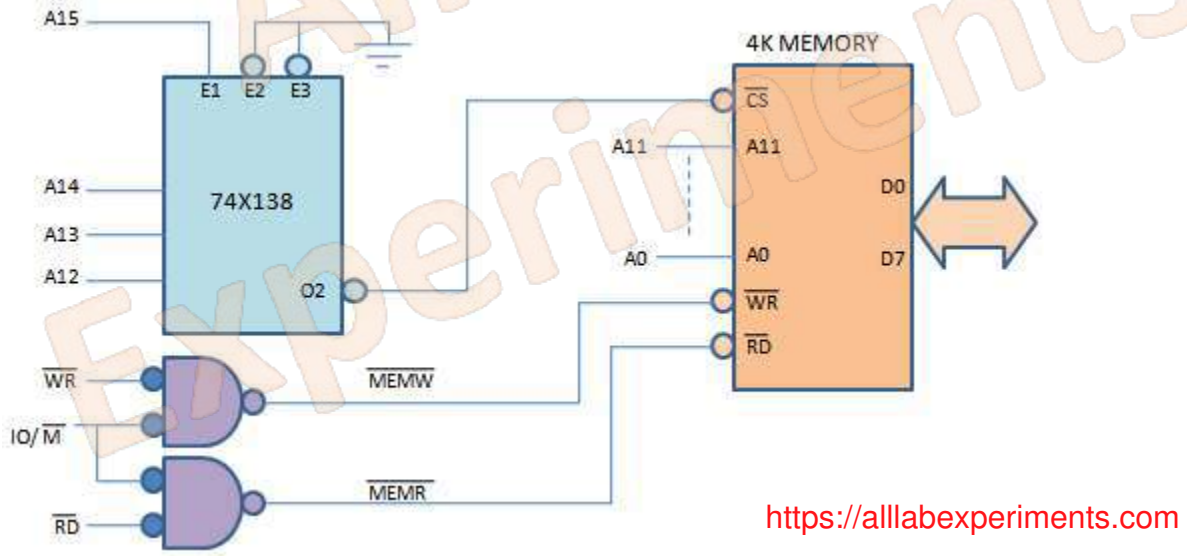
Interface 4KB memory to 8085 with starting address A000H.

1. 4KB memory requires 12 address lines for addressing as already mentioned. But 8085 has 16 address lines. Hence four of address lines are used for address decoding
2. Given that starting address for memory is A000H. So for 4KB memory ending address becomes A000H+0FFFH (4KB) = AFFFH.

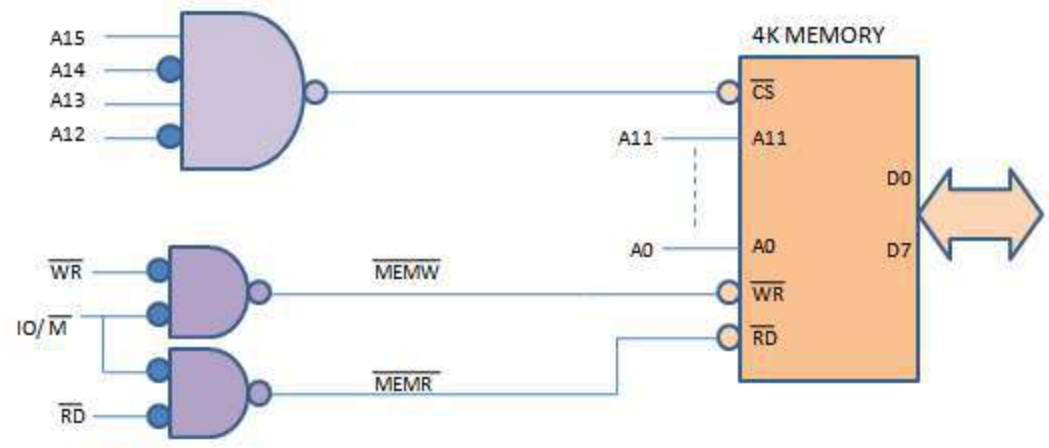
A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1

A0-A11 address lines are directly connected to address bus of memory chip. A12-A15 are used for generating chip select signal for memory chip.

Address decoding circuit using 3X8 decoder



Address decoding circuit using only NAND gates:



A15, A14, A13, A12 inputs should be 1010, for enabling the chip. So the circuit for this is as shown.

Chapter – 12

Intel 8085 Microprocessor Architecture

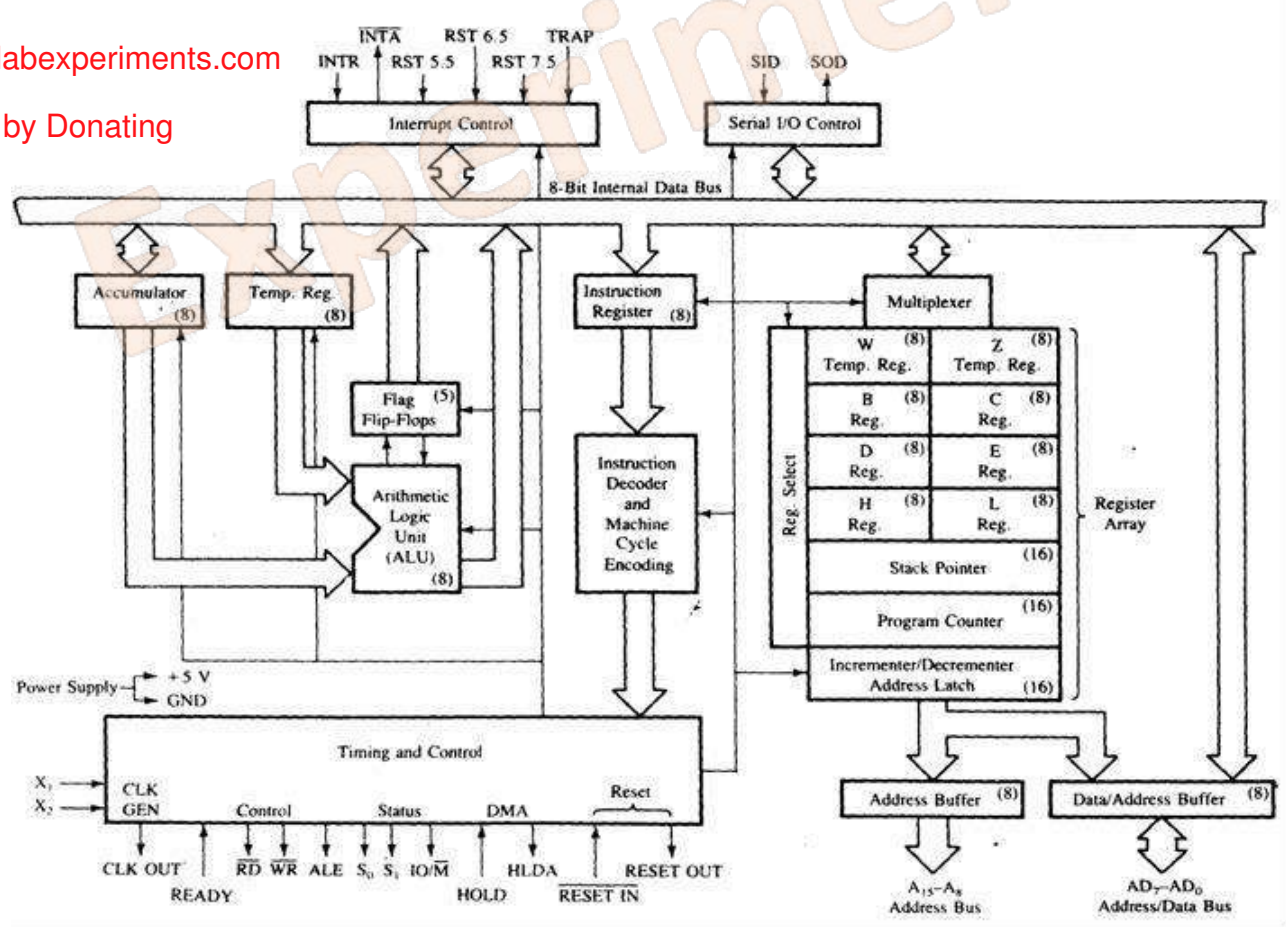
Intel 8085 Microprocessor Architecture: Main features of 8085. Block diagram. Components. Pin-out diagram. Buses. Registers. ALU. Memory. Stack memory. Timing and Control circuitry. Timing states. Instruction cycle, Timing diagram of MOV and MVI. **(9 Lectures)**

Que1. Define microprocessors?

Ans: A semiconductor device (integrated circuit) manufactured by using the LSI technique. It includes the ALU, register arrays, and control circuits on a single chip.

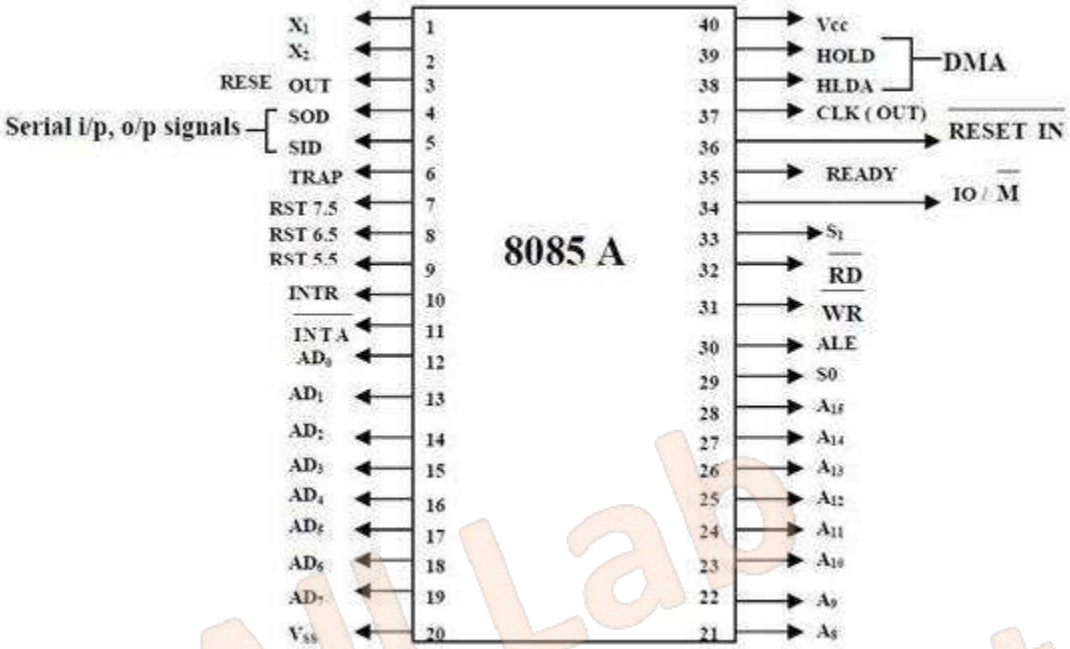
Que2: Draw the Block diagram of intel 8085 microprocessor.

Ans:



Que3: Draw the pin out diagram of intel 8085 microprocessor.

Ans:



Que 4. What is meant by the statement that 8085 is a 8-bit microprocessor?
Ans. A microprocessor which has n data lines is called an n-bit microprocessor i.e., the width of the data bus determines the size of the microprocessor. Hence, an 8-bit microprocessor like 8085 can handle 8-bits of data at a time.

Que5. What is the operating frequency of 8085?
Ans. 8085 operates at a frequency of 3 MHz, and the minimum frequency of operation is 500 kHz. The version 8085 A-2 operates at a maximum frequency of 5 MHz.

Que6. What is the purpose of CLK signal of 8085?
Ans. The CLK (out) signal obtained from pin 37 of 8085 is used for synchronizing external devices.

Que7. The address capability of 8085 is 64 KB. Explain.
Ans. Microprocessor 8085 communicates via its address bus of 2-bytes width – the lower byte $AD_0 – AD_7$ (pins 12-19) and upper byte $D_8 – D_{15}$ (pins 21–28). Thus it can address a maximum of 2^{16} different address locations. Again each address (memory location) can hold 1 byte of data/instruction. Hence the maximum address capability of 8085 is
$= 2^{16} \times 1$ Byte
= 65, 536 × 1 Byte
= 64 KB (where 1 K = 1024 bytes)

Que8: What jobs ALU of 8085 can perform?

Ans. The Arithmetic Logic Unit (ALU) of 8085 can perform the following jobs:

z 8-bit binary addition with or without carry.
z 16-bit binary addition.
z 2-digit BCD addition.
z 8-bit binary subtraction with or without borrow.
z 8-bit logical OR, AND, EXOR, complement (NOT function).
z bit shift operation.

Que9:. How many hardware interrupts 8085 supports?

Ans. It supports five (5) hardware interrupts—TRAP, RST 7.5, RST 6.5, RST 5.5 and INTR.

Que10: List the various registers of 8085 micro-processor.

Ans:

The various registers of 8085, their respective quantities and capacities are tabulated below:

Table 2.1: List of Various Registers in 8085

S. No.	Name of the Register	Quantity	Capacity
1.	Accumulator (or) Register A	1	8-bit
2.	Temporary register	1	8-bit
3.	General purpose registers (B, C, D, E, H and L)	6	8-bit each
4.	Stack pointer (SP)	1	16-bit
5.	Program counter (PC)	1	16-bit
6.	Instruction register	1	8-bit
7.	Incrementer/Decrementer address latch	1	16-bit
8.	Status flags register	1	8-bit

Que11: Describe the accumulator register of 8085.

Ans. This 8-bit register is the most important one amongst all the registers of 8085. Any data input/output to/from the microprocessor takes place via the accumulator (register). It is generally used for temporary storage of data and for the placement of final result of arithmetic/logical operations. Accumulator (ACC or A) register is extensively used for arithmetic, logical, store and rotate operations.

Que12: What are the temporary registers of 8085?

Ans. The temporary registers of 8085 are temporary data register and W and Z registers. These registers are not available to the programmer, but 8085 uses them internally to hold

temporary data during execution of some instructions.

Que13: Discuss the two registers program counter and stack pointer

Ans. Program counter (PC) is a sixteen bit register which contains the address of the instruction to be executed just next. PC acts as a address pointer (also known as memory pointer) to the next instruction. As the processor executes instructions one after another, the PC is incremented—the number by which the PC increments depends on the nature of the instruction. For example, for a 1-byte instruction, PC is incremented by one, while for a 3-byte instruction, the processor increments PC by three address locations.

Stack pointer (SP) is a sixteen bit register which points to the 'stack'. The stack is an area in the R/W memory where temporary data or return addresses (in cases of subroutine CALL) are stored. Stack is a auto-decrement facility provided in the system. The stack top is initialised by the SP by using the instruction LXI SP, memory address.

In the memory map, the program should be written at one end and stack should be initialised at the other end of the map—this is done to avoid crashing of program. If sufficient gap is not maintained between program memory location and stack, then when the stack gets filled up by PUSH or subroutine calls, the stack top may run into the memory area where program has been written. This is shown in Fig. 2.5.

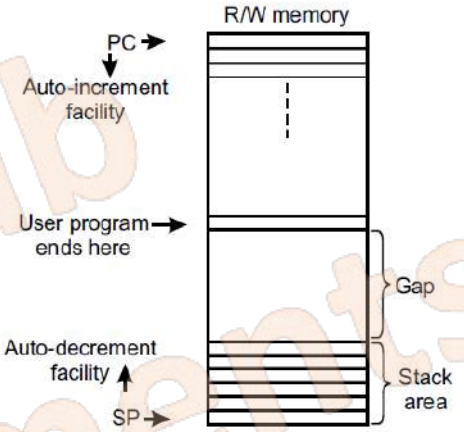


Fig. 2.5: Auto-increment and auto-decrement facility for PC and SP respectively

Que14: Describe the (status) flag register of 8085.

Ans. It is an 8-bit register in which five bit positions contain the status of five condition flags which are Zero (Z), Sign (S), Carry (CY), Parity (P) and Auxiliary carry (AC). Each of these five flags is a 1 bit F/F. The flag register format is shown in Fig. 2.6:

D7	D6	D5	D4	D3	D2	D1	D0
S	Z	X	AC	X	P	X	CY

Fig. 2.6: The flag register format

- *Sign (S) flag:* – If the MSB of the result of an operation is 1, this flag is set, otherwise it is reset.
- *Zero (Z) flag:*– If the result of an instruction is zero, this flag is set, otherwise reset.
- *Auxiliary Carry (AC) flag:*– If there is a carry out of bit 3 and into bit 4 resulting from the execution of an arithmetic operation, it is set otherwise reset.

  This flag is used for BCD operation and is not available to the programmer to change the sequence of an instruction.
- *Carry (CY) flag:*– If an instruction results in a carry (for addition operation) or borrow (for subtraction or comparison) out of bit D_7, then this flag is set, otherwise reset.
- *Parity (P) flag:*– This flag is set when the result of an operation contains an even number of 1's and is reset otherwise.

Que15: What is the function of the internal data bus?

Ans. The width of the internal data bus is 8-bit and carries instructions/data between the CPU registers. This is totally separate from the external data bus which is connected to memory chips, I/O, etc.
The internal and external data bus are connected together by a logic called a bidirectional bus (transreceiver).

Que16: Describe in brief the timing and control circuitry of 8085.

Ans. The T&C section is a part of CPU and generates timing and control signals for execution of instructions. This section includes Clock signals, Control signals, Status signals, DMA signals as also the Reset section. This section controls fetching and decoding operations. It also generates appropriate control signals for instruction execution as also the signals required to interface external devices.

Que17: Mention the following:
(*a*) **Control and Status signals**
(*b*) **Interrupt signals**
(*c*) **Serial I/O signals**
(*d*) **DMA signals**
(*e*) **Reset signals.**

Ans.

The control and status signals are ALE, $\overline{RD}$, $\overline{WR}$, IO/$\overline{M}$, S_0, S_1 and READY.

The interrupt signals are TRAP, RST 7.5, RST 6.5, RST 5.5, INTR. $\overline{INTA}$ is an interrupt acknowledgement signal indicating that the processor has acknowledged an INTR interrupt.

Serial I/O signals are SID and SOD
DMA signals are HOLD and HLDA
Reset signals are $\overline{RESET\ IN}$ and RESET OUT.

Que18: Define machine cycle?

Ans: Machine cycle is defined, as the time required completing one operation of accessing memory, I/O, or acknowledging an external request.

Que19: Define instruction cycle?

Ans: Instruction cycle is defined, as the time required completing the execution of the instruction.

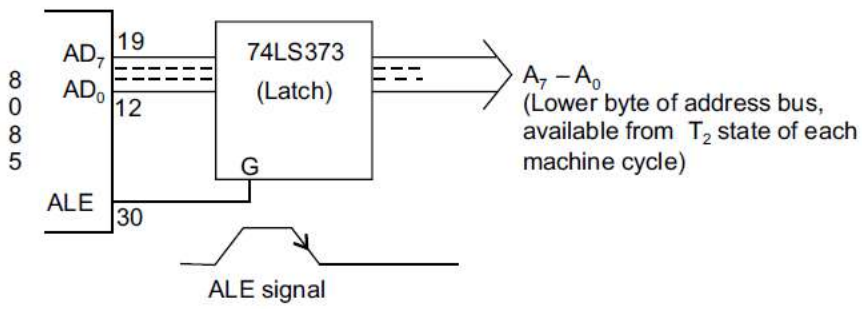
Que20: What is the function of ALE and how does it function

Pin 30 of 8085 is the ALE pin which stands for 'Address Latch Enable'. ALE signal is used to demultiplex the lower order address bus ($AD_0 - AD_7$).

Pins 12 to 19 of 8085 are $AD_0 - AD_7$ which is the multiplexed address-data bus. Multiplexing is done to reduce the number of pins of 8085.

Lower byte of address ($A_0 - A_7$) are available from $AD_0 - AD_7$ (pins 12 to 19) during T_1 of machine cycle. But the lower byte of address ($A_0 - A_7$), along with the upper byte $A_8 - A_{15}$ (pins 21 to 28) must be available during T_2 and rest of the machine cycle to access memory location or I/O ports.

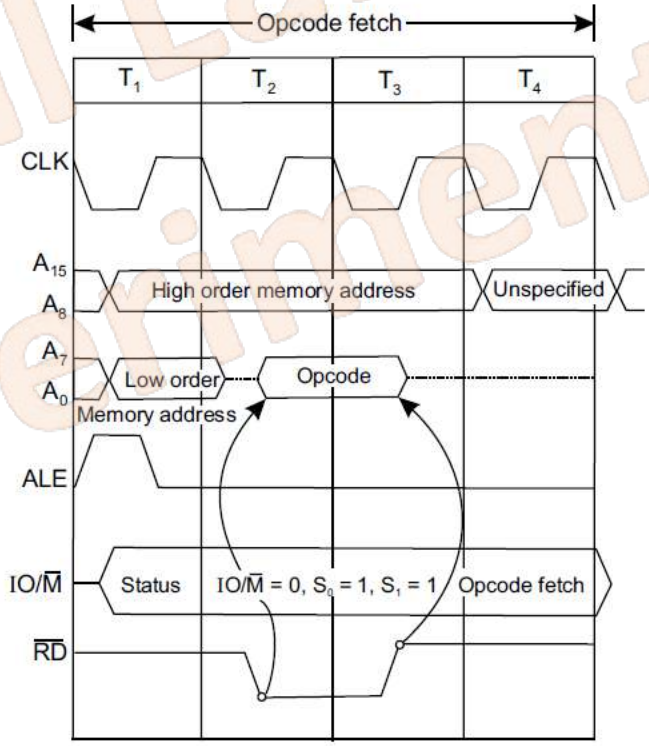
Now ALE signal goes high at the beginning of T_1 of each machine cycle and goes low at the end of T_1 and remains low during the rest of the machine cycle. This high to low transition of ALE signal at the end of T_1 is used to latch the lower order address byte ($A_0 - A_7$) by the latch IC 74LS373, so that the lower byte $A_0 - A_7$ is continued to be available till the end of the machine cycle. The situation is explained in the following figure:



Que21: Draw the Opcode Fetch machine cycle of 8085 and discuss.

Ans:

The first machine cycle of every instruction is the Opcode Fetch. This indicates the kind of instruction to be executed by the system. The length of this machine cycle varies between 4T to 6T states—it depends on the type of instruction. In this, the processor places the contents of the PC on the address lines, identifies the nature of machine cycle (by IO/$\overline{M}$, S_0, S_1) and activates the ALE signal. All these occur in T_1 state.



In T_2 state, $\overline{RD}$ signal is activated so that the identified memory location is read from and places the content on the data bus ($D_0 - D_7$).

In T_3, data on the data bus is put into the instruction register (IR) and also raises the $\overline{RD}$ signal thereby disabling the memory.

In T_4, the processor takes the decision, on the basis of decoding the IR, whether to enter into T_5 and T_6 or to enter T_1 of the next machine cycle.

One byte instructions that operate on eight bit data are executed in T_4. Examples are ADD B, MOV C, B, RRC, DCR C, etc.

Que:22 Explain a typical fetch cycle (FC).

Ans. The time required to fetch an opcode from a memory location is called Fetch Cycle.

A typical FC may consist of 3T states. In the first T-state, the memory address,

residing in the PC, is sent to the memory. The content of the addressed memory (i.e., the opcode residing in that memory location) is read in the second T-state, while in the third T-state this opcode is sent via the data bus to the instruction register (IR). For slow memories, it may take more time in which case the processor goes into 'wait cycles'. Most microprocessors have provision of wait cycles to cope with slow memories.
A typical FC may look like the following: