

Peripheral and its characteristics

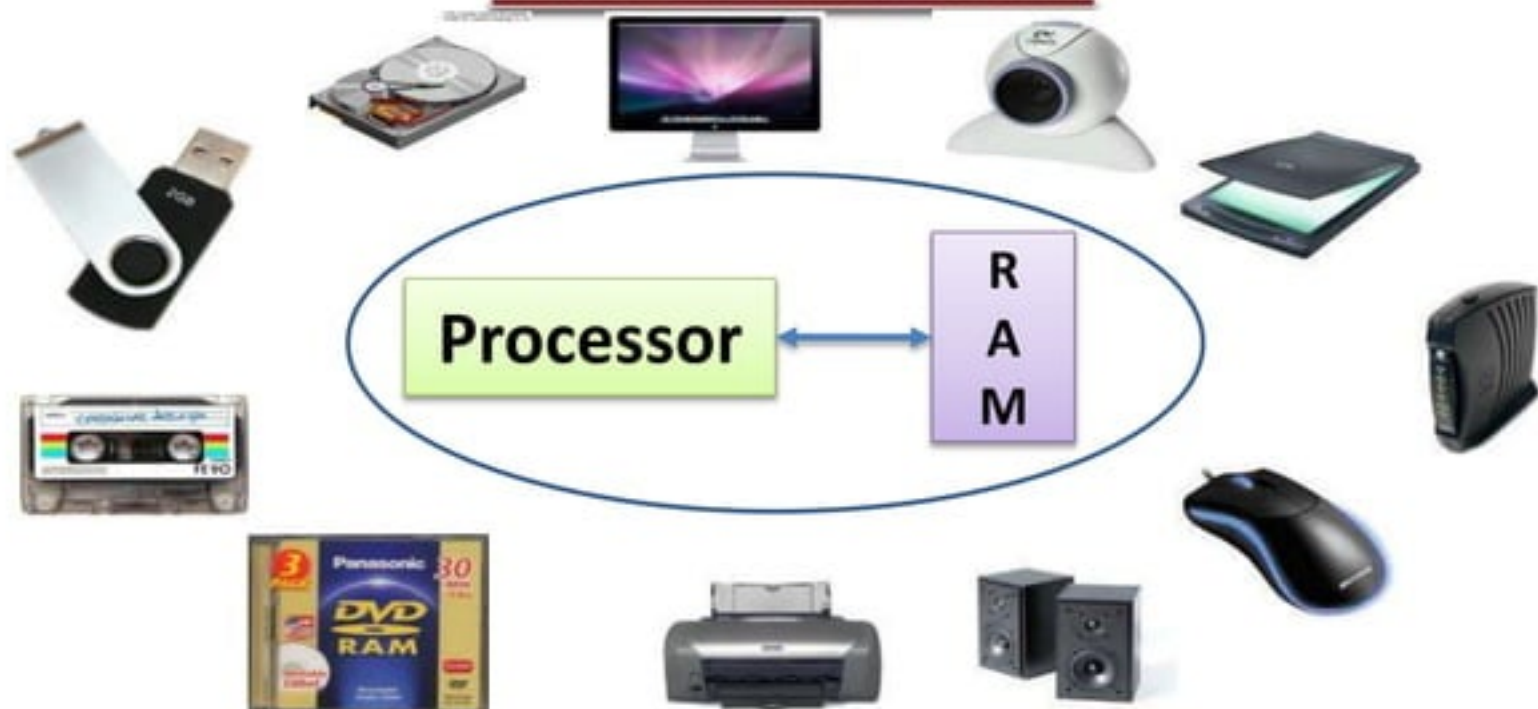
Dr A Sahu

Dept of Computer Science &
Engineering
IIT Guwahati

Outline

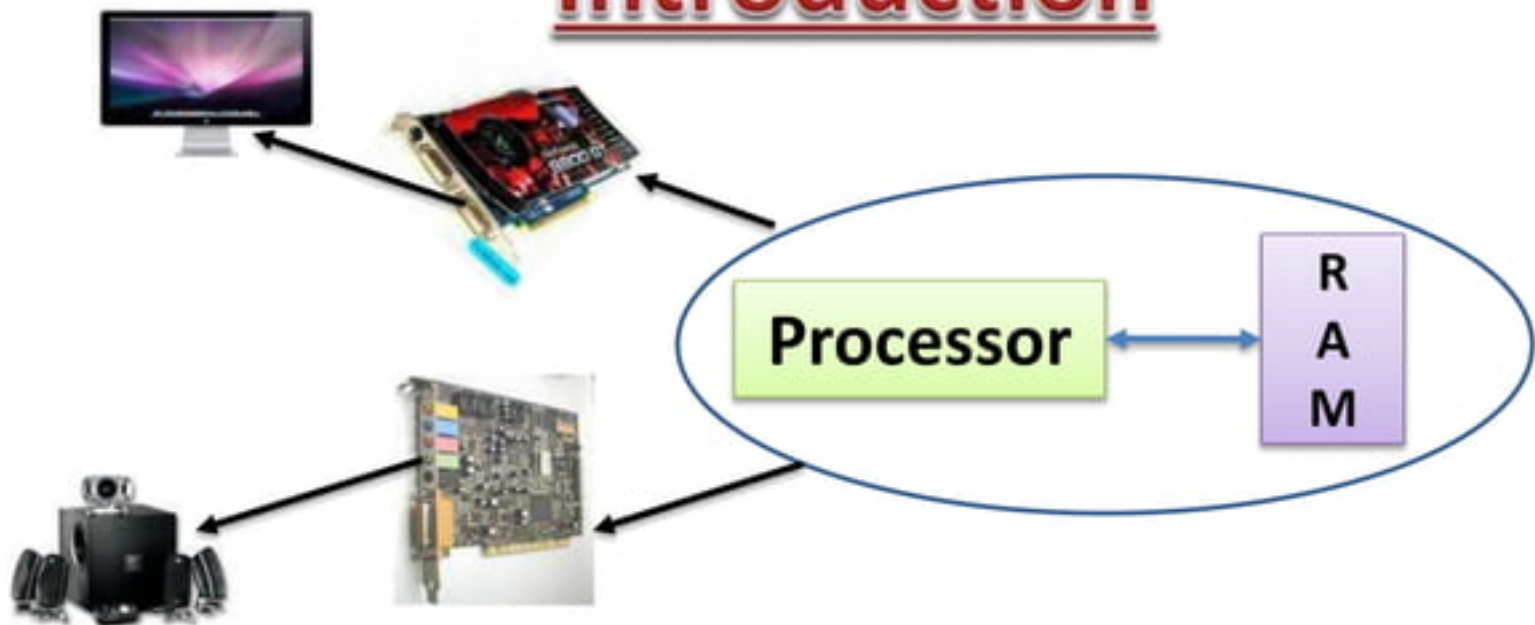
- Introduction to peripheral
- Non peripheral but outside MPU
 - Memory (RAM)
- Type of peripheral (I/O)
- Characteristics of peripheral (I/O)
- Method of getting/sending data from/to I/O
- Interrupt & ISR
- Peripheral controller (DMA/8255A)

Introduction



- Computer Systems
 - Internal (processor + memory (RAM))
 - Peripheral (Disk, Display, Audio, Eth,..)

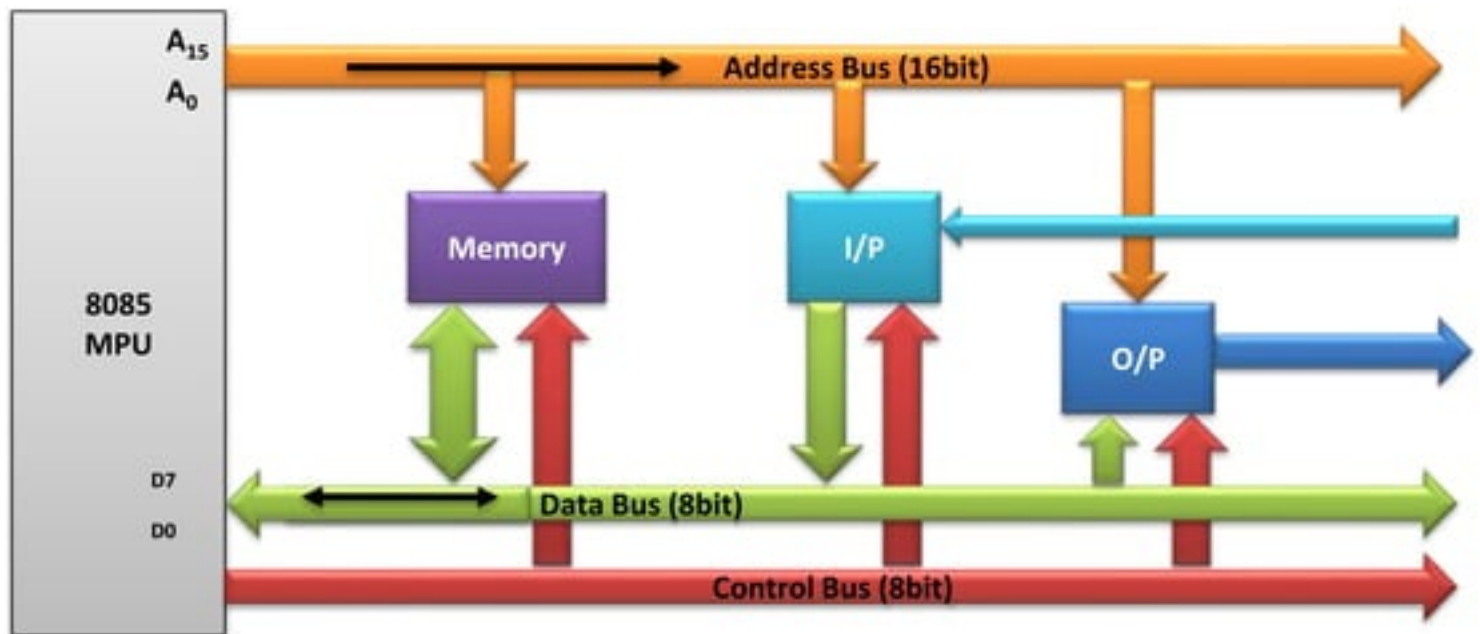
Introduction



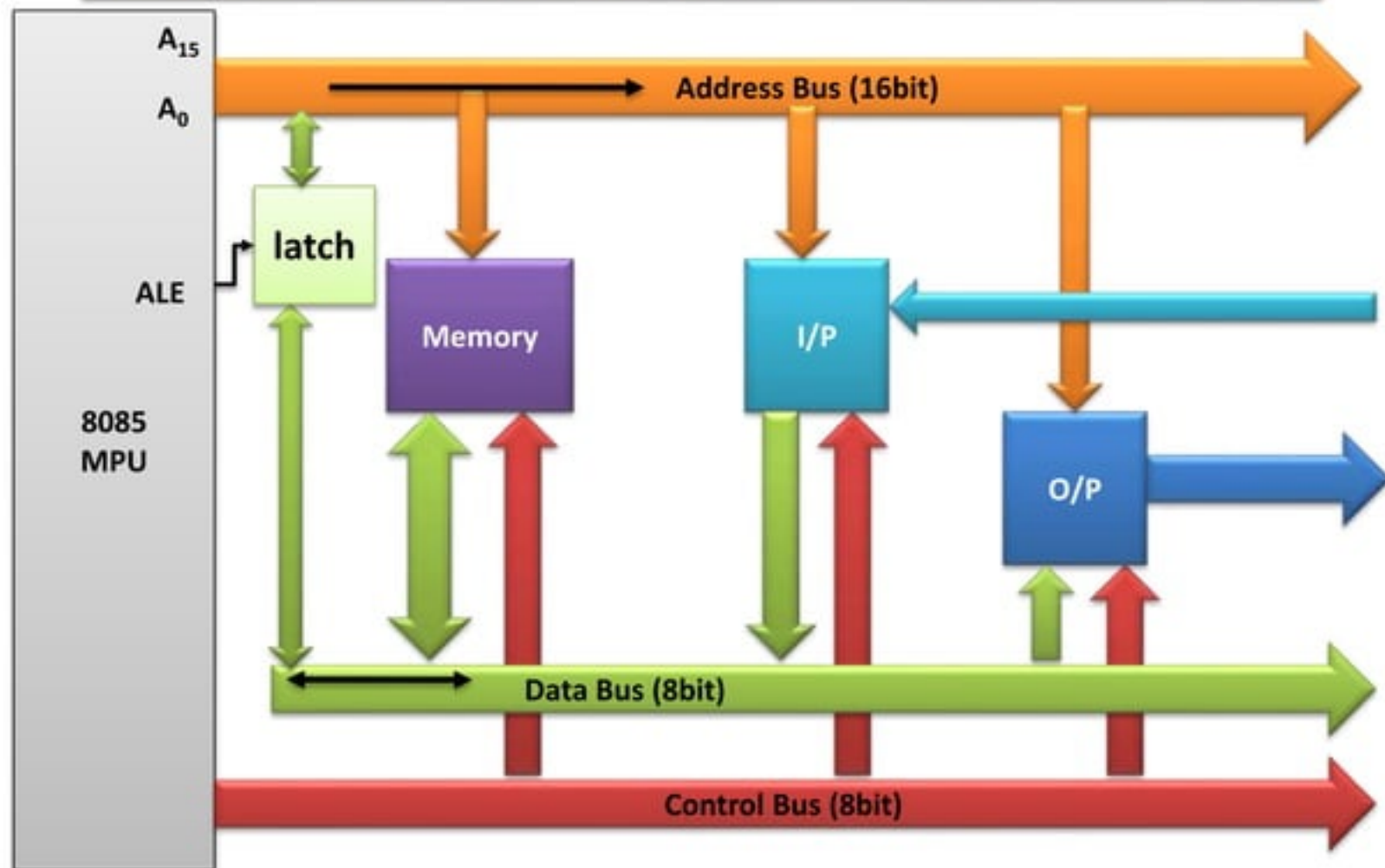
- Peripherals : HD monitor, 5.1 speaker
- Interfaces : Intermediate Hardware
 - Nvidia GPU card, Creative Sound Blaster card
- Interfaces : Intermediate Software/Program
 - Nvidia GPU driver , Sound Blaster Driver software

Non Peripheral But Outside MPU

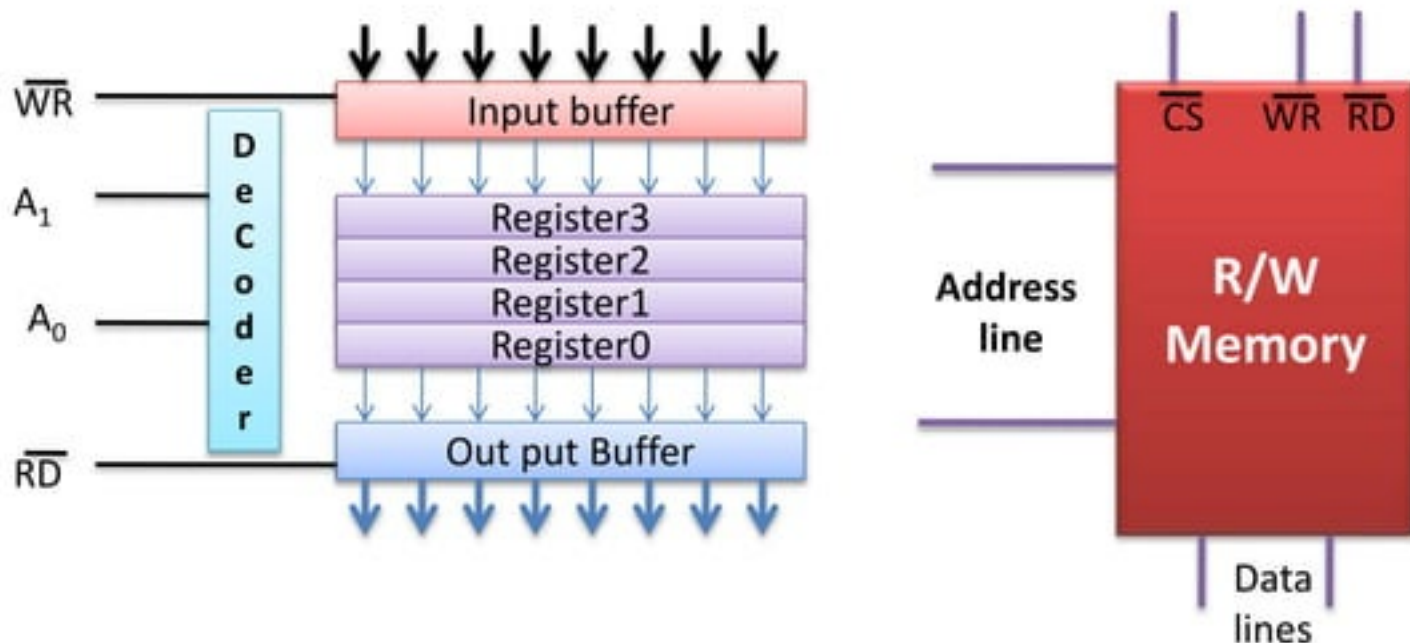
- RAM Memory is integral part of MP System
 - MPU fetch instruction from RAM
 - MPU RD and WR data to RAM (same speed as MPU)
- How Ram is interfaced



Lower 8 bit Address bus is Data bus

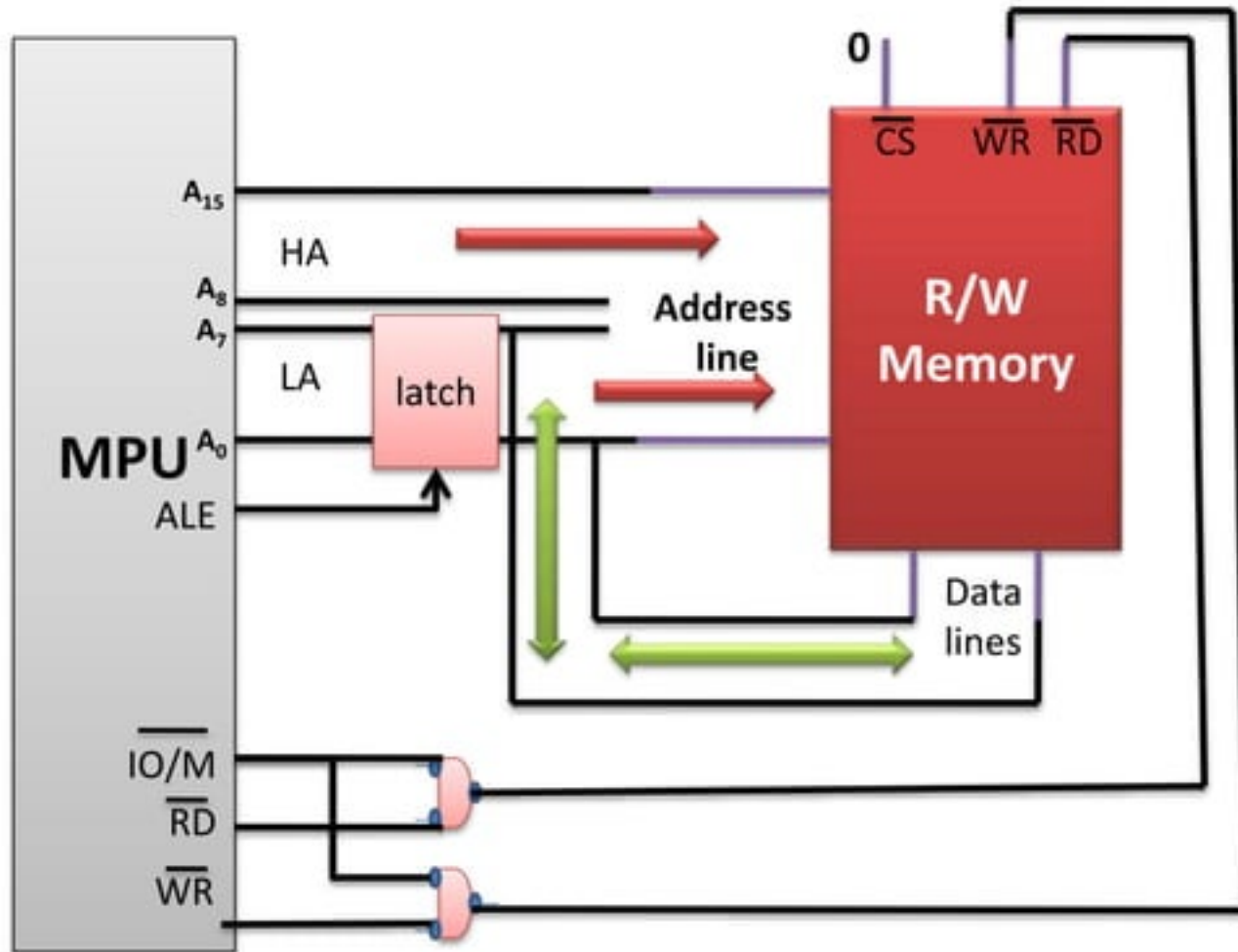


Basic Memory Elements



- \overline{WR} enable memory input buffer
- \overline{RD} enable memory output buffer

Memory Interfacing with 8085



Primary function of MPU

- Read Instruction from memory
- Execute instruction
- Read/Write data to memory
- Some time send result to output device
 - LEDs, Monitor, Printer
- Interfacing a peripheral
 - Why: To enable MPU to communicate with I/O
 - Designing logic circuit H/W for a I/O
 - Writing instruction (S/W)

Format of communications

- Synchronous : At the same time, high speed I/O
 - Transmitter & Receiver Synchronized with same clock
 - 7 Segment LEDs can work same/higher speed then MPU
 - RAM (may be I/O) can work at same speed as MPU (Not the current Processor & DRAM)
- Asynchronous : Irregular interval, low speed I/O
 - I/O are slower
 - Keyboard, ADC/DAC, Disk

Type of I/O

- Peripheral I/O
 - IN port (Instruction), OUT port (instruction)
 - Identified with 8 bit address (Immediate)
 - Example: IN 01H ; Receive data from port 1
- Memory mapped I/O
 - A peripheral is connected as if it were a memory location
 - Identified with 16 bit address
 - Data transfer by : LDA, STA, MOV M R, MOV R M

Mode of Data transfer

- Parallel
 - Entire 8bit or 16 bit transfer at one time
 - In 8085 entire 8 bit transferred simultaneous using 8 data lines
 - Seven Segment LEDs, Data converter (ASCIItoHEX), Memory
- Serial
 - Data transferred one bit at a time
 - Parallel to serial conversion (parallel 8 bit to stream of serial 8 bit)
 - Serial to Parallel conversion
 - Modem, USB, SATA, and (sometimes monitor/printer)
- UART: Universal Asynchronous Receiver & Transmitter

Transmission controller (low speed I/O)

- MPU control
 - You will pick up and drop friend from Station (I/O) to Home (Memory)
 - When ever you will get a call (interrupt you) from him, you have do that work of pickup (Execute ISR)
- Device Control (DMA)
 - You hire a Taxi wala, he will pick up and drop friend from Station to Home
 - Give mobile number of taxi driver to your friend, your friend will to the taxi driver (coordinate) and Taxi wala drop your friend in your home

Conditions to be satisfied for Data Transfer

- Unconditional
 - Assume I/O is always available (Ex LED port)
 - MPU simply enable port, transfer data & execute Next Instruction
- Polling (Status check of device)
 - You will call to your friend how much distance he have come (check whether he have landed at Airport)
- Interrupt (Let me work, when you are ready INTR me)
- With READY Signal (Mix of both approach)
 - Status Check, Interrupt
- With handshake signal (Mutual Understanding)
 - A/D converter, When ready send a signal to MPU, MPU keep check the DR signal
 - MPU check it own DR signal pin

Funny project : Song changer

- Design a song changer using a Bed lamp switch
- You don't want to go near to computer and change the song
- Press the button (on-off) of the switch and let the computer change to song
- Design Hardware & Software for this
 - Take USB cable, dismantle it, connect two port to switch
 - Write to C program to handle Interrupt & song change

Steps to Implement interrupt

- Set EI
- Check INTR line during execution of each instruction
- If INTR is high & EI=1 then set DI and put INTA^{bar} low
- INTA^{bar} is used to insert a RST instruction, it transfer the program control to specific location (ISR)
- Perform the task using ISR
- At end of ISR it enable interrupt (EI)
- After RETurn form ISR it continue to execute the normal execution

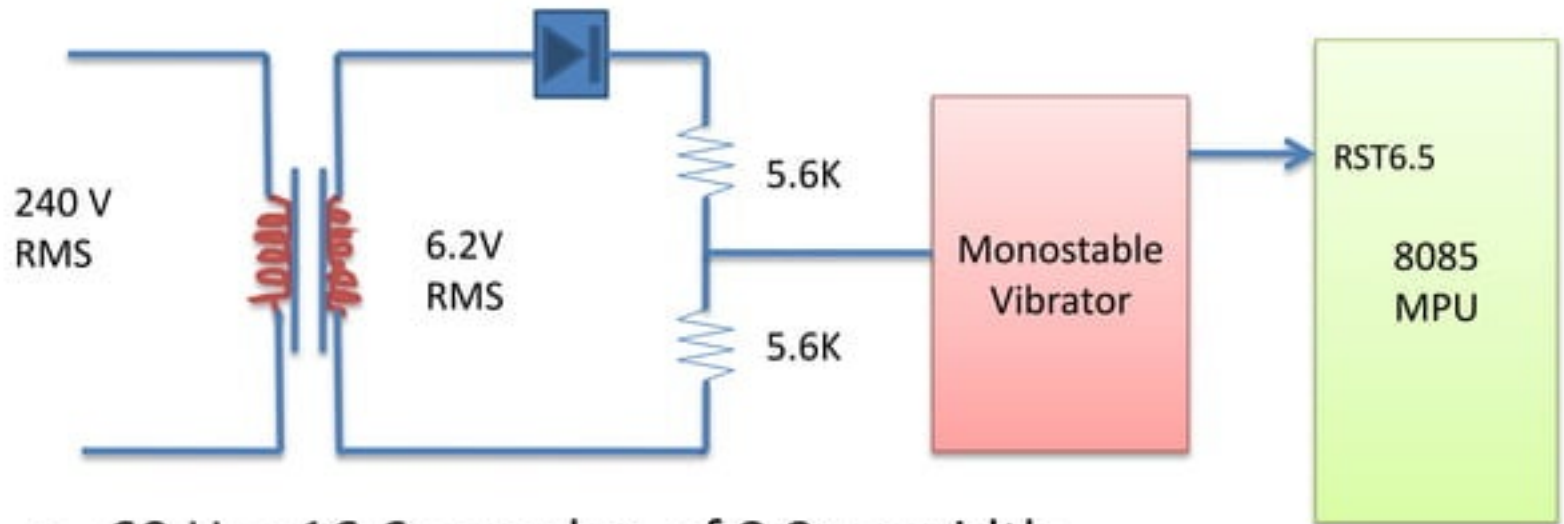
Interrupts & Priority

- Assume you are the MPU
- What is your priority to different person
 - Unknown (Some INTR), Friends (RST5), Boss (RST 6), Parents (RST7), Medical/Accident case (TRAP)
 - Higher priority make other Disable
 - TRAP (1), RST7 (2), RST6 (3), RST5 (4), INT (5)
- When you don't want any interrupt
 - When I am serving to my Boss at company
 - When I am sleeping
 - DI instruction (Disable Interrupt)
- When I want interrupt (EI= Enable Interrupt)
 - I am free
 - I am not doing any valuable thing

Design problem: Interrupt driven clock

- Design a 1-minute timer using a 60Hz Power line interrupt source
- Out put port should display minute and Seconds in Hex/BCD
- At the end port should continue displaying 1 minute and zero seconds

Schematic diagram of Interrupt driven Timer clock



- 60 Hz= 16.6ms pulse, of 8.3ms width
- Too long for interrupt
- Make it 6micro second using a constable vibrator in 16 ms

Software for drive it: Monitor program

34 JMP RWM : RST 6.5 goes to 0034 location and Jump to ISR

Main: LXI SP, STACK
RIM ; Read Mask
ORI 08H ; bit pattern to enable 6.5
SIM ; Enable RST 6.5
LXI B,0000H ; Set up B for minute & C for Sec
MVI D, 3C ; Set up register D to count 60₁₀ interrupt
EI ; Allow system to interrupt

Display:

MOV A,B
OUT PORT1 ; Display on port Minute LEDs
MOV A,C
OUT PORT2 ; Display on port Second LEDs
JMP DELAY

RWM: JMP TIMER

ISR for timer

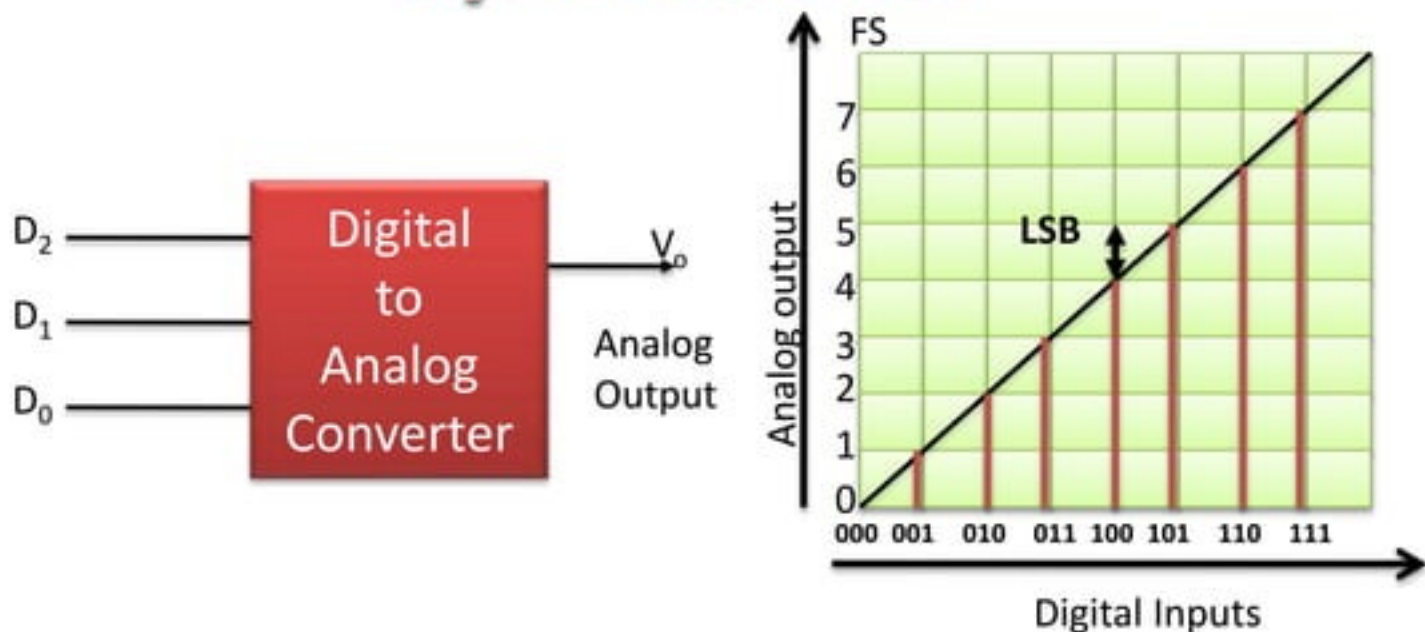
TIMER:

DCR	D	; one interrupt occur reduce count by 1
EI		; Enable Intr
RNZ		; Has 1 Sec elapsed ? If not return
DI		; No other interrupt allowed
MVI	D, 3CH	; 1 sec is complete, load D with 60 ₁₀ intr count
INR	C	; Increment second register
MOV	C,A	
CPI	3CH	; compare with 60
EI		
RNZ		; Is time=60 second? Is not return
DI		; Disable interrupt
MVI	C,00H	; 60 second complete, clear "Second" reg
INR	B	; Increment Minute
RET		; Return

Digital to Analog Converter

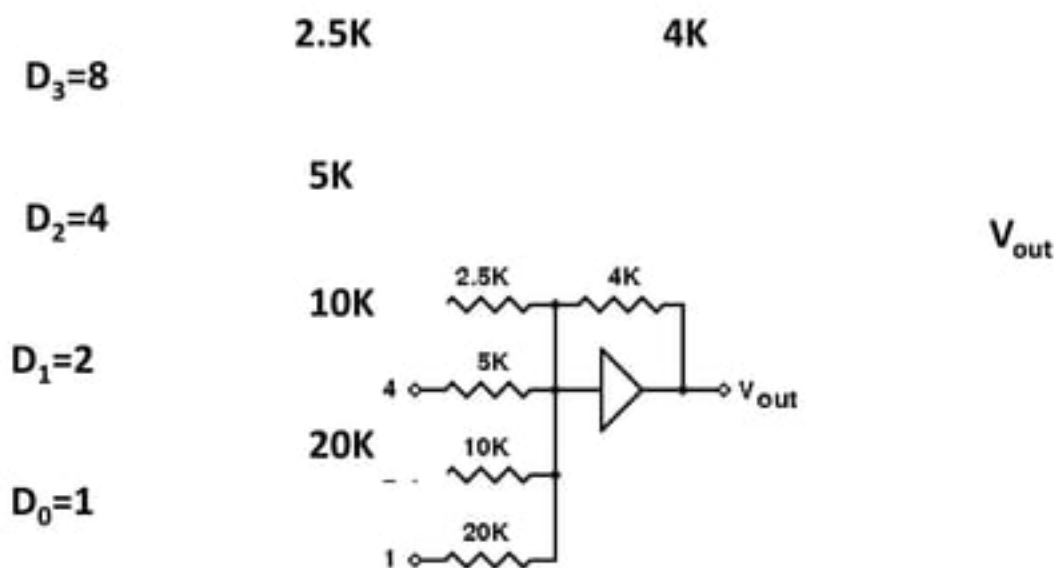
- Used for play sound in speaker
- Used by AC97 (Audio codec)
- MP3 Sound store digital format in HDD
- Slow as compared to processor/MPU
- Parameters
 - Resolution (8 bit/16 bit)
 - Settling time (1micro sec)

D/A converter



- $\text{FullScaleOutput} = (\text{FullScaleValue} - 1\text{LSBValue})$
- $1\text{MSB Value} = 1/2 * \text{FSV}$

Circuit Realization



- $V_o = V_{ref}/R * (A_1/2 + A_2/4 + \dots A_n/2^n)$
- V_o is proportional to values of Data Bits Value

Reference

- R S Gaonkar, "Microprocessor Architecture", Unit II preface, Chapter 12,13,14

Thanks