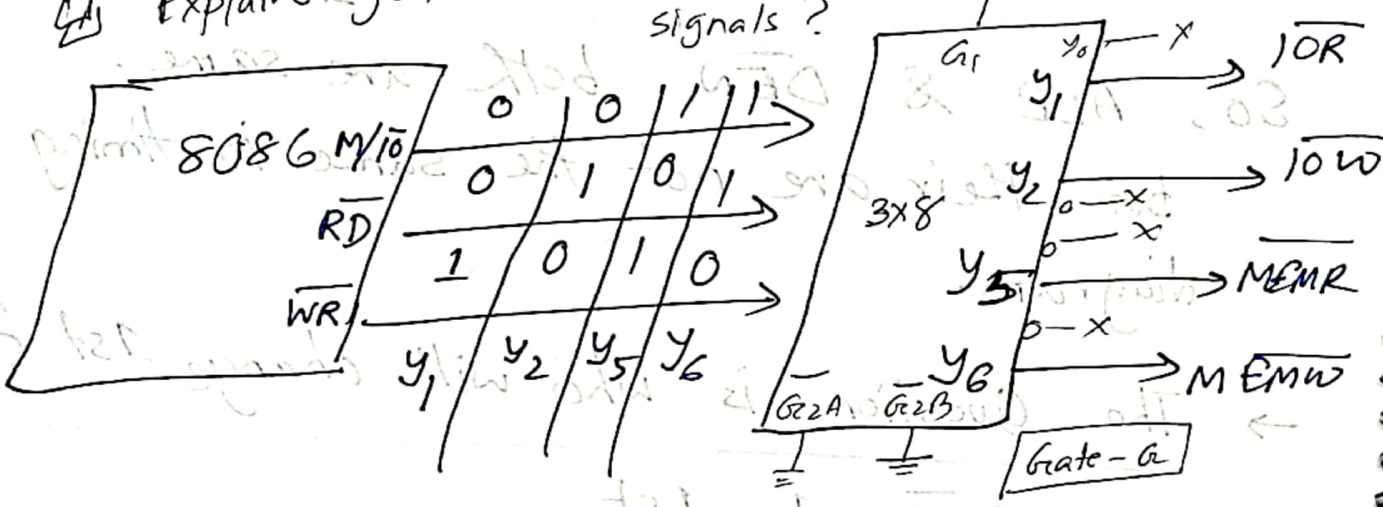
8286 → 2 ~~latches~~

8282 → 3 latches

Explain generation of control signals?



// $\overline{RD}$ → 0 1 0 1 only read operation.

// For RAM, read we should activate $\overline{MEMR}$ signal Y_5 line.

→ So, at last; here we have a add. bus, data bus & control bus together they are called system bus.

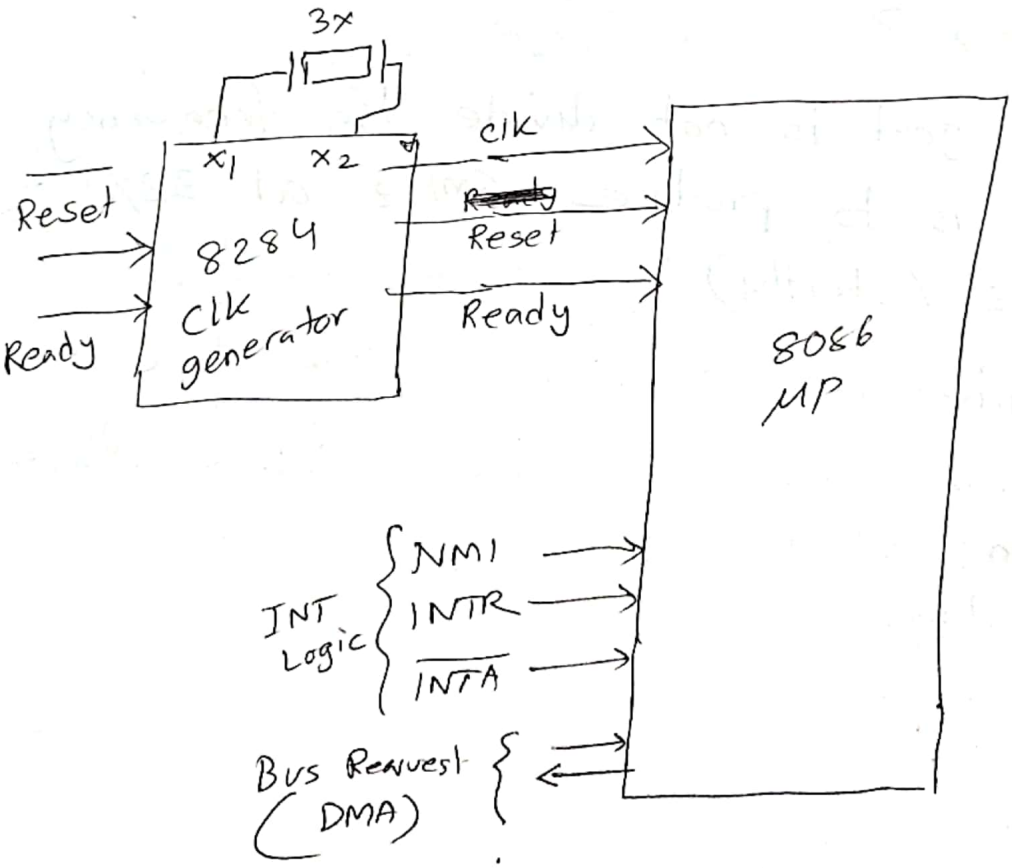
8284 clock generator

→ Any MP requires a clock.

→ Each clk pulse is a trigger to MP, to change the state or to do something new.

→ initiates a new activity in MP.

→ 8086 works at 6 MHz. (Standard value)



→ clk is connected to an oscillator.
via oscillation clock is produced.

→ 18MHz divided into 3 signals.
clk, Reset, Ready.

→ crystal oscillator is of 18 MHz gets inside & then divided into 6MHz.

→ Why we are conneeting 18 MHz instead of 6 MHz?

Ans: our goal is not divide the freauency, our goal is to produce 6MHz at 33% duty cycle (strictly)

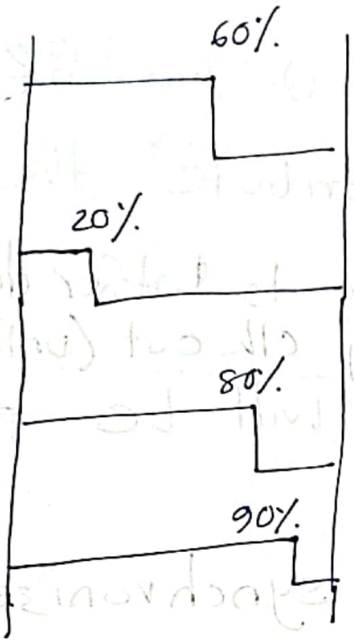
→ duty cycle is the ratio on total time

→ At what time transition takes place define it duty cycle.

→ clk pulse is on for some time
off " "

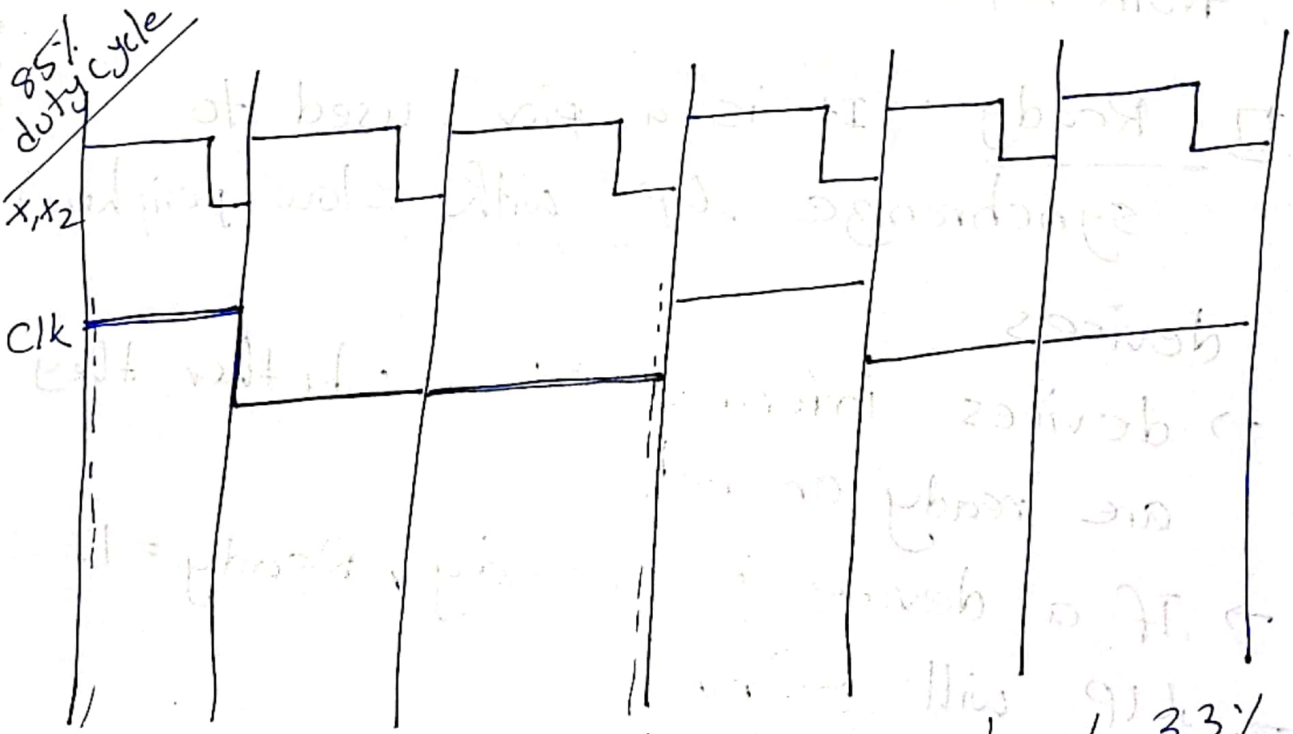
→

Various duty cycle



All of them has same time period i.e. same frequency.

Now, we will convert random (85%) cycle into 33% duty cycle.



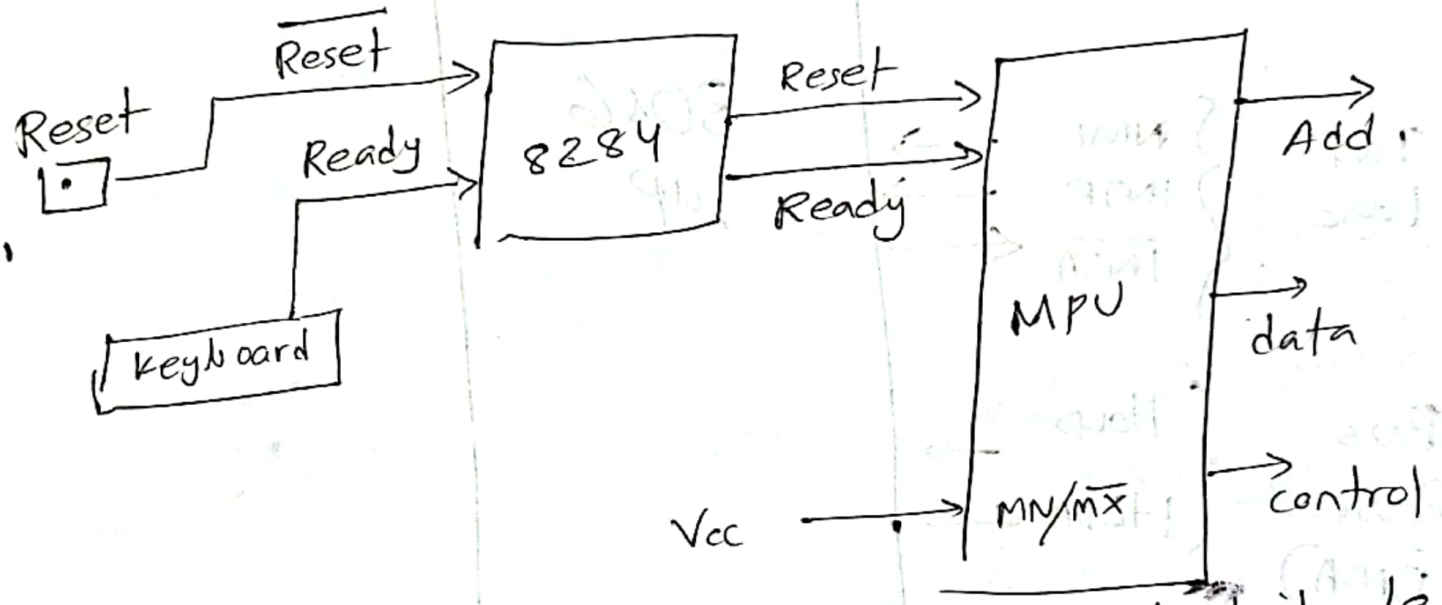
→ high for 1/3 time i.e. we have produced 33% duty cycle clk.

- Clk is given to MPU via 8284. MP does not distribute the clk pulses. As well as to other device eliminating clk out (unlike 8085)

- Reset: the signal will be applied to 8284.
  → 8284 will give synchronize signal to MP & other connected devices eliminating the "Reset out" signal from MP.

- Ready: It is a pin, used to synchronize MP with slow peripheral devices
  → devices informs MP, whether they are ready or not.
  → If a device is ready, Ready = 1.
  → MP will continue.

→ If READY pin = 0,
µp will enter into WAIT state & will keep waiting till the device is ready.



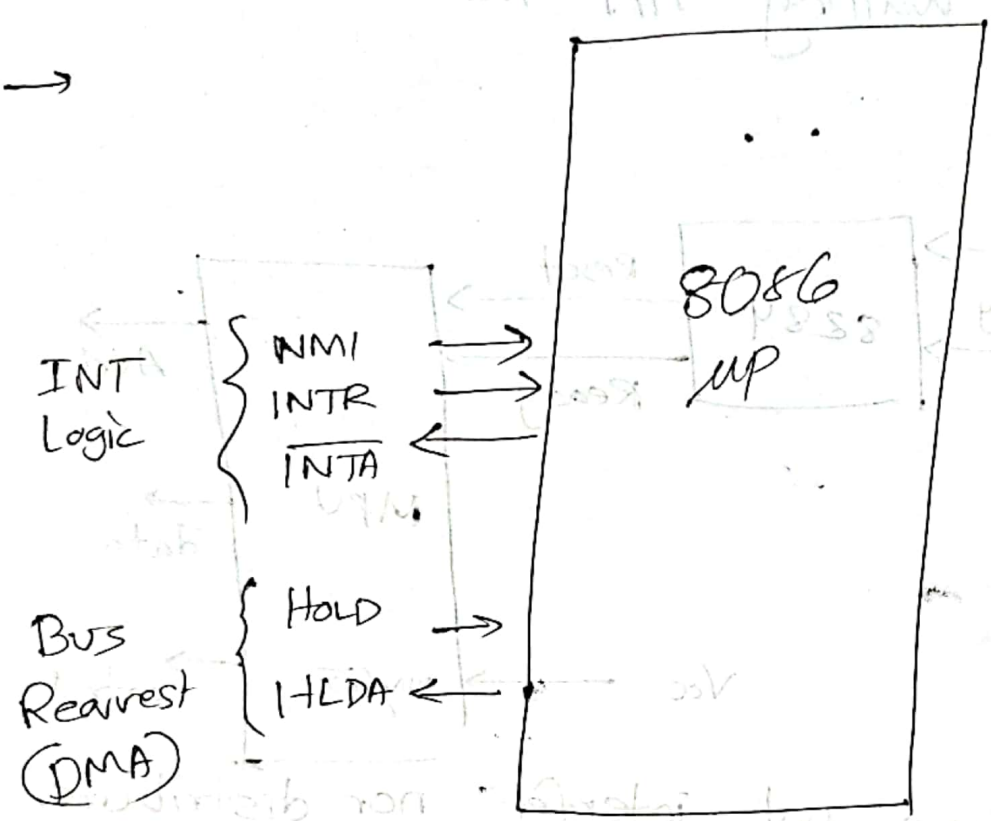
→ So, µp does not interfer nor distribute the Ready/Reset/clk signals.

→ This is the whole working procedure of Active µp in min^m mode.

→

☑ Now, there are things may disturb MP.

→



→ either INT Logic or Bus Reavest.

☑ INT Logic

Whenever 8086 gets NMI, (NMI is vector) vector num is fixed which is

2.

→ INTR is non-vector. (not fixed)
→ So, MP doesn't know what to execute.

→ So, µP will provide $\overline{INTA}$ via which µP will ask the vector number.

→ $\overline{INTA}$ is used to Ask for a vector number.

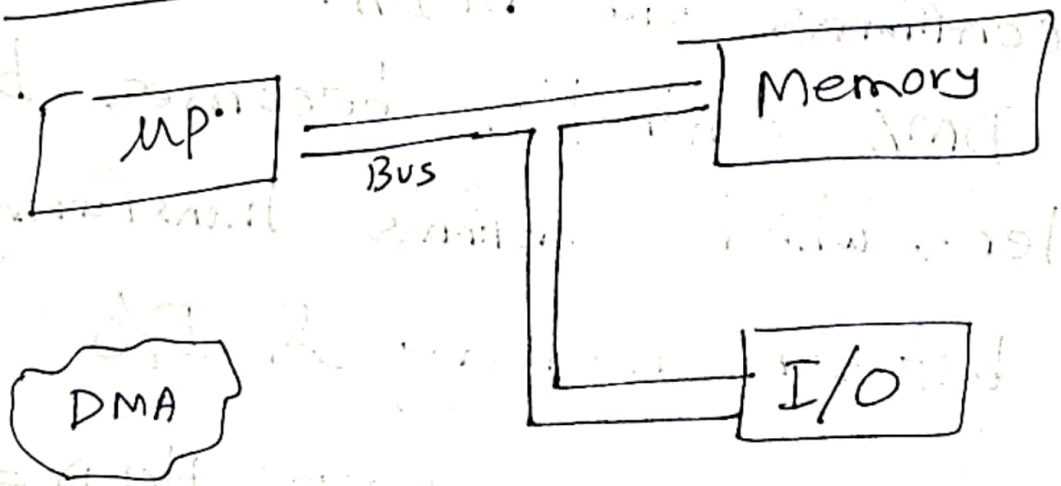
→ two $\overline{INTA}$ is given.

during 1st $\overline{INTA}$, device will calculate the vector number

during 2nd $\overline{INTA}$, device will give the vector number to µP for loading.

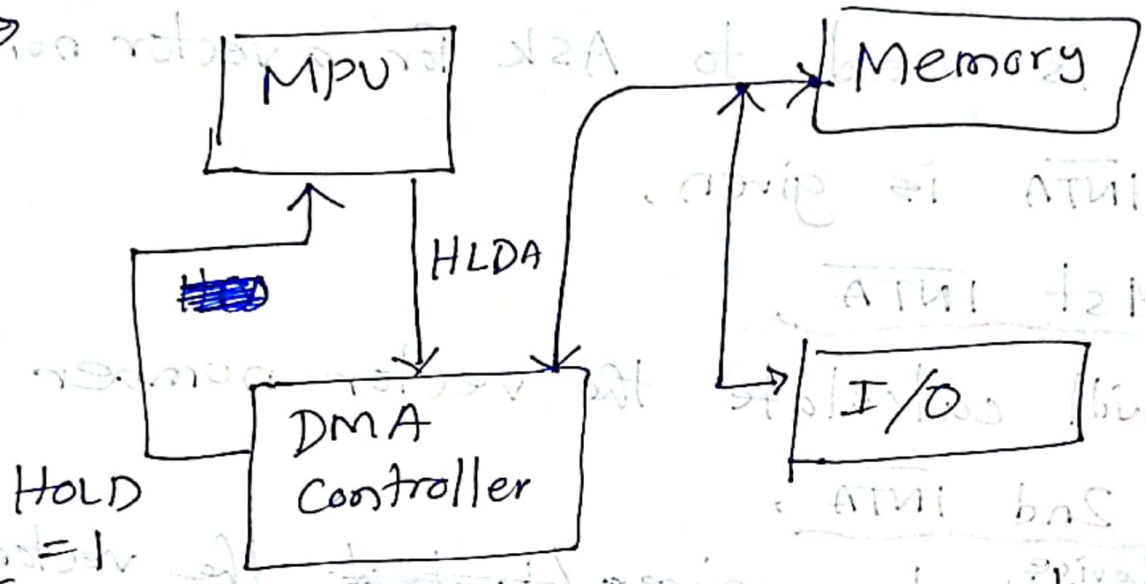
→ 2 $\overline{INTA}$ pulse is given back to back

DMA : Direct Memory access.



→ DMA controller can be bus master.

→ But by default, MP is the bus master.



↳ telling MP to hold its operation.

→ Then, MP will release the control of the system bus,

→ MP confirms via HLDA.

→ Then DMA controller becomes bus master., which confirms transferring data between memory & I/O

→ When transfer is over, HOLD = 0, then MP becomes the bus master again.