Lecture 17: 80x86 Microprocessor Architecture

Seyed-Hosein Attarzadeh-Niaki

Some slides due to Hongzi Zhu

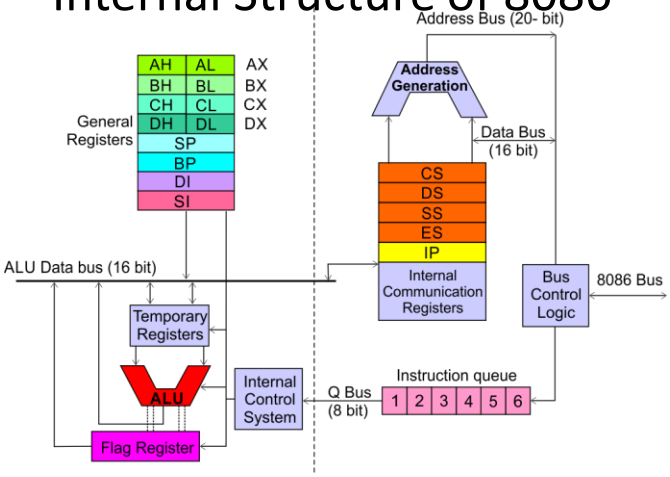
Outline

- 8086 internal structure
- 8086/88 pins, signals, and buses

Evolution of 80X86 Family

- 8086, born in 1978
  - First **16**-bit microprocessor
  - **20**-bit address data bus, i.e. 2^{20} = 1MB memory
  - First pipelined microprocessor
- 8088
  - Data bus: 16-bit internal, 8-bit external
  - Fit in the 8-bit world, e.g., motherboard, peripherals
  - Adopted in the IBM PC + MS-DOS open system
- 80286, 80386, 80486
  - Real/protected modes
  - Virtual memory

Internal Structure of 8086



Execution Unit (EU)

EU executes instructions that have already been fetched by the BIU.

BIU and EU functions separately.

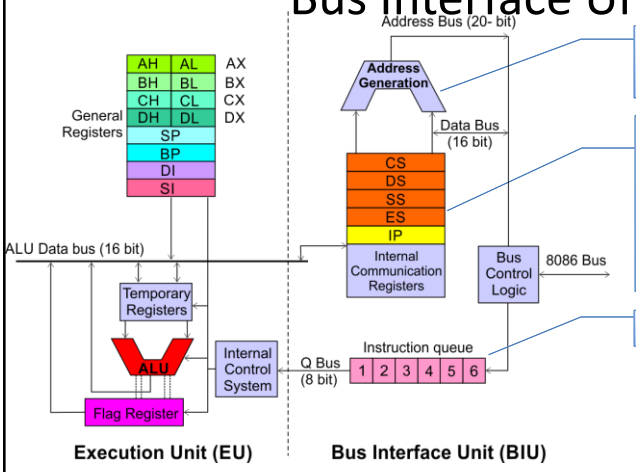
Bus Interface Unit (BIU)

BIU fetches instructions, reads data from memory and I/O ports, writes data to memory and I/ O ports.

Bus Interface Unit

- Take in charge of data transfer between CPU and memory and I/O devices as well
  - Instruction fetch, instruction queuing, operand fetch and storage, address relocation and Bus control
- Consists of :
  - Four 16-bit segment registers: CS, DS, ES, SS
  - One 16-bit instruction pointer: IP
  - One 20-bit address adder: e.g., CS left-shifted by 4 bits + IP (CS*16+IP)
  - A 6-byte instruction queue
- While the EU is executing an instruction, the BIU will fetch the next one or several instructions from the memory and put in the queue

Bus Interface Unit



Dedicated Adder to generate 20 bit address

Four 16-bit segment registers

Code Segment (CS)
Data Segment (DS)
Stack Segment (SS)
Extra Segment (ES)

Instruction queue

- **A group of First-In-First-Out (FIFO) in which up to 6 bytes of instruction code are pre fetched from the memory ahead of time.**
- **This is done in order to speed up the execution by overlapping instruction fetch with execution.**
- **This mechanism is known as pipelining.**

Execution Unit

EU decodes and executes instructions.

A decoder in the EU control system translates instructions.

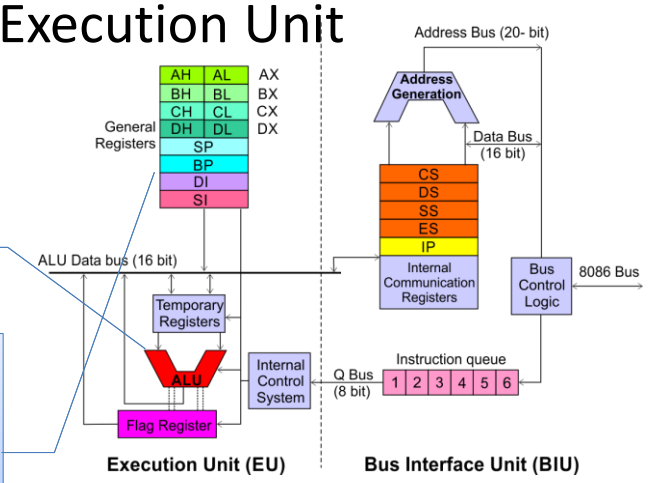
16-bit ALU for performing arithmetic and logic operation

Four general purpose registers (AX, BX, CX, DX);

Pointer registers (Stack Pointer, Base Pointer);

and

Index registers (Source Index, Destination Index) each of 16-bits



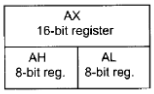
Some of the 16 bit registers can be used as two 8 bit registers as :

AX can be used as AH and AL
BX can be used as BH and BL
CX can be used as CH and CL
DX can be used as DH and DL

7

Registers

- On-chip storage: super fast & expensive
- Store information temporarily



8-bit register: D7 D6 D5 D4 D3 D2 D1 D0

16-bit register: D15 D14 D13 D12 D11 D10 D9 D8 D7 D6 D5 D4 D3 D2 D1 D0

- Six groups

Category	Bits	Register Names
General	16	AX, BX, CX, DX
	8	AH, AL, BH, BL, CH, CL, DH, DL
Pointer	16	SP (stack pointer), BP (base pointer)
Index	16	SI (source index), DI (destination index)
Segment	16	CS (code segment), DS (data segment), SS (stack segment), ES (extra segment)
Instruction	16	IP (instruction pointer)
Flag	16	FR (flag register)

Note:
The general registers can be accessed as the full 16 bits (such as AX), or as the high byte only (AH) or low byte only (AL).

Execution Unit Registers I

Accumulator Register (AX)

- The I/O instructions use the AX or AL for inputting / outputting 16 or 8 bit data to or from an I/O port.
- Multiplication and Division instructions also use the AX or AL.

Base Register (BX)

- This is the only general purpose register whose contents can be used for addressing the 8086 memory.
- All memory references utilizing this register content for addressing use DS as the default segment register.

Counter Register (CX)

- Instructions such as SHIFT, ROTATE and LOOP use the contents of CX as a counter.

Data Register (DX)

- Used to hold the high 16-bit result (data) in 16x16 multiplication or the high 16-bit dividend (data) before a 32/16 division and the 16-bit reminder after division.

Execution Unit Registers II

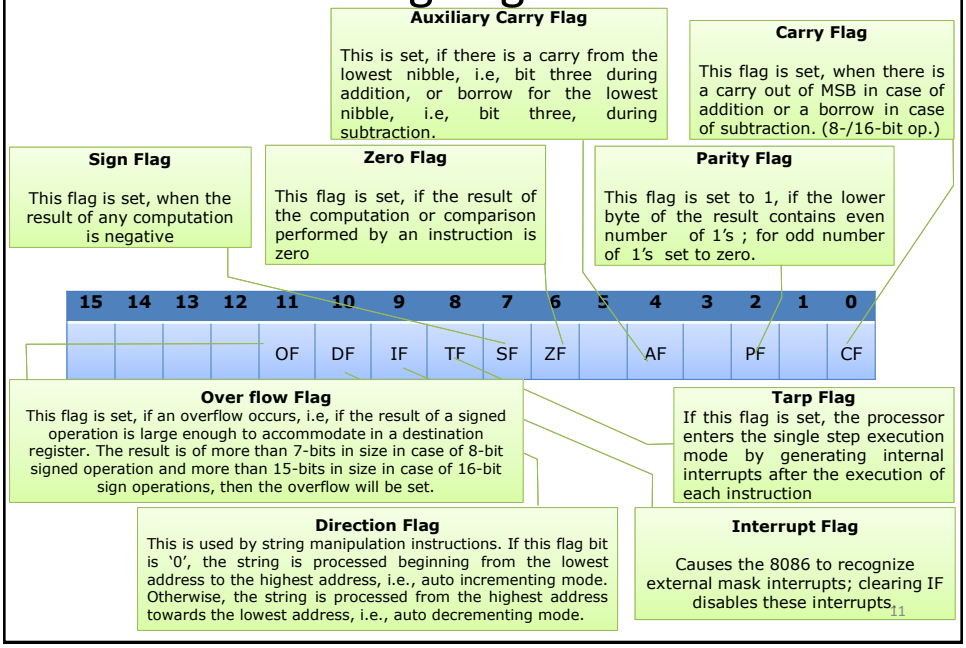
Stack Pointer (SP) and Base Pointer (BP)

- SP and BP are used to access data in the stack segment.
- SP is used as an offset from the current SS during execution of instructions that involve the stack segment in the external memory.
- SP contents are automatically updated (incremented/ decremented) due to execution of a POP or PUSH instruction.
- BP contains an offset address in the current SS, which is used by instructions utilizing the based addressing mode.

Source Index (SI) and Destination Index (DI)

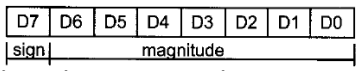
- Used in indexed addressing.
- Instructions that process data strings use the SI and DI registers together with DS and ES respectively in order to distinguish between the source and destination addresses.

Flag Register



More about Signed Numbers
CF and OF

- The most significant bit (MSB) as sign bit, the rest of bits as magnitude



  - For negative numbers, D7 is 1, but the magnitude is represented in 2's complement
- CF is used to detect errors in unsigned arithmetic operations
- OF is used to detect errors in signed arithmetic operations
  - E.g., for 8-bit ops, OF is set when there is a carry from d6 to d7 or from d7 out, but not both

Examples of Conditional Flags

```
    38      0011  1000
+   2F      0010  1111
    67      0110  0111
```

CF = 0 since there is no carry beyond d7
PF = 0 since there is an odd number of 1s in the result
AF = 1 since there is a carry from d3 to d4
ZF = 0 since the result is not zero
SF = 0 since d7 of the result is zero
OF = 0 since there is no carry from d6 to d7 and no carry beyond d7

How can CPU know whether an operation is unsigned or signed?

```
 + 96    0110 0000
 + 70    0100 0110
+166     1010 0110
```

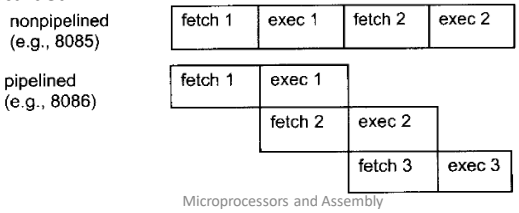
According to the CPU, this is −90, which is wrong. (OF = 1, SF = 1, CF = 0)

```
−128     1000 0000
+ − 2    1111 1110
−130     0111 1110
```

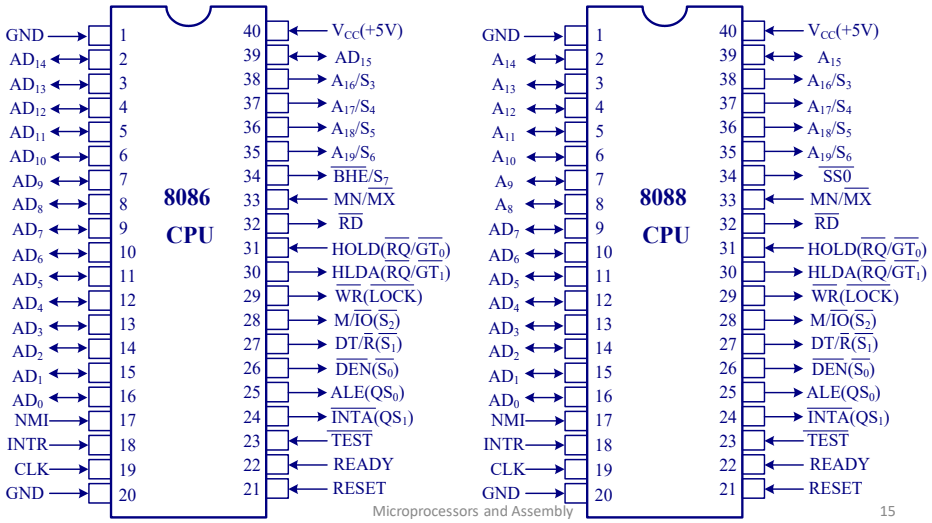
According to the CPU, the result is +126
OF=1, SF=0 (positive), CF=1

Pipelining in 8086

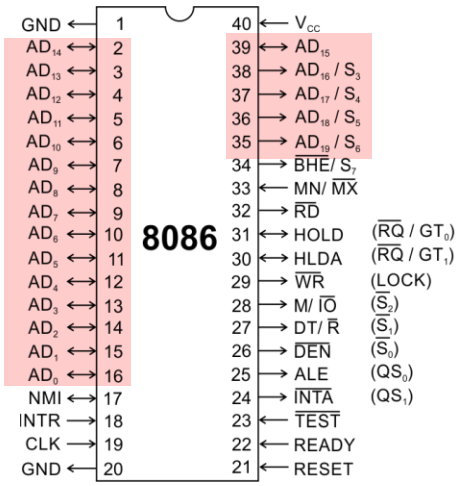
- BIU fetches and stores instructions once the queue has more than 2 empty bytes
- EU consumes instructions pre-fetched and stored in the queue at the same time
- Increases the efficiency of CPU
- *When it works?*
  - Sequential instruction execution
  - *Branch penalty:* when jump instruction executed, all pre-fetched instructions are discarded



8086/8088 Pins
(Compare them and tell the difference)



Signals Common to Min/Max Modes



AD_0-AD_{15} (Bidirectional)

Address/Data bus

Low order address bus; these are multiplexed with data.

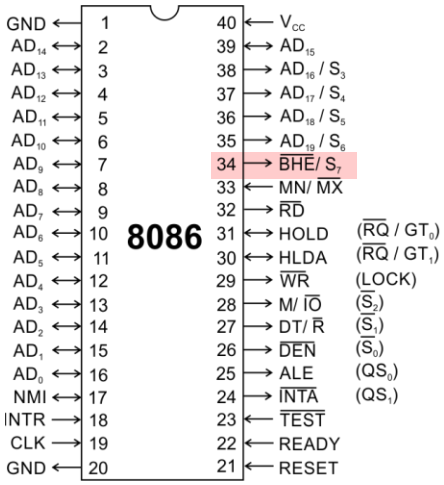
When AD lines are used to transmit memory address the symbol A is used instead of AD, for example A_0-A_{15}.

When data are transmitted over AD lines the symbol D is used in place of AD, for example D_0-D_7, D_8-D_{15} or D_0-D_{15}.

A_{16}/S_3, A_{17}/S_4, A_{18}/S_5, A_{19}/S_6

High order address bus. These are multiplexed with status signals

Signals Common to Min/Max Modes



BHE (Active Low)/S_7 (Output)

Bus High Enable/Status

It is used to enable data onto the most significant half of data bus, D_8-D_{15}. 8-bit device connected to upper half of the data bus use BHE (Active Low) signal. It is multiplexed with status signal S_7.

MN/ MX

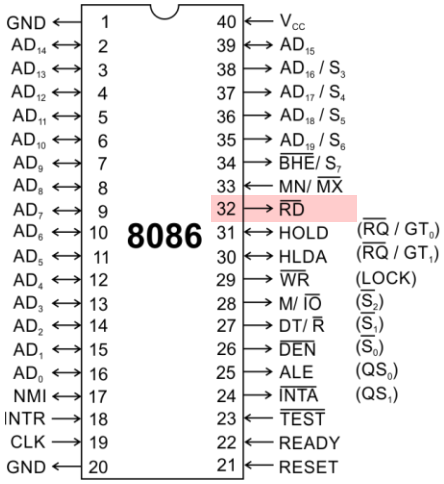
MINIMUM / MAXIMUM

This pin signal indicates what mode the processor is to operate in.

RD (Read) (Active Low)

The signal is used for read operation.
It is an output signal.
It is active when low.

Signals Common to Min/Max Modes



TEST

TEST input is tested by the 'WAIT' instruction.

8086 will enter a wait state after execution of the WAIT instruction and will resume execution only when the TEST is made low by an active hardware.

This is used to synchronize an external activity to the processor internal operation.

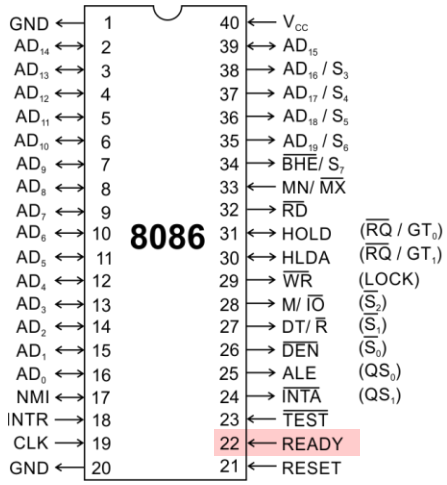
READY

This is the acknowledgement from the slow device or memory that they have completed the data transfer.

The signal made available by the devices is synchronized by the 8284A clock generator to provide ready input to the 8086.

The signal is active high.

Signals Common to Min/Max Modes



RESET (Input)

Causes the processor to immediately terminate its present activity.

The signal must be active HIGH for at least four clock cycles.

CLK

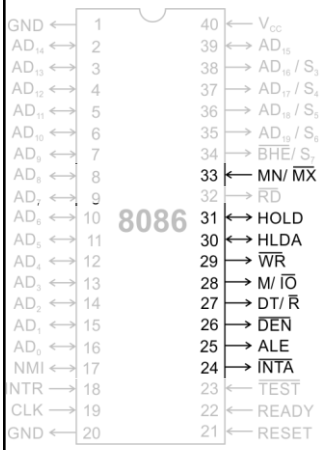
The clock input provides the basic timing for processor operation and bus control activity. Its an asymmetric square wave with 33% duty cycle.

INTR Interrupt Request

This is a triggered input. This is sampled during the last clock cycles of each instruction to determine the availability of the request. If any interrupt request is pending, the processor enters the interrupt acknowledge cycle.

This signal is active high and internally synchronized.

Min/Max Pins



The 8086 microprocessor can work in two modes of operations : Minimum mode and Maximum mode.

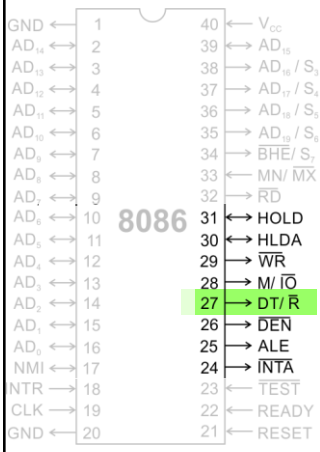
In the minimum mode of operation the microprocessor is not associated with any co-processors and can not be used for multiprocessor systems.

In the maximum mode the 8086 can work in multi-processor or co-processor configuration.

Minimum or maximum mode operations are decided by the pin MN/MX(Active low).

When this pin is high 8086 operates in minimum mode otherwise it operates in Maximum mode.

Minimum Mode Signals I



Pins 24 -31

For minimum mode operation, the MN/ $\overline{MX}$ is tied to VCC (logic high)

8086 itself generates all the bus control signals

DT/$\overline{R}$ — (**Data Transmit/ Receive**) Output signal from the processor to control the direction of data flow through the data transceivers

$\overline{DEN}$ — (**Data Enable**) Output signal from the processor used as out put enable for the transceivers

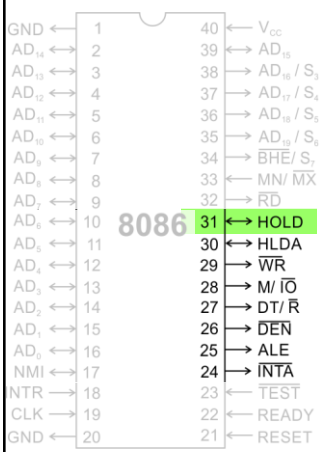
ALE — (**Address Latch Enable**) Used to demultiplex the address and data lines using external latches

M/$\overline{IO}$ — Used to differentiate memory access and I/O access. For memory reference instructions, it is **high**. For IN and OUT instructions, it is **low**.

$\overline{WR}$ — Write control signal; asserted **low** Whenever processor writes data to memory or I/O port

$\overline{INTA}$ — (**Interrupt Acknowledge**) When the interrupt request is accepted by the processor, the output is **low** on this line.

Minimum Mode Signals II



Pins 24 -31

For minimum mode operation, the MN/ $\overline{MX}$ is tied to VCC (logic high)

8086 itself generates all the bus control signals

HOLD — Input signal to the processor form the bus masters as a request to grant the control of the bus.

Usually used by the DMA controller to get the control of the bus.

HLDA — (**Hold Acknowledge**) Acknowledge signal by the processor to the bus master requesting the control of the bus through HOLD.

The acknowledge is asserted high, when the processor accepts HOLD.

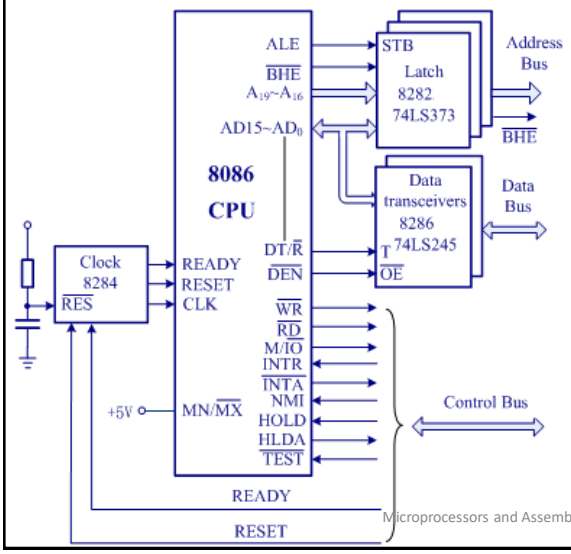
Control Signals

- **MN/~MX:** Minimum mode (high level), Maximum mode (low level)
- **~RD:** output, CPU is reading from memory or IO
- **~WR:** output, CPU is writing to memory or IO
- **M/~IO:** output, CPU is accessing memory (high level) or IO (low level)
- **READY:** input, memory/IO is ready for data transfer
- **~DEN:** output, used to enable the data transceivers
- **DT/~R:** output, used to inform the data transceivers the direction of data transfer, i.e., sending data (high level) or receiving data (low level)
- **~BHE:** output, ~BHE=0, AD8-AD15 are used, ~BHE=1, AD8-AD15 are not in use
- **ALE:** output, used as the latch enable signal of the address latch

Control Signals

- **HOLD:** input signal, hold the bus request
- **HLDA:** output signal, hold request ack
- **INTR:** input, interrupt request from 8259 interrupt controller, maskable by clearing the IF in the flag register
- **INTA:** output, interrupt ack
- **NMI:** input, non-maskable interrupt, CPU is interrupted after finishing the current instruction; cannot be masked by software
- **RESET:** input signal, reset the CPU
  - IP, DS, SS, ES and the instruction queue are cleared
  - CS = FFFFH
  - *What is the address of the first instruction that the CPU will execute after reset?*

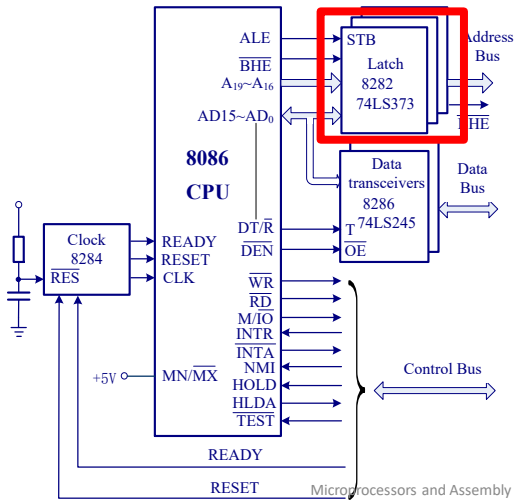
Minimum Mode Configuration

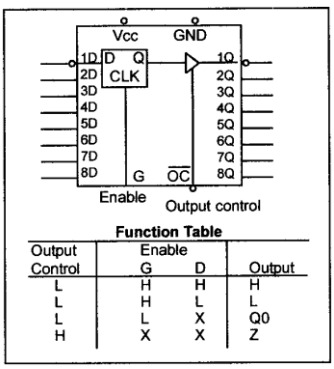


8086/88's two work modes:

- Minimum mode: $MN/\overline{MX}$ =1
  - Single CPU;
  - Control signals from the CPU
- Maximum mode: $MN/\overline{MX}$ =0
  - Multiple CPUs(8086+8087)
  - 8288 control chip supports

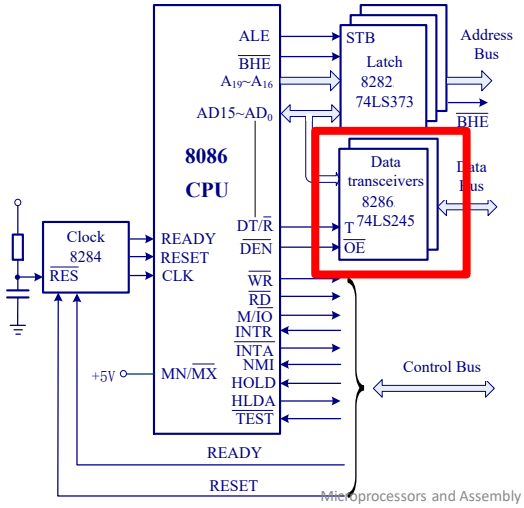
Address/Data Demultiplexing & Address latching

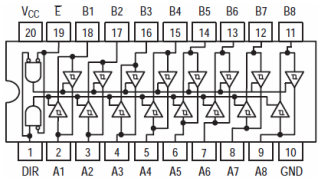




74LS373 D Latch

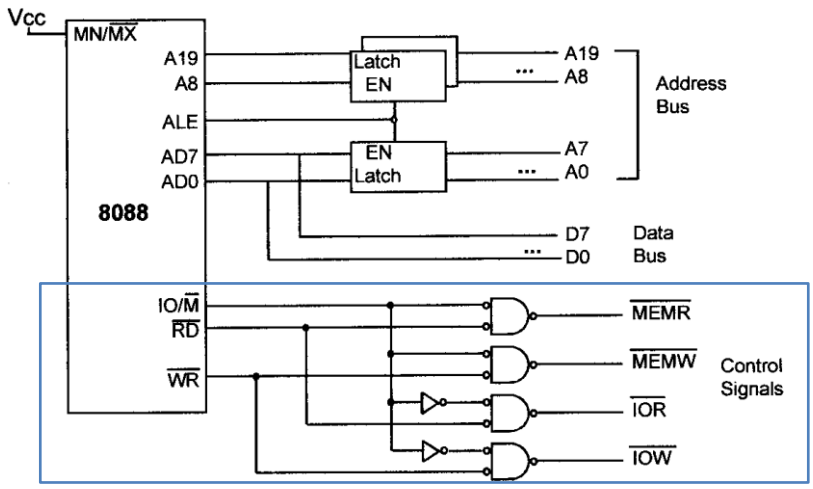
Data Bus Transceiver



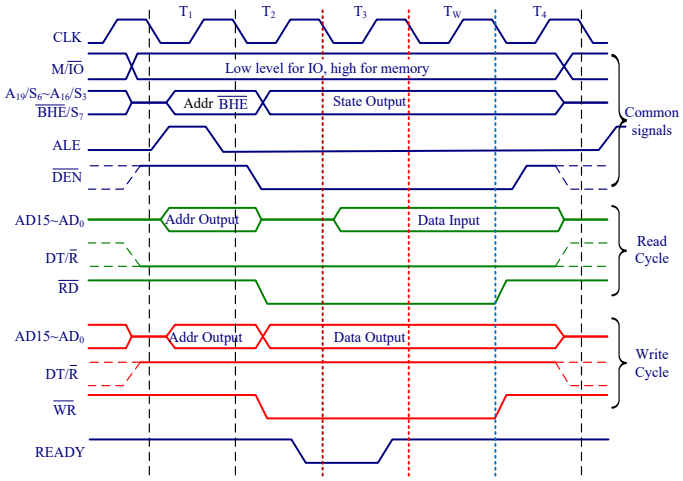


INPUTS		OUTPUT
$\overline{E}$	DIR	
L	L	Bus B Data to Bus A
L	H	Bus A Data to Bus B
H	X	Isolation

Memory/IO Control Signals



8086/88 Bus Cycle (for data transfers)



At least 4 clock cycles

Next Lecture

- 80x86 memory organization
  - Memory segments
- Addressing modes