8086

Prof.Ms.Aaradhana Deshmukh
aaradhana-deshmukh.in
Some Questions

1. 8086 is how many bit processor?
2. 16 bit means?
3. What is the size of data bus?
4. How many bits can be read from memory and write into memory? Why?
5. What is the size of address bus? It can access how many ports?
6. Is 8086 possible to perform bit? Byte? Word? Block operations?
7. Is 8086 supports multiprogramming?
8. Long form of Intel?
Software architecture of the INTEL 8086

- Memory segmentation and addressing
- Block diagram of 8086
- Address space & Data organization
- Data Types
- Registers
- Stack
- I/O space
Memory segmentation and addressing

- Von – Newman architecture & Harvard architecture
- Program Memory & Data Memory
- Need for Segmentation
  - To implement Harvard architecture
  - Easy to debug
  - Same Interfacing ICs can be used
  - To avoid overlap of stack with normal memory
  - Compatible with 8085

Prof.Ms.Aaradhana A. Deshmukh, SKNCOE, Comp
Memory Address Generation

• The BIU has a dedicated adder for determining physical memory addresses.
Segment : Offset Address

- Logical Address is specified as `segment:offset`
- Physical address is obtained by shifting the segment address 4 bits to the left and adding the offset address.
- Thus the physical address of the logical address `A4FB:4872` is:

\[
\begin{align*}
A4FB0 & \\
+ & 4872 \\
\hline
A9822
\end{align*}
\]
Segments, Segment Registers & Offset Registers

- Segment Size = 64KB
- Maximum number of segments possible = 14
- Logical Address – 16 bits
- Physical Address – 20 bits
- 2 Logical Addresses for each Segment.
  - Base Address (16 bits)
  - Offset Address (16 bits)
- Segment registers are used to store the Base address of the segment.
Segments, Segment Registers & Offset Registers

• 4 Segments in 8086
  – Code Segment (CS)
  – Data Segment (DS)
  – Stack Segment (SS)
  – Extra Segment (ES)

<table>
<thead>
<tr>
<th>SEGMENT</th>
<th>SEGMENT REGISTER</th>
<th>OFFSET REGISTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code Segment</td>
<td>CSR</td>
<td>Instruction Pointer (IP)</td>
</tr>
<tr>
<td>Data Segment</td>
<td>DSR</td>
<td>Source Index (SI)</td>
</tr>
<tr>
<td>Extra Segment</td>
<td>ESR</td>
<td>Destination Index (DI)</td>
</tr>
<tr>
<td>Stack Segment</td>
<td>SSR</td>
<td>Stack Pointer (SP) / Base Pointer (BP)</td>
</tr>
</tbody>
</table>
FIGURE 2-7 8086 internal block diagram. (Intel Corp.)
Figure 3.1 Processor model for the 8086 microprocessor. A separate execution unit (EU) and bus interface unit (BIU) are provided.
Pipelined architecture of the 8086 microprocessors

Diagram:

- Execution Unit
- Instruction Pipeline
- Bus Interface Unit
- System Bus
Execution and bus interface units

Prof. Ms. Aaradhana A. Deshmukh,
SKNCOE, Comp
Software Model of the 8086 Microprocessors

Prof. Ms. Aaradhana A. Deshmukh,
SKNCOE, Comp
Microprocessor Architecture

Day 5
Prof. Aaradhana Deshmukh
Address space & Data organization

Storing a word in memory

What is the word in (b) in Hex?

Memory address space
8086 Registers

General Purpose

- AX: AH, AL
- BX: BH, BL
- CX: CH, CL
- DX: DH, DL

Index

- BP
- SP
- SI
- DI

Segment

- CS
- SS
- DS
- ES

Status and Control

- Flags
- IP
General Purpose Registers

- Normally used for storing temporary results
- Each of the registers is 16 bits wide \((AX, BX, CX, DX)\)
- Can be accessed as either 16 or 8 bits \(AX, AH, AL\)

<table>
<thead>
<tr>
<th></th>
<th>H</th>
<th>8</th>
<th>7</th>
<th>L</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>AX</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AH</td>
<td></td>
<td>AH</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BX</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BH</td>
<td></td>
<td>BH</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CX</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CH</td>
<td></td>
<td>CH</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DX</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DH</td>
<td></td>
<td>DH</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

AX - the Accumulator
BX - the Base Register
CX - the Count Register
DX - the Data Register
General Purpose Registers

• **AX**
  – Accumulator Register
  – Preferred register to use in arithmetic, logic and data transfer instructions because it generates the shortest Machine Language Code
  – Must be used in multiplication and division operations
  – Must also be used in I/O operations

• **BX**
  – Base Register
  – Also serves as an address register
General Purpose Registers

- **CX**
  - Count register
  - Used as a loop counter
  - Used in shift and rotate operations

- **DX**
  - Data register
  - Used in multiplication and division
  - Also used in I/O operations
Pointer and Index Registers

- All 16 bits wide, L/H bytes are not accessible

- Used as memory pointers
  - Example: MOV AH, [SI]
    - Move the byte stored in memory location whose address is contained in register SI to register AH

- IP is not under direct control of the programmer
Flag Register
- Is a flip flop – which indicates some condition – produced by execution if instruction or controls certain operations of EU
Flag Register

6 are status flags
3 are control flag
# 8086 Programmer’s Model

## BIU registers (20 bit adder)

<table>
<thead>
<tr>
<th>ES</th>
<th>Extra Segment</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS</td>
<td>Code Segment</td>
</tr>
<tr>
<td>SS</td>
<td>Stack Segment</td>
</tr>
<tr>
<td>DS</td>
<td>Data Segment</td>
</tr>
<tr>
<td>IP</td>
<td>Instruction Pointer</td>
</tr>
</tbody>
</table>

## EU registers

<table>
<thead>
<tr>
<th>AX</th>
<th>AH</th>
<th>AL</th>
</tr>
</thead>
<tbody>
<tr>
<td>BX</td>
<td>BH</td>
<td>BL</td>
</tr>
<tr>
<td>CX</td>
<td>CH</td>
<td>CL</td>
</tr>
<tr>
<td>DX</td>
<td>DH</td>
<td>DL</td>
</tr>
<tr>
<td></td>
<td>SP</td>
<td></td>
</tr>
<tr>
<td></td>
<td>BP</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SI</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DI</td>
<td></td>
</tr>
<tr>
<td></td>
<td>FLAGS</td>
<td></td>
</tr>
</tbody>
</table>
The Stack

- The stack is used for temporary storage of information such as data or addresses.
- When a **CALL** is executed, the 8086 automatically **PUSHes** the current value of CS and IP onto the stack.
- Other registers can also be pushed.
- Before return from the **subroutine**, **POP** instructions can be used to pop values back from the stack into the corresponding registers.
The Stack

Prof. Ms. Aaradhana A. Deshmukh,
SKNCOE, Comp
8086
5th Day

Prof. Ms. Aaradhana Deshmukh
aaradhana-deshmukh.in
Hardware Architecture of INTEL 8086
Hardware Architecture of INTEL 8086

- Pin Diagram and Pin Details
- min/max mode
- Hardware organization of address space
- Control signals
- Coprocessor and Multiprocessor configuration
- I/O interfaces
INTEL 8086 - Pin Diagram

GND 1 40 VCC
AD14 2 39 AD15
AD13 3 38 A16/S3
AD12 4 37 A17/S4
AD11 5 36 A18/S5
AD10 6 35 A19/S6
AD9 7 34 BHE/S7
AD8 8 33 MN/MX
AD7 9 32 RD
AD6 10 31 RO/GT0 (HOLD)
AD5 11 30 RO/GT1 (HLDA)
AD4 12 29 LOCK (WR)
AD3 13 28 S2 (M/IO)
AD2 14 27 S1 (DT/R)
AD1 15 26 S0 (DEN)
AD0 16 25 QS0 (ALE)
NMI 17 24 QS1 (INTA)
INTR 18 23 TEST
CLK 19 22 READY
GND 20 21 RESET
INTEL 8086 - Pin Details

**Power Supply**
- 5V ± 10%

**Reset**
- Registers, seg regs, flags
- CS: FFFFH, IP: 0000H
- If high for minimum 4 clks

**Ground**

**Clock**
- Duty cycle: 33%
Address/Data Bus:
Contains address bits $A_{15}-A_0$ when ALE is 1 & data bits $D_{15}-D_0$ when ALE is 0.

Address Latch Enable:
When high, multiplexed address/data bus contains address information.
INTEL 8086 - Pin Details

- Interrupt
- Non-maskable interrupt
- Interrupt request
- Interrupt acknowledge
INTEL 8086 - Pin Details

Direct Memory Access

Hold acknowledge

Hold
INTEL 8086 - Pin Details

S6: Logic 0.

S5: Indicates condition of IF flag bits.

S4-S3: Indicate which segment is accessed during current bus cycle:

<table>
<thead>
<tr>
<th>S4</th>
<th>S3</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Extra segment</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Stack segment</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Code or no segment</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Data segment</td>
</tr>
</tbody>
</table>

Address/Status Bus
Address bits $A_{19}$ – $A_{16}$ & Status bits $S_6$ – $S_3$
**INTEL 8086 - Pin Details**

**BHE#, A₀:**
- **0,0:** Whole word (16-bits)
- **0,1:** High byte to/from odd address
- **1,0:** Low byte to/from even address
- **1,1:** No selection

**Bus High Enable/S7**
Enables most significant data bits $D_{15} - D_8$ during read or write operation.

**$S_7$: Always 1.**
INTEL 8086 - Pin Details

Min/Max mode
Minimum Mode: +5V
Maximum Mode: 0V

Minimum Mode Pins

Maximum Mode Pins
Minimum Mode - Pin Details

- Read Signal
- Write Signal
- Memory or I/O
- Data Transmit/Receive
- Data Bus Enable
Minimum Mode

• When only one 8086 CPU is to be used in microcomputer system.
• CPU issues the control signals required by memory and I/O devices
Maximum Mode - Pin Details

S2 S1 S0

000: INTA
001: read I/O port
010: write I/O port
011: halt
100: code access
101: read memory
110: write memory
111: none - passive

Status Signal
Inputs to 8288 to generate eliminated signals due to max mode.
Lock Output

Used to lock peripherals off the system

Activated by using the LOCK: prefix on any instruction
Maximum Mode - Pin Details

**Queue Status**
Used by numeric coprocessor (8087)

**QS1 QS0**
00: Queue is idle
01: First byte of opcode
10: Queue is empty
11: Subsequent byte of opcode
Maximum Mode

- When more than one 8086 CPU is to be used in microcomputer system.
- The control signals generated with the help of external bus controller.
Minimum Mode 8086 System
Maximum Mode 8086 System
Maximum Mode 8086 System

• Here, either a numeric coprocessor of the type 8087 or another processor is interfaced with 8086.

• The Memory, Address Bus, Data Buses are shared resources between the two processors.

• The control signals for Maximum mode of operation are generated by the Bus Controller chip 8788.

• The three status outputs S0*, S1*, S2* from the processor are input to 8788.

• The outputs of the bus controller are the Control Signals, namely DEN, DT/R*, IORC*, IOWTC*, MWTC*, MRDC*, ALE etc.
8086 Interrupts

- External Hardware Interrupts
- Nonmaskable Interrupt
- Software Interrupts
- Internal Interrupts
- Reset
8086 Interrupts Procedure

The Operation of Real Mode Interrupt

1. The contents of the FLAG REGİSTERS are pushed onto the stack.
2. Both the interrupt (IF) and (TF) flags are cleared. This disables the INTR pin and the trap or single-step feature. (Depending on the nature of the interrupt, a programmer can unmask the INTR pin by the STI instruction.)
3. The contents of the code segment register (CS) is pushed onto the stack.
4. The contents of the instruction pointer (IP) is pushed onto the stack.
5. The interrupt vector contents are fetched, and then placed into both IP and CS so that the next instruction executes at the interrupt service procedure addressed by the interrupt vector.
6. While returning from the interrupt-service routine by the instruction IRET, flags return to their state prior to the interrupt and operation restarts at the prior IP address.
8086 External Interrupts

8086 MPU

Vcc

MN/MX

External hardware interrupt interface circuitry

AD0-AD15

INTR

INTA

DT/R

DEN

ALE

M/IO

RD

WR

INT32

INT33

INT34

INT255

Prof.Ms.Aaradhana A. Deshmukh,
SKNCOE, Comp
8086 Control Signals

1. ALE
2. BHE
3. M/IO
4. DT/R
5. RD
6. WR
7. DEN
Coprocessor and Multiprocessor configuration

- Multiprocessor Systems refer to the use of multiple processors that executes instructions simultaneously and communicate with each other using mail boxes and Semaphores.

- Maximum mode of 8086 is designed to implement 3 basic multiprocessor configurations:
  1. Coprocessor (8087)
  2. Closely coupled (8089)
  3. Loosely coupled (Multibus)
Coprocessor and Multiprocessor configuration

- Coprocessors and Closely coupled configurations are similar in that both the 8086 and the external processor shares the:
  - Memory
  - I/O system
  - Bus & bus control logic
  - Clock generator
Coprocessor / Closely Coupled Configuration

CLOCK

8086

Bus Control Logic

Coprocessor or Independent Processor

System Bus

Memory

I/O

Prof. Ms. Aaradhana A. Deshmukh, SKNCOE, Comp
TEST pin of 8086

• Used in conjunction with the WAIT instruction in multiprocessing environments.

• This is input from the 8087 coprocessor.

• During execution of a wait instruction, the CPU checks this signal.

• If it is low, execution of the signal will continue; if not, it will stop executing.
Coprocessor Execution Example

Coprocessor cannot take control of the bus, it does everything through the CPU

Starting with the 8086/8088:

1. ESC
   - Wake up the coprocessor
2. Execute 8086 instructions
3. WAIT
   - Wake up the 8086 or 8088

The coprocessor (i.e., 8087) monitors the 8086 or 8088:

1. Monitor the 8086 or 8088
2. Deactivate the host's !TEST pin and execute the specified operation
3. Activate the !TEST pin

Closely Coupled Execution Example

- Closely Coupled processor may take control of the bus independently.
- Two 8086’s cannot be closely coupled.
Loosely Coupled Configuration

• has shared system bus, system memory, and system I/O.

• each processor has its own clock as well as its own memory (in addition to access to the system resources).

• Used for medium to large multiprocessor systems.

• Each module is capable of being the bus master.

• Any module could be a processor capable of being a bus master, a coprocessor configuration or a closely coupled configuration.
Loosely Coupled Configuration

- **No direct connections** between the modules.
- Each share the system bus and **communicate through shared resources**.
- Processor in their separate modules can simultaneously access their private subsystems through their local busses, and perform their local data references and instruction fetches independently. This results in **improved degree of concurrent processing**.
- **Excellent** for real time applications, as separate modules can be assigned specialized tasks.
Advantages of Multiprocessor Configuration

1. High **system throughput** can be achieved by having more than one CPU.

2. The system can be **expanded** in modular form.
   Each bus master module is an independent unit and normally resides on a separate PC board. One can be added or removed without affecting the others in the system.

3. A **failure** in one module normally does not affect the breakdown of the entire system and the faulty module can be easily detected and replaced.

4. Each bus master has its own local bus to access dedicated memory or IO devices. So a greater **degree of parallel processing** can be achieved.

Prof.Ms.Aaradhana A. Deshmukh,
SKNCOE, Comp
8086 System Memory Circuitry

1. Minimum Mode System Memory Circuitry

2. Maximum Mode System Memory Circuitry
Minimum Mode System Memory Circuitry

Diagram showing the connections and components of a system memory circuit, including parts labeled 8284A CLOCK GENERATOR, MN/MX, WD, RD, WR, DT/R, DEN, ALE, AD0-AD15, A18-A19, BHE, 8086, LATCH, TRANSCEIVER, and various memory chips such as 2142 RAM (4), 2716-2 PROM (2), and Chip Select Logic.
Maximum Mode System Memory Circuitry

- **CLK**: Clock input
- **AD0-AD15**, **A16-A19**: Address bus
- **BHE**: Bus enables
- **CE0-CE7**, **A1L-A16L**: Address decoder inputs
- **WRU**, **WR_L**: Write signals
- **RDu**, **RD_L**: Read signals
- **8086 MPU**, **MN/MX**: Central processing unit
- **S0, S1, S2**: Controller signals
- **MWTC**: Memory write control
- **8288 MRDC**: Memory read control
- **DT/R DEN**: Data bus transceiver buffer
- **DIR EN**: Directional enable
- **D0-D15**: Data bus
- **Ready**: Readiness signal