Introduction of Assembly Language in 8086

Yuan Long
Csc3210
8086 Microprocessor

• Intel 8086 was launched in 1978.
• The first 16-bit microprocessor.
• 1 megabyte of memory.
• 20-bit address bus
**Bus Interface Unit (BIU)** fetches instructions from memory, passes the instruction to the instruction stream byte queue and starts to fetch the next instruction immediately.

**Execution Unit (EU)** removes instructions from the instruction queue.
Program data and storage

- Psedo-ops to define data or reserve storage
  - DB – bytes
  - DW – words

- These directives require one or more operands
  - Define memory contents
  - Specify amount of storage to reserve for run-time data
Define Data

• Numeric data values
  – 100 decimal
  – 100B binary
  – 100H hex
  – ‘x’ ASCII
  – “100” ASCII

E.g. The following creates 4 consecutive words
dw 40, 10b, -13,0
Define Data

• Names can be associated with storage locations
  – X dw 100 ; int x=100;
  – Y db -4 ; byte y=-4;
Arrays

• Any consecutive storage locations of the same size can be called an array

X dw 40h,10b,-13,0
Y db ‘This is an array’

Components of x are at x, x+2,x+4,x+8
Components of y are at y,y+1,...,y+15
Data transfer instructions

• Mov target, source
  – Reg,reg
  – Mem,reg
  – Reg,mem
  – Mem,immed
  – Reg,immed

Size of both operands must be the same.
Sample Mov instructions

<table>
<thead>
<tr>
<th>Instruction</th>
<th>AH</th>
<th>AL</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOV AH,01010101B</td>
<td>0 1 0 1 0 1 0 1</td>
<td></td>
</tr>
<tr>
<td>MOV AL,00100111B</td>
<td>0 1 0 1 0 1 0 1 0 0 1 0 0 1 1 1</td>
<td></td>
</tr>
<tr>
<td>MOV AX,3</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1</td>
<td></td>
</tr>
<tr>
<td>MOV AH,AL</td>
<td>0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 1 1</td>
<td></td>
</tr>
<tr>
<td>MOV AL,10D</td>
<td>0 0 0 0 0 0 1 1 0 0 0 0 0 1 0 1 0</td>
<td></td>
</tr>
<tr>
<td>MOV AL,10H</td>
<td>0 0 0 0 0 0 1 1 0 0 0 1 0 0 0 0 0</td>
<td></td>
</tr>
</tbody>
</table>

B for binary.
D for decimal.
H for hex.
Sample MOV instructions

b db 4fh ; byte b=0x4fh
w dw 2048 ; int w=2048
y db 4fh,20h,3ch ; array y

Mov bl, dh ; bl=dh
Mov ax, w  ; ax=w
Mov ch, b  ; ch=b
Mov w, -100 ; w=-100
Mov b, 0  ; b=0

Mov ah, y+1
Arithmetic Instructions

Add dest,source
Sub dest,source
Inc dest
Dec dest
Neg dest

Source: general register, memory location, or constant
Dest: register or memory location
Program segment structure

- Data segment
  - Storage for variables
  - Variable addresses are computed as offset from start of this segment

- Code segment
  - Contains executable instructions

- Stack segment
  - Set aside storage for the stack
Program skeleton

- Select a memory model
- Define the stack size
- Declare variables
- Write code
- Mark the end of the source file
Software Interrupts

• Interrupt is a process where an external device can get the attention of the microprocessor.
  e.g. I/O device

• An interrupt is called using the instruction INT followed by the interrupt number.
  e.g. INT 21H
    - branches in the operating system and permits the use of PC-DOS functions.
## Some INT 21H functions

<table>
<thead>
<tr>
<th>Function Number</th>
<th>Description</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Keyboard Input</td>
<td>Waits until a character is typed at the keyboard and then puts the ASCII code for that character in register AL and echoed to screen</td>
</tr>
<tr>
<td>2</td>
<td>Display Output</td>
<td>Prints the character whose ASCII code is in DL</td>
</tr>
<tr>
<td>8</td>
<td>Keyboard Input</td>
<td>Waits until a character is typed at the keyboard and then puts the ASCII code for that character in register AL and NOT echoed to screen</td>
</tr>
<tr>
<td>9</td>
<td>Display String</td>
<td>Prints a series of characters stored in memory starting with the one in the address given in DX (relative to DS). Stop when the ASCII code for $ is encountered</td>
</tr>
</tbody>
</table>
Some Basic Graphics

• “The principal reason for using assembler is because of the increased speed of execution which it gives, and one area where this speed is most appreciated is in the area of computer graphics.”

• Available at http://www.skynet.ie/~darkstar/assembler/tut6.html
Some Basic Graphics

• **Interrupt 10h**

```assembly
;========================================= ;
Basic program to change graphics modes ;
========================================= ;
; Resolution: 640*480, 16 color
mov ah,00 ; subfunction 0
mov al,18 ; select mode 18 (or 12h if prefer)
int 10h ; call graphics interrupt ;

==== Graphics code here ====
mov ah,00 ; again subfunc 0
mov al,03 ; text mode 3
int 10h ; call int

mov ah,04ch
mov al,00 ; end program normally int 21h
```
Some Basic Graphics

• Interrupt 10h

    STRING DB ‘WORLD SCENERY’

    ;
    MOV     AL, 3       ; 80*25
    MOV     AH, 0
    INT     10H

    MOV     DX, SEG STRING ; Address of string
    MOV     DS, DX
    MOV     DX, OFFSET STRING ;

    MOV     AH, 9       ; Display string
    INT     10H         ; Call BIOS

80*25:
Text interface 80 characters wide and 25 characters lines per screen.
Demo

- Hello.asm
Reference

• Some Basic Graphics
  http://www.skynet.ie/~darkstar/assembler/tut6.html
• Interrupts
  http://www.ustudy.in/node/939
• The 8086 Chip
  http://www.skynet.ie/~darkstar/assembler/tut1.html
• 8086 Assembler Tutorial for Beginners
  http://frz.ir/dl/tuts/8086_Assembly.pdf
• Slides for Assembly language 8086 intermediate
  http://www.slideshare.net/JohnCutajar/assembly-language-8086-intermediate
• Slides for 8086 assembly language
  http://www.slideshare.net/mir_majid_kant/8086-assembly-language