**LAB 1: INTRODUCTION TO BGC-8088 AND MICROPROCESSORS**

**Objective**

1. Understanding the built and operation of BGC 8088.

2. Introduction to Assembly Language Programming with Intel Microprocessor 8088.
S ECTION A: Introduction to BGC 8088

BGC 8088 Microengineer is a standalone Microprocessor Training System. Standalone means it can be used with or without Computers. This section will brief you about its properties,

i) General

a. Observe the BGC 8088. Name 5 of the IC that is available inside of BGC.

b. Which one of the IC is the microprocessor? Draw a simple schematic diagram with pin assignments of this IC.

c. Draw a simple schematic diagram of BGC 8088

ii) Hardware Specifications

1. CPU: Intel 8088 running at Clock Rate of 4.77 Mhz
2. System Memory consist of 32 KB RAM and 16 KB ROM (expandable to 32 KB)
3. Display: 2 X 40 character LCD module
4. Keyboard: 56 keys keyboard (alphanumeric and ASCII Symbols)
5. Built in Printer Interface: Provides the most general parallel printer interface. Connection is similar to any IBM PC
6. RS232. To allow program to be transferred from host PC to BGC 8088 and vice versa
7. Parallel Control Interface This interface is built from 3 IC chips (8254, 8255 and 8259 A) It provides 3 8 bits I/O Ports, 3 Timers, 5 interrupt signals
d. Explain the following terms

Clock Rate:

RAM:

ROM:

Memory:

Interrupt Signals

e. Name 3 IC chips mentioned in the above section (no. 7)

iii) Software Functions

Monitor Commands

Monitor Commands are the key commands in BGC 8088 Microengineer to enable the user to use the system. It’s the key to ‘tell’ the Microengineer on ‘what to do’, for example: when to start typing program or when to execute a program. At the end of this lab, you should know the letters to use for which purpose. The monitor commands must be typed in a command prompt.

Types of Monitor Commands

1. Assembler Commands (A, I and U)
   A: Assemble the assembly language instructions
   I: Insert Mode
   U: Dissembler: (List)

2. Program Control Commands (G and R)
   G: Execute
   R: Display / Modify the register content
   T: Trace program execution

3. Memory Management (C, D, E, F, M)
   C: Compare the memory contents
   D: Display the memory content
   E: Substitute the memory content
F: Fill into a memory block  
M: Move the memory content  

4. Numerical treatment (B, H, J, S, V)  

B: Convert decimal into binary  
H: Calculate the sum and difference of two hexadecimal numbers  
J: Convert decimal number into hexadecimal number  
S: Convert hexadecimal number into decimal number  

5. I/O Commands (N, O)  

N: Input data from I/O device  
O: Output data to I/O device  

6. Communication commands (L, Z)  

L: Download a program from host  
Z: Upload a program to host  

**Step 1: Command Prompt and Address Prompt**  
On the BGC8088 and type `a` at the * prompt (command prompt), and press CR (carriage return).  

```
* A
```

The output will be:  
0100:0000  

This prompt means that the Microengineer is now ready for program statements. The statement typed at the above prompt will be written to memory address 0000 in the memory block 0100. 'a' is a Monitor Command. Commands should be entered at the * prompt. Program statements must be entered at the address prompt. (The address is like 0100:0000.)  

**Note:**  
**0100: 000**  
0100 is segment address  
0000 is offset address  

f. Type `a` at the address prompt. What do you observe? Why does this happen?
Section B: Introduction to Assembly Language Programming with BGC 8088

Type the following program.
[You have to go to the address prompt before you can type program statements. Note the address of each instruction. Also note the address after the last instruction.]

Address 1    MOV AX,5
Address 2    MOV BX,4
Address 3    ADD BX,AX

Hit CR to exit the address prompt.

g. Return to the address prompt. Which address does it take you to? Comment.

Step 2: Correcting errors
If you want to write your program from any address that you choose, or you want to go to a previous address in your program, use the ‘i’ command.

h. Now type in a change to the program:

*i003
0100:0003 MOV BX,3
0100:0006 DEC AX
0100:0007 ADD AX,BX

Step 3: Displaying the program
The next command is the ‘u’ command. This is used to display the program in a certain range of memory. Therefore, an address range must be specified.

For example, enter:

* U0000,0006

The above means "display the program in memory from 0000 to 0006". Compare this with

* u0,6
i. Now, list the program you have just typed in with their address.

j. Is the BGC8088 case sensitive? If we were to use the hexadecimal number F instead of 000F, would the result be the same?
   What are the hexadecimal numbers between the address and the instruction? Copy the hexadecimal number for each instruction.
   Compare the number of hexadecimal digits with the number of address locations that the corresponding instruction requires. Hence, explain why different instructions need different amounts of space.
**Step 4: Executing the program**
In order to execute a program, we have to tell the BGC 8088 which instructions to execute. For that, you have to give the address of the first instruction and the address following the last instruction.

Format:  \textbf{g = starting address, address after the last instruction}

For example, to execute the first line of our program, the command would be:

* \text{g=0,3}

k. Display the first line of your program. You will get a display of the registers and their contents. Explain the contents of the registers.

l. Now execute the whole program. Explain the results.

**Step 5: Displaying and changing the register contents**
To display the registers at any time, give the ‘r’ command.

* \text{R}

To modify the contents of a register without executing a program, for example the register CX, you would type ‘rcx’, to modify IP, you would type ‘rip’, and so on.

m. Change the value inside register BX. How you do that?

**Step 6: Executing a single instruction**
Using the ‘t’ command, you can execute (trace) a single statement.

n. Set the contents of AX, BX and IP to 0. Then execute a single instruction:

* \text{T}

Repeat this three times. Comment on the result of each execution.
o. Set IP to 3, AX and BX to 0. Use the ‘t’ command. Which instruction was executed? What does this tell you about the ‘t’ command?

Summary

After finishing all the steps above, one should have basic knowledge on writing and executing an assembly language program with BGC 8088. More monitor commands will be explored in coming labs together with more Intel’s Instruction Set for 8088.

Following questions is to be answered in a separate paper (hand written only). Please submit this manual in 1 (one) week from date of practical section. Late reports will carry –2 marks. Extremely late reports (i.e. reports submitted after lab manual is returned to class) will not graded.

Questions

1. What’s the difference between a Microprocessor and a Micro controller?
2. What are registers? List all the registers for 8088 and 8086 and their usage.
3. What is memory? What are they made of and how are they arranged in Intel 8088?
4. Explain following terms
   a. ALU
   b. Pipelining
   c. Micron
5. What is Effective address, briefly how do we calculate from
   Segment: offset  ex: 0100:0000