Komar University of Science and Technology

College of Engineering

Subject: Term Report– Spring 2020

Department Name: Computer Engineering

Title: Fibonacci number generator

Course Code: CPE3435C

Course Name: Microprocessor

Student ID: F170064

Student Name: Muhammad HamaSaeed Ahmad

Submission Date: Jun 20th 2020

Introduction:

Every personal computer has a processor that called microprocessor as well. This processor manages the inside of computer which consists of arithmetical, logical and control activities. Each type of processor has its own set of instructions for operating different operations such as getting input from the pointer (mouse). These sets of instructions are called ‘machine language instructions’,

A processor can handle understanding machine language which are strings of 0’s and 1’s, but won’t understand any other languages. Nevertheless, machine language is too hard and complicated to be used in software developments and programming. Therefore; the low-level programming language, which is called assembly language, is designed for a specific type of processors that represents different instructions in symbolic code and a more logical form.

An 8086 microprocessor is an improved version of 8085 microprocessor that was designed by intel in 1979, which also one of those processor types that uses assembly language as its low programming language. 8086 microprocessor is a 16-bit processor that has 20 address lines and 16 data and bus lines, and also it provides up to 1MB storage. It contains powerful instruction sets, like having operations such us multiplication and division easily.

This version of processor supports two modes of operations, which are maximum mode and minimum mode. Maximum mode is suitable for system having multiple processors and Minimum mode is suitable for system having a single processor.

This report is an explanation about a code that has written in assembly language, which generates and finds Fibonacci numbers from 0-100 using a loop. The Fibonacci series is one of the most famous formulas in mathematics. Each number in the series is the sum of the two numbers that precede it. So, the sequence goes: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, and so on.

The code:

```
Line 1:     org 100h
Line 2:     mov ax, 00h
Line 3:     mov bx, 500h
Line 4:     mov [bx], ax
Line 5:     add bx, 1
Line 6:     add ax, 1
Line 7:     mov [bx], ax
Line 8:     mov cx, 0Ch
Line 9:     sub cx,02h
Line 10:    L1:
Line 11:        mov ax, [bx-1]
Line 12:        add ax, [bx]
Line 13:        add bx, 1
Line 14:        mov [bx], ax
Line 15:        dec cx
Line 16:      jnz Loop L1
Line 17:    hlt
```

Let’s go through the lines step by step:

Line 1: this is the assembler directive. every single code in assembly language.

Line 2: this line moves that data(00h) which is in hexadecimal to register (ax), therefore; this register contains the initial point of the Fibonacci sequence.

Line 3: also, this instruction moves (0500h) to (bx) register.

Line 4: this instruction transfers all the data from register (ax) to a memory location (bx) which will be on 0500h in the memory.

Line 5: the instruction (add) used to add two operands together, here the operands will be as adding 1 to the data that is in register bx, bx will be (0501h). this memory location address will be increased each time the loop finishes a circle.

Line 6: this line adds 1 to the data in register ax, and ax becomes (01h).

Line 7: in this line, the data from register ax will be sent to memory bx again because of the change that happens in line 6.

Line 8: the cx register is the register that we put the counter for the loop to terminate inside that register, so we set the counter to 12 decimal which will be equal to 0C in hexadecimal, and move it to register cx.

Line 9: since we already have sat the two initial points of the sequence which were in line 2 and 5, so the counter should be reduced by 2, and this will be done by subtracting the counter (register cx) by 02h.

Line 10: this is the beginning of the loop, the loop instructions shall start after this line, and L1 represents the name of the loop.

Line 11: the first instruction inside the loop is to send the data in memory location bx in case to make the changes that needed for it.

Line 12: this line adds up the two numbers that comes after one, to create the sequence.

Line 13: this line increases the number of the memory location address each time the loop terminates, so all the data won’t be inside one memory location.

Line 14: here the result will be moved to the new memory location.

Line 15: this instruction decreases the counter of the loop.

Line 16: this line is the end of the loop and it will jump out of the loop when the counter of the loop becomes 0.

Line 17: this is the end of the code and the program terminates once it reached this point.

The output will be like this:

0700:0500: 00 000

0700:0501: 01 001

0700:0502: 01 001

0700:0503: 02 002

0700:0504: 03 003

0700:0505: 05 005

0700:0506: 08 008

0700:0507: 0D 013

0700:0508: 15 021

0700:0509: 22 034

0700:050A: 37 055

0700:050B: 59 089

0700:050C: 8F 143

Conclusion:

In conclusion, assembly language is a very useful language. For instance, it makes understanding concepts easier and make awareness of what's going to happen closely inside the hardware of the computer such how data represents in the memory, how the processor accesses and executes instructions, and how a program accesses external device. Low language programming or assembly language is used to write such that Fibonacci series generator that has been explained earlier.

References:

1. Assembly programming/assembly instructions – TutorialsPoint.com
2. Microprocessor 8086 – TutorialsPoint.com
3. 8086-microprocessor – JavaTPoint.com