CS 271 Computer Architecture

Lab1 (Due on Oct. 20, 4:30pm in ET-125E)

The following problems are based on Chapter 4 of Textbook "Essentials of Computer Organization and Architecture" by Linda Null. Jones & Bartlett Pub. 4th edition.

Please Download "MARIE and Datapath Simulators" package from the following link: http://computerscience.jbpub.com/ecoa/3e/simulators.aspx

In the package, there are four examples provided.

Use the simulator to write the MARIE assembly code and show the running results. In the submission, please hand in both source code (i.e., MARIE assembly language code) and the running results (i.e., screen shot) in MARIE simulator.

(1) Find the seventh value of the Fibonacci number sequence. The following formula describes the sequence:

$$Fib(1) = 1$$
, $Fib(2) = 1$, $Fib(n) = Fib(n-1) + Fib(n-2)$.

The program calculates Fib(7).

- (2) Print "Hello CS271" to output.
- (3) In textbook, page 286, problem 29. Consider two cases for the initial value of X: a) X = 1; b) X = 2. Assume Y = 0 initially.
- (4) Create a subroutine to implement the following statement in C language:

int num = condition ? First number : Second number

That is, if "condition" is 0, "num" gets the value of "Second number"; otherwise, it gets

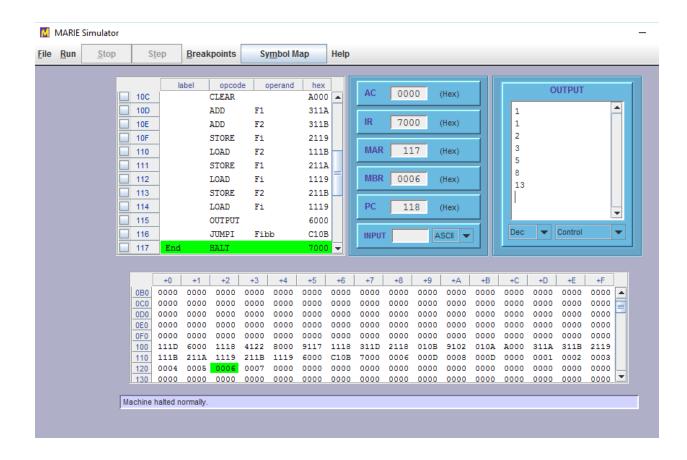
the value of "First number". Consider two second of an dition = 1 th beautition = 0.

the value of "First number". Consider two cases: a) condition = 1; b) condition = 0.

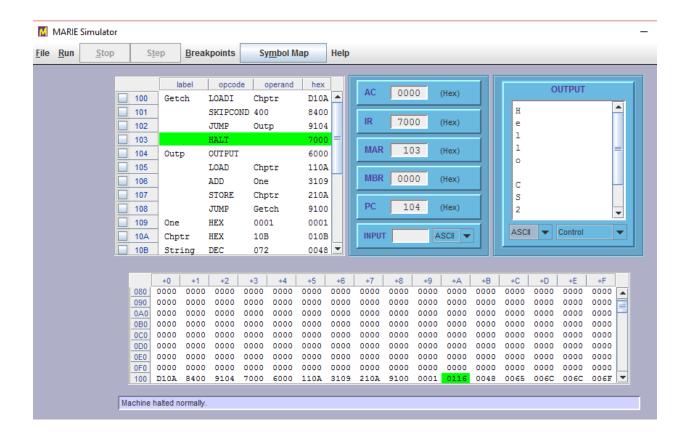
Assume "First number" is 10, and "Second number" is 20.

Please submit the hard copy of this lab, as well the running results in MARIE simulator by outputting the results to Output window or circling the results shown in register or in memory.

```
(1) / Lab 1.1
    / CS 271
    / 1
    / 1
    / 1+1 = 2
   / 1+2 = 3
   / 2+3 = 5
    / 3+5 = 8
    / 5+8 = 13 <-- Target
    / 8+13 = 21
   ORG 100
           LOAD
                         ONE / Load 1 into AC.
           Output
                                 / Output 1.
                         COUNT / Load Count into AC
          LOAD
    Cond,
           SUBT
                         SIX / Remove 6 from the count
           SKIPCOND
                        000 / Skip next instruction if AC < 0
           JUMP
                         End / Jump to the End Section which ends the loop.
                         COUNT / Load Count into AC.
    Loop,
          LOAD
           ADD
                         ONE
                                 / Increment Count by 1
           STORE
                        COUNT / Store AC in count
                        Fibb / Jump to Fibb section and store JNS.
           JUMP
                         Cond / Jump to Cond section.
    Fibb,
           HEX
                         000
                                 / Store value for JNS.
           CLEAR
                                 / Clear the AC.
           ADD
                        F1
                                / Add F1 to AC.
           ADD
                        F2
                               / Add F2 to AC.
           STORE
                        Fi
                                / Store AC contents into Fi.
                        F2
                                 / Load F2 into AC.
          LOAD
           STORE
                         F1
                                 / Store F2 into F1.
                               / Load Fi into AC.
          LOAD
                         Fi
                        F2
          STORE
                               / Store Fi into F2.
          LOAD
                        Fi
                                / Load Fi into AC.
           OUTPUT
                                 / Output the current Fibonaci.
           JUMPI
                        Fibb
                               / Jump back to caller.
   End,
           HALT
                                 / Terminate program.
   COUNT, DEC
                                / declare variable to hold count for loop
                        0
   Fi,
          DEC
                         0
                                 / declare variable to hold the result of F1+F2
   F1,
          DEC
                         0
                                 / declare variable to hold the Fib(n-1)
                                / declare variable to hold the Fib(n-1)
           DEC
                         1
    F2,
                               / declare variable to hold 0
   ZERO, DEC
                         0
   ONE,
          DEC
                         1
                               / declare variable to hold 1
   TWO,
          DEC
                        2
                                / declare variable to hold 2
   THREE, DEC
                         3
                                 / declare variable to hold 3
    FOUR, DEC
                         4
                                 / declare variable to hold 4
                               / declare variable to hold 5
    FIVE, DEC
                         5
   SIX,
          DEC
                        6
                                 / declare variable to hold 6
```

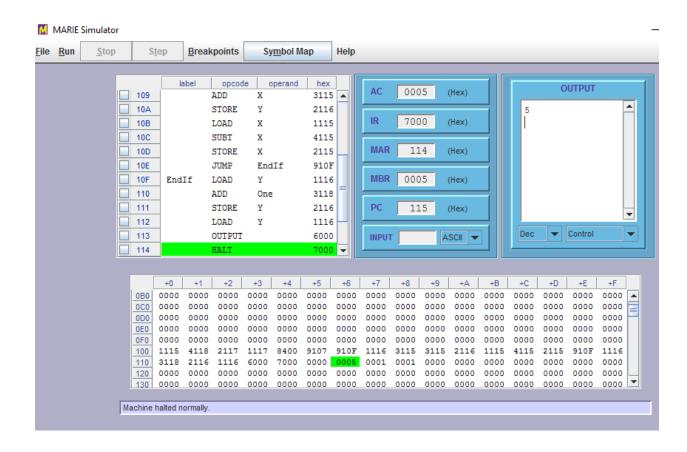


```
/ Matthew Hunt
/ 10/20/2016
/ Lab 1.2
/ CS 271
       ORG 100
                               / Store information in memory starting at address 100.
Getch, LoadI
                  Chptr
                              / Load the character found at address chptr.
       Skipcond
                  400
                               / If the character is a null, we are done.
       Jump
                  Outp
                               / Otherwise, jump to output section.
       Halt
                               / End Operation
Outp,
       Output
                              / Output the character.
       Load
                  Chptr
                              / Load the character pointer.
                              / Add one to the character pointer.
       Add
                  One
       Store
                  Chptr
                              / Store the character pointer.
       Jump
                  Getch
                               / Jump back to get character section.
One,
       Hex
               0001
                              / Declare hex varriable and assign it to 1.
               10B
Chptr, Hex
                              / Declare hex varriable and assign it to 10B.
String, Dec
               072
                             / Declare decimal varriable and assign it to ASCII value of H.
                              / Declare decimal varriable and assign it to ASCII value of e.
       Dec
               101
               108
                              / Declare decimal varriable and assign it to ASCII value of 1.
       Dec
                              / Declare decimal varriable and assign it to ASCII value of 1.
       Dec
               108
       Dec
               111
                              / Declare decimal varriable and assign it to ASCII value of o.
               032
                              / Declare decimal varriable and assign it to ASCII value of [space].
       Dec
       Dec
               067
                              / Declare decimal varriable and assign it to ASCII value of C.
               083
                              / Declare decimal varriable and assign it to ASCII value of S.
       Dec
               050
       Dec
                              / Declare decimal varriable and assign it to ASCII value of 2.
               055
                              / Declare decimal varriable and assign it to ASCII value of 7.
       Dec
       Dec
               049
                              / Declare decimal varriable and assign it to ASCII value of 1.
               000
                              / Declare decimal varriable and assign it to ASCII value of [null].
       Dec
       END
                               / End of file.
```



```
(3)
```

```
/ Matthew Hunt
/ 10/20/2016
/ Lab 1.3
/ CS 271
/ if X > 1 then
      Y = X + X;
     X = 0;
/ endif;
/ Y = Y + 1;
/ Case: X = 0 then Y should be 1
/ Case: X = 1 then Y should be 1
/ Case: X = 2 then Y should be 5
       ORG 100
If,
      Load
                    X / Load X into AC.
       Subt
                    One
                             / Subtract one from X.
       Store
                     Temp / Store X-1 in Temp.
       Load
                     Temp / Load temp into AC
                     400
                             / If AC = 0, skip the next instruction
       Skipcond
       Jump
                     Then / Jump to Then section.
                     EndIf / Jump to EndIf section.
       Jump
Then, Load
                     Y
                             / Load Y into AC.
       Add
                     Х
                             / Y+X
       Add
                    X
                             / Y+X
                    Y
                             / Store AC into Y.
       Store
       Load
                    X
                             / Load X into AC.
       Subt
                     X
                             / X-X
                     Х
                             / Set X to zero.
       Store
       Jump
                     EndIf / Jump to EndIf section.
EndIf, Load
                     Y
                             / Load Y into AC.
       Add
                             / Y+1
                     One
                             / Store Y+1
                     Y
       Store
       Load
                     Y
                             / Load Y into AC.
                             / Output AC to console.
       Output
       Halt
                             / Terminate program
                    02
                             / Declare variable to hold X.
Х,
       Dec
Υ,
       Dec
                     00
                             / Declare variable to hold Y.
                     00
                             / Declare variable to hold temporary value of X.
Temp,
       Dec
One,
       Dec
                     01
                             / Declare varriable to hold one.
       END
```



```
1 #include <stdio.h>
   3 /*Create a subroutine to implement the following statement in C language:
   4 int num = condition ? First number : Second number
   That is, if "condition" is 0, "num" gets the value of "Second number"; otherwise, it gets the value of "First number". Consider two cases: a) condition = 1; b) condition = 0.
       Assume "First number" is 10, and "Second number" is 20.
   8
   9
  10
       int x = 10;
  11
       int y = 20;
  12
  13
       int check_condition(int condition)
  14
  15
            if(condition == 0) {
  16
                return x;
  17
            }else{
  18
                return y;
  19
  20
      }
  21
  22
      int main()
  23
       {
  24
            int condition;
  25
            condition = 0;
  26
  28
           printf("%d\n", check_condition(condition));
  29
      }
  30
  bash - "masterprogr: ×
                                                    application.c - Stopp ×
                           Immediate
Run
                                                Command:
                                                              application.c
Running "/home/ubuntu/workspace/application.c"
```

```
/ Matthew Hunt
/ 10/20/2016
/ Lab 1.4
/ CS 271
/ Case: cond = 0 then Y should be output
/ Case: cond = 1 then X should be output
       ORG
             100
       Load X
                    / Load X into AC
       Store Temp
                     / Store X in Temp.
       Load Cond / Load Cond into AC.
       Skipcond 400 / Skip next instruction if X is zero.
       JnS
            Subr / Store the return address, and jump to the procedure.
       Load Y
                    / Load Y into Ac.
       Store Temp / Store Y in Temp.
       Load Cond / Load Cond into AC
       Skipcond 800 / Skip if Cond is 1.
       JnS Subr / Store the return address and jump to the procedure.
                    / Load Num into AC.
       Load Num
       Output
                     / Output Num to console.
       Halt
                     / End program.
                  / Declare varriable to hold X.
Х,
       DEC 10
Υ,
       DEC
             20
                    / Declare varriable to hold Y.
                    / Declare varriable to hold the number to output
Num,
       DEC
             00
Cond,
      DEC 01
                    / Declare varriable to hold the condition of 1 or 0.
      DEC
            0
                     / Declare varriable to to temporarily hold X and Y.
Temp,
Subr,
       HEX
             0
                    / Declare varriable to hold return address.
       Load
            Temp / Load Number into AC.
                    / Store Temp in Num.
       Store Num
       JumpI Subr / Return to caller.
       END
```

