



# Computer Systems and Networks

ECPE 170 – Jeff Shafer – University of the Pacific

## MARIE Simulator

# Prelab Setup – MARIE Simulator

- **If you are using your own laptop, make sure Java is installed**
  - <http://java.oracle.com> (Java SE, then download Java 7 JRE)
  
- **Get the MARIE simulator now**
  - ECPE 170 Sakai site under “Resources”
  - *or* Textbook website:
    - <http://computerscience.jbpub.com/ecoa/3e/simulators.aspx>
  
- Today’s goals:
  - Run some sample programs
  - And write your own!

# Recap – MARIE Instructions (Full)

Binary	Hex	Instruction	Meaning
0001	1	<b>LOAD X</b>	Load contents of address X into AC
0010	2	<b>STORE X</b>	Store contents of AC at address X
0011	3	<b>ADD X</b>	Add contents of address X to AC
0100	4	<b>SUBT X</b>	Subtract contents of address X from AC
0101	5	<b>INPUT</b>	Input value from keyboard into AC
0110	6	<b>OUTPUT</b>	Output value in AC to display
0111	7	<b>HALT</b>	Terminate program
1000	8	<b>SKIPCOND</b>	Skip next instruction on condition based on AC value
1001	9	<b>JUMP X</b>	Load value of X into PC
1010	A	<b>CLEAR</b>	Set AC to 0
1011	B	<b>ADDI X</b>	Add contents of address Mem[X] to AC
1100	C	<b>JUMPI X</b>	Load contents of address Mem[X] into PC
1101	D	<b>LOADI X</b>	Load contents of address Mem[X] into AC
1110	E	<b>STOREI X</b>	Store contents of AC at address Mem[X]

See table  
4.7 in  
book!

# MARIE Assembly

## High-Level Language

➔ X = 5  
Y = 7  
Z = X + 7

## Assembly

➔ LOAD X  
ADD Y  
STORE Z  
X, DEC 5  
Y, DEC 7  
Z, DEC 0

# Using the MARIE Simulator

- To use the simulator
  1. Unzip the downloaded archive into a folder on your U: drive
  2. Browse the files and locate MarieSim.jar
- MarieSim is a JAVA application
  - Unless your computer has .JAR files associated with the Java machine, you will need to run the program “by hand”
    - Go to Start Menu, pick “Run...”
    - (or) Run “java -jar <mariesimfile>”



# Using the MARIE Simulator

- Our programs are written in MARIE assembly language
  - “.mas” files
- Need to use the **assembler** before running (*simulating*) the program!
  - **What does the assembler do again?**
- To start, do “File → Edit”
  - Opens editor
  - Type in your file, or “File → Open” to load
    - Choose file “Ex4\_1.mas”

## MARIE Assembler Code Editor

File Edit Assemble Help

```

Loop,  ORG 100      / Example 4.1
      Load  Addr  /Load address of first number to be added
      Store Next  /Store this address is our Next pointer
      Load  Num   /Load the number of items to be added
      Subt  One   /Decrement
      Store Ctr   /Store this value in Ctr to control looping
      Load  Sum   /Load the Sum into AC
      AddI  Next  /Add the value pointed to by location Next
      Store Sum   /Store this sum
      Load  Next  /Load Next
      Add   One   /Increment by one to point to next address
      Store Next  /Store in our pointer Next
      Load  Ctr   /Load the loop control variable
      Subt  One   /Subtract one from the loop control variable
      Store Ctr   /Store this new value in loop control variable
      Skipcond 000 /If control variable < 0, skip next instruction
      Jump  Loop  /Otherwise, go to Loop
      Halt                /Terminate program

Addr,  Hex      117  /Numbers to be summed start at location 118
Next,  Hex      0    /A pointer to the next number to add
Num,   Dec      5    /The number of values to add
Sum,   Dec      0    /The sum
Ctr,   Hex      0    /The loop control variable
One,   Dec      1    /Used to increment and decrement by 1
      Dec      10   /The values to be added together
      Dec      15
      Dec      20
      Dec      25
      Dec      30

```

/ECPE 170 Computer Systems and Networks/Instructor Resources - Third Edition/MARIE\_Datapath\_Simulators/Ex4\_1.ma



# Using the MARIE Simulator

- Assembly file format:
  - **Labels:** define addresses we want to access
    - End with a comma (,)
  - **Opcode:** the operation to perform
  - **Operands:** other data needed by the instruction
  - **Comments:** you know, comments
    - Start with / in Marie

Typical MARIE line: (Label is optional)

```
Label, opcode operands / comments
```

# Special Commands

- **What is DEC? HEX? ORG? END?**
  - **Are they assembly commands for the processor?**
- **No – these are commands for the assembler only!**
  - **DEC X** – The number to follow is written in base 10 (please convert to binary)
  - **HEX X** – The number to follow is written in base 16 (please convert to binary)
  - **ORG X** – Please store this program in memory starting at memory address X (in Hex)
  - **END** - Stop Assembly (finished!)

# Using the MARIE Simulator

- Ready to run simulator?
- Assemble source code
  - “Assemble → Assemble Current File”
- Files produced by assembler
  - .lst file = Original assembly code + machine code
  - .map file = Symbol table from assembly process
  - .mex file = Machine code (only)
- Errors? Listing file will indicate line and problem
- No errors? Ready to simulate!

Assembly listing for: Ex4\_2.mas

Assembled: Mon Oct 03 10:37:06 PDT 2011

```

                                     / Example 4.1
100 ?10C   If      ORG 100
                LOADX X           /Load the first value
        **** Instruction not recognized.
101 410D   SUBT Y           /Subtract the value of Y, store result in AC
102 8400   SKIPCOND 400     /If AC=0, skip the next instruction
103 9108   JUMP Else       /Jump to Else part if AC is not equal to 0
104 110C   Then  LOAD X     /Reload X so it can be doubled
105 310C   ADD X           /Double X
106 210C   STORE X        /Store the new value
107 910B   JUMP Endif      /Skip over the false, or else, part to end of if
108 110D   Else  LOAD Y     /Start the else part by loading Y
109 410C   SUBT X         /Subtract X from Y
10A 210D   STORE Y        /Store Y-X in Y
10B 7000   Endif  HALT      /Terminate program (it doesn't do much!)
10C 000C   X      DEC 12    /Load the loop control variable
10D 0014   Y      DEC 20    /Subtract one from the loop control variable
                END

```

1 error found. Assembly unsuccessful.

## SYMBOL TABLE

Symbol	Defined	References
Else	108	103
Endif	10B	107
If	100	
Then	104	
X	10C	100, 104, 105, 106, 109
Y	10D	101, 108, 10A

# Using the MARIE Simulator

- To simulate, “File → Load”
  - Pick the **.mex** file created by the assembler
- Code shows up in upper left window
  - Addresses shown on the left
  - Machine code shown on the right
- Registers shown in the middle
- Output (from OUTPUT instruction) on right
- Bottom windows shows “memory dump”

# Using the MARIE Simulator

- Ways to simulate
  - **Run:** run continuously until you choose “Stop” or CPU executes a HALT
  - **Step**
    - Choose “Run → Set stepping mode → on” first
    - Lets you examine one instruction at a time
  - **Breakpoints**
    - Lets you pick instructions to stop at
    - Check the box next to the instructions’ address
    - Choose “Breakpoints → Run to Breakpoints”

# Lab Exercise 1

- **With a partner (*if desired*), take 5 minutes to:**
  - **Examine** the assembly code in the file Ex4\_1.mas (already open)
  - **Write the equivalent C++** (or Java, or pseudocode) for the operation being performed
  
- Equivalent code doesn't have to be perfect
  - You could write several different C++ programs that accomplish the same tasks!

# Lab Exercise 1

```
int myArray[5]={10,20,30,40,50};
int num=5;
int counter=0;
int sum=0;

counter = num - 1;
do
{
    sum = sum + myArray[counter];
    counter = counter - 1;
} while(counter >=0)
```



```

ORG 100
Load Addr      /Load address of first number to be added
Store   Next   /Store this address is our Next pointer
Load     Num   /Load the number of items to be added
Subt  One     /Decrement
Store   Ctr    /Store this value in Ctr to control looping
Loop,   Load Sum /Load the Sum into AC
AddI Next     /Add the value pointed to by location Next
Store   Sum   /Store this sum
Load Next    /Load Next
Add  One     /Increment by one to point to next address
Store  Next  /Store in our pointer Next
Load  Ctr    /Load the loop control variable
Subt  One    /Subtract one from the loop control variable
Store  Ctr   /Store this new value in loop control variable
Skipcond 000 /If control variable < 0, skip next instruction
Jump Loop   /Otherwise, go to Loop
Halt       /Terminate program
Addr,     Hex 117 /Numbers to be summed start at location 118
Next,     Hex  0 /A pointer to the next number to add
Num,     Dec  5 /The number of values to add
Sum,     Dec  0 /The sum
Ctr,     Hex  0 /The loop control variable
One,     Dec  1 /Used to increment and decrement by 1
         Dec 10 /The values to be added together
         Dec 15
         Dec 20
         Dec 25
         Dec 30

```

# Lab Exercise 2

- **With a partner (*if desired*), write and run a complete MARIE assembly program to accomplish the follow task:**

```
if X==Y then
    X = X * 2;
else
    Y = Y - X;
```

**Show me the running program with  $X=12_{10}$ ,  $Y=20_{10}$**

# Lab Exercise 2

```
ORG 100
If,   LOAD    X /Load the first value
      SUBT    Y /Subtract the value of Y, store result in AC
      SKIPCOND 400 /If AC=0, skip the next instruction
      JUMP    Else /Jump to Else part if AC is not equal to 0
Then, LOAD    X /Reload X so it can be doubled
      ADD     X /Double X
      STORE   X /Store the new value
      JUMP    Endif /Skip over the false (else) part to end of if
Else, LOAD    Y /Start the else part by loading Y
      SUBT    X /Subtract X from Y
      STORE   Y /Store Y-X in Y
Endif, HALT    /Terminate program (it doesn't do much!)
X,    Dec 12
Y,    Dec 20
      END
```

# Homework #8

- **Exercises 4.28 + 4.29**
  - Work individually **or in teams of 2**
  - Each person must submit assignment!
  - Put your name and partner's name in comments
  
- Sakai submission
  - Turn in each “.mas” source file **separately**
  - Name them “ex428.mas”, “ex429.mas”, ...
  
- Files should be PLAIN ASCII TEXT (use NotePad or MARIE editor)
  - **Zero points** if you give me a .doc, .docx, .pdf, scanned copy of a printout, smoke signals, etc...
  
- **You MUST comment your code! (at least 90% of the lines!)**
  - **No points for uncommented code**