CMPE224 Dıgıtal Logic Systems

MARIE Assembly Language Programming Exercise Samples

IMPORTANT NOTES:

I. Prepare solutions of questions 5 and 6, and demonstrate your solutions to lab. Assistants on 21 MAY, 2019 Tuesday, at 16:30 in CMPE227.

II. Prepare solutions of questions 8 and 10, and demonstrate your solutions to lab. Assistants on 28 MAY, 2019 Tuesday, at 16:30 in CMPE227.

Q.1. Write a Marie Assembly Language Program (MALP) to compute the first 10 elements of the following sequence: $F_0=0$, $F_1=1$, $F_2=3$, $F_i=F_{i-1}+F_{i-2}+F_{i-3}$, $i > 2$.

Q.2. Write a MALP, with a procedure of DivideByTwo, to check if an input integer is either even or odd.

Q.3. Write a MALP to compute the expression z= (x*y+3x+3y)/x*y -2x -2y) for x=20 and y=10. Keep the quotient and remainder of division in variablea q and r, respectively. Output values of q and r before termination.

Q.4. Input numbers A, B, C and D from input register and perform computations A-B, A-B-C and A-B-C-D using a single procedure SubtractXY which computes (X-Y).

Q.5. (EXP-1) Write a MALP to print the string "I Love CMPE224" to output area.

Q.6. (EXP-1) Write a MALP to compute the sum of integers in the array A=[2,3,5,8,4,8,1,9,3]

Q.7. Write a MALP to compute the sum of elements in two arrays A and B that are defined as follows:

A=[1,2,3,4,5], B=[6,7,8,9,10]. Store the result in Array C.

Q.8. (EX-2) Write a MALP to compute maximum of elements in the two arrays A and B. That is, $C_i=max(A_i,B_i)$.

Q.9. Write a MALP to compute the absolute values of elements within an array A.

Q.10. (EXP-2) Write a MALP to compute 2's complements of numbers stored in an array A.

MARIE INSTRUCTION SET

Type	Instruction	Hex Opcode	Summary
Arithmetic	Add X	3	Adds value in AC at address X into AC, AC ← AC + X
	Subt X	4	Subtracts value in AC at address X into AC, AC ← AC - X
	AddI X	B	Add Indirect: Use the value at X as the actual address of the data operand to add to AC. AC←M[M[X]]
	Clear	A	AC ← 0
Data Transfer	Load X	1	Loads Contents of Address X into AC. AC←M[X]
	Store X	2	Stores Contents of AC into Address X. M[X]←AC.
I/O	Input	5	Request user to input a value. AC←InREG
	Output	6	Prints value from AC. OutREG ←AC.
Branch	Jump X	9	Jumps to Address X. PC←M[X]
	Skipcond (C)	8	Skips the next instruction based on C: if (C) = - 000: Skips if AC < 0 - 400: Skips if AC = 0 - 800: Skips if AC > 0
Subroutine	JnS X	0	Jumps and Store: Stores value of PC at address X then increments PC to X+1. M[X]←PC, PC←M[X]+1
	JumpI X	C	Uses the value at X as the address to jump to. PC←M[M[X]]
Indirect Addressing	StoreI X	D	Stores value in AC at the indirect address. e.g. StoreI addresspointer Gets value from addresspointer, stores the AC value into the address. M[M[X]] ←AC
	LoadI X	E	Loads value from indirect address into AC e.g. LoadI addresspointer Gets address value from addresspointer, loads value at the address into AC AC←M[M[X]]
	Halt	7	End the program