

Joseph Godinez

Professor Oscar Ho

Computer Science 240-20

October 28, 2021

- 1.
2. The program is storing positive values into Register 0 and adding them in R0

The screenshot shows the LC3 Simulator interface. The main window displays the state of the LC3 registers and the instruction stream. A breakpoint has been encountered at PC = x300B. The registers are as follows:

Register	Value
R0	x4000
R1	x0000
R2	x8000
R3	x0000
R4	x0000
R5	x0000
R6	x0000
R7	x0000

The instruction stream shows the following instructions:

```
* x3000 0010000000001011 x200B LD R0, NUMBERS
* x3001 0010010000001011 x240B LD R2, MASK
* x3002 0110001000000000 x6200 LOOP LDR R1, R0, #0
* x3003 0000010000000111 x0407 BRZ DONE
* x3004 0101101001000010 x5A42 AND R5, R1, R2
* x3005 0000010000000001 x0401 BRZ L1
* x3006 0000111000000010 x0E02 BRNZP NEXT
* x3007 0001001001000000 x0000 R1, R1, R1
* x3008 0111001000000000 x0000 R1, R0, #0
* x3009 0001000000100000 x0000 R0, R0, #1
* x300A 0000111111111100 x0000 RZP LOOP
* x300B 1111000000100000 x0000 LP HALT
* x300C 0100000000000000 x0000 IR R0
* x300D 1000000000000000 x0000 ?
* x300E 0000000000000000 x0000 ?
* x300F 0000000000000000 x0000 ?
* x3010 0000000000000000 x0000 ?
* x3011 0000000000000000 x0000 NOP
* x3012 0000000000000000 x0000 NOP
* x3013 0000000000000000 x0000 NOP
* x3014 0000000000000000 x0000 NOP
* x3015 0000000000000000 x0000 NOP
* x3016 0000000000000000 x0000 NOP
* x3017 0000000000000000 x0000 NOP
* x3018 0000000000000000 x0000 NOP
* x3019 0000000000000000 x0000 NOP
* x301A 0000000000000000 x0000 NOP
* x301B 0000000000000000 x0000 NOP
* x301C 0000000000000000 x0000 NOP
* x301D 0000000000000000 x0000 NOP
* x301E 0000000000000000 x0000 NOP
* x301F 0000000000000000 x0000 NOP
```

The status bar at the bottom indicates "0 instructions executed" and "Idle".

3. The program in 7.16 clears Register 3 and 4 then the loop starts. Basically the programs count all the even and odd integers. Which are then stored in R3 and R4 the value of how many even and odd numbers.

LC3 Simulator - ProblemSet4\_7\_16.obj

File Execute Simulate Help

Jump to: x3000

PC	R0	R1	R2	R3	R4	R5	R6	R7	PC	IR	PSR	CC
x3000	0101100100100000	x5920	AND	R4, R4, #0					x300D	12301		
x3001	0101011011100000	x5E00	AND	R3, R3, #0					x0407	1031		
x3002	0010000000001011	x200B	LD	R0, NUMBERS					x8002	-32766		
x3003	0110001000000000	x6200	LDR	R1, R0, #0								
x3004	1001010001111111	x947F	NOT	R2, R1								
x3005	0000010000000111	x0407	BRZ	DONE								
x3006	0101010001100001	x5461	AND	R2, R1, #1								
x3007	0000010000000000		L1									
x3008	0001100100100100		L1	R4, R4, #1								
x3009	0000111000000000		IFZ	NEXT								
x300A	0001011011100100			R3, R3, #1								
x300B	0001000000010000			R0, R0, #1								
x300C	0000111111111100		IFZ	LOOP								
x300D	1111000000010000		IF	HALT								
x300E	0100000000000000		IR	R0								
x300F	0000000000000000											
x3010	0000000000000000											
x3011	0000000000000000	x0000	NOP									
x3012	0000000000000000	x0000	NOP									
x3013	0000000000000000	x0000	NOP									
x3014	0000000000000000	x0000	NOP									
x3015	0000000000000000	x0000	NOP									
x3016	0000000000000000	x0000	NOP									
x3017	0000000000000000	x0000	NOP									
x3018	0000000000000000	x0000	NOP									
x3019	0000000000000000	x0000	NOP									
x301A	0000000000000000	x0000	NOP									
x301B	0000000000000000	x0000	NOP									
x301C	0000000000000000	x0000	NOP									
x301D	0000000000000000	x0000	NOP									
x301E	0000000000000000	x0000	NOP									
x301F	0000000000000000	x0000	NOP									

Breakpoint Encountered

PC = x300D

OK

4. The missing lines are {LDR R3, R4, #0| NOT R3, R3| ADD R3, R3, #1} LDR is loading the address first. Then compare the characters and last is to add the characters in R3.

LC3 Simulator - ProblemSet4\_7\_18.obj

File Execute Simulate Help

Jump to: x3000

PC	R0	R1	R2	R3	R4	R5	R6	R7	PC	IR	PSR	CC
x3000	0100010000100000	x2210	LD	R1, FIRST					x3010	12304		
x3001	0010010000010000	x2410	LD	R2, SECOND					x1B61	7009		
x3002	0101000000100000	x5020	AND	R0, R0, #0					x8001	-32767		
x3003	0110011001000000	x6640	LDR	R3, R1, #0								
x3004	0110100010000000	x6880	LDR	R4, R2, #0								
x3005	0000010000000100	x0408	BRZ	NEXT								
x3006	0001001001100001	x1261	ADD	R1, R1, #1								
x3007	0001010010100100			R2, R2, #1								
x3008	1001011011111111			R3, R3								
x3009	0001011011100100			R3, R3, #1								
x300A	0001011011100100			R3, R3, R4								
x300B	0000010111111100			LOOP								
x300C	0101101101100100			R5, R5, #0								
x300D	0000111000000000		IFZ	DONE								
x300E	0101101101100100			R5, R5, #0								
x300F	0001101101100100			R5, R5, #1								
x3010	1111000000010000		IF	HALT								
x3011	0100000000000000	x4000	JSRR	R0								
x3012	0100000100000000	x4100	JSRR	R4								
x3013	0000000000000000	x0000	NOP									
x3014	0000000000000000	x0000	NOP									
x3015	0000000000000000	x0000	NOP									
x3016	0000000000000000	x0000	NOP									
x3017	0000000000000000	x0000	NOP									
x3018	0000000000000000	x0000	NOP									
x3019	0000000000000000	x0000	NOP									
x301A	0000000000000000	x0000	NOP									
x301B	0000000000000000	x0000	NOP									
x301C	0000000000000000	x0000	NOP									
x301D	0000000000000000	x0000	NOP									
x301E	0000000000000000	x0000	NOP									
x301F	0000000000000000	x0000	NOP									

Breakpoint Encountered

PC = x3010

OK