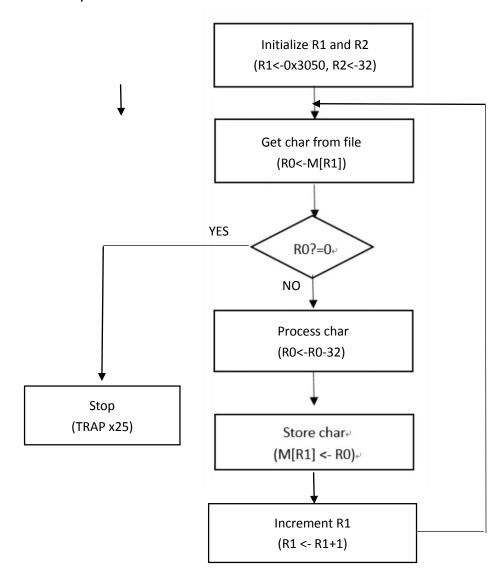
Report for Lab1 Student Name:Jin Zewen Student Number:PB15111604

Part 1 Purpose

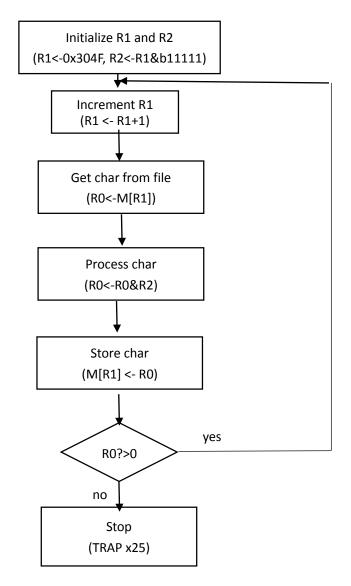
In this assignment, I'm asked to write a program in LC-3 machine language that converts a string of lower case letters in the memory into uppercase and stores the resulting value in memory.

Part 2+3 Principles+Procedures

For this lab, we need only tow general purpose register R0 which will contain the value of the processed character, R1, which will be regarded as the pointer for the char to be processed and R2, which will be used to calculate R0-32. The flowchart of the algorithm that solves the problem can be as follows, which needs 9 or 10 instructions.



Then after several hours, I found the solution with only 8 instructions, with the flowchart of the algorithm that solves the problem as follows.



Some details:

Because the chars will start from b 0110 0001 to b 0111 1010, the 6th bits from right side are all 1,which means R0-32 can be realized by R0 & b 1111 1111 1101 1111. So in this method, a mask is used. Since the first address for characteristics is 0x3050, the previous will be 0x304f, i.e. b0011 0000 0100 1111, which can be used as a mask for R2, which should result in R0 & R2 being R0-32.

Here is my code:

0011 0000 0000 0000; start at 0x3000 1110 0010 0100 1110; LEA R1, 0x304f 0101 0100 0111 1111; ADD R2, R1, #b11111 0001 0010 0110 0001; ADD R1, #1 0110 0000 0100 0000; LDR R0, R1 #0

0101 0000 0000 0010; AND R0, R0, R2

0111 0000 0100 0000 ; STR R0, R1, #0 0000 0101 1111 1011 ; BRZ 0x3002 1111 0000 0010 0101 ; TRAP x25

The result of the compilation

```
File Edit Translate Help

0911 0909 0909 0909 ; start at 0x3090
1119 0919 0109 1110 ; LEA R1, 0x304f
6101 0109 0111 1111 ; AND R2, R1, #b11111
0901 0919 0119 0901 ; ADD R1, #1
0119 0909 0109 0909 ; LDR R0, R1 #9
0101 0909 0909 0910 ; AND R0, R0, R2
0111 0909 0109 0909 ; STR R0, R1, #9
0909 0101 1111 1011 ; BRZ 0x3092
1111 0909 0919 0101 ; TRAP x25

Converting D:\csapp\CS\LAB1\lab01.bin from base 2...
Convert complete - 0 error(s)
```

1	- A	ZEEL		UU	AUUUU	NOF			
	- X	2FFF	000000000000000000000000000000000000000	00	x0000	NOP			
	- x	3000	11100010010011	10	xE24E	LEA	R1,	x304	1F
	- x	3001	01010100011111	11	x547F	AND	R2,	R1,	#-1
	- x	3002	00010010011000	01	x1261	ADD	R1,	R1,	#1
	- x	3003	01100000010000	00	x6040	LDR	RO,	R1,	#0
	- x	3004	01010000000000	10	x5002	AND	RO,	RO,	R2
	- x	3005	01110000010000	00	x7040	STR	RO,	R1,	#0
	- x	3006	00000011111110	11	x03FB	BRP	x300)2	
	- x	3007	11110000001001	.01	xF025	TRAP	HALT	1	
	- x	3008	0000000000000000	00	x0000	NOP			
	- x	3009	0000000000000000	00	x0000	NOP			
	- x	300A	0000000000000000	00	x0000	NOP			
	- x	300B	0000000000000000	00	x0000	NOP			
		2000	0000000000000000	000	***	MOD			

Part 4 Result.

Set the values from 0x3050 IN DATA.hex as follows: .

1	3050
2	0061
3	0062
4	0063
5	0064
6	0065
7	0066
8	0067
9	0068
10	0069
11	006A
12	0000
13	0079

run it.:

x3050	0000000001100001	x0061	x3050	0000000001000001	x0041	NOP
x3051	0000000001100010	x0062	x3051	0000000001000010	x0042	NOP
x3052	0000000001100011	x0063	x3052	0000000001000011	x0043	NOP
x3053	0000000001100100	x0064	x3053	0000000001000100	x0044	NOP
x3054	0000000001100100	x0065	x3054	0000000001000101	x0045	NOP
x3055	0000000001100101	x0065	x3055	0000000001000110	x0046	NOP
			x3056	00000000001000111	x0047	NOP
x3056	0000000001100111	x0067	x3057	0000000000100111	x0047	NOP
x3057	0000000001101000	x0068				
x3058	0000000001101001	x0069	x3058	0000000001001001	x0049	NOP
x3059	0000000001101010	x006A	x3059	0000000001001010	x004A	NOP
x305A	00000000000000000	x0000	x305A	000000000000000000	x0000	NOP
x305B	0000000001111001	x0079	x305B	0000000001111001	x0079	NOP

The result is correct.