SYSC 4507 Assignment 2

Jessica Morris 100882290

February 13th, 2017

1. If the item is at array[3], then the while loop will execute 2 times. First execution has R6 = 0 and R5 = 17 and checks index 8, the second execution has R6 = 0 and R5 = 7 and checks index 3.

   The total number of instructions executed is: $5 + 10 + 10 = 25$

2. To execute this program, it would take:

$$n_{clocks} = 5 \text{ CPI} \times n_{LDR} + 4 \text{ CPI} \times n_{other}$$
$$n_{clocks} = 5 \text{ CPI} \times 2 \text{ instructions} + 4 \text{ CPI} \times 23 \text{ instructions}$$
$$n_{clocks} = 102 \text{ clocks}$$

3. The program takes 56 clock cycles when using a 5-stage pipeline. See space-time diagrams on the next few pages.

4. A one-slot delayed-branch will only save one clock cycle. With the first BLT all_done, before the start of the while loop, the MOV R8, #-1 could be executed in the delay. However, the remaining branches could not be refactored to take advantage of the one-slot delay, as the other instructions all depend on each other.

Cycle	IF	ID	EX	MEM	WB	Notes
1	MOV R8,#-1					
2	CMP R5,0	MOV R8,#-1				
3	nop	CMP R5,0	MOV R8,#-1			
4	nop	nop	CMP R5,0	MOV R8,#-1		
5	BLT all_done	nop	nop	CMP R5,0	MOV R8,#-1	
6	MOV R6,#0	BLT all_done	nop	nop	CMP R5,0	
7	SUB R5,R5,#1	MOV R6,#0	BLT all_done	nop	nop	Branch not taken
8	nop	SUB R5,R5,#1	MOV R6,#0	BLT all_done	nop	
9	nop	nop	SUB R5,R5,#1	MOV R6,#0	BLT all_done	
10	CMP R5,R6	nop	nop	SUB R5,R5,#1	MOV R6,#0	
11	nop	CMP R5,R6	nop	nop	SUB R5,R5,#1	
12	nop	nop	CMP R5,R6	nop	nop	
13	BLT all_done	nop	nop	CMP R5,R6	nop	
14	ADD R7,R6,R5	BLT all_done	nop	nop	CMP R5,R6	
15	nop	ADD R7,R6,R5	BLT all_done	nop	nop	Branch not taken
16	nop	nop	ADD R7,R6,R5	BLT all_done	nop	
17	LSR R7,R7,#1	nop	nop	ADD R7,R6,R5	BLT all_done	
18	nop	LSR R7,R7,#1	nop	nop	ADD R7,R6,R5	
19	nop	nop	LSR R7,R7,#1	nop	nop	
20	LDR R9,[R4,R7]	nop	nop	LSR R7,R7,#1	nop	
21	nop	LDR R9,[R4,R7]	nop	nop	LSR R7,R7,#1	
22	nop	nop	LDR R9,[R4,R7]	nop	nop	
23	CMP R9,R3	nop	nop	LDR R9,[R4,R7]	nop	
24	nop	CMP R9,R3	nop	nop	LDR R9,[R4,R7]	
25	nop	nop	CMP R9,R3	nop	nop	
26	BLT adjust_min	nop	nop	CMP R9,R3	nop	
27	BGT adjust_max	BLT adjust_min	nop	nop	CMP R9,R3	
28	MOV R8,R7	BGT adjust_max	BLT adjust_min	nop	nop	Branch not taken
29	B all_done	MOV R8,R7	BGT adjust_max	BLT adjust_min	nop	Branch taken
30	SUB R5,R7,#1	nop	nop	BGT adjust_max	BLT adjust_min	
31	B test_while	SUB R5,R7,#1	nop	nop	BGT adjust_max	
32	nop	B test_while	SUB R5,R7,#1	nop	nop	Branch early
33	CMP R5,R6	nop	B test_while	SUB R5,R7,#1	nop	
34	nop	CMP R5,R6	nop	B test_while	SUB R5,R7,#1	
35	nop	nop	CMP R5,R6	nop	B test_while	
36	BLT all_done	nop	nop	CMP R5,R6	B test_while	

37	ADD R7,R6,R5	BLT all_done	nop	nop	CMP R5,R6	
38	nop	ADD R7,R6,R5	BLT all_done	nop	nop	Branch not taken
39	nop	nop	ADD R7,R6,R5	BLT all_done	nop	
40	LSR R7,R7,#1	nop	nop	ADD R7,R6,R5	BLT all_done	
41	nop	LSR R7,R7,#1	nop	nop	ADD R7,R6,R5	
42	nop	nop	LSR R7,R7,#1	nop	nop	
43	LDR R9,[R4,R7]	nop	nop	LSR R7,R7,#1	nop	
44	nop	LDR R9,[R4,R7]	nop	nop	LSR R7,R7,#1	
45	nop	nop	LDR R9,[R4,R7]	nop	nop	
46	CMP R9,R3	nop	nop	LDR R9,[R4,R7]	nop	
47	nop	CMP R9,R3	nop	nop	LDR R9,[R4,R7]	
48	nop	nop	CMP R9,R3	nop	nop	
49	BLT adjust_min	nop	nop	CMP R9,R3	nop	
50	BGT adjust_max	BLT adjust_min	nop	nop	CMP R9,R3	
51	MOV R8,R7	BGT adjust_max	BLT adjust_min	nop	nop	Branch not taken
52	B all_done	MOV R8,R7	BGT adjust_max	BLT adjust_min	nop	Branch not taken
53	nop	B all_done	MOV R8,R7	BGT adjust_max	BLT adjust_min	
54		nop	B all_done	MOV R8,R7	BGT adjust_max	Branch early
55			nop	B all_done	MOV R8,R7	
56				nop	B all_done	Done