

Course Project - Computer Architecture

180010030

180010032

180020006

Multiply two numbers using Booth's Algorithm in ARM

1.

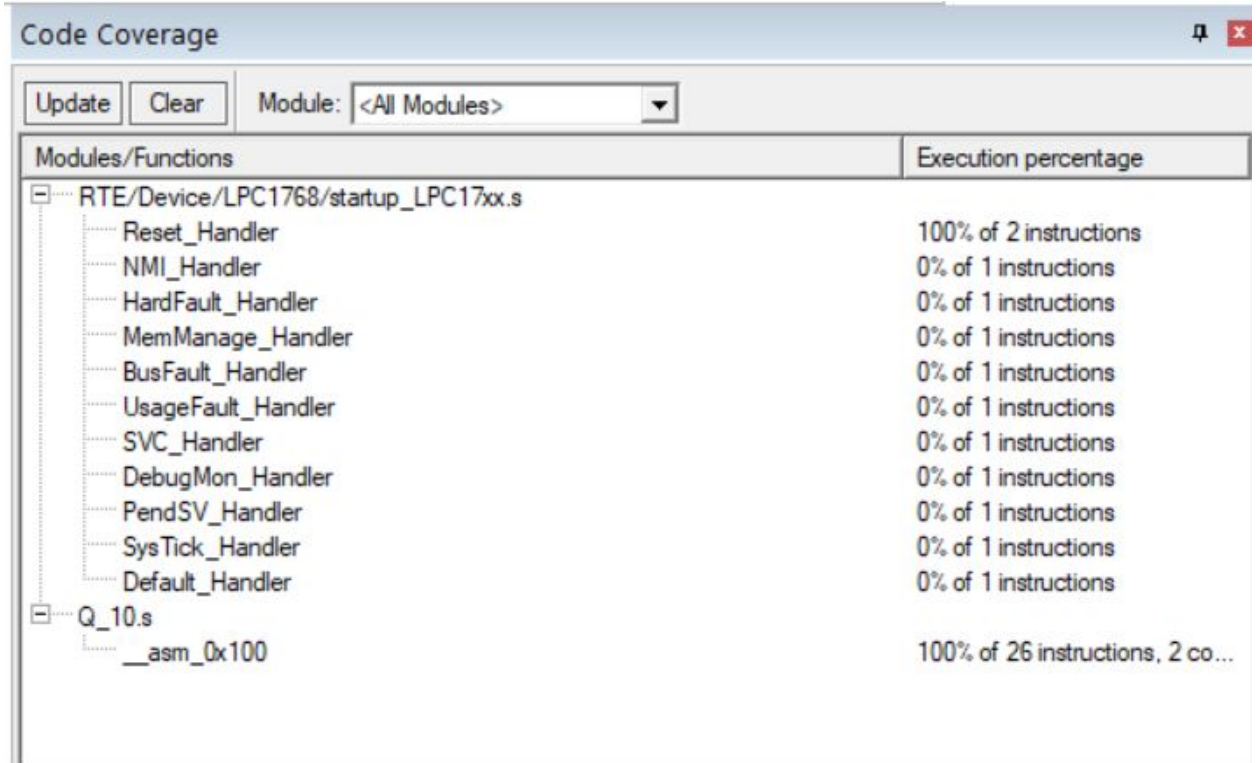
- > There are 23 Static instructions in the code.
- > There are 552 Dynamic instructions in the code.

2.

- > There are 29 instructions if we don't include conditional execution.
- > There are 23 instructions using conditional execution.
- > There are 24 instructions if we write code in THUMB instructions.

3.

>



The image shows a 'Code Coverage' window with a toolbar containing 'Update' and 'Clear' buttons, and a 'Module:' dropdown menu set to '<All Modules>'. The main area is a table with two columns: 'Modules/Functions' and 'Execution percentage'. The table lists several modules and functions, including 'RTE/Device/LPC1768/startup_LPC17xx.s' and 'Q_10.s', along with their respective execution percentages and instruction counts.

Modules/Functions	Execution percentage
[-] RTE/Device/LPC1768/startup_LPC17xx.s	
Reset_Handler	100% of 2 instructions
NMI_Handler	0% of 1 instructions
HardFault_Handler	0% of 1 instructions
MemManage_Handler	0% of 1 instructions
BusFault_Handler	0% of 1 instructions
UsageFault_Handler	0% of 1 instructions
SVC_Handler	0% of 1 instructions
DebugMon_Handler	0% of 1 instructions
PendSV_Handler	0% of 1 instructions
SysTick_Handler	0% of 1 instructions
Default_Handler	0% of 1 instructions
[-] Q_10.s	
__asm_0x100	100% of 26 instructions, 2 co...

> Total execution time :-

Performance Analyzer			
<div>Reset</div> <div>Show: Modules</div>			
Module/Function	Calls	Time(Sec)	Time(%)
+ Assembly_code		59.917 us	2%

4		START
5	0.083 us	MOV R0,#7 ; MULTIPLIER OR B
6	0.083 us	MOV R1,#-1 ; MULTIPLICAND
7	0.083 us	MOV R2,#0 ; A
8	0.083 us	MOV R3,#0 ; Q_-1
9	0.083 us	MOV R4,R1 ; Q AND (AT LAST WILL GIVE F
10	0.083 us	MOV R6,#0 ; COUNT
11		LOOP
12	2.750 us	CMP R6,#32
13	2.917 us	BEQ ENDL
14	2.667 us	AND R7,R4,#1
15	2.667 us	ADD R7,R3,R7,LSL #1 ; Q_0Q_-1
16	2.667 us	CMP R7,#1
17	5.333 us	ADDEQ R2,R2,R0 ; A+B IF EQ
18	2.667 us	CMP R7,#2
19	5.333 us	SUBEQ R2,R2,R0 ; A-B IF EQ
20	2.667 us	AND R7,R4,#1
21	2.667 us	MOV R3,R7 ; Q_-1 SHIFTED
22	2.667 us	MOV R4,R4,LSR #1
23	2.667 us	AND R7,R2,#1
24	2.667 us	CMP R7,#1
25	5.333 us	ADDEQ R4,R4,#0X80000000 ; Q SHIFTED
26	2.667 us	MOV R2,R2,ASR #1 ; A SHIFTED
27	2.667 us	ADD R6,R6,#1
28	8.083 us	B LOOP
29		ENDL
30		END

>

```

4      START
5      1 *      MOV R0,#7      ; MULTIPLIER OR B
6      1 *      MOV R1,#-1     ; MULTIPLICAND
7      1 *      MOV R2,#0      ; A
8      1 *      MOV R3,#0      ; Q_-1
9      1 *      MOV R4,R1      ; Q AND (AT LAST WILL GIVE FINI
10     1 *      MOV R6,#0      ; COUNT
11     LOOP
12     33 *      CMP R6,#32
13     33 *      BEQ ENDL
14     32 *      AND R7,R4,#1
15     32 *      ADD R7,R3,R7,LSL #1 ;Q_0Q_-1
16     32 *      CMP R7,#1
17     32 *      ADDEQ R2,R2,R0  ; A+B IF EQ
18     32 *      CMP R7,#2
19     32 *      SUBEQ R2,R2,R0  ; A-B IF EQ
20     32 *      AND R7,R4,#1
21     32 *      MOV R3,R7      ; Q_-1 SHIFTED
22     32 *      MOV R4,R4,LSR #1
23     32 *      AND R7,R2,#1
24     32 *      CMP R7,#1
25     32 *      ADDEQ R4,R4,#0X80000000 ; Q SHIFTED
26     32 *      MOV R2,R2,ASR #1      ; A SHIFTED
27     32 *      ADD R6,R6,#1
28     32 *      B LOOP
29     ENDL
30     END

```