# Course Project - Computer Architecture

180010030 180010032 180020006

## Multiply two numbers using Booth's Algorithm in ARM

1.

> There are 32 Static instructions in the code.
> There are 652 Dynamic instructions in the code.

2.

> There are 32 instructions if we don't include conditional execution.
> There are 23 instructions using conditional execution.

**Code:-**

```
        AREA BOOTHALGO,CODE
                ENTRY
        EXPORT __START
__START
        MOV R0,#-4      ; MULTIPLIER OR B
        MOV R1,#-2      ; MULTIPLICAND
        MOV R2,#0       ; A
        MOV R3,#0       ; Q_-1
        MOV R4,R1       ; Q AND (AT LAST WILL GIVE FINAL ANSWER)
        MOV R6,#0       ; COUNT
LOOP
        CMP R6,#32
        BEQ ENDL
        AND R7,R4,#1
        ADD R7,R3,R7,LSL #1     ;Q_0Q_-1
        CMP R7,#1
        ADDEQ R2,R2,R0          ; A+B IF EQ
        CMP R7,#2
```

```
            SUBEQ R2,R2,R0              ; A-B IF EQ
            AND R7,R4,#1
            MOV R3,R7           ; Q_-1 SHIFTED
            MOV R4,R4,LSR #1
            AND R7,R2,#1
            CMP R7,#1
            ADDEQ R4,R4,#0X80000000       ; Q SHIFTED
            MOV R2,R2,ASR #1          ; A SHIFTED
            ADD R6,R6,#1
            B LOOP
ENDL
            END
```

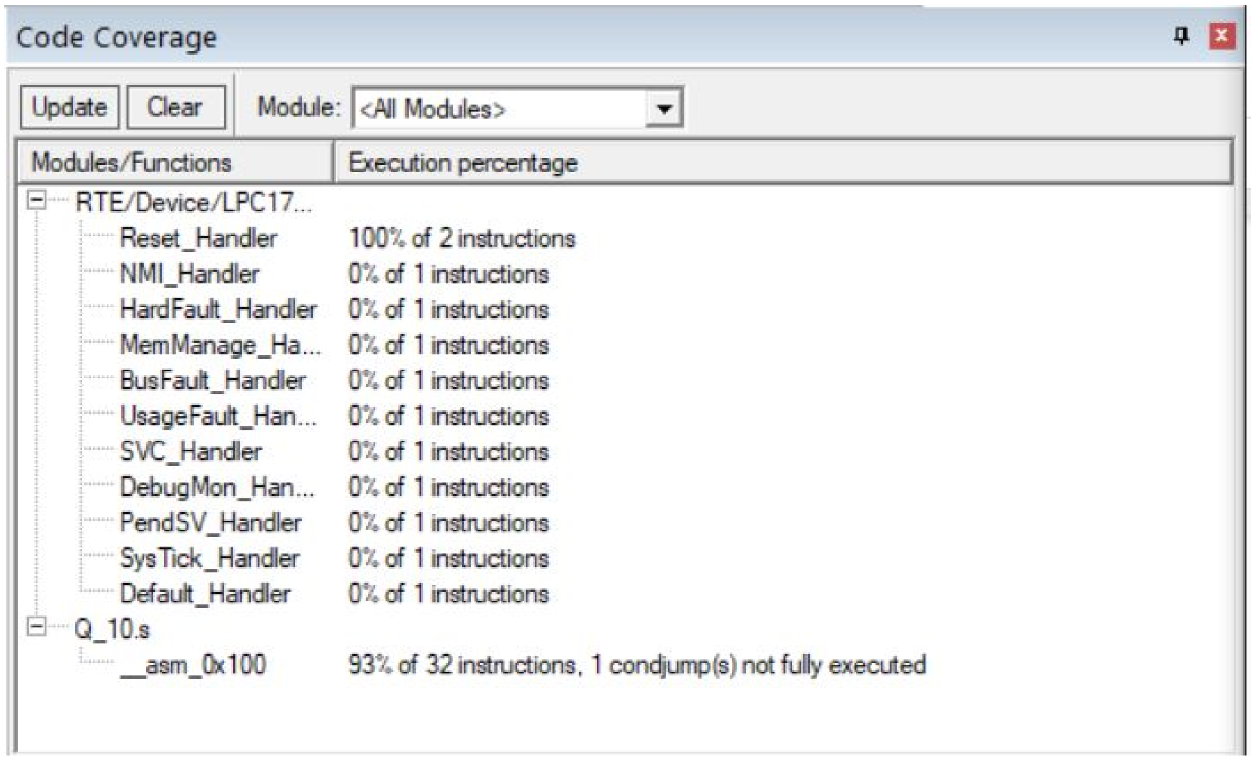
> There are 33 instructions if we write code in THUMB instructions.

Code:-

```
            AREA BOOTHALGO,CODE
                        ENTRY
            EXPORT __START
__START
            MOV R0,#-4          ; MULTIPLIER OR B
            MOV R1,#-2          ; MULTIPLICAND
            MOV R2,#0           ; A
            MOV R3,#0           ; Q_-1
            MOV R4,R1           ; Q AND (AT LAST WILL GIVE FINAL ANSWER)
            MOV R6,#0           ; COUNT
LOOP
            CMP R6,#32
            BEQ ENDL
            AND R7,R4,#1
            MOV R7,R7,LSL #1
            ADD R7,R3,R7        ;Q_0Q_-1
            CMP R7,#1
            ADD LR,PC,#2
            BEQ ADDM
            CMP R7,#2
            ADD LR,PC,#2
            BEQ SUBM
            AND R7,R4,#1
            MOV R3,R7        ; Q_-1 SHIFTED
            MOV R4,R4,LSR #1
            AND R7,R2,#1
            CMP R7,#1
            ADD LR,PC,#4
            BEQ ADDQ
            MOV R2,R2,ASR #1        ; A SHIFTED
            ADD R6,R6,#1
            B LOOP
ADDM
            ADD R2,R2,R0    ; A+B IF EQ
            MOV PC,LR
SUBM
            SUB R2,R2,R0    ; A-B IF EQ
            MOV PC,LR
ADDQ
            ADD R4,R4,#0X80000000          ; Q SHIFTED
            MOV PC,LR
ENDL
            END
```

3.

>

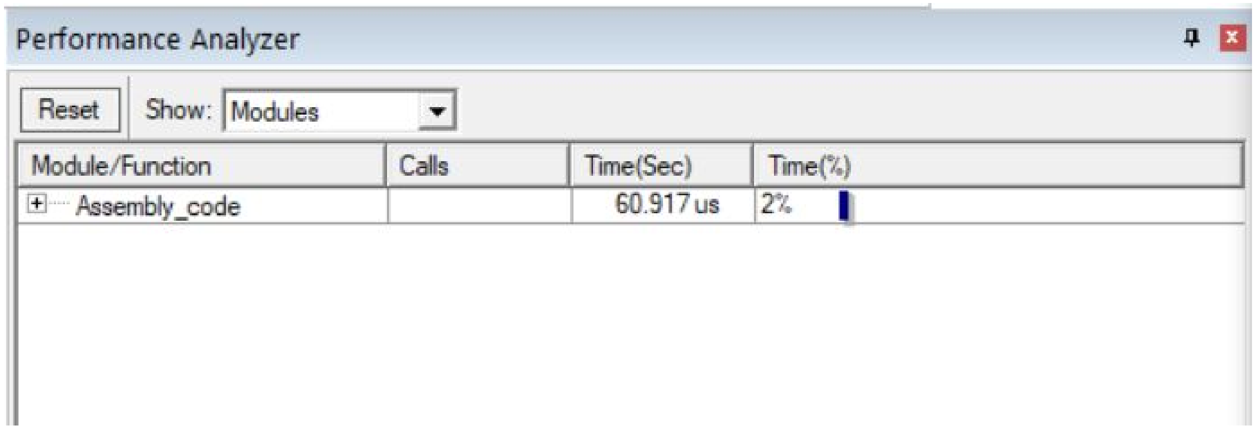


Code Coverage

Update | Clear | Module: <All Modules>

| Modules/Functions | Execution percentage |
|---|---|
| RTE/Device/LPC17... | |
| Reset_Handler | 100% of 2 instructions |
| NMI_Handler | 0% of 1 instructions |
| HardFault_Handler | 0% of 1 instructions |
| MemManage_Ha... | 0% of 1 instructions |
| BusFault_Handler | 0% of 1 instructions |
| UsageFault_Han... | 0% of 1 instructions |
| SVC_Handler | 0% of 1 instructions |
| DebugMon_Han... | 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 | 93% of 32 instructions, 1 condjump(s) not fully executed |

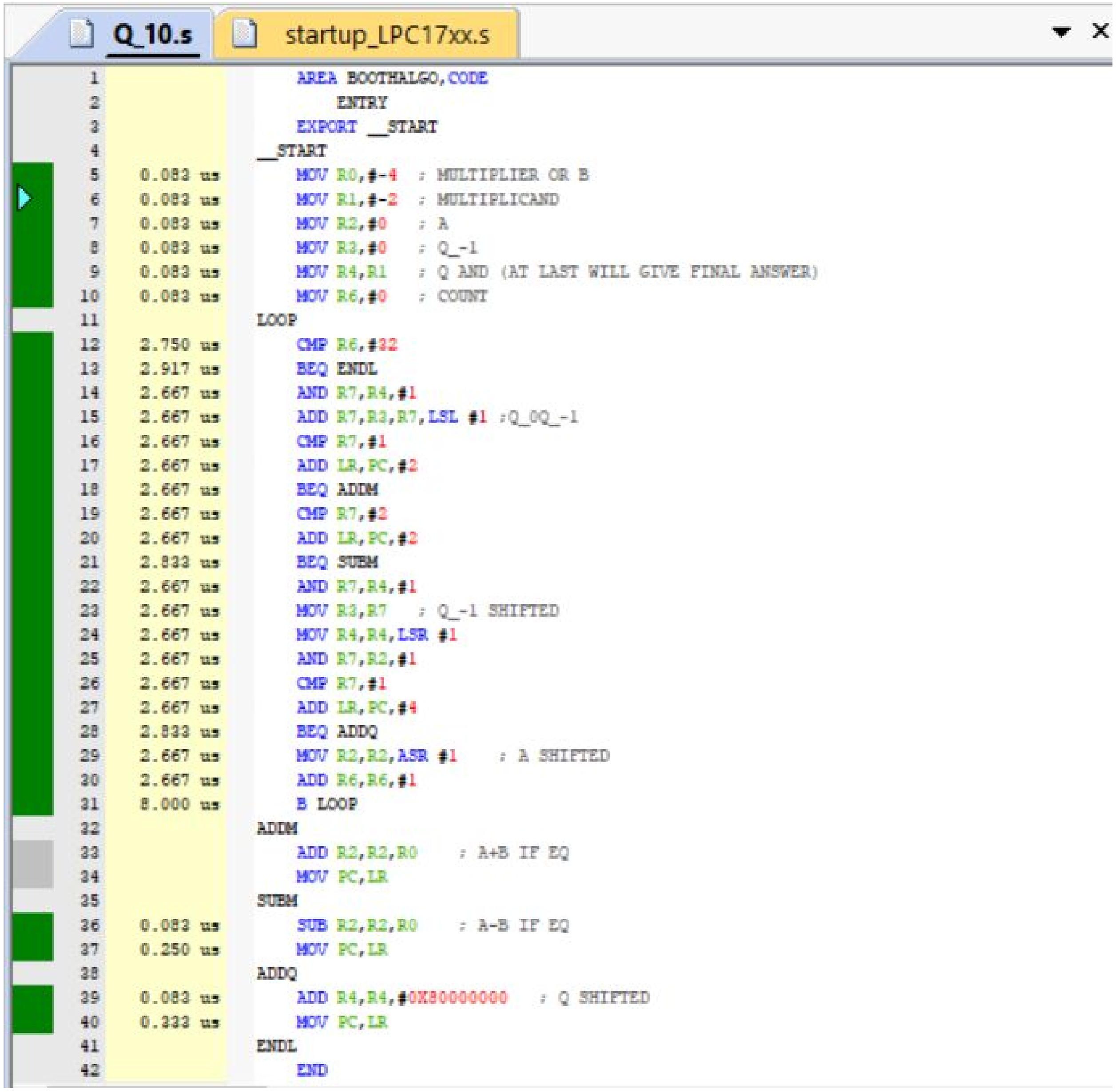
> Total code execution time:-



Performance Analyzer

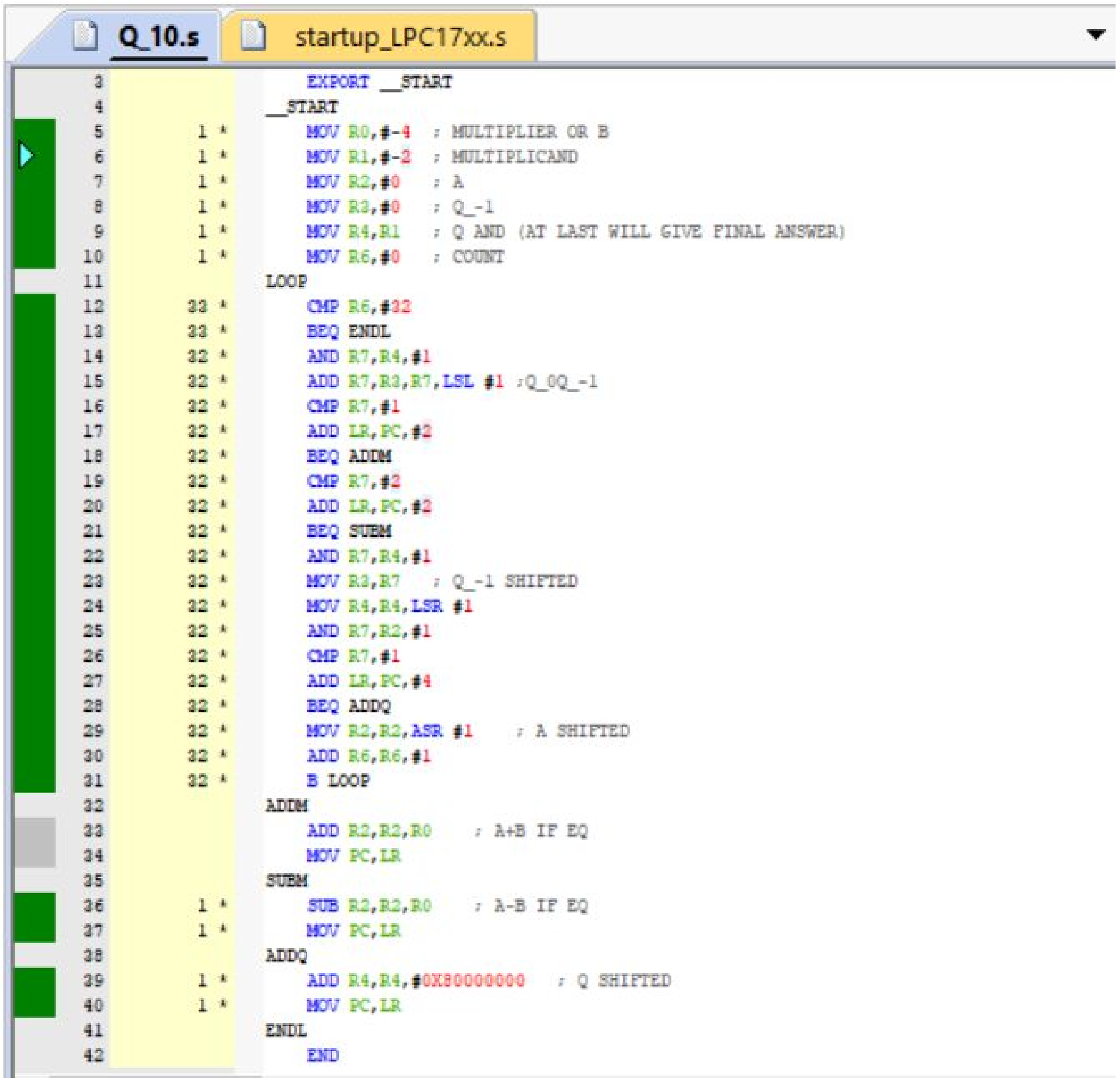
Reset | Show: Modules

| Module/Function | Calls | Time(Sec) | Time(%) |
|---|---|---|---|
| Assembly_code | | 60.917 us | 2% |



```
                  AREA BOOTHALGO,CODE
                      ENTRY
                  EXPORT __START
              __START
0.083 us          MOV R0,#-4   ; MULTIPLIER OR B
0.083 us          MOV R1,#-2   ; MULTIPLICAND
0.083 us          MOV R2,#0    ; A
0.083 us          MOV R3,#0    ; Q_-1
0.083 us          MOV R4,R1    ; Q AND (AT LAST WILL GIVE FINAL ANSWER)
0.083 us          MOV R6,#0    ; COUNT
              LOOP
2.750 us          CMP R6,#32
2.917 us          BEQ ENDL
2.667 us          AND R7,R4,#1
2.667 us          ADD R7,R3,R7,LSL #1 ;Q_0Q_-1
2.667 us          CMP R7,#1
2.667 us          ADD LR,PC,#2
2.667 us          BEQ ADDM
2.667 us          CMP R7,#2
2.667 us          ADD LR,PC,#2
2.833 us          BEQ SUBM
2.667 us          AND R7,R4,#1
2.667 us          MOV R3,R7     ; Q_-1 SHIFTED
2.667 us          MOV R4,R4,LSR #1
2.667 us          AND R7,R2,#1
2.667 us          CMP R7,#1
2.667 us          ADD LR,PC,#4
2.833 us          BEQ ADDQ
2.667 us          MOV R2,R2,ASR #1      ; A SHIFTED
2.667 us          ADD R6,R6,#1
8.000 us          B LOOP
              ADDM
                  ADD R2,R2,R0      ; A+B IF EQ
                  MOV PC,LR
              SUBM
0.083 us          SUB R2,R2,R0      ; A-B IF EQ
0.250 us          MOV PC,LR
              ADDQ
0.083 us          ADD R4,R4,#0X80000000    ; Q SHIFTED
0.333 us          MOV PC,LR
              ENDL
                  END
```

>



```
         EXPORT __START
    __START
1 *      MOV R0,#-4    ; MULTIPLIER OR B
1 *      MOV R1,#-2    ; MULTIPLICAND
1 *      MOV R2,#0     ; A
1 *      MOV R3,#0     ; Q_-1
1 *      MOV R4,R1     ; Q AND (AT LAST WILL GIVE FINAL ANSWER)
1 *      MOV R6,#0     ; COUNT
    LOOP
33 *     CMP R6,#32
33 *     BEQ ENDL
32 *     AND R7,R4,#1
32 *     ADD R7,R3,R7,LSL #1 ;Q_0Q_-1
32 *     CMP R7,#1
32 *     ADD LR,PC,#2
32 *     BEQ ADDM
32 *     CMP R7,#2
32 *     ADD LR,PC,#2
32 *     BEQ SUBM
32 *     AND R7,R4,#1
32 *     MOV R3,R7      ; Q_-1 SHIFTED
32 *     MOV R4,R4,LSR #1
32 *     AND R7,R2,#1
32 *     CMP R7,#1
32 *     ADD LR,PC,#4
32 *     BEQ ADDQ
32 *     MOV R2,R2,ASR #1      ; A SHIFTED
32 *     ADD R6,R6,#1
32 *     B LOOP
    ADDM
         ADD R2,R2,R0      ; A+B IF EQ
         MOV PC,LR
    SUBM
1 *      SUB R2,R2,R0      ; A-B IF EQ
1 *      MOV PC,LR
    ADDQ
1 *      ADD R4,R4,#0X80000000    ; Q SHIFTED
1 *      MOV PC,LR
    ENDL
         END
```

Total instructions call is equal to dynamic instructions.