



## ARM ASSEMBLY LAB-6

By

Swapneel Pimparkar (CS18M516)

CS6620a

Prof. Madhu Mutyam

## STATUS UPDATE

The ARM assembly code was written for 32 bit processor and verified using ARMSim simulator successfully.

## EXERCISE 1 – SEARCH IN UNSORTED ARRAY

Write an assembly program for searching a given integer number in an array of integer numbers. Assume that the numbers in the array are not in sorted order. The program must ask the user to enter the number of elements of the array and accept each element of the array through keyboard (for this, you need to use software interrupts). Also, the user must enter the element to be searched through keyboard. You must pass the array and the searching element as parameters to a subroutine, SEARCH. The program outputs the position of the given element, if it is present in the array, otherwise, it outputs -1.

**The logic to solve this exercise is hand-coded in 32 bit ARM Assembly and verified on ARMSim Simulator.**

The logic used is mentioned in the code file itself and all the necessary instructions are supplied with comments.

It was run for various inputs and verified.

## SCREENSHOT - TEST 1

ARMSim# - The ARM Simulator Dept. of Computer Science

File View Cache Debug Watch Help

RegistersView

General Purpose Floating Point

Hexadecimal  
Unsigned Decimal  
Signed Decimal

R0 : 00000001  
R1 : 00000004  
R2 : 00000004  
R3 : 00000000  
R4 : 00000000  
R5 : 00000002  
R6 : 00001210  
R7 : 6dfffffe  
R8 : 00000005  
R9 : 00001204  
R10 (s1) : 00000000  
R11 (fp) : 00000000  
R12 (ip) : 00000000  
R13 (sp) : 00000000  
R14 (lr) : 0000113c  
R15 (pc) : 00001138

CPSR Register  
Negative (N) : 0  
Zero (Z) : 0  
Carry (C) : 0  
Overflow (V) : 0  
IRQ Disable : 1  
FIQ Disable : 1  
Thumb (T) : 0  
CPU Mode : Undefined

0x000000db

CodeView

Lab6\_1.o

```

00001000:E59F90F8    LDR    R9, =OUTPUT
00001004:E59F80F8    LDR    R8, =ARR_SIZE
00001008:E59F70F8    LDR    R7, =ARR_ELEM_SEARCH
0000100C:E59F60F8    LDR    R6, =ARR
00001010:E3A05000    MOV    R5, #0                @ This will be used as counter.

                                @@@ Read the number of array elements (i.e. size) @@@
00001014:E59F00F4    LDR    R0, =STDOUT_HANDLE
00001018:E5900000    LDR    R0, [R0]
0000101C:E59F10F0    LDR    R1, =FIRST_INPUT_MSG    @ Read address of FIRST_INPUT_MSG address into R1
00001020:EF000069    SWI    0x69                @ Output at console. (Ref. https://www.lri.fr/~de/ARM-Tutor

00001024:E59F00EC    LDR    R0, =STDIN_HANDLE
00001028:E5900000    LDR    R0, [R0]
0000102C:EF00006C    SWI    0x6c

                                @@ Check if the input number is greater than 512.
00001030:E3500C02    CMP    R0, #0x200
00001034:CAFFFFFE    BGT    _END

                                @@ Check if 0 is input or any negative.
00001038:E3500000    CMP    R0, #0
0000103C:DAFFFFFE    BLE    _END

00001040:E5880000    STR    R0, [R8]
00001044:E5988000    LDR    R8, [R8]
00001048:E1A05008    MOV    R5, R8

                                @@@ Read the array elements @@@
0000104C:E59F00BC    LDR    R0, =STDOUT_HANDLE
00001050:E5900000    LDR    R0, [R0]
00001054:E59F10C0    LDR    R1, =SECOND_INPUT_MSG    @ Read address of SECOND_INPUT_MSG address into R1
00001058:EF000069    SWI    0x69                @ Output at console.

```

OutputView

Console stdin/stdout/stderr stdin/stdout/stderr

```

Please Enter the integer number of array elements (Size < 512): 5
Please Enter the array elements separated by comma : 1,9,5,3,7
Please Enter the array element to search (only one): 3
Input element is found at position : 4

```

## SCREENSHOT - TEST 2

The screenshot displays the ARMSim simulator interface. The top menu bar includes File, View, Cache, Debug, Watch, and Help. The main window is divided into three panes:

- RegistersView:** Shows the state of 16 general-purpose registers (R0-R15) in hexadecimal, unsigned decimal, and signed decimal. The CPSR register is also shown with fields like Negative (N), Zero (Z), Carry (C), Overflow (V), and IRQ/FIQ Disable. The CPU mode is set to Undefined.
- CodeView:** Displays the assembly code for 'Lab6\_1.o'. The code includes instructions for loading array size, searching for an element, and handling input/output. Comments explain the logic, such as reading the number of array elements and checking for input validity.
- OutputView:** Shows the console output, which includes prompts for the user to enter the number of array elements, the array elements, and the element to search. The final output indicates that the input element is found at position -1.

## EXERCISE 2 – SEARCH IN SORTED ARRAY

In the above problem, as the elements of the array are not in sorted order, we have to search all the elements to find whether a given element is present or not. Now, assume that the elements of the array are in sorted order. Write an assembly language program that can efficiently search a given element in the sorted array of elements. (Note that here we define the efficiency in terms of the number of searches.)

**The logic to solve this exercise is hand-coded in 32 bit ARM Assembly and verified on ARMSim Simulator.**

The logic used is mentioned in the code file itself and all the necessary instructions are supplied with comments.

It is separately written in two programs, main for ordering namely, Ascending and Descending Ordering.

## SCREENSHOT – TEST 1: ARRAY IN ASCENDING ORDER

The screenshot displays the ARMSim# interface. The **RegistersView** on the left shows the state of various registers. The **CodeView** in the center shows the assembly code for **Lab6\_2\_Ascending.o**. The **OutputView** at the bottom shows the program's execution output.

**RegistersView:**

Register	Value
R0	:00000001
R1	:00000000
R2	:000012e8
R3	:00000007
R4	:00000000
R5	:00000004
R6	:000012d8
R7	:00000007
R8	:00000005
R9	:00000004
R10 (s1)	:00000000
R11 (fp)	:00000000
R12 (ip)	:00000000
R13 (sp)	:00000000
R14 (lr)	:000011a8
R15 (pc)	:000011a4

**CPSR Register:**

Field	Value
Negative (N)	:0
Zero (Z)	:1
Carry (C)	:0
Overflow (V)	:0
IRQ Disable	:1
FIQ Disable	:1
Thumb (T)	:0
CPU Mode	:Undefin

**CodeView:**

```
000010E0:EF00006B      SWI      0x6b          @ Output at console.
000010E4:EAF000FE      B        _END
                                SEARCH:
000010E8:E92D41C0      STMFD    SP!, {R6, R7, R8, LR} @ R8 is array size.
                                @ R0 is Min
                                @ R1 is Mid
                                @ R2 is Max
000010EC:E3A03004      MOV      R3, #4
000010F0:E2482001      SUB      R2, R8, #1
000010F4:E0226392      MLA      R2, R2, R3, R6
000010F8:E5922000      LDR      R2, [R2]      @ R2 now contains last element.
000010FC:E5960000      LDR      R0, [R6]      @ R0 contains first element.
00001100:E5999000      LDR      R9, [R9]
                                @ Check if the element is out of bounds.
00001104:E1570002      CMP      R7, R2
00001108:CA00001F      BGT      RET_SEARCH_NOT_FOUND
0000110C:E1570000      CMP      R7, R0
00001110:BA00001D      BLT      RET_SEARCH_NOT_FOUND
                                @ Check if first element is the one.
00001114:E2899001      ADD      R9, #1        @ Increment search performance counter.
00001118:E1500007      CMP      R0, R7
0000111C:0A00001C      BEQ      RET_SEARCH    @ We found that element to be searched is first one.
                                @ Check if last element is the one.
00001120:E2899001      ADD      R9, #1        @ Increment search performance counter.
00001124:E1520007      CMP      R2, R7
00001128:0A000019      BEQ      RET_SEARCH    @ We found that element to be searched is last one.
                                @ If we are here, it means that we need to search in between first and last.
```

**OutputView:**

```
Console  stdin/stdout/stderr  stdin/stdout/stderr
Please Enter the integer number of array elements (Size < 512): 5
Please Enter the array elements separated by comma : 2,3,5,7,9
Please Enter the array element to search (only one): 7

Input element is found at position : 4
Efficiency (No. of Searches) : 4
```

## SCREENSHOT – TEST 2: ASCENDING ORDER – EVEN NO. OF ELEMENTS

ARMSim# - The ARM Simulator Dept. of Computer Science

File View Cache Debug Watch Help

RegistersView

General Purpose Floating Point

Hexadecimal  
Unsigned Decimal  
Signed Decimal

R0 : 00000001  
R1 : 00000000  
R2 : 000012e4  
R3 : 00000062  
R4 : 00000000  
R5 : 00000003  
R6 : 000012d8  
R7 : 00000062  
R8 : 00000004  
R9 : 00000003  
R10 (s1) : 00000000  
R11 (fp) : 00000000  
R12 (ip) : 00000000  
R13 (sp) : 00000000  
R14 (lr) : 000011a8  
R15 (pc) : 000011a4

CPSR Register  
Negative (N) : 0  
Zero (Z) : 1  
Carry (C) : 0  
Overflow (V) : 0  
IRQ Disable : 1  
FIQ Disable : 1  
Thumb (T) : 0  
CPU Mode : Undefined

0x40000db

CodeView

Lab6\_2\_Ascending.o

```

00001000:E59F9198    LDR    R9, =OUTPUT
00001004:E59F8198    LDR    R8, =ARR_SIZE
00001008:E59F7198    LDR    R7, =ARR_ELEM_SEARCH
0000100C:E59F6198    LDR    R6, =ARR

    @@@ Read the number of array elements (i.e. size) @@@

00001010:E59F0198    LDR    R0, =STDOUT_HANDLE
00001014:E5900000    LDR    R0, [R0]
00001018:E59F1194    LDR    R1, =FIRST_INPUT_MSG    @ Read address of FIRST_INPUT_MSG address into R1
0000101C:EF000069    SWI    0x69                    @ Output at console. (Ref. https://www.lri.fr/~de/ARM-Tutor

00001020:E59F0190    LDR    R0, =STDIN_HANDLE
00001024:E5900000    LDR    R0, [R0]
00001028:EF00006C    SWI    0x6c

    @@ Check if the input number is greater than 512.
0000102C:E3500C02    CMP    R0, #0x200
00001030:CAFFFFFE    BGT    _END

    @@ Check if 0 is input or any negative.
00001034:E3500000    CMP    R0, #0
00001038:DFFFFFFFE    BLE    _END

0000103C:E5880000    STR    R0, [R8]
00001040:E5988000    LDR    R8, [R8]
00001044:E1A05008    MOV    R5, R8

    @@@ Read the array elements @@@
00001048:E59F0160    LDR    R0, =STDOUT_HANDLE
0000104C:E5900000    LDR    R0, [R0]
00001050:E59F1164    LDR    R1, =SECOND_INPUT_MSG    @ Read address of SECOND_INPUT_MSG address into R1
00001054:EF000069    SWI    0x69                    @ Output at console.

```

OutputView

Console stdin/stdout/stderr stdout/stdout/stderr

```

Please Enter the integer number of array elements (Size < 512): 4
Please Enter the array elements separated by comma : 65,98,156,958
Please Enter the array element to search (only one): 98

Input element is found at position : 3
Efficiency (No. of Searches) : 3

```

## SCREENSHOT – TEST 3: ASCENDING ORDER – BAD SEARCH ELEMENT

ARMSim# - The ARM Simulator Dept. of Computer Science

File View Cache Debug Watch Help

RegistersView

General Purpose Floating Point

Hexadecimal  
Unsigned Decimal  
Signed Decimal

R0 : 00000001  
R1 : 00000000  
R2 : 000012e8  
R3 : 0000002d  
R4 : 00000000  
R5 : ffffffff  
R6 : 000012d8  
R7 : 00000019  
R8 : 00000008  
R9 : 00000005  
R10 (s1) : 00000000  
R11 (fp) : 00000000  
R12 (ip) : 00000000  
R13 (sp) : 00000000  
R14 (lr) : 000011a8  
R15 (pc) : 000011a4

-----  
CPSR Register  
Negative (N) : 0  
Zero (Z) : 1  
Carry (C) : 0  
Overflow (V) : 0  
IRQ Disable : 1  
FIQ Disable : 1  
Thumb (T) : 0  
CPU Mode : Undefined  
0x400000db

CodeView

Lab6\_2\_Ascending.o

```
00001000:E59F9198    LDR    R9, =OUTPUT
00001004:E59F8198    LDR    R8, =ARR_SIZE
00001008:E59F7198    LDR    R7, =ARR_ELEM_SEARCH
0000100C:E59F6198    LDR    R6, =ARR

    @@@ Read the number of array elements (i.e. size) @@@

00001010:E59F0198    LDR    R0, =STDOUT_HANDLE
00001014:E5900000    LDR    R0, [R0]
00001018:E59F1194    LDR    R1, =FIRST_INPUT_MSG    @ Read address of FIRST_INPUT_MSG address into R1
0000101C:EF000069    SWI    0x69                    @ Output at console. (Ref. https://www.lri.fr/~de/ARM-Tutor

00001020:E59F0190    LDR    R0, =STDIN_HANDLE
00001024:E5900000    LDR    R0, [R0]
00001028:EF00006C    SWI    0x6c

    @@ Check if the input number is greater than 512.
0000102C:E3500C02    CMP    R0, #0x200
00001030:CAFFFFFE    BGT    _END

    @@ Check if 0 is input or any negative.
00001034:E3500000    CMP    R0, #0
00001038:DAFFFFFE    BLE    _END

0000103C:E5880000    STR    R0, [R8]
00001040:E5988000    LDR    R8, [R8]
00001044:E1A05008    MOV    R5, R8

    @@@ Read the array elements @@@
00001048:E59F0160    LDR    R0, =STDOUT_HANDLE
0000104C:E5900000    LDR    R0, [R0]
00001050:E59F1164    LDR    R1, =SECOND_INPUT_MSG    @ Read address of SECOND_INPUT_MSG address into R1
00001054:EF000069    SWI    0x69                    @ Output at console.
```

OutputView

Console | stdin/stdout/stderr | stdin/stdout/stderr

```
Please Enter the integer number of array elements (Size < 512): 8
Please Enter the array elements separated by comma : 1,5,8,9,45,65,85,98
Please Enter the array element to search (only one): 25

Input element is found at position : -1
Efficiency (No. of Searches) : 5|
```

## SCREENSHOT – TEST 3: DESCENDING ORDER – BAD SEARCH ELEMENT

ARMSim# - The ARM Simulator Dept. of Computer Science

File View Cache Debug Watch Help

RegistersView

General Purpose Floating Point

Hexadecimal  
Unsigned Decimal  
Signed Decimal

R0 : 00000001  
R1 : 00000000  
R2 : ffffffff  
R3 : 00000004  
R4 : 00000000  
R5 : ffffffff  
R6 : 000012d8  
R7 : ffffffff  
R8 : 00000005  
R9 : 00000000  
R10 (s1) : 00000000  
R11 (fp) : 00000000  
R12 (ip) : 00000000  
R13 (sp) : 00011800  
R14 (lr) : 000010a4  
R15 (pc) : 000011d0

CPSR Register  
Negative (N) : 1  
Zero (Z) : 0  
Carry (C) : 0  
Overflow (V) : 0  
IRQ Disable : 1  
FIQ Disable : 1  
Thumb (T) : 0  
CPU Mode : System

0x800000df

CodeView

Lab6\_2\_Descending.o

```

00001000:E59F9198      LDR    R9, =OUTPUT
00001004:E59F8198      LDR    R8, =ARR_SIZE
00001008:E59F7198      LDR    R7, =ARR_ELEM_SEARCH
0000100C:E59F6198      LDR    R6, =ARR

    @@@ Read the number of array elements (i.e. size) @@@

00001010:E59F0198      LDR    R0, =STDOUT_HANDLE
00001014:E5900000      LDR    R0, [R0]
00001018:E59F1194      LDR    R1, =FIRST_INPUT_MSG    @ Read address of FIRST_INPUT_MSG address into R1
0000101C:EF000069      SWI    0x69                    @ Output at console. (Ref. https://www.lri.fr/~de/ARM-Tutor

00001020:E59F0190      LDR    R0, =STDIN_HANDLE
00001024:E5900000      LDR    R0, [R0]
00001028:EF00006C      SWI    0x6c

    @ Check if the input number is greater than 512.
0000102C:E3500C02      CMP    R0, #0x200
00001030:CAFFFFFE      BGT    _END

    @ Check if 0 is input or any negative.
00001034:E3500000      CMP    R0, #0
00001038:DAPFFFFFE      BLE    _END

0000103C:E5880000      STR    R0, [R8]
00001040:E5988000      LDR    R8, [R8]
00001044:E1A05008      MOV    R5, R8

    @@@ Read the array elements @@@
00001048:E59F0160      LDR    R0, =STDOUT_HANDLE
0000104C:E5900000      LDR    R0, [R0]
00001050:E59F1164      LDR    R1, =SECOND_INPUT_MSG    @ Read address of SECOND_INPUT_MSG address into R1
00001054:EF000069      SWI    0x69                    @ Output at console.

```

OutputView

Console stdin/stdout/stderr stdin/stdout/stderr

```

Please Enter the integer number of array elements (Size < 512): 5
Please Enter the array elements separated by comma : 5,4,1,0,-5
Please Enter the array element to search (only one): -10

Input element is found at position : -1
Efficiency (No. of Searches) : 0

```



## SCREENSHOT – TEST 3: DESCENDING ORDER – VALID SEARCH ELEMENT

ARMSim# - The ARM Simulator Dept. of Computer Science

File View Cache Debug Watch Help

RegistersView

General Purpose Floating Point

Hexadecimal  
Unsigned Decimal  
Signed Decimal

R0 : 00000001  
R1 : 00000000  
R2 : 000012f0  
R3 : ffffffff  
R4 : 00000000  
R5 : 00000006  
R6 : 000012d8  
R7 : ffffffff  
R8 : 00000007  
R9 : 00000003  
R10 (s1): 00000000  
R11 (fp): 00000000  
R12 (ip): 00000000  
R13 (sp): 00000000  
R14 (lr): 000011a8  
R15 (pc): 000011a4

-----  
CPSR Register  
Negative (N) : 0  
Zero (Z) : 1  
Carry (C) : 0  
Overflow (V) : 0  
IRQ Disable: 1  
FIQ Disable: 1  
Thumb (T) : 0  
CPU Mode : Undefin

-----  
0x400000db

CodeView

Lab6\_2\_Descending.o

```

00001000:E59F9198    LDR    R9, =OUTPUT
00001004:E59F8198    LDR    R8, =ARR_SIZE
00001008:E59F7198    LDR    R7, =ARR_ELEM_SEARCH
0000100C:E59F6198    LDR    R6, =ARR

    @ Read the number of array elements (i.e. size) @
00001010:E59F0198    LDR    R0, =STDOUT_HANDLE
00001014:E5900000    LDR    R0, [R0]
00001018:E59F1194    LDR    R1, =FIRST_INPUT_MSG    @ Read address of FIRST_INPUT_MSG address into R1
0000101C:EF000069    SWI    0x69                    @ Output at console. (Ref. https://www.lri.fr/~de/ARM-Tutor

00001020:E59F0190    LDR    R0, =STDIN_HANDLE
00001024:E5900000    LDR    R0, [R0]
00001028:EF00006C    SWI    0x6c

    @ Check if the input number is greater than 512.
0000102C:E3500C02    CMP    R0, #0x200
00001030:CAF0FFFE    BGT    _END

    @ Check if 0 is input or any negative.
00001034:E3500000    CMP    R0, #0
00001038:DAFFFFFE    BLE    _END

0000103C:E5880000    STR    R0, [R8]
00001040:E5988000    LDR    R8, [R8]
00001044:E1A05008    MOV    R5, R8

    @ Read the array elements @
00001048:E59F0160    LDR    R0, =STDOUT_HANDLE
0000104C:E5900000    LDR    R0, [R0]
00001050:E59F1164    LDR    R1, =SECOND_INPUT_MSG    @ Read address of SECOND_INPUT_MSG address into R1
00001054:EF000069    SWI    0x69                    @ Output at console.

```

OutputView

Console stdin/stdout/stderr stdin/stdout/stderr

```

Please Enter the integer number of array elements (Size < 512): 7
Please Enter the array elements separated by comma : 9,5,0,-1,-50,-58,-100
Please Enter the array element to search (only one): -1

Input element is found at position : 6
Efficiency (No. of Searches) : 3

```

### EXERCISE 3 – COMPUTE FIBONACCI NUMBER (RECURSIVELY)

Fibonacci number sequence is defined as 1, 1, 2, 3, 5, 8, 13, 21, ... A number in the Fibonacci sequence is the sum of the immediate two previous numbers, i.e.  $F_n = F_{n-1} + F_{n-2}$ ,  $n > 2$ . Note that  $F_1 = F_2 = 1$ . Write an assembly language program that accepts an integer number, N, through keyboard and computes the  $N^{\text{th}}$  Fibonacci number in recursive way.

**The logic to solve this exercise is hand-coded in 32 bit ARM Assembly and verified on ARMSim Simulator.**

The logic used is mentioned in the code file itself and all the necessary instructions are supplied with comments.

### SCREENSHOT – TEST 1 – WRONG INPUT

ARMSim# - The ARM Simulator Dept. of Computer Science

File View Cache Debug Watch Help

RegistersView CodeView

General Purpose Floating Point

Hexadecimal  
Unsigned Decimal  
Signed Decimal

R0 : 00000001  
R1 : 00001166  
R2 : 00000000  
R3 : 00000000  
R4 : 00000000  
R5 : 00000000  
R6 : 00000000  
R7 : 00000000  
R8 : 00001190  
R9 : 0000118c  
R10 (s1): 00000000  
R11 (fp): 00000000  
R12 (ip): 00000000  
R13 (sp): 00011400  
R14 (lr): 00000000  
R15 (pc): 00001108

CPSR Register  
Negative (N): 1  
Zero (Z): 0  
Carry (C): 0  
Overflow (V): 0  
IRQ Disable: 1  
FIQ Disable: 1  
Thumb (T): 0  
CPU Mode : System

0x800000df

```
00001094:1A000001    BNE GO_RECURSIVE    @ If not equal to 1 then call fibonacci in recursive way.
00001098:E3A03001    MOV    R3, #1        @ Output is 1.
0000109C:EA00000A    B     GO_UNWIND      @ Unwind the stack (from recursion) if any.

GO_RECURSIVE:
000010A0:E5183010    LDR    R3, [R8, #-16]
000010A4:E2433001    SUB    R3, R3, #1    @ This is essentially (n-1)
000010A8:E1A00003    MOV    R0, R3
000010AC:EBFFFFED    BL     FIBONACCI     @ Call fibonacci with (n-1)
000010B0:E1A04000    MOV    R4, R0
000010B4:E1A04003    MOV    R4, R3
000010B8:E5183010    LDR    R3, [R8, #-16]
000010BC:E2433002    SUB    R3, R3, #2    @ This is essentially (n-2)
000010C0:E1A00003    MOV    R0, R3
000010C4:EBFFFFE7    BL     FIBONACCI     @ Call fibonacci with (n-2)
000010C8:E0843003    @MOV    R3, R0
                        ADD    R3, R4, R3    @ Add numbers.

GO_UNWIND:
000010CC:E248D008    SUB    SP, R8, #8
000010D0:E8BD4110    LDMFD  SP!, {R4, R8, LR}
000010D4:E12FFF1E    BX     LR

_ERR:
000010D8:E59F0010    LDR    R0, =STDOUT_HANDLE
000010DC:E5900000    LDR    R0, [R0]
000010E0:E59F1018    LDR    R1, =ERR_MSG    @ Read address of ERR MSG address into R1
000010E4:EF000069    SWI    0x69            @ Output at console.

_END:
000010E8:00000088    .end
0000008C
00000084
00000000
00000080
```

OutputView

Console stdin/stdout/stderr stdout/stdout/stderr

Please Enter the integer to find fibonacci (>2 and <1024): -5

Input Error! Terminating!

## SCREENSHOT – TEST 2: WRONG INPUT

ARMSim# - The ARM Simulator Dept. of Computer Science

File View Cache Debug Watch Help

RegistersView

General Purpose Floating Point

Hexadecimal  
Unsigned Decimal  
Signed Decimal

R0 : 00000001  
R1 : 00001166  
R2 : 00000000  
R3 : 00000000  
R4 : 00000000  
R5 : 00000000  
R6 : 00000000  
R7 : 6e000000  
R8 : 00001190  
R9 : 0000118c  
R10 (s1) : 00000000  
R11 (fp) : 00000000  
R12 (ip) : 00000000  
R13 (sp) : 00000000  
R14 (lr) : 00001118  
R15 (pc) : 00001114

-----  
CPSR Register  
Negative (N) : 0  
Zero (Z) : 0  
Carry (C) : 0  
Overflow (V) : 0  
IRQ Disable : 1  
FIQ Disable : 1  
Thumb (T) : 0  
CPU Mode : Undefin

-----  
0x000000db

CodeView

Lab6\_3.o

```

00001094:1A000001    BNE GO_RECURSIVE    @ If not equal to 1 then call fibonacci in recursive way.
00001098:E3A03001    MOV    R3, #1        @ Output is 1.
0000109C:EA00000A    B     GO_UNWIND      @ Unwind the stack (from recursion) if any.

GO_RECURSIVE:
000010A0:E5183010    LDR    R3, [R8, #-16]
000010A4:E2433001    SUB    R3, R3, #1    @ This is essentially (n-1)
000010A8:E1A00003    MOV    R0, R3
000010AC:EBFFFFFFED    BL     FIBONACCI     @ Call fibonacci with (n-1)
000010B0:E1A04000    MOV    R4, R0
000010B4:E1A04003    MOV    R4, R3
000010B8:E5183010    LDR    R3, [R8, #-16]
000010BC:E2433002    SUB    R3, R3, #2    @ This is essentially (n-2)
000010C0:E1A00003    MOV    R0, R3
000010C4:EBFFFFFFE7    BL     FIBONACCI     @ Call fibonacci with (n-2)
@MOV    R3, R0
000010C8:E0843003    ADD    R3, R4, R3    @ Add numbers.

GO_UNWIND:
000010CC:E248D008    SUB    SP, R8, #8
000010D0:E8BD4110    LDMFD  SP!, {R4, R8, LR}
000010D4:E12FFF1E    BX     LR

_ERR:
000010D8:E59F0010    LDR    R0, =STDOUT_HANDLE
000010DC:E5900000    LDR    R0, [R0]
000010E0:E59F1018    LDR    R1, =ERR_MSG    @ Read address of ERR_MSG address into R1
000010E4:EF000069    SWI    0x69            @ Output at console.

_END:
000010E8:00000088    .end
:0000008C
:00000084
:00000000
:00000080

```

OutputView

Console stdin/stdout/stderr stdout/stdout/stderr

Please Enter the integer to find fibonacci (>2 and <1024): 1055

Input Error! Terminating!

## SCREENSHOT – TEST 3: GOOD/VALID INPUT

ARMSim# - The ARM Simulator Dept. of Computer Science

File View Cache Debug Watch Help

RegistersView

General Purpose Floating Point

Hexadecimal  
Unsigned Decimal  
Signed Decimal

R0 : 00000000  
R1 : 00000000  
R2 : 00000000  
R3 : 00000015  
R4 : 00000000  
R5 : 00000000  
R6 : 00000000  
R7 : 6e000000  
R8 : 00000008  
R9 : 0000118c  
R10 (s1) : 00000000  
R11 (fp) : 00000000  
R12 (ip) : 00000000  
R13 (sp) : 00000000  
R14 (lr) : 00001118  
R15 (pc) : 00001114

CPSR Register  
Negative (N) : 0  
Zero (Z) : 1  
Carry (C) : 0  
Overflow (V) : 0  
IRQ Disable : 1  
FIQ Disable : 1  
Thumb (T) : 0  
CPU Mode : Undefined

0x400000db

CodeView

Lab6\_3.o

```

00001000:E59F90E0      LDR    R9, =OUTPUT_F_N
00001004:E59F80E0      LDR    R8, =INPUT_N

                                @@@ Read the number @@@

00001008:E59F00E0      LDR    R0, =STDOUT_HANDLE
0000100C:E5900000      LDR    R0, [R0]
00001010:E59F10DC      LDR    R1, =INPUT_MSG      @ Read address of INPUT_MSG address into R1
00001014:EF000069      SWI    0x69                @ Output at console. (Ref. https://www.lri.fr/~de/ARM-Tutor

00001018:E59F00D8      LDR    R0, =STDIN_HANDLE
0000101C:E5900000      LDR    R0, [R0]
00001020:EF00006C      SWI    0x6c

                                @@ Check if the input number is greater than 1024.
00001024:E3500B01      CMP    R0, #0x400
00001028:CA00002A      BGT    _ERR

                                @@ Check if 0 or 1 is input or any negative.
0000102C:E3500001      CMP    R0, #1
00001030:DA000028      BLE    _ERR

00001034:E5880000      STR    R0, [R8]            @ Store the input number.
00001038:E5988000      LDR    R8, [R8]

0000103C:EB000009      BL     FIBONACCI

00001040:E59F00A8      LDR    R0, =STDOUT_HANDLE
00001044:E5900000      LDR    R0, [R0]
00001048:E59F10AC      LDR    R1, =OUTPUT_MSG      @ Read address of OUTPUT_MSG address into R1
0000104C:EF000069      SWI    0x69                @ Output at console.
00001050:E59F0098      LDR    R0, =STDOUT_HANDLE
00001054:E5900000      LDR    R0, [R0]
00001058:E1A01003      MOV    R1, R3              @ FIBONACCI number.
0000105C:E5893000      STR    R3, [R9]

```

OutputView

Console stdin/stdout/stderr stdin/stdout/stderr

Please Enter the integer to find fibonacci (>2 and <1024): 8  
Fibonacci Number for input integer : 21