

# CSE 3113 - Microprocessor and Assembly Lab

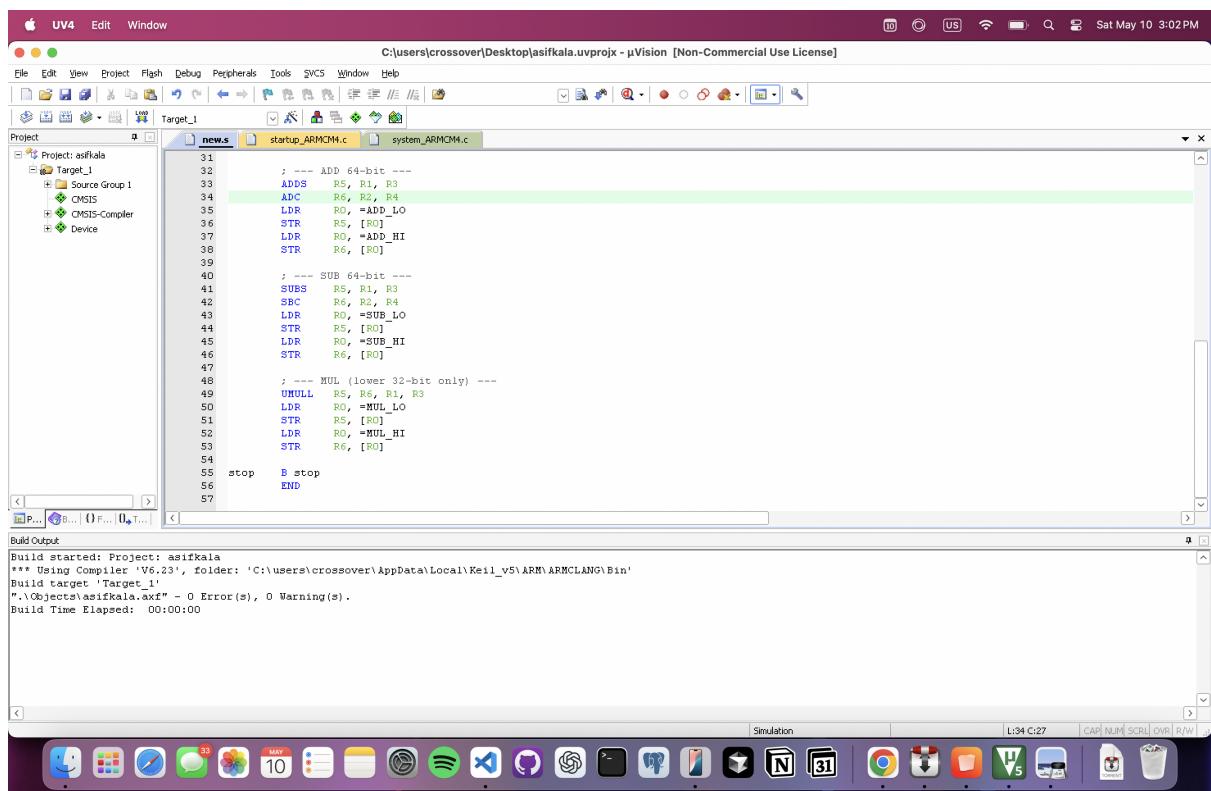
## Lab Report

Tazkia Malik  
Class Roll: 07

May 10, 2025

## 1 Perform addition, subtraction, and multiplication on 64-bit numbers

### Build Status



The screenshot shows the Keil µVision IDE interface. The main window displays assembly code for a 64-bit arithmetic project. The code includes instructions for addition, subtraction, and multiplication. The assembly code is as follows:

```
31      ; --- ADD 64-bit ---
32      ADDS  R5, R1, R3
33      ADC   R6, R2, R4
34      LDR   R0, =ADD_LO
35      STR   R5, [R0]
36      LDR   R0, =ADD_HI
37      STR   R6, [R0]
38
39      ; --- SUB 64-bit ---
40      SUBS  R5, R1, R3
41      SBC   R6, R2, R4
42      LDR   R0, =SUB_LO
43      STR   R5, [R0]
44      LDR   R0, =SUB_HI
45      STR   R6, [R0]
46
47      ; --- MUL (lower 32-bit only) ---
48      UMULL R5, R6, R1, R3
49      LDR   R0, =MUL_LO
50      STR   R5, [R0]
51      LDR   R0, =MUL_HI
52      STR   R6, [R0]
53
54      stop  B stop
55      END
56
57
```

The build output window at the bottom shows the following message:

```
Build started: Project: asifkala
*** Using Compiler: 'V6.23', folder: 'C:\users\crossover\AppData\Local\Keil_v5\ARM\ARMCLANG\Bin'
Build target: 'Target_1'
".\Objects\asifkala.axf" - 0 Error(s), 0 Warning(s).
Build Time Elapsed: 00:00:00
```

Figure 1: State of the system after the project is built.

## Register Status After Execution

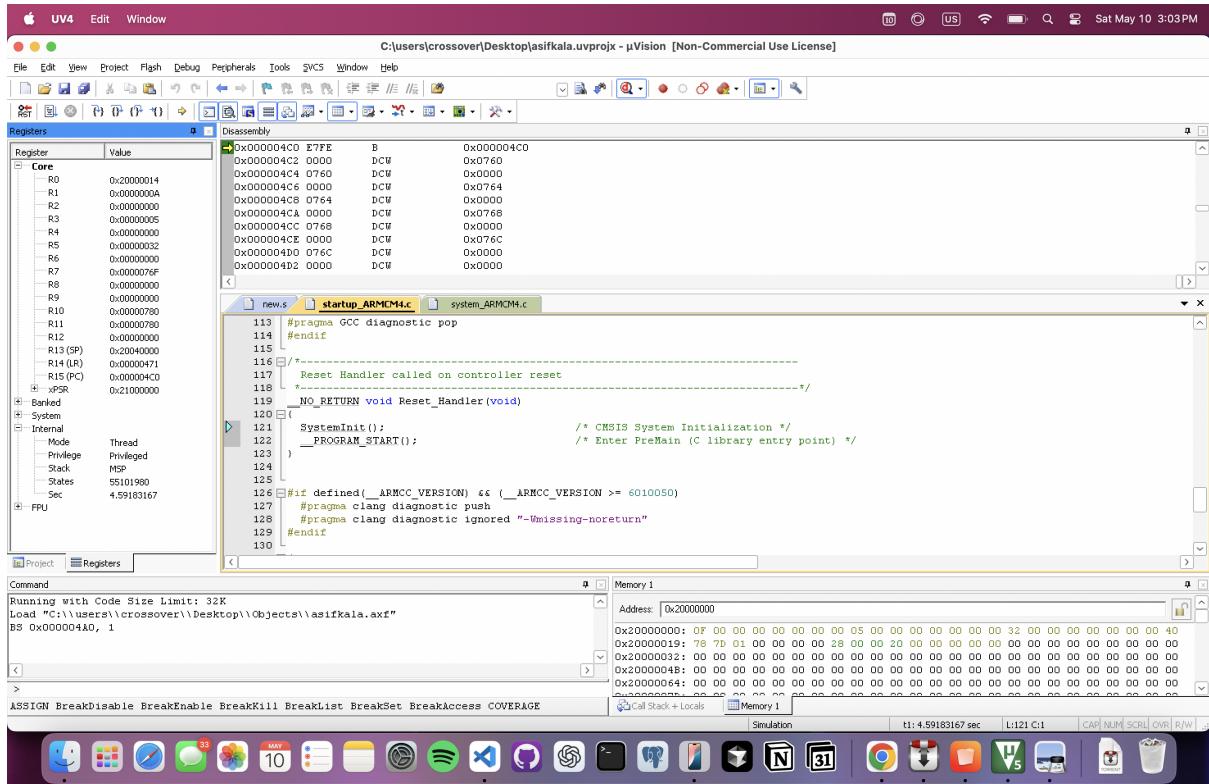


Figure 2: State of the system after the code has been executed.

## 2 Perform division on 8-bit, 16-bit, and 32-bit numbers

### Build Status

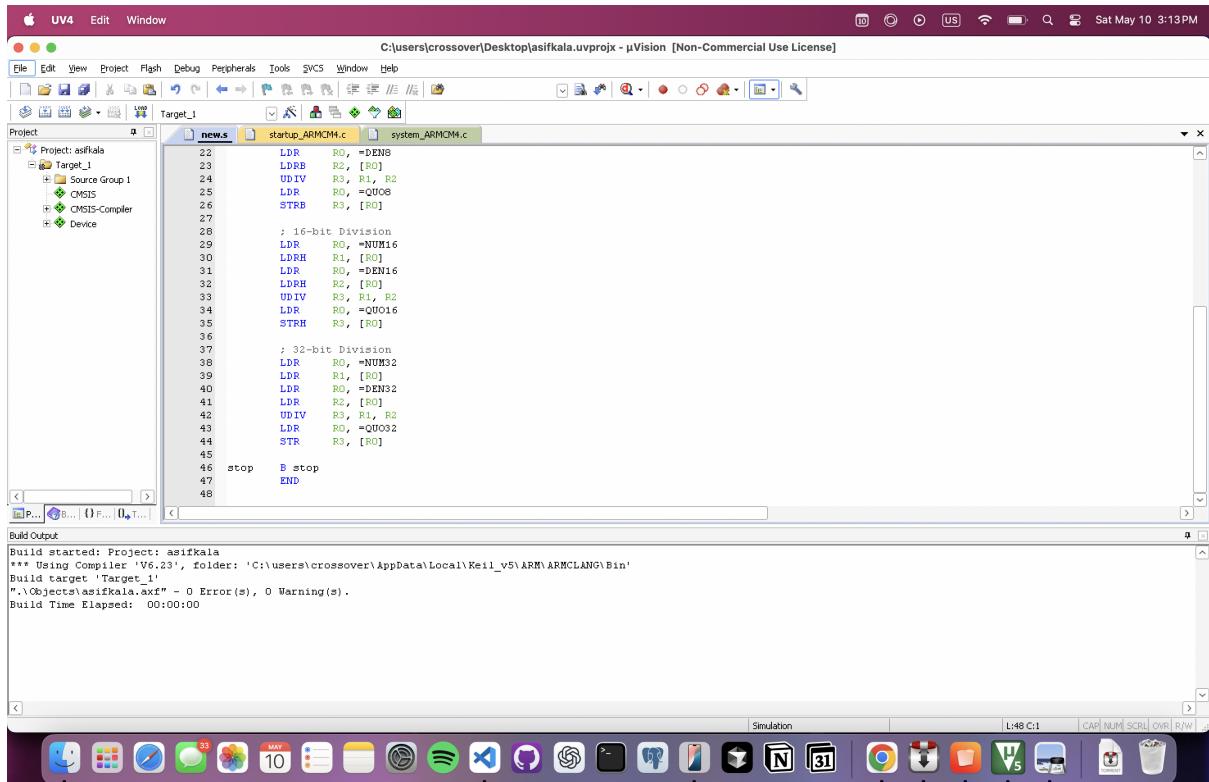


Figure 3: State of the system after the project is built.

## Register Status After Execution

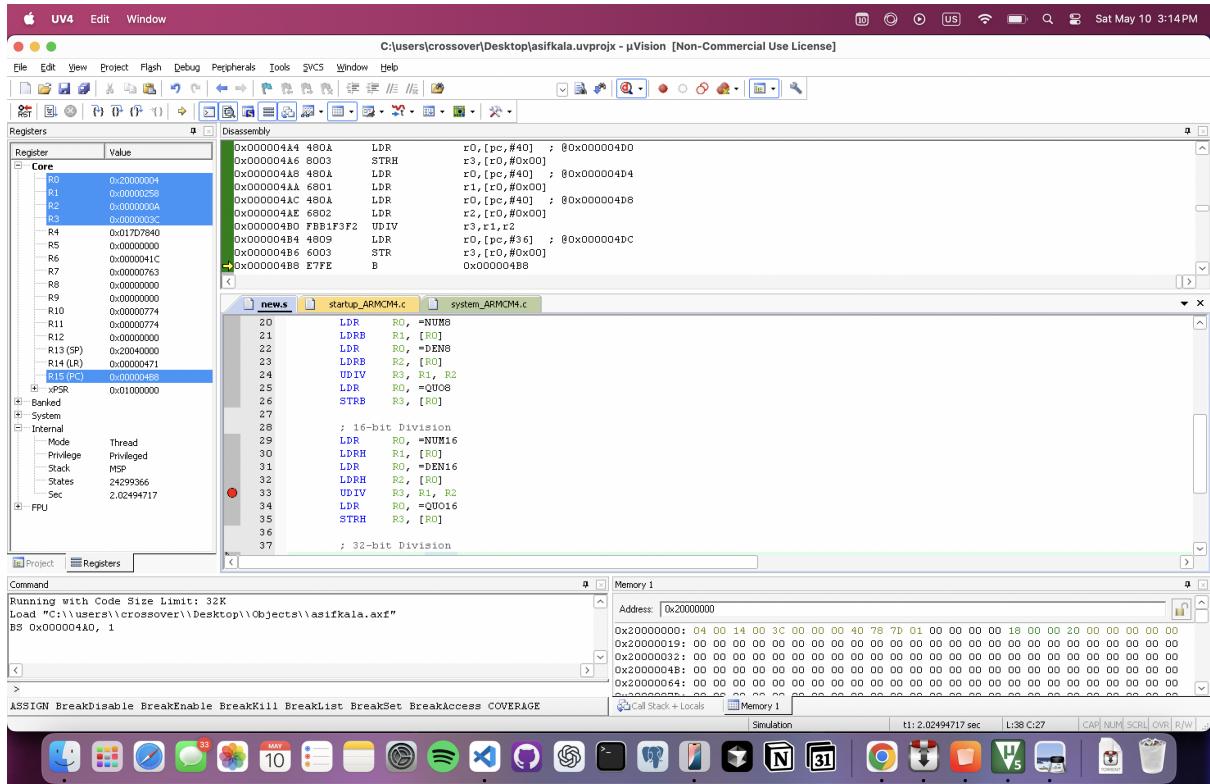
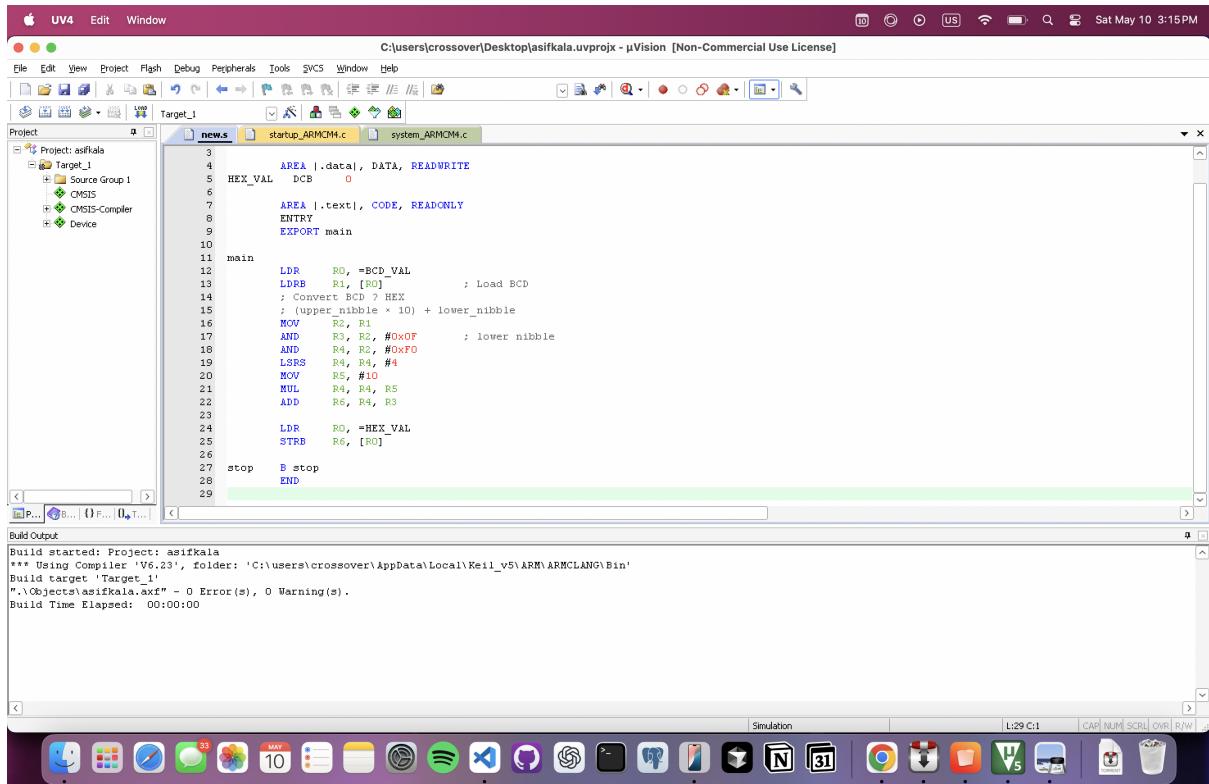


Figure 4: State of the system after the code has been executed.

### 3 Convert a BCD number in memory to its equivalent HEX number

#### Build Status



The screenshot shows the µVision 4 IDE interface. The top menu bar includes File, Edit, View, Project, Flash, Debug, Peripherals, Tools, SVCS, Window, and Help. The title bar indicates the project is 'asifkala' and the file is 'startup\_ARMCM4.c'. The main window displays the assembly code for the 'main' function:

```
3      AREA .data!, DATA, READWRITE
4      HEX_VAL    DCB    0
5
6      AREA .text!, CODE, READONLY
7      ENTRY
8      EXPORT main
9
10
11     main
12        LDR    R0, =BCD_VAL
13        LDRB   R1, [R0]          ; Load BCD
14        : Convert BCD ? HEX
15        : (upper_nibble * 10) + lower_nibble
16        MOV    R2, R1
17        AND    R3, R2, #0x0F      ; lower nibble
18        AND    R4, R2, #0xF0
19        LSRS   R4, R4, #4
20        MOV    R5, #10
21        MUL    R4, R4, R5
22        ADD    R6, R4, R3
23
24        LDR    R0, =HEX_VAL
25        STRB   R6, [R0]
26
27        stop    B stop
28
29        END
```

The bottom pane shows the 'Build Output' window with the following log:

```
Build started: Project: asifkala
*** Using Compiler 'V6.23', folder: 'C:\users\crossover\AppData\Local\Keil_v5\ARM\ARMCLANG\Bin'
Build target 'Target_1'
".\Object\asifkala.axf" - 0 Error(s), 0 Warning(s).
Build Time Elapsed: 00:00:00
```

Figure 5: State of the system after the project is built.

## Register Status After Execution

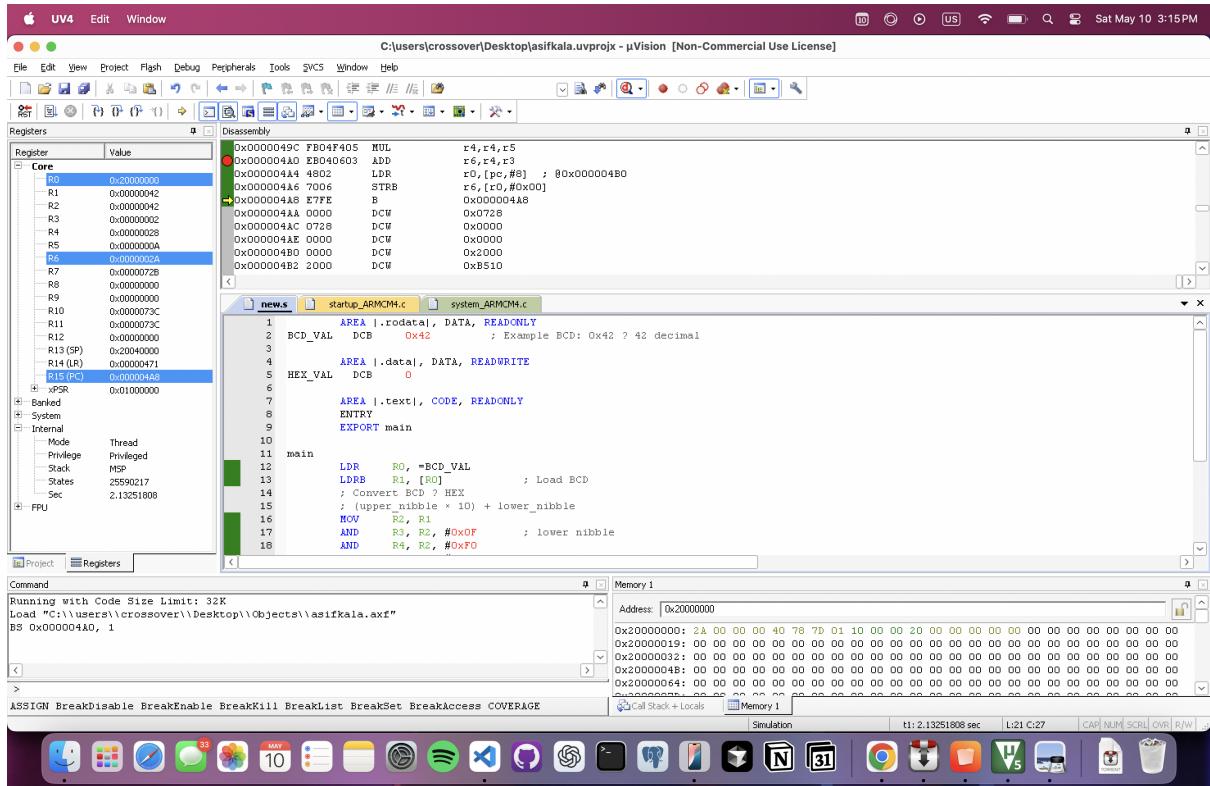


Figure 6: State of the system after the code has been executed.

## 4 Convert a hexadecimal number to its equivalent BCD number

### Build Status

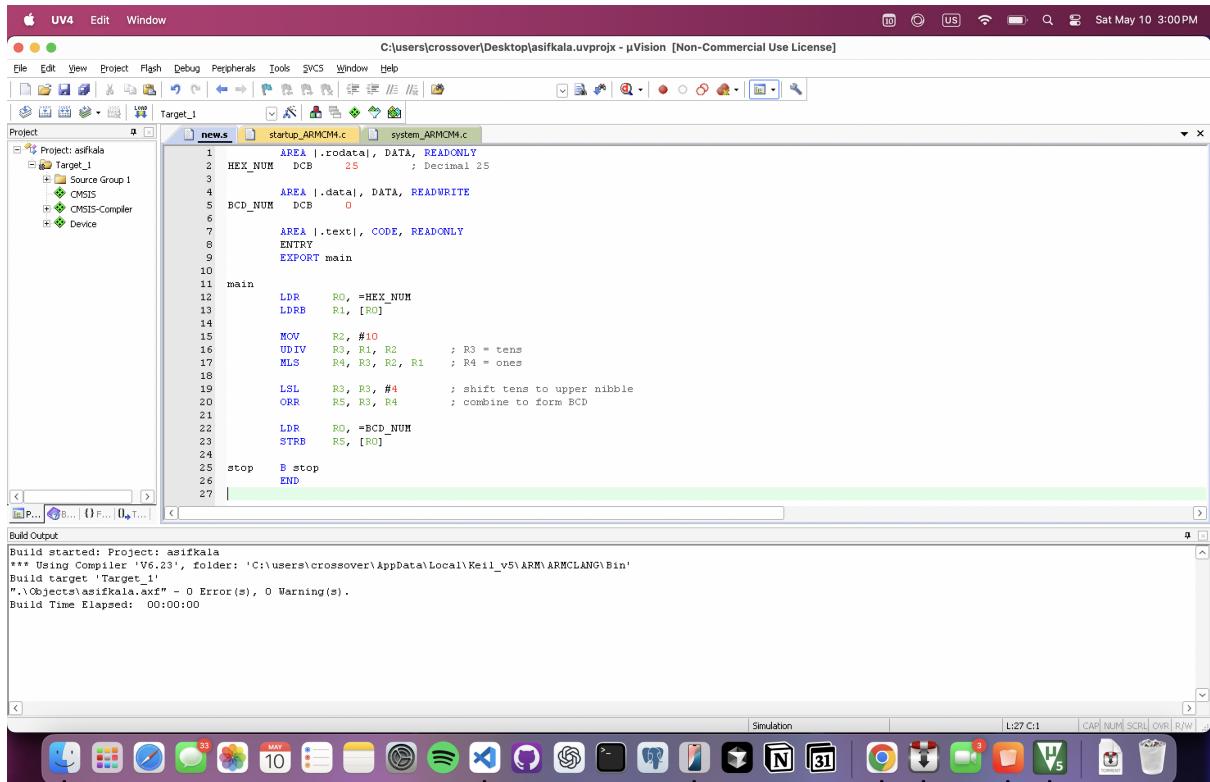


Figure 7: State of the system after the project is built.

## Register Status After Execution

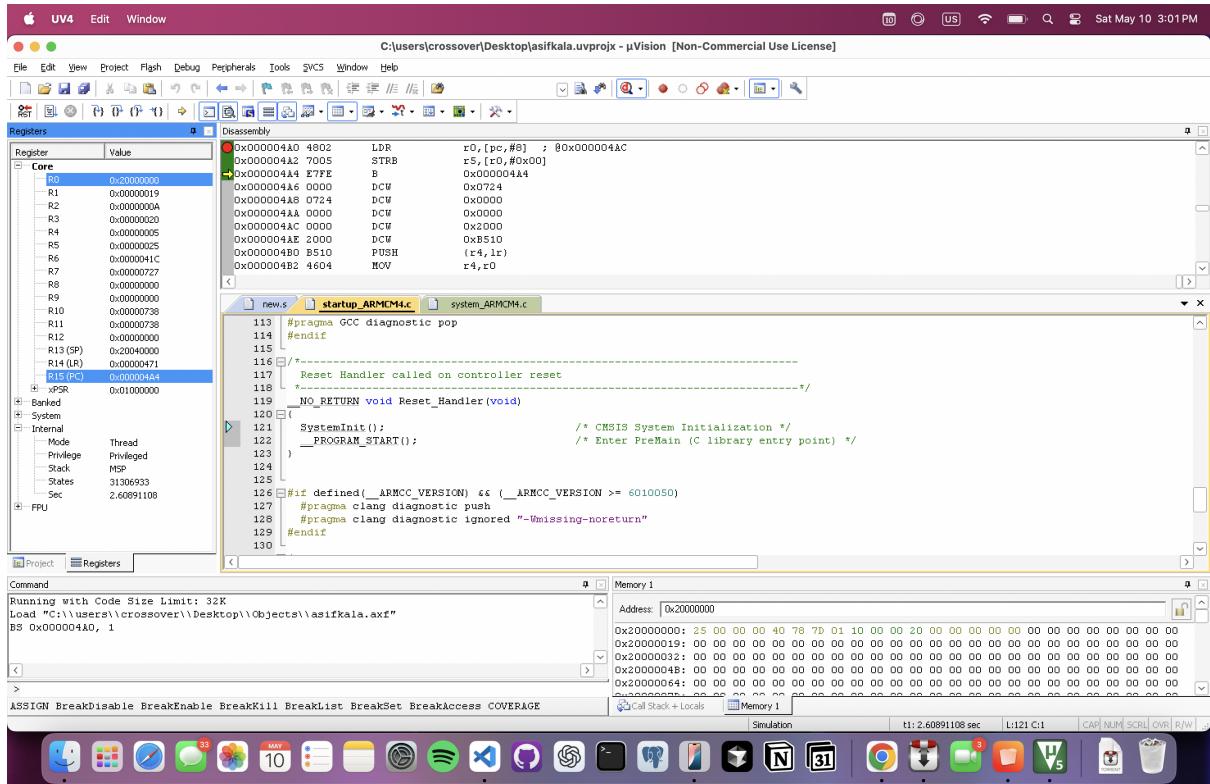
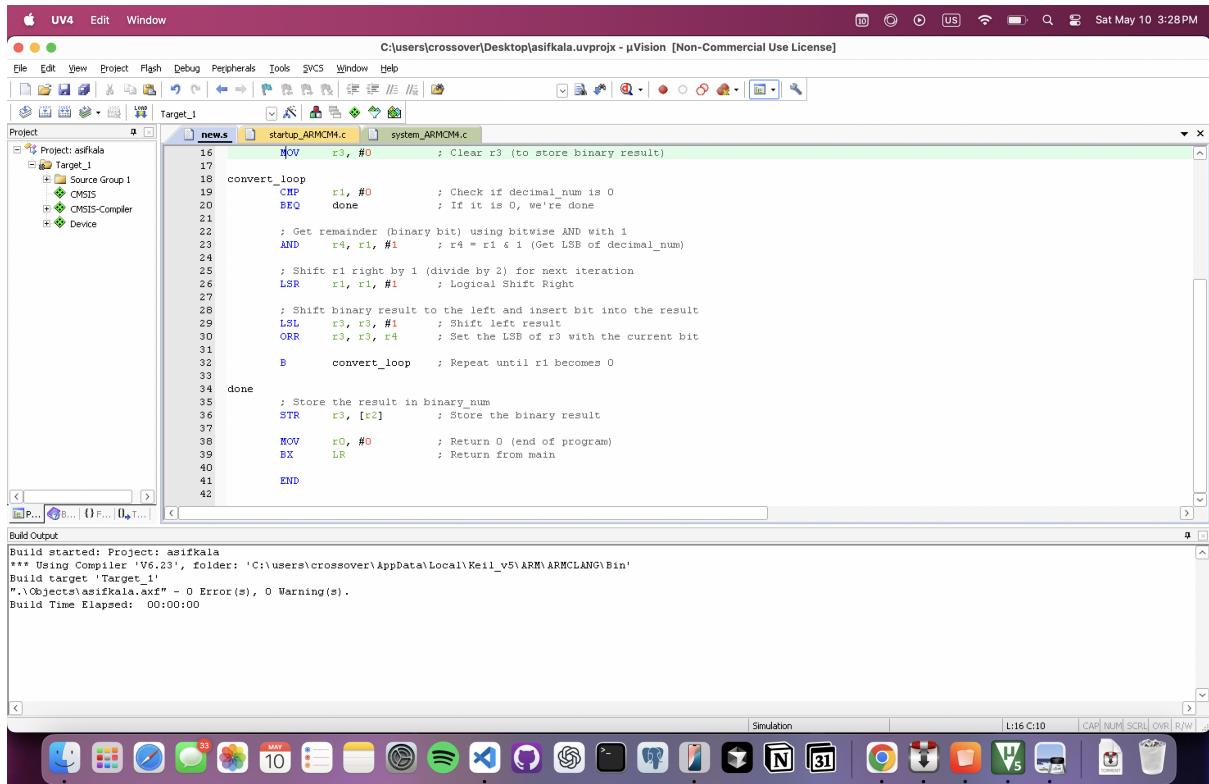


Figure 8: State of the system after the code has been executed.

## 5 Convert a decimal number to its equivalent binary number

### Build Status



The screenshot shows the µVision IDE interface. The main window displays an assembly code listing for a project named 'asifkala'. The code implements a function to convert a decimal number to binary. It includes comments explaining the logic: clearing r3, checking if the decimal number is zero, getting the remainder (using bitwise AND with 1), shifting r1 right by 1, shifting the binary result left, inserting the remainder bit into the result, and repeating the loop until r1 becomes zero. Finally, it stores the result in binary\_num and returns from the main function.

```

UV4 Edit Window
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
C:\users\crossover\Desktop\asifkala.uvprojx - µVision [Non-Commercial Use License]
Project Target_1 startup_ARMCM4.c system_ARMCM4.c
16    MOV   r3, #0          ; Clear r3 (to store binary result)
17
18 convert_loop
19    CMP   r1, #0          ; Check if decimal_num is 0
20    BEQ   done             ; If it is 0, we're done
21
22    ; Get remainder (binary bit) using bitwise AND with 1
23    AND   r4, r1, #1        ; r4 = r1 & 1 (Get LSB of decimal_num)
24
25    ; Shift r1 right by 1 (divide by 2) for next iteration
26    LSR   r1, r1, #1        ; Logical Shift Right
27
28    ; Shift binary result to the left and insert bit into the result
29    LSL   r3, r3, #1        ; Shift left result
30    ORR   r3, r3, r4        ; Set the LSB of r3 with the current bit
31
32    B     convert_loop     ; Repeat until r1 becomes 0
33
34 done
35    ; Store the result in binary_num
36    STR   r3, [r2]          ; Store the binary result
37
38    MOV   r0, #0          ; Return 0 (end of program)
39    BX   LR               ; Return from main
40
41 END

```

Build Output

```

Build started: Project: asifkala
*** Using Compiler "V6.23", folder: 'C:\users\crossover\AppData\Local\Keil_v5\ARM\ARMCLANG\Bin'
Build target: 'Target_1'
".\Object\asifkala.axf" - 0 Error(s), 0 Warning(s).
Build Time Elapsed: 00:00:00

```

Figure 9: State of the system after the project is built.

## Register Status After Execution

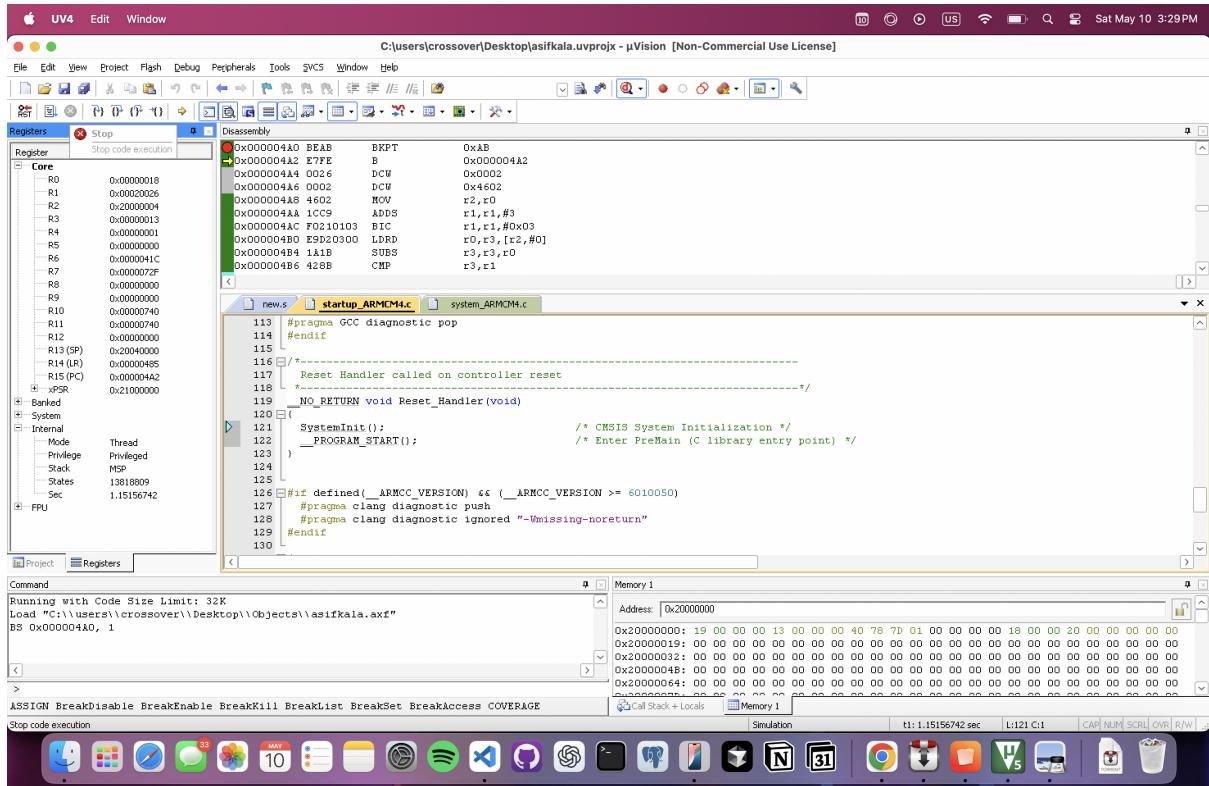


Figure 10: State of the system after the code has been executed.