ECE 2510

Lab 3

Objectives:

- Practicing ARM assembly instructions and assembler directives
- Perform multi-precision arithmetic
- Practicing RTL

Task 1

Create a new ASM project
• Type the code into Keil and simulate it.
• Step through each instruction and watch the destination register(s) change. If the instruction writes to a register, take a screenshot of the register window. If the instruction modifies memory, take a screenshot of the memory window at the correct address

```
MOV R0, #0

MOV R0, #0x27

MOV R1, #5

ADD R0, R0, R1

LDR R2, =0x20000000

STRB R0, [R2]

STRB R1, [R2]

LDR R3, =0x20000000

LDRB R0, [R3]

LDRB R0, [R3]
```

Task 2

The ASPR is a 32-bit register that records specific events during program execution. For example, if an instruction causes an overflow, the V flag will be set high. If the result of an instruction is zero, the Z flag will be set, etc

Create a new ASM project
• Write a program that adds the following 32-bit integers together:
0x00000000 + 0xFFFFFFFF + 0x00000001 + 0x12345678 + 0x89ABCDEF + 0x7FFFFFFF + 0x80000000 + 0x00000001 + 0xFFFFFFFF
• Simulate the program step-by-step and take a screenshot of the register window after each step.

Task 3

Write an assembly program that performs multiprecision subtraction on the following 64-bit numbers.
0x0FDF2DC20FDF2DC2 - 0x125CE62C125CE62C
• Write a program that subtracts the numbers one byte at a time, starting with the least significant byte.
• Store the results starting at address 0x200000028.

• Include your assembly code in your report

Task 4

For each of the following tasks, label the values initialized by DCB or DCD as array/var1, 2, etc. For example, if asked to define 2 separate arrays, the first array would be labeled array1, and the second array would be labeled array2.

- Use **DCB** to define **0xB5**
- Use **DCB** to define the following: **{0x15, 0x1A, 0x15, 0x2C}**
- Use **DCD** to define {**0x230D, 0x203F, 0x02D0, 0x00A2}**
- Use **SPACE** to define **100 bytes** in memory
- Use **Fill** to set **80 bytes** in memory to 0x1F
- Place the following directive directly below the first (where you defined **0xB5**): **DCB 0x47**
- For the report, take snapshots of each memory location used. Contrast the differences between **DCB**, **Fill,** and **SPACE**.

Task 5

Write an assembly program that multiplies the following 32-bit numbers together using signed multiplication. After each multiplication, truncate the result to 2 bytes. Include your assembly code in your report.
array = {-2560, 740, -840, -2560, 4580}.

Example:

Multiply -2560* 740. Then truncate the result to 2 bytes. Multiply that by -840, truncate result to 2 bytes, etc.

Task 6

Write a program that multiplies the following numbers together.
Verify the correctness of your results by comparing them with your expected values. Include your assembly code in your report.
5 * 6 = ?
0xFF * 0x3 = ?
0b01101100 * 0b00001010 = ?
255 * 255 = ?

Lab Report

Task 1:

- **ARM Assembly** code with **Register Transfer Level (RTL)** and addressing modes for each instruction.
- Screenshots of each step of the program:
    - If the instruction modifies the register, screenshot the **Register Window**.
    - If the instruction modifies memory, screenshot the **Memory Window** displaying the correct address.

Task 2:

- **ARM Assembly** code.
- Screenshot of each step of the program.

- Hand calculated values

Task 3:

- Code

Task 4:

- Code.
- Memory snapshots.
- Comment on the difference between the assembler directives used (**DCB, DCD, SPACE**).

Task 5:

- Code

- Screenshots of the registers after each step

Task 6:

- Code
- Hand calculated values
- Screenshots of the multiplication products for verification