

# CT Lab: Arithmetic Operations

## 1 Introduction

In this lab you will write your own assembly programs to perform additions and subtractions on the CT Board.

## 2 Learning Objectives

- You can apply addition and subtraction operations in assembly programs.
- You understand the implications of these operations on the processor flags.
- You can implement additions, which exceed the word size of the processor.

## 3 Task 1 – Sum and Difference

### 3.1 Introduction

You shall write a program that adds and subtracts two 8 bit values. To be able to observe the carry and overflow flags, the two 8 bit values will be shifted to the left by 24 bits. The right part will be filled with zeroes.

### 3.2 Implementation

Open the given project frame *sum\_diff* and expand it, so that one 8 bit value is read from the DIP-switches S15 to S8 (operand A) and another from S7 to S0 (operand B). Both operands shall be expanded to 32 bit, as mentioned in the introduction. Display the most significant byte of the sum on LED7 to LED0 and the most significant byte of the difference on LED23 to LED16.

- The instruction **LSLS** (i.e. **LSLS R1, R1, #24**) shifts the content of register R1 to the left by 24 bits. The right part gets filled with zeroes. You will learn about the shift instructions in a future lesson.

Additionally you shall display the flags set by these two operations on the LEDs. The flags from the addition shall be displayed on LED15 to LED12 and the ones from the subtraction on LED31 to LED28 (See Figure 1).

- To read the processor flags, you can use the **MRS** instruction.  
i.e.: **MRS R1, APSR** copies the content of the application program status register to register R1.
- The four flag bits are positioned from bit 31 to bit 28. To display these on the LEDs, you need to shift those bits to the right by 24 bits. Use the **LSRS** instruction.  
i.e.: **LSRS R1, R1, #24** shifts the content of R1 to the right by 24 bits.



Use Figure 2 to verify the plausibility of your results.

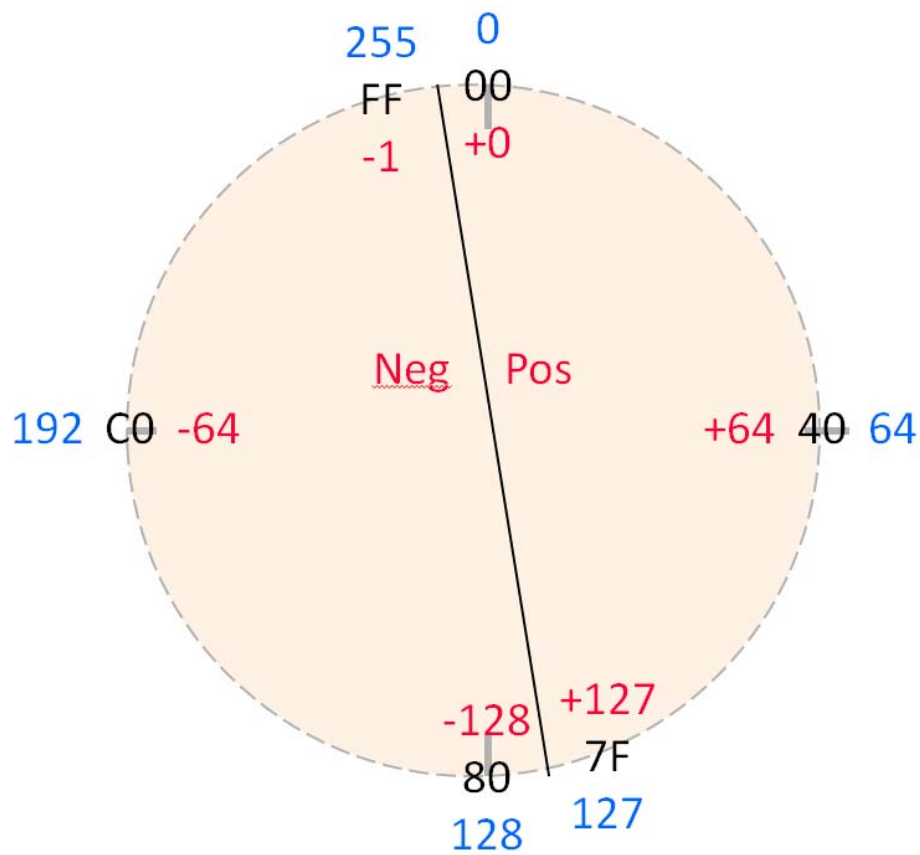


Figure 2: 8 bit circle of numbers



## 5 Task 3 – Arithmetic Commands (optional)

Assemble, link and load the given project *arith\_operations* with the debugger. Execute the program step by step. Watch and comprehend the changes in the registries and flags.

## 6 Grading

The working programs have to be presented to the lecturer. The student has to understand the solution / source code and has to be able to explain it to the lecturer.

Task	Criteria	Weight
1	The program meets the requirements described in chapter 3.2.	2/4
2	The program meets the requirements described in chapter 4.1.	2/4