EEL319 Digital Signal Processing

Experiment 2: Number Representation,Addressing modes, and Instructions

Aim:
The aim of this part of the experiment is to familiarize you with the number representations, and the addressing modes possible in C5510. You are also expected to get familiar with various types of instructions.
Goals:
At the end of the experiment, you should be able to run programs (in C and assembly) that use different Q formats, and load and store in different formats. You should be able to add/multiply numbers accessed in different addressing modes, and get familiar with the instruction set.
Procedure:
PART A: (Assembly)

1. Multiply two numbers in Q15 format, and store the result. How will you convert the result the format automatically? Repeat for addition of two numbers.
2. Multiply two numbers with one in Q15 and another in Q12, and store the result. Repeat for addition of two numbers. Make sure you read the instruction manual for each instruction that you use. Some MPY instructions use bit 16-31 of the accumulators for example. Check out various instructions for addition/multiplication.
3. Using simple examples, demonstrate *various direct and indirect, absolute, and other addressing modes in 1*. In one of the examples, demonstrate the use of a dbl-pair instruction. You might have to load the HI portion of the Accumulator – do this using a MOV with a shift. For all instructions used, read up relevant page of the manual.

PART B: (C code)

The size of each C data type for C55xx processor is listed below:

Data Type	**Size in bits**
char	16
short	16
Int	16
Long	32
Long long	40
float	32
double	64

To multiply two numbers a and b, simply declare them to be int, and the result of their product as long:

result = (long) a*b;

This ensures that single instruction multiplies are used in the compiled code.
You can then use:

y = (int)((result>>15)&0x0000ffffl);

to perform a Q15 format multiplication. This ensures that the result of the multiply is a 16 bit number.
Under the project build options menu, you can choose various compiler optimization settings. In addition, C55xx intrinsics are available for performing common signal processing tasks. These are translated (replaced line by line) by the compiler to very efficient DSP code, making it much less portable. You can find more information in the "TMS320C55x optimizing C/C++ compiler users guide". We will not deal with these intrinsics in this course.

1. Write a simple C program to add two numbers in Q15 format. Repeat for multiplication.
2. Multiply two numbers with one in Q15 and another in Q12, and store the result. Repeat for addition.