**ELET3405: Course Project**

**LEGv8 Encoder**

In this course project, you will simulate a simple LEGv8 encoder. Your LEGv8 encoder will be capable of loading a specified file recording a sequence of LEGv8 instructions and outputting their corresponding machine code.

You are encouraged to use C/C++/Python/Java to finish the project.

***Input file****:*

The input file consists of a sequence of lines, each line containing a single instruction.

We will use the following types of LEGv8 instructions:

two data transfers: load double word (e.g., LDUR X2, [X20, #8]), store double word (e.g., STUR X4, [X20, #16]),

four arithmetic operations: add (e.g., ADD X1, X2, X3), subtract (e.g., SUB X1, X2, X3), and (e.g., AND X1, X2, X3), orr (e.g., ORR X1, X2, X3),

and two immediate instructions: add immediate (e.g., ADDI X1, X2, #46) and subtract immediate (SUBI X3, X4, #67).

You may assume that the input data file does not contain any errors.

***LEGv8 instruction encoder:***

Your LEGv8 instruction encoder should be able to read data from the input file and extract the code for encoding.

It should support R-type (ADD, SUB, AND, ORR), D-type (LDUR, STUR) and I-type (ADDI, SUBI) instructions.

The conversion from LEGv8 assembly language to machine code should be based on the LEGv8 reference sheet.

**BONUS:** Convert the machine code from binary to hexadecimal.

When running your simulator, it should first ask the user for the name of the input and output files. The expected response of the user is the input file name, and output file name followed by pressing the <ENTER> key.

***Output file:***

The output file should include the machine code of the instruction sequence along with the instructions, e.g.,

ADD X9, X20, X21 10001011000101010000001010001001 0x8B150289