**Lab 4 – Assembly Manipulations Instructions**

This lab includes implementing the assembler that translates the low level language to machine understandable language, which is binary numbers. For syntax, the manipulation words such as add, sub, not ,and ,or ,shra , cpy and swap are written before the Ri and Rj registers. There has to be a space between instruction word and registers. Similarly, a comma between Ri and Rj registers is needed. Addc and subc instruction words require the # key followed by the constant numbers.

The op code for the manipulation words including load, store and jump is as follow.

ADD\_IC = 4'b0000, SUB\_IC = 4'b0001, ADDC\_IC = 4'b0010,

SUBC\_IC = 4'b0011 , NOT\_IC = 4'b0100, AND\_IC = 4'b0101, OR\_IC = 4'b0110,

SHRA\_IC = 4'b0111 , ROTR\_IC = 4'b1000, ST\_IC= 4'b1001, LD\_IC = 4'b1010,

JMP\_IC= 4'b1011, CPY\_IC = 4'b1100, SWAP\_IC = 4'b1101;

The opcode is 4 bits wide whereas Ri and Rj registers are 5 bits wide. The 32 registers can be used in the assembly. Everything is completed in this lab. The assembler is used via Python language programming. To run the Python, use any python compilers with “python filename.py” command. Linux can be used as well to compile and execute the assembler. Note that the default input and output files are used.