CSCI 6461 Fall 2023 Sec 10 Group 10 Project IIa – Hand Guide

JDK Compiler compliance level : 17
JRE version : jre.full.win32.x86_64_17.0.8.v20230831-1047
Refer to "CSCI 6461 Fall 2023 Sec 10 Group 10 Project 2 – Hand Guide" to know more about our console.

*If you encounter error during program run, refer to Notes section at the end of this document to see possible fixes.

Project IIa :

The project objective is to translate the input text file into the hexadecimal code text file. It means to translate each instruction in input text file and convert it into the hexadecimal code of format XXXX YYYY where XXXX is location and YYYY is decoded instruction(address).

The actions that need to be perform are super simple to translate the file using our console, which is to just click the 'IPL' button.

Before, here is the different files description :

ProgramLoadFile.txt : This is the text file where we need to insert our high-level instructions. The program will interpret these instructions and then translate them in to hex code.

HexFile.txt : This is our primary feed to our simulator, where the decoded/translated instructions in hexadecimals are written to by our program.

InstructionsMetaData.json : This file is a json file containing of array of elements. Each element describes three fields namely instructionParts, bytePattern and opcode. This file act as a mapping between high-level instructions and hexadecimal instructions.

Running the jar file to generate the resultant hexadecimal instructions :

*It is important that the above mentioned files and our jar file should be in one folder.

1. Execute the program using command.

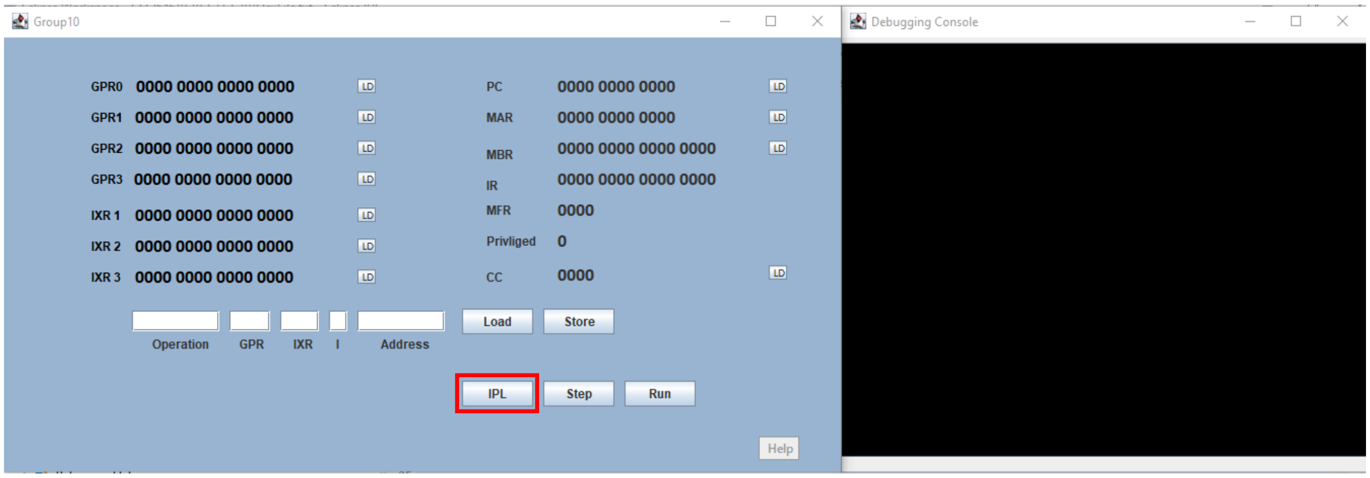
```
java -jar CSCI6461F23S10G10Project2.jar
```

2. You'll see two windows, one for simulator and another for debugger. For now, our interest is in simulator. Find the button name '**IPL**' and click on it.
3. Now open the project structure in file explorer or through command prompt. Try locating the file named 'HexFile.txt'. Try opening it. You'll find the decoded hexadecimal code instructions.

Note : Make sure that the HexFile.txt is empty before running the program to see the difference between runs. Not clearing content in this doesn't harm the program.

Running different Set of Instructions :

1. To try out different set of instructions, you just need to stop the running program if it is already running.
2. Open project structure and try to locate ProgramLoadFile.txt. Open the file and insert your set of high-level instructions. Refer to "Notes" if you encounter any errors when you run the program.
3. Try running the project again and check for HexFile.txt to locate decoded hexadecimal instructions.



GPR0 0000 0000 0000 0000
GPR1 0000 0000 0000 0000
GPR2 0000 0000 0000 0000
GPR3 0000 0000 0000 0000
IXR 1 0000 0000 0000 0000
IXR 2 0000 0000 0000 0000
IXR 3 0000 0000 0000 0000

PC 0000 0000 0000
MAR 0000 0000 0000
MBR 0000 0000 0000 0000
IR 0000 0000 0000 0000
MFR 0000
Privliged 0
CC 0000

Try out this input file where we have added all the instructions :

```
LOC 6
 Data 5
 Data 10
 Data 12
 Data 17
 Data 1
 Data 8
 LDX 1,9
 JZ 2,1,6
 JZ 2,1,7,1
 JSR 1,20
 JSR 1,21,1
 JMA 2,35,1
 JMA 2,24
 JNE 0,1,31
 JNE 0,2,14,1
 JMA 2,25
 JMA 1,24,1
 AMR 3,0,10
 SMR 2,0,5
 AMR 3,1,10,1
 SMR 2,1,5,1
 AIR 1,7
 SIR 2,3
 MLT 0,2
 DVD 2,0
 TRR 1,2
 AND 0,2
 ORR 0,1
 NOT 3
 SRC 1,2,1,1
 RRC 0,1,0,1
 IN 2,15
 OUT 1,30
 CHK 2,29
 LDX 0,18,1
 STR 1,1,15,1
 STR 1,0,20
 LDA 0,1,0,1
 JCC 0,3,31
 JCC 3,0,13,1
 RFS 17
 SOB 1,0,17,1
 SOB 3,3,18
 JGE 1,1,30,1
 JGE 3,3,1,1
 STX 0,18,1
 STX 0,6
 LOC 1024
End: HLT
```

*****Notes*****

- Make sure you give end as "**End:**" and halt as "**HLT**", data as "**Data**" and location as "**LOC**" since these are case sensitive the program might throw error. Rest all the opcodes should be given in upper case letters.
- If any instructions throws error, try writing it manually in the format XXXX YYYY, where these two strings are separated by **single space**.
- Running the simulator (clicking the load/step/run buttons) will throw error, since the logic behind implementation of the simulator to decode all the instructions is not yet implemented (as it is for project 3).
- To know more about different file formats and their usage, refer to the design document of project 2.