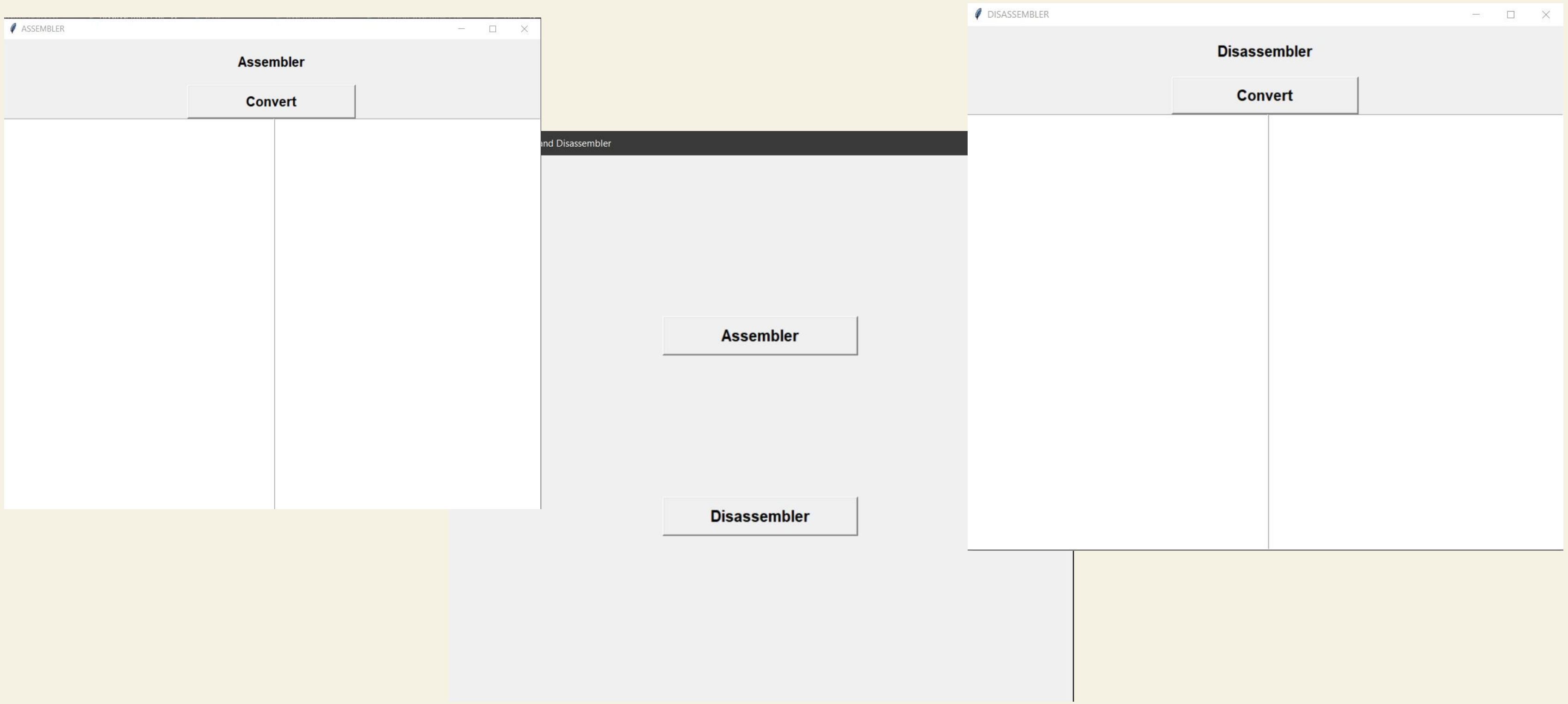
ES215 Project Report

MIPS Assembler and Disassembler

Goal

- The goal of our project is to create a **MIPS Assembler** and **Disassembler**.
- The MIPS assembler that has been coded by us takes in assembly code as input and outputs the corresponding machine language code in hexadecimal.
- Following **if wished** we can use the Disassembler programme to take in the produced machine code as input and reproduce the original MIPS Assembly Code.
- We have also built a interactive **GUI** to display these outputs, for the ease of the user.

GUI


ASSEMBLER
Assembler
Convert
nd Disassembler
Assembler
Disassembler
DISASSEMBLER
Disassembler
Convert

Pseudo Code and explanation

Assembler:

All the MIPS instructions is divided into these 5 sets

```
r_type=['add', "sllv", …]
i_type=["addi","beq",...]
j_type=["j", "jal"]
s_type=["sw","lw","lb","sb"]
shift_type=["sll","sra"]

def assemble_it (a):
    l=[] → final output stored here
    for i in range(len(a)):
        a[i]=split the instruction
```

Pseudo Code and explanation

Now checking the 0th index of split instruction and mapping the instruction according to he bifurcation in the above 5 sets and printing the instruction accordingly

```
for i in (a):
    if i[0] in r_type:  …….
    elif i[0] in i_type:  …….
    elif i[0] in j_type:  ……
    elif i[0] in s_type:  …….
    elif i[0] in shift_type: ……
    elif i[0]=="jr": ……
    elif i[0]=="syscall": ……..
return l
```

Pseudo Code and explanation

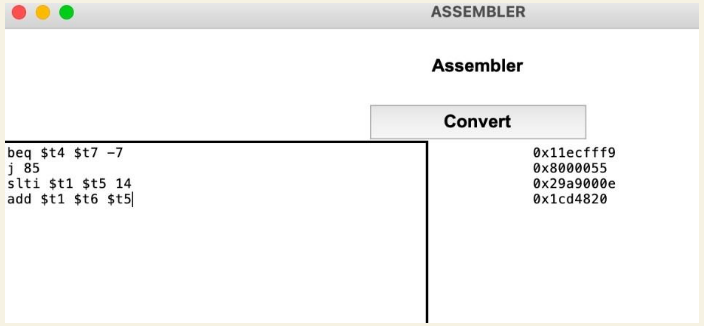
Disassembler:

```
if opcode_integer not in opcodes dictionary:
          output = "THE OPERATION IS NOT IN OUR DATA CARD

elif opcode_integer == 0:  → r type instruction

          # special r type instruction
          if(funct_integer == 0 or funct_integer == 2 or funct_integer == 3): →shift instructions

          elif(funct_integer == 8 or funct_integer == 9):  → jr/jalr
```

Pseudo Code and explanation

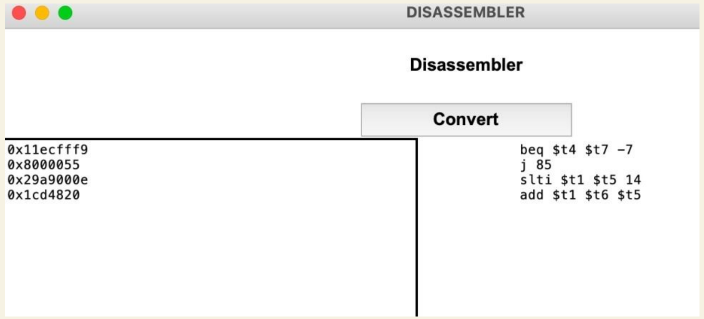
```
          elif(funct_integer == 12 or funct_integer == 13):  → syscall or break

             # mfhi, mthi, mflo, mtlo
          elif(funct_integer == 24 or funct_integer == 26): → mult and div

elif (opcode_integer == 2 or opcode_integer == 3): → j type instruction

else: → i type instruction

          if (opcode_integer == 32 or opcode_integer == 35 or opcode_integer
== 40 or opcode_integer == 43):  → lw lb sb sw

          else: → For rest of the i type instructions
```

Test Cases



```
beq $t4 $t7 -7
j 85
slti $t1 $t5 14
add $t1 $t6 $t5
```

```
0x11ecfff9
0x8000055
0x29a9000e
0x1cd4820
```



```
0x11ecfff9
0x8000055
0x29a9000e
0x1cd4820
```

```
beq $t4 $t7 -7
j 85
slti $t1 $t5 14
add $t1 $t6 $t5
```

Team

Chirag Sarda

20110047

Dhruv Parekh

20110058

Bhavesh Jain

20110038

Rahul C

20110158

Thank You!