Disassembler_Assignment

January 27, 2025

1 Disassembler Assignment

In this assignment you will be writing a disassembler. This assignment is worth 50 points. * 5 points for submitted code and well formatted comments. * 5 points for the disassembled instructions. You may submit a simple text file. * 40 points for instructions: 5 points each for each of the 8 instructions.

2 Disassembler

A disassembler is a program that will read the binary encoded instructions, interpret them, and present them back to the user in a human readable assembly language. You may have used several examples of a disassembler without realizing; an example is the GNU Debugger (GDB) which uses a disassembler and debugging objects or labels.

You may be curious to try one built into your linux system:

```
objdump -d <binary_executable> | less
```

2.1 Instructions and formats

Type	funct7	rs2	rs1	funct3	rd	Opcode
R	7	5	5	3	5	7
I	12		5	3	5	7
S	imm[11:5]	5	5	3	imm[4:0]	7
SB	imm[12|10:5]	5	5	3	imm[4:1|11]	7
U	20				5	7

Here is an example to get started. We'd like to know what the opcode is to start, then the value of rd or immediate.

[1]:
```
import numpy as np
```

[2]:
```
instructions_as_bytes = np.fromfile('risc-v_instructions.bin', dtype=np.int32)
# You might also seek to use python's file reader directly
with open('risc-v_instructions.bin', 'rb') as rv_instrs:
    binary_instructions = rv_instrs.read()
print(bin(binary_instructions[0]))
```

```
0b10000011
```

[3]:
```
instructions_as_bytes.shape[0], len(binary_instructions)
```

[3]:
```
(8, 32)
```

[4]:
```
bin(instructions_as_bytes[0] & (2**7 - 1))
```

[4]:
```
'0b11'
```

If we examine the reference sheet, we see that for a value of `0b11` the instructions must be: `lb`, `lh`, `lw`, `lbu`, or `lhu`. We will need to check the higher bits from `funct3` to be sure which specific one.

[5]:
```
bin((instructions_as_bytes[0] >> 7) & (2**8 - 1))
```

[5]:
```
'0b1000111'
```