Report

- Kotlo Charusheel

Input is given in the form of a file.So,I had read the file and stored it into a vector of strings (since c++ supports vectors(similar to array) and strings).Converted each hex instruction into a binary instruction by using “hextobin”(created by me).After that I pass both hex instruction and binary instruction into a function “instructions”(created by me).In this function I categorized the type of instruction using opcode obtained from the binary instruction and then for each type of instruction I had written different functions i.e;

- “Rtype” for R-type instruction
- “Itype” for I-type instruction
- “Stype” for S-type instruction
- “Utype” for U-type instruction
- “B1type” for B-type instruction **without labels**
- “B2type” for B-type instruction **with labels**
- “J1type” for B-type instruction **without labels**
- “J2type” for B-type instruction **with labels**

Rtype :

As R-type instructions have a single opcode.so,I didn’t pass that as an argument to the Rtype function,passed bin instruction as string as arguments.To calculate rs1,rs2,rd , I

used a function “bintounsignedint” to part of bin string into unsigned integer and differentiated add, sub, and, or, xor, sll, srl, sra using func7 and func3 respectively.After that I added the respective operation string to ans[i] where ans is the output vector of strings.

```
void Rtype(string s,vector<string>&ans,int i
```

Itype :

As I-type instructions have a multiple opcodes.so,I had passed that as an argument to the Itype function.To calculate the immediate value , I used “bintosignedint” function ,passed bin instruction as string as arguments.To calculate rs1,rd , I used a function “bintounsignedint” to part of bin string into unsigned integer and differentiated addi, andi, ori, xori, slli, srli, srai, ld, lw, lh, lb, lwu, lhu, lbu, jalr using opcode,func7 and func3 respectively.After that I added the respective operation string to ans[i] where ans is the output vector of strings.

```
void Itype(string s,string opcode,vector<string>&ans,int i)
```

Stype :

As S-type instructions have a single opcode.so,I didn’t pass that as an argument to the Stype function,passed bin instruction as string as arguments .To calculate the immediate value I used “bintosignedint” function i.e; rearranged the bits

and then passed it to “bintosignedint” function .To calculate rs1,rs2 , I used a function “bintounsignedint” to part of bin string into unsigned integer and differentiated sd, sw, sh, sb using func3 respectively.After that I added the respective operation string to ans[i] where ans is the output vector of strings.

```
void Stype(string s,vector<string>&ans,int i)
```

Utype :

As U-type instructions have multiple opcodes.so,I had passed opcode as an argument to the Utype function and even passed hex instruction and bin instruction as strings .To calculate the immediate value I had directly added to the required substring from the hex instruction.To calculate rd I used a function “bintounsignedint” to part of bin string into unsigned integer and differentiated lui , auipc using opcode respectively.After that I added the respective operation string to ans[i] where ans is the output vector of strings.

```
void Utype(string sbin,string shex,string opcode,vector<string>&ans,int i)
```

B1type(without labels) :

As B-type instructions have a single opcode.so,I didn’t pass opcode that as an argument to the B1type function and passed bin instruction as string as argument .To calculate the immediate value I had used “bintosignedint” i.e; rearranged

the bits and then passed it to “bintosignedint” function.To calculate rs1,rs2 I used a function “bintounsignedint” to part of bin string into unsigned integer and differentiated beq, bne, blt, bge, bltu, bgeu using func3 respectively.After that I added the respective operation string to ans[i] where ans is the output vector of strings.

```
void Btype1(string sbin,vector<string>&ans,int i)
```

J1type(without labels) :

As J-type instructions have a single opcode.So,I didn’t pass that as an argument to the J1type function and passed bin instruction as strings .To calculate the immediate value I had used “bintosignedint” i.e; rearranged the bits and then passed it to “bintosignedint” function.To calculate rd , I used a function “bintounsignedint” to part of bin string into unsigned integer. After that I added the respective operation string to ans[i] where ans is the output vector of strings.

```
void Jtype1(string sbin,vector<string>&ans,int i)
```

B2type(with labels) :

As B-type instructions have a single opcode.so,I didn’t pass opcode that as an argument to the B2type function and passed bin instruction as string as argument .To calculate the immediate value I had used “bintosignedint” i.e; rearranged

the bits and then passed it to “bintosignedint” function.To calculate rs1,rs2 I used a function “bintounsignedint” to part of bin string into unsigned integer and differentiated beq, bne, blt, bge, bltu, bgeu using func3 respectively. I go to ans[i-(imm/4)] and add “L”+label(label no.) at the start .After that I added the respective operation string to ans[i] with label where ans is the output vector of strings.

```
void Btype2(string sbin,vector<string>&ans,int i,int&label)
```

J2type(without labels) :

As J-type instructions have a single opcode.So,I didn’t pass that as an argument to the J2type function and passed bin instruction as strings .To calculate the immediate value I had used “bintosignedint” i.e; rearranged the bits and then passed it to “bintosignedint” function.To calculate rd I used a function “bintounsignedint” to part of bin string into unsigned integer.I go to ans[i-(imme/4)] and add “L”+label(label no.) at the start . After that I added the respective operation string to ans[i] with label where ans is the output vector of strings.

```
void Jtype2(string sbin,vector<string>&ans,int i,int&label)
```

Note : If label is out of bounds i.e; (i+imm/4) < 0 or (i+imm/4) > n(no. of instructions) then just performing same as B1,J1 respectively