CO_Project2

S1081535_2021/05/18

設計理念

1.像Project_1一樣，讀取外部文字檔，並針對每一條Instruction做整理：

- check_label()：檢查此字串是否含label。是則將label的名字存入label_name陣列中，且在label_loc陣列相對應位置紀錄起始Instruction是第幾個，方便Taken時做使用。若有需求則整理instruction字串，只剩下operation與PC部分。
- st_pc()：檢查文字檔中是否有PC。有則存放進pc陣列當中；反之由0開始，相隔4，分別給PC值，做index之用。
- cut()：將要執行的動作單獨存放在act陣列，方便判斷operation的類別。

2.由透過cut()存取的Instruction之act要執行動作，將Instruction分成Branch operations和Arithmetic & Logical operations：

- Branch：包含b,beq,beqz,bge,bgez,bgt,bgtz,ble,blez,blt,bltz,bne,bnez共13種operations。
- Arithmetic：addi,add,mul共3種operations。
- Logical：and,andi,or,ori共4種operations。
- 其他：li,move operation。

3.將整理好的Instruction逐一執行，若為Branch operation則

- 先進行2-bits history prediction(predict())：先利用get_index()依照entry取的index(pc_index)，透過bht二維陣列找到2-bits history，再查找對應的2BC，則為預測結果(predict_value)。
- 再檢查實際行為(actual())：根據operation，整理出 rs 或 rs1 ,rs2 以及label，將指定的Register(s)做比較，並得出實際結果是否Taken。若為Taken則用find_label()從label_name陣列中找到一樣的名稱，並將下一個執行為第i_next個的Instruction記錄下來存下來。在後輸出實際結果(actual_value)。
- 依實際動作更新BHT(updatebht())：更新當下bht[pc_index]陣列中的2BC之值，並將2-bits histort 依Taken(1)或Nontaken(0)向左推一位。
- 比較actual_value和predict_value，輸出predictor是否預測正確以及輸出更新後的BHT。
- 如果實際為Taken，則須將下一個須執行的Instruction(i_next)傳給i_current；反之則繼續接續的Instruction。

4.若為Other operations則先整理出 rd ,rs1 ,rs2 (若無值則留空)，執行相對應動作，並依照Instruction更改Register：R0 ~ R31(reg_value)之值。

5.重複執行第3,4步驟值到Instruction結束。(例：Branch至End且接下來沒有其他Instructions)。

程式操作與說明

- 編譯執行程式前，請先輸入讀入的檔案名稱

```
#include<fstream>
using std::ifstream;
using std::ofstream;
ifstream infile("test.txt", ios::in);
//ofstream outfile("ans.txt", ios::out);
```

- Instruction 只接受設計理念中第二點所提及的operation，其餘並不適用，且有可能造成Branch錯誤。
- Label後要加冒號。
- 文字檔中可提供PC(0x....)，也可不提供。
- Resgister名稱請使用R0~R31(R0恆等於0)。
- 文字檔中可以註解：若以行為註解請在每行最開頭加入“/”；再Instruction後註解請在註解前加“/”和Instruction 後加“;”。