Final Project - simple CPU

黃士軒, 0416208

*Abstract*—用verilog模擬出一個簡單的CPU架構

Keywords—CPU; ALU; Decoder;

# Introduction

使用Verilog設計模擬出一個CPU架構，此架構包含了一個Instruction Memory(將指令讀入)，Registers(把根據指令讀到的值存入暫存)，Decoder(將指令依照固定格式放入程式)，最後經過ALU把結果運算出來存回Register(運算必須使用Gate Level)，即完成一個簡單的CPU架構。.

# Project inside

## pc\_instruction.v

這份檔案為assign原先已經提供的檔案，主要目的就是作為cpu內部的Instruction Memory，從利用data陣列給入一連串的指令，每一個clock給一串指令。

## final\_project.v

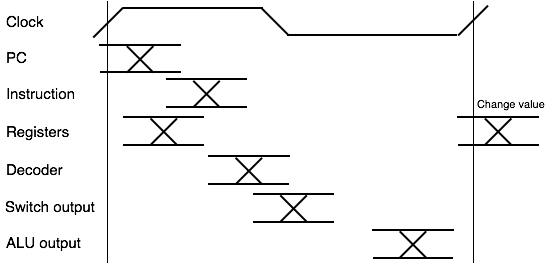
這份檔案主要是驅動集合整個專案的檔案，內部主要包含兩個function（pc\_instruction, data\_memory），從pc\_instruction得到的檔案會透過wire的方式傳給data\_memory，而在data\_memory裡面則包含了register之外，同時也包含了ALU 運算的程式，因此在final\_project的module裡面將會得到register1,2,3,4之外還有zero\_flag從data\_memory output出來。

## data\_memory.v

這份檔案裡面原本是放置register1,2,3,4，而修改過後將ALU 運算也都放在裡面，首先在每一個clock都會得到一個instruction，我們先把得到的instruction [15:8]、[7:0]，做過全部必要的運算，也就是先將得到的src1,src2做add,addi,sub,subi（透過呼叫function 的方式同步執行把結果傳回），接著在always block中利用case的方式，把instruction[19:16]的指令解析 decode，判斷instruction是需要哪一個實作方式，再把實作的結果拿給register[src1\_i]。

# Project analysis

## Timing diagram



## Event diagram

# 

# Conclusion

透過verilog的模擬，了解到CPU在運算時一個一個步驟是如過運作的，雖然只是簡單的CPU架構，但是亦足以了解到每一個block的內部運作過程與block與block之間的delay。並且透過gate level的實作，讓加法與減法的運算模擬更接近真實狀況，透過一個一個bit的控制才是真正的電路運作的方式，因此透過final project的研究，貫通整學習所學真是收益良多。