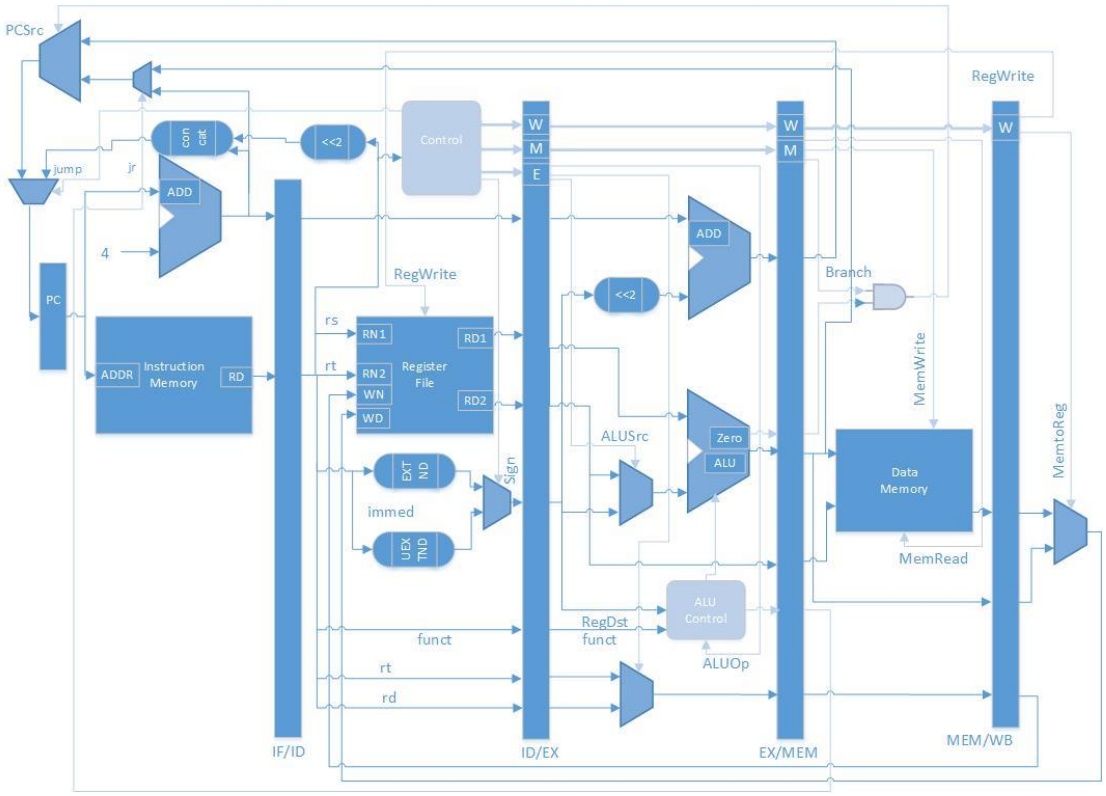
Final Project: Pipelined Processor

105 學年度第 1 學期

一、背景

這次期末 Project 是實作一個 Pipelined CPU，共有 and、or、add、sub、sll、slt、andi、beq、j、jr、lw、sw 十二個指令可以使用，這次作業讓我們了解到 pipeline 的優點以及其必須克服的種種困難。

二、方法



(1)Pipeline Registers:(ifid.v、idex.v、exmem.v、memwb.v)
我們切 pipeline 切 5 個部分，由四個 register 切分，分別為
ifid, idex, exmem, memwb 切成 IF/ID/EX/MEM/WB 五個部分，每個 register 分別把該往下一個部分的變數接為該 register 的 input，並且在每個 clock 的 posedge 時 assign 至 output reg。
四個 register 的 input 與 output 分別為:
ifid:
input:clk, PCin, InstructionIn
output:PCout, InstructionOut

idex:
input:RegWriteI, MemtoRegI, MemReadI, MemWriteI, BranchI, ALUOpI, RegDstI, ALUSrcI, PCI, RD1I, RD2I, EXTNDEDImmI, rtI, rdI, functI, clk

output:RegWrite0, MemtoReg0, MemRead0, MemWrite0, Branch0, ALUOp0, RegDst0, ALUSrc0, PC0, RD10, RD20, EXTNDEDImm0, rt0, rd0, funct0

exmem:
input:RegWriteI, MemtoRegI, MemReadI, MemWriteI, BranchI, PCI, zeroI, ALUtoDataMemAddrI, RD2toDataMemWDI, WNI, clk
output:RegWrite0, MemtoReg0, MemRead0, MemWrite0, Branch0, PC0, zero0, ALUtoDataMemAddr0, RD2toDataMemWD0, WN0

memwb:
input:RegWriteI, MemtoRegI, DataMemRDI, ALUResultI, WNI, clk
output:RegWrite0, MemtoReg0, DataMemRD0, ALUResult0, WN0

其暫存的資料多半是控制訊號，剩下的就是使用者給的值。
(2)ALU:(TotalALU.v)
這次的 ALU 依照要求，是使用上次期中 project 的 ALU，但是把除法器的部分移除，其他功能照舊，只是 input 的訊號增加了給 shifter 的 shamt 和 Zero 的 output 訊號，然後將 ALU 的 output 接上 EX/MEM register。
(3)ALUControl:(alu_ctl.v)
除了原本會針對 ALUOp 以及 funct 來判斷要送出哪種 Operation 訊號給 ALU，還增加了給 JRMUX 的控制訊號，如果是 jr 指令，則為 1，其餘則為 0。
(4)Extend:(sign_extend.v、unsign_extend.v)
Extend 有兩種，一種是有號數的擴充，另一個是無號數的擴充，前者用於大部分 I-type 的訊號後者用於 ANDI，透過 Control 裡的 sign 控制訊號來控制。
(5)JUMP: (mips_pipeline.v)
除了要應付 J-type 指令，還有 jr 指令，所以多加了 JRMUX 的多工器來判斷說是否需要用 jr 新的值來取代 PC。

三、結果

and $s2, $s0, $s2:

```
VSIM 50> run
#                    0, reading data: InstrMem[         x] => xxxxxxxx(hex)
# 18446744073709551615, PC:         x
#                    0, reading data: InstrMem[         0] => 02509024(hex)
#                    0, PC:         0
#                    1, reg_file[16] =>          1 (Port 2)
#                    1, reg_file[18] =>          3 (Port 1)
#                    1, PC:         0
#                    1, wd:          x
#                    1, AND
#
#                    2, PC:         0
#                    2, wd:          x
#                    2, AND
#
#                    3, PC:         0
#                    3, wd:          x
#                    3, AND
#
#                    4, reading data: InstrMem[         4] => xxxxxxxx(hex)
#                    5, reg_file[18] <=          1 (Write)
#                    4, PC:         4
#                    4, wd:          1
#                    4, AND
#
#                    5, reg_file[ x] =>          x (Port 2)
#                    5, reg_file[ x] =>          x (Port 1)
# control_single unimplemented opcode  x
#                    6, reg_file[18] <=          1 (Write)
#                    5, PC:         8
#                    7, reg_file[18] <=          1 (Write)
#                    6, PC:         8
#                    8, reg_file[18] <=          1 (Write)
#                    7, PC:         8
#                    8, PC:         8
#                    9, PC:         8
```

or $s2, $s0, $s2:

```
VSIM 48> run
#                    0, reading data: InstrMem[         x] => xxxxxxxx(hex)
# 18446744073709551615, PC:         x
#                    0, reading data: InstrMem[         0] => 02509025(hex)
#                    0, PC:         0
#                    1, reg_file[16] =>          1 (Port 2)
#                    1, reg_file[18] =>          3 (Port 1)
#                    1, PC:         0
#                    1, wd:          x
#                    1, OR
#
#                    2, PC:         0
#                    2, wd:          x
#                    2, OR
#
#                    3, PC:         0
#                    3, wd:          x
#                    3, OR
#
#                    4, reading data: InstrMem[         4] => xxxxxxxx(hex)
#                    5, reg_file[18] <=          3 (Write)
#                    4, PC:         4
#                    4, wd:          3
#                    4, OR
#
#                    5, reg_file[ x] =>          x (Port 2)
#                    5, reg_file[ x] =>          x (Port 1)
# control_single unimplemented opcode  x
#                    6, reg_file[18] <=          3 (Write)
#                    5, PC:         8
#                    7, reg_file[18] <=          3 (Write)
#                    6, PC:         8
#                    8, reg_file[18] <=          3 (Write)
#                    7, PC:         8
#                    8, PC:         8
#                    9, PC:         8
```

add $s1, $s0, $s1, 0:

```
VSIM 44> run
#                    0, reading data: InstrMem[         x] => xxxxxxxx(hex)
# 18446744073709551615, PC:         x
#                    0, reading data: InstrMem[         0] => 02509020(hex)
#                    0, PC:         0
#                    1, reg_file[16] =>          1 (Port 2)
#                    1, reg_file[18] =>          3 (Port 1)
#                    1, PC:         0
#                    1, wd:          x
#                    1, ADD
#
#                    2, PC:         0
#                    2, wd:          x
#                    2, ADD
#
#                    3, PC:         0
#                    3, wd:          x
#                    3, ADD
#
#                    4, reading data: InstrMem[         4] => xxxxxxxx(hex)
#                    5, reg_file[18] <=          4 (Write)
#                    4, PC:         4
#                    4, wd:          4
#                    4, ADD
#
#                    5, reg_file[ x] =>          x (Port 2)
#                    5, reg_file[ x] =>          x (Port 1)
# control_single unimplemented opcode  x
#                    6, reg_file[18] <=          4 (Write)
#                    5, PC:         8
#                    7, reg_file[18] <=          4 (Write)
#                    6, PC:         8
#                    8, reg_file[18] <=          4 (Write)
#                    7, PC:         8
#                    8, PC:         8
#                    9, PC:         8
```

sub $s2, $s0, $s2:

```
VSIM 46> run
#                    0, reading data: InstrMem[         x] => xxxxxxxx(hex)
# 18446744073709551615, PC:          x
#                    0, reading data: InstrMem[         0] => 02509022(hex)
#                    0, PC:          0
#                    1, reg_file[16] =>          1 (Port 2)
#                    1, reg_file[18] =>          3 (Port 1)
#                    1, PC:          0
#                    1, wd:           x
#                    1, SUB
#
#                    2, PC:          0
#                    2, wd:           x
#                    2, SUB
#
#                    3, PC:          0
#                    3, wd:           x
#                    3, SUB
#
#                    4, reading data: InstrMem[         4] => xxxxxxxx(hex)
#                    5, reg_file[18] <=          2 (Write)
#                    4, PC:          4
#                    4, wd:           2
#                    4, SUB
#
#                    5, reg_file[ x] =>          x (Port 2)
#                    5, reg_file[ x] =>          x (Port 1)
# control_single unimplemented opcode  x
#                    6, reg_file[18] <=          2 (Write)
#                    5, PC:          8
#                    7, reg_file[18] <=          2 (Write)
#                    6, PC:          8
#                    8, reg_file[18] <=          2 (Write)
#                    7, PC:          8
#                    8, PC:          8
#                    9, PC:          8
```

beq $s1, $s2, 2:

```
# Loading work.bALU
# Loading work.bitFullAdder
# Loading work.bMUX4x1
# Loading work.Shifter
# Loading work.TwoToOneMux
# Loading work.mux2
# Loading work.control_single
# Loading work.alu_ctl
# Loading work.reg_file
# Loading work.memory
VSIM 64> run
#                    0, reading data: InstrMem[         x] => xxxxxxxx(hex)
# 18446744073709551615, PC:          x
#                    0, reading data: InstrMem[         0] => 12310002(hex)
#                    0, PC:          0
#                    1, reg_file[17] =>          2 (Port 2)
#                    1, reg_file[17] =>          2 (Port 1)
#                    1, PC:          0
#                    1, BEQ
#
#                    2, PC:          0
#                    2, BEQ
#
#                    3, PC:          0
#                    3, BEQ
#
#                    4, reading data: InstrMem[        24] => 8e2f0000(hex)
#                    4, PC:         24
#                    4, BEQ
#
#                    5, reg_file[15] =>         21 (Port 2)
#                    5, PC:         24
#                    5, LW
#
#                    6, PC:         24
#                    6, LW
#
#                    7, reading data: DataMem[         2] =>        256
#                    7, PC:         24
#                    7, LW
#
#                    8, reading data: InstrMem[        28] => xxxxxxxx(hex)
#                    9, reg_file[15] <=        256 (Write)
#                    8, PC:         28
#                    8, LW
#
#                    9, reg_file[ x] =>          x (Port 2)
#                    9, reg_file[ x] =>          x (Port 1)
# control_single unimplemented opcode  x
#                   10, reg_file[15] <=        256 (Write)
#                    9, PC:         32
```

```
// beq  $s1, $s2, 2
02
00
31
12
// stall
24
00
00
00
// stall
24
00
00
00
// stall
24
00
00
00
// add  $s2, $s0, $s2
20
90
50
02
// sub  $s2, $s0, $s2
22
90
50
02
// lw   $s1, $t7, 0
00
00
2F
8E
```

slt $s2, $s0, $s2:

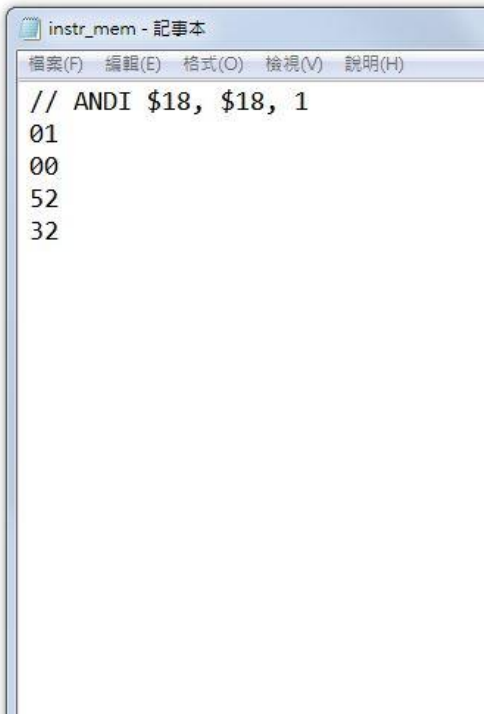
```
VSIM 52> run
#                    0, reading data: InstrMem[         x] => xxxxxxxx(hex)
# 18446744073709551615, PC:         x
#                    0, reading data: InstrMem[         0] => 0250902a(hex)
#                    0, PC:         0
#                    1, reg_file[16] =>          1 (Port 2)
#                    1, reg_file[18] =>          3 (Port 1)
#                    1, PC:         0
#                    1, wd:          x
#                    1, SLT
#
#                    2, PC:         0
#                    2, wd:          x
#                    2, SLT
#
#                    3, PC:         0
#                    3, wd:          x
#                    3, SLT
#
#                    4, reading data: InstrMem[         4] => xxxxxxxx(hex)
#                    5, reg_file[18] <=          0 (Write)
#                    4, PC:         4
#                    4, wd:          0
#                    4, SLT
#
#                    5, reg_file[ x] =>          x (Port 2)
#                    5, reg_file[ x] =>          x (Port 1)
# control_single unimplemented opcode  x
#                    6, reg_file[18] <=          0 (Write)
#                    5, PC:         8
#                    7, reg_file[18] <=          0 (Write)
#                    6, PC:         8
#                    8, reg_file[18] <=          0 (Write)
#                    7, PC:         8
#                    8, PC:         8
#                    9, PC:         8
```

sll x, $18, $16, 2:

```
VSIM 54> run
#                    0, reading data: InstrMem[         x] => xxxxxxxx(hex)
# 18446744073709551615, PC:         x
#                    0, reading data: InstrMem[         0] => 00128080(hex)
#                    0, PC:         0
#                    1, reg_file[18] =>          3 (Port 2)
#                    1, reg_file[ 0] =>          0 (Port 1)
#                    1, PC:         0
#                    1, wd:          x
#                    1, SLL
#
#                    2, PC:         0
#                    2, wd:          x
#                    2, SLL
#
#                    3, PC:         0
#                    3, wd:          x
#                    3, SLL
#
#                    4, reading data: InstrMem[         4] => xxxxxxxx(hex)
#                    5, reg_file[16] <=         12 (Write)
#                    4, PC:         4
#                    4, wd:         12
#                    4, SLL
#
#                    5, reg_file[ x] =>          x (Port 2)
#                    5, reg_file[ x] =>          x (Port 1)
# control_single unimplemented opcode  x
#                    6, reg_file[16] <=         12 (Write)
#                    5, PC:         8
#                    7, reg_file[16] <=         12 (Write)
#                    6, PC:         8
#                    8, reg_file[16] <=         12 (Write)
#                    7, PC:         8
#                    8, PC:         8
#                    9, PC:         8
```

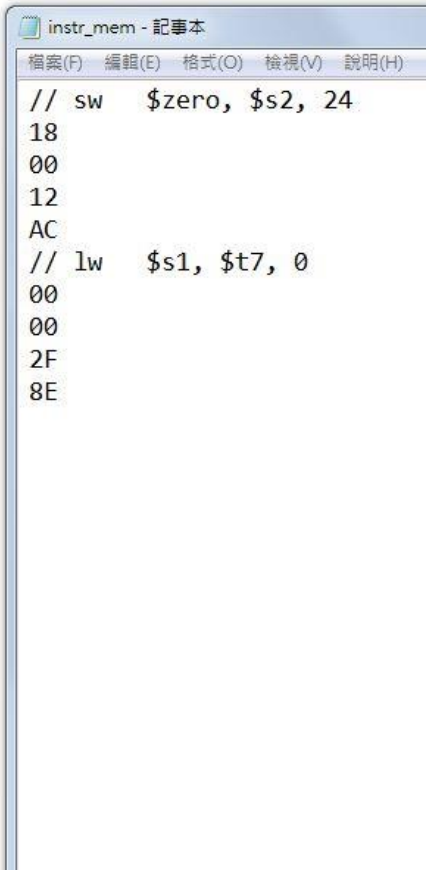
andi $18, $18, 1:

```
VSIM 58> run
#                    0, reading data: InstrMem[         x] => xxxxxxxx(hex)
# 18446744073709551615, PC:         x
#                    0, reading data: InstrMem[         0] => 32520001(hex)
#                    0, PC:         0
#                    1, reg_file[18] =>          3 (Port 2)
#                    1, reg_file[18] =>          3 (Port 1)
#                    1, PC:         0
#                    1, ANDI
#
#                    2, PC:         0
#                    2, ANDI
#
#                    3, PC:         0
#                    3, ANDI
#
#                    4, reading data: InstrMem[         4] => xxxxxxxx(hex)
#                    5, reg_file[18] <=          1 (Write)
#                    4, PC:         4
#                    4, ANDI
#
#                    5, reg_file[ x] =>          x (Port 2)
#                    5, reg_file[ x] =>          x (Port 1)
# control_single unimplemented opcode  x
#                    6, reg_file[18] <=          1 (Write)
#                    5, PC:         8
#                    7, reg_file[18] <=          1 (Write)
#                    6, PC:         8
#                    8, reg_file[18] <=          1 (Write)
#                    7, PC:         8
#                    8, PC:         8
#                    9, PC:         8
```



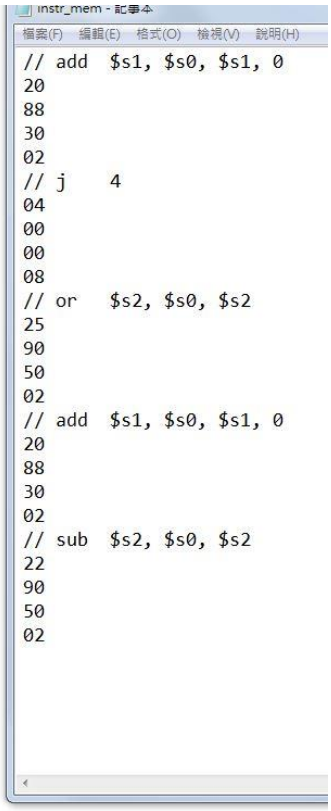
sw $zero, $s2, 24 / lw $s1, $t7, 0:

```
VSIM 60> run
#                    0, reading data: InstrMem[         x] => xxxxxxxx(hex)
# 18446744073709551615, PC:         x
#                    0, reading data: InstrMem[         0] => ac120018(hex)
#                    0, PC:         0
#                    1, reg_file[18] =>          3 (Port 2)
#                    1, reg_file[ 0] =>          0 (Port 1)
#                    1, PC:         0
#                    1, SW
#
#                    2, PC:         0
#                    2, SW
#
#                    4, writing data: DataMem[        24] <=          3
#                    3, PC:         0
#                    3, SW
#
#                    4, reading data: InstrMem[         4] => 8e2f0000(hex)
#                    5, writing data: DataMem[        24] <=          3
#                    4, PC:         4
#                    4, SW
#
#                    5, reading data: InstrMem[         8] => xxxxxxxx(hex)
#                    5, reg_file[15] =>         21 (Port 2)
#                    5, reg_file[17] =>          2 (Port 1)
#                    6, writing data: DataMem[        24] <=          3
#                    5, PC:         8
#                    5, LW
#
#                    6, reg_file[ x] =>          x (Port 2)
#                    6, reg_file[ x] =>          x (Port 1)
# control_single unimplemented opcode  x
#                    7, writing data: DataMem[        24] <=          3
#                    6, PC:        12
#                    7, reading data: DataMem[         2] =>        256
#                    7, PC:        12
#                    9, reg_file[15] <=        256 (Write)
#                    8, PC:        12
#                    9, PC:        12
```



j 4:

```
VSIM 56> run
#                    0, reading data: InstrMem[          x] => xxxxxxxx(hex)
# 18446744073709551615, PC:          x
#                    0, reading data: InstrMem[          0] => 02308820(hex)
#                    0, PC:          0
#                    1, reg_file[16] =>          1 (Port 2)
#                    1, reg_file[17] =>          2 (Port 1)
#                    1, PC:          0
#                    1, wd:           x
#                    1, ADD
#
#                    2, PC:          0
#                    2, wd:           x
#                    2, ADD
#
#                    3, PC:          0
#                    3, wd:           x
#                    3, ADD
#
#                    4, reading data: InstrMem[          4] => 08000004(hex)
#                    5, reg_file[17] <=          3 (Write)
#                    4, PC:          4
#                    4, wd:           3
#                    4, ADD
#
#                    5, reading data: InstrMem[          8] => 02509025(hex)
#                    5, reg_file[ 0] =>          0 (Port 2)
#                    5, reg_file[ 0] =>          0 (Port 1)
#                    6, reg_file[17] <=          3 (Write)
#                    5, PC:          8
#                    5, J
#
#                    6, reading data: InstrMem[         16] => 02509022(hex)
#                    6, reg_file[16] =>          1 (Port 2)
#                    6, reg_file[18] =>          3 (Port 1)
#                    7, reg_file[17] <=          3 (Write)
#                    6, PC:         16
#                    6, wd:           3
#                    6, OR
#
#                    7, reading data: InstrMem[         20] => xxxxxxxx(hex)
#                    8, reg_file[17] <=          3 (Write)
#                    7, PC:         20
#                    7, wd:           3
#                    7, SUB
#
#                    8, reg_file[ x] =>          x (Port 2)
#                    8, reg_file[ x] =>          x (Port 1)
# control_single unimplemented opcode  x
#                    8, PC:         24
#                   10, reg_file[18] <=          3 (Write)
#                    9, PC:         24
```



jr $10:

```
#                    0, reading data: InstrMem[          x] => xxxxxxxx(hex)
# 18446744073709551615, PC:          x
#                    0, reading data: InstrMem[          0] => 01400008(hex)
#                    0, PC:          0
#                    1, reg_file[ 0] =>          0 (Port 2)
#                    1, reg_file[10] =>         16 (Port 1)
#                    1, PC:          0
#                    1, wd:           x
#                    1, JR
#
#                    2, PC:          0
#                    2, wd:           x
#                    2, JR
#
#                    3, PC:          0
#                    3, wd:           x
#                    3, JR
#
#                    4, reading data: InstrMem[         16] => 02509025(hex)
#                    4, PC:         16
#                    4, wd:          16
#                    4, JR
#
#                    5, reg_file[16] =>          1 (Port 2)
#                    5, reg_file[18] =>          3 (Port 1)
#                    5, PC:         16
#                    5, wd:          16
#                    5, OR
#
#                    6, PC:         16
#                    6, wd:          16
#                    6, OR
#
#                    7, PC:         16
#                    7, wd:          16
#                    7, OR
#
#                    8, reading data: InstrMem[         20] => 8e2f0000(hex)
#                    9, reg_file[18] <=          3 (Write)
#                    8, PC:         20
#                    8, wd:           3
#                    8, OR
#
#                    9, reading data: InstrMem[         24] => xxxxxxxx(hex)
#                    9, reg_file[15] =>         21 (Port 2)
#                    9, reg_file[17] =>          2 (Port 1)
#                   10, reg_file[18] <=          3 (Write)
#                    9, PC:         24
#                    9, LW
#
#                   10, reg_file[ x] =>          x (Port 2)
#                   10, reg_file[ x] =>          x (Port 1)
# control_single unimplemented opcode  x
```