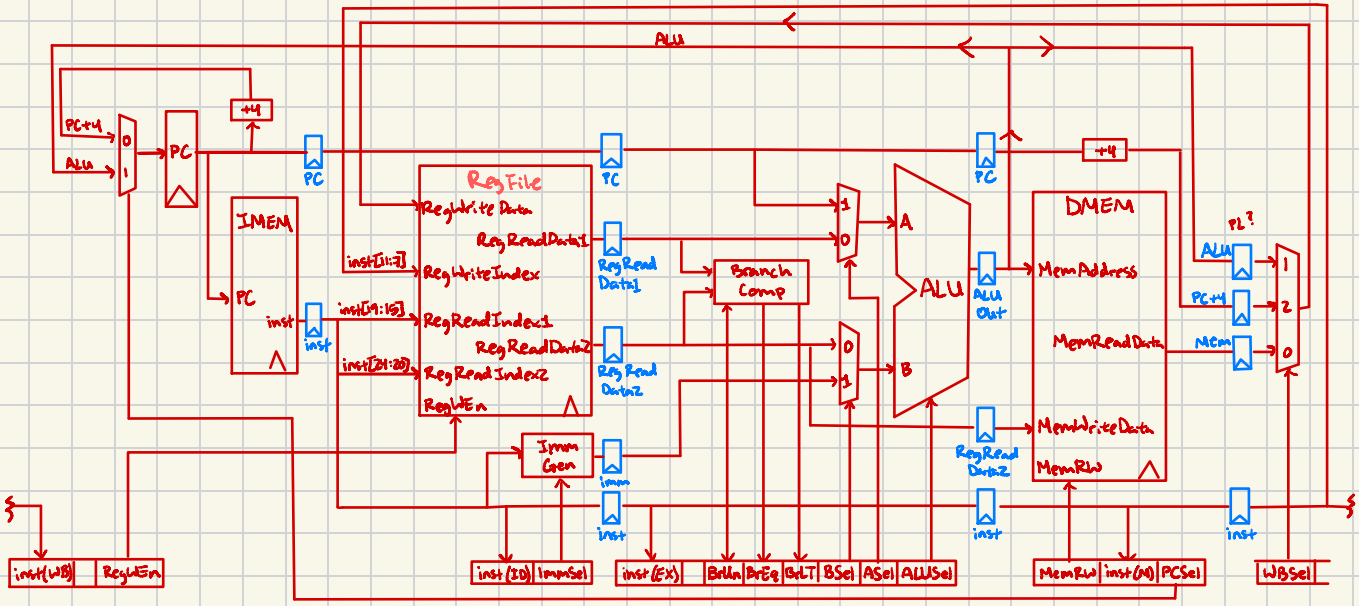

ALU
PC+4
ALU
0
1
PC
+4
PC
IMEM
PC
inst
inst
RegFile
RegWriteData
RegReadData1
RegWriteIndex
RegReadIndex1
RegReadData2
RegReadIndex2
RegWEn
inst[11:7]
inst[19:15]
inst[24:20]
PC
RegRead Data1
RegRead Data2
Imm Gen
imm
inst
Branch Comp
1
0
0
1
A
B
ALU
PC
ALU Out
RegRead Data2
inst
+4
DMEM
MemAddress
MemReadData
MemWriteData
MemRW
PL?
ALU
PC+4
Mem
1
2
0
inst
inst(WB)
RegWEn
inst(ID)
ImmSel
inst(EX)
BrUn
BrEq
BLT
BSel
ASel
ALUSel
MemRW
inst(M)
PCSel
inst
WBSel

Working Version of Pipelined Diagram:

3-stage Datapath: Memory relative in data path

ALU-stage: DMEM & IMEM are in sync

1) When Reads and Writes occur in register file and memory relative to the pipeline stages.

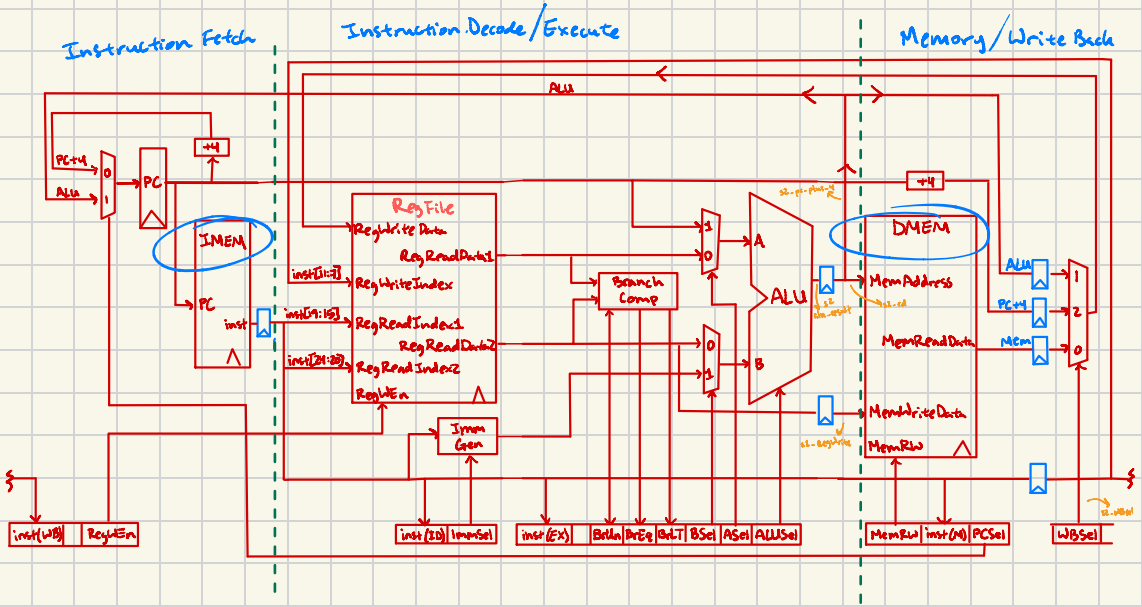
↳ Instruction Fetch

Read: the instruction memory (IMEM) is read using PC then fetch instruction passed to next stage

Write: written to PC reg to fetch next instruction in the cycle

ID/EX: Decode instruction, read registers, generate imm., and execute ALU operations

Mem/WB:
- Access DMEM (load/store) and write back to register file



Checkpoint 2:

Implement (control & status reg read/write) & (control & status reg read/write immediate)

Control and Status Register (CSR)

- some state that is stored independent of the register file and the memory
- 2^{12} possible CSR addresses

(tohost = 0x51E)

- register is monitored by the test harness, and simulation ends when a value is written to register
  - output ⇒ 1 success
  - > 1 gives clues to location of failure

csrw tohost, t2 → (short for csrrw x0, csr, rs1) csr = 0x51E

csrwi tohost, 1 → (short for csrrwi x0, csr, zimm) csr = 0x51E

csrw: writes a value from register 't2' and writes it into register tohost which has address 0x51E
used by the test harness to check if the processor is working correctly

31	20 19	15 14	12 11	7 6	0
csr	rs1	funct3	rd	opcode	
12	5	3	5	7	
source/dest	source	CSRRW	dest	SYSTEM	
source/dest	zimm[4:0]	CSRRWI	dest	SYSTEM	

Rest

Need Input Reset Signal → for testbench

↳ file SRC/const.vh

- reset should start PC address 0x2000, then begin executing instructions

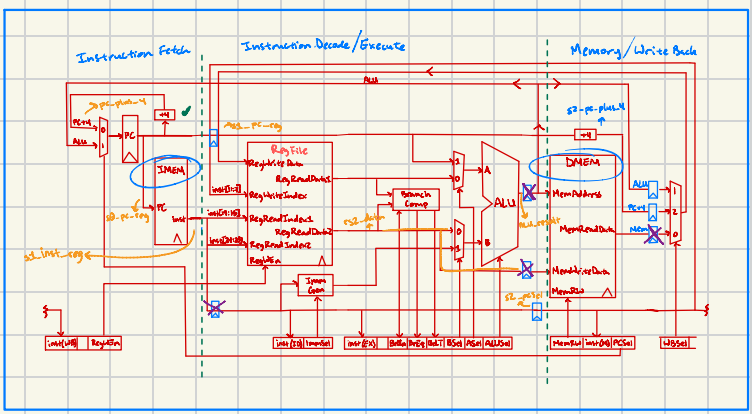
Misaligned Addresses

↳ ignore the misaligned bits (set them to zero) by using address bits 31:2 (Memory1S1.v)

Files needing edits:

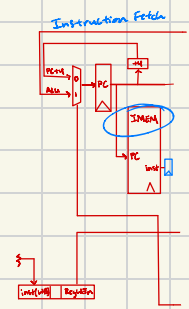
Risc1S1.v, risc_test_harness.v,

Memory1S1.v,



Components to Implement Via Stages

Stage 1) Instruction Fetch



implement Reg IMEM

input to IMEM is another register PC counter which is mux from ALU output and PC + 4

IMEM/DMEM are Synchronous

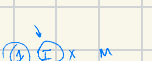
↳ reads/writes happen on the rising edge of the clk

ALU(A, B, ALUop, Out)

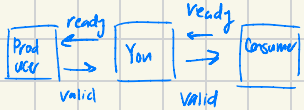
↳ calling alu module

ALU(.A( ), .B( ), .ALUop( ), .Out( ))

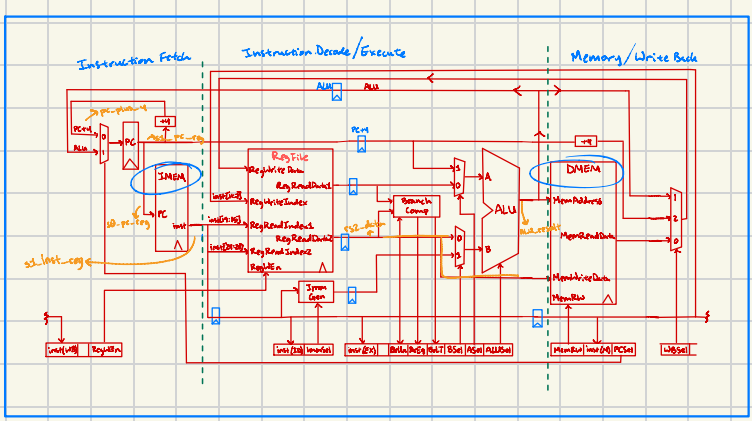
#pipeline rd_addr



each icache uses ready/valid interface



data transfer happens when both conditions are met (agreement protocol)

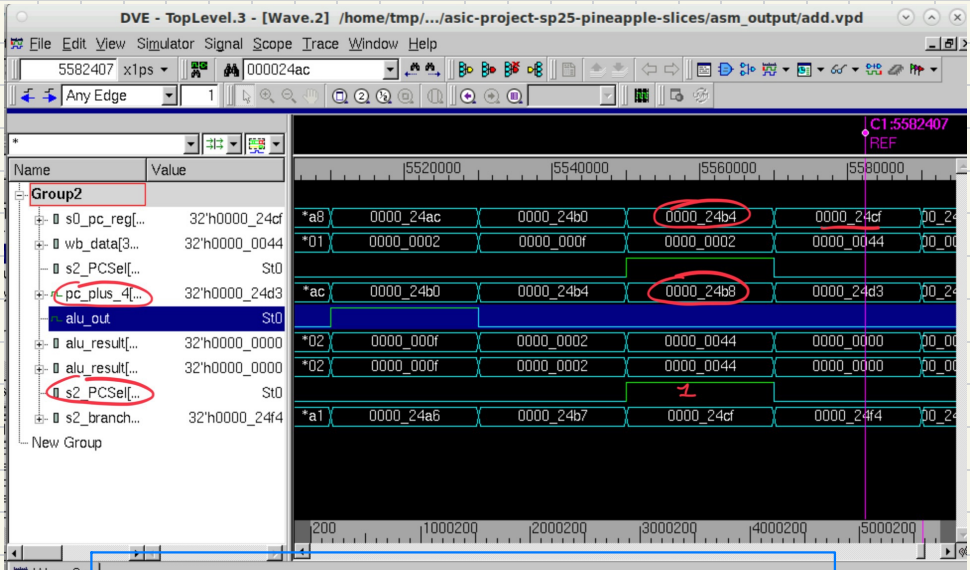


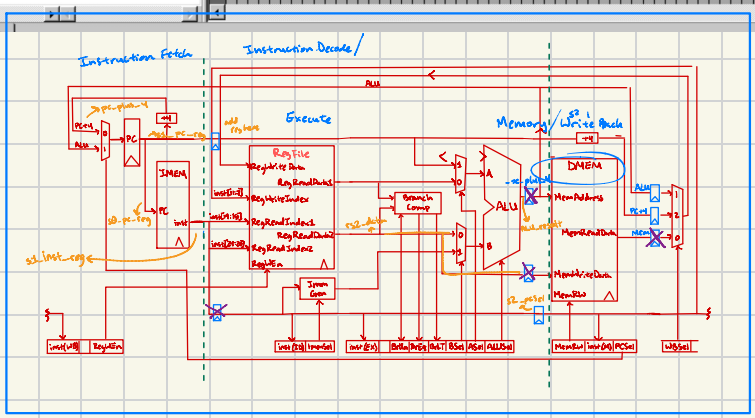
* not branching correctly @ <test 4>

bne x14, x7, 2294 <fail>

↳ error in

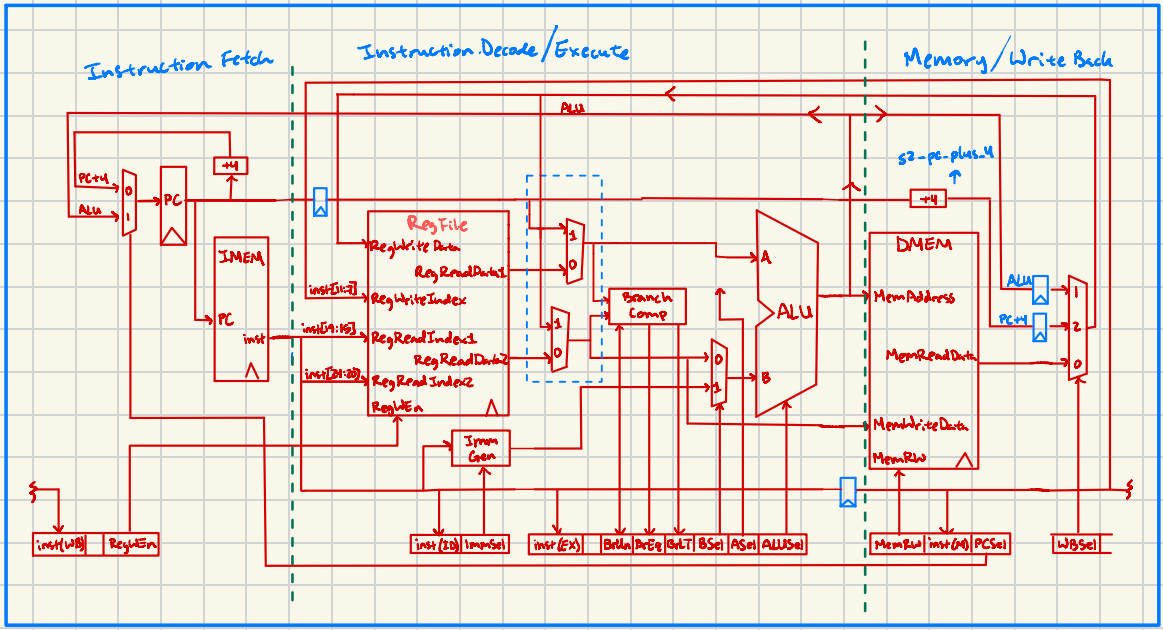
is_branch=0, branch_taken=0, branch_decision=0



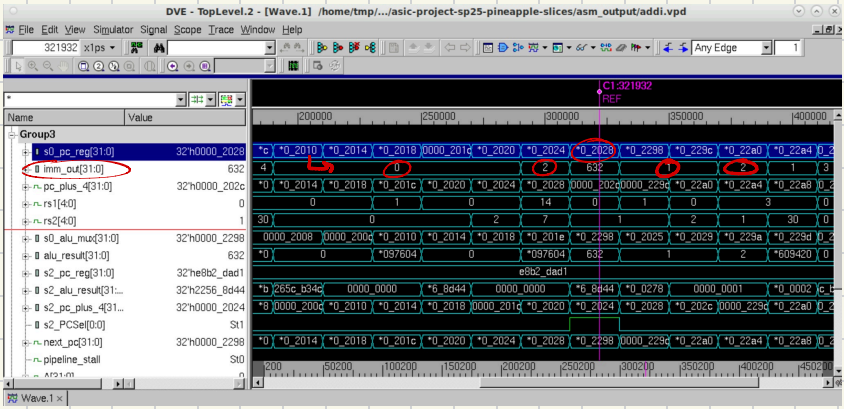


asic-project-sp25-pineapple-slices > tests > asm > add.dump

```
375     24a4:   02300193            li  x3,35
376     24a8:   04711263            bne x2,x7,24ec <fail>
377
378 000024ac <test_36>:
379     24ac:   02000093            li  x1,32
380     24b0:   00008133            add x2,x1,x0
381     24b4:   02000393            li  x7,32
382     24b8:   02400193            li  x3,36
383     24bc:   02711863            bne x2,x7,24ec <fail>
384
385 000024c0 <test_37>:
386     24c0:   000000b3            add x1,x0,x0
387     24c4:   00000393            li  x7,0
388     24c8:   02500193            li  x3,37
389     24cc:   02709063            bne x1,x7,24ec <fail>
390
391 000024d0 <test_38>:
392     24d0:   01000093            li  x1,16
393     24d4:   01e00113            li  x2,30
394     24d8:   00208033            add x0,x1,x2
395     24dc:   00000393            li  x7,0
396     24e0:   02600193            li  x3,38
```


Instruction Fetch
Instruction Decode/Execute
Memory/Write Back
ALU
PC+4
ALU
0
1
PC
+4
IMEM
PC
inst
RegFile
RegWrite Data
RegReadData1
inst[11:7]
RegWriteIndex
inst[19:15]
RegReadIndex1
RegReadData2
inst[24:20]
RegReadIndex2
RegWEn
Imm Gen
Branch Comp
A
B
ALU
DMEM
MemAddress
MemReadData
MemWriteData
MemRW
s2 = pc_plus_4
+4
ALU
PC+4
1
2
0
inst(WB)
RegWEn
inst(ID)
ImmSel
inst(EX)
BrUn
BrEq
BrLT
BSel
ASel
ALUSel
MemRW
inst(M)
PCSel
WBSel

issue w/ addi test




```
0000200c <test_2>:
    200c:   00000093            li  x1,0
    2010:   00008713            mv  x14,x1
    2014:   00000393            li  x7,0
    2018:   00200193            li  x3,2
    201c:   26771c63            bne x14,x7,2294 <fail>

00002020 <test_3>:
    2020:   00100093            li  x1,1
    2024:   00108713            addi    x14,x1,1
    2028:   00200393            li  x7,2
    202c:   00300193            li  x3,3
    2030:   26771263            bne x14,x7,2294 <fail>
```

load immediate 0

load immediate 2

at 2020 li of 1 does not happen