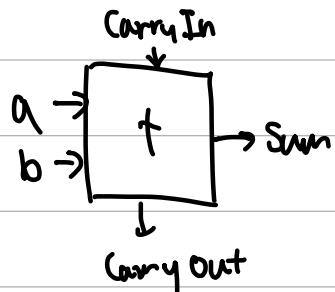


CA04-1

- register write : write signal이 1인 경우, edge trigger 되었을 때만!

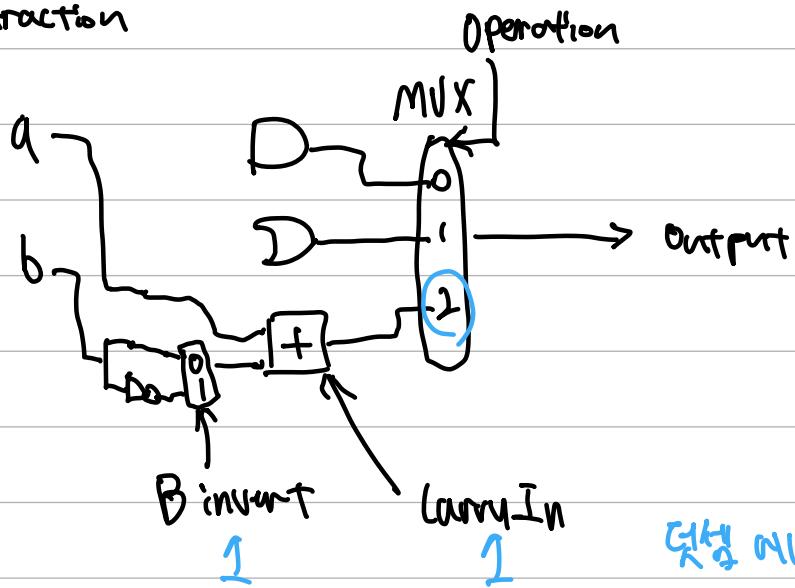
- 1bit ALU adder



$$\text{Sum} = a \oplus b \oplus \text{CarryIn}$$

$$\text{CarryOut} = a \cdot b + a \cdot \text{CarryIn} + b \cdot \text{CarryIn}$$

- Subtraction

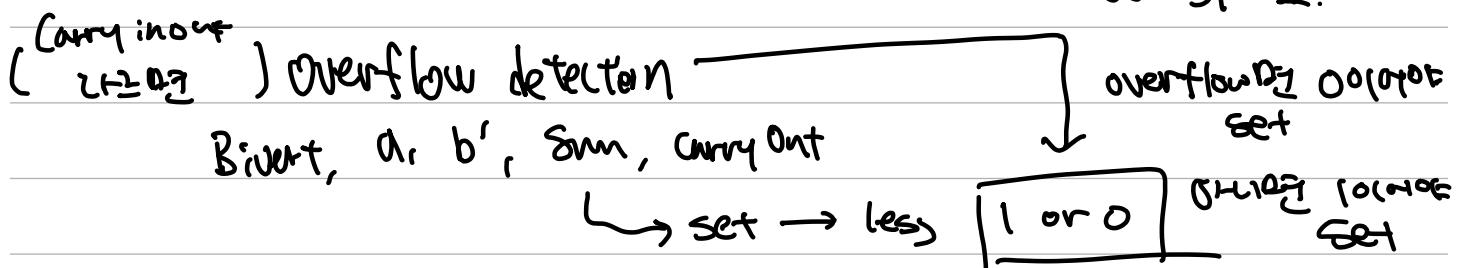


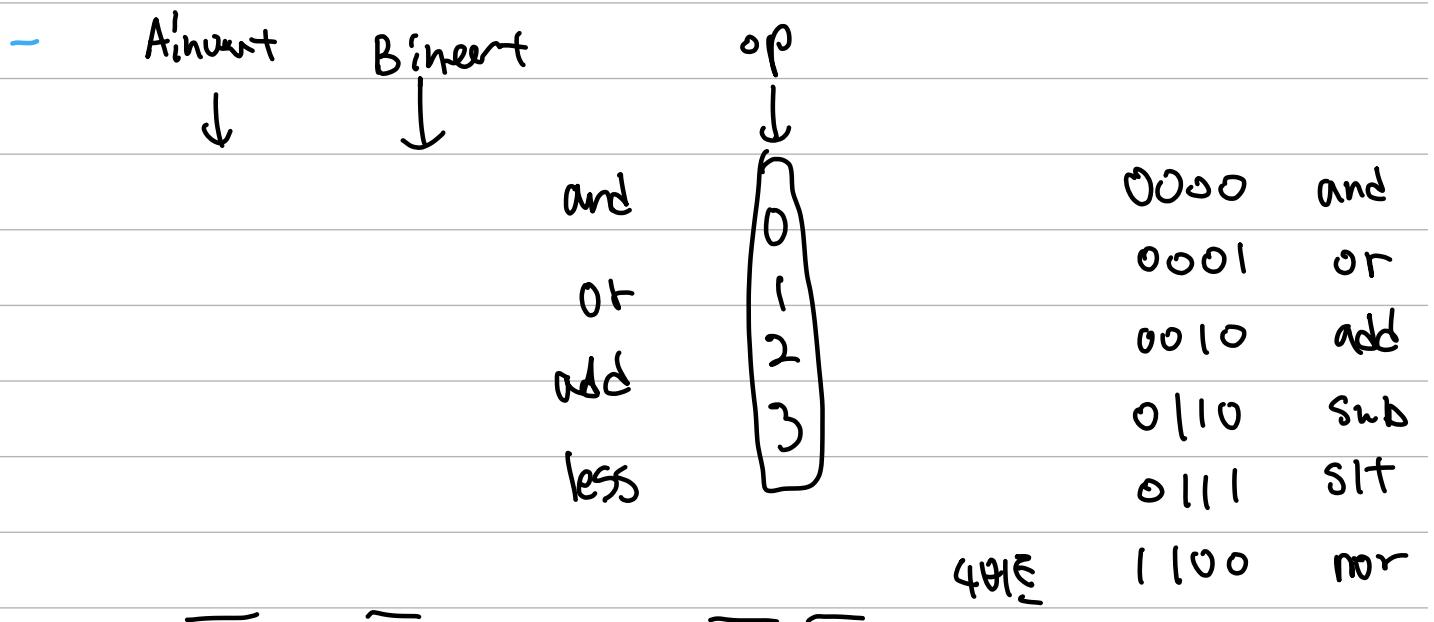
덧셈에 이어가 선택하면

B가 2의 보수가 됨.

- slt, beq : $a - b < 0$, $= 0$ 으로 판단.

↓ 부정비트가 1 overflow 0 는 부정비트
overflow 1.





- SW: 메모리에 파일 \leftarrow Reg write 0
메모리 Read \rightarrow Mem write 2

(W는 Memory 주소이고 read \rightarrow 메모리 write data3)

문자 (32 bit로 sign extend) + rsel 주소 \rightarrow rsel 주소
+ rsel 주소 \leftarrow rsel (10bit)

SW
IW rt immediate(rs)

- branch

설정 후 zero 풀이 10비트 (경과) 000...0이랑 같음



PC+4 풀, immediate 풀 (32bit sign extend) << shift left 2

- jump

Op 주소 $\xrightarrow{26}$ shift left 2 $\xrightarrow{28}$ 쌍바이트 PC(MM 4816)에 위치
6bit 26bit $\xrightarrow{\text{결과 } 32\text{bit}}$ 1이기

— **ମହାକୁଳ**

R	6 (op)	5 (rs)	5 (rt)	5 (rd)	5 (shamt)	6 (funct)
(I)	lw \$av	6 (op)	5 (rs)	5 (rt)	16 (address)	
	B	6 (op)	5 (rs)	5 (rt)	16 (address)	

↓ ↓ ↓

\$000001 lw \$av sign extended.

J 6 (op) 26 (address)

1

old PC의 Shift + 26 번호 + 00
(Shift left)

_ (control) Signal (inputs to opcode)

Req Bst 쓰기별 전자화상 rt or rd

Jump

Branch zero π , Branch 1 π , Branch 2 π

Mem Read .Aug 21 2019

Mem to Reg ALU 74181 or 65200 값 212(851 1113

ALU op ALU 컨트롤 2bit (R은 0, 1swz 00, beqz 01)

Mem Write 메모리 쓰기 기능

ALU Src

Req write 2개 이상의 쓰기 내용

- ALU control

우리에서 본 ALU 4bit를

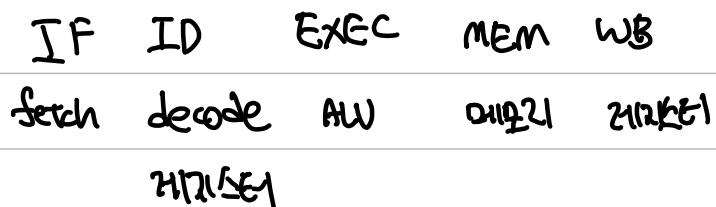
정령의 function 필드 6bit로 고려 - ALUOp 20로 3개로.

(A04-2)

- pipelining 단계 간의 latency를 줄이자 않음.

전체 간의 throughput (처리량)을 늘리자 것.

- MIPS



- Single

cycle

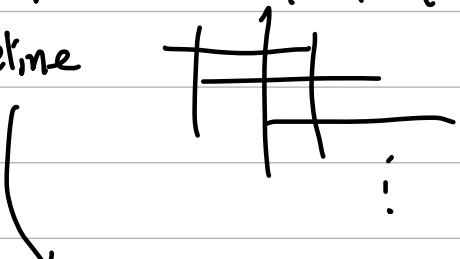
단 면경에 당 한 사이클

- multiple

EXEC

한 사이클 당 한 사이클

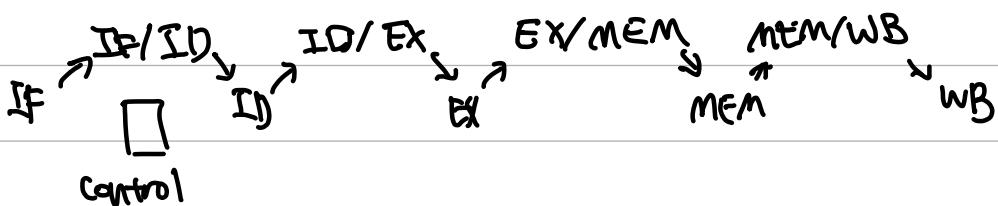
- pipeline

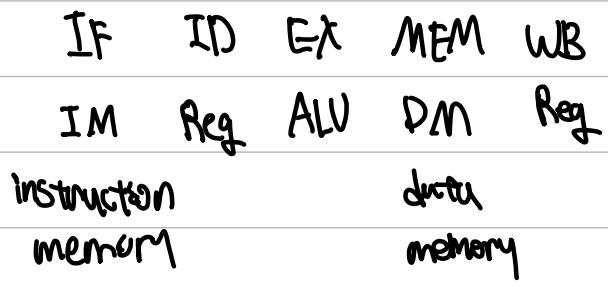


정령의 개수 + (스텝이지수 - 1)

이상적으로 CPI = 1 (drain 때마다 cycle 추가)

- Pipeline register





- 부트 쿠

$$IF/ID \quad PC\ 32 + 명령어\ 32 = 64$$

$$ID/EX \quad 제어신호\ 9 + 32 \times 4 + 10 = 147$$

$$EX/MEM \quad 5 + 1 + 32 \times 3 + 5 = 107$$

$$MEM/WB \quad 2 + 32 \times 2 + 5 = 71$$

(A04-3)

- pipeline hazards

- structural : 같은 자원

- data : 주소오류가 있는 경우

- control : 불가 평가오기 전에 다음 명령어

- internal bypass

no register forwarding

yes ..

IM Reg ALU DM Reg

nop

nop

nop

IM Reg ALU DM Reg

IM Reg ALU DM Reg

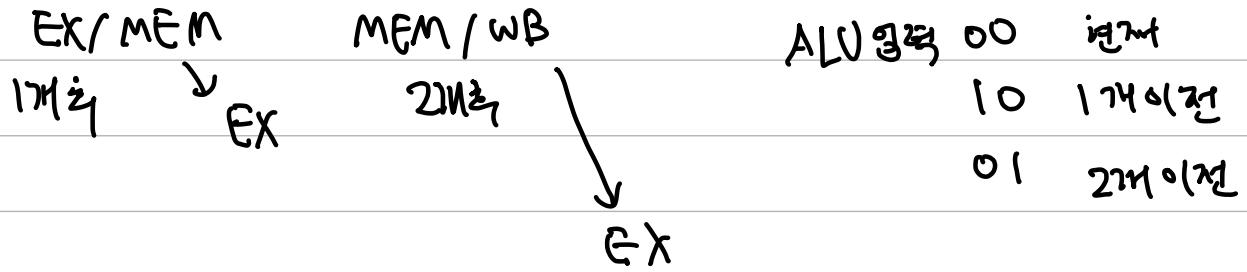
nop

nop

IM Reg ALU DM Reg



-forwarding



ALU 결과 00 현재
10 174 이전
01 271 이전

-data hazard when ...

$$\begin{array}{l} \text{EX/MEM} \\ \left[\begin{array}{ll} \text{EX/MEM rd} = \text{ID/EX 의 rs} & \\ \text{EX/MEM rd} = & \text{rt} \end{array} \right] \text{RAW} \\ \text{MEM/WB} \\ \left[\begin{array}{ll} \text{MEM/WB rd} : & \text{rs} \\ \text{MEM/WB rd} : & \text{rt} \end{array} \right] \begin{array}{l} \text{: read after} \\ \text{write} \\ \text{MI 가능하지} \end{array} \end{array}$$

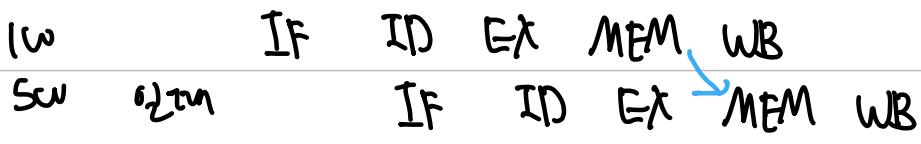
if EX/MEM 혹은 MEM/WB 의 RegWrite가 1이고,
and EX/MEM 혹은 MEM/WB 가 \$zero가 아니면 (\$zero != 0) 아님(여야)이고

(MEM/WB일땐 EX/MEM rd가 아님(=rt))

and EX/MEM 혹은 MEM/WB 와 rd = rs 혹은 rt 일경우

rs와 rt forward 가능 (EX/MEM이 10일 때
MEM/WB는 01)

-Pipeline - Hazard



포워드



- (W 이후에) 의존적인 rtype 일 때 hazard detect.

ID/EX의 Memread가 102 (load 명령: P)

ID/EX의 ~~Memread~~가 IF/ID의 rs1 r2와 같다면

P(Gr-IF/ID 유지) ← Stall,

CH04-4

- Branch hazard

Branch 후 3개 단계이 걸림 (4사이클 필요) ALU 후 MEM 단계에서 stall

- 7번 (1개 단계이 (2사이클 필요) ID/ALU 단계)

- 예측 성공 No 단계이 (1사이클)

단계 1개 단계이 (2사이클)

- Jump 2M⁰ 를 필요 (1개 단계이) ID 소스(이전)에서 짜증