

## Alu Ctrl

### ALU Control

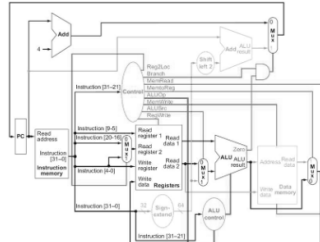
- ALU used for
  - Load/Store: F = add
  - CBZ: F = pass input b
  - R-type: F depends on opcode

ALU control	Function
0000	AND
0001	OR
0010	add
0110	subtract
0111	Pass input b
1100	NOR

Chapter 4 — The Processor — 22

## Control instructions

### R-Type Instruction



Chapter 4 — The Processor — 38

## N-stage pipeline

P2 is a 7-stage pipelined processor with the clock cycle time of 200ps. If you are given a program with 1000 instructions without any pipeline hazards, calculate the execution times on P1 and P2

$$P_2 = (7 * 200ps) + (200ps * 999 \text{ insts})$$

## Hazards

Structure A required resource is busy

Hazards

Data Need to wait for previous instruction to complete its data read/write

Control Deciding on control action

Hazard depends on previous instruction

## Pipeline stages

Five stages:

1. IF: Instruction fetch from memory
2. ID: Instruction decode & register read
3. EX: Execute operation or calculate address
4. MEM: Access memory operand
5. WB: Write result back to register

## ALU Ctrl

### ALU Control

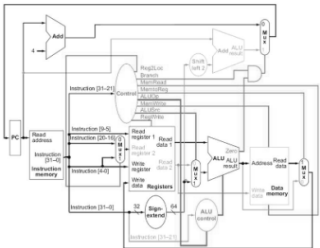
- Assume 2-bit ALUOp derived from opcode
  - Combinational logic derives ALU control

Instruction	ALUOp	Instruction operation	Opcode field	Desired ALU action	ALU control input
LDUR	00	load register	XXXXXXXXXX	add	0010
STUR	00	store register	XXXXXXXXXX	add	0010
CBZ	01	compare and branch on zero	XXXXXXXXXX	pass input b	0111
R type	10	ADD	10001011000	add	0010
R type	10	SUB	11001011000	subtract	0110
R type	10	AND	10001010000	AND	0000
R type	10	ORR	10101010000	OR	0001

Chapter 4 — The Processor — 23

## control instructions

### Load Instruction



Chapter 4 — The Processor — 27

## upgraded adder table

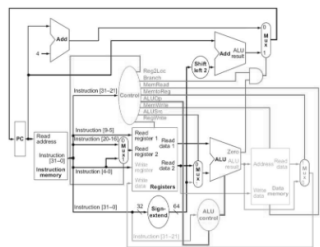
### Example (1111<sub>2</sub> X 1001<sub>2</sub>)

Mcand	Prod	Step
1111	0000 1001	Load Miller in lower half of product
1111	0000 1001	If Prod(0) == 1, add Mcand to upper half of Prod
1111	1111 1001	Then shift Prod
1111	0111 1100	Prod(0) = 0. Just shift
1111	0011 1110	Prod(0) = 0. Just shift
1111	0001 1111	Prod(0) = 1. Add Mcand to upper half of Prod
1111	1 0000 1111	Shift
1111	1000 0111	Final product

Chapter 3 — Arithmetic for Computers — 9

## control instructions

### CBZ Instruction

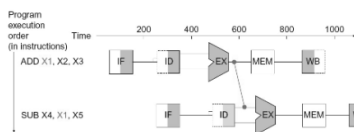


Chapter 4 — The Processor — 28

## Forwarding

### Forwarding (aka Bypassing)

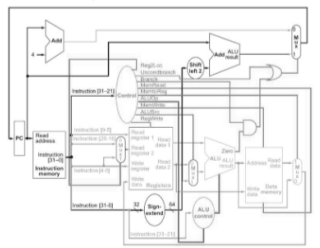
- Use result when it is computed
  - Don't wait for it to be stored in a register
  - Requires extra connections in the datapath
  - EXE-forwarding shown below – no stall needed



Chapter 4 — The Processor — 41

## control instructions

### Datapath With B Added

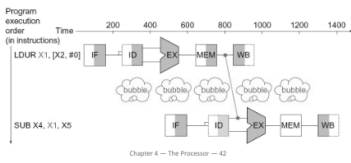


Chapter 4 — The Processor — 33

## Forwarding(Men)

## Cashe Preformance

- Can't always avoid stalls by forwarding
  - If value not computed when needed
  - Can't forward backward in time!
  - MEM-forwarding shown below – 1 stall CC needed



- Components of CPU time
  - Program execution cycles
    - Includes cache hit time
  - Memory stall cycles
    - Mainly from cache misses
- With simplifying assumptions:
  - Memory stall cycles
 
$$= \frac{\text{Memory accesses}}{\text{Program}} \times \text{Miss rate} \times \text{Miss penalty}$$

$$= \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Misses}}{\text{Instruction}} \times \text{Miss penalty}$$

Chapter 5 – Large and Fast: Exploring  
Memory Hierarchy – 39



By datonecklund

[cheatography.com/datonecklund/](https://cheatography.com/datonecklund/)

Not published yet.

Last updated 11th December, 2024.

Page 1 of 2.

Sponsored by [ApolloPad.com](https://apollopad.com)

Everyone has a novel in them. Finish Yours!

<https://apollopad.com>