Levels in Processor Design

- Circuit design
  - Keywords: transistors, wires etc.Results in gates, flip-flops etc.

- Logical design
  - Putting gates (AND, NAND, …) and flip-flops together to build basic blocks such as registers, ALU's etc (cf. CSE 370)

- **Register transfer**
  - Describes execution of instructions by showing data flow between the basic blocks

- **Processor description** (the ISA)

- *System description*
  - Includes memory hierarchy, I/O, multiprocessing etc

CSE378 Single cycle implementation.

Register transfer level

- Two types of components (cf. CSE 370)
  - *Combinational* : the output is a function of the input (e.g., adder)
  - *Sequential*: state is remembered (e.g., register)

CSE378 Single cycle
implementation.

Synchronous design

- Use of a periodic clock
  - *edge-triggered* clocking determines when signals can be read and when the output of circuits is stable
  - Values in storage elements can be updated only at clock edges
  - Clock tells when events can occur, e.g., when signals sent by control unit are obeyed in the ALU

Note: the same storage element can be read/written in the same cycle

| Stor. Elem 1 | → Comb.logic → | Stor. Elem 2 |

Clock cycle

CSE378 Single cycle implementation.

Stor. Elem 1

Comb.logic

Stor. Elem 2

Write signal

Write signal
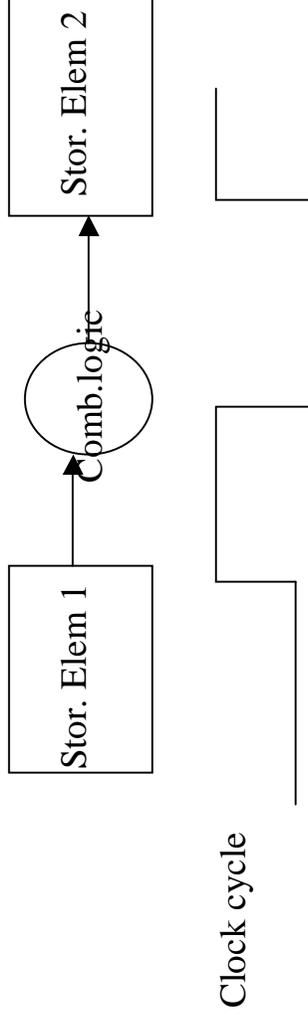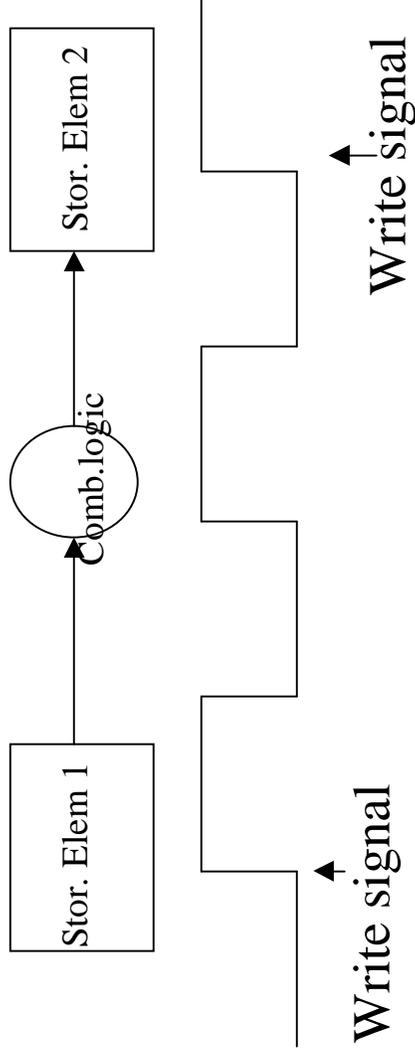
Logic may need several cycles to propagate values

CSE378 Single cycle
implementation.

10/18/99

Processor design: data path and control unit

CPU

control

ALU — Combinational

PC — Sequential
state — Sequential

Registers

Data path

Memory
hierarchy

Memory
bus

10/18/99

CSE378 Single cycle
implementation.

5
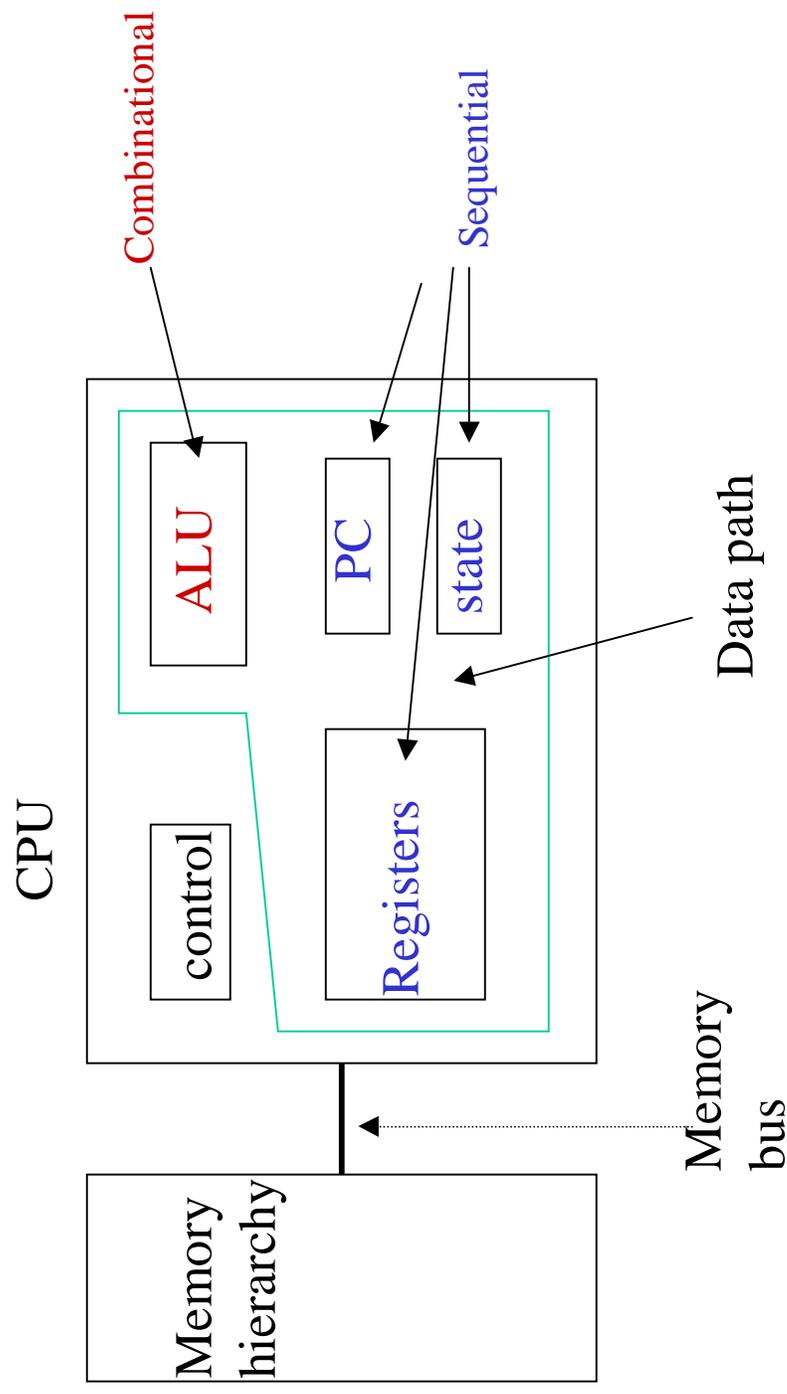
Processor design

- **Data path**
  - How does data flows between various basic blocks
  - What operations can be performed when data flows
  - What can be done in one clock cycle

- **Control unit**
  - Sends signals to data path elements
  - Tells what data to move, where to move it, what operations are to be performed

- **Memory hierarchy**
  - Holds program and data

CSE378 Single cycle implementation.

Data path basic building blocks. Storage elements

- Basic building block (at the RT level) is a register

- In our mini-MIPS implementation registers will be 32-bits

- A register can be read or written

Write enable signal

Input bus

Register

Output bus

CSE378 Single cycle
implementation.
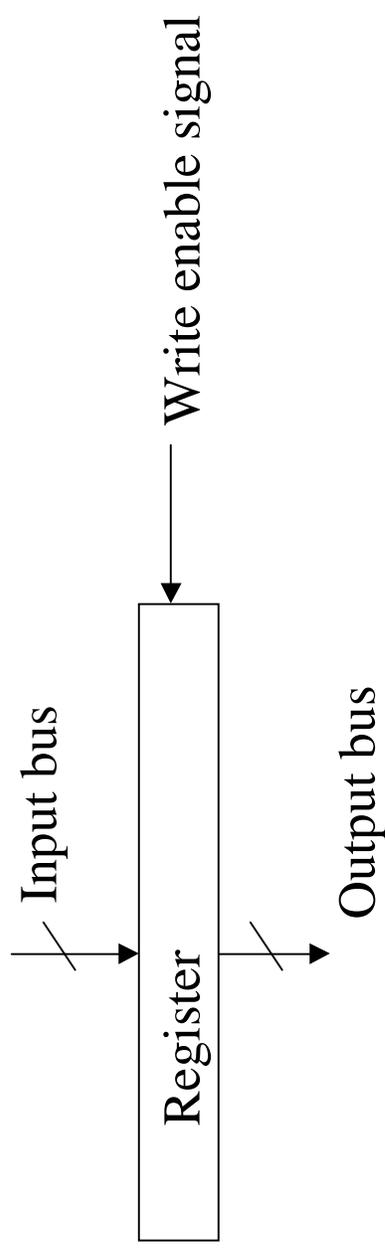
Register file

- Array of registers (32 for the integer registers in MIPS)

- ISA tells us that we should be able to:

  - read 2 registers, write one register in a given instruction (at this point we want one instruction per cycle)

 - Register file needs to know which registers to read/write

Write register number

Write enable

Write data input bus

Read register number bus 0

Read register number bus 1

Read data output bus 0

Read data output bus 1

Register file

CSE378 Single cycle implementation.
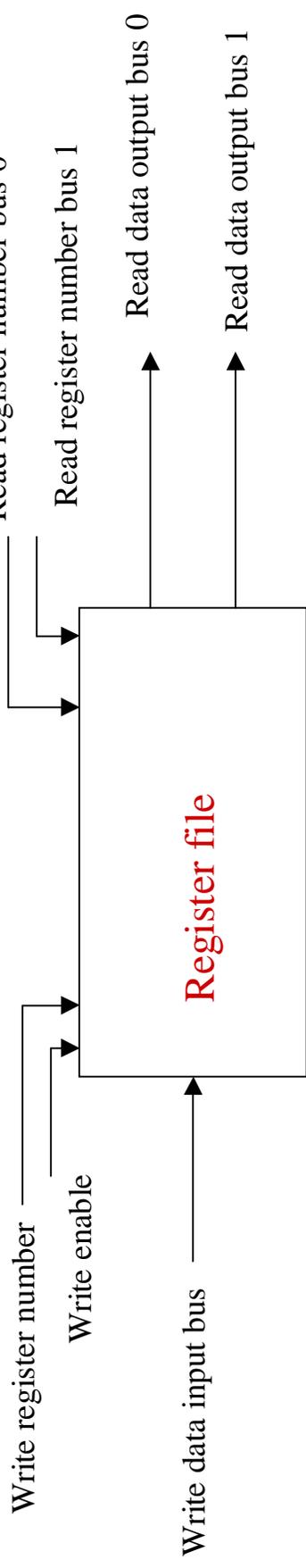
Memory

- Conceptually, like register file but much larger

- Can only read one location or write to one location per cycle

```
          Read memory address
          Read control signal                    Read data bus
                   │                                   │
                   │                                   ▲
                   ▼                                   │
          ┌────────────────────────────────────────────┐
          │                                            │
   ──────▶│                                            │
          │                  Memory                    │
   ──────▶│                                            │
   Write  │                                            │
   memory └────────────────────────────────────────────┘
   address            ▲
   Write enable       │
                 Write data bus
```
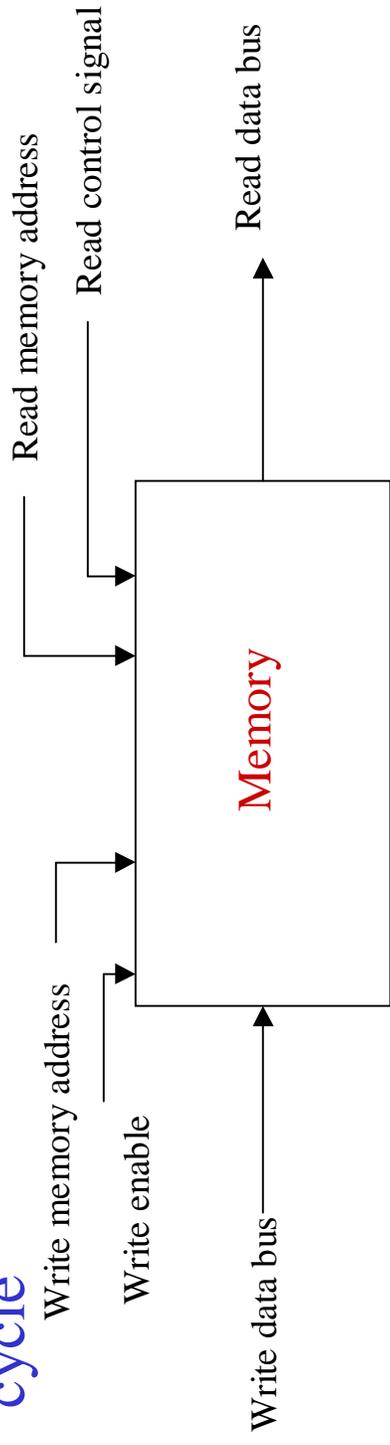
Combinational elements

Multiplexor (Mux): selects the value of one of its inputs to be routed to the output

Input busses

Mux

Select control signal

Output bus
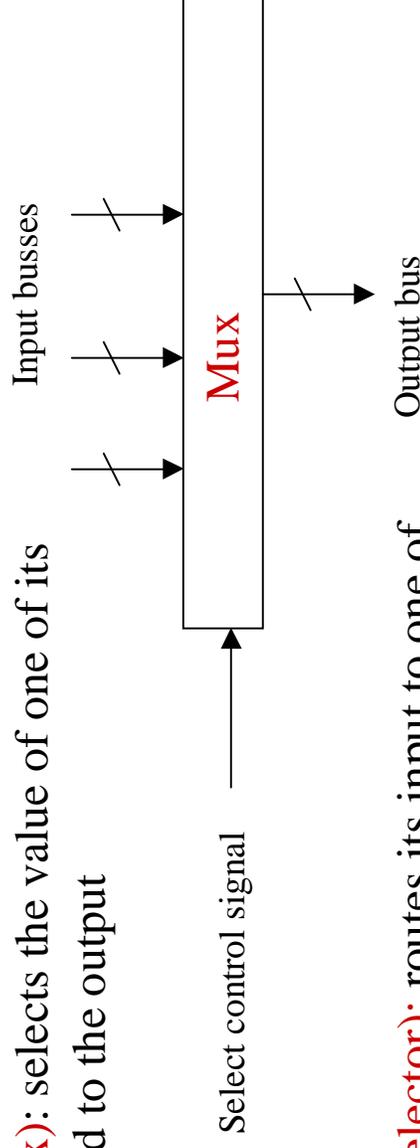
Demultiplexor (selector): routes its input to one of its outputs

Output busses

Sel

Input bus

Select control signal

CSE378 Single cycle implementation.

10
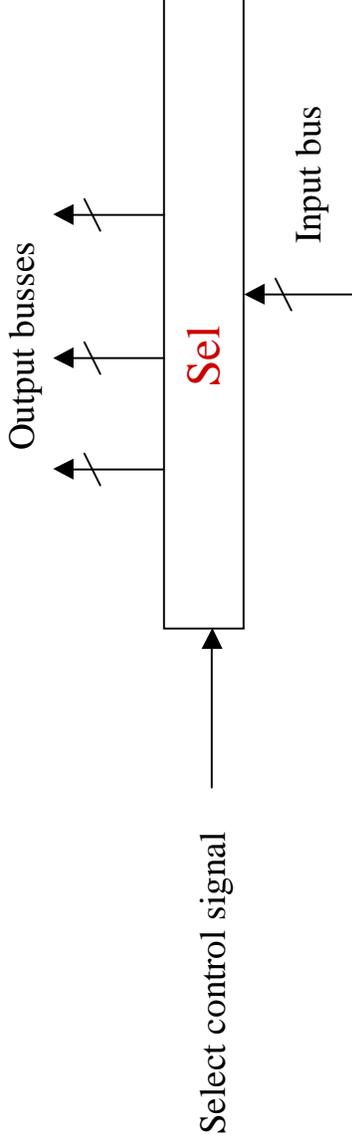
Arithmetic and Logic Unit (ALU - combinational)

- Computes (arithmetic or logical operation) output from its two inputs

Input bus 0

Input bus 1

ALU

ALU control
(opcode/function)

Zero result bit

Output bus

CSE378 Single cycle
implementation.
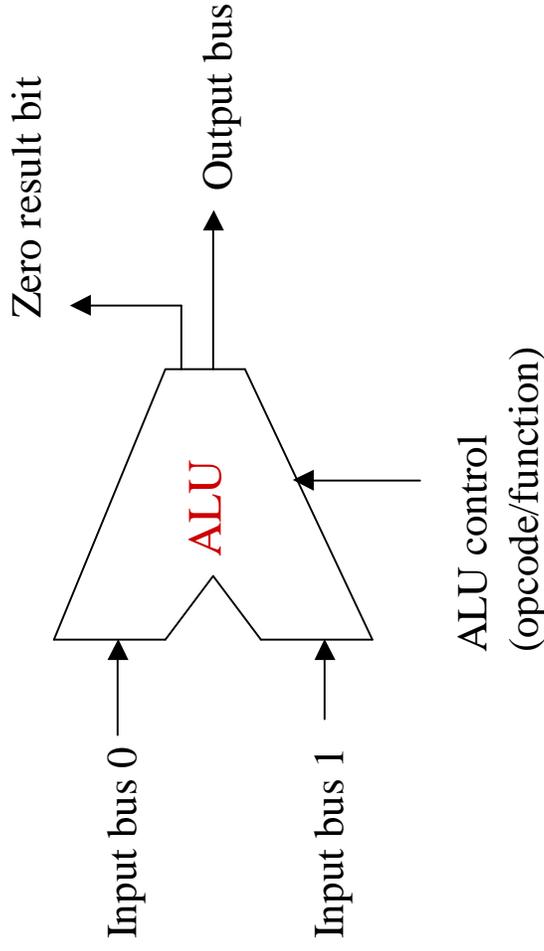
Putting basic blocks together (skeleton of data path for arith/logical operations)

CSE378 Single cycle implementation.
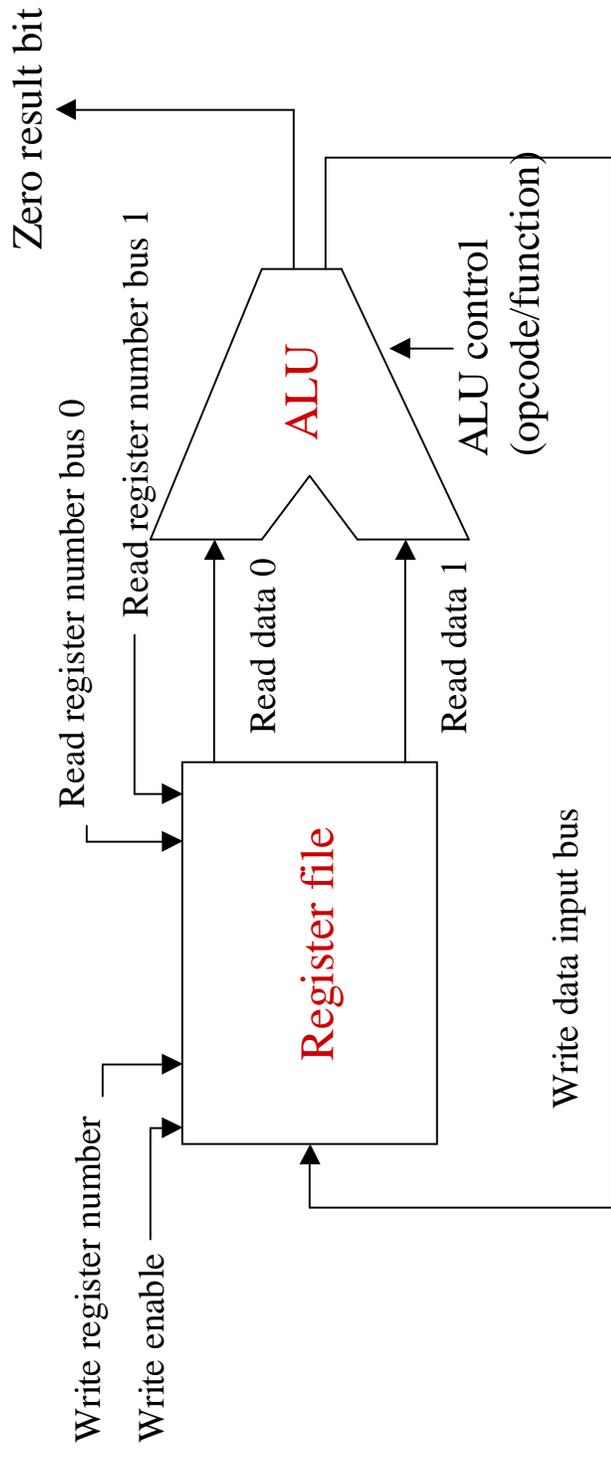
Introducing instruction fetch

Zero result bit

ALU

Read data 0

Read data 1

ALU control
(opcode/function)

Read Reg #0

Read Reg #1

Write Reg #

Reg. File

Write data

Instr. memory

Instruction address

PC

CSE378 Single cycle
implementation.

10/18/99

13
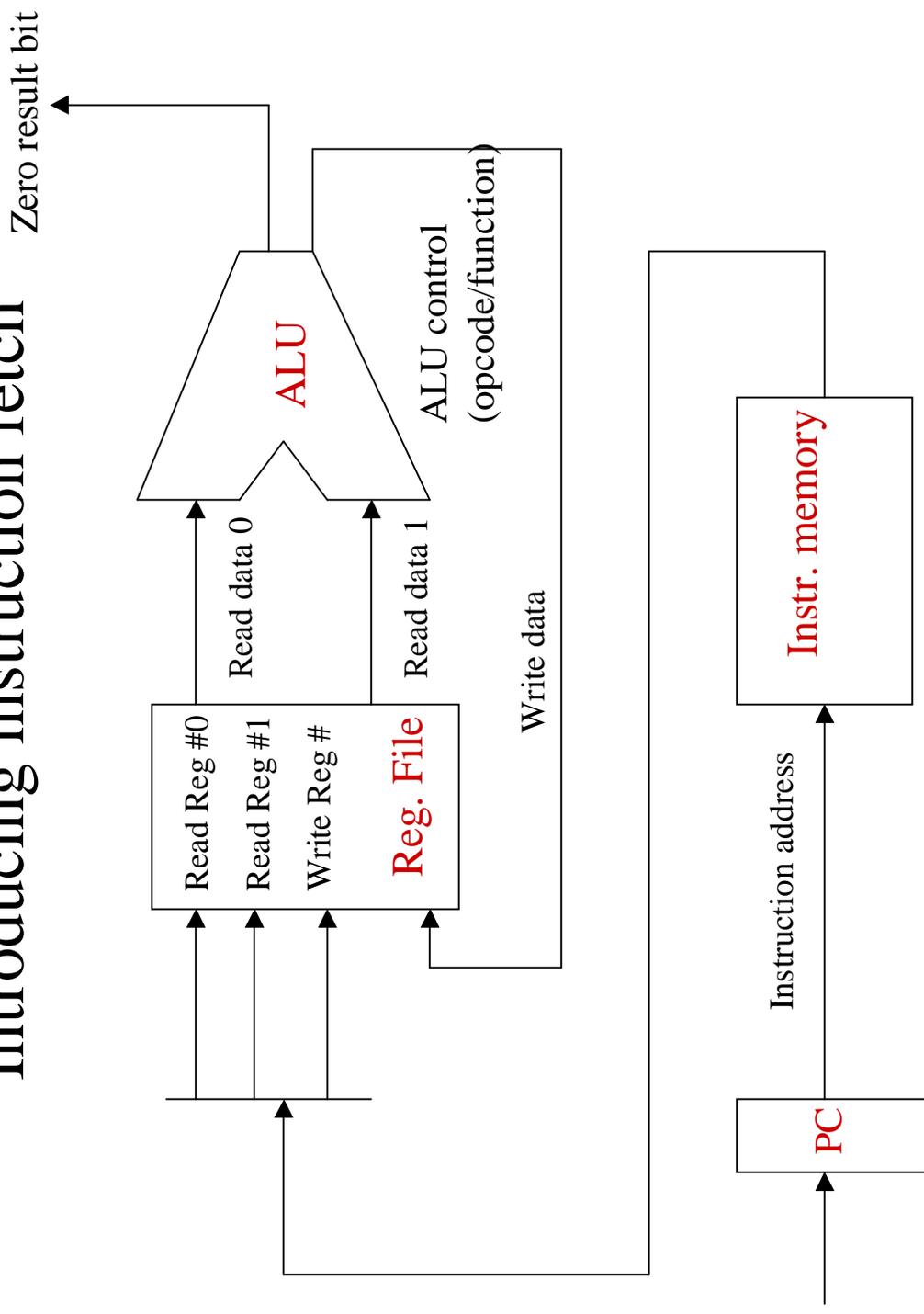
PC has to be incremented (assume no branch)

ALU

4

PC

Instruction address

Instr. memory

Instruction

CSE378 Single cycle
implementation.

Load-Store instructions

CSE378 Single cycle implementation.

10/18/99

15
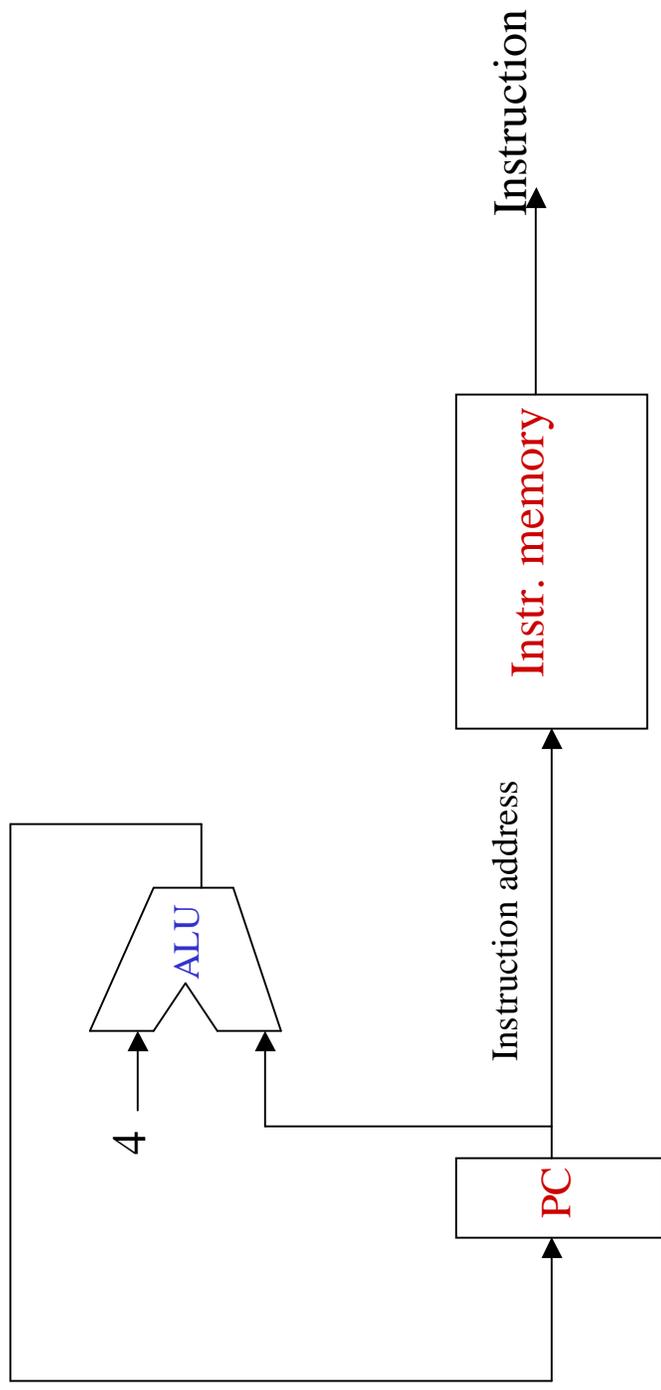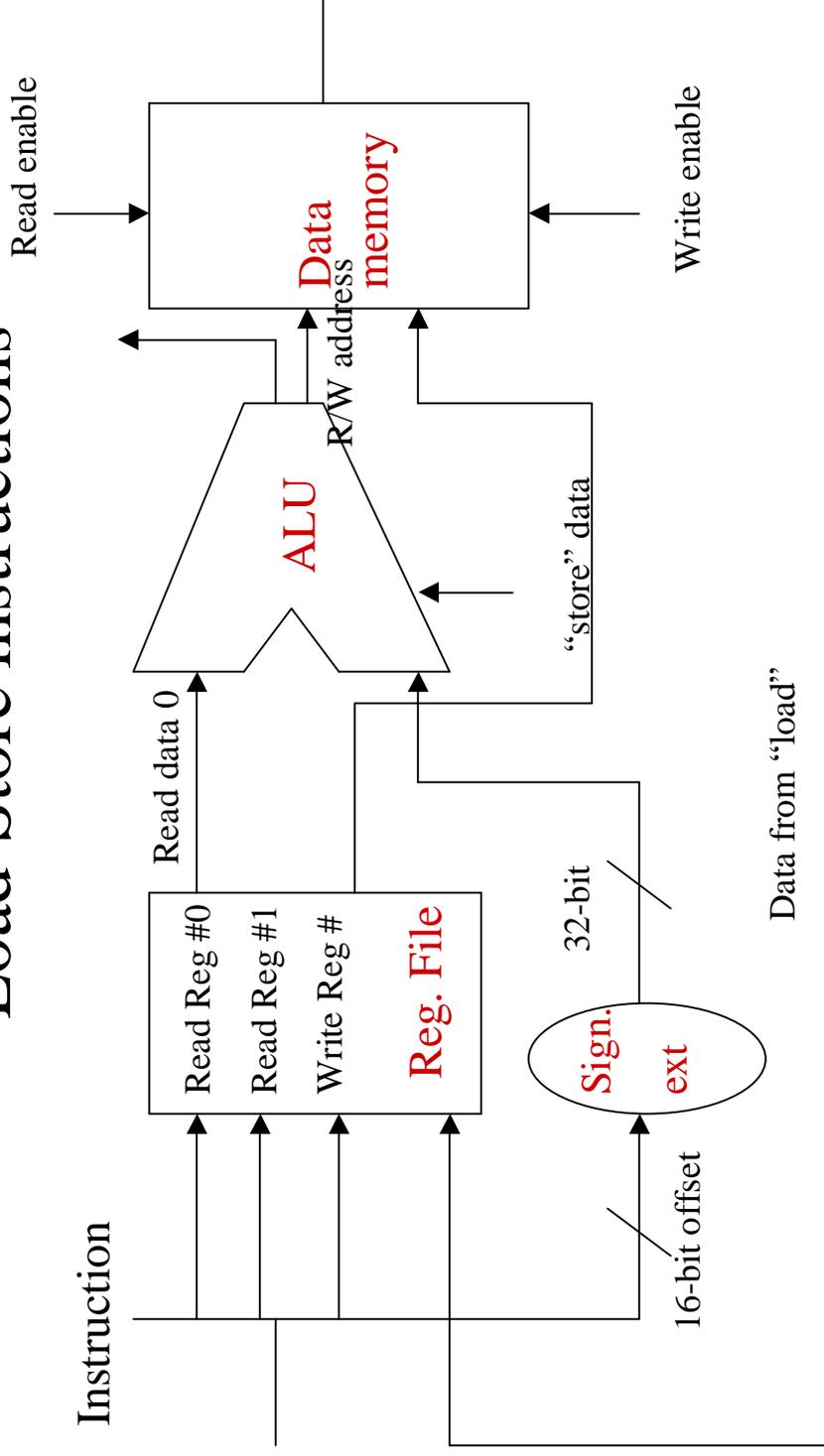
Data path for straight code(reg-reg,imm,load/store)

Instruction

Read enable

Read data 0

Read data 1

Read Reg #0

Read Reg #1

Write Reg #

Reg. File

ALU

Data memory

R/W address

Write enable

"store" data

32-bit

Sign. ext

16-bit offset

Data for result register

Mux

10/18/99

CSE378 Single cycle implementation.

Branch data path

ALU

Sftl 2

32-bit

Sign. ext

16-bit

Instruction

Inst. memory

4

PC

ALU

CSE378 Single cycle implementation.

10/18/99

17