COL216
Computer Architecture

Processor design -
Introduction
22nd February, 2016

Overview

- Given an ISA, how to design a Micro architecture
- CAD tools are required for physical implementation
- There are numerous possibilities
- Different combinations of performance – cost – power consumption are possible
- CAD tools help in analyzing the trade offs between these parameters

Where do we start?

- Subset of instructions for implementation
- Design overview
- Division into data path and control
- Building blocks - combinational and sequential
- Clock and timings

ARM Instruction Subset

Branch: <b | bl> {*cond*}

Add-sub: <add|sub|rsb|adc|sbc|rsc> {*cond*} {s}

Logical: <and | orr | eor | bic> {*cond*} {s}

Test: <cmp | cmn | teq | tst> {*cond*} {s}

Move: <mov | mvn> {*cond*} {s}

Multiply: <mul | mla> {*cond*} {s}

Load/store: <ldr | str> {*cond*} {b | h | sb | sh }

Some assumptions

- The program consists of valid instructions
  - no checking is performed by hardware
- Some features are excluded to simplify the design
  - limit the number of read and write accesses to register file
- pc register is used in a restricted way
  - pc is not arbitrarily modified
  - no writes into program area

Operand 2 in DP instructions

- 12 bit operand2 in DP instructions

4 2 1 4 1 4 4 12

  - 8 bit unsigned number,
    4 bit rotate spec, or
  - 4 bit register number,
    8 bit shift specification
    shift type: LSL, LSR, ASR, ROR
    shift amount: 5 bit constant ~~or 4 bit register no~~.

Immediate in DT instructions

- 12 bit offset field in DT instructions

– 12 bit unsigned constant

or

– 4 bit register number, 8 bit shift specification (same format as DP instructions)

Opcode field in DT instructions

- 6 opcode bits specify
  - I (immediate): constant ~~or register with shift~~
  - P (pre/post) pre ~~or post indexing~~
  - U (up/down) whether to add or subtract offset
  - B (byte) byte or word transfer
  - W (write back) ~~write back or~~ no write back address into base register (Rn)
  - L (load/store) load from memory or store into memory

Extending the design

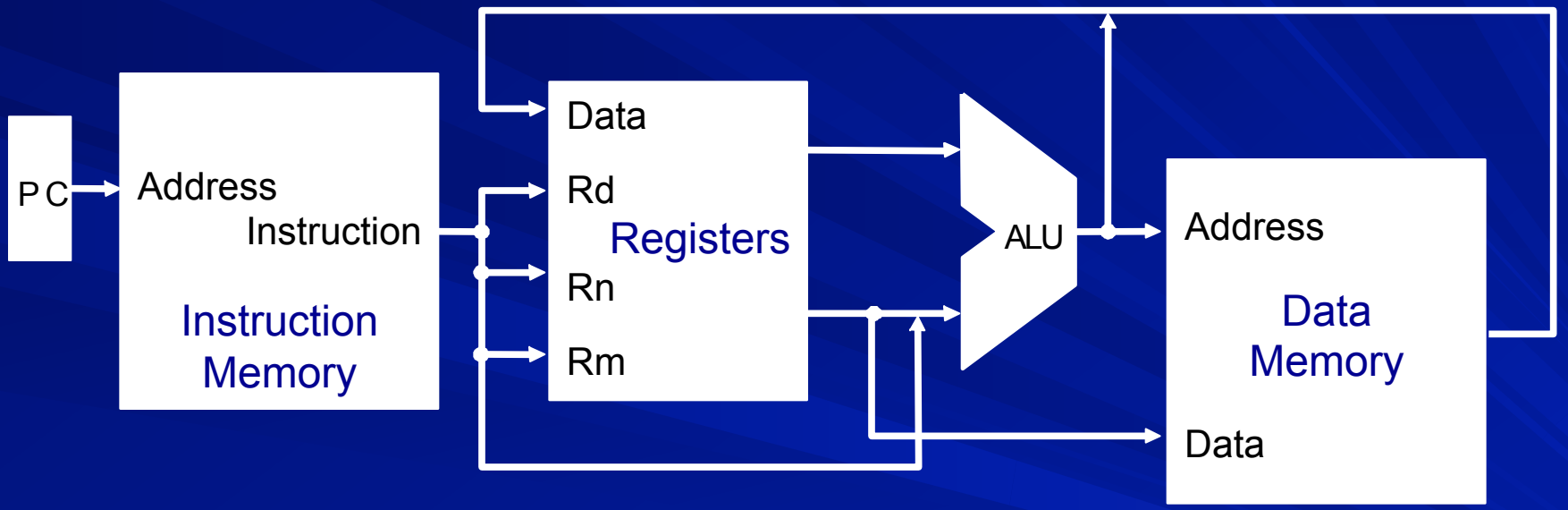
Will be discussed later

- Effect of relaxing the assumptions
- Incremental changes in the design to include other instructions

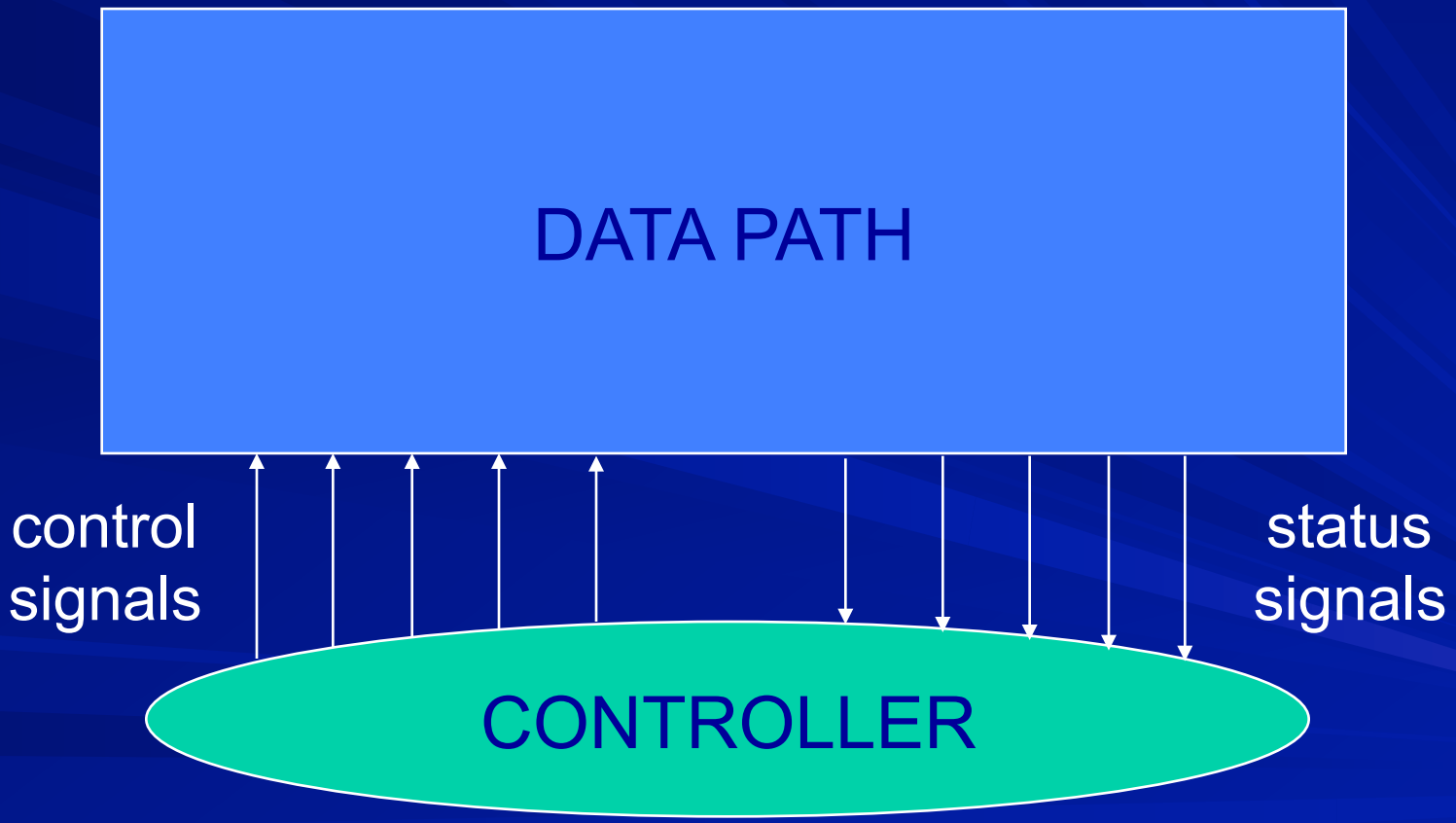
Generic Implementation

- use the program counter (PC) to supply instruction address
- get the instruction from memory
- read registers
- use the instruction to decide exactly what to do
- state change – registers, flags, memory

Design overview



Division into data path and control



Building block types

Two types of functional units:

- elements that operate on data values (combinational)
  - output is function of current input
  - no memory
- elements that contain state (sequential)
  - output is function of current and previous inputs
  - state = memory

Combinational circuit examples

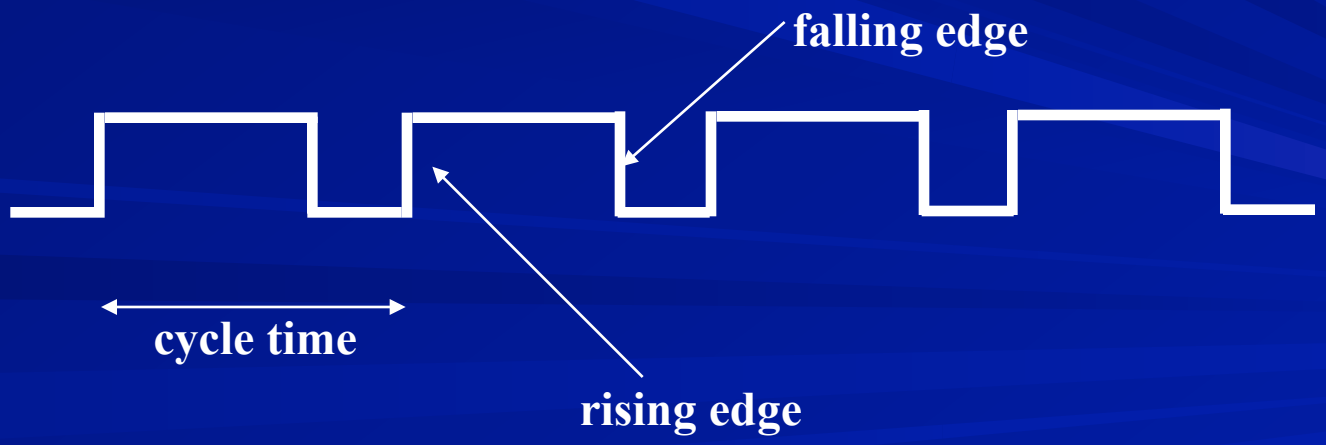
- gates: and, or, nand, nor, xor, inverter
- multiplexer
- decoder
- adder, subtractor, comparator
- ALU
- array multipliers

Sequential circuit examples

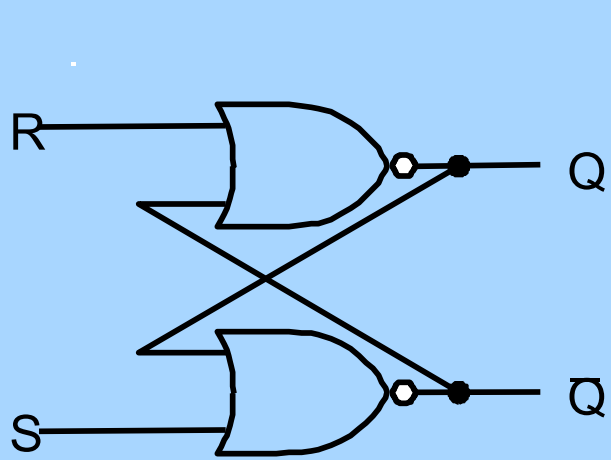
- flip-flops
- counters
- registers
- register files
- memories

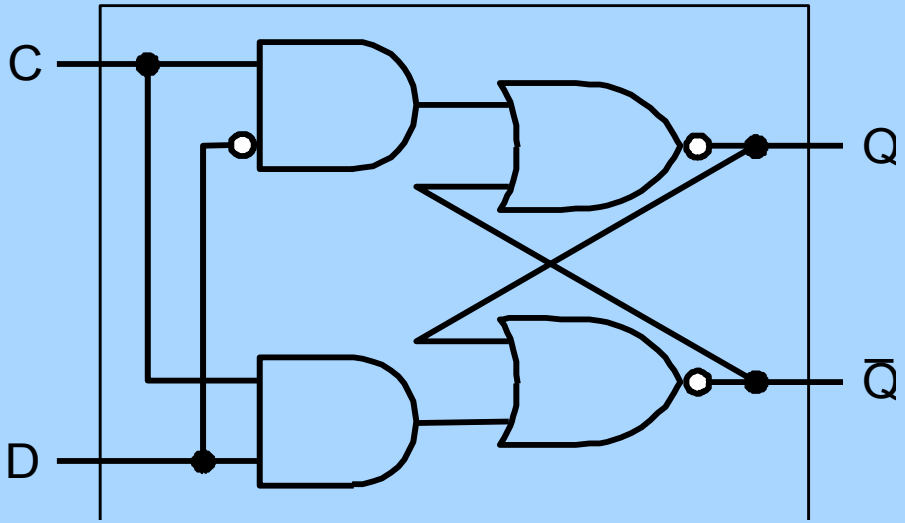
Clocked vs. unclocked circuit

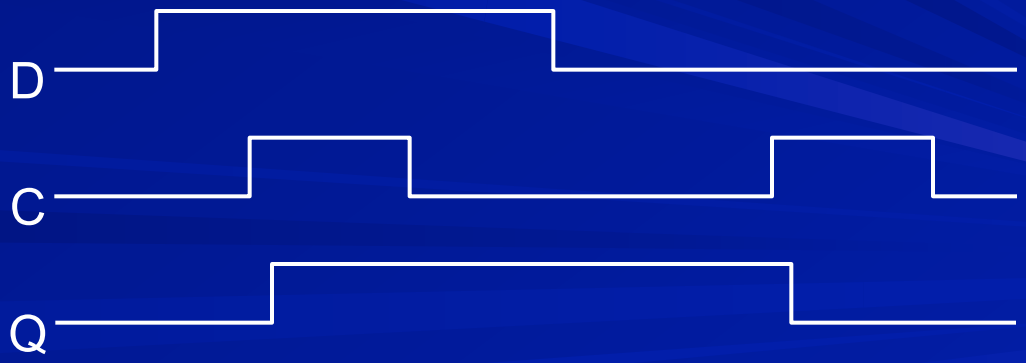
- Clocked state element
  - state changes only with clock edge
- Unclocked state element
  - state changes can occur with changes in other inputs



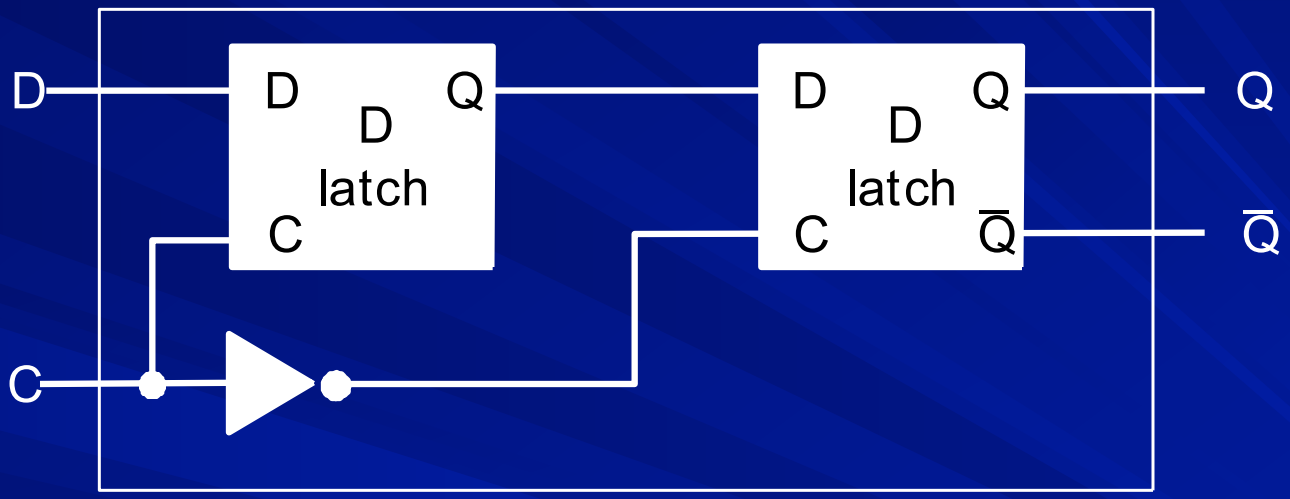
Unclocked state elements

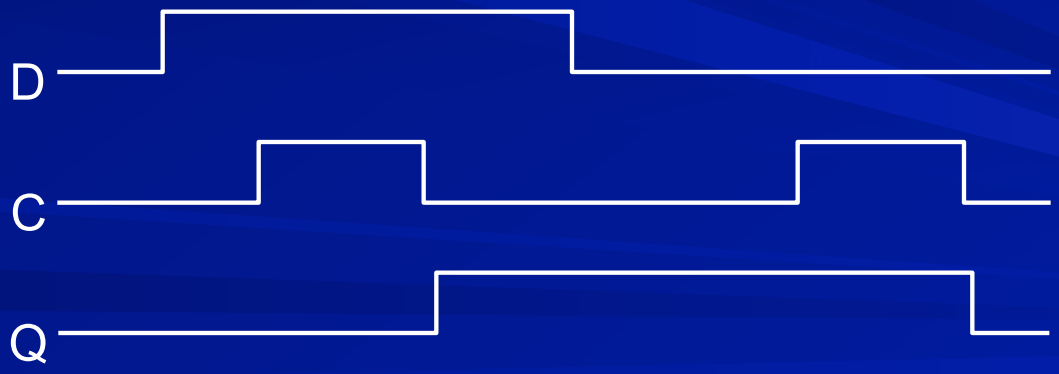




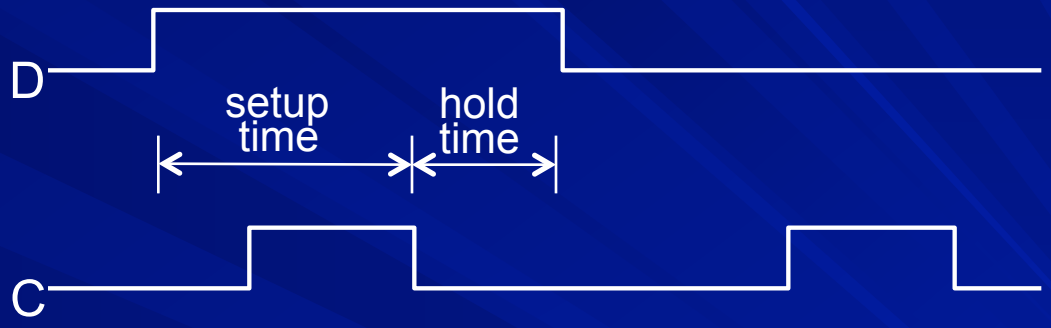


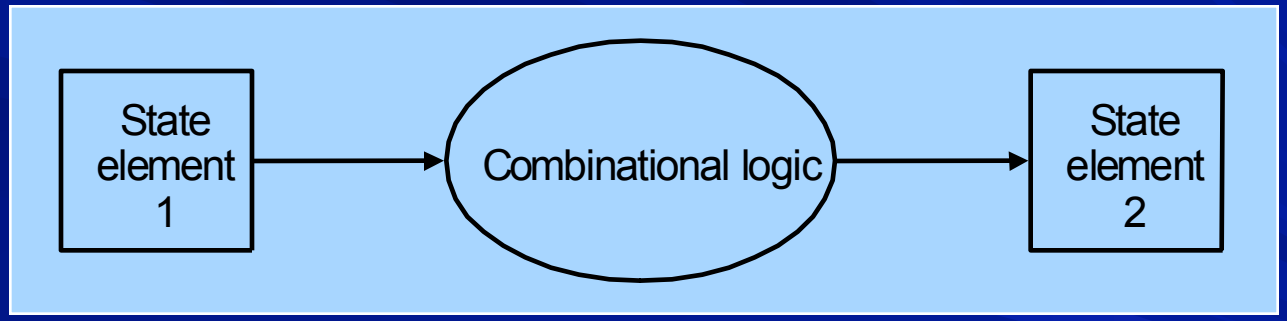
Clocked state elements





Clock and timings

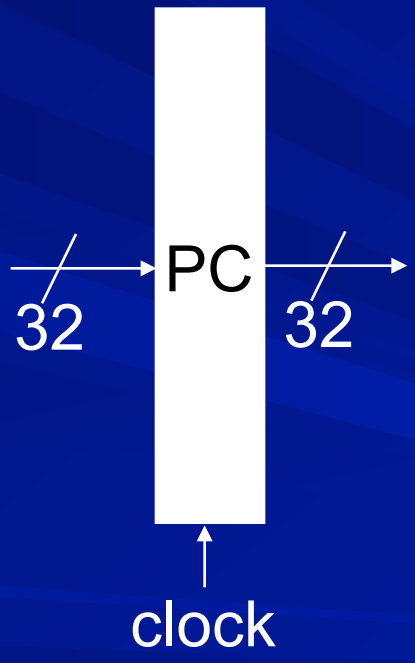




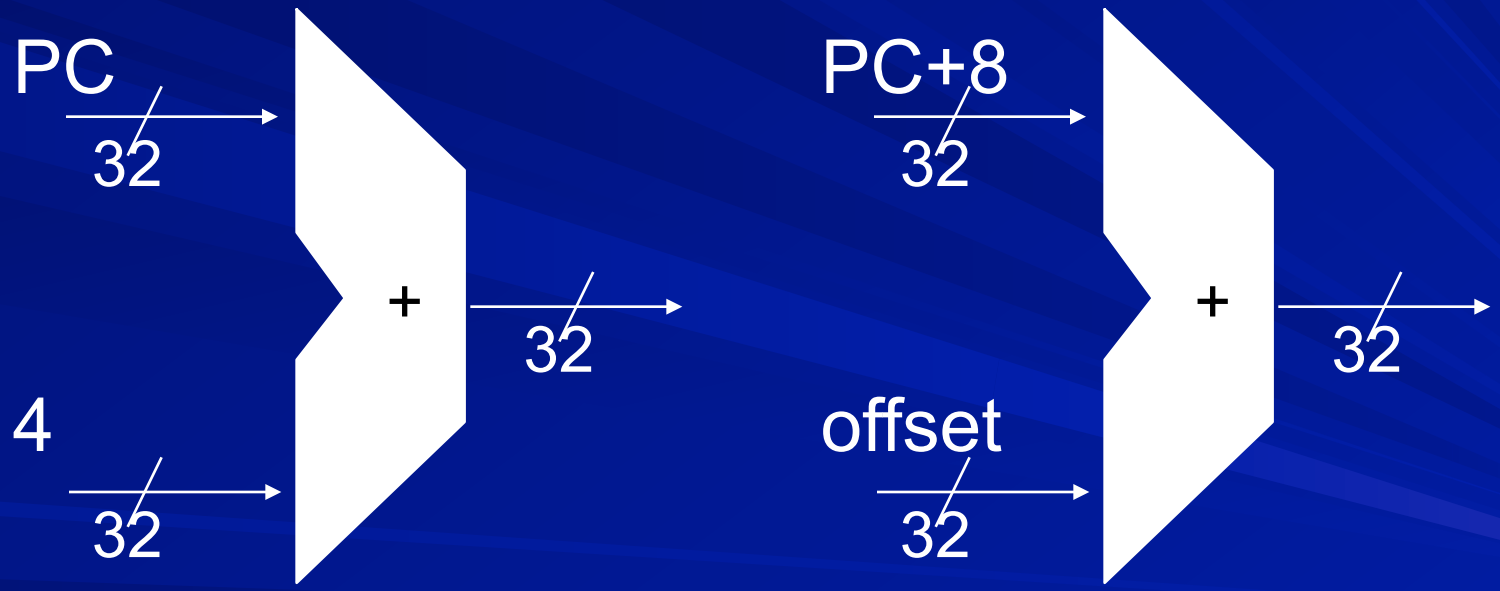
Components for Data Path

- Register
- Adder
- ALU
- Multiplexer
- Register file
- Program memory
- Data memory
- Bit manipulation components

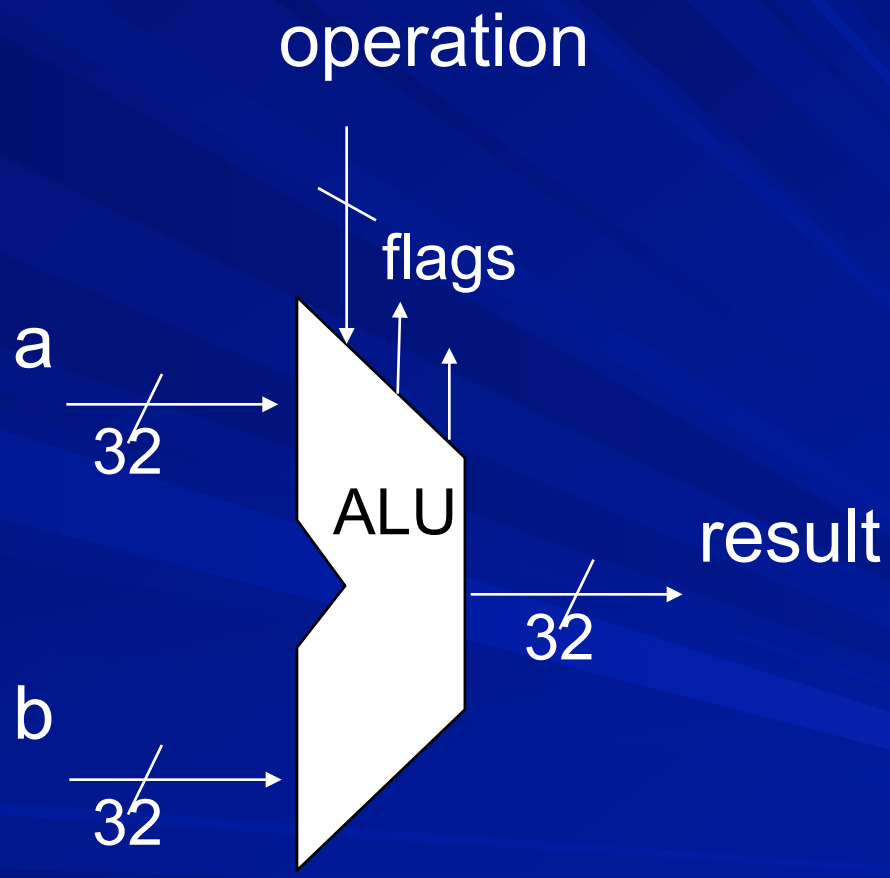
Components - register



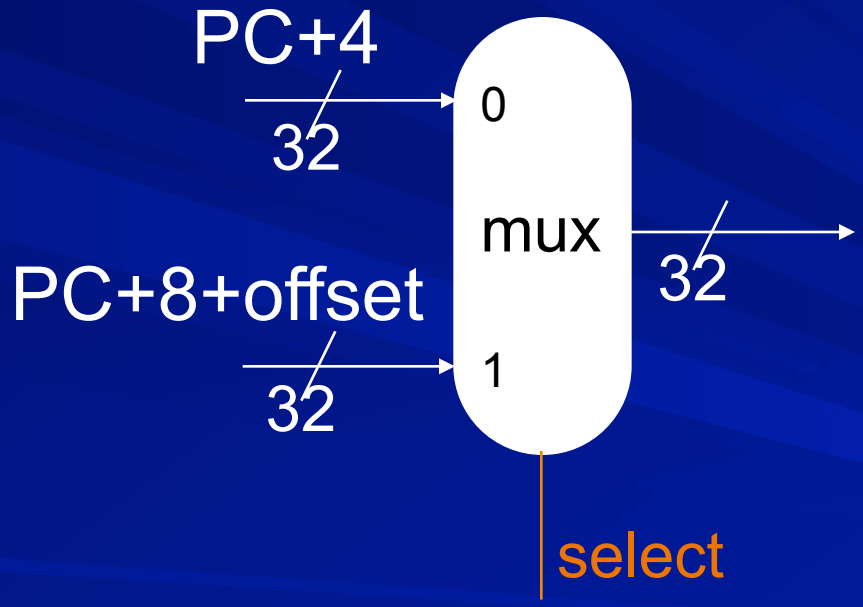
Components - adder



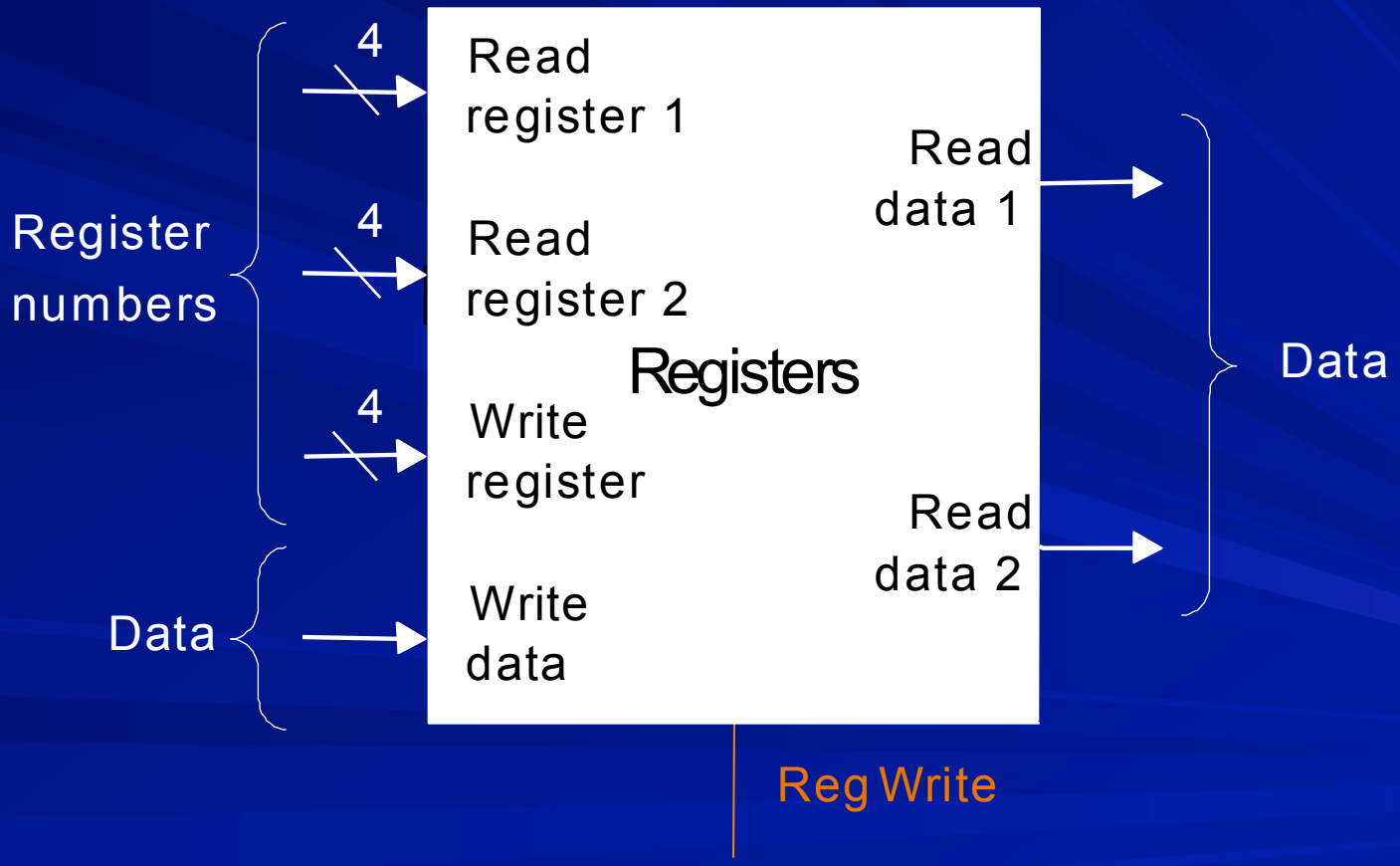
Components - ALU



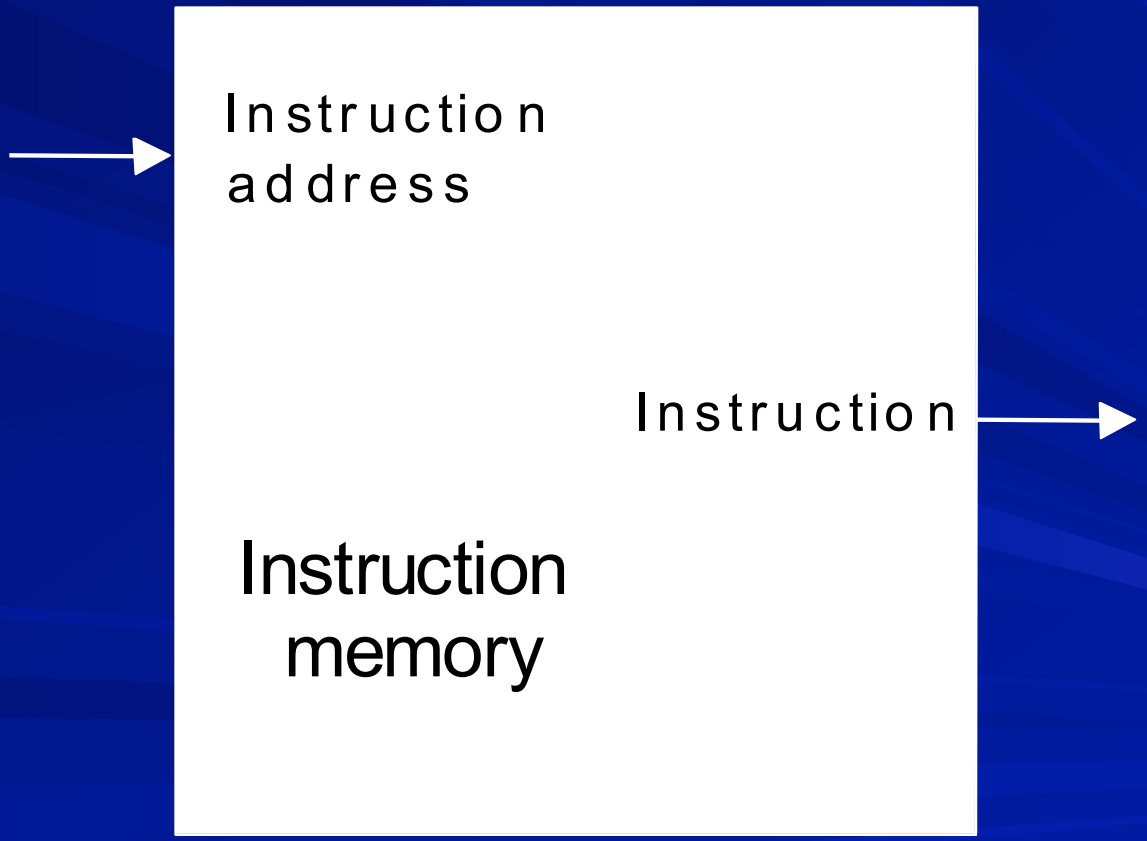
Components - multiplexers



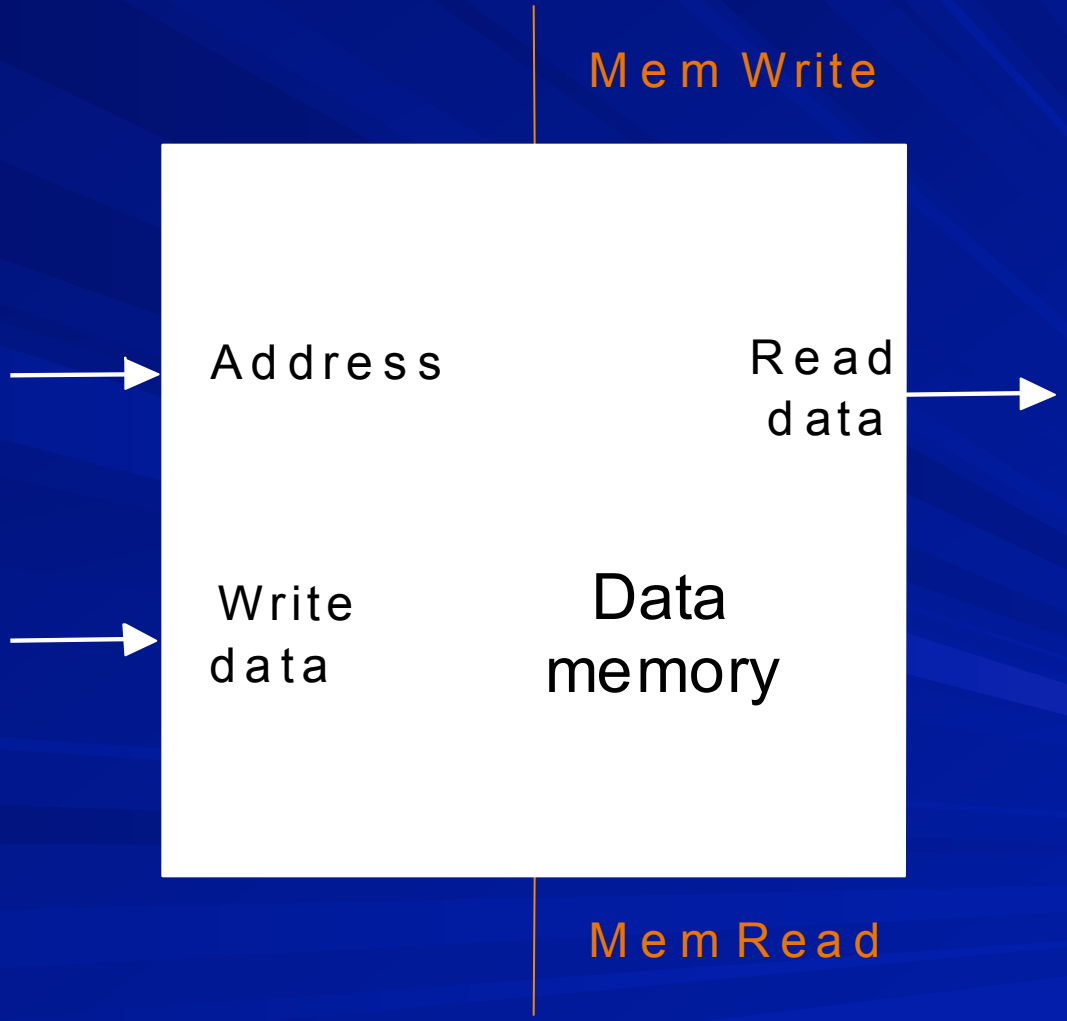
Components - register file



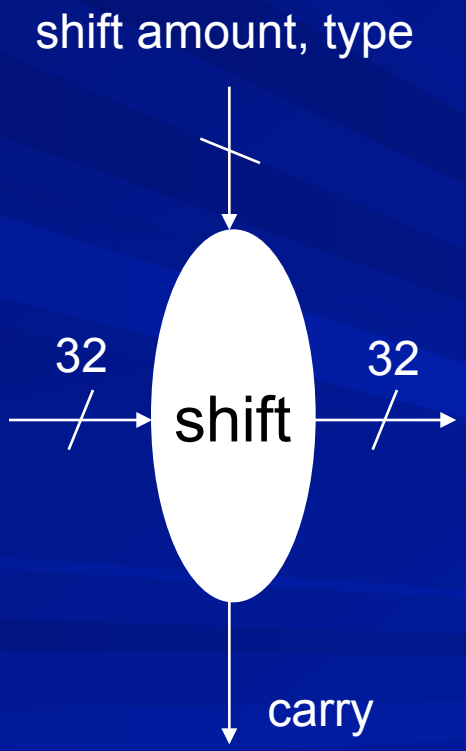
Components - program memory



Components - data memory



Components - shifter



Summary

- Instruction subset chosen
- Design organized as data path + control
- Building blocks - combinational and sequential, clocked and unclocked
- Clock and timings
- Components for Data path - register, mux, adder, ALU, register file, program and data memories