CpE111 Introduction to Computer Engineering

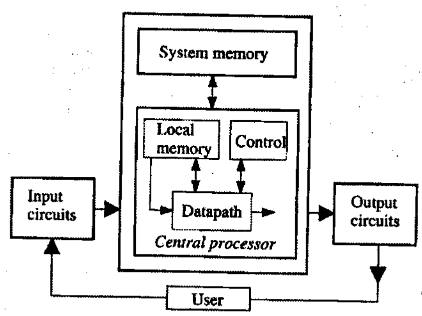
Dr. Minsu Choi

CH 10: Computer Basics

UNIVERSITY OF MISSOURI-ROLLA
The Name. The Degree. The Difference.

Major Components of a Computer

1. Input Networks: keyboard, mouse, etc,. provides input data to the computer.
2. Output Networks: monitor, printer, etc,. -> 1 and 2 are called I/O (input/output) devices.
3. Memory: provides the data storage.
4. Datapath: represents the paths that the data follow during the processing events.
5. Control: the control unit is responsible for insuring that the data is sent to the correct set of processing circuits. -> 4 and 5 are usually grouped together to form the central processing unit (CPU).

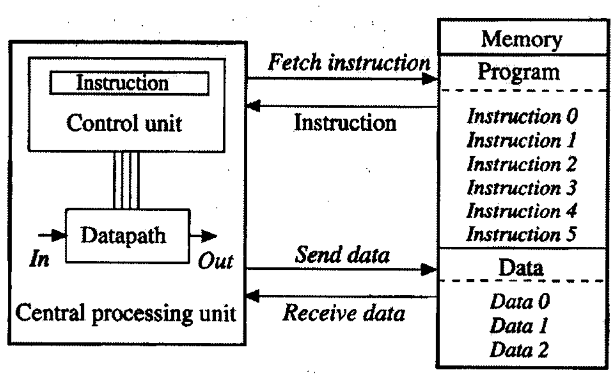


What Can A Computer Do?

- A computer can perform extremely complex tasks.
- But the internal operational modes are surprisingly limited.
- In general, a computer provides only two basic types of operations:
  1. Data movement.
  2. Binary operations (both logic and arithmetic).
- Every operation that a computer can perform is called an instruction.
- The group of instructions is called the instruction set.
- The number and types of instructions are determined by the structure of the datapath.

The von Neumann Model

- Most computers are based on this model.
- The main memory has program and data at the same time.
- CPU fetches and executes instructions sequentially.



4-Cycle Execution Procedure

- Instruction fetch: from memory to instruction register.
- Instruction decode: interpret the instruction and determine what needs to be done.
- Instruction execute: execute the instruction – necessary data also accessed.
- Storage: the results are stored back in the memory (if any).
- One instruction requires a total time of $t_{inst} = t_{IF} + t_{ID} + t_{EX} + t_S$.
- Obviously, a smaller value of t_{inst} implies a faster computer since more insts can be computed in a second.

ex) $t_{inst} = 0.1\,\mu s$ (It takes $0.1\,\mu s$ to execute 1 inst)

$f = 1/t_{inst} = 1/0.1\,\mu s = 10$ million insts per sec

$= 10$ MIPS

↳ useful CPU speed measure.

Programming

- High-level programming languages such as C or Java are used to program a computer.
  - Program: an ordered list of commands that tell the sequence of operations to accomplish a specific task.
  - Syntax: predefined manner in which commands are constructed.
  - Machine language: computer executable binary code – it is quite cumbersome for humans to use.
  - Compiler: a specific program which can translate high-level language programs into machine language codes.
  - Assembly language: human-understandable representation of machine code.

The Central Processing Unit (CPU)

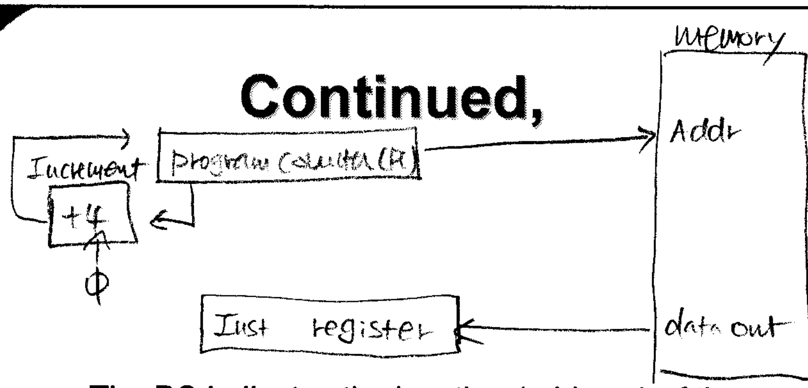
- Instruction fetch network
  - A program is a sequential listing of binary words.
  - Each word provides the information needed by the logic networks to perform a specific operation.
  - The size of a word depends on the computer.
  - Ex) 32-bit computer

32 bits = 4 bytes.
suppose that the given CPU is byte-addressible
⇒ the addresses for two in-order instructions differ by 4.

ex) Addr	Binary inst	Order
0400	--- 32bit inst	Inst Ø
		1
0404		2
0408		3

← Starting addr

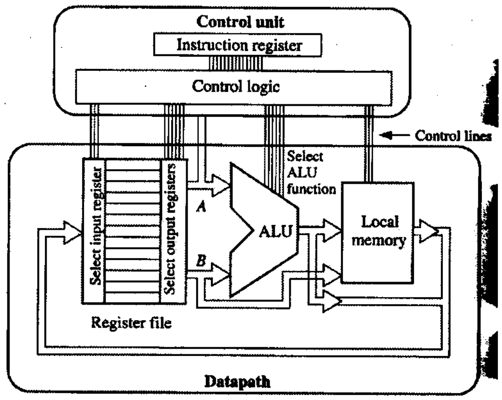
Continued,



1. The PC indicates the location (address) of the instruction being fetched.
2. The address is directed to memory.
3. Corresponding inst is fetched and stored in IR.
4. Do the rest of fetch-execution procedure.
5. PC = PC + 4 and go to 1. ← continue until the end of the program.

⇒ called inst fetch cycle

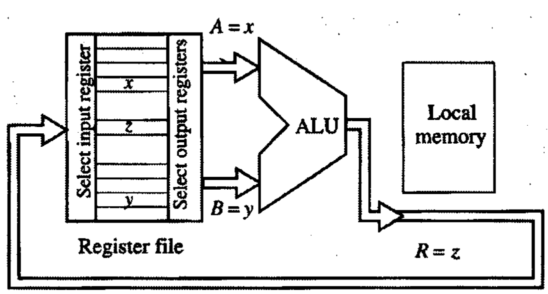
Concept of Datapath



- Control unit consists of IR & control logic.
- Datapath consists of reg file, ALU and local memory.

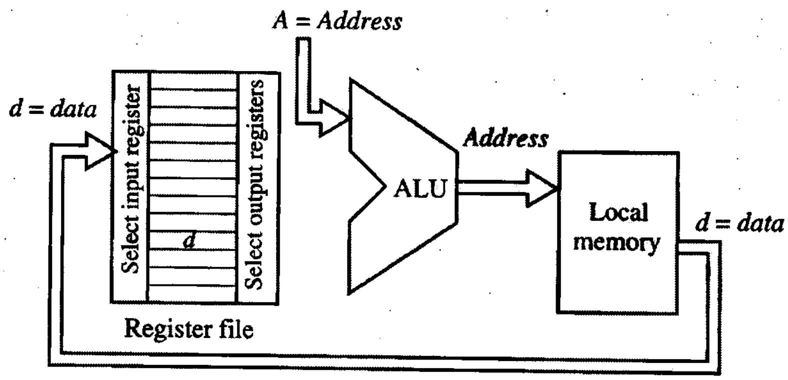
Datapath Operations

- Reg-to-reg ops: Takes data words from the reg file (fast temporary storage for words) and uses them as inputs into the ALU. Then result is stored back into the reg file.



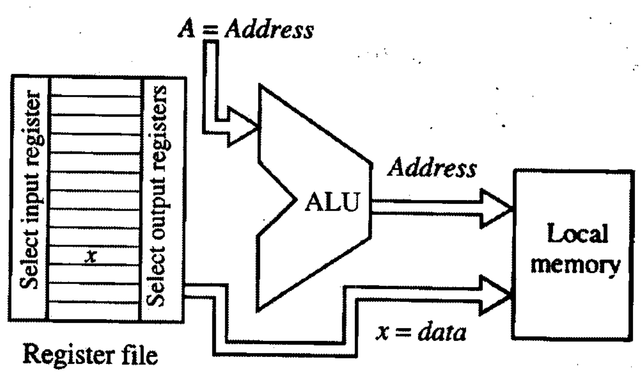
Continued,

- Load op: Move data from local mem to register.
- Ex) load word operation



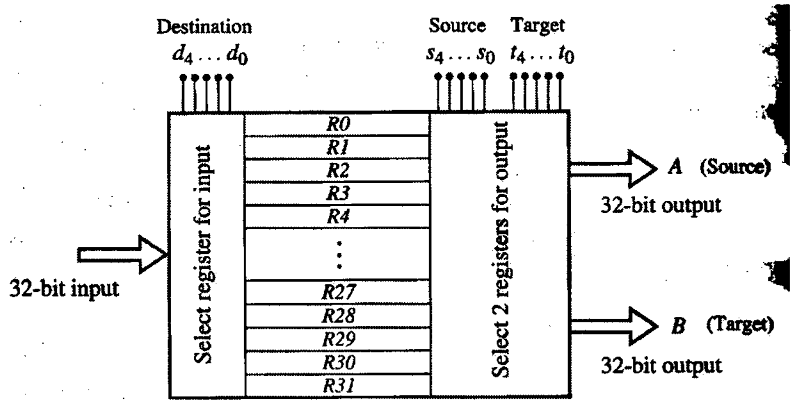
Continued,

- Store op: move data from reg to mem.
- Ex) store word operation

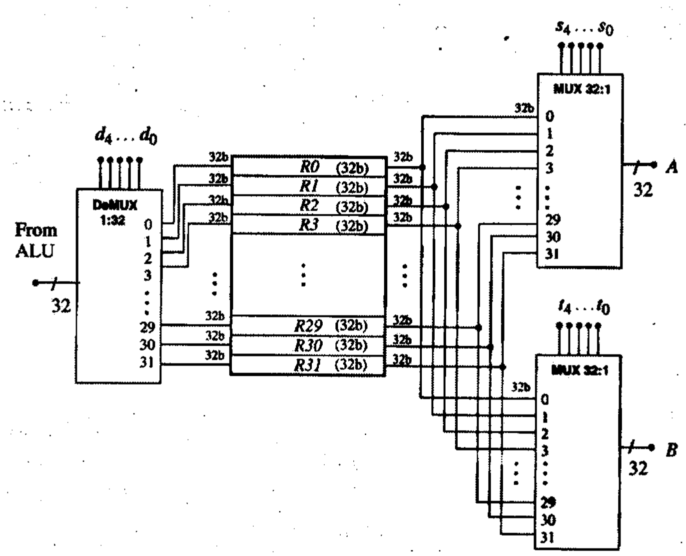


Datapath Components

- Register file architecture: ex) 32-register RF.



Unit-Level View of RF

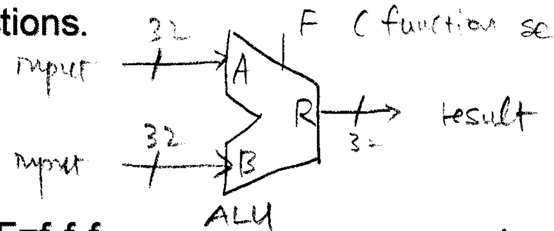


Arithmetic and Logic Unit (ALU)

- Provides both arithmetic functions and logic functions.

→ ADD, SUB ...

→ AND, OR ...



inputs control

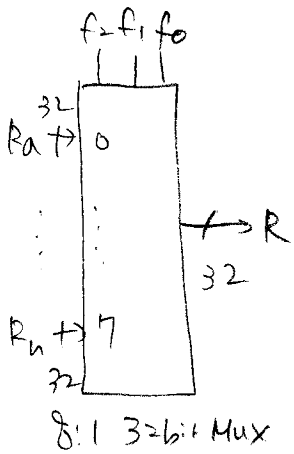
So, $R = R(A, B, F)$

- Ex) $F=f_2f_1f_0$

→ 8 possible values → 8 different operations can be specified.

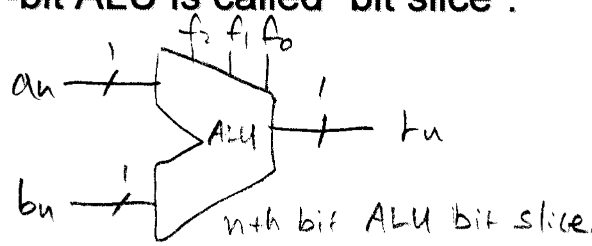
suppose that R_a R_b ... R_n are corresponding functions

$R = R_a \cdot (\bar{f_2}\bar{f_1}\bar{f_0}) + R_b(\bar{f_2}\bar{f_1}f_0) \cdots + R_n(f_2 f_1 f_0)$



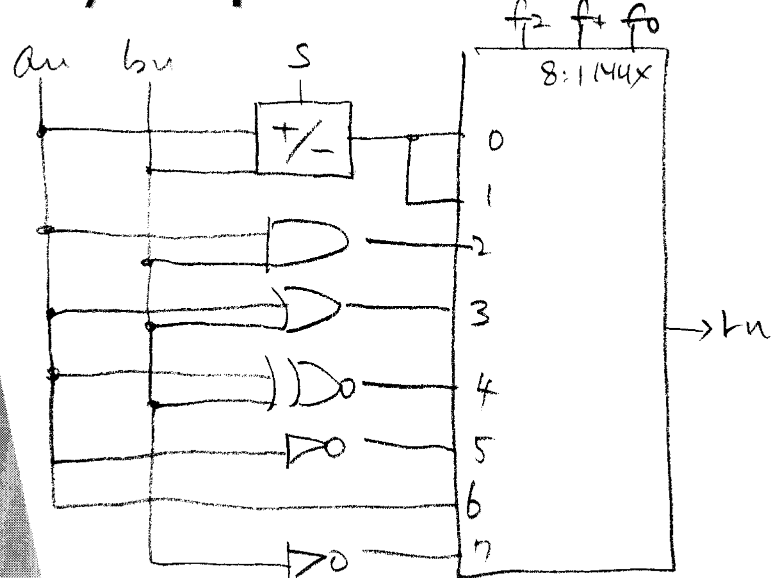
Parallel Structure of ALU

- 1-bit ALU is called "bit slice".



- A parallel grouping of 32 bit slices -> 32 bit ALU.

Ex) Sample ALU Bit-Slice Design



$f_2 f_1 f_0$	r_n
000	a_n plus b_n
001	a_n - b_n
010	$a_n \cdot b_n$
011	$a_n + b_n$
100	$\overline{a_n \oplus b_n}$
101	$\overline{a_n}$
110	a_n
111	$\overline{b_n}$

Concept of Local Memory (Cache)

- CPU, in general, has local memory called cache memory.
- Cache memory provides fast read/write storage. (usually SRAM used)
- But it is very small when compared to the size of the main memory. (SRAM is expensive)
- Hierarchical arrangement of different storage units is referred to as the memory hierarchy.
- If properly designed, considerable speed-up in read/write operations can be achieved.

Data flow path → ← Cache (small fast) → ← main mem (large slower)

SRAM DRAM.

Architecture

- 3 main components of CPU: register file, ALU and local memory.
  1. The ALU functions determine the type of arithmetic and logic ops that can be performed.
  2. The RF provides a set of fast local storage locations.
  3. The cache mem allows us to access to the large system memory.
- The instructions that can be implemented on a given computer are determined by the properties of each unit and how they are connected to form the system -> called "instruction set architecture (ISA)".

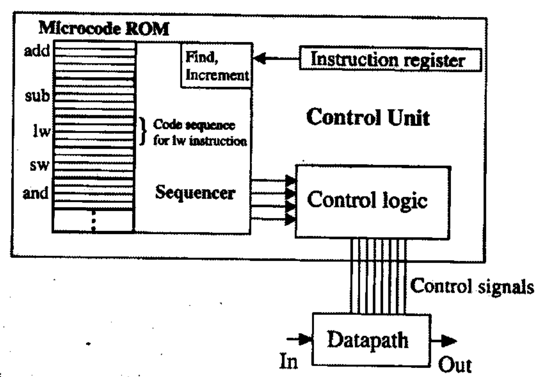
CISC & RISC Architectures

- CISC:
  1. Complex instruction set computer.
  2. Instruction set has larger # of insts.
  3. Easy to program since various instructions are provided.
  4. But, hardware implementation is complex and execution time per inst is longer.
- RISC:
  1. Reduced instruction set computer.
  2. Only reduced # of instructions provided.
  3. Hard to program since limited # of instructions are provided.
  4. But, hardware implementation is simple and execution time per inst is shorter.

CISC & Microprogramming

- Microprogramming embeds a sequential logic network inside of the control unit.
- It is similar to having a small computer that operates inside of the main computer.
- Breaks down every basic operations into a microinstruction.
- Designer creates an instruction by combining the needed microinstructions.
- Each microcode sequence is stored in a microcode ROM array.

Block Diagram



- An inst in the IR is sent to the sequencer which determines the location of the specified operation.
- Then, sequencer executes the sequence of micro insts.

Pros and Cons

- Pros:
  1. Powerful & flexible approach for increasing the inst set.
  2. Easy to program (compilation is also easy).
  3. Adding a new inst is easy.
  4. Modification to the inst set is also easy.
- Cons:
  1. The internal sequencer circuit must be added to the unit.
  2. Each microinst requires a time t_{micro} to complete. So, each computer-level inst takes different clock cycles to complete.
  3. The hardware tends to grow in complexity with the richness of the microcode inst set.

RISC

- Designed based on 80/20 rule: 80% of the program only uses 20% of the available insts.
- Ex) A computer has 500 insts then 80% of a typical program will use about 100 of them.
- RISC concept:
  1. Only include the most useful insts in the datapath.
  2. Insure that the datapath yields fast execution of every instruction.
  3. Single-pass datapath: each unit in the datapath can only be accessed once during execution of an instruction.

Pros and Cons

- Pros:
  1. The datapath can be optimized for the fastest throughput.
  2. Every inst takes the same amount of time (clock cycles).
  3. Only insts that can be completed in a single-pass through the data path are allowed.
- Cons:
  1. Compliers must be optimized to produce efficient codes.
  2. Length of a program tends to be longer.

Floating-Point Operations

- Arithmetic operations often require us to use fractional values (real numbers).
- Ex) $\pi = 3.141592\ldots$
  $e = 2.718\ldots$ etc ..
  4.25×10^{-4}
- In modern computers, a floating point representation is used to represent a real number using a binary bit pattern.
- Basic expression:

$$(-1)^S \times (1 + \text{significand}) \times 2^{(\text{exponent} - \text{bias})}$$

32-Bit IEEE Single Precision FP Representation

31	30 … 23	22 … 0
S	exponent	significand
1b	8b	23b.

ex) $a = -0.75_{10}$

$a = -1.1 \times 2^{-1}$

$$\Rightarrow (-1)^{1} \times (1 + \underbrace{.100\cdots}_{\text{significand}}) \times 2^{(126-127)}$$

sign bit

exponent

bias.

1b	8b	23 bits.
1	0111 1110	10 . . . 0
sign bit	126_{10} exponent.	significand

64-Bit IEEE Double Precision FP Representation

31	30 … 20	
S	exp	significand
1b	11b	20b

"

32b

32b.

bias = 1023, not 127!

ex) $P_x = 1/1 = 1$
$P_y = 1/2 = 0.5$ ⇒ X is ~~twice~~ faster than Y.

Computing Speed

- How to compare two computers?
  1. Choose a program of reasonable length and complexity.
  2. Run it on both systems.
  3. Compare the total execution times.
- Ex) Suppose that two systems, x and y are given.

for system X, the total exe time is T_x
" Y, " T_y
If $T_x > T_y$ then system Y is faster (at least for the program)
performance rating $p = 1/T$
If $P_x > P_y$ then Y has a higher performance.

Another Performance Evaluation Concept

- Instruction throughput = # insts processed per sec.
- Ex) X has 2 MIPS (mega insts per sec)
  Y has 1 MIPS (")
  ⇒ X has twice faster processing speed than Y.

Instruction Pipelining

- Datapath consists of a number of individual components. They are used sequentially for each inst.
- Ex) 5 stage datapath

inst → IF t_{IF} → ID t_{ID} → EXE t_{EXE} → MEM t_{MEM} → WB t_{WB} → retire

IF	ID	EXE	MEM	WB
instruction fetch cycle	inst decode cycle	inst exe cycle	mem access cycle	write back cycle.

Continued,

① non-pipelined (sequential) execution

1	2	3	4	5
Inst1	idle	idle	idle	idle

∈ stages in Datapath

inst1

inst1

inst1

inst1

inst2	

. . . . .

⇒ If each stage takes 1 clock cycle, throughput = 1 inst / 5 clock cycles

② next sheet!

Program Completed

University of Missouri-Rolla

UMR UNIVERSITY OF MISSOURI-ROLLA
The Name. The Degree. The Difference.

(2) pipelined execution

I_1				
I_2	I_1			
I_3	I_2	I_1		
I_4	I_3	I_2	I_1	
I_5	I_4	I_3	I_2	I_1

after

⇒ steady-state throughput = 1 inst / 1 clock cycle.

example next sheet

ex) # of data path stage = K = 5

# of inst = N = 1000

cycle time = T.

then, t_1 = total execution time for the non-pipelined execution :

$= K \cdot N \cdot T = 5 \cdot 100 \cdot T = 5000T.$

t_{pipe} = total exe time for the pipelined exe.

⇒ The first $(K-1)T$ will be used to fill the pipeline, then 1 inst retires at each clock cycle afterwards.

$$= \underbrace{K \cdot T}_{\text{After } K \text{ clock cycle the first one retires}} + \underbrace{(N-1) \cdot T}_{\text{then 1 inst/1 clock}}$$

$= K \cdot T + N \cdot T - T$

$= (K + N - 1)T$

$\Rightarrow (5 + 1000 - 1)T = \underline{\underline{1004T}}$ #