

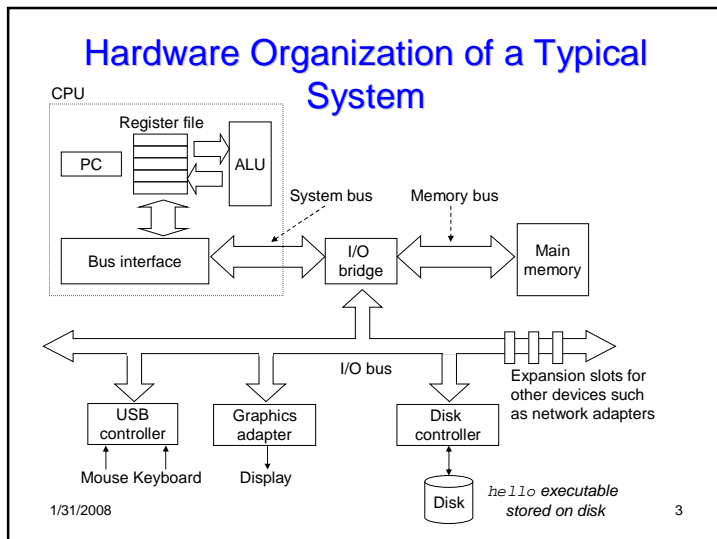
CSC 252: Data Representation

1/31/2008 1

Hardware Components of a Computer System

- Processor
 - Datapath
 - Control
- Memory
- Input and Output devices

1/31/2008 2



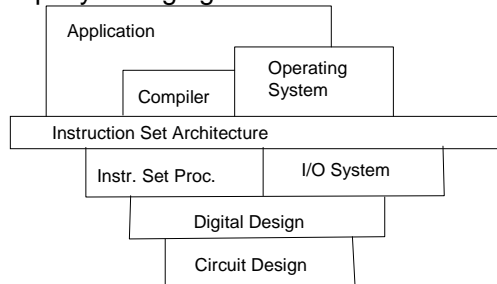
The Principle of Abstraction

- Grouping principle
 - Levels/layers of abstraction by which each layer only needs to understand that immediately above and below it

1/31/2008 4

What is Computer Architecture?

- Coordination of levels of abstraction under a set of rapidly changing forces



1/31/2008

5

Topics to be covered:

- Data representation and computer arithmetic
- Assembly-level programs and instruction-set architectures
- Processor architectures
- Memory and storage hierarchies
- Performance optimization
- Exceptional control flow
- I/O devices
- Concurrency

1/31/2008

6

Data Representation

- Memory: a large single-dimensional, conventionally byte-addressable, untyped array
- Byte ordering – big versus little endian
- Possible common interpretations
 - Instruction
 - Integer
 - Floating point
 - character

1/31/2008

7

Number Representation

An n digit number can be represented in any base as

MSD	...	LSD
n-1	...	0

The value of the *i*th digit *d* is $d \times \text{base}^i$, where *i* starts at 0 and increases from right to left

Decimal (base 10) is the natural human representation,
binary (base 2) is the natural computer representation

E.g. $1100_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 0 \times 2^0 = 12_{10}$

1/31/2008

8

Integer Representation

- Sign-magnitude (MSB as sign)
- Ones' complement (complement individual bits)
- Two's complement (most common representation – complement the power of 2)

1/31/2008

9

Bit-Level Operations in C

- Logical
 - `||` `&&` `!`
- Bitwise
 - `|` `&` `^` `~` `>>` `<<`
 - Arithmetic versus logical right shift

1/31/2008

10

Integer Arithmetic

- Normal base 2 2's complement addition works on both positive and negative numbers
- Shortcuts
 - 2's complement = 1s' complement + 1
 - 2's complement representation of n digit number as n+m digit number --- sign extend

1/31/2008

11

Algebraic Laws for Logical Expressions

- AND and OR are commutative
- AND and OR are associative
- AND is distributive over OR; OR is distributive over AND
- TRUE is the identity for AND; FALSE is the identity for OR
- FALSE annihilates AND; TRUE annihilates OR
- AND and OR are idempotent ($p \text{ AND } p \equiv p$ OR $p \equiv p$)
- Subsumption
 - $(p \text{ OR } (p \text{ AND } q)) \equiv (p \text{ AND } (p \text{ OR } q)) \equiv p$
- DeMorgan's Laws:
 - $\text{NOT}(p \text{ AND } q) \equiv (\text{NOT } p) \text{ OR } (\text{NOT } q)$
 - $\text{NOT}(p \text{ OR } q) \equiv (\text{NOT } p) \text{ AND } (\text{NOT } q)$

1/31/2008

12

Circuit Design

- Gate:
 - Basic electronic device.
 - Computes a Boolean function.



- AND, OR, NOT, NAND:
 - Easy to implement
 - Conceptually any number of inputs
 - Used in practice



1/31/2008

13

Where does power go in CMOS

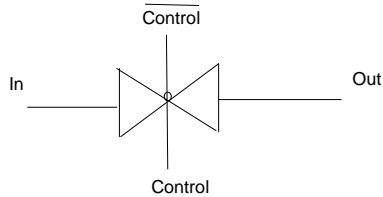
- Dynamic power – charge and discharge output capacitance – CV^2fN
- Short-circuit power – for the short time that both pull-up and pull-down transistors are on
- Leakage power

1/31/2008

14

Steering Logic

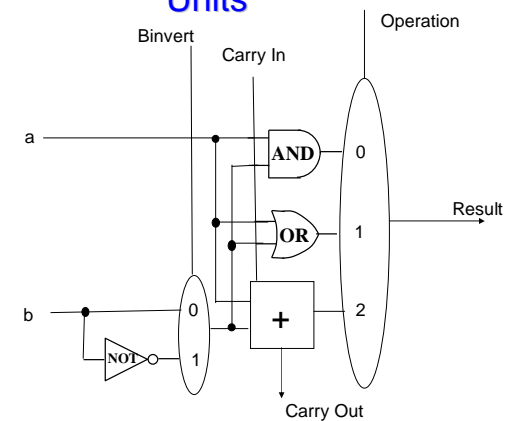
- Multiplexors and demultiplexors: transmission gates



1/31/2008

15

Combinational Logic: Arithmetic Logic Units



1/31/2008

16

Sequential Logic: Memory and Control

- Sequential:
 - Output depends on the **current** input values and the **previous** sequence of input values.
 - Are **Cyclic**:
 - Output of a gate feeds its input at some future time.
 - **Memory**:
 - Remember results of previous operations
 - Use them as inputs.
 - Example of use:
 - Build registers and memory units.

1/31/2008

17

Sequential Circuits for Memory Elements.

- Memory element:
 - A collection of gates capable of producing its last input as output
 - They are **sequential** circuits
 - Their behavior depends on current and past inputs
- Flip-flop:
 - A 1-bit memory element
 - Typical flip-flop:
 - Takes two inputs (load and data-in)
 - Produces one output (data-out)

1/31/2008

18

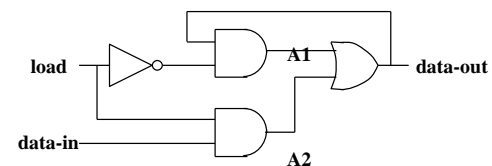
Flip-Flops

- Load==0:
 - The circuit produces the stored value as output
- Load==1:
 - The circuit stores the value **data-in** and
 - Produces it as output

1/31/2008

19

Flip-Flop Circuit



- Load=1 \Rightarrow A1=0 \Rightarrow data-out=A2=data-in
- Load=0 \Rightarrow A2=0 \Rightarrow data-out=A1
 - Which is the previously stored value

1/31/2008

20

Floating Point

- Recall scientific notation
 - $65.4 = 6.54 \times 10^1$ (normal form – 1 non-zero leading digit)
 - Mantissa = 6.54, exponent = 1, radix (base) = 10
 - Mantissa – sign/magnitude representation
 - Exponent – biased notation
- Single-precision
- S|exp+127|significand
1| 8-bits | 23-bits
- Double-precision
- S|exp+1023|significand
1| 11-bits | 52-bits

1/31/2008

21

Floating Point Addition

- Align decimal point by matching larger exponent
- Add significands (possibly unnormalized)
- Normalize – shift sum/adjust exponent
- Round number to fit in significand field

1/31/2008

22

Floating Point – Additional bits for accuracy

- Borrow/carry bit – to take into account overflow (carry/borrow) in result
- Guard bit – to take care of normalizing left shift
- Round bit – for correct rounding
- Sticky bit – fine-tune rounding – additional bit to the right of round that is set to 1 if any 1 bit falls off the end of the round digit

1/31/2008

23

IEEE 754 Floating Point Rounding

- Four modes:
 - Round to +infinity – add 1 if round or sticky bit is set and result ≥ 0
 - Round to -infinity – add 1 if round or sticky bit is set and result < 0
 - Round to 0 (truncate)
 - Round to nearest number (default) – round to even when exactly halfway
 - Add 1 if round bit and LSB of result are set, or, if round bit and sticky bit are set
 - Minimizes mean error introduced by rounding

1/31/2008

24

Zero, Infinity, NAN

- Zero – 0|0...0|0...0
- +/- infinity – S|1...1|0...0
- Nan – S|1...1|non-zero

1/31/2008

25

Floating Point Multiplication

- Exponent of product = sum of exponents – bias
- Multiply significands – decimal point location = sum of digits to the right of decimals
- Normalize – check for overflow/underflow
- Round
- Sign – positive if both are the same, negative if otherwise

1/31/2008

26

Exceptions

- IEEE standard specifies defaults and allows traps to permit user to handle exceptions
 - Invalid operation: e.g., sqrt of –ve number, 0/0, inf/inf
 - Result is NaN
 - Overflow: result is +/- inf unless overflow exception is enabled
 - Divide by 0: result is +/- inf if exception not enabled
 - Underflow: non-zero result underflows to 0
 - Inexact: rounded result not the actual result

1/31/2008

27