CSE 111

Fall 2010

September 27 – October 1

Announcements

- Lab 2 Part 1 assigned last week
  - Due 10/3
- Lab 2 Part 2 this week in lab
  - Due at your next lab session
    - (Week of October 11th for most of you)
- Some schedule changes (see website)
  - Labs will not meet the week of October 4th
  - Exam 2 date change
  - B5 recitation changes
- Exam 1 – Monday, October 4th in lecture

Floating Point Numbers (Fractional Numbers)

- Convert the base 10 number .25 to base 2

$$\begin{array}{r} 0.25 \\ * \quad 2 \\ \hline 0.5 \\ * \quad 2 \\ \hline 1.0 \end{array}$$

in base 2, .25 is .01

[illegible]

Floating Point Numbers in Binary

- Let's see the conversion of the base 10 number .1

0.1 ↓ *2
0.2 ↓ *2
0.4 ↓ *2
0.8 ↓ *2
1.6 ↓ *2
1.2 ↓ *2
0.4 ↓ *2
0.8 ↓ *2
1.6 ↓ *2
1.2 ↓ *2

.0001100110011001100110011

Floating Point Numbers

- Are always estimated when we store them inside the computer
- Standard representation (using 32 bits)

1bit	8 bits	23 bits
Sign bit	Exponent	Mantissa

Floating Point Numbers

- The exponent is stored as a biased exponent. In the process, we add 127 to the exponent and store that value.
- The mantissa takes the form 1.__________

Floating Point Numbers Example

- Convert 11.375
- First, convert both parts of the number (to the left of the decimal point and to the right of the decimal point) to base 2
- 1011.011
- Then, make this number take on the form of the mantissa by moving the decimal point. The number of places you move the decimal point becomes the exponent.

Floating Point Numbers Example

- 1011.011
- 1.011011 e^3
- Now, we are ready to store the information:

1bit	8 bits	23 bits
0	3+127=130	01101100000000000000000
	in binary	We don't store the 1 because all of them start with 1

Consequences

- Since we can only approximate the floating point number, we can get errors with those numbers

3 Boolean operations

AND

Input 1	Input 2	Output
True	T	T
T	False	F
F	T	F
F	F	F

Computer
True = 1
False = 0

0	0	0
0	1	0
1	0	0
1	1	1

OR

0	0	0
0	1	1
1	0	1
1	1	1

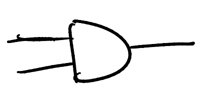
XOR – "Exclusive or"

0	0	0
0	1	1
1	0	1
1	1	0

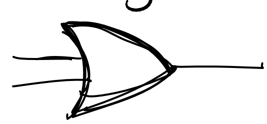
NOT

0	1
1	0

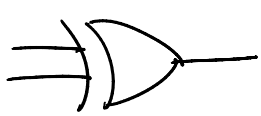
AND gate



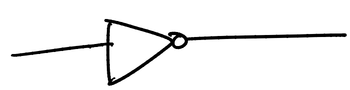
OR gate

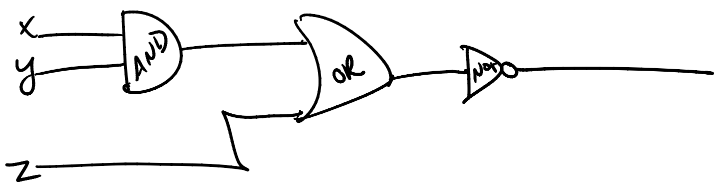


XOR gate

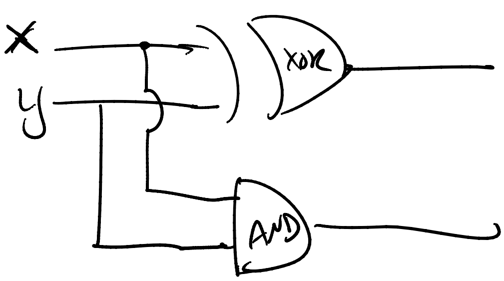


NOT



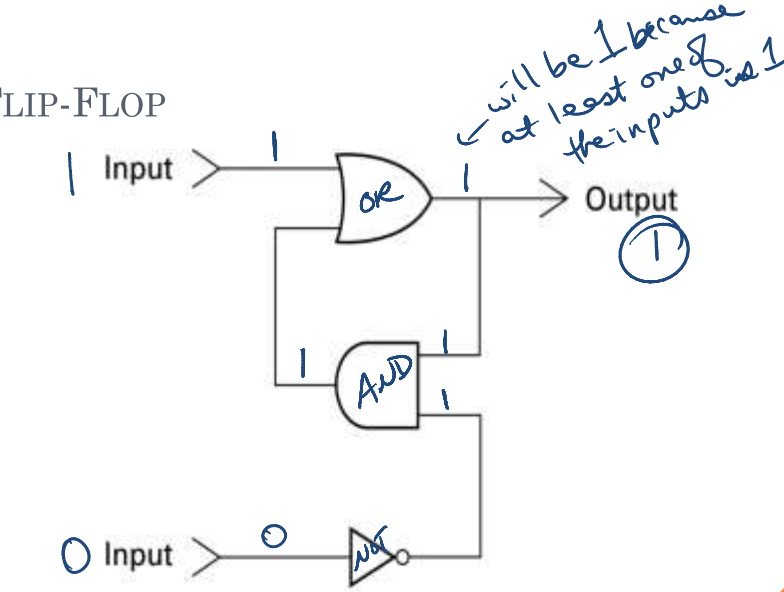


NOT((x AND y) OR z)

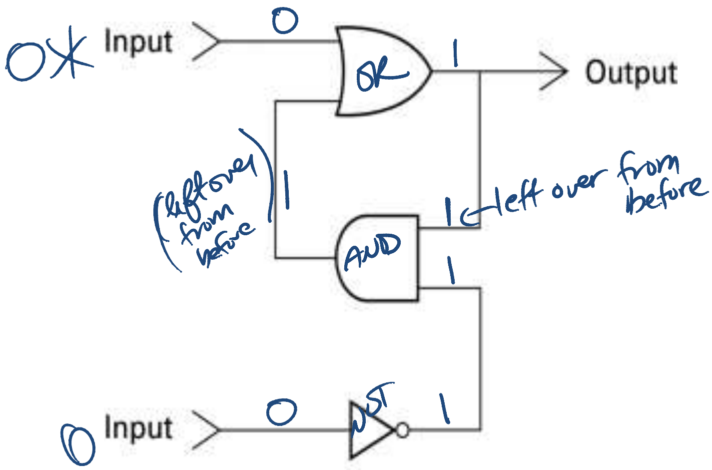


x	y	AND	XOR
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

FLIP-FLOP



FLIP-FLOP



MAIN MEMORY

- Composed of addressed cells that can be accessed as needed (randomly).

RAM (Random Access Memory)
- Fastest
- Volatile – once the power goes off, any data stored is lost

NON-VOLATILE STORAGE MEDIA

- Magnetic disks
  (Hard drives)
- Optical disks
  CD's & DVDs
- Flash memory

Computer Architecture

- CPU Central Processing Unit
  ↳ Control the manipulation of data

Parts of the CPU

- ALU
  Arithmetic/Logic Unit
- Control Unit
  Coordinating machine activities
- Register Unit
  Storage inside the CPU

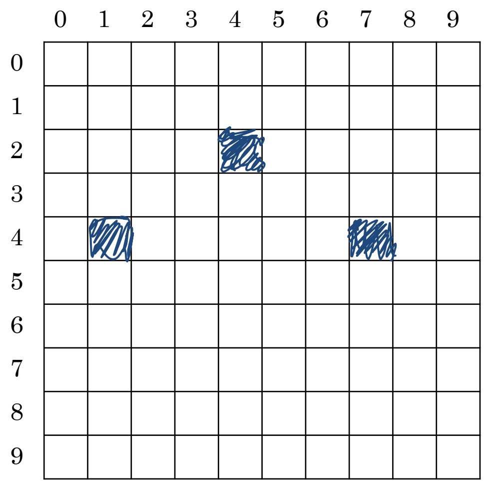
How Instructions get Executed

- Information moved from the main memory into the CPU, processed, and then the results are moved back to main memory.

Example

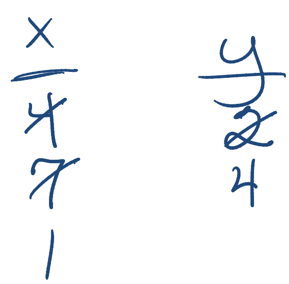
- Two volunteers to be ALU/Memory
- Volunteer to be CPU
- I'll be the display

DISPLAY



INSTRUCTIONS EXECUTED

- Add 4 to x
- Add 2 to y
- Plot (x,y)
- Add 3 to x
- Add 2 to y
- Plot (x,y)
- Subtract 6 from x
- Plot (x,y)



Full Set of Instructions

- Add 4 to x
- Add 2 to y
- Plot (x,y)
- Add 3 to x
- Add 2 to y
- Plot (x,y)
- Subtract 6 from x
- Plot (x,y)
- Add 5 to x
- Subtract 3 from y
- Plot (x,y)
- Subtract 1 from x
- Add 5 to y
- Plot (x,y)
- Subtract 3 from x
- Subtract 1 from y
- Plot (x,y)
- Subtract 4 from y
- Plot (x,y)
- Subtract 1 from x
- Add 1 to y
- Plot (x,y)
- Add 4 to x
- Subtract 1 from y
- Plot (x,y)
- Add 1 to x
- Add 4 to y
- Plot (x,y)
- Subtract 3 from x
- Add 1 to y
- Plot (x,y)
- Subtract 5 from y
- Plot (x,y)
- Add 4 to x
- Add 2 to y
- Plot (x,y)
- Subtract 6 from x
- Plot (x,y)
- Add 3 to x
- Add 4 to y
- Plot (x,y)
- Add 3 to x
- Subtract 5 from y
- Plot (x,y)

Finished Display