CS 61 Lecture 2: Data Representation

Agenda:

1. How do we represent #s?
2. How does a machine add?
3. Will the computer always produce the expected answer?

How do we represent #s

binary numbers 0/1

↳ 4 different techniques for doing so

most common today

#1: charge on a capacitor

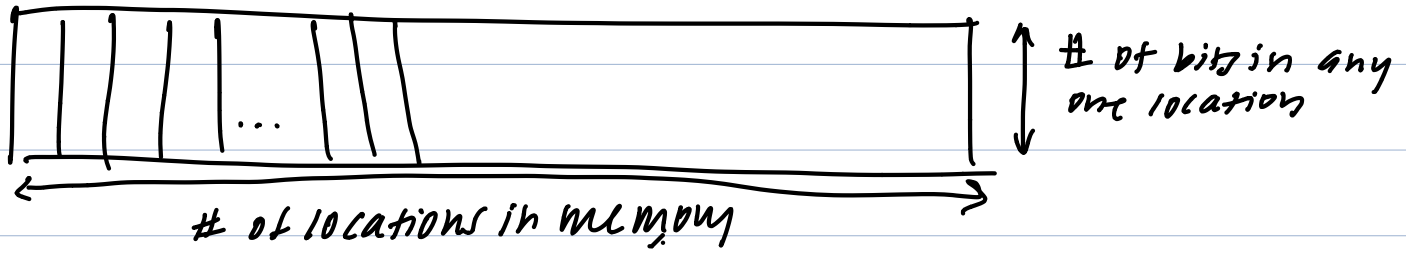
#2: current flowing or not (old ECL circuits)

#3: polarity of a magnetic material

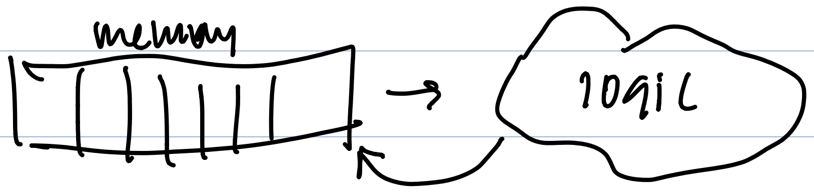
#4: presence or absence of something

Computer Memory

computer memories ≡ like bricks



1. How many do I actually have?
2. How many could I address

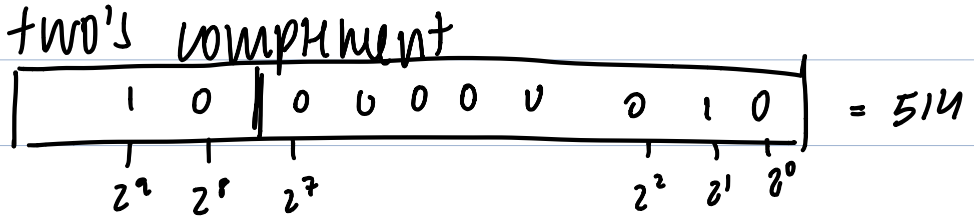


add

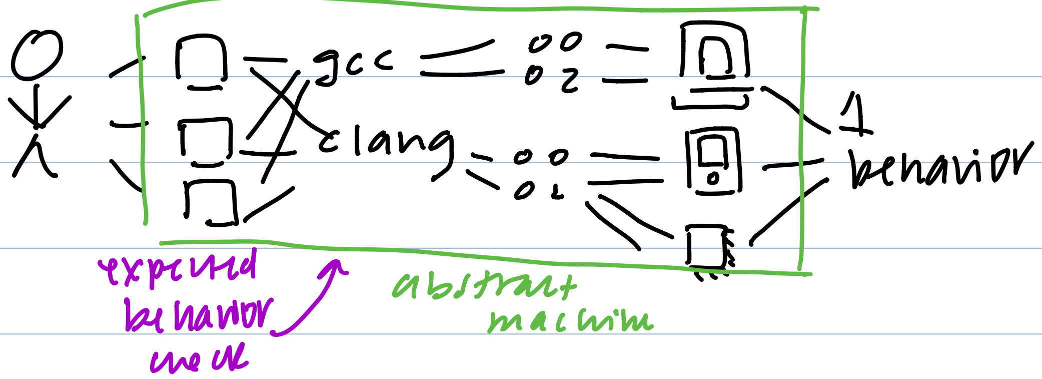
a	0011
⊕ b	0101
total	1001

Behavior

two's complement



Who/what defines expected behavior?



- C++ memory model in documentation
  - memory available is 1+ sequences of contiguous bytes w/ a unique address
- C++ object model
  - constructs that create, destroy, refer to, access, and manipulate objects
  - occupies a region of storage
- Program Execution
  - observable behavior conformity
- well-formed program
  - syntax rules, diagnosable semantic rules, one-definition rules

↓ machine-understandable code

↓ nothing that results in undefined behavior

↓ no ambiguity