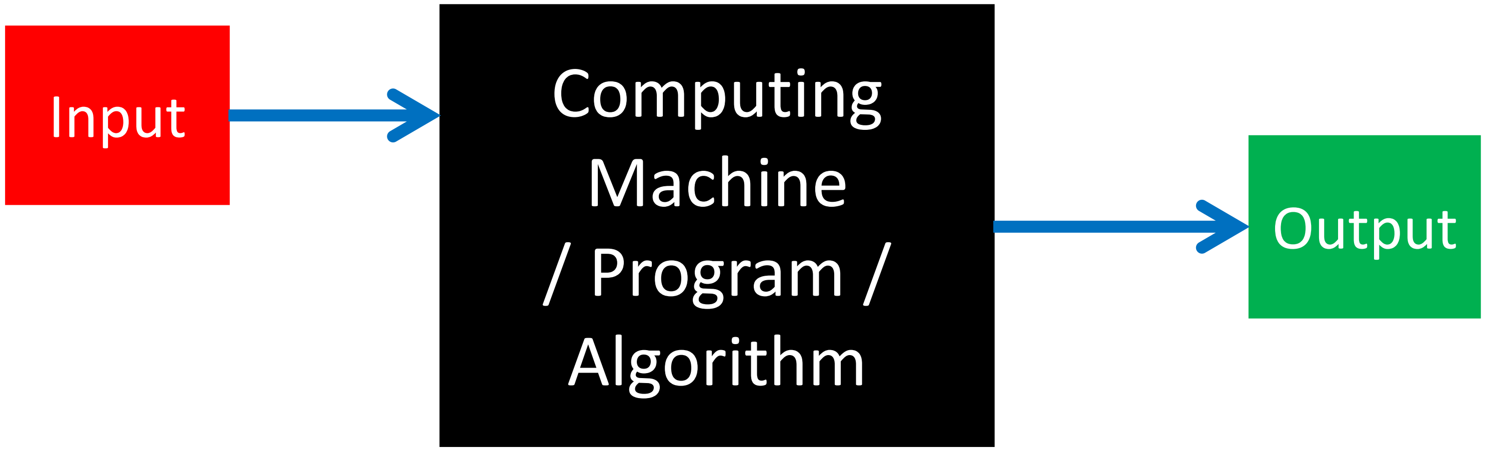
CS3102

September 6

Questions

Recall: “Carnot Engine” of computing

- What “type” is the input?
  - String (bit strings)
- What “type” is the output?
  - String
- What have we put in the black box so far?
  - Circuits of logic gates
  - Python/Java/C++
  - X8086
  - NAND-CIRC



What goes in here?

What do we compute on?

- Always strings! (that represent other things)
  - Images
  - DNA
  - Web pages
  - Temperature
- To compute on non-string things, we need a *representation scheme*
  - Function $E: U \to \{0,1\}^*$
    - E should be one-to-one

Model of Computing

- To define a model of computing, we need:
  - A way to give it input
    - Pre-defining the values of the “input” gates
  - A way to get output out of it
    - Pre-defined some gates as “output” gates, and when execution stopped, their values defined the output
  - How do we go from input to output?
    - Execution model: deterministic (non-random)
    - For any gate where all of its incoming wires had defined values, its value becomes the result of applying a particular operation to the values its incoming wires
  - Representation
    - Wires, gates, gates have labels (AON: AND/OR/NOT/INPUT/OUTPUT; NAND:NAND/INPUT/OUTPUT)

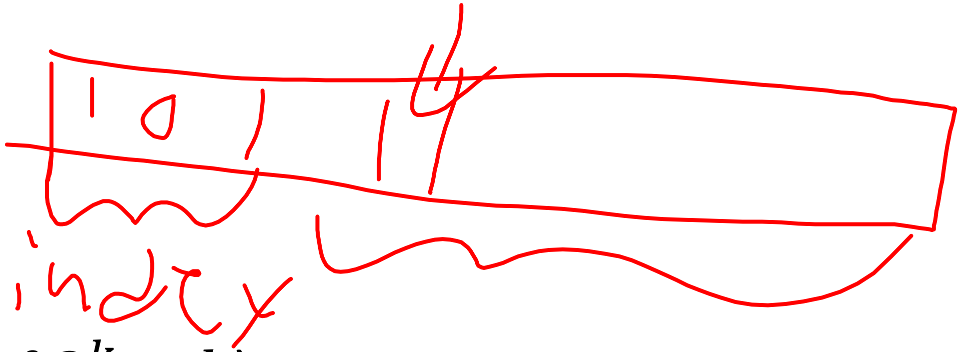
Showing AON=NAND

- Anything I can do with AON I can also do with NAND
  - Vice-versa
- Two models of computing are “equivalent” if every function you can implement with one of them you can also implement with the other
- Find a way of converting instances of one model into instances of the other such that we compute the same function
- To go from AON to NAND:
  - Do AND with NANDs
  - Do OR with NANDs
  - NOT with NANDs
- To go from NAND to AON:
  - Do NAND with ANDs/Ors/NOTs

Syntactic Sugar

- What is it?
  - Adding in capability of naming subroutines to make your model more human-readable
- Why use it?
  - Readability
    - Similar to relationship between assembly named memory locations vs. explicit
  - Succinctness b/c you can reuse “code”

$Lookup_k$ and Add_n



- What does the subscript mean?
  - What the size of the input it
  - k is the number of bits in the index (total of $2^k + k$)
  - n is the number of bits in each number we're adding together (total of $2n$)

$Compare_k: \{0,1\}^{2k} \rightarrow \{0,1\}$

- Using only NAND gates, write a function to compare two-bit integers
  - $Compare_2(a_1, a_0, b_1, b_0) = 1$ if integer $a_1 a_0 >$ integer $b_1 b_0$
  - $Compare_2(1100) = 1$
  - $Compare_2(0101) = 0$
- $Compare_1(a, b) =$

a	b	$c_1(a,b)$
0	0	0
0	1	0
1	0	1
1	1	0

$$Compare_1(a, b) = \neg(a \rightarrow b) \equiv a \wedge \neg b$$

$Compare_1(a, b)$:

```
notb= NAND(b,b)
anb = NAND(a,notb)
notanb = NAND(anb, anb)
```

$Compare_k(s_a, s_b)$:

eq0 = NOT(XOR($s_a[0]$, $s_b[0]$)) # 3 gates

compkm1 = $Compare_{k-1}(s_a[1:], s_b[1:])$ # C_{k-1}

comp1 = $Compare_1(S_a[0], S_b[0])$ # 3

return IF(eq0, compkm1, comp1) # 4 gates

If a more significant bit differs between the two, then we know the answer, otherwise we need to check more bits

$$C_k = C_{k-1} + 10$$
$$C_1 = 3$$

- $k = 1$
  - 3 gates
- $k = 2$
  - 13 gates
- $k = 3$
  - 23 gates
- $k = 4$
  - 33 gates
- …
- In general for k
  - $3 + 10(k - 1)$

Induction to show: $C_k \leq 3 + 10(k-1)$

- Base case: $k = 1$
  - $C_1 \leq 3 + 10(1-1)$
  - $3 \leq 3$
- Inductive step: to show if $C_k \leq 3 + 10(k-1)$ then $C_{k+1} \leq 3 + 10((k+1)-1)$

$$C_{k+1} = C_k + 10$$

$C_k \leq 3 + 10(k-1)$

$C_{k+1} \leq 3 + 10(k-1) + 10$

$C_{k+1} \leq 3 + 10k$

$C_{k+1} \leq 3 + 10((k+1)-1)$