Lecture 10: Basics of Quantum Computing

[In fact, let's start with basics of classical computing]

[Quantum computing is most clearly & naturally done in the "circuit model" we've been using, so let's do our classical computing in this model too to have clear comparison.]

Classical Circuit C:

[often focus on $m=1$ output bit, for simplicity.]

Computes a fcn. $F: \{0,1\}^n \to \{0,1\}^m$

[E.g.: take an n-bit #, want to output 0/1 ($m=1$) depending on if it's prime or not.]

Focus on efficiency: # of gates, and how it scales with n

[Could consider other measures too, but let's keep it simple.]

Gates "$\approx$" steps/time.

[[Certainly steps/time $\leq$ Gates.
Possible that time $\ll$ gates if you can parallelize effectively. Important in practice, but let's ignore for now.]]

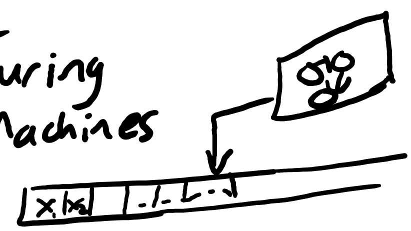
[[You might be more used to doing your classical computing in, say...]]

python:

```
def C(x):

    return y
```

Turing Machines



fact/thm: Given python code computing F on length-n inputs in T "time steps", can (easily) produce circuit computing F with $\leq \text{const}_{\text{python}} \cdot T \cdot \log T$ gates

[[main term T. $\log T \leq 100$ in life, $\text{const}_{\text{python}} \leq 100$? Diff. constant for TMs, Java, etc., but indep. of T, n, etc.]]

(Claude) <u>Shannon's Theorems, '37</u> (master's thesis)

- Every $F: \{0,1\}^n \to \{0,1\}$ can be computed by an AND/OR/NOT circuit with $\leq 2^n$ gates ($\lesssim 2^n/n$ in fact)

[Not great. Unphysically large for $n \geq$ few hundred]

- Almost all such F <u>need</u> $\gtrsim \frac{2^n}{n}$ gates

[So almost all fcns are effectively uncomputable in real life.
Fortunately, we care about computing specific, interesting functions. And sometimes (not always) they can be computed efficiently/physically/in P:
$\propto n$ or n^2 or n^3 gates...]

Aside: Probabilistic Computing (cf. Lecture 2)

- Adds a $\boxed{\text{FLIP}}$— gate

[[No inputs, 1 output.]]

- output $\begin{cases} 0 & \text{with prob. } 1/2 \\ 1 & \qquad\qquad 1/2 \end{cases}$

Prob'ic circuit C "computes" F if

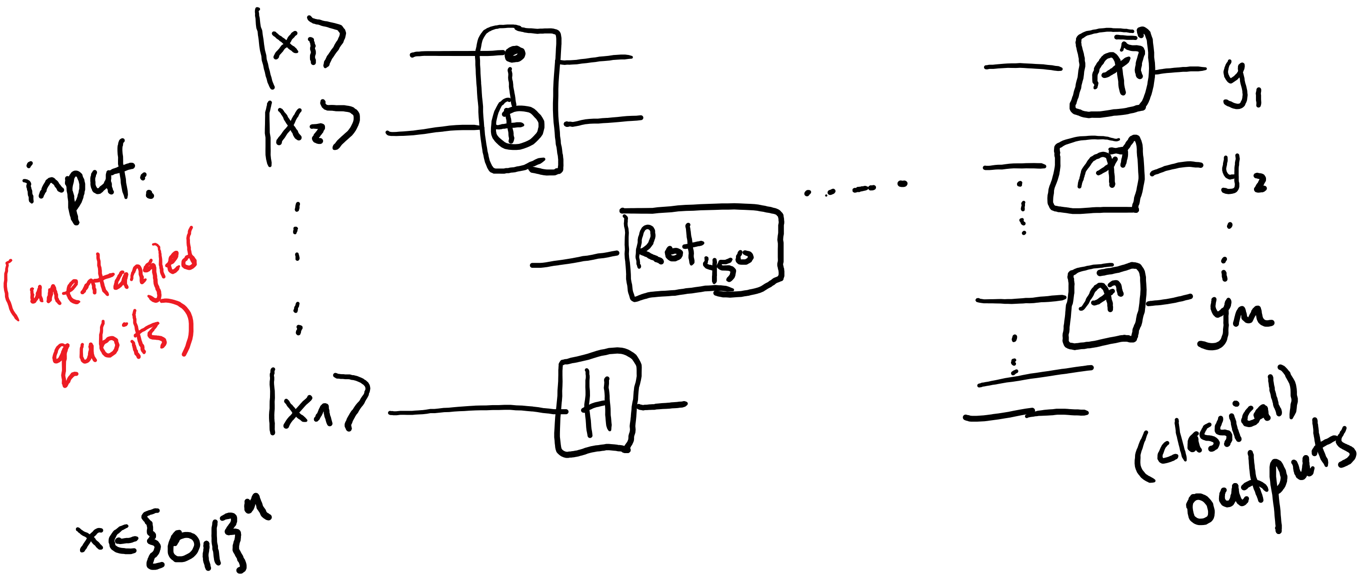
for <u>all</u> $x \in \{0,1\}^n$,

$$\Pr_{\text{flips}}[C(x) \neq F(x)] \text{ is "small"; say } \leq 1\%$$

[[You explored all this, & 0-sided, 1-sided, 2-sided error, in HW #1, Problem #5.]]

[[Recall from Lec. 2: believed to not help much for efficiency. <u>Not</u> believed that there are F computable in P with randomness but not in P deterministically.]]

[finally!] Quantum Computing



Computes $F: \{0,1\}^n \to \{0,1\}^m$ if...

<same as in probabilistic computing>

[Rem: WLOG all measurements at end, by "Principle of Deferred Measurement": HW5, #2.]

Q1: $\exists$? F which quantum computers can compute much more efficiently than classical comps?

[A1: seemingly yes, eg. Factoring. Half-dozen lectures from now.]

Q2: Reverse question! [A2: No. Today we'll see that Q.C.'s are at least as powerful as classical comps.]

Q3: Does the exact quantum gate set matter?

[A3: Not really, but a little subtler than classical case. Next lecture....]

Q.C.'s $\geq$ Classical Comps?

Can a Q.C. even compute $x_1, x_2 \to$ AND $\to$?

[In fact, not directly, no! Recall...]

Q. gates are unitary: $UU^\dagger = I$

$$\Rightarrow U^\dagger = U^{-1}$$

U is invertible / reversible

(In fact, almost all q. gates we've seen — NOT, Z, H, CNOT, ... are their own inverse. Rot_θ gates an exception.)

AND: not reversible: $00, 01, 10 \longmapsto 1.$

"erases information" $\xleftarrow{?}$

[Same with OR. On the other hand...]

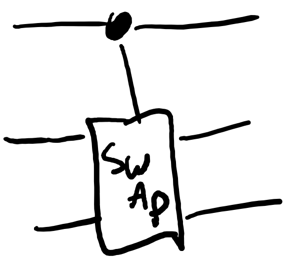
NOT: is reversible [In fact, it is a quantum gate.]

[Reversible computation was a topic studied by physicists in '60s, '70s, ... independent of quantum. Key early figure: Ralf Landauer @IBM. Physicists tried to understand the theoretical min. energy needed to do a "step" of computation. Landauer saw that it wasn't computation that req'd energy per se — it was "erasing info." Classical laws of physics are time-reversible. In principle, a reversible computation — like a NOT (or a CNOT) can be done in a "closed system" w/ no energy loss. A non-reversible op., like AND, "erases info" → "loses entropy" → by 2nd law of thermodynamics it must go somewhere → can't be a "closed system" → must leak, say, heat. Physicists thought: if you can make computation fully reducible, in principle it requires no min. energy. In practice, '70s & '80s saw very energy-efficient non-reversible computers, so idea ended up unnecessary. Apparently it's making a comeback these days, though.]

[Key names in reversible computing: Yves Lecerf (French-Syrian ethnologist!!), Fredkin, Toffoli, Feynman, Bennett, Huffman, Lorens...]

Useful reversible gate: CSWAP

[[Controlled-SWAP; aka "Fredkin gate"; HW#3, #1]]



[[As usual: if control bit is 0, do nothing, pass all bits thru. If control bit is 1, pass it thru & swap other two bits.]]

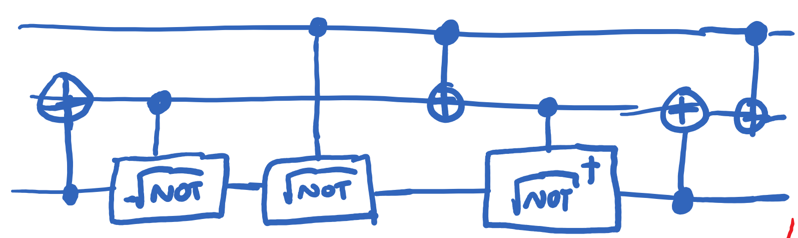
$|000\rangle \mapsto |000\rangle$, ... $110 \mapsto 101$, $111 \mapsto 111$

[[Easy to see it's reversible. In fact, easy to see it's...]] <u>self-inverse</u>.

8x8 permutation mtx, hence unitary matrix.

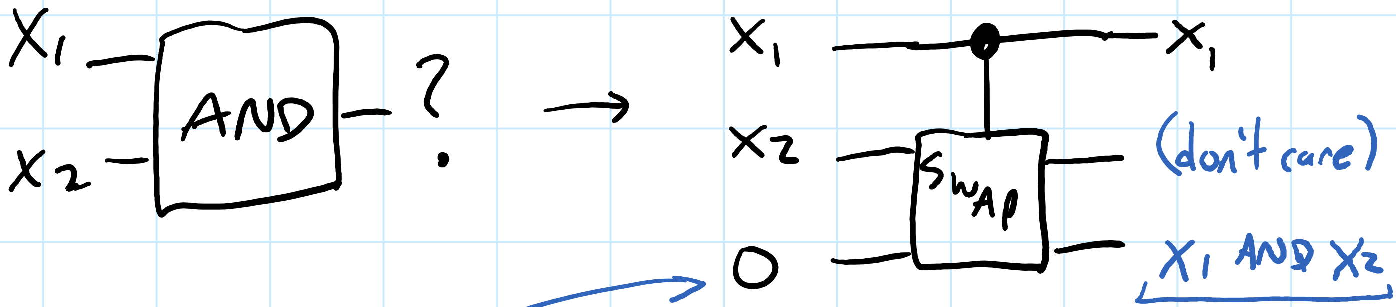
(So, like NOT, CNOT, and SWAP, can think of it as a reversible classical gate, and also as a 3-qubit quantum gate. We'll assume we can physically build it.

Here's one way, assuming you can build 2-qubit gates:



)

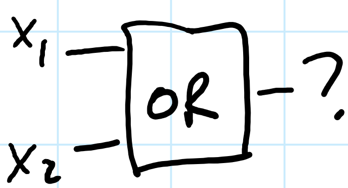
[Now check this out:]



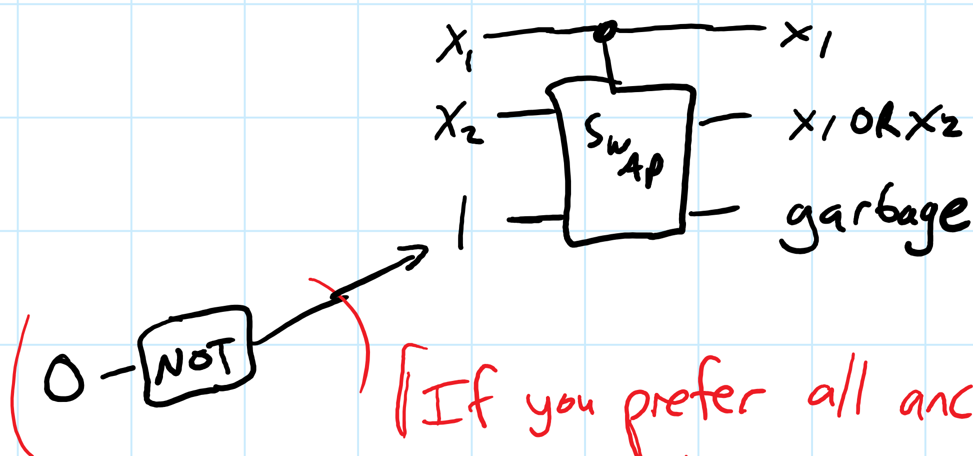
[certainly, if $x_1 = 0$, it's 0;
if $x_1 = 1$, it's 1 iff $x_2 = 1$]

"ancilla"
[just a "catalyst" you plug in]

(don't care): "garbage"

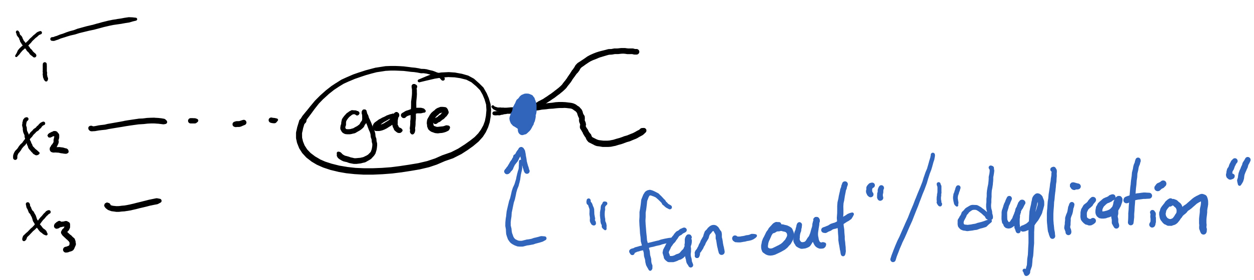


[Well, by de Morgan's laws, can be built from AND & NOT, so who cares? But anyway, it's...]



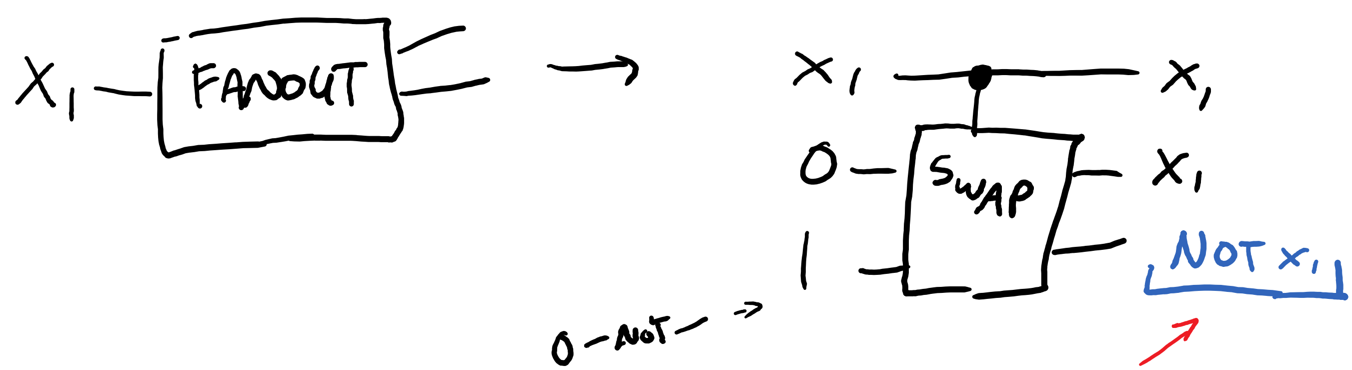
[If you prefer all ancillas be 0, as is traditional.]

(Are we done, for simulating classical circuits by quantum? No! One more "gate" that's rarely mentioned....)



(Traditionally you can just draw this in logic circuits, but it must actually correspond to a physical gadget.)

(Luckily, can also sim. w/ CSWAP & ancillas.)



[Garbage. Or can use it.]

[Fun: if you allow 0&1 ancillas, can use <u>only</u> CSWAPs to sim. AND, OR, NOT, FANOUT!]

Thm: Any classical circuit C computing $F: \{0,1\}^n \to \{0,1\}^m$ can be efficiently converted to a reversible (hence quantum) circuit

$$QC: \{0,1\}^{n+a} \longrightarrow \{0,1\}^{m+g}, \quad \boxed{n+a=m+g}$$

$$(\underbrace{X}_{n}, \underbrace{00\cdots0}_{a}) \longmapsto (\underbrace{f(x)}_{m}, \underbrace{\text{garbage}(x)}_{g})$$

(Can assume final output bits are in first m positions by doing SWAPs if nec.
If orig. circuit had T gates, QC has $O(T)$ gates & ancillas.

If inputs classical, final state has no superposition; measuring gives correct answer w/ 100% prob.)

Aside: $\boxed{\text{FLIP}}$ — ? ✓ $|0\rangle - \boxed{H} - \boxed{A?} -$ ← Can move meas. to end by Princ. of Def'd Meas.)

$(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$. 50-50 0/1.

$$\boxed{\therefore \text{ Q.C.} \geq \text{classical (even probabilistic) computing}}$$

Puzzle:

$\exists$ efficient classical <u>multiplication</u> circuits,

$$(P,Q) \longmapsto P\cdot Q.$$

Build the reversible version, C.

Then reverse it!

$$C^{rev}(P\cdot Q) = (P,Q)$$

$\rightarrow$ an efficient classical circuit for factoring?

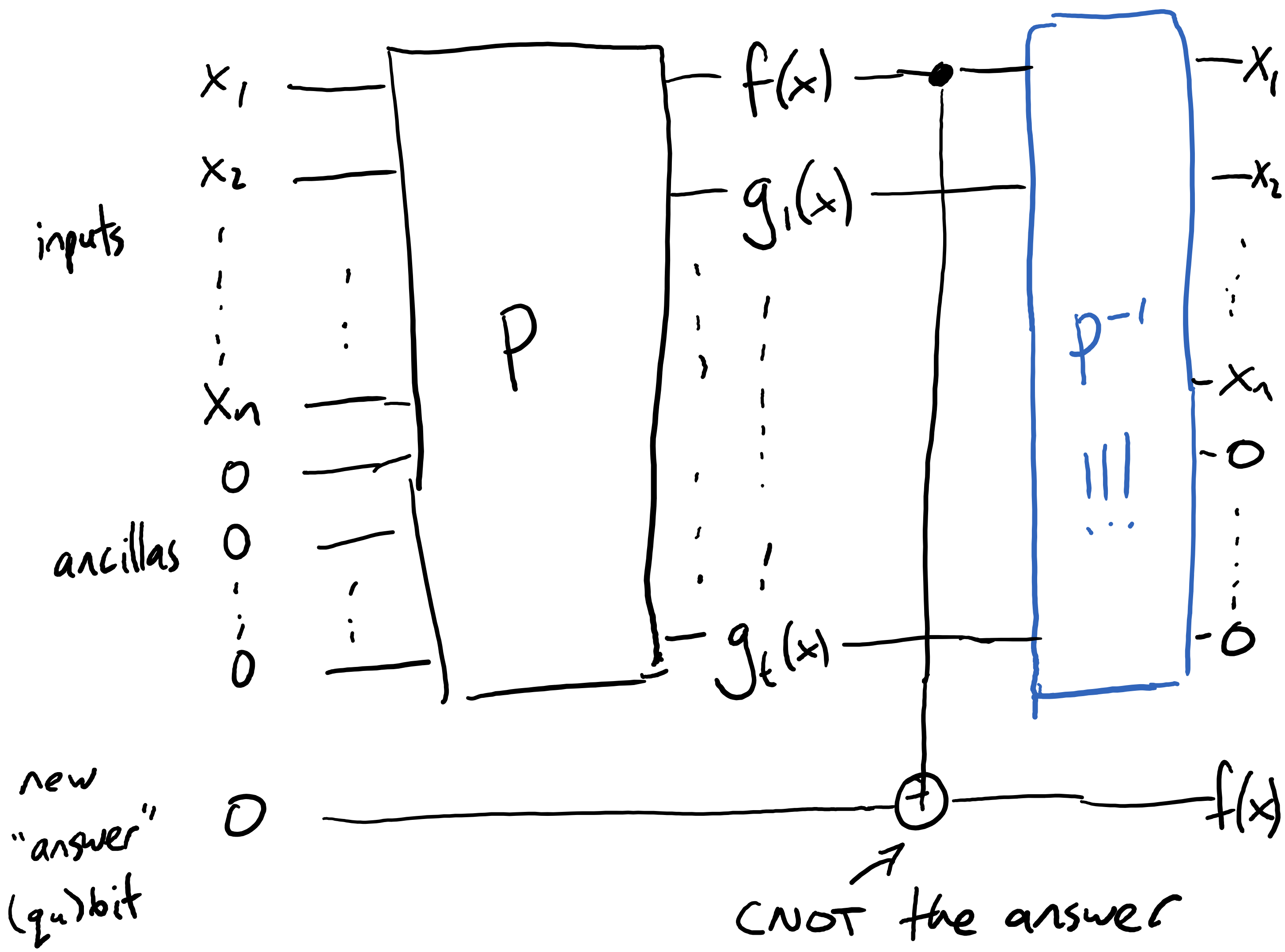
(Answer... no. Why not?)

Have to guess what <u>garbage</u> to put into C^{rev} to produce all-0 ancillas.

(In general, we don't like garbage. In fact, we'll later see that it's very bad to leave it around in q. computation. Can we get rid of it?)

Uncomputing Garbage [Bennett '80s]

Say $F: \{0,1\}^n \to \{0,1\}$ [1-bit output, for simplicity]
computed reversibly/quantumly:



$$|x\rangle \otimes |0^a\rangle \otimes |0\rangle \longmapsto |x\rangle \otimes |0^a\rangle \otimes |f(x)\rangle$$

(Works for multi-bit outputs, too. Usually we add SWAPs to get answer first, ancillas last.)

Small remark: If answer bit(s) initialized to $|y\rangle$ (instead of $|0\rangle$), CNOT converts to...

$$\underbrace{|y \oplus f(x)\rangle}_{\text{XOR (bitwise)}}$$

Overall: $$|x\rangle|y\rangle|0\cdots0\rangle \longmapsto |x\rangle|y\oplus f(x)\rangle|0\cdots0\rangle$$

(Puzzle: Now that garbage is uncomputed, can we reverse a multip. circuit to get a factoring circuit? No: the orig. input is part of the final output.)

Def: A quantum circuit <u>implements</u> $F : \{0,1\}^n \to \{0,1\}^m$ if it computes it in above garbage-free manner.

(So now we've seen that any classical circuit computing a fcn F can easily be converted to a quantum circuit <u>implementing</u> F.)

(Remark: The ancillas are really now like a "catalyst". You have to put them in to get things to work, but in the end they just come out as unentangled $|0\rangle$'s. This is <u>great</u> for subroutine purposes. Can string Q. circuits together, no worry about ancillas/garbage interfering. Can actually reuse ancillas.

Eventually, we might lazily not even mention them, writing

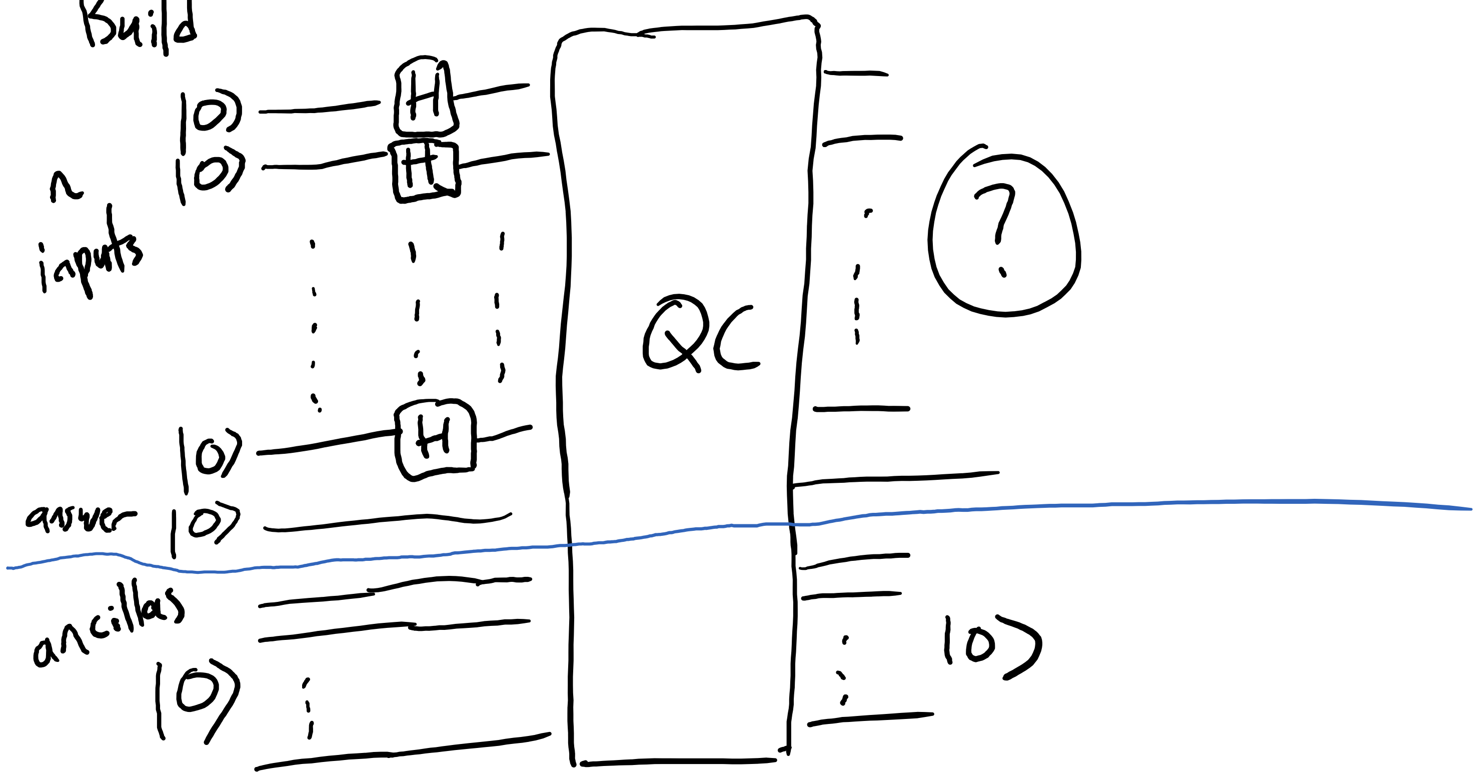
$$|x\rangle|y\rangle \mapsto |x\rangle|y \oplus f(x)\rangle.$$

But truth is, there's always ancilla $|0\rangle$'s attached on both sides.)

(Quick preview of the power(?) of quantum computing.)

Say $\boxed{QC}$ implements $F:\{0,1\}^n \to \{0,1\}$.

Build



Input state: $(|+\rangle \otimes \cdots \otimes |+\rangle) \otimes |0\rangle$

$= \left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\right) \otimes \cdots \otimes \left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\right) \otimes |0\rangle$

$= \left(\frac{1}{\sqrt{2}}\right)^n \sum_{x \in \{0,1\}^n} |x\rangle \otimes |0\rangle$ (!! "Unif. superpos. of all inputs")

$\therefore$ output is $\left(\frac{1}{\sqrt{2}}\right)^n \sum_{x \in \{0,1\}^n} |x\rangle \otimes |F(x)\rangle$

(all 2^n answers encoded in state!!! But how to use....?)