Yale ENGINEERING

https://caslab.io/

Quantum Circuit Reconstruction from Power Side-Channel Attacks on Quantum Computer Controllers

Computer Architecture and Security Lab (CASLAB)

https://caslab.io/

Ferhat Erata

ferhat.erata@yale.edu

Chuanqi Xu

chuanqi.xu@yale.edu

Ruzica Piskac

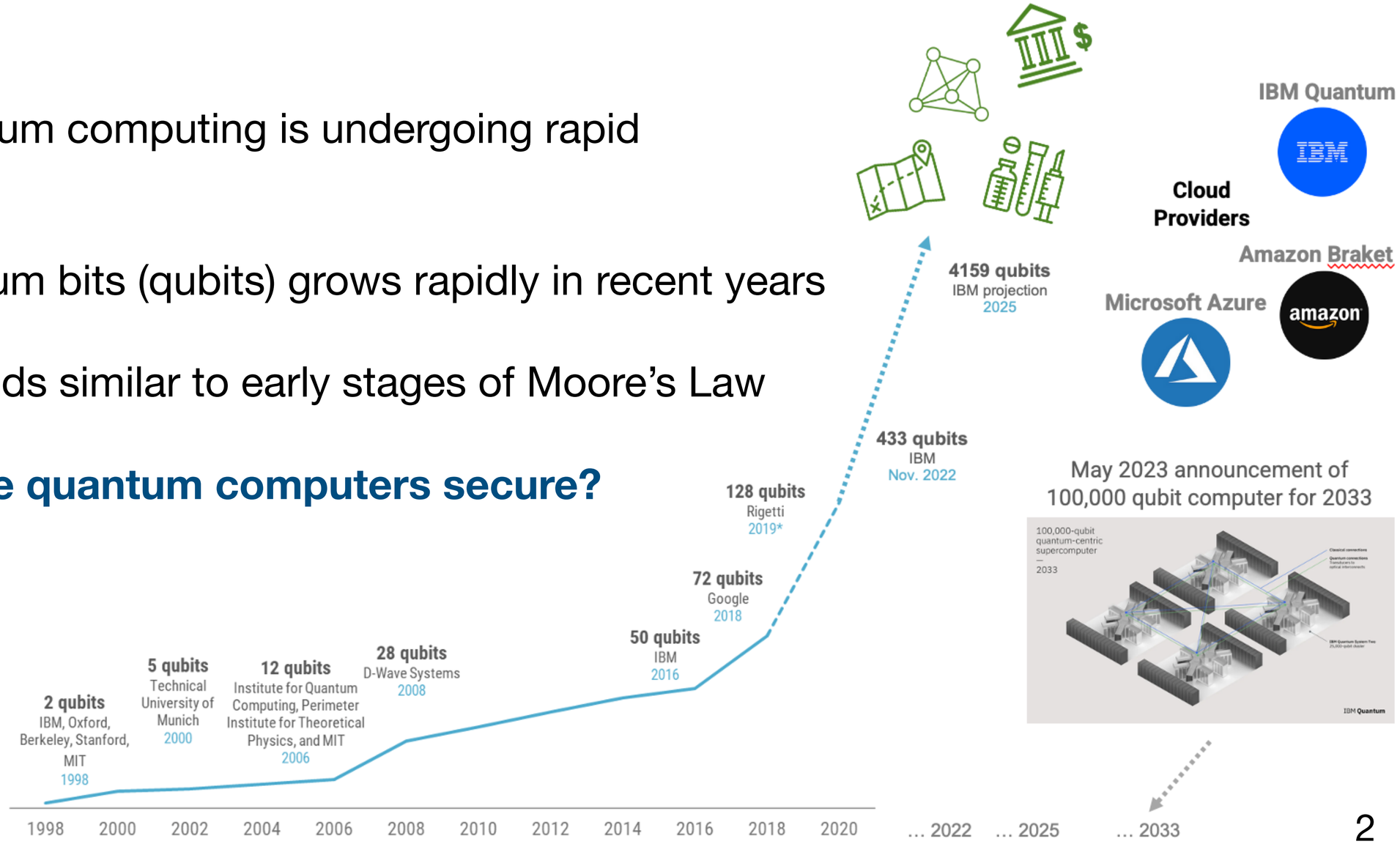
ruzica.piskac@yale.edu

Jakub Szefer

jakub.szefer@yale.edu

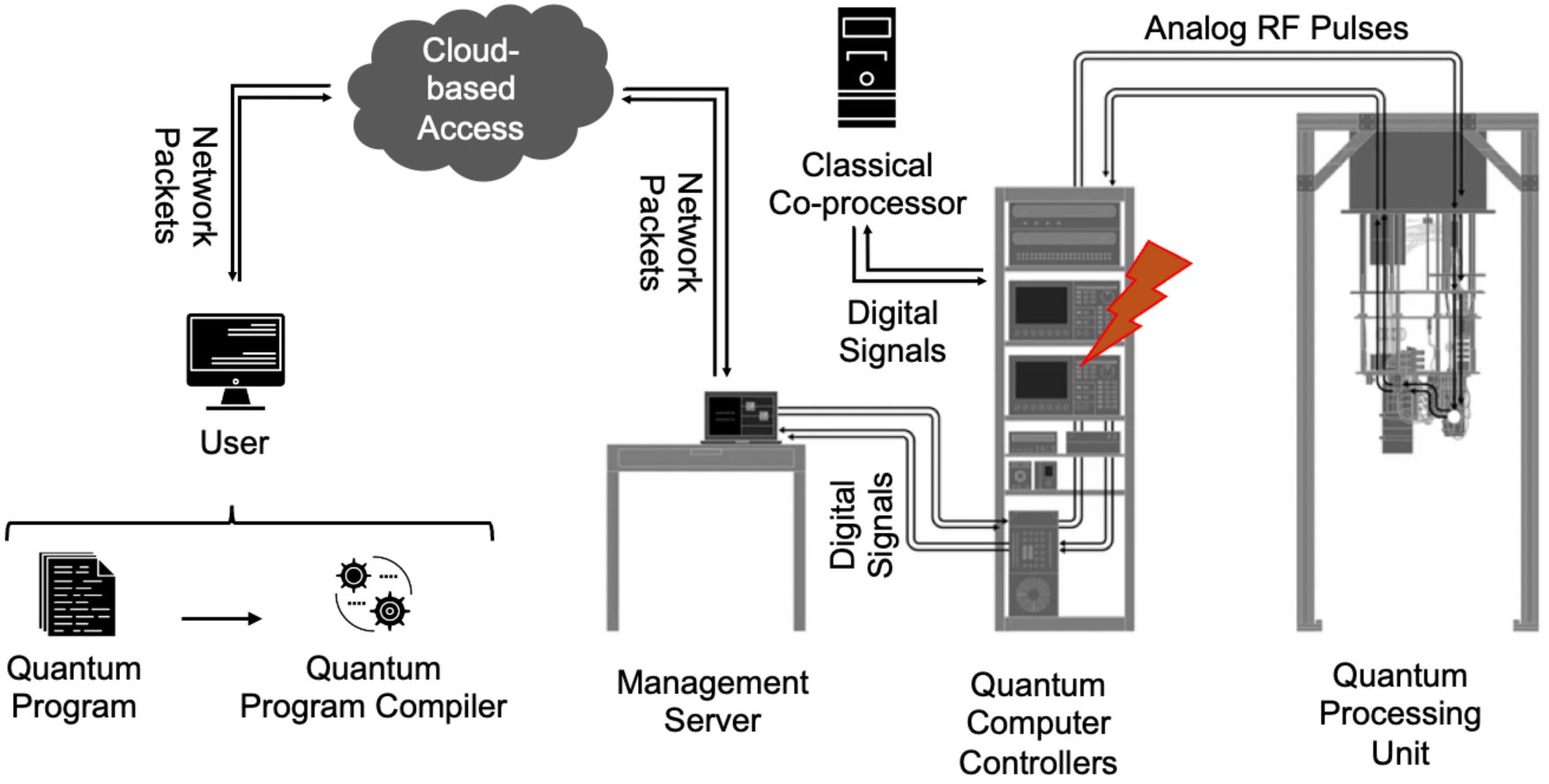
Why Research Security of Quantum Computing Systems?

- The field of quantum computing is undergoing rapid development
- Number of quantum bits (qubits) grows rapidly in recent years
- Development trends similar to early stages of Moore's Law
- **How do we make quantum computers secure?**



Quantum Computer Systems and their Attack Surface

We cannot clone quantum states, cannot directly “steal” information like in classical computers

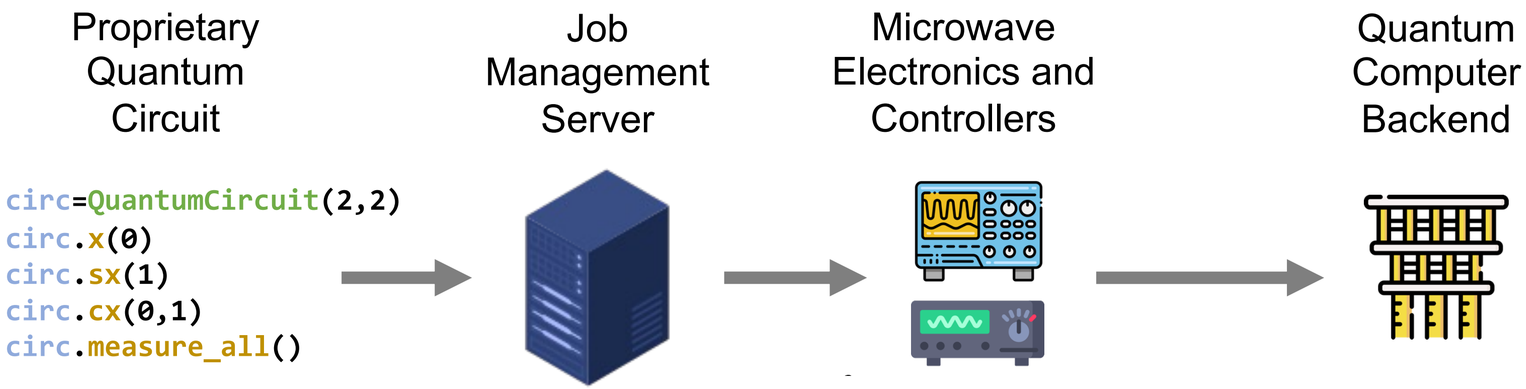


Quantum Computers require extensive control equipment which itself can be vulnerable to attacks

Control equipment not well studied from security perspective before

Power Side-Channel **Threat Model**

We consider attacker with access to collect side-channel information from the controller:



Power Side-Channel **Threat Model**

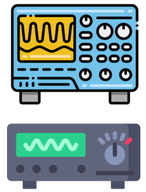
We consider attacker with access to collect side-channel information from the controller:

Proprietary Quantum Circuit

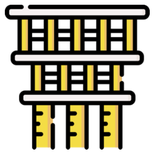
```
circ=QuantumCircuit(2,2)
circ.x(0)
circ.sx(1)
circ.cx(0,1)
circ.measure_all()
```

Job Management Server

Microwave Electronics and Controllers



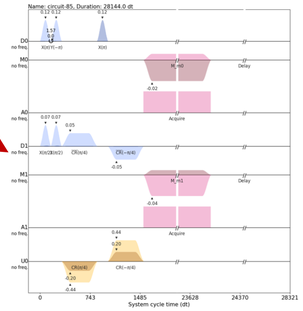
Quantum Computer Backend



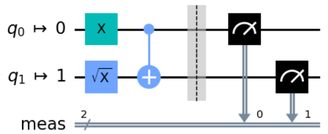
Leaks

Power Side-Channel Attacker

Recover Control Pulses



Obtain Transpiled Quantum Circuit

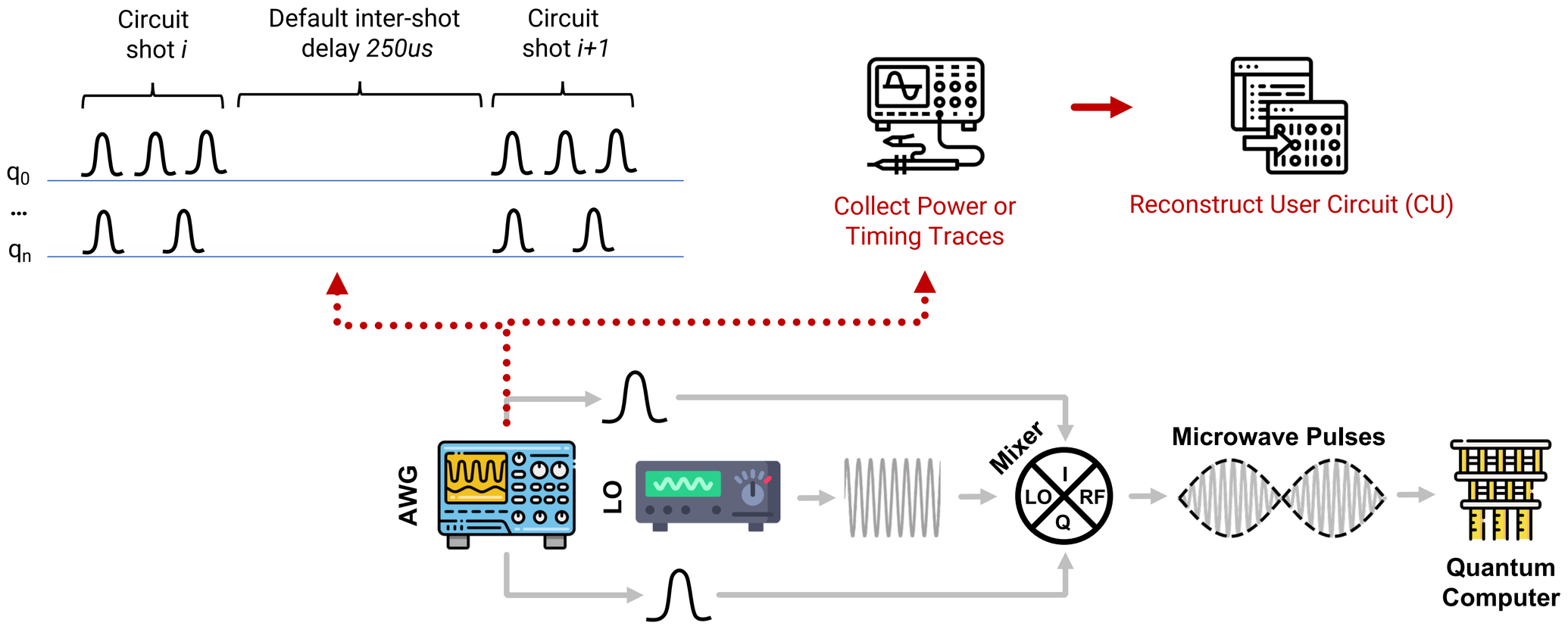


Retrieve User's Quantum Program

```
circ=QuantumCircuit(2,2)
circ.x(0)
circ.sx(1)
circ.cx(0,1)
```

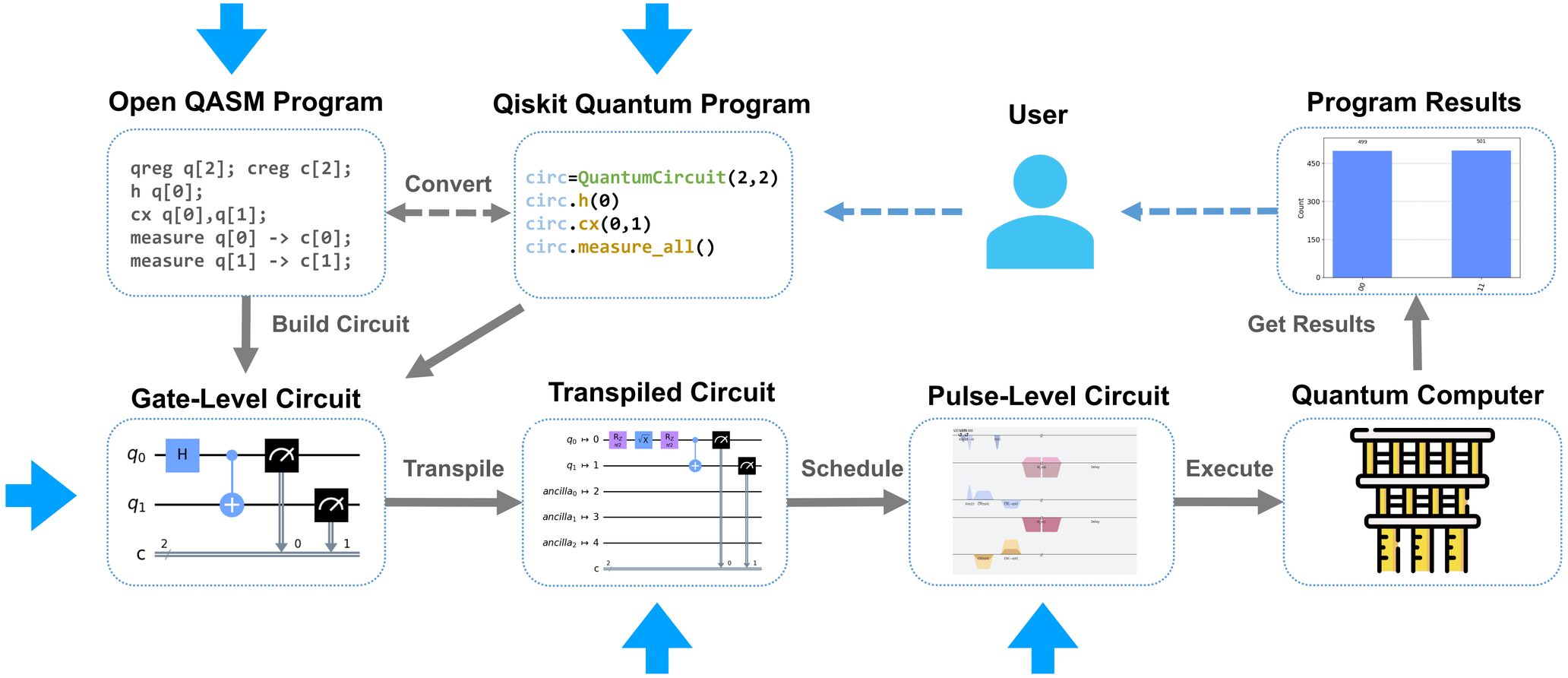
Recovery of Control Pulses

Key target for the side-channel attacks are the arbitrary waveform generators (AWGs)

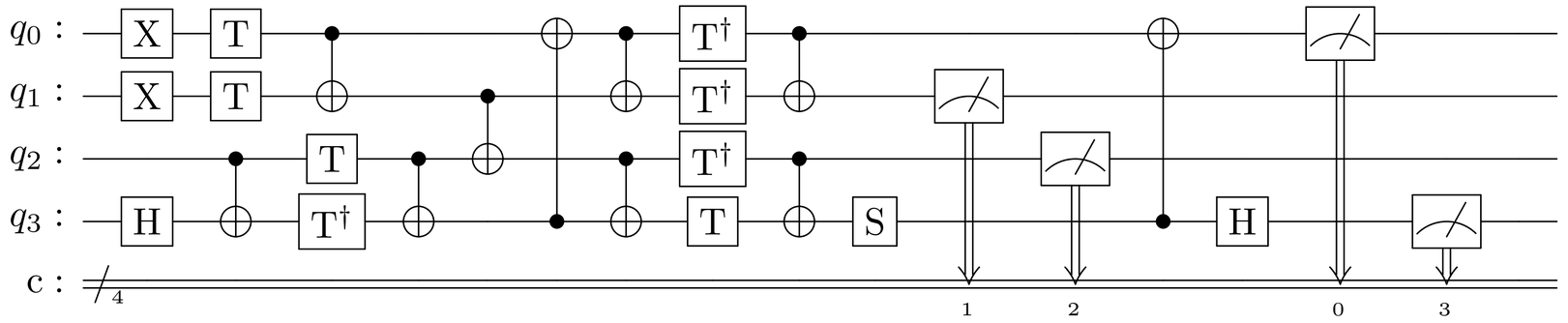


Process for **Running Quantum Circuits**

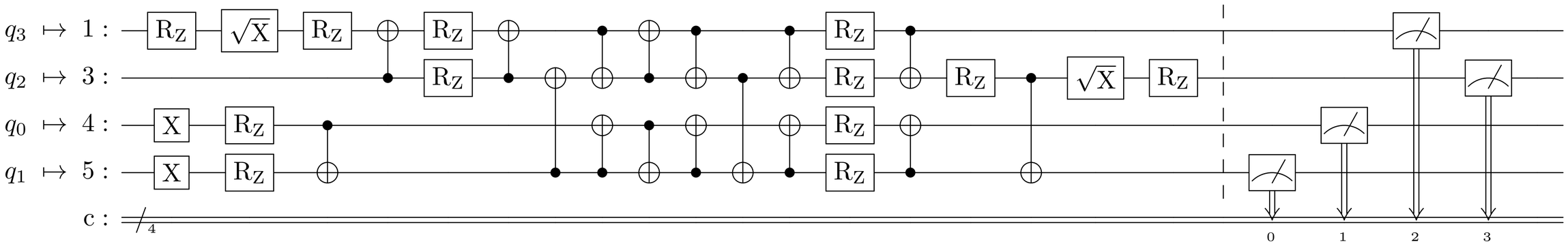
running quantum circuits on superconducting quantum computers in Qiskit.



example quantum circuits



a quantum adder circuit with width=4 (4 qubits) followed by measurement.

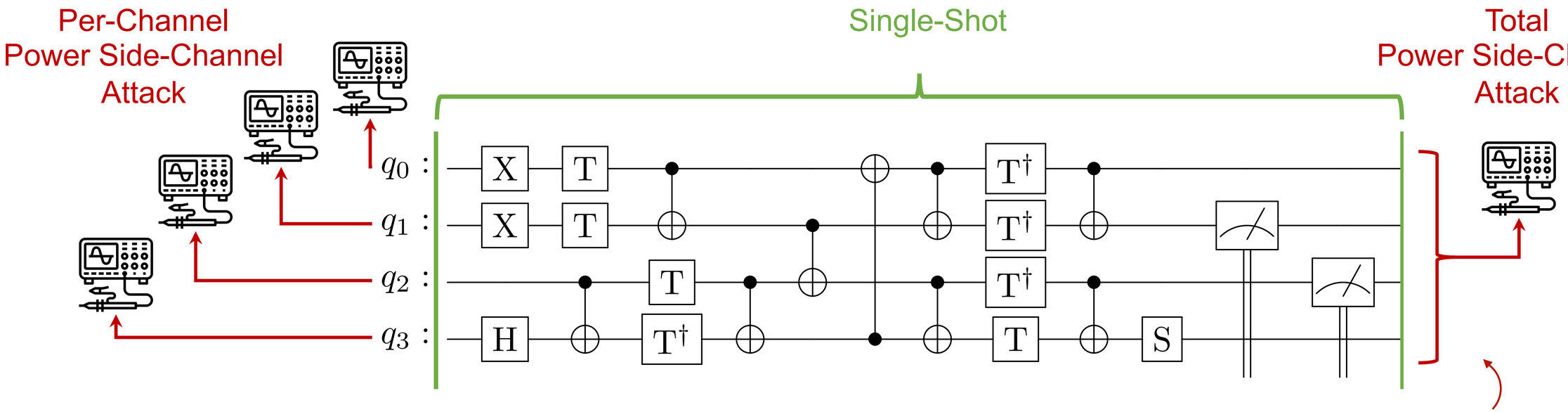


a victim adder circuit with width=4 **transpiled** with optimization level 3.

native gates used → **{CX, ID, RZ, SX, X}**

attack scenarios based on the **attacker's measurement capabilities.**

Power Side-Channels for Quantum Circuit Reconstruction



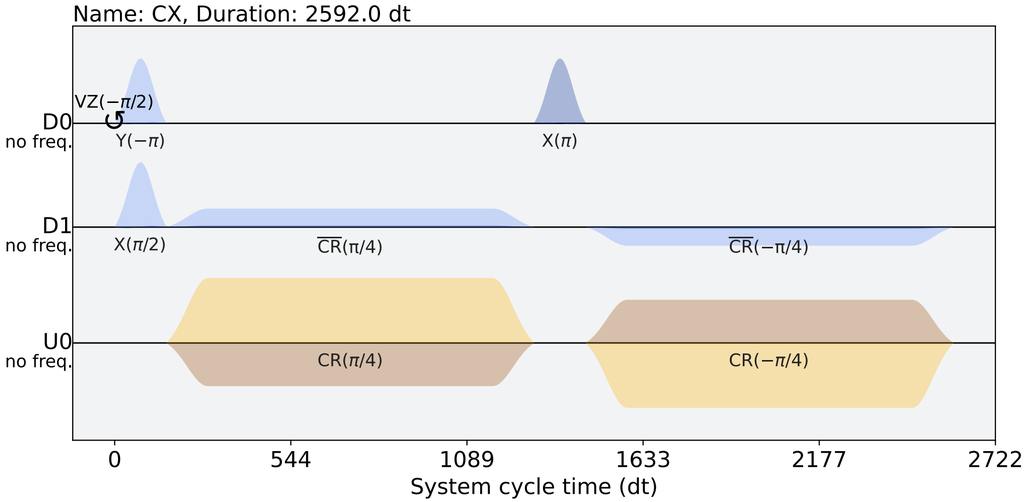
This restricts attacker to using only a **single total power trace** to reconstruct the quantum program

check out our previous work for a taxonomy of side-channel attacks!

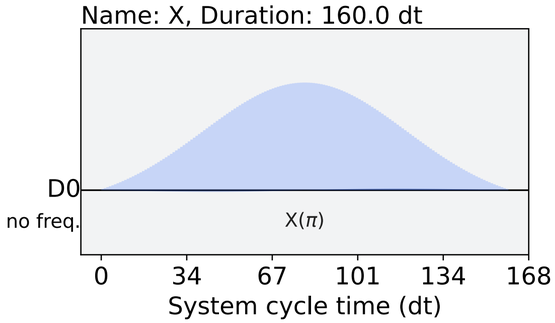
Chuanqi Xu, Ferhat Erata, Jakub Szefer (2023).
Exploration of Power Side-Channel Vulnerabilities in Quantum Computer Controllers (CCS).
https://dl.acm.org/doi/10.1145/3576915.3623118

Example SX, X, and CX gate control pulses

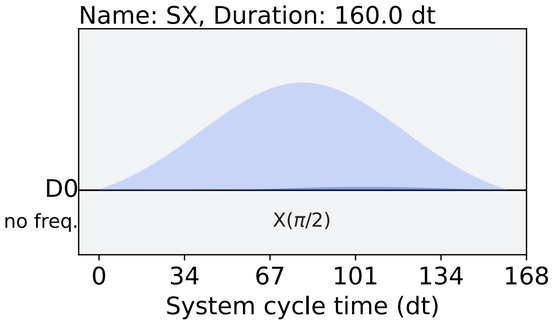
examples of control pulses for the SX, X, and CX gates



CX pulse



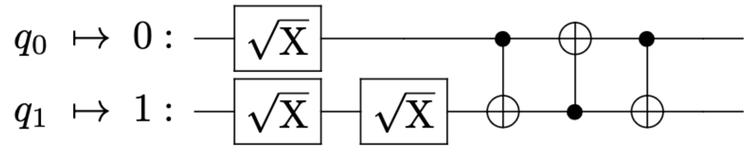
X pulse



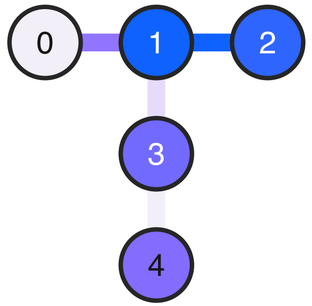
SX pulse

an example victim circuit

a simple circuit transpiled on a 5-qubit IBM Lima machine



Transpiled



ibmq_lima (Falcon r4T)

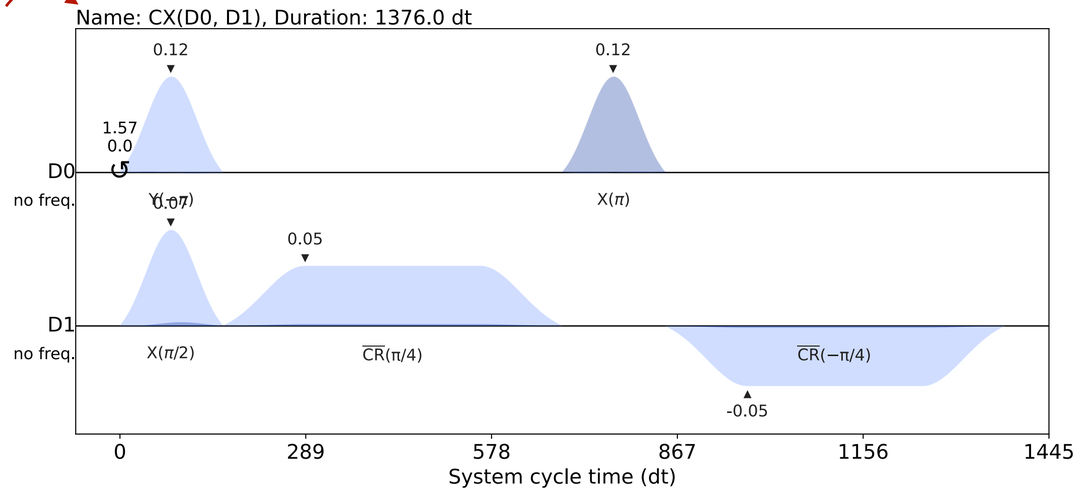
$\mathbf{dt_{start}}$	**Instruction**	`circ = QuantumCircuit(2)`
0	SX: d_1	`circ.sx(1)`
160	SX: d_0	`circ.sx(0)`
160	SX: d_1	`circ.sx(1)`
320	CX: d_0-d_1	`circ.cnot(0, 1)`
1696	CX: d_1-d_0	`circ.cnot(1, 0)`
3232	CX: d_0-d_1	`circ.cnot(0, 1)`

We aim to recover this table from the measured waveform from each drive channel for the **per-channel side-channel attacker**, or from the total power trace for the **total-power side-channel attacker**.

example pulse schedules

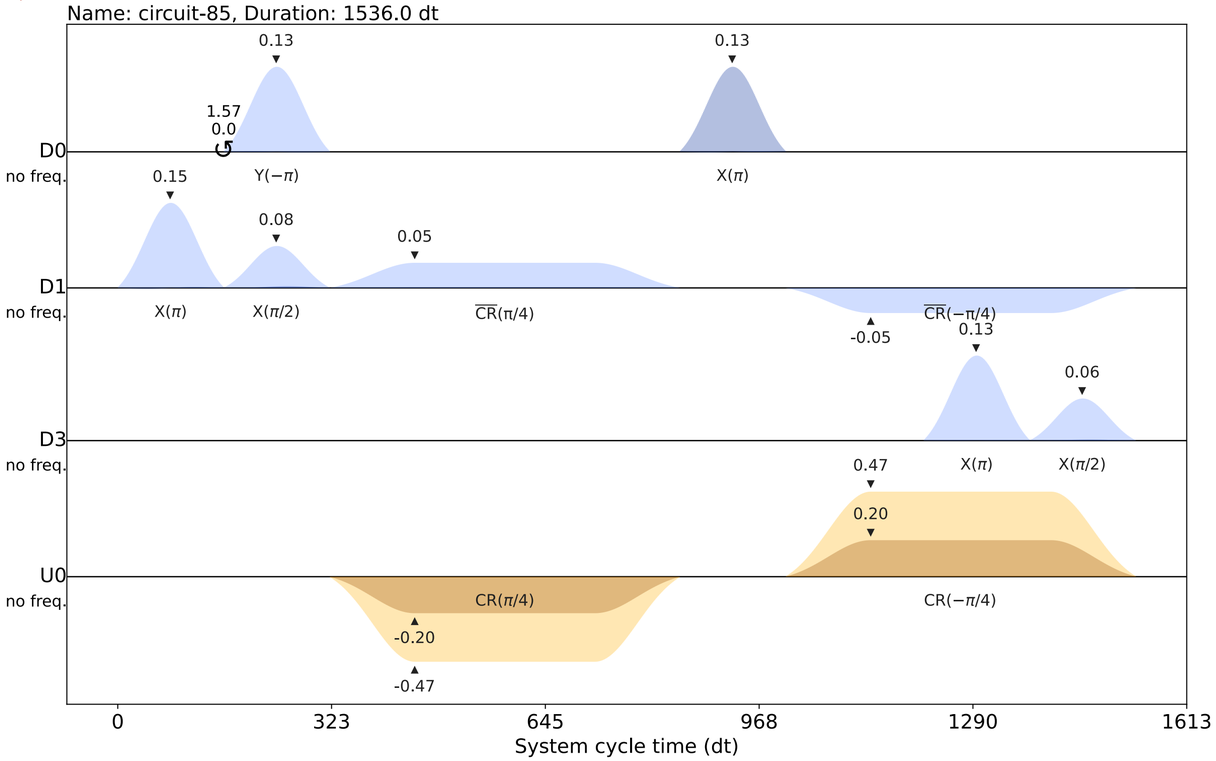
SX

CX



As seen from the figure, the first waveform on the drive channel d1 of CX is exactly the same pulse as in SX

Superimposed



You can see there are different waveforms superimposed on a same channel.

Step 1: formalize the circuit reconstruction problem

$$BG = \{\texttt{I}, \texttt{RZ}, \texttt{X}, \texttt{SX}, \texttt{CX}\}. \quad (1)$$

the set of basis gates of a quantum device D

$$C = \{drive_0, drive_1, \ldots, drive_{n-1}, control_0, control_1, \cdots, control_{m-1}\} \quad (2)$$

the set of drive and control channels

$$L = \{(gate, C') | gate \in BG, C' \subset C\} \quad (3)$$

L represents all basis gates with their channel information,

$$p_{l,c}(x) \begin{cases} \text{Not always } 0 & \text{if } x \in [0, d_l] \\ = 0 & \text{if } x \notin [0, d_l] \end{cases} \quad (4)$$

a basis pulse on the channel c

$$p_l(x) = \{p_{l,c}(x) | c \in l[C']\}. \quad (5)$$

one basis pulse may include pulses on several channels, such as CX gate

$$P_L = \{p_l(x) | l \in L\}. \quad (6)$$

basis pulse library

Step 1: formalize the circuit reconstruction problem

$$A_{P_L} = \{a_{l,t} \cdot p_l(x-t) | l \in L, a_{l,t} \in \{0,1\}, p_l(x) \in P_L\} \quad (7)$$

pulse-level circuit

$$Power_c[A_{P_L}](x) = \sum_{A_{P_L}} \mathrm{Re}^2[a_{l,t} \cdot p_{l,c}(x-t)] + \mathrm{Im}^2[a_{l,t} \cdot p_{l,c}(x-t)] \quad (8)$$

per-channel power traces

$$Total[A_{P_L}](x) = \sum_{c \in C} Power_c[A_{P_L}](x) \quad (9)$$

total power traces are directly the summation of per-channel power traces:

$$\{c_1 = c_2 \wedge [t_1, t_1+1, ..., t_1+d_{gate_1,c_1}] \cap [t_2, t_2+1, ..., t_2+d_{gate_2,c_2}] = \emptyset\} \vee c_1 \neq c_2 \implies \forall t_1 \text{ and } t_2 \in [t_1, t_1+1, ..., t_1+d_{gate_1,c_1}],\ a_{c,gate_1,t_1} + a_{c,gate_2,t_2} \in \{0,1\} \quad (10)$$

channel constraint

$$A_{P_L} = \arg\min_{A'_{P_L}} \sum_{c \in C} \left(d\left\{Power_c[A'_{P_L}](x), v_c(x)\right\}\right) \quad (11)$$

per-channel single trace attack

$$A_{P_L} = \arg\min_{A'_{P_L}} d\left\{Total[A'_{P_L}](x), v(x)\right\} \quad (12)$$

total power single trace attack

Step 2: how to **solve combinatorial optimization problem**

Candidate Solution 1: Model the problem in Linear Mixed Integer Real Arithmetic (LIRA)

$$3x_1 + 2x_2 \leq 5x_3 \vee 2x_1 - 2x_2 = 0$$

Single-Shot Total Power-Trace

Linear Mixed Integer Real Arithmetic (LIRA)

Z3 SMT Solver

Victim Circuit

Pros: we can encode arbitrary boolean combinations of Int and Real arithmetic

We need Single Objective Optimization

Does not Scale

Step 2': how to **solve combinatorial optimization** problem

Candidate Solution 2: Model the problem as Mixed Integer Linear Programming (MILP)

$$\begin{aligned} \min \quad & f_0(x) \\ \text{s.t.} \quad & f_i(x) \leq b_i, \quad i = 1, \ldots, m \end{aligned}$$

Single-Shot Total Power-Trace

Mixed Integer Linear Programming (MILP)

Gurobi and PuLP Solvers

Victim Circuit

Cons: arbitrary combination of boolean operations are **not** supported

Step 3: From LIRA to MILP: linearize distance function

Linearization of Absolute Valued Objective Function

$$d_1 : (v(x), Total_{A_P}(x)) \mapsto \|v(x), Total_{A_P}(x)\|_1 = \sum_{i=1}^{n} \left| v_i(x) - Total_{A_P}(x)_i \right|$$

Sum of Absolute Differences (SAD)

$$|x| = \begin{cases} -x, & \textbf{if } x < 0 \\ x, & \textbf{if } x \geq 0 \end{cases}.$$

If $|X|$ is the absolute value term in our objective function, two additional constraints are added to the linear program: $X \leq Z \wedge -X \leq Z$.

The $|X|$ term in the objective function is then replaced by Z, relaxing the original function into a collection of linear constraints.

Step 3: From LIRA to MILP: linearize logical conditions

Linearization of Logical Conditions over Binary variables

In MILP lingo, binary variables means decision variables that must take either the value 0 or the value 1, sometimes called 0/1 variables.

	Logical	Linearized
Not	$\neg x_1$	$1 - x_1$
And	$x_1 \wedge x_2$	$x_1 + x_2 = 2$
Or	$x_1 \vee x_2$	$x_1 + x_2 \geq 1$
Exclusive Or	$x_1 \oplus x_2$	$x_1 + x_2 = 1$
If-then	$x_1 \rightarrow x_2$	$x_2 \geq x_1$
If x_1 then $x_2 \vee x_3$	$x_1 \rightarrow x_2 \vee x_3$	$x_2 + x_3 \geq x_1$
If $x_2 \vee x_3$ then x_1	$x_2 \vee x_3 \rightarrow x_1$	$x_1 \geq x_2 \wedge x_1 \geq x_3$

Step 3: From LIRA to MILP: Encode Pseudoboolean constraints

Encoding Pseudoboolean constraints

In MILP lingo, binary variables means decision variables that must take either the value 0 or the value 1, sometimes called 0/1 variables.

	Linearized
At-least c of $x_1, x_2, \cdots, x_n$	$x_1 + x_2 + \cdots + x_i \geq c$
At-most c of $x_1, x_2, \cdots, x_n$	$x_1 + x_2 + \cdots + x_i \leq c$
Exactly c of $x_1, x_2, \cdots, x_n$	$x_1 + x_2 + \cdots + x_i = c$

Step 3: From LIRA to MILP: Linearize Disjunctive Constraints

Linearization of Disjunctive Constraints

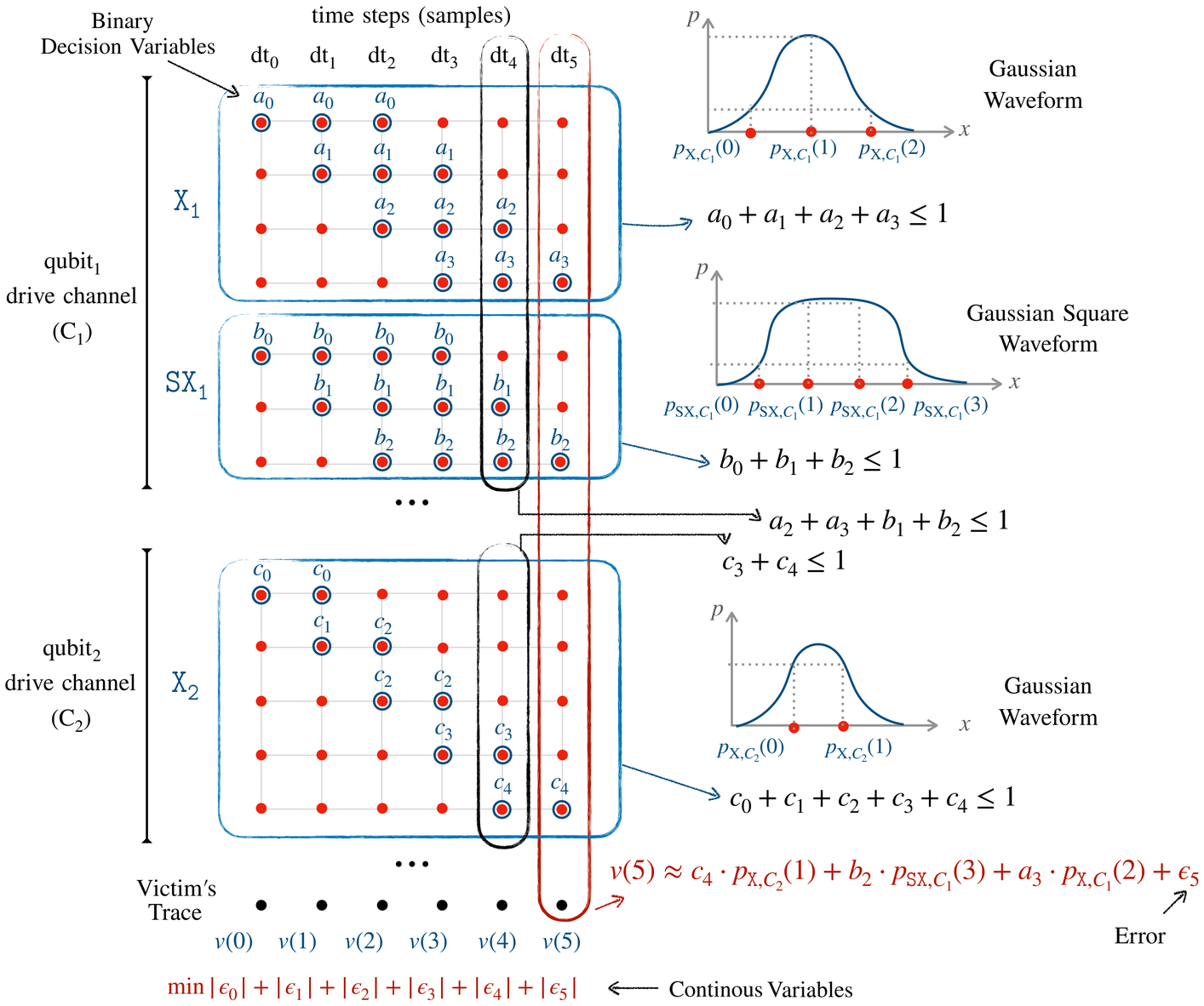
In order to encode the channel constraint given in Equation 10 using Big-M Reformulation

$$\sum_i a_i^1 x_i \leq b^1 \vee \sum_i a_i^2 x_i \leq b^2 \vee \sum_i a_i^3 x_i \leq b^3 \vee \cdots \vee \sum_i a_i^k x_i \leq b^k$$

$$\sum_i a_i^1 x_i \leq b^1 + M_1\left(1 - y^1\right) \wedge \sum_i a_i^2 x_i \leq b^2 + M_2\left(1 - y^2\right) \wedge \cdots$$

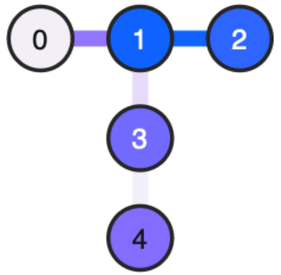
$$y^1 + y^2 + y^3 + \cdots + y^k \geq 1 \wedge y^1, y^2, y^3, \cdots, y^k \in \{0,1\}$$

MILP encoding method

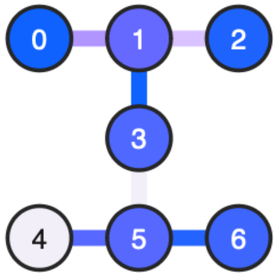


Evaluation Results

IBM Quantum computers used in the evaluation



(a) ibmq_lima (Falcon r4T)
{CX, ID, RZ, SX, X}



(b) ibm_lagos (Falcon r5.11H)
{CX, ID, RZ, SX, X}

(c) ibmq_manila (Falcon r5.11L)
{CX, ID, RZ, SX, X}

QASMBench benchmark suite version 1.4

Benchmark	Algorithm	Reference	Benchmark	Algorithm	Reference
deutsch	Hidden Subgroup	[CBSG17]	hs4	Hidden Subgroup	[JPK+14]
iswap	Logical Operation	[CBSG17]	bell	Logic Operation	[Dev22]
quantumwalks	Quantum Walk	[Micnd]	qft	Hidden Subgroup	[CBSG17]
grover	Search	[Age19]	variational	Quantum Sim.	[MRS+20]
ipea	Hidden Subgroup	[CBSG17]	vqe	Linear Equation	[JPK+14]
dnn	Machine Learn.	[SLM+21]	vqe_uccsd	Linear Equation	[JPK+14]
teleportation	Quantum Comm.	[Fed16]	basis_trotter	Quantum Sim.	[MRS+20]
qaoa_n3	Optimization	[Iosnd]	qec_sm	Error Correction	[CBSG17]
toffoli	Logical Operation	[JPK+14]	lpn	Machine Learn.	[Sam17]
linearsolver	Linear Equation	[BGKS05]	qec_en	Error Correction	[Sam17]
fredkin	Logical Operation	[JPK+14]	shor	Hidden Subgroup	[IBMnda]
wstate	Logical Operation	[CBSG17]	pea	Hidden Subgroup	[CBSG17]
basis_change	Quantum Sim.	[MRS+20]	error_cd3	Error Correction	[MNW+17]
qrng	Quantum Arith.	[TS21]	simons	Hidden Subgroup	[Age19]
cat_state	Logical Operation	[JPK+14]	qaoa_n6	Optimization	[Dev22]
inverseqft	Hidden Subgroup	[CBSG17]	vqe_uccsd	Linear Equation	[JPK+14]
adder	Quantum Arith.	[JPK+14]	hhl	Linear Equation	[IBMndb]

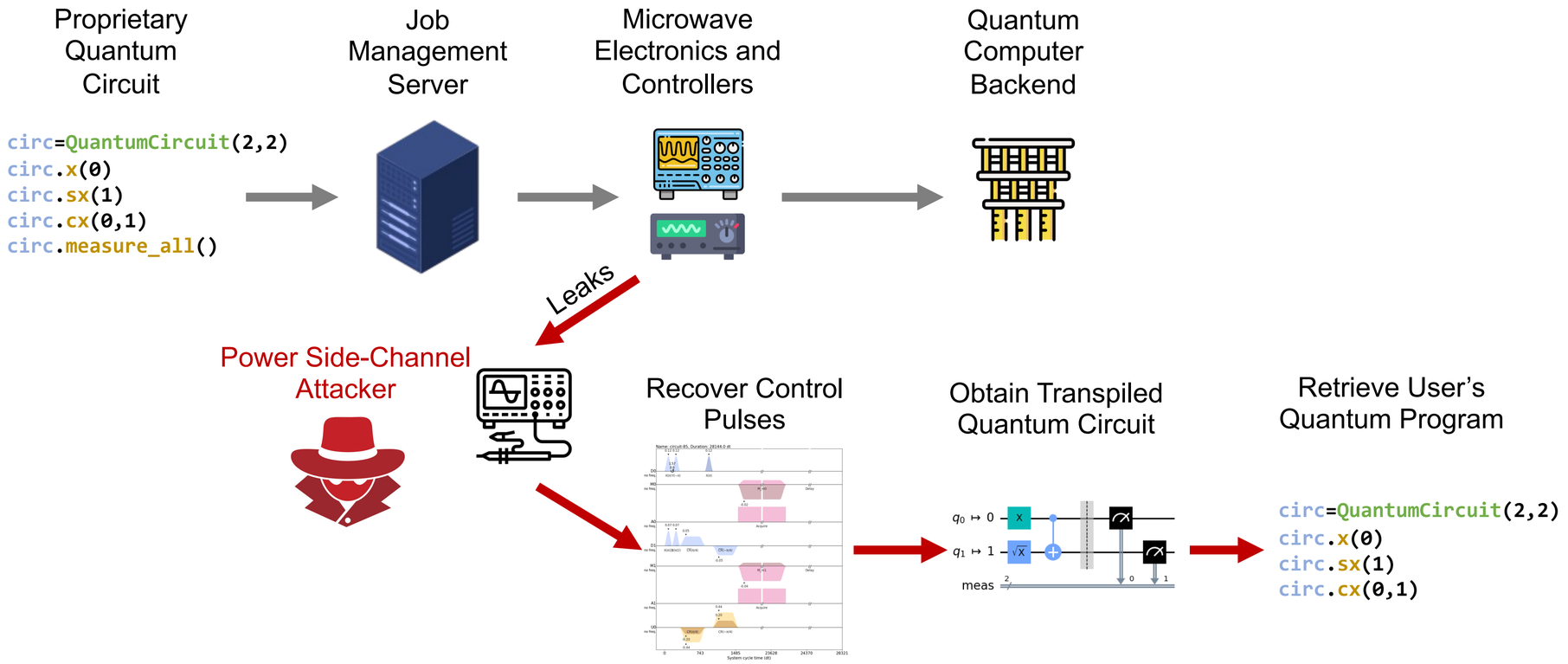
MILP encoding complexities of the benchmarks and pulse-level recovery results

Quantum Circuit	Total Qubits	Total Gates	Circ. Depth	Total dt	Real Vars	Integer Vars	Total Constrs.	Solver Time
deutsch	2	10	7	28432	58	6032	336	0.13s
dnn	2	306	17	31696	159	19080	756	0.51s
grover	2	15	12	30000	106	11872	536	0.30s
iswap	2	14	9	30096	111	12432	556	0.37s
quantumwalks	2	38	17	31600	156	39000	874	0.51s
basis_change	3	85	45	43440	526	117824	2328	4.58s
fredkin	3	31	32	48720	691	136818	2962	6.72s
linearsolver	3	26	14	32688	190	23370	883	0.99s
qaoa_n3	3	35	20	14464	412	64272	1804	4.40s
teleportation	3	12	9	29648	95	10545	491	0.24s
toffoli	3	24	20	39312	397	61535	1743	3.15s
wstate	3	47	34	50896	759	157872	3244	6.73s
adder	4	33	28	57808	975	282750	4190	13.3s
basis_trotter	4	2353	469	221248	1361	1503905	2722	150.2s
bell	4	53	18	33680	221	39780	1064	2.27s
cat_state	4	6	7	32688	190	32870	933	1.12s
hs4	4	28	12	31600	156	26520	794	2.0s
inverseqft	4	30	22	26928	10	200	60	0.04s
qft	4	50	40	57072	952	273224	4095	17.97s
qrng	4	12	4	26768	5	50	30	0.01s
variational	4	58	30	41200	456	97128	2037	3.92s
vqe	4	78	24	14624	346	161928	1852	5.47s
vqe_uccsd	4	238	198	104128	1876	1130000	3752	53.91
error_cd3	5	249	161	96672	4017	584577	6042	98.43
lpn	5	17	9	30352	117	23634	670	0.36s
pea	5	126	64	73744	1473	720297	6381	40.34s
qec_en	5	52	23	49072	702	262548	3182	12.08s
qec_sm	5	8	19	60000	600	189600	2716	19.11s
qaoa_n6	6	408	109	70464	2202	1212456	4404	162.27
simon	6	65	54	62976	1186	849176	5460	40.95s
vqe_uccsd	6	3865	2883	2009248	62800	32872200	125600	254s
hhl	7	565	380	265088	8288	5216264	16576	136.1s

Total # of gates increases complexity

Circuit Depth increases complexity

Quantum Circuit Reconstruction from Power Side-Channel Attacks on Quantum Computer Controllers



Ferhat Erata
ferhat.erata@yale.edu

Chuanqi Xu
chuanqi.xu@yale.edu

Ruzica Piskac
ruzica.piskac@yale.edu

Jakub Szefer
jakub.szefer@yale.edu