Compiling for a Bicycle Architecture

Ali Javadi-Abhari
Principal Research Scientist
IBM Quantum

github.com/qiskit-community/bicycle-architecture-compiler

Introduction

Scaling beyond the surface code

Surface code qubit

1 logical qubit requires $2(d+1)^2$ physical qubits.

Bivariate bicycle codes

Gross ($d = 12$): 12 logical qubits per 288 physical qubits.
Two-gross ($d = 18$): 12 logical qubits per 576 physical qubits.

Surface codes require 288 ($d = 11$) and 648 ($d = 17$) physical qubits.

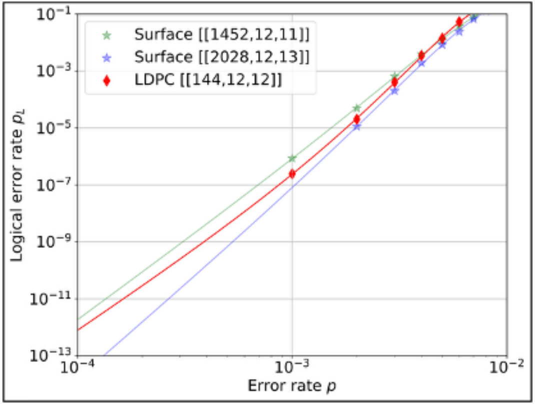


Figure: Compared to surface code, the qLDPC code requires about 10x fewer physical qubits at comparable k and d [Bravyi et al., Nature 627 (2024)].

The gross code and friends

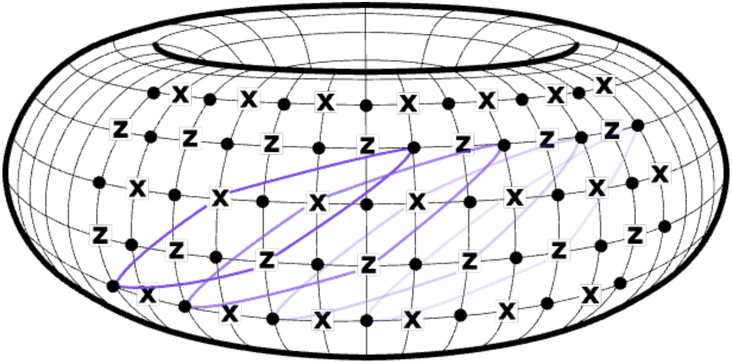


Figure: The gross code is visually represented by a torus due to its Tanner graph embedding plus two long-range connections.

HW demo: 6-way couplers

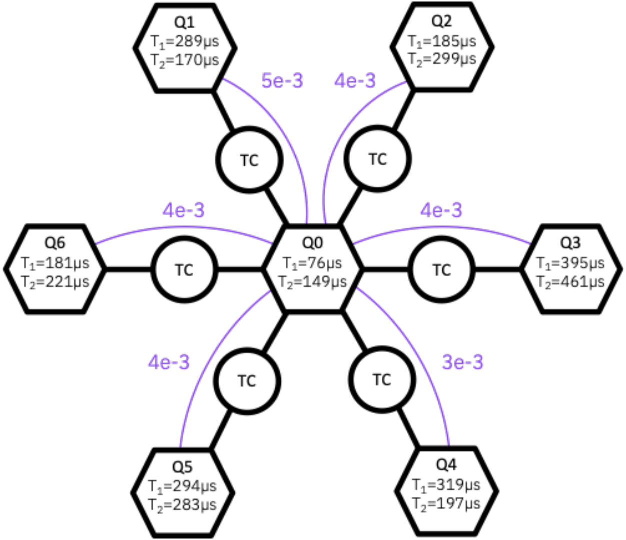


Figure: Degree-6 connectivity without loss of gate fidelity (purple) using tunable couplers (TC)

IBM **Quantum**

HW demo: Long-range gates

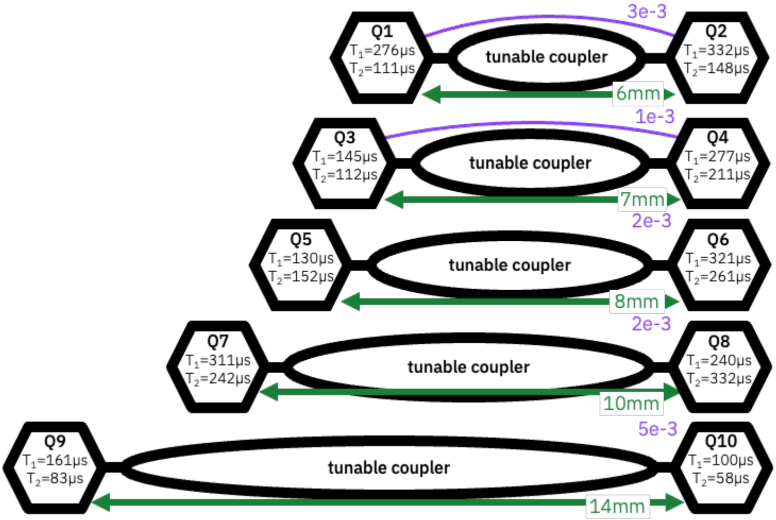
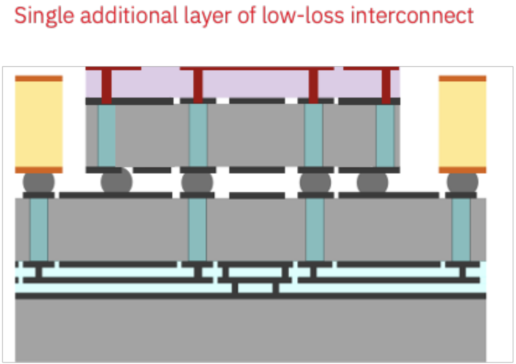


Figure: Demonstration of long-range gates required for the gross code.

HW demo: Non-planar interconnect



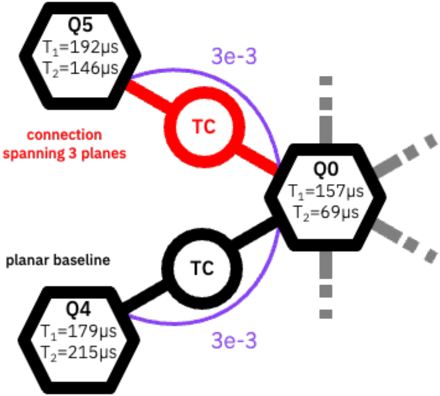
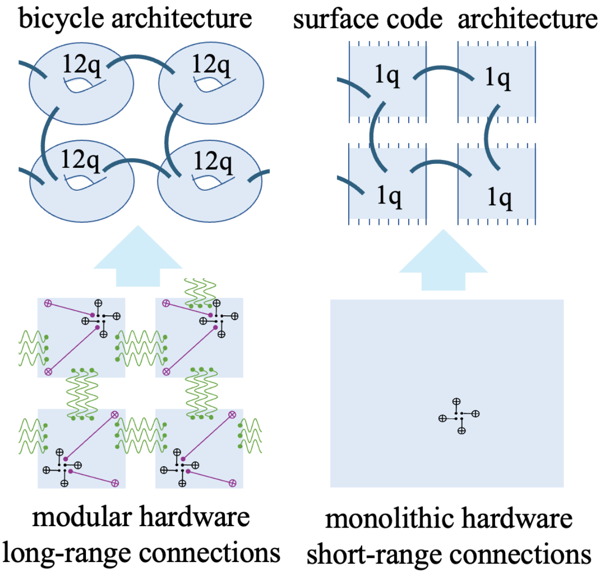


Figure: Demonstration of multi-layer interconnect required for non-planar connectivity in the gross code

Long-range connections enable modularity and scale



IBM **Quantum**

Bicycle architecture

Two code modules

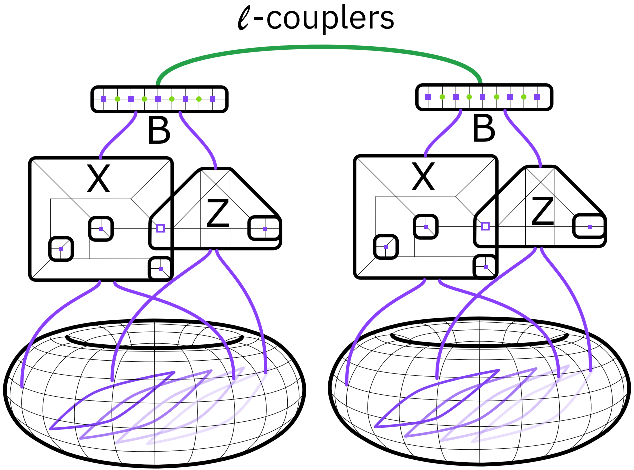


Figure: A gross code is just a memory. We can perform computation by attaching an ancilla system called the *logical processing unit* (LPU) [Cross, He, Rall, Yoder (2024); Williamson, Yoder (2024)]. By connecting LPUs through a bridge system, we can perform joint measurements between two codes.

Bicycle architecture

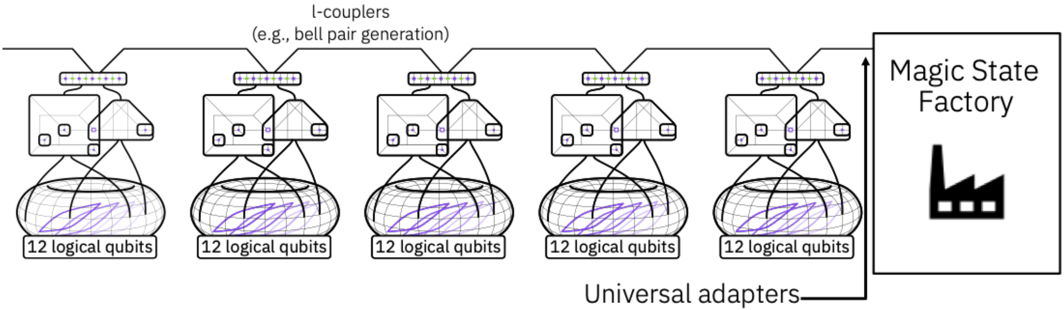
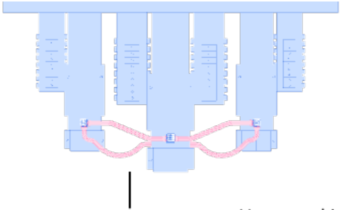


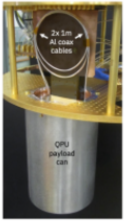
Figure: The bicycle architecture consists of many connected code modules and a (magic state) factory module.

HW demo: L-coupler

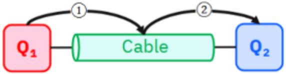


"l-couplers"

Demonstrating key L-coupler technology required for Flamingo



Transfer of quantum state by relay operation



Vacuum rabi oscillations from qubit to cable modes

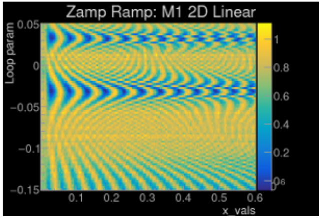


Figure: Demonstration of entangling gate operation over L-coupled interface of up to 96.5% fidelity through interconnect of about 1 meter [Shawn Hall, Kentaro Heya, Yadav Prasad Kandel, Moein Malek, Yves Martin, Jae-Woong Nah, Jason Orcutt, Timothy Phung, Rachel Steiner, Neereja Sundaresan].

Physical qubit costs

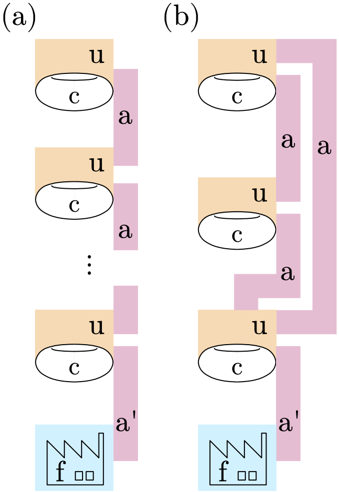


Figure: Code modules (c) with logical processing units (u) and a factory module (f) connected in two ways via adapters (a, a').

system	p	gross	two-gross
c		288	576
u		90	158
a		22	34
f	10^{-3}	454	810
	10^{-4}	463	18,600
a'	10^{-3}	29	13
	10^{-4}	29	49

Table: Physical qubit counts for each system in the architecture

IBM **Quantum**

Bicycle instructions

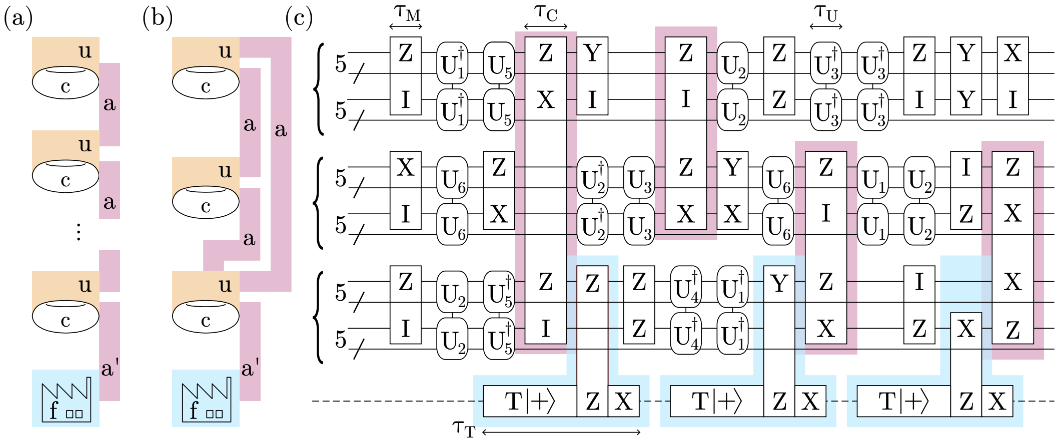


Figure: Given connectivity (a,b) we define the available *bicycle instructions* (c) on the bicycle architecture: In-module measurement and inter-module measurement (red); Automorphisms $U_i \otimes U_i$ and T state injection (blue).

IBM **Quantum**

Bicycle instruction error rates

Bicycle instruction properties

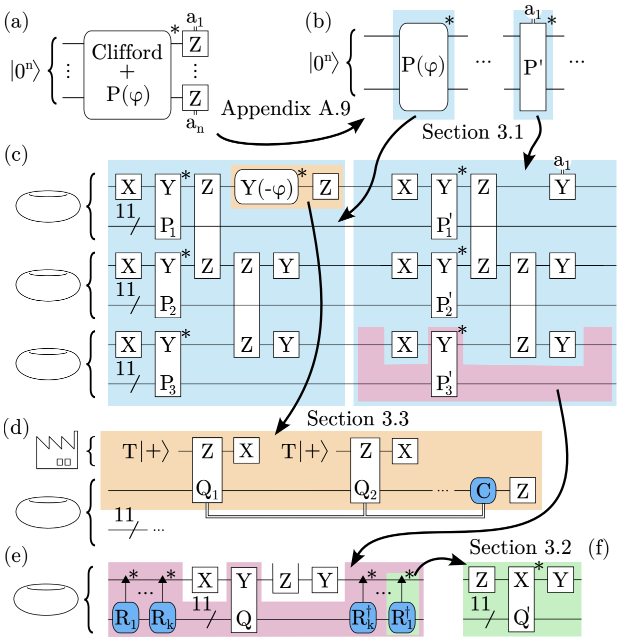
			P_i (logical error rate)	
instruction	module	τ_i (timesteps)	$p = 10^{-3}$	$p = 10^{-4}$
idle	gross	8	$10^{-8.8\pm0.2}$	$10^{-14.8\pm0.4}$
	two-gross	8	$10^{-20.1\pm0.5}$	$10^{-38.3\pm0.9}$
shift automorphism	gross	14	$10^{-6.4\pm0.2}$	$10^{-12.2\pm0.5}$
	two-gross	14	$10^{-14.5\pm0.4}$	$10^{-37\pm1}$
in-module meas.	gross	120	$10^{-5.0\pm0.1}$	$10^{-9.0\pm0.2}$
	two-gross	216	$[10^{-11}]$	$[10^{-20}]$
inter-module meas.	gross	120	$10^{-2.7\pm0.1}$	$10^{-7.3\pm0.3}$
	two-gross	216	$[10^{-9}]$	$[10^{-18}]$

Table: Time duration and logical error rate properties of bicycle instructions. We omit here T state injection, using magic state cultivation [Gidney, Shutty, Jones (2024)] or distillation [Litinski (2019)]. Bracketed numbers are assumptions without data.

IBM **Quantum**

Compiling for the bicycle architecture

Compiler summary



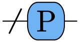
Graphical language

Round boxes are unitary gates

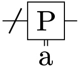
$$V := -\!\!/\!\!-\boxed{V}-\,, \qquad e^{i\frac{\pi}{4}P} := -\!\!/\!\!-\boxed{P}-\,, \tag{1}$$

Graphical language

Round boxes are unitary gates

$V :=$  , $e^{i\frac{\pi}{4}P} :=$  , (1)

and square boxes are measurements

$\frac{\mathbb{1} + (-1)^a P}{2} :=$  , (2)

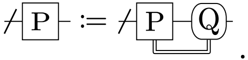
for measurement outcome $a \in \{0, 1\}$.

Measurement projection

Frequently, we don't care about the measurement outcome because it's random.

Measurement projection

Frequently, we don't care about the measurement outcome because it's random. We draw a measurement without an outcome wire and define it as a *measurement projection*,

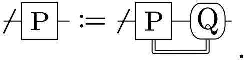


(3)

Effectively, it is the projection $\frac{\mathbb{1}+P}{2}$.

Measurement projection

Frequently, we don’t care about the measurement outcome because it’s random. We draw a measurement without an outcome wire and define it as a *measurement projection*,



(3)

Effectively, it is the projection $\frac{\mathbb{1}+P}{2}$.
This is only possible if we know an anti-commuting stabilizer Q before measuring P so that

$$Q^a \frac{\mathbb{1}+(-1)^a P}{2} |\psi\rangle = \frac{\mathbb{1}+P}{2} Q^a |\psi\rangle = \frac{\mathbb{1}+P}{2} |\psi\rangle. \tag{4}$$

Measurement ancilla

Let us focus on Pauli-generated rotations,

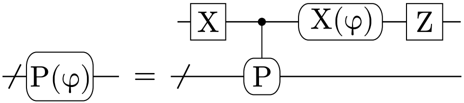
$$P(\varphi) := e^{i\varphi P}. \tag{5}$$

Measurement ancilla

Let us focus on Pauli-generated rotations,

$$P(\varphi) := e^{i\varphi P}. \tag{5}$$

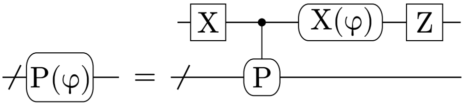
A Pauli-generated rotation can be implemented by



(6)

Distributing across data modules

We can add some ancillas to

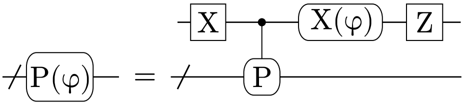


(7)

and assume $P = P_1 \otimes P_2 \otimes P_3$

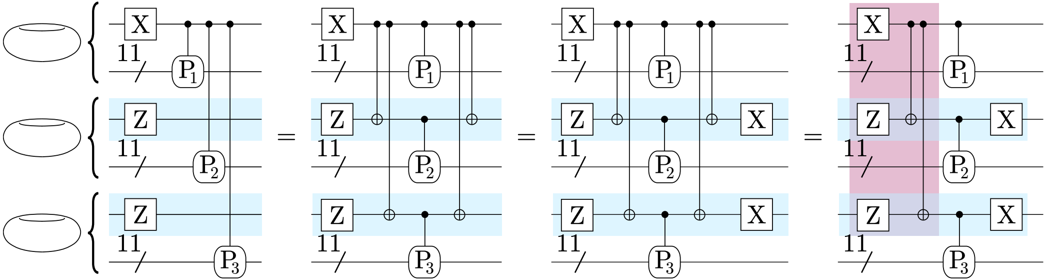
Distributing across data modules

We can add some ancillas to



(7)

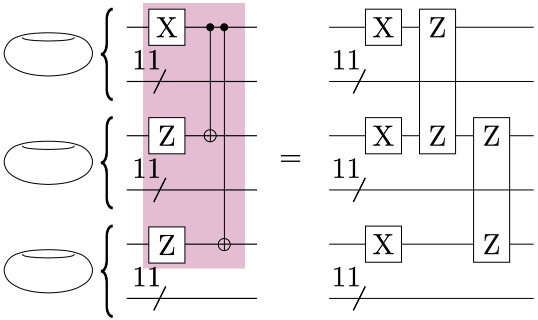
and assume $P = P_1 \otimes P_2 \otimes P_3$



(8)

It’s a GHZ state

We identify the state preparation as a GHZ state

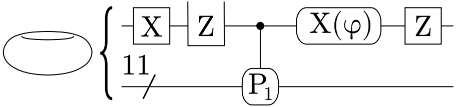


, (9)

which uses just bicycle instructions (assuming connected modules).

The result

Thus, the first code module has the following circuit

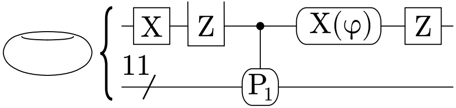


, (10)

with one end of a ZZ-measurement depicted.

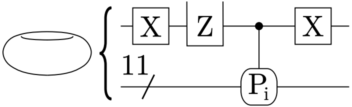
The result

Thus, the first code module has the following circuit



, (10)

with one end of a ZZ-measurement depicted.
Code modules $i = 2, \ldots, M$ have

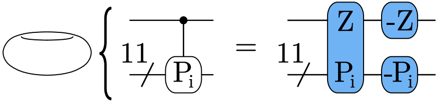


, (11)

where we depict only one ZZ measurement.
(If we were compiling a Pauli measurement, then there are no instances of (10).)

Decomposing controlled-P_i

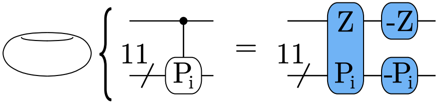
We use the identity, for any Pauli P_i,



(12)

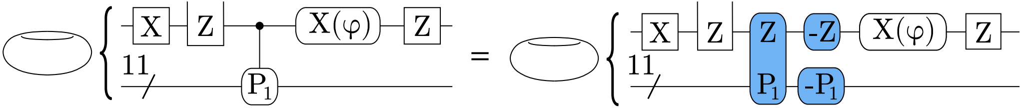
Decomposing controlled-P_i

We use the identity, for any Pauli P_i,

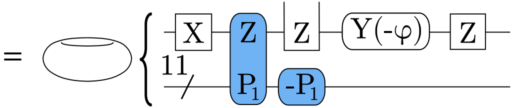


(12)

to simplify



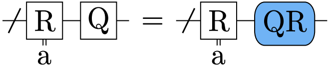
(13)



(14)

Measurement to rotation identity

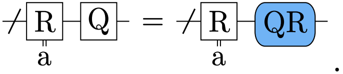
For anti-commuting Paulis Q and R, it holds that



. (15)

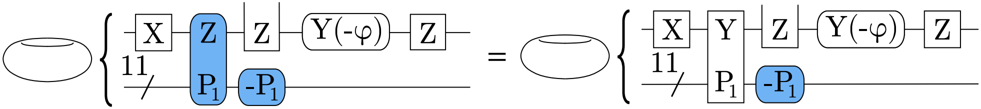
Measurement to rotation identity

For anti-commuting Paulis Q and R, it holds that



(15)

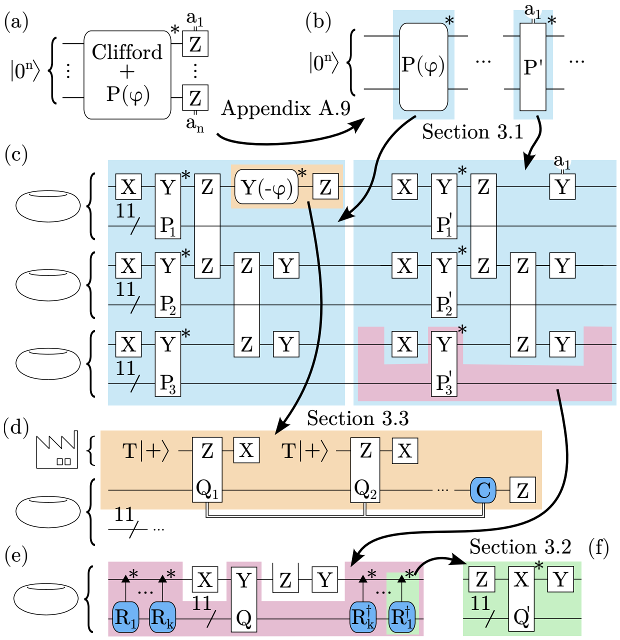
Now, we can rewrite



(16)

The Clifford rotation generated by P_1 can be compiled away using standard techniques.

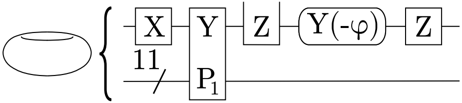
Recap



(a)
$|0^n\rangle$
Clifford + P(φ)
a_1
Z
Z
a_n
Appendix A.9
(b)
$|0^n\rangle$
P(φ)
P'
a_1
Section 3.1
(c)
X
Y
P_1
Z
Y(-φ)
Z
11
X
Y
P_2
Z
Z
Y
X
Y
P_3
Z
Y
X
Y
P'_1
Z
a_1
Y
X
Y
P'_2
Z
Z
Y
X
Y
P'_3
Z
Y
Section 3.3
(d)
T|+⟩
Z
Q_1
X
T|+⟩
Z
Q_2
X
C
Z
11
(e)
R_1
R_k
X
11
Y
Q
Z
Y
$R_k^\dagger$
$R_1^\dagger$
Section 3.2
(f)
Z
11
X
Q'
Y

Measuring arbitrary Paulis in a module

We are faced with implementing the circuit

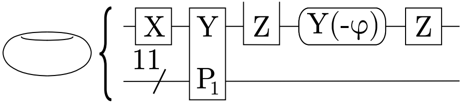


(17)

The measurement $Y \otimes P_1$ is not a bicycle instruction.

Measuring arbitrary Paulis in a module

We are faced with implementing the circuit



(17)

The measurement $Y \otimes P_1$ is not a bicycle instruction.
But we can conjugate P_1 with some 11-qubit Clifford gate so that it becomes easy to implement [Cross, He, Rall, Yoder (2024)].

Overhead of arbitrary Pauli measurement

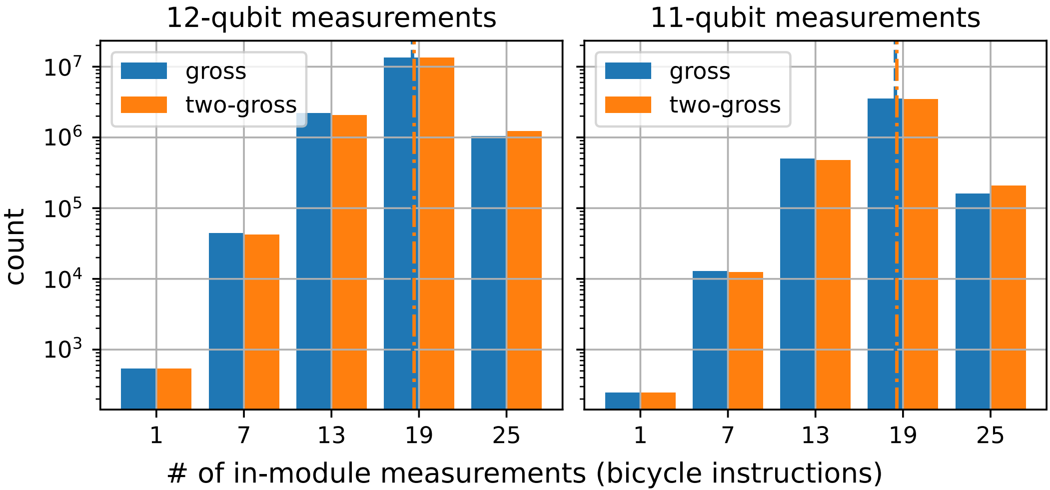


Figure: Number of bicycle measurements needed to measure the 12-qubit Paulis, $Q \otimes P_i$, (left) with mean values 18.55 (gross) and 18.67 (two-gross). In practice, we can minimize over Q (right) to get mean values 18.48 (gross) and 18.58 (two-gross).

End-to-end resource estimates

Logical capability estimates

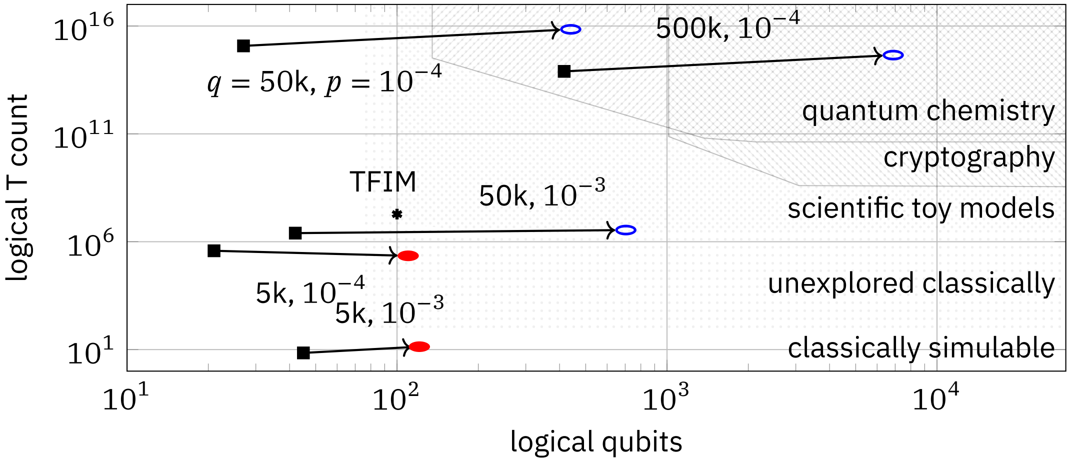


Figure: Bicycle architecture capabilities using gross (red filled ellipse) and two-gross (blue hollow ellipse) codes when compared to surface code architectures (black square) given q physical qubits and p physical error rate.

Conclusion

Scalable fault-tolerance criteria

The bicycle architecture satisfies our six sufficient critera for a scalable fault-tolerant architecture:

(i) *Fault-tolerant* – logical errors are suppressed enough for meaningful algorithms to succeed.

(ii) *Addressable* – individual logical qubits can be prepared or measured throughout the computation.

(iii) *Universal* – a universal set of quantum instructions can be applied to the logical qubits.

(iv) *Adaptive* – measurements are real-time decoded and can alter subsequent quantum instructions.

(v) *Modular* – the hardware is distributed across a set of replaceable modules connected quantumly.

(vi) *Efficient* – meaningful algorithms can be executed with reasonable physical resources.

Opportunities for improvement

Reduce the time overhead The synthesis of arbitrary Pauli measurements incurs significant overhead.

Increase parallelism The Pauli-based compilation scheme leads to sequential T gate execution.

Decoding with speed and accuracy For instance, decoding speed and error rates of inter-module measurement.

Increasing code and circuit distances With long-range coupling, can consider a wide variety of qLDPC codes.

Modifying the bicycle instructions Simplify circuits for bicycle instructions, especially for complicated and error-prone instructions.