SINGLE STEP PARITY CHECK GATE SET for QUANTUM ERROR CORRECTION

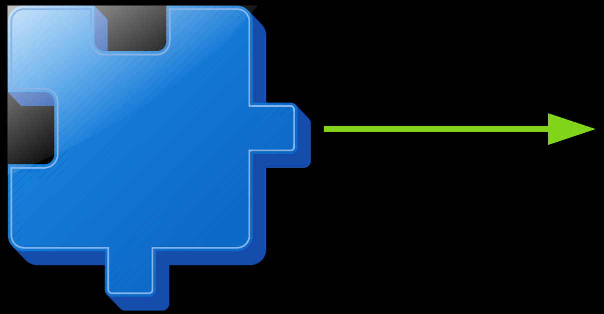
Gözde Üstün[1,2]
Andrea Morello[1,2] & Simon Devitt[3]

[1] University of New South Wales, Sydney
[2] Centre for Quantum Computation & Communication Technologies
[3] University of Technology Sydney

1. INTRODUCTION - THRESHOLD

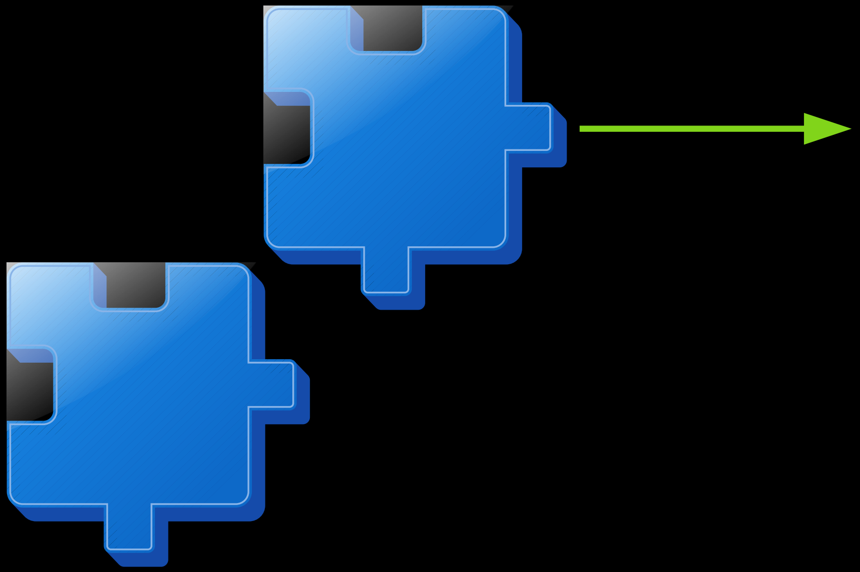
1. INTRODUCTION - THRESHOLD

1. INTRODUCTION - THRESHOLD



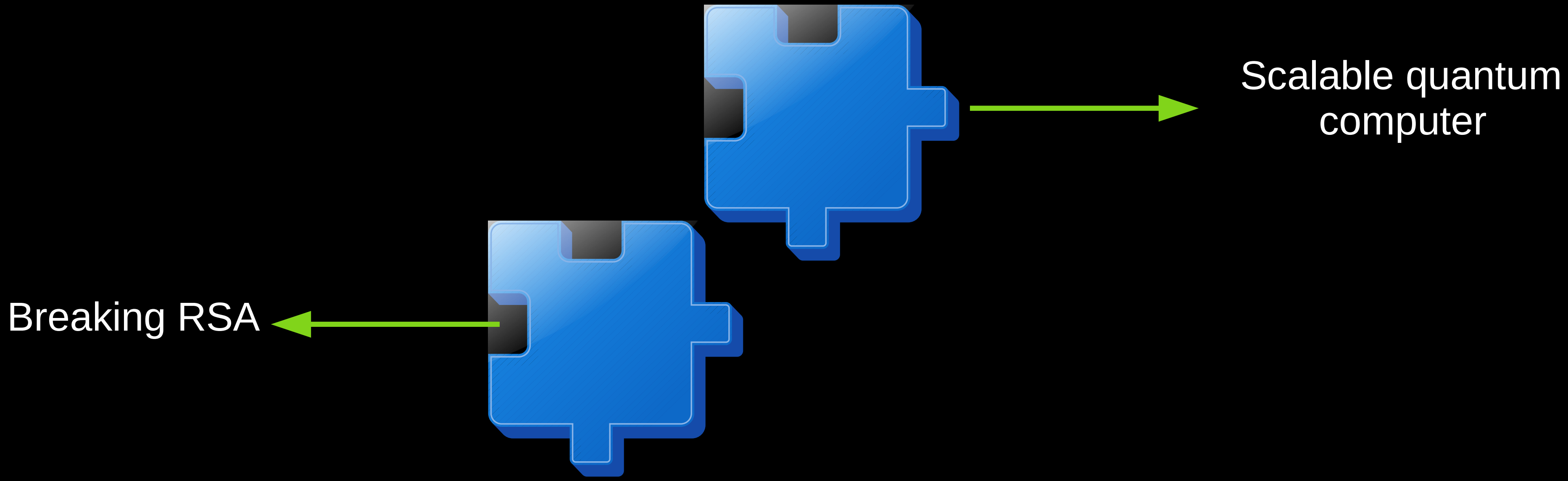
Scalable quantum computer

1. INTRODUCTION - THRESHOLD

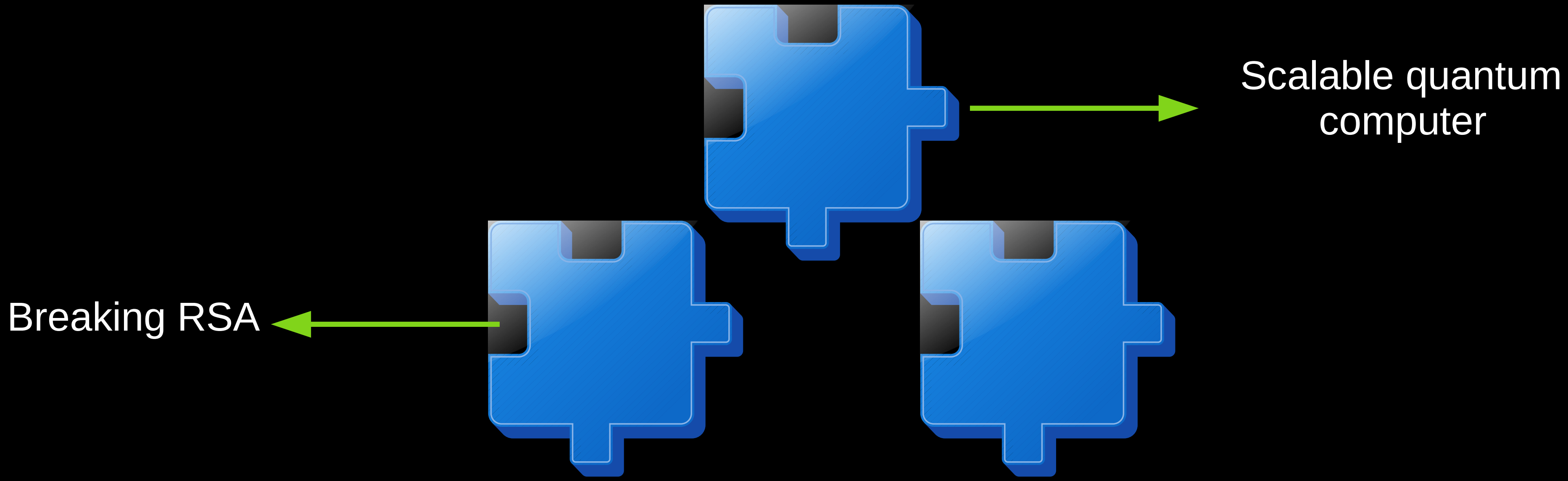


Scalable quantum computer

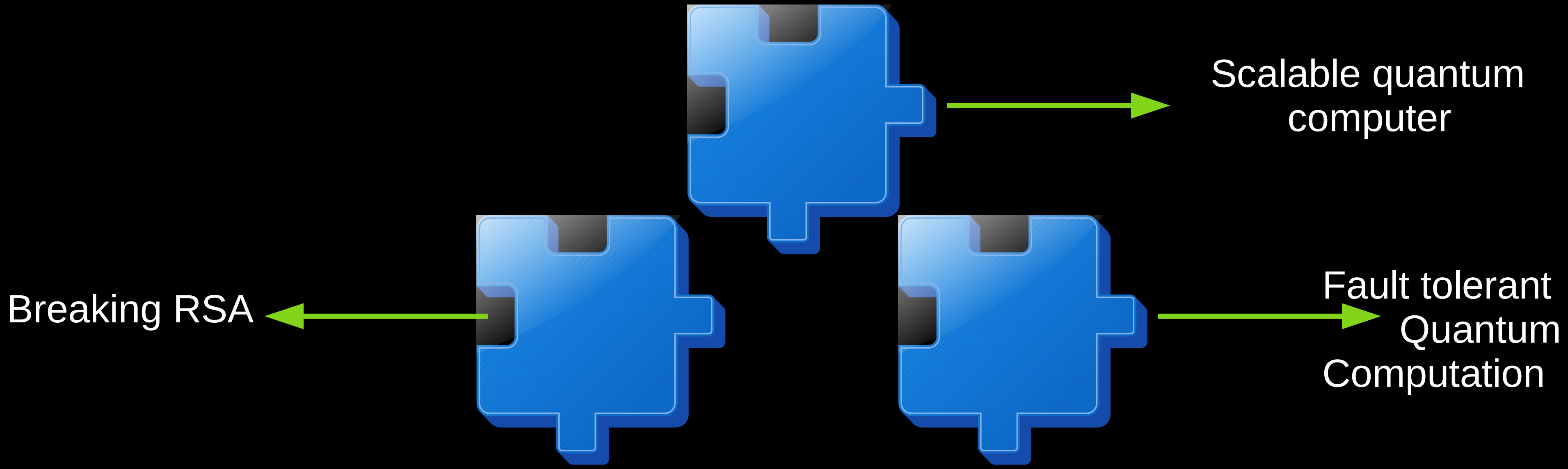
1. INTRODUCTION - THRESHOLD



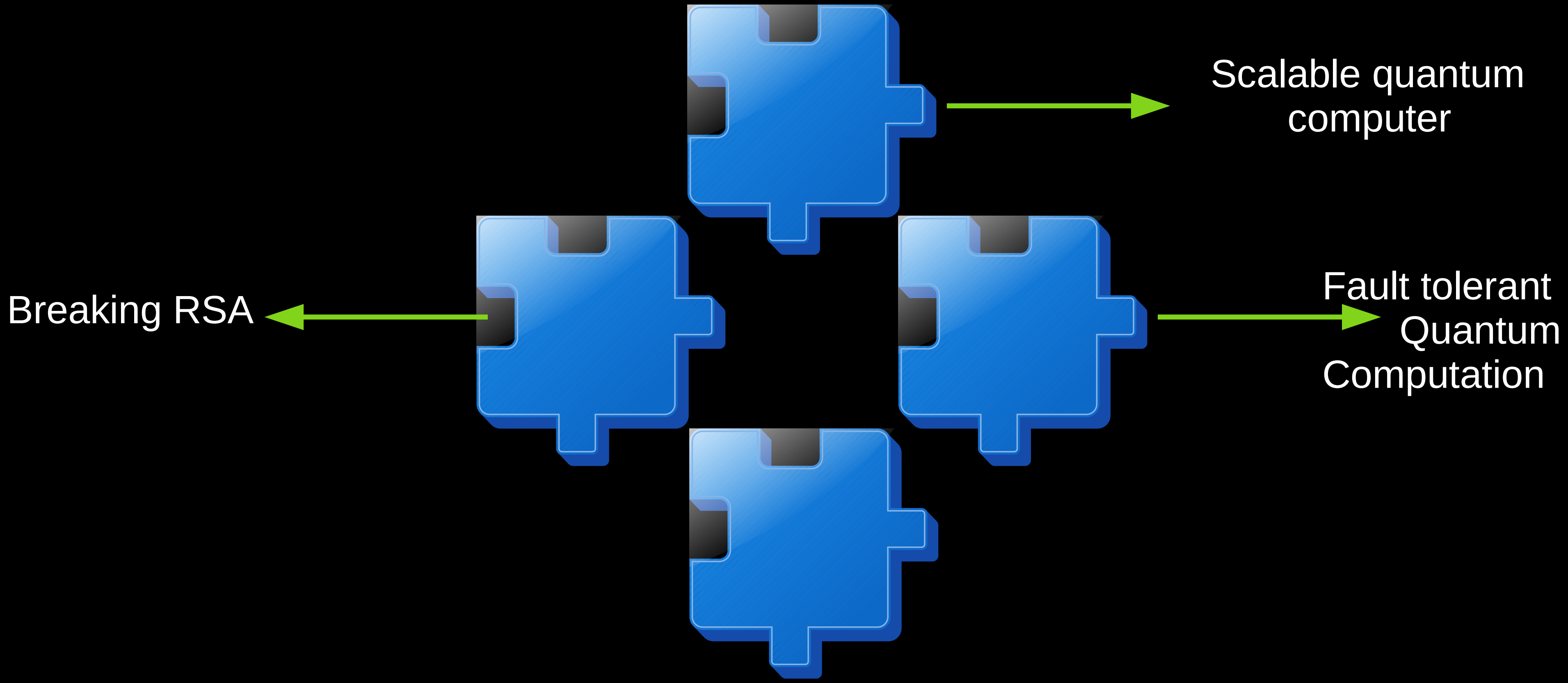
1. INTRODUCTION - THRESHOLD



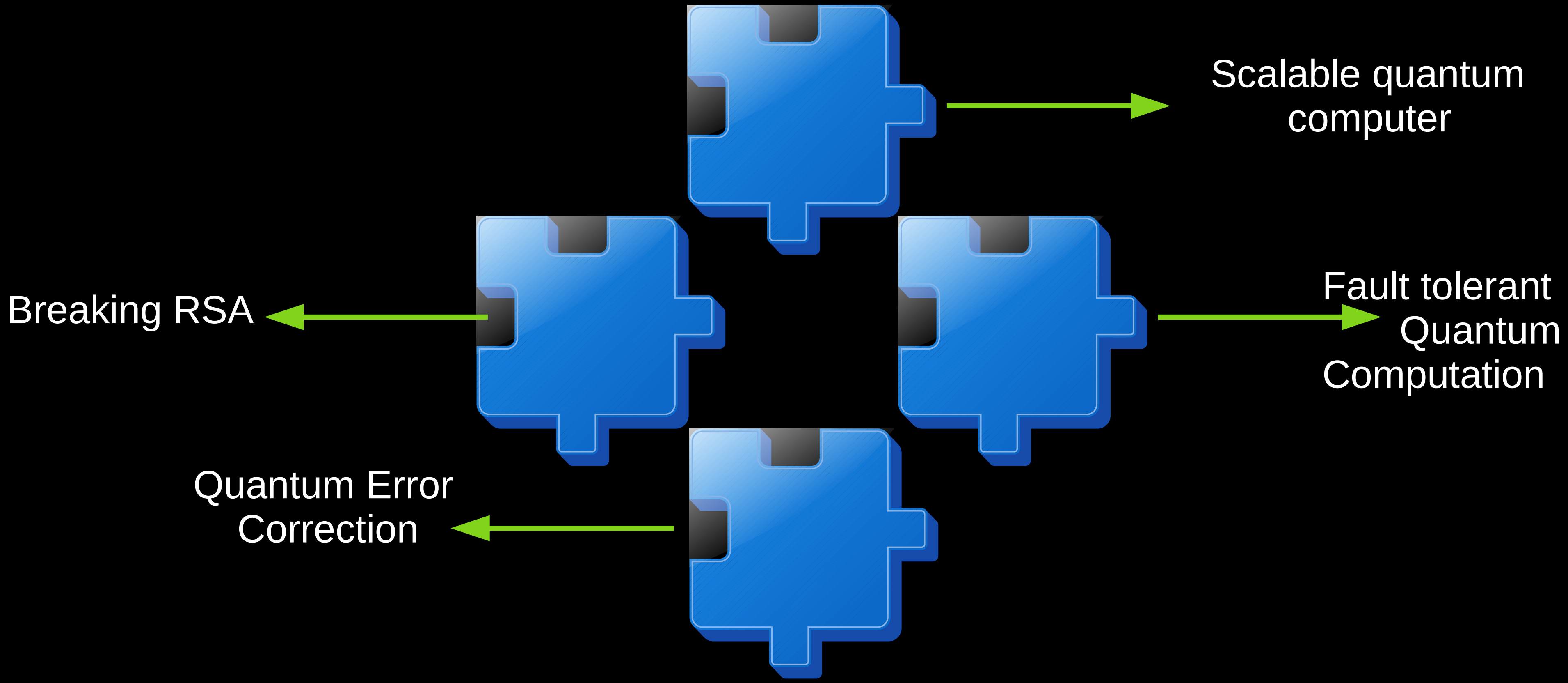
1. INTRODUCTION - THRESHOLD



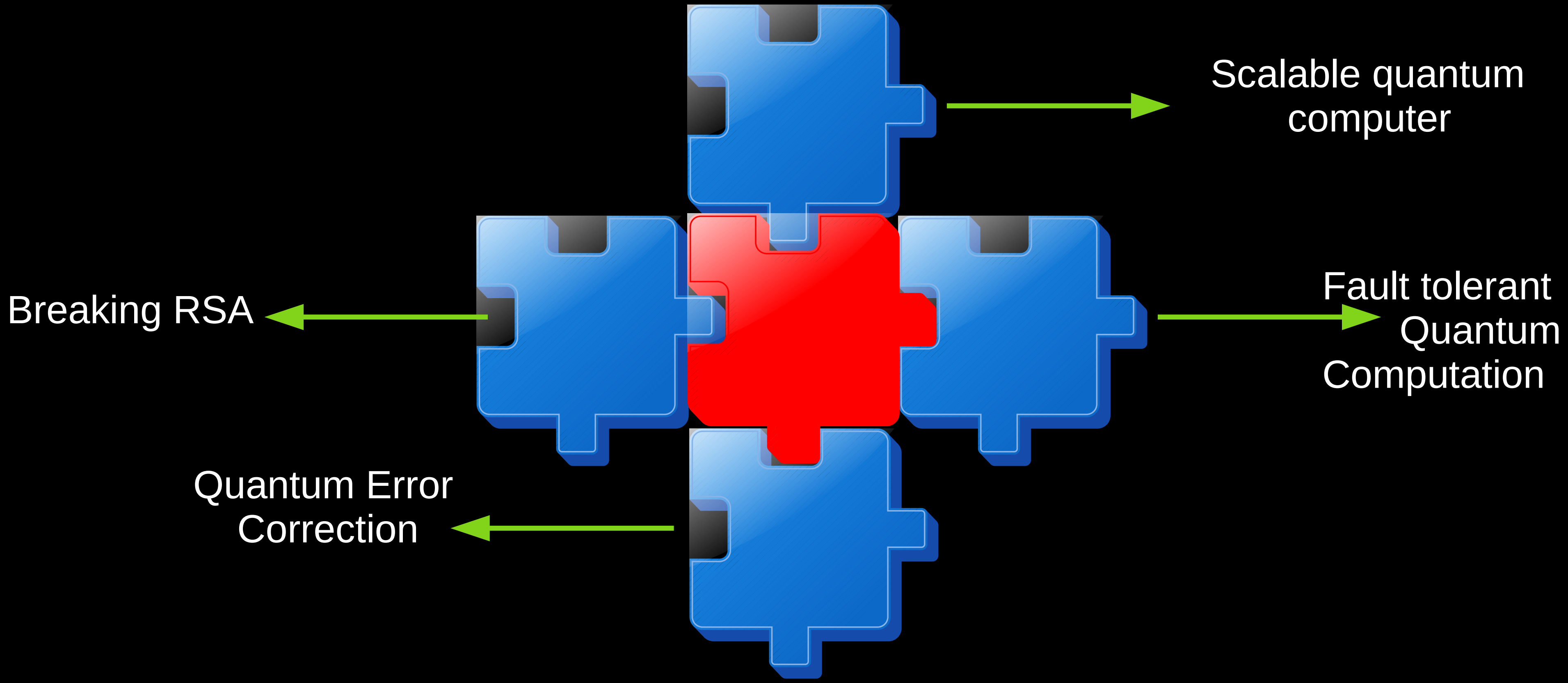
1. INTRODUCTION - THRESHOLD



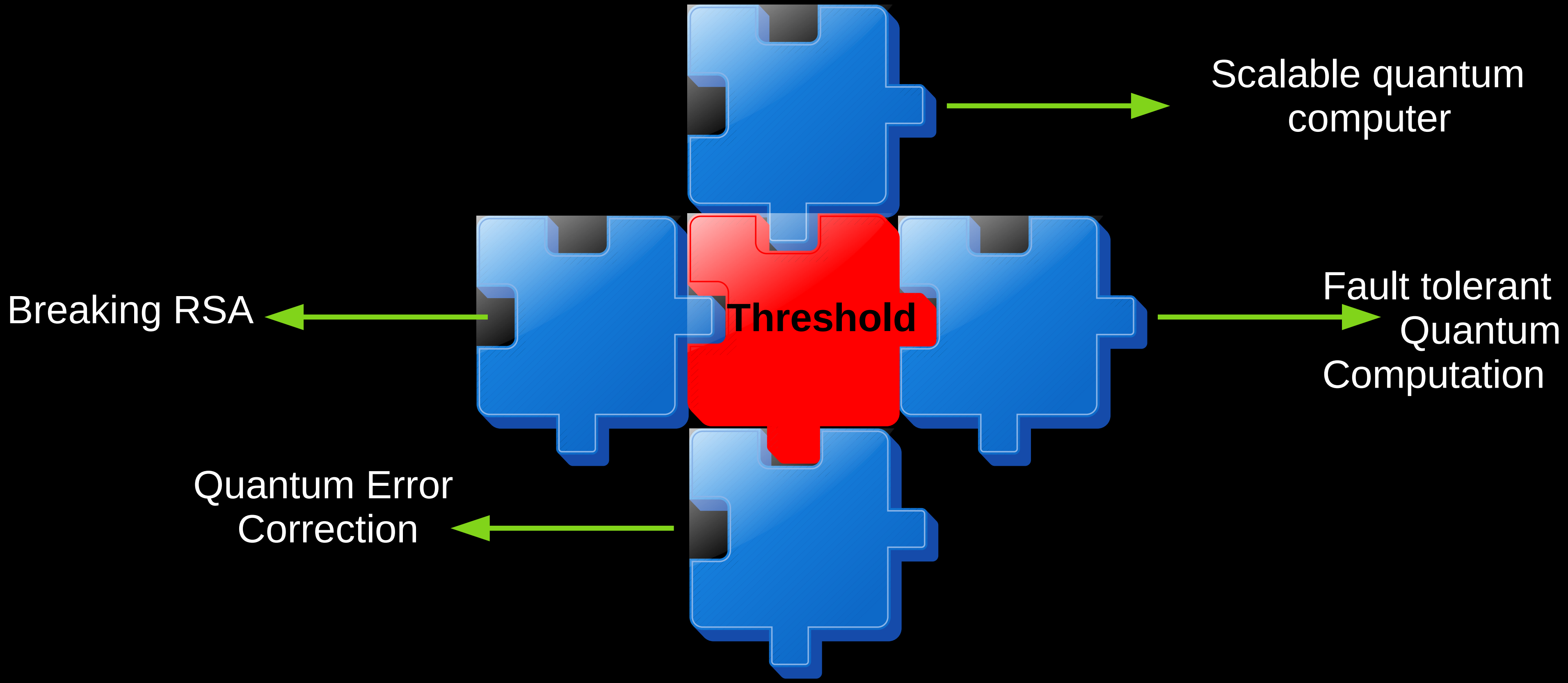
1. INTRODUCTION - THRESHOLD



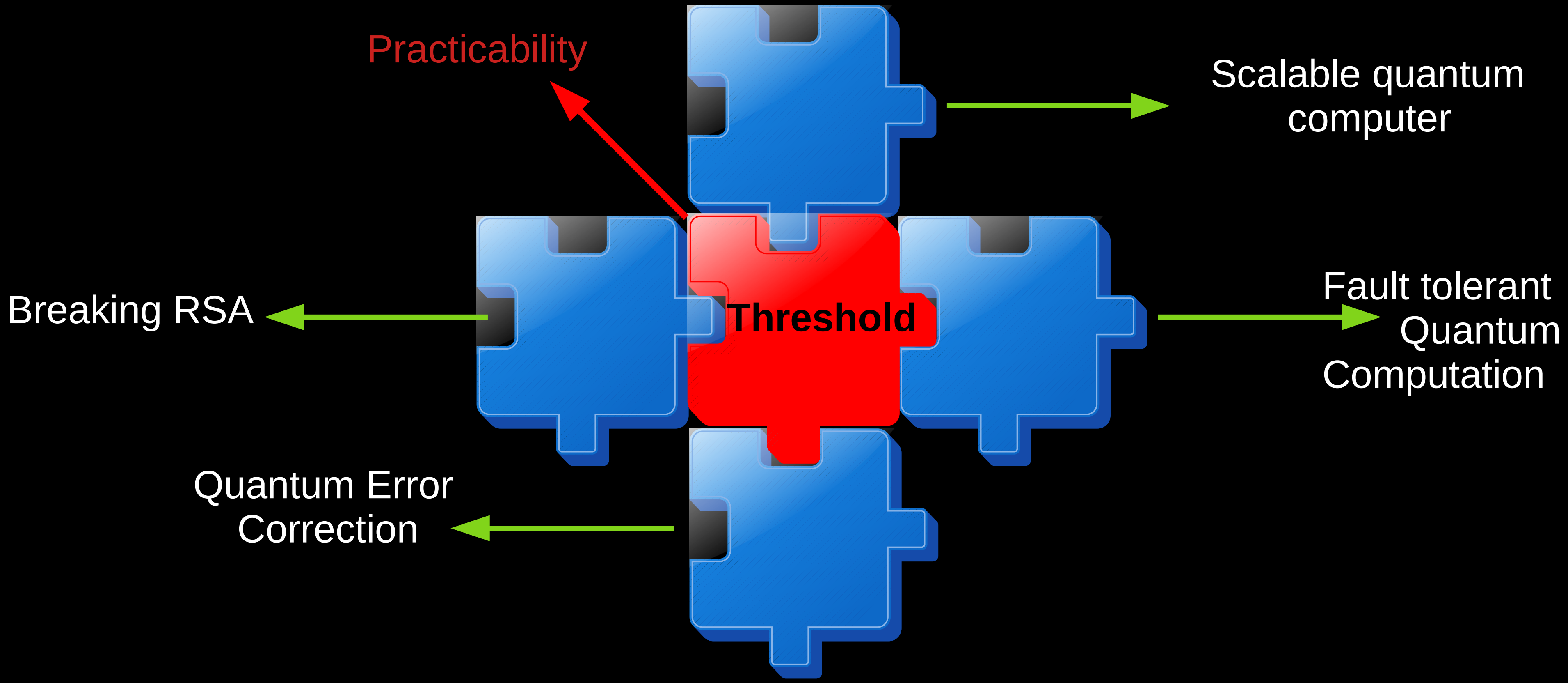
1. INTRODUCTION - THRESHOLD



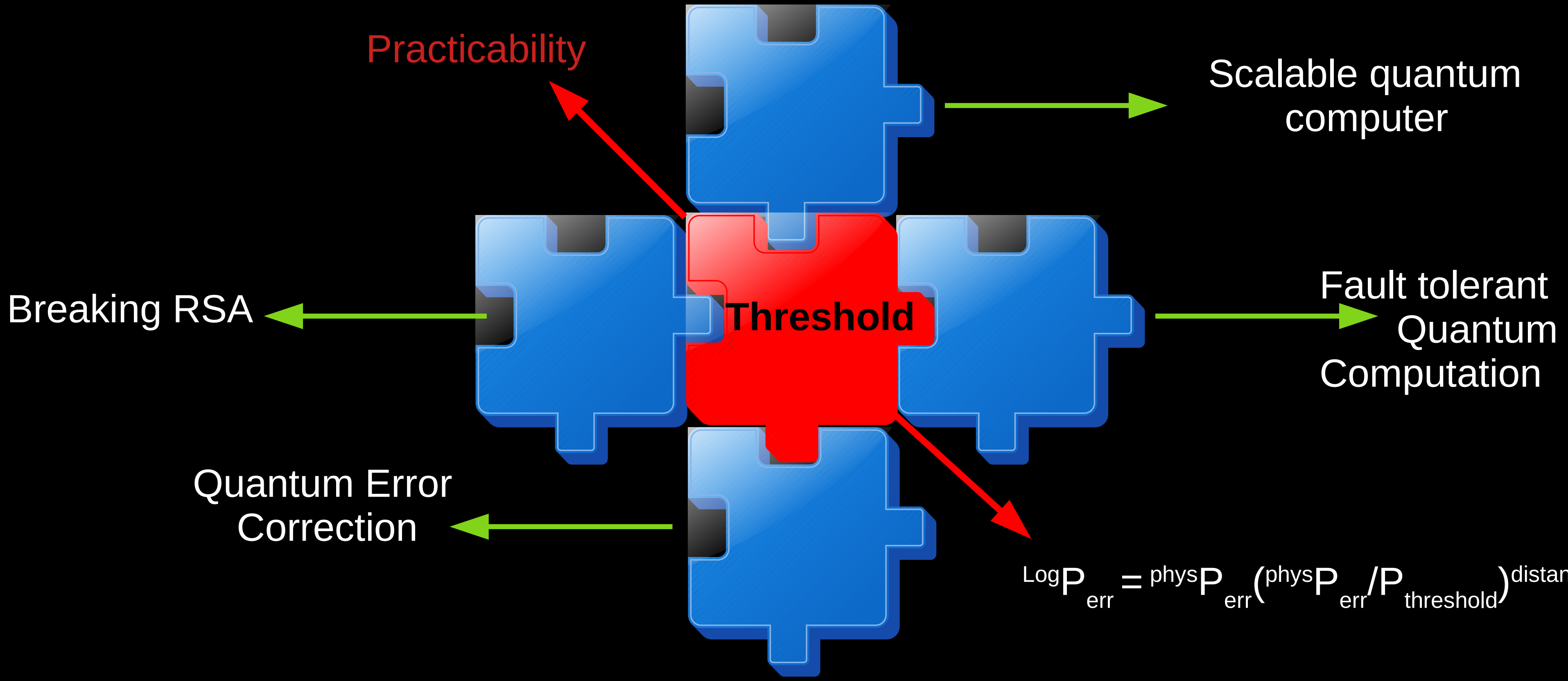
1. INTRODUCTION - THRESHOLD



1. INTRODUCTION - THRESHOLD



1. INTRODUCTION - THRESHOLD



$$^{Log}P_{err} = {}^{phys}P_{err}\left({}^{phys}P_{err}/P_{threshold}\right)^{distance-1}$$

2. WHAT IS THE PROBLEM?

2. WHAT IS THE PROBLEM?

2. WHAT IS THE PROBLEM?

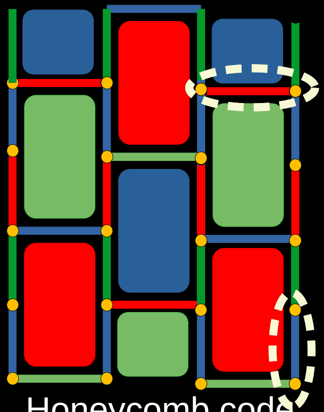
2. WHAT IS THE PROBLEM?

2. WHAT IS THE PROBLEM?

Building blocks...

2. WHAT IS THE PROBLEM?

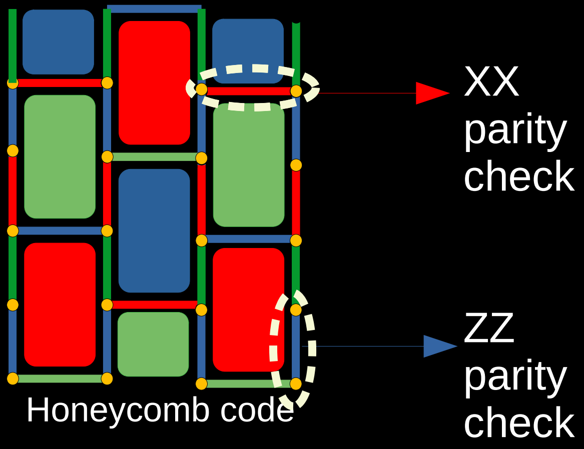
Building blocks...



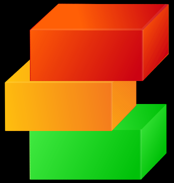
Honeycomb code

2. WHAT IS THE PROBLEM?

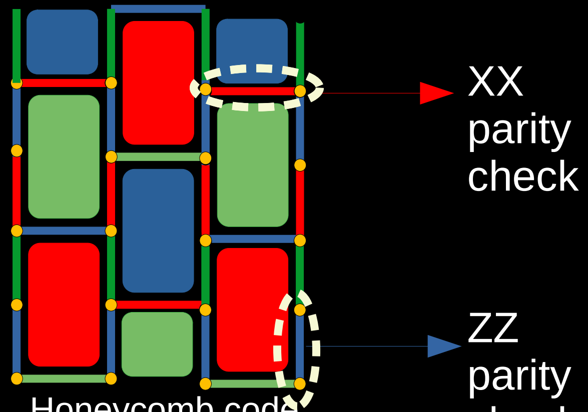
Building blocks...



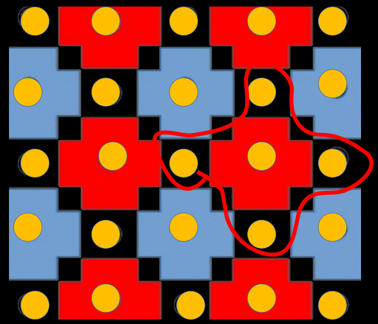
2. WHAT IS THE PROBLEM?



Building blocks...



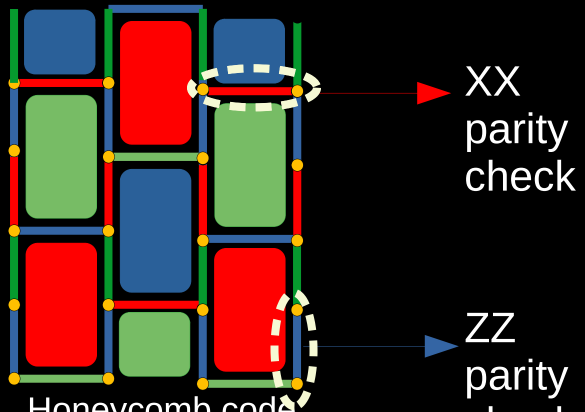
Honeycomb code



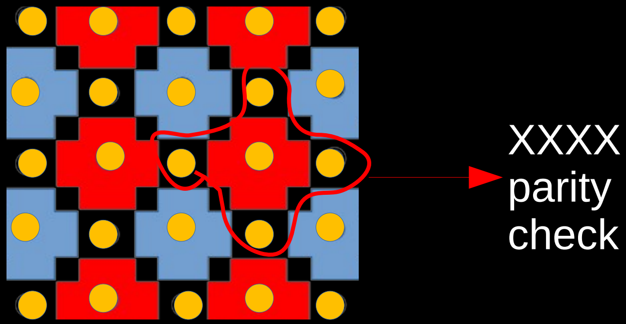
Surface code

2. WHAT IS THE PROBLEM?

Building blocks...

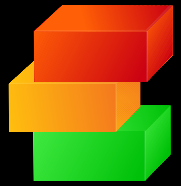


Honeycomb code

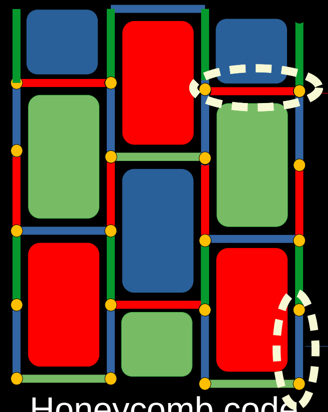


Surface code

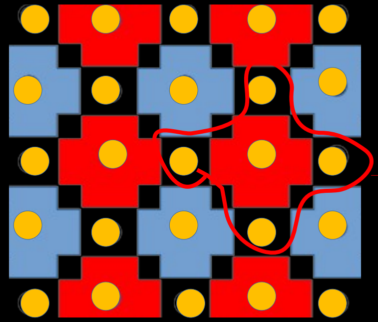
2. WHAT IS THE PROBLEM?



Building blocks...



Honeycomb code

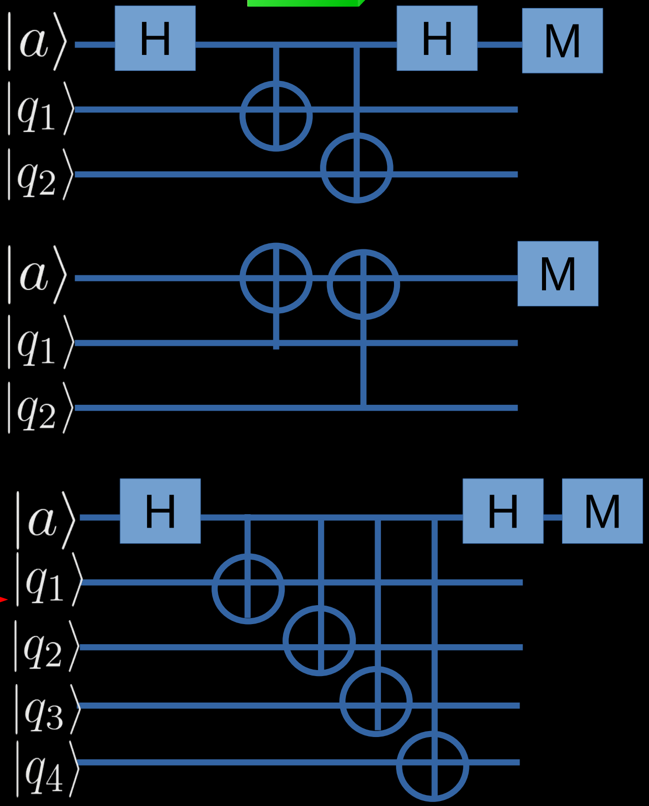


Surface code

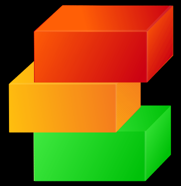
XX parity check

ZZ parity check

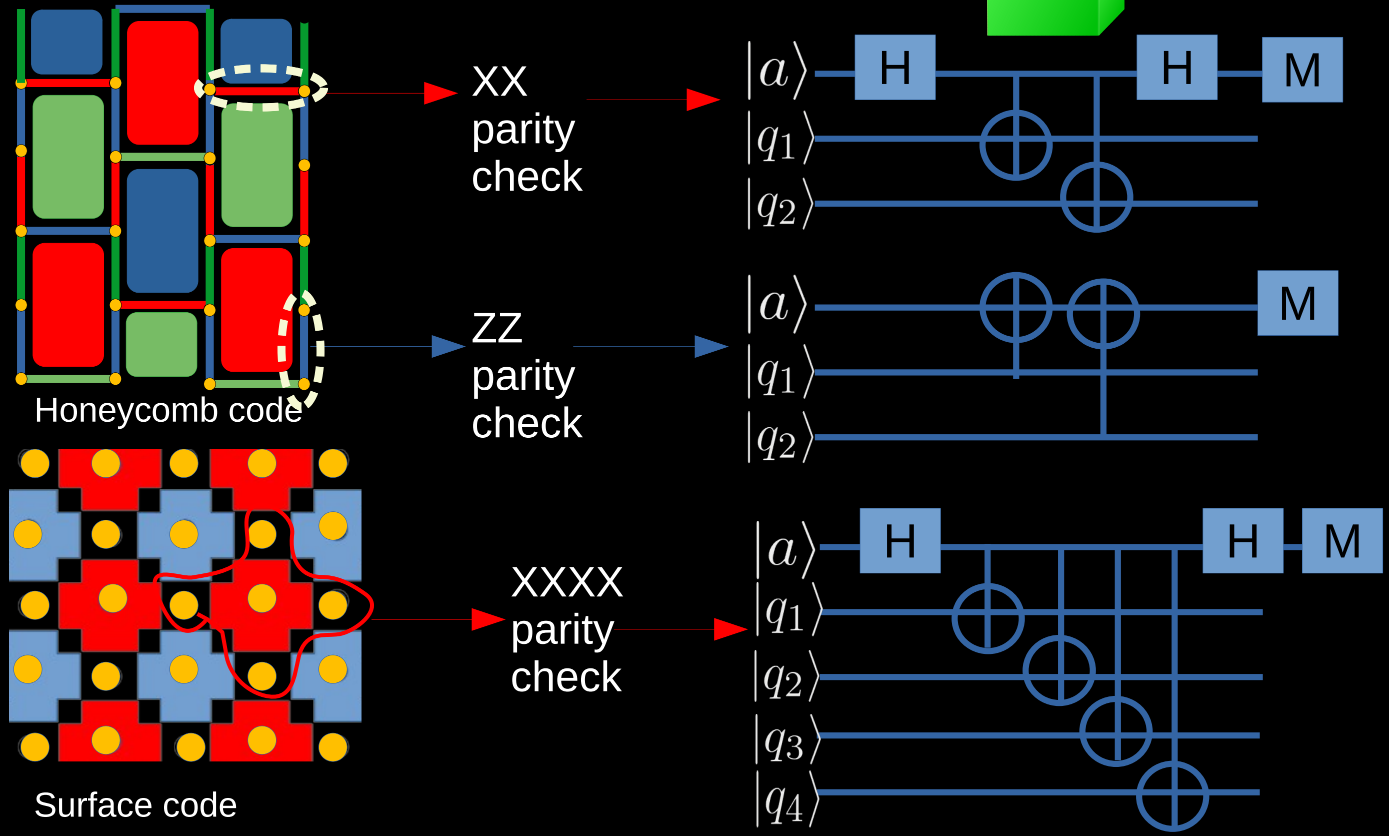
XXXX parity check

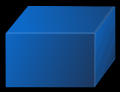


2. WHAT IS THE PROBLEM?



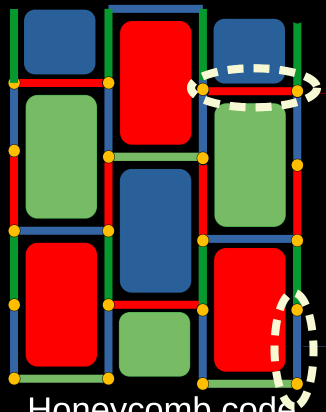
Building blocks...





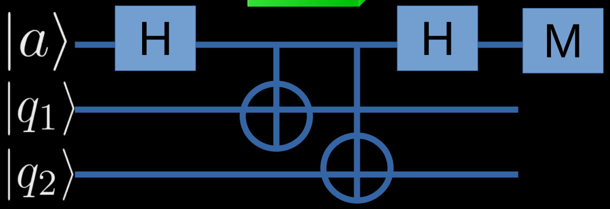
2. WHAT IS THE PROBLEM?

Building blocks...

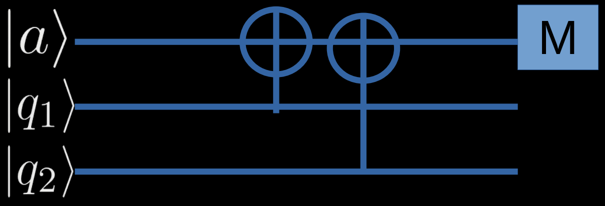


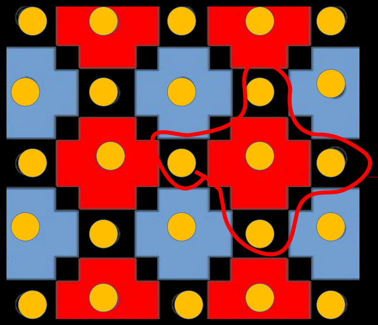
Honeycomb code

XX parity check



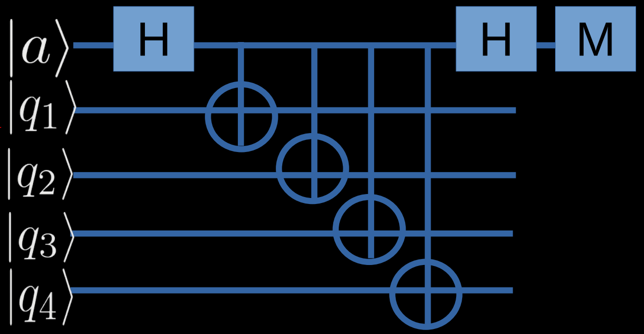
ZZ parity check

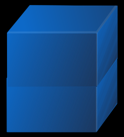




Surface code

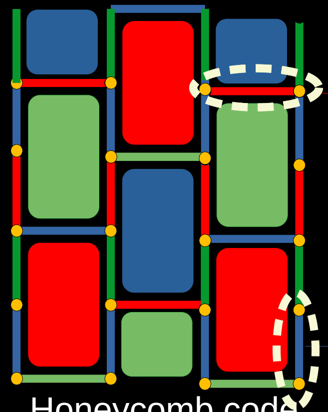
XXXX parity check



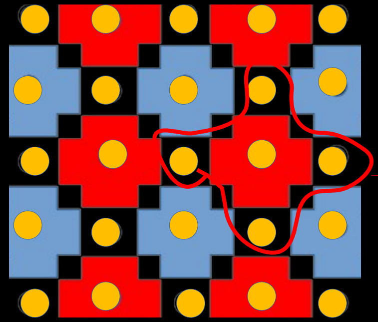


2. WHAT IS THE PROBLEM?

Building blocks...



Honeycomb code

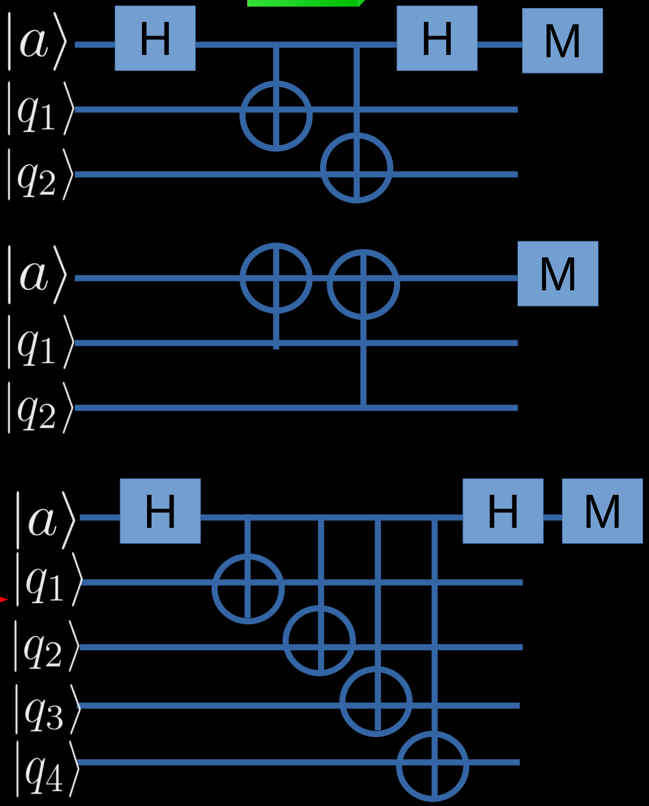


Surface code

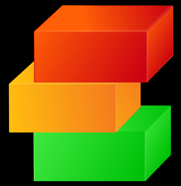
XX parity check

ZZ parity check

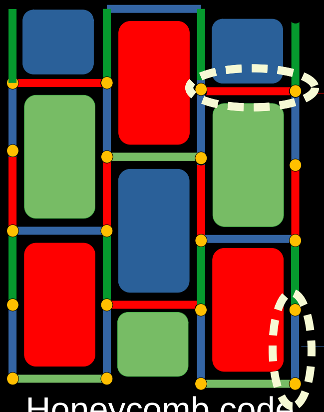
XXXX parity check



2. WHAT IS THE PROBLEM?



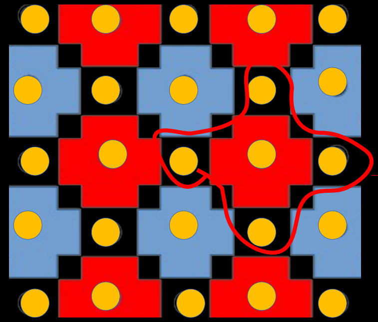
Building blocks...



Honeycomb code

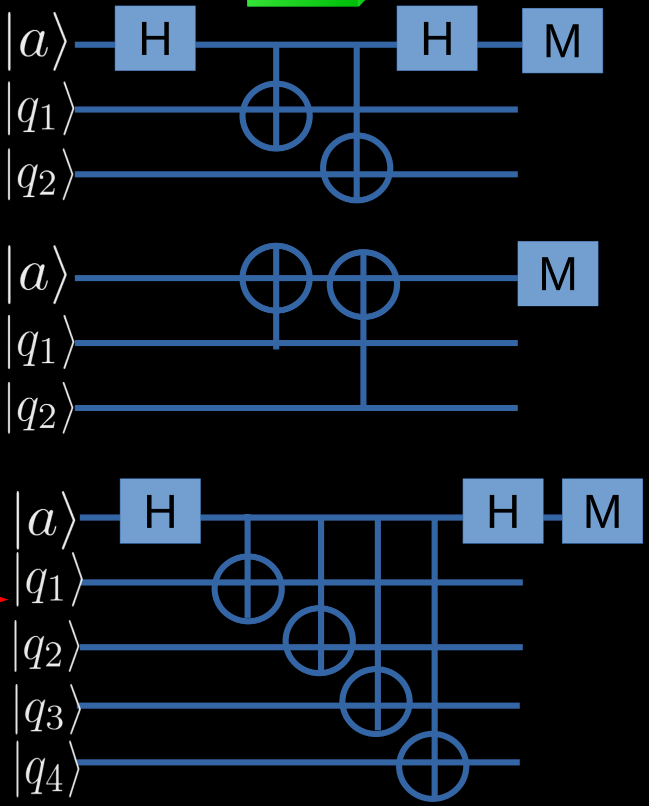
XX parity check

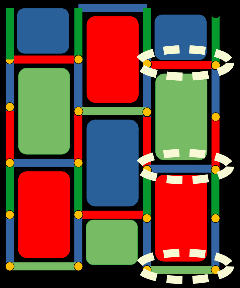
ZZ parity check



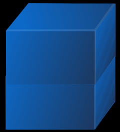
Surface code

XXXX parity check



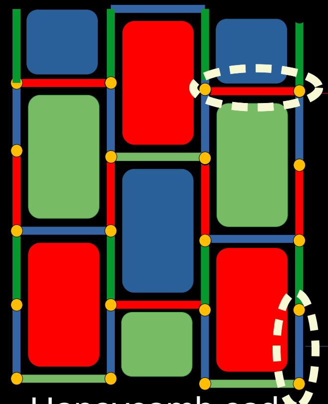


A Fault-Tolerant Honeycomb Memory, Gidney et all,Quantum 5, 605 (2021).



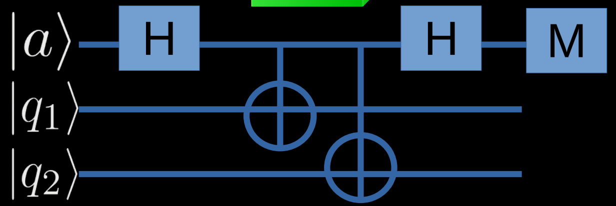
2. WHAT IS THE PROBLEM?

Building blocks...

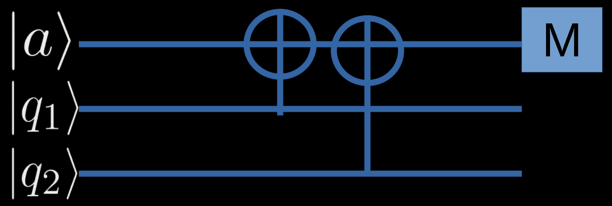


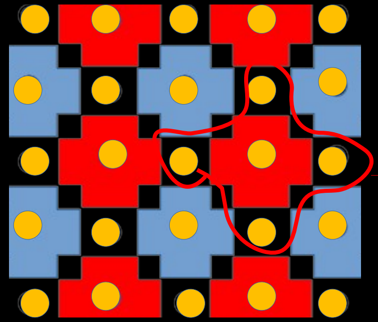
Honeycomb code

XX parity check



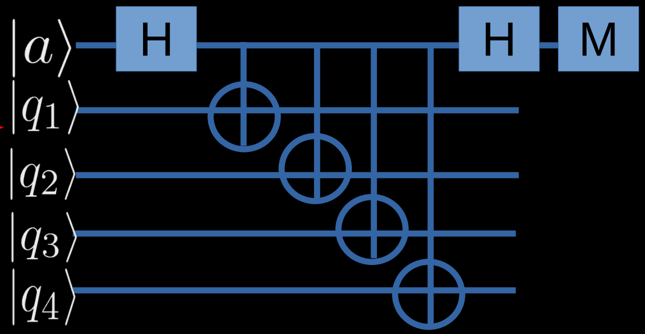
ZZ parity check

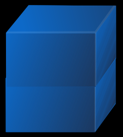


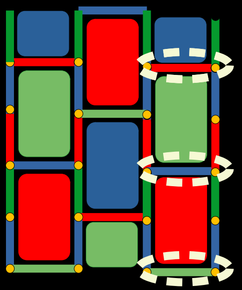


Surface code

XXXX parity check







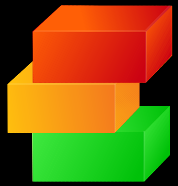
Direct XX parity check

Direct YY parity check

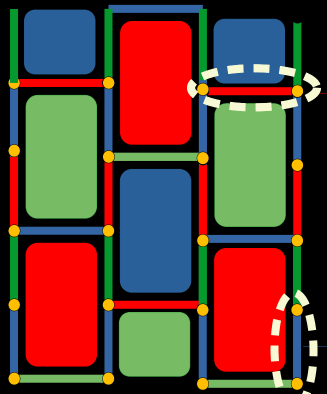
Direct ZZ parity check

A Fault-Tolerant Honeycomb Memory, Gidney et all,Quantum 5, 605 (2021).

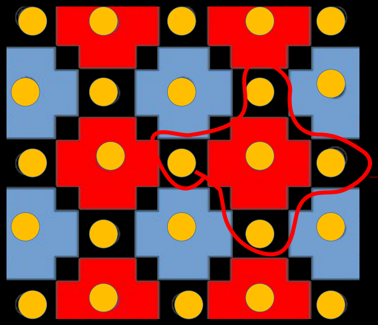
2. WHAT IS THE PROBLEM?



Building blocks...



Honeycomb code

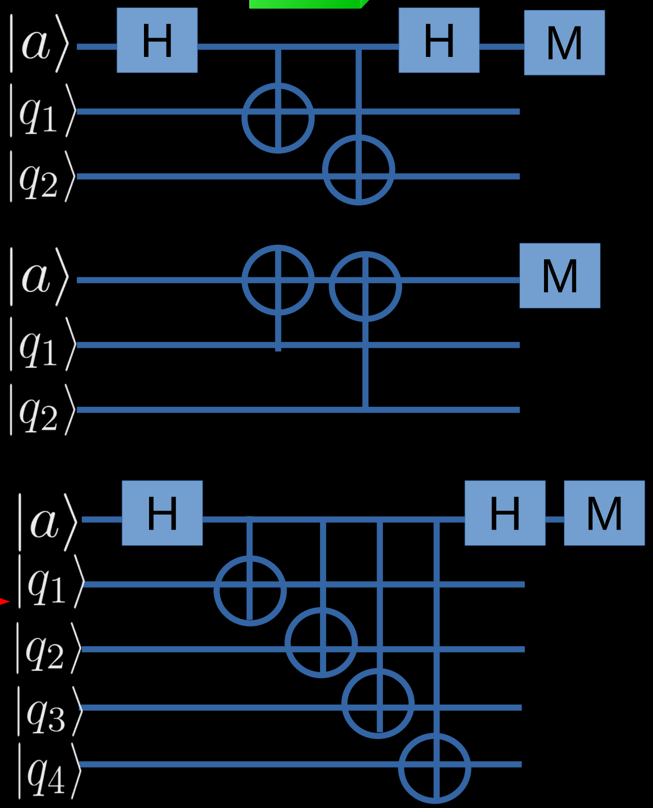


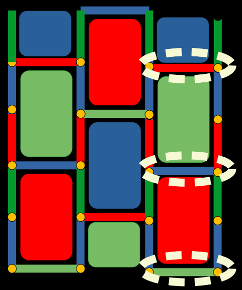
Surface code

XX parity check

ZZ parity check

XXXX parity check





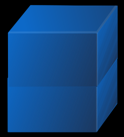
Direct XX parity check

Direct YY parity check

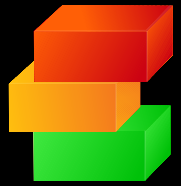
Direct ZZ parity check

A Fault-Tolerant Honeycomb Memory, Gidney et all,Quantum 5, 605 (2021).

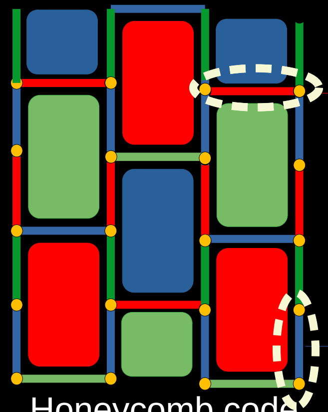
Three times higher threshold than surface code!



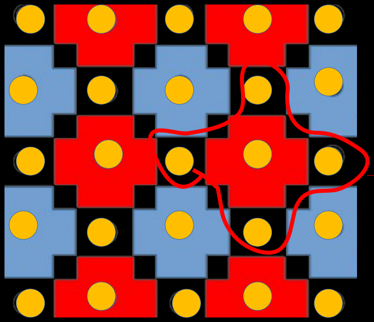
2. WHAT IS THE PROBLEM?



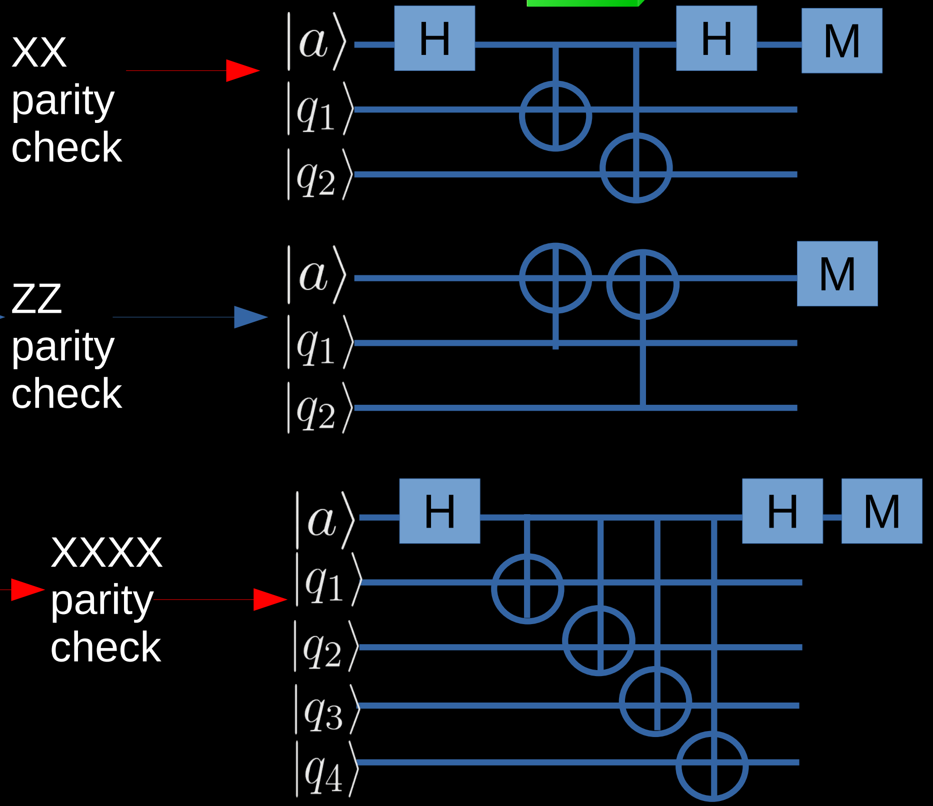
Building blocks...

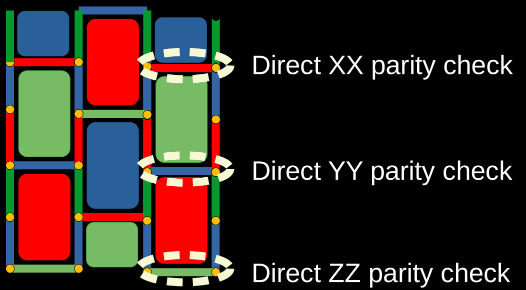


Honeycomb code



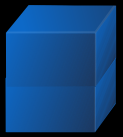
Surface code





A Fault-Tolerant Honeycomb Memory, Gidney et all,Quantum 5, 605 (2021).

Three times higher threshold than surface code!
If there is M_{pp2} …???



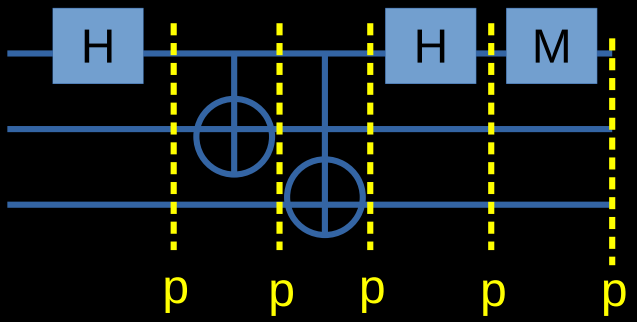
3. CURRENT METHODS

3. CURRENT METHODS

From fidelity to p

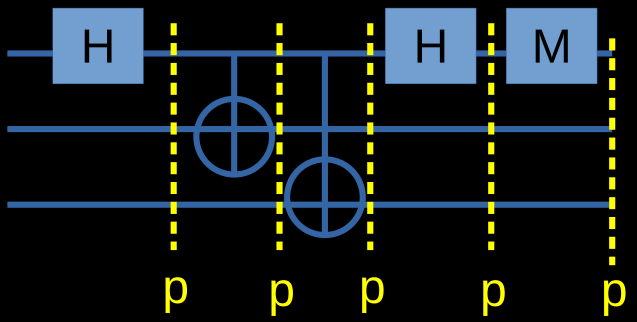
3. CURRENT METHODS

From fidelity to p



3. CURRENT METHODS

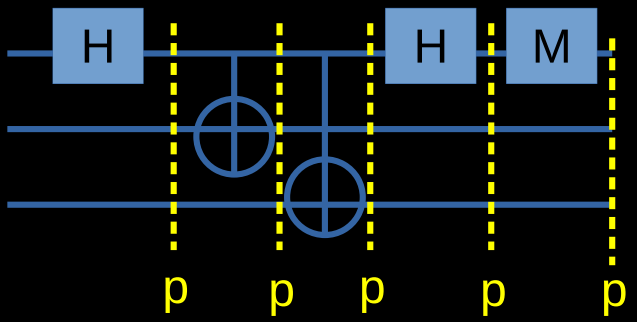
From fidelity to p



1. Take GST matrices

3. CURRENT METHODS

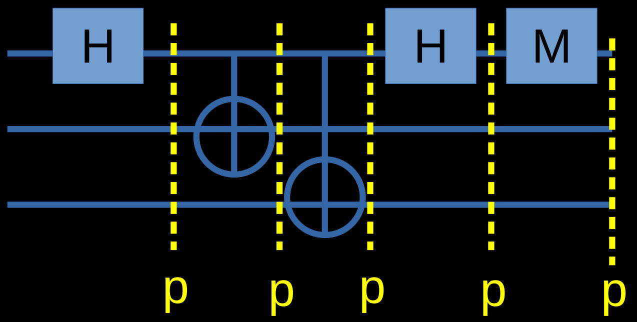
From fidelity to p



1. Take GST matrices
2. They are actually PTM matrices

3. CURRENT METHODS

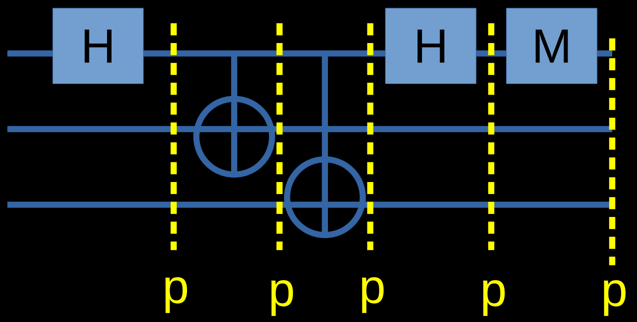
From fidelity to p



1. Take GST matrices
2. They are actually PTM matrices
3. Find Kraus Operators

3. CURRENT METHODS

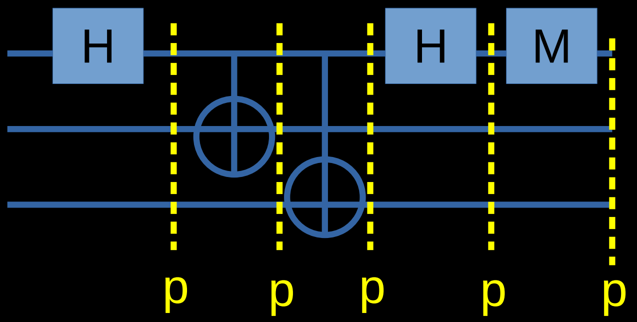
From fidelity to p



1. Take GST matrices
2. They are actually PTM matrices
3. Find Kraus Operators
4. Write Kraus operators in terms of Pauli Matrices

3. CURRENT METHODS

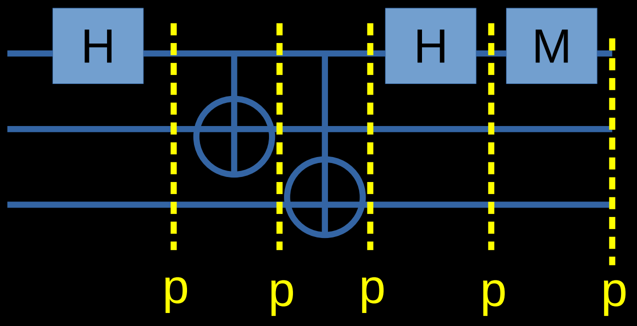
From fidelity to p



1. Take GST matrices
2. They are actually PTM matrices
3. Find Kraus Operators
4. Write Kraus operators in terms of Pauli Matrices
5. Write final density matrices

3. CURRENT METHODS

From fidelity to p

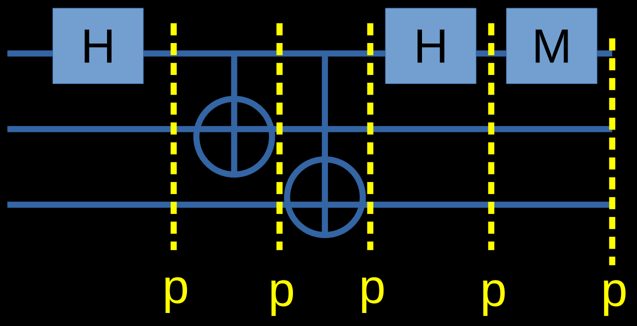


1. Take GST matrices
2. They are actually PTM matrices
3. Find Kraus Operators
4. Write Kraus operators in terms of Pauli Matrices
5. Write final density matrices

As a result, we analysed the succession/perfection rate in terms of accumulated error in the stochastic Pauli channels

3. CURRENT METHODS

From fidelity to p

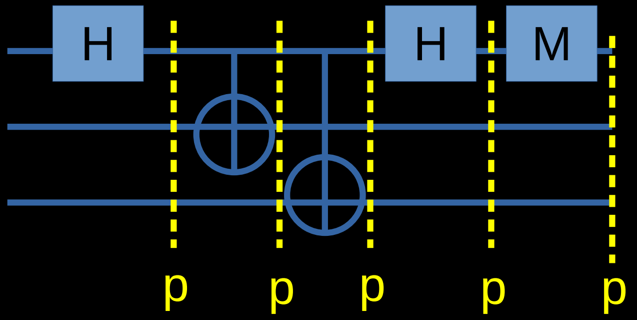


1. Take GST matrices
2. They are actually PTM matrices
3. Find Kraus Operators
4. Write Kraus operators in terms of Pauli Matrices
5. Write final density matrices

As a result, we analysed the succession/perfection rate in terms of accumulated error in the stochastic Pauli channels

3. CURRENT METHODS

From fidelity to p



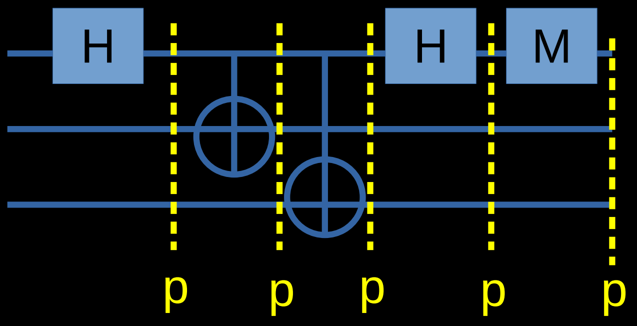
1. Take GST matrices
2. They are actually PTM matrices
3. Find Kraus Operators
4. Write Kraus operators in terms of Pauli Matrices
5. Write final density matrices

As a result, we analysed the succession/perfection rate in terms of accumulated error in the stochastic Pauli channels

- Each gate is created with 99.43% fidelity

3. CURRENT METHODS

From fidelity to p



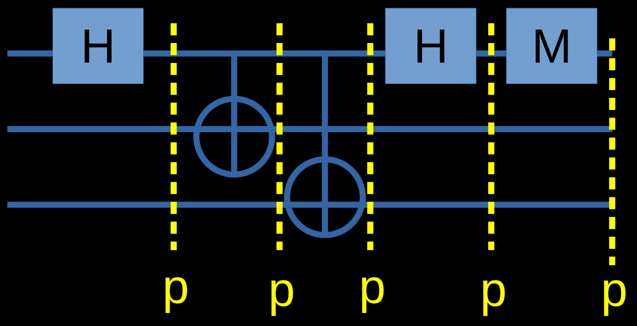
1. Take GST matrices
2. They are actually PTM matrices
3. Find Kraus Operators
4. Write Kraus operators in terms of Pauli Matrices
5. Write final density matrices

As a result, we analysed the succession/perfection rate in terms of accumulated error in the stochastic Pauli channels

- Each gate is created with 99.43% fidelity

3. CURRENT METHODS

From fidelity to p



1. Take GST matrices
2. They are actually PTM matrices
3. Find Kraus Operators
4. Write Kraus operators in terms of Pauli Matrices
5. Write final density matrices

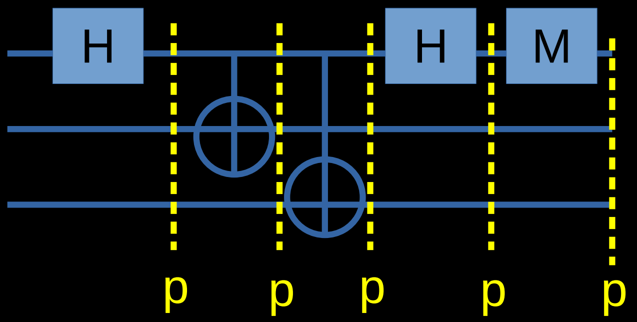
As a result, we analysed the succession/perfection rate in terms of accumulated error in the stochastic Pauli channels

- Each gate is created with 99.43% fidelity

Surface codes: Towards practical large-scale quantum computation, Fowler et all,Phys. Rev. A 86, 032324

3. CURRENT METHODS

From fidelity to p



1. Take GST matrices
2. They are actually PTM matrices
3. Find Kraus Operators
4. Write Kraus operators in terms of Pauli Matrices
5. Write final density matrices

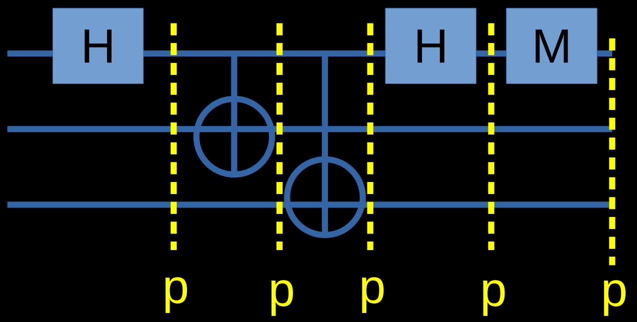
As a result, we analysed the succession/perfection rate in terms of accumulated error in the stochastic Pauli channels

- Each gate is created with 99.43% fidelity
- Phase flip errors are used to simulate process matrices - 0.0085

Surface codes: Towards practical large-scale quantum computation, Fowler et all,Phys. Rev. A 86, 032324

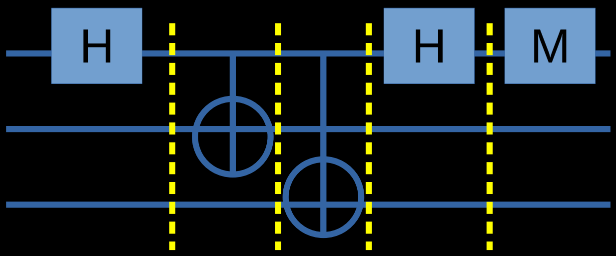
3. CURRENT METHODS

From fidelity to p



1. Take GST matrices
2. They are actually PTM matrices
3. Find Kraus Operators
4. Write Kraus operators in terms of Pauli Matrices
5. Write final density matrices

As a result, we analysed the succession/perfection rate in terms of accumulated error in the stochastic Pauli channels

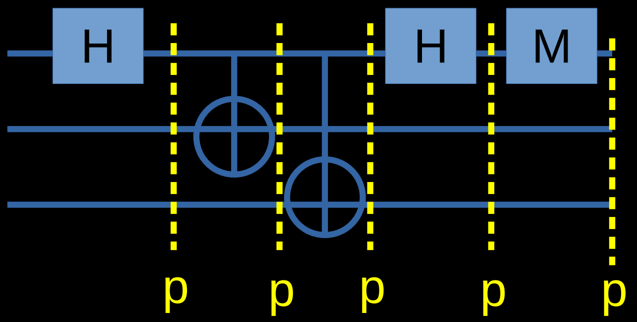


- Each gate is created with 99.43% fidelity
- Phase flip errors are used to simulate process matrices - 0.0085

Surface codes: Towards practical large-scale quantum computation, Fowler et all,Phys. Rev. A 86, 032324

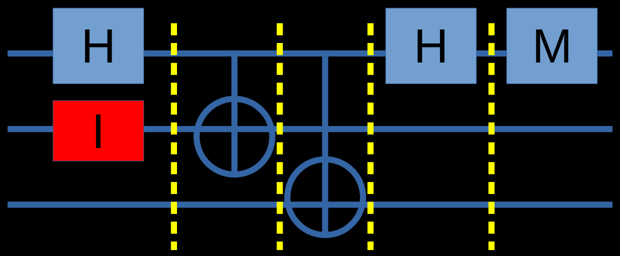
3. CURRENT METHODS

From fidelity to p



1. Take GST matrices
2. They are actually PTM matrices
3. Find Kraus Operators
4. Write Kraus operators in terms of Pauli Matrices
5. Write final density matrices

As a result, we analysed the succession/perfection rate in terms of accumulated error in the stochastic Pauli channels

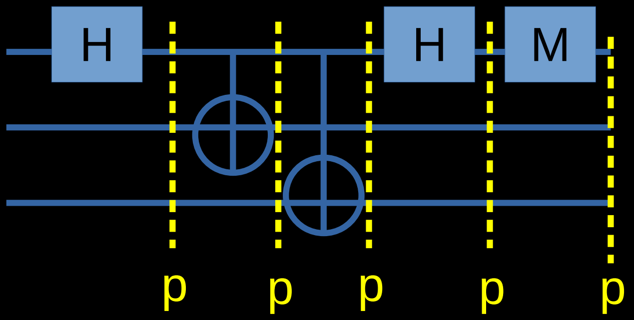


- Each gate is created with 99.43% fidelity
- Phase flip errors are used to simulate process matrices - 0.0085

Surface codes: Towards practical large-scale quantum computation, Fowler et all,Phys. Rev. A 86, 032324

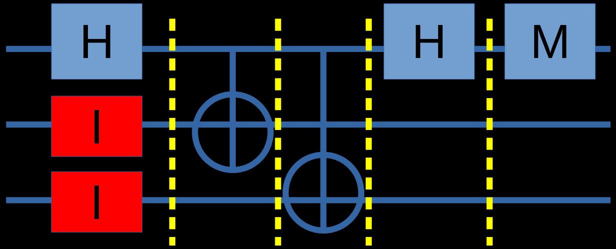
3. CURRENT METHODS

From fidelity to p



1. Take GST matrices
2. They are actually PTM matrices
3. Find Kraus Operators
4. Write Kraus operators in terms of Pauli Matrices
5. Write final density matrices

As a result, we analysed the succession/perfection rate in terms of accumulated error in the stochastic Pauli channels

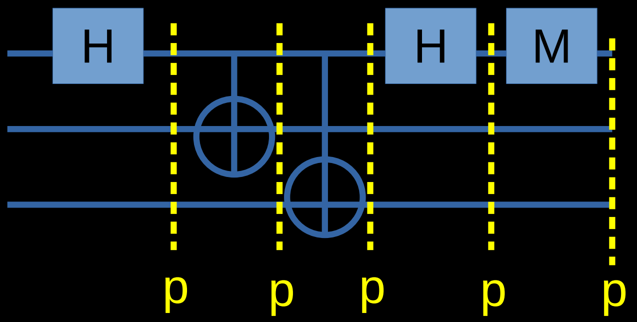


- Each gate is created with 99.43% fidelity
- Phase flip errors are used to simulate process matrices - 0.0085

Surface codes: Towards practical large-scale quantum computation, Fowler et all,Phys. Rev. A 86, 032324

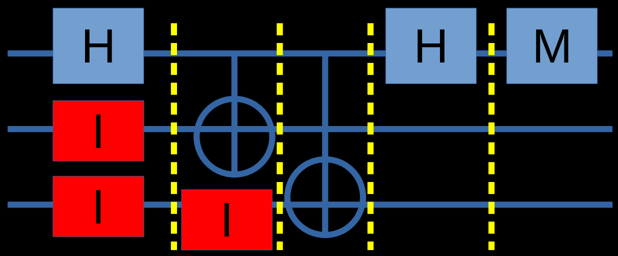
3. CURRENT METHODS

From fidelity to p



1. Take GST matrices
2. They are actually PTM matrices
3. Find Kraus Operators
4. Write Kraus operators in terms of Pauli Matrices
5. Write final density matrices

As a result, we analysed the succession/perfection rate in terms of accumulated error in the stochastic Pauli channels

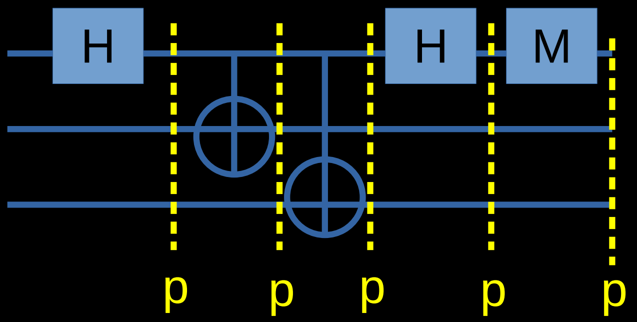


- Each gate is created with 99.43% fidelity
- Phase flip errors are used to simulate process matrices - 0.0085

Surface codes: Towards practical large-scale quantum computation, Fowler et all,Phys. Rev. A 86, 032324

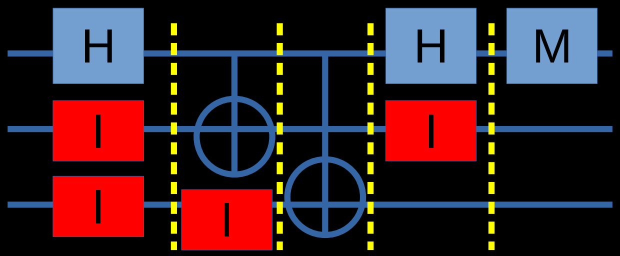
3. CURRENT METHODS

From fidelity to p



1. Take GST matrices
2. They are actually PTM matrices
3. Find Kraus Operators
4. Write Kraus operators in terms of Pauli Matrices
5. Write final density matrices

As a result, we analysed the succession/perfection rate in terms of accumulated error in the stochastic Pauli channels

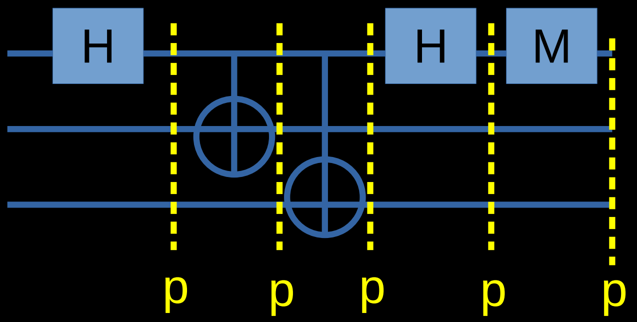


- Each gate is created with 99.43% fidelity
- Phase flip errors are used to simulate process matrices - 0.0085

Surface codes: Towards practical large-scale quantum computation, Fowler et all,Phys. Rev. A 86, 032324

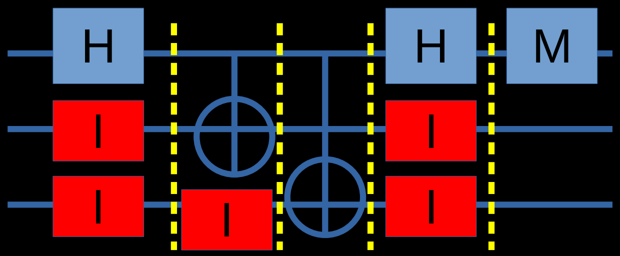
3. CURRENT METHODS

From fidelity to p



1. Take GST matrices
2. They are actually PTM matrices
3. Find Kraus Operators
4. Write Kraus operators in terms of Pauli Matrices
5. Write final density matrices

As a result, we analysed the succession/perfection rate in terms of accumulated error in the stochastic Pauli channels

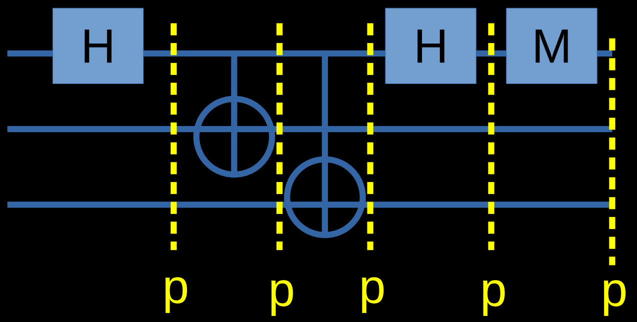


- Each gate is created with 99.43% fidelity
- Phase flip errors are used to simulate process matrices - 0.0085

Surface codes: Towards practical large-scale quantum computation, Fowler et all,Phys. Rev. A 86, 032324

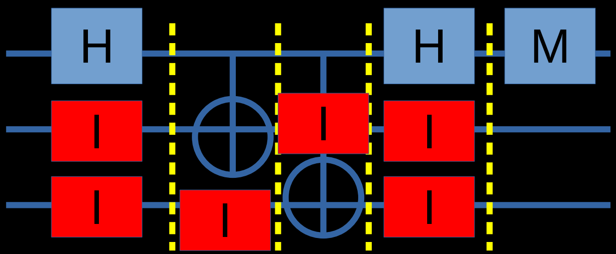
3. CURRENT METHODS

From fidelity to p



1. Take GST matrices
2. They are actually PTM matrices
3. Find Kraus Operators
4. Write Kraus operators in terms of Pauli Matrices
5. Write final density matrices

As a result, we analysed the succession/perfection rate in terms of accumulated error in the stochastic Pauli channels

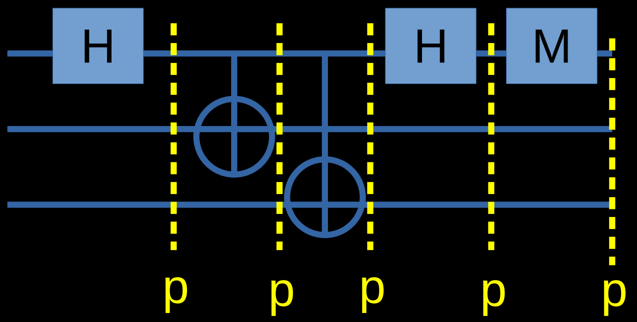


- Each gate is created with 99.43% fidelity
- Phase flip errors are used to simulate process matrices - 0.0085

Surface codes: Towards practical large-scale quantum computation, Fowler et all,Phys. Rev. A 86, 032324

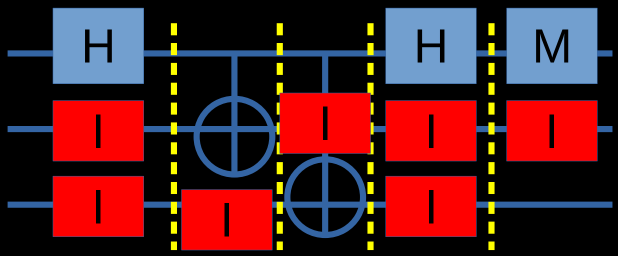
3. CURRENT METHODS

From fidelity to p



1. Take GST matrices
2. They are actually PTM matrices
3. Find Kraus Operators
4. Write Kraus operators in terms of Pauli Matrices
5. Write final density matrices

As a result, we analysed the succession/perfection rate in terms of accumulated error in the stochastic Pauli channels

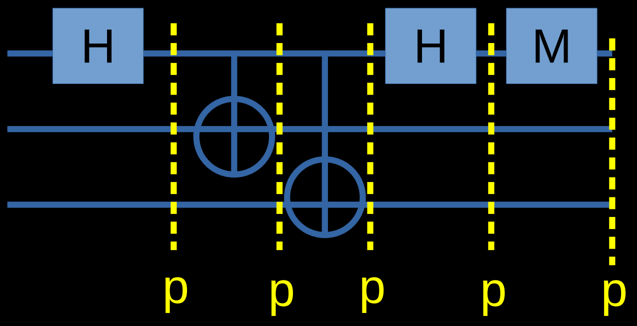


- Each gate is created with 99.43% fidelity
- Phase flip errors are used to simulate process matrices - 0.0085

Surface codes: Towards practical large-scale quantum computation, Fowler et all,Phys. Rev. A 86, 032324

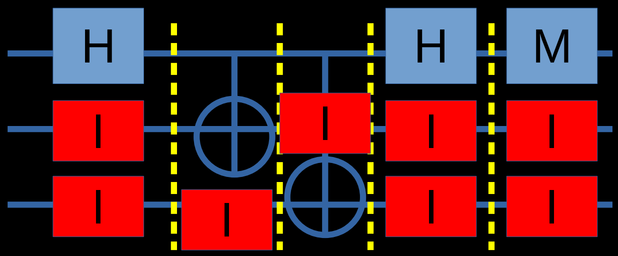
3. CURRENT METHODS

From fidelity to p



1. Take GST matrices
2. They are actually PTM matrices
3. Find Kraus Operators
4. Write Kraus operators in terms of Pauli Matrices
5. Write final density matrices

As a result, we analysed the succession/perfection rate in terms of accumulated error in the stochastic Pauli channels

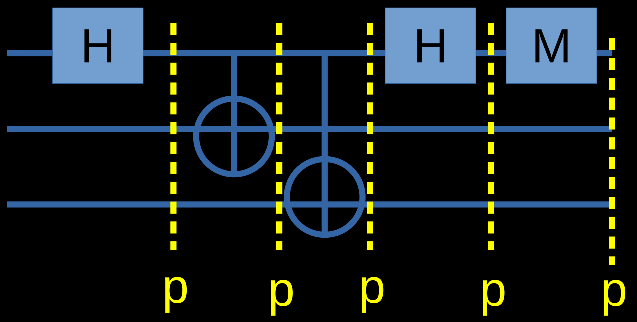


- Each gate is created with 99.43% fidelity
- Phase flip errors are used to simulate process matrices - 0.0085

Surface codes: Towards practical large-scale quantum computation, Fowler et all,Phys. Rev. A 86, 032324

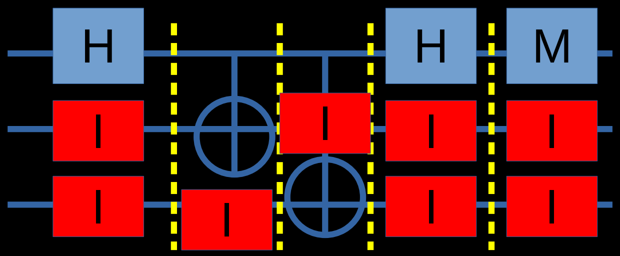
3. CURRENT METHODS

From fidelity to p



1. Take GST matrices
2. They are actually PTM matrices
3. Find Kraus Operators
4. Write Kraus operators in terms of Pauli Matrices
5. Write final density matrices

As a result, we analysed the succession/perfection rate in terms of accumulated error in the stochastic Pauli channels



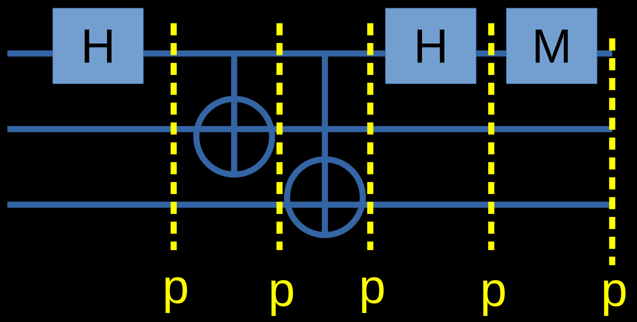
$P_I = 0.974$

- Each gate is created with 99.43% fidelity
- Phase flip errors are used to simulate process matrices - 0.0085

Surface codes: Towards practical large-scale quantum computation, Fowler et all,Phys. Rev. A 86, 032324

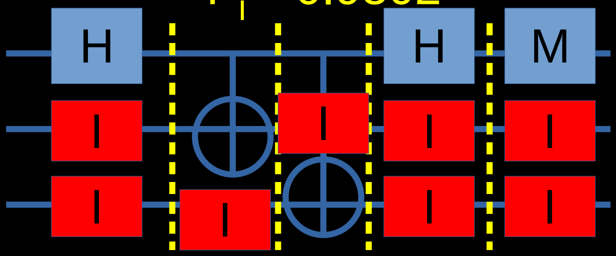
3. CURRENT METHODS

From fidelity to p



1. Take GST matrices
2. They are actually PTM matrices
3. Find Kraus Operators
4. Write Kraus operators in terms of Pauli Matrices
5. Write final density matrices

As a result, we analysed the succession/perfection rate in terms of accumulated error in the stochastic Pauli channels

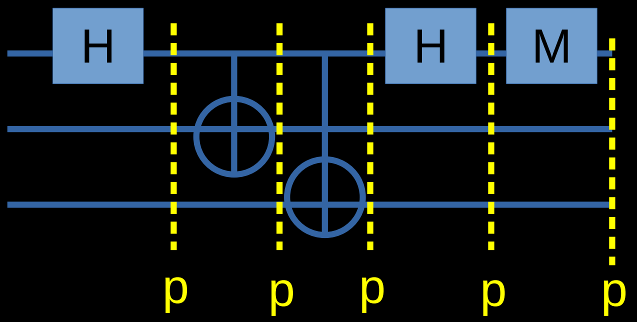


- Each gate is created with 99.43% fidelity
- Phase flip errors are used to simulate process matrices - 0.0085

Surface codes: Towards practical large-scale quantum computation, Fowler et all,Phys. Rev. A 86, 032324

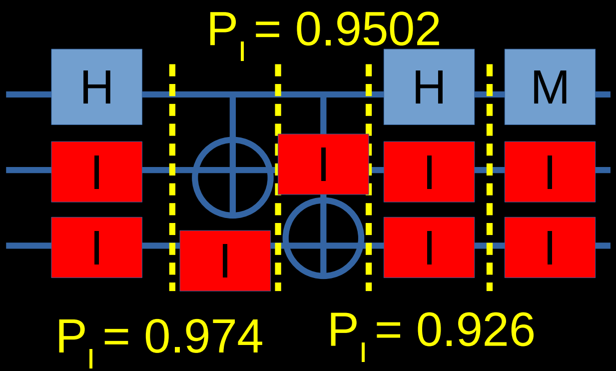
3. CURRENT METHODS

From fidelity to p



1. Take GST matrices
2. They are actually PTM matrices
3. Find Kraus Operators
4. Write Kraus operators in terms of Pauli Matrices
5. Write final density matrices

As a result, we analysed the succession/perfection rate in terms of accumulated error in the stochastic Pauli channels

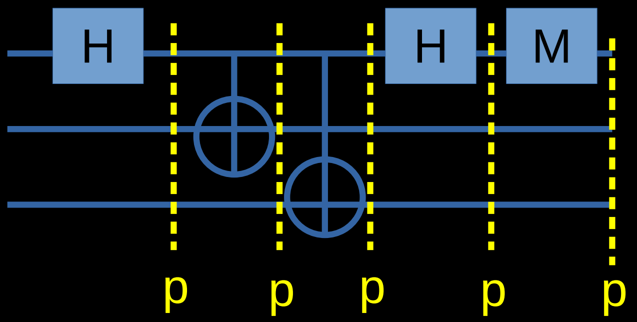


- Each gate is created with 99.43% fidelity
- Phase flip errors are used to simulate process matrices - 0.0085

Surface codes: Towards practical large-scale quantum computation, Fowler et all,Phys. Rev. A 86, 032324

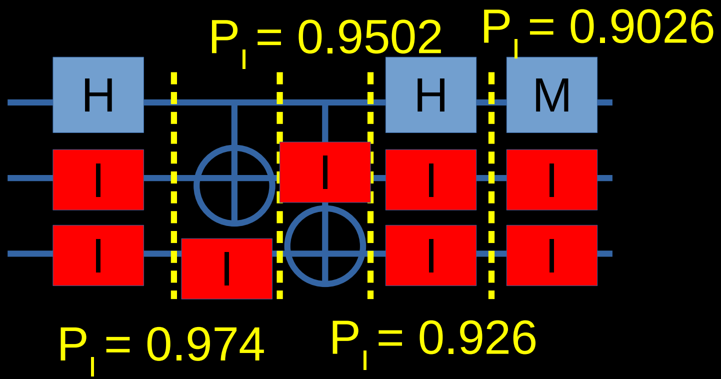
3. CURRENT METHODS

From fidelity to p



1. Take GST matrices
2. They are actually PTM matrices
3. Find Kraus Operators
4. Write Kraus operators in terms of Pauli Matrices
5. Write final density matrices

As a result, we analysed the succession/perfection rate in terms of accumulated error in the stochastic Pauli channels



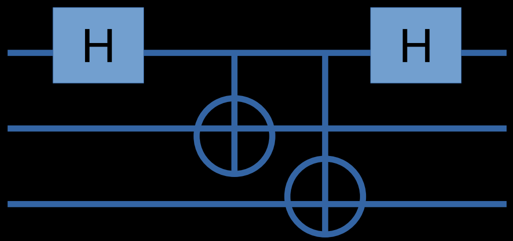
- Each gate is created with 99.43% fidelity
- Phase flip errors are used to simulate process matrices - 0.0085

Surface codes: Towards practical large-scale quantum computation, Fowler et all,Phys. Rev. A 86, 032324

CAN WE FIND A BETTER WAY?

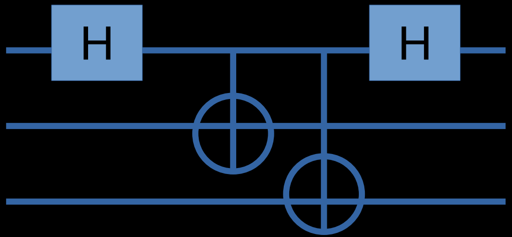
4. SINGLE STEP PARITY CHECK GATE SET

4. SINGLE STEP PARITY CHECK GATE SET

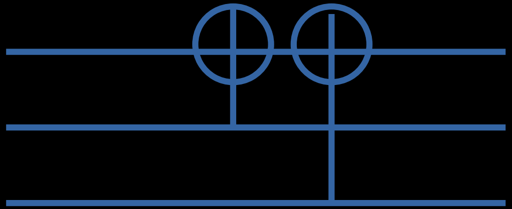


XX parity check circuit

4. SINGLE STEP PARITY CHECK GATE SET

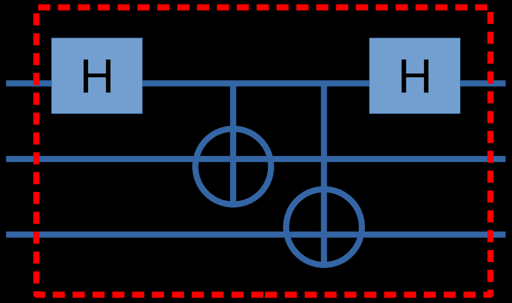


XX parity check circuit

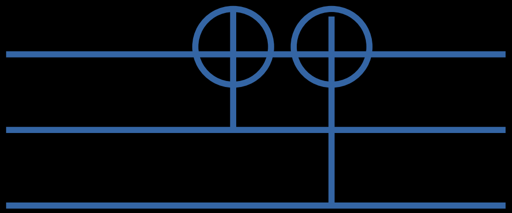


ZZ parity check circuit

4. SINGLE STEP PARITY CHECK GATE SET

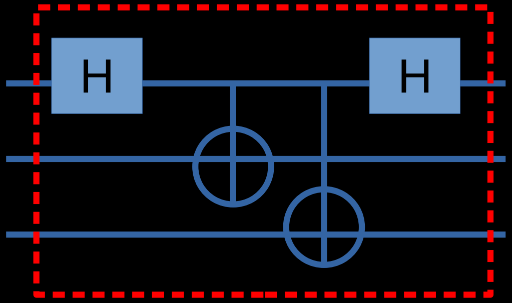


XX parity check circuit

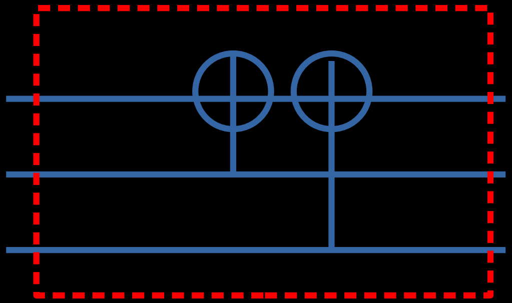


ZZ parity check circuit

4. SINGLE STEP PARITY CHECK GATE SET

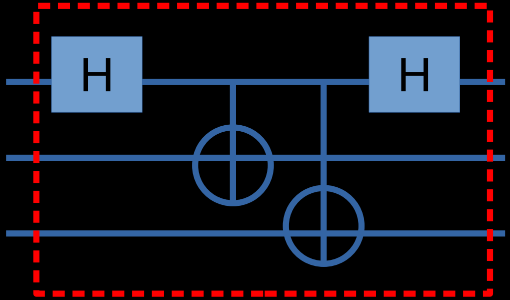


XX parity check circuit

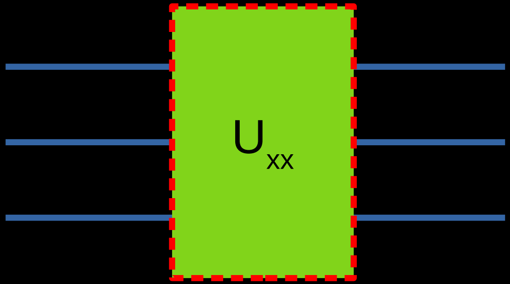


ZZ parity check circuit

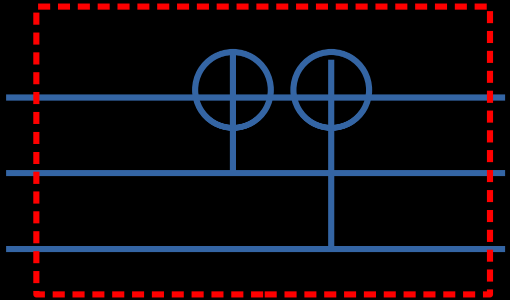
4. SINGLE STEP PARITY CHECK GATE SET



=

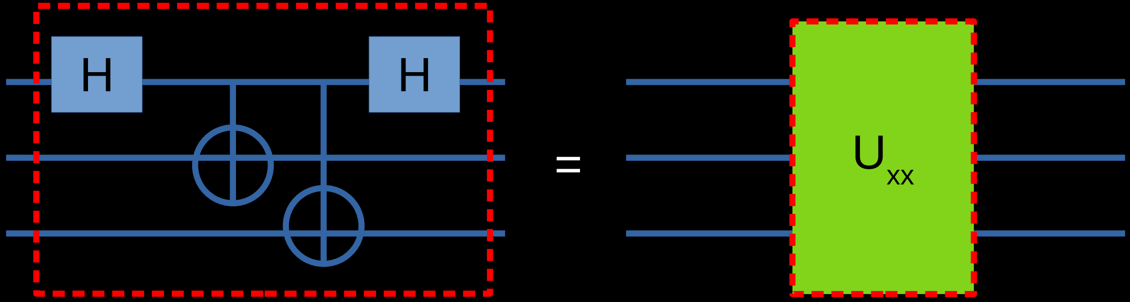


XX parity check circuit

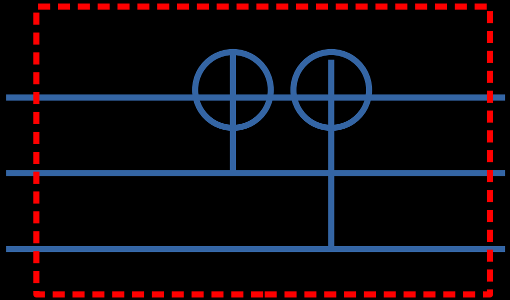


ZZ parity check circuit

4. SINGLE STEP PARITY CHECK GATE SET

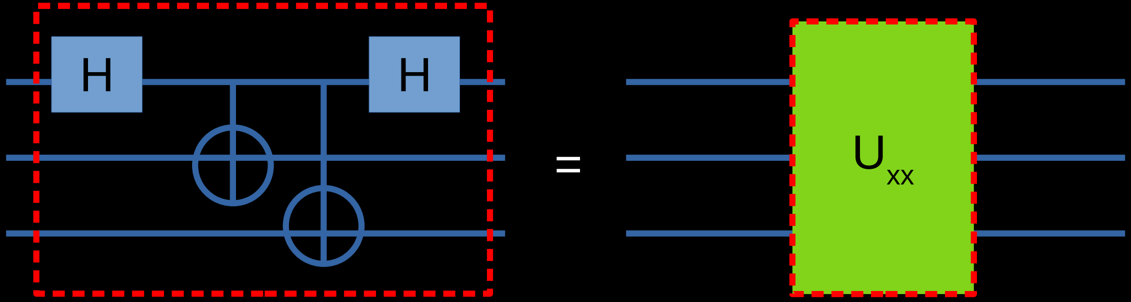


XX parity check circuit

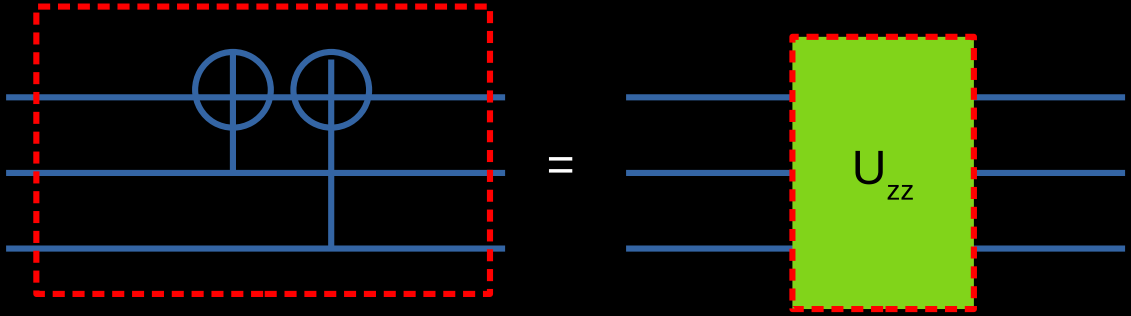


ZZ parity check circuit

4. SINGLE STEP PARITY CHECK GATE SET

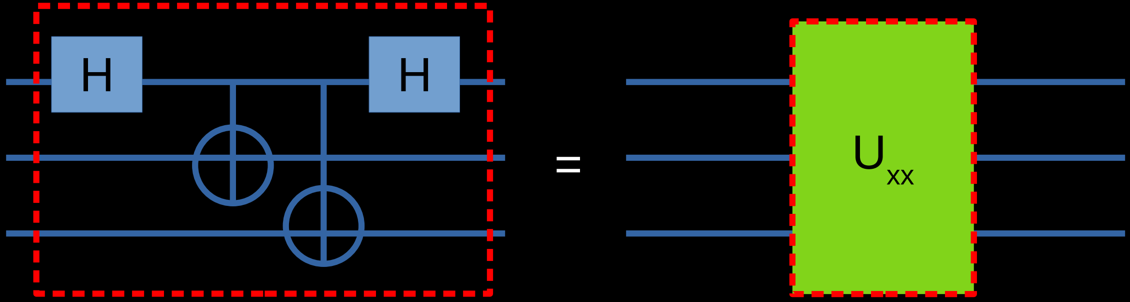


XX parity check circuit



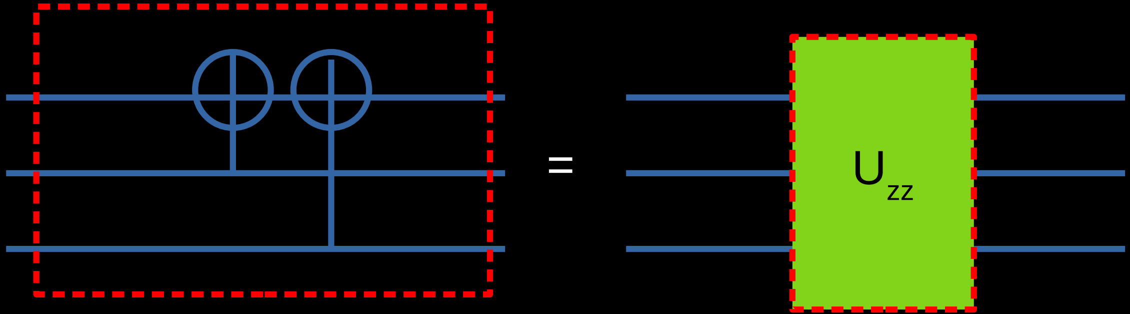
ZZ parity check circuit

4. SINGLE STEP PARITY CHECK GATE SET



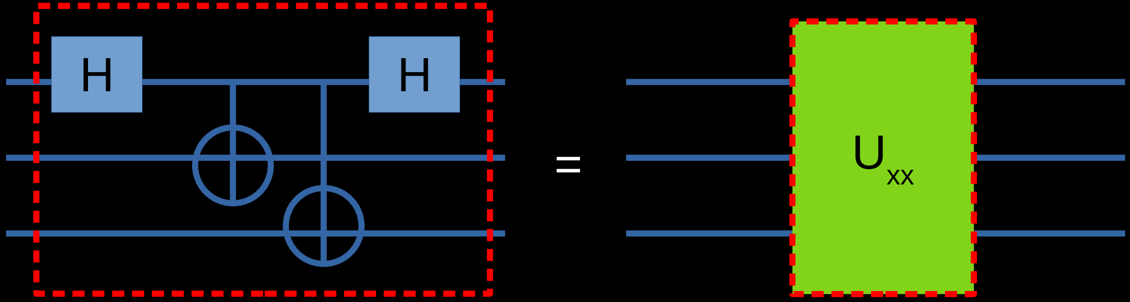
= $$U_{XX} = \frac{1}{2}\begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 0 & -1 \\ 0 & 1 & 1 & 0 & 0 & 1 & -1 & 0 \\ 0 & 1 & 1 & 0 & 0 & -1 & 1 & 0 \\ 1 & 0 & 0 & 1 & -1 & 0 & 0 & 1 \\ 1 & 0 & 0 & -1 & 1 & 0 & 0 & 1 \\ 0 & 1 & -1 & 0 & 0 & 1 & 1 & 0 \\ 0 & -1 & 1 & 0 & 0 & 1 & 1 & 0 \\ -1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$

XX parity check circuit



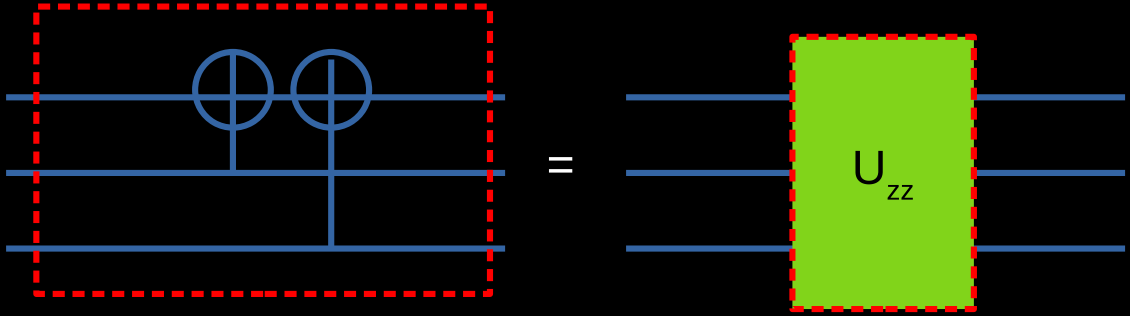
ZZ parity check circuit

4. SINGLE STEP PARITY CHECK GATE SET



$$= U_{XX} = \frac{1}{2}\begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 0 & -1 \\ 0 & 1 & 1 & 0 & 0 & 1 & -1 & 0 \\ 0 & 1 & 1 & 0 & 0 & -1 & 1 & 0 \\ 1 & 0 & 0 & 1 & -1 & 0 & 0 & 1 \\ 1 & 0 & 0 & -1 & 1 & 0 & 0 & 1 \\ 0 & 1 & -1 & 0 & 0 & 1 & 1 & 0 \\ 0 & -1 & 1 & 0 & 0 & 1 & 1 & 0 \\ -1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$

XX parity check circuit

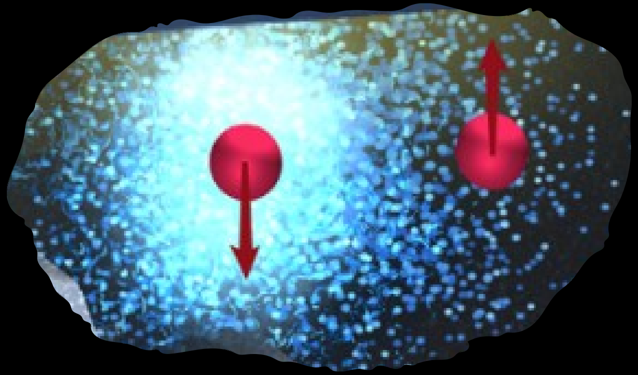


$$= U_{ZZ} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

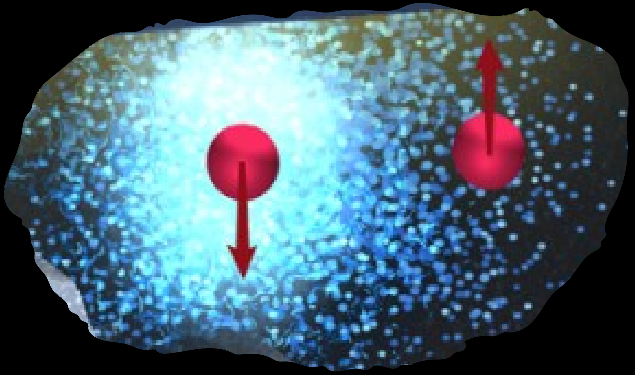
ZZ parity check circuit

5. FEASIBILITY – AN EXAMPLE

5. FEASIBILITY – AN EXAMPLE

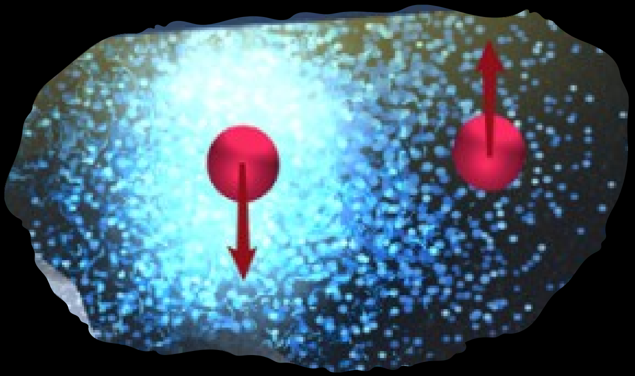


5. FEASIBILITY – AN EXAMPLE

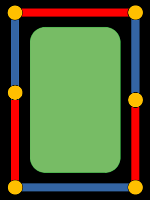


Precision tomography of a three-qubit donor quantum processor in silicon by Madzik & Asaad et all, Nature. 2022 Jan;601(7893):348-353.

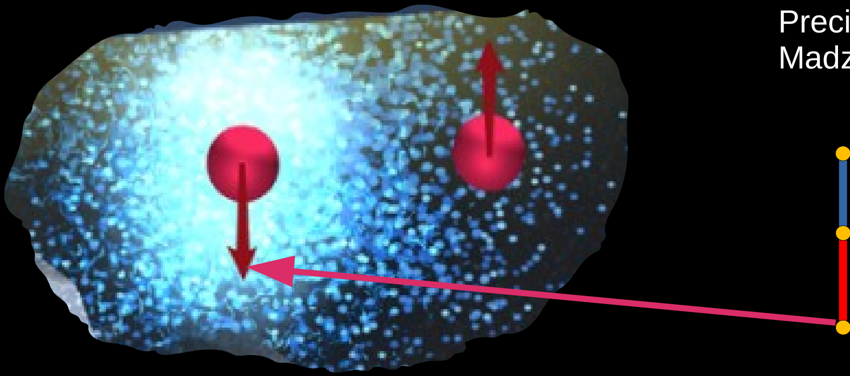
5. FEASIBILITY – AN EXAMPLE



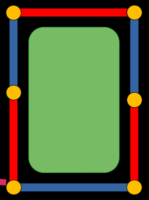
Precision tomography of a three-qubit donor quantum processor in silicon by Madzik & Asaad et all, Nature. 2022 Jan;601(7893):348-353.



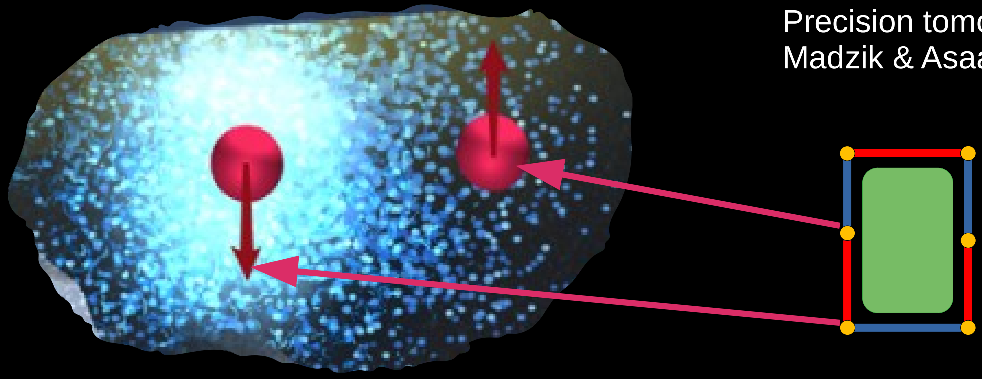
5. FEASIBILITY – AN EXAMPLE



Precision tomography of a three-qubit donor quantum processor in silicon by Madzik & Asaad et all, Nature. 2022 Jan;601(7893):348-353.

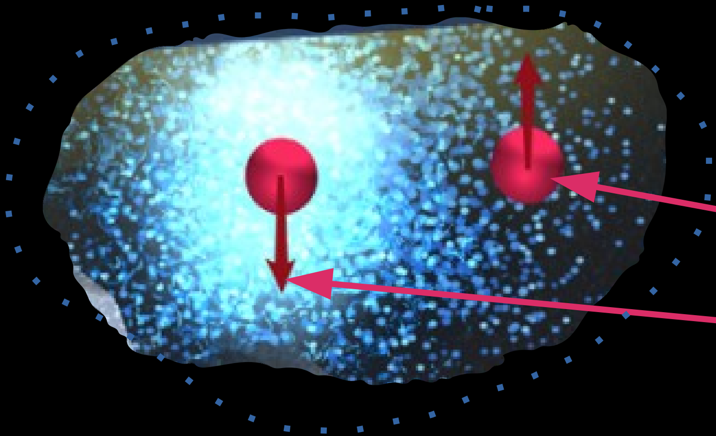


5. FEASIBILITY – AN EXAMPLE

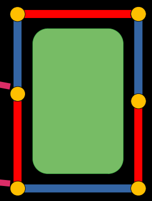


Precision tomography of a three-qubit donor quantum processor in silicon by Madzik & Asaad et all, Nature. 2022 Jan;601(7893):348-353.

5. FEASIBILITY – AN EXAMPLE

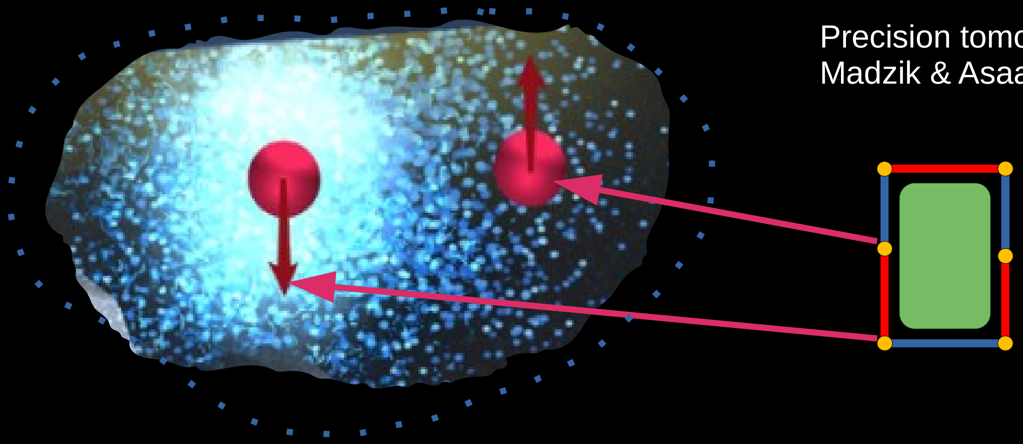


Precision tomography of a three-qubit donor quantum processor in silicon by Madzik & Asaad et all, Nature. 2022 Jan;601(7893):348-353.



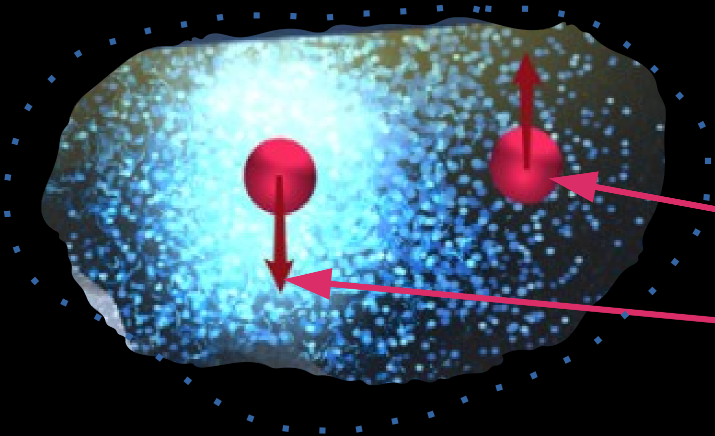
5. FEASIBILITY – AN EXAMPLE

Precision tomography of a three-qubit donor quantum processor in silicon by Madzik & Asaad et all, Nature. 2022 Jan;601(7893):348-353.

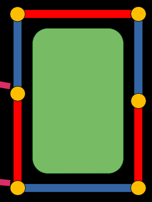


$$\vec{H}_d = -\gamma_e B_0 \hat{S} - \gamma_n B_0 (\hat{I_{z1}} + \hat{I_{z2}}) + A_1 \vec{S}.\vec{I_1} + A_2 \vec{S}.\vec{I_2}$$

5. FEASIBILITY – AN EXAMPLE



Precision tomography of a three-qubit donor quantum processor in silicon by Madzik & Asaad et all, Nature. 2022 Jan;601(7893):348-353.



$$\vec{H}_d = -\gamma_e B_0 \hat{S} - \gamma_n B_0 (\hat{I_{z1}} + \hat{I_{z2}}) + A_1 \vec{S}.\vec{I_1} + A_2 \vec{S}.\vec{I_2}$$

$$H_{rf}(t) = -\gamma_e \vec{B_1} \vec{S} sin(wt) - \gamma_n \vec{B_1} (\vec{I_1} + \vec{I_2}) sin(wt)$$

Method 2: Analytical Analysis of Hamiltonian + Numerical simulation

Method 2: Analytical Analysis of Hamiltonian + Numerical simulation

Method 2: Analytical Analysis of Hamiltonian + Numerical simulation → Clock Dynamics of Hamiltonian + GRAPE algorithm

Method 2: Analytical Analysis of Hamiltonian + Numerical simulation

→ Clock Dynamics of Hamiltonian + GRAPE algorithm

$$\vec{H} = -\gamma_e B_0 S_z - \gamma_n B_0 I_{z1} - \gamma_n B_0 I_{z2} + A_1 S_z I_{z1} + A_2 S_z I_{z2}$$

Method 2: Analytical Analysis of Hamiltonian + Numerical simulation → Clock Dynamics of Hamiltonian + GRAPE algorithm

$$\vec{H} = -\gamma_e B_0 S_z - \gamma_n B_0 I_{z1} - \gamma_n B_0 I_{z2} + A_1 S_z I_{z1} + A_2 S_z I_{z2}$$

$$U = e^{iHt}$$

Method 2: Analytical Analysis of Hamiltonian + Numerical simulation → Clock Dynamics of Hamiltonian + GRAPE algorithm

$$\vec{H} = -\gamma_e B_0 S_z - \gamma_n B_0 I_{z1} - \gamma_n B_0 I_{z2} + A_1 S_z I_{z1} + A_2 S_z I_{z2}$$

$$U = e^{iHt}$$

$$U = e^{-i\gamma_e B_0 S_Z t} e^{-i\gamma_n B_0 I_{z1} t} e^{-i\gamma_n B_0 I_{z2} t} e^{iA_1 S_z I_{z1} t} e^{iA_2 S_z I_{z2} t}$$

Method 2: Analytical Analysis of Hamiltonian + Numerical simulation

→ Clock Dynamics of Hamiltonian + GRAPE algorithm

$$\vec{H} = -\gamma_e B_0 S_z - \gamma_n B_0 I_{z1} - \gamma_n B_0 I_{z2} + A_1 S_z I_{z1} + A_2 S_z I_{z2}$$

$$U = e^{iHt}$$

$$U = e^{-i\gamma_e B_0 S_z t} e^{-i\gamma_n B_0 I_{z1} t} e^{-i\gamma_n B_0 I_{z2} t} e^{iA_1 S_z I_{z1} t} e^{iA_2 S_z I_{z2} t}$$

Gate_1 Gate_2 Gate_3 Gate_4 Gate_5

Method 2: Analytical Analysis of Hamiltonian + Numerical simulation

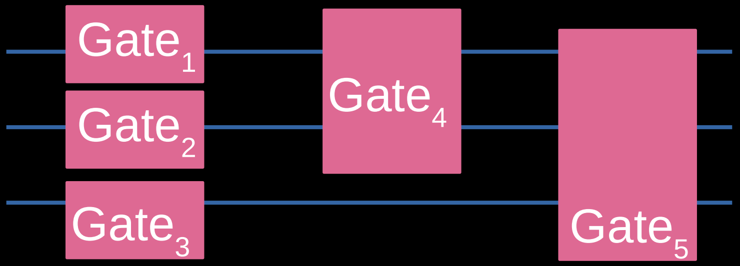
→ Clock Dynamics of Hamiltonian + GRAPE algorithm

$$\vec{H} = -\gamma_e B_0 S_z - \gamma_n B_0 I_{z1} - \gamma_n B_0 I_{z2} + A_1 S_z I_{z1} + A_2 S_z I_{z2}$$

$$U = e^{iHt}$$

$$U = e^{-i\gamma_e B_0 S_Z t}\, e^{-i\gamma_n B_0 I_{z1} t}\, e^{-i\gamma_n B_0 I_{z2} t}\, e^{iA_1 S_z I_{z1} t}\, e^{iA_2 S_z I_{z2} t}$$

Gate$_1$	Gate$_2$	Gate$_3$	Gate$_4$	Gate$_5$



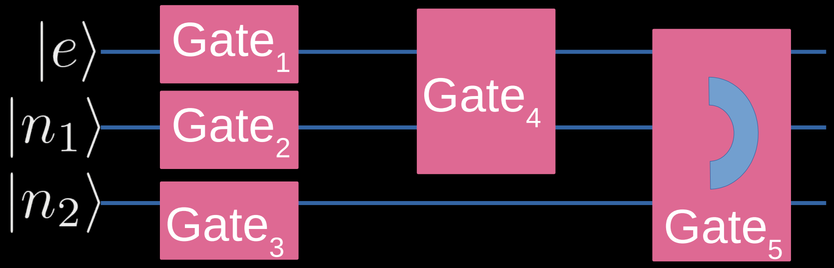
Method 2: Analytical Analysis of Hamiltonian + Numerical simulation → Clock Dynamics of Hamiltonian + GRAPE algorithm

$$\vec{H} = -\gamma_e B_0 S_z - \gamma_n B_0 I_{z1} - \gamma_n B_0 I_{z2} + A_1 S_z I_{z1} + A_2 S_z I_{z2}$$

$$U = e^{iHt}$$

$$U = e^{-i\gamma_e B_0 S_z t} e^{-i\gamma_n B_0 I_{z1} t} e^{-i\gamma_n B_0 I_{z2} t} e^{iA_1 S_z I_{z1} t} e^{iA_2 S_z I_{z2} t}$$

$Gate_1$ $Gate_2$ $Gate_3$ $Gate_4$ $Gate_5$



Method 2: Analytical Analysis of Hamiltonian + Numerical simulation

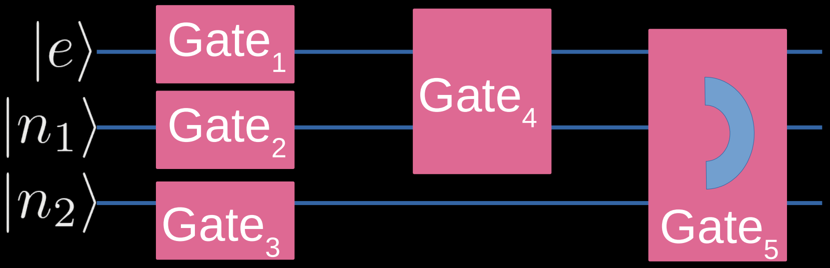
→ Clock Dynamics of Hamiltonian + GRAPE algorithm

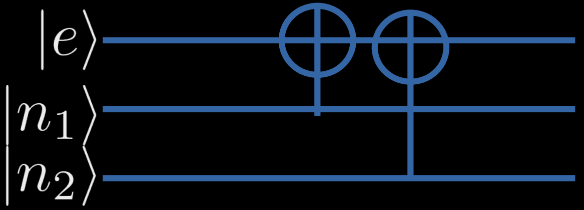
$$\vec{H} = -\gamma_e B_0 S_z - \gamma_n B_0 I_{z1} - \gamma_n B_0 I_{z2} + A_1 S_z I_{z1} + A_2 S_z I_{z2}$$

$$U = e^{iHt}$$

$$U = e^{-i\gamma_e B_0 S_z t} e^{-i\gamma_n B_0 I_{z1} t} e^{-i\gamma_n B_0 I_{z2} t} e^{iA_1 S_z I_{z1} t} e^{iA_2 S_z I_{z2} t}$$

$Gate_1$ $Gate_2$ $Gate_3$ $Gate_4$ $Gate_5$





Method 2: Analytical Analysis of Hamiltonian + Numerical simulation

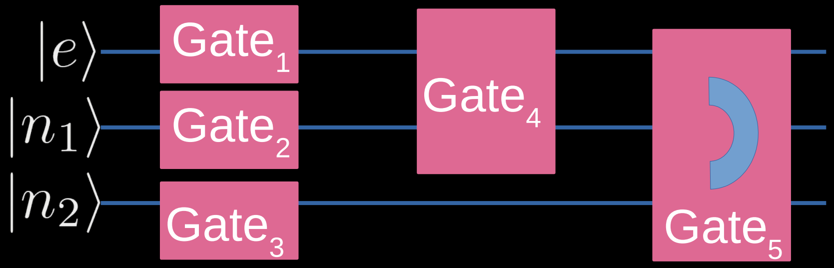
→ Clock Dynamics of Hamiltonian + GRAPE algorithm

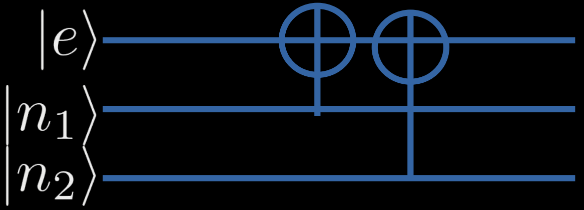
$$\vec{H} = -\gamma_e B_0 S_z - \gamma_n B_0 I_{z1} - \gamma_n B_0 I_{z2} + A_1 S_z I_{z1} + A_2 S_z I_{z2}$$

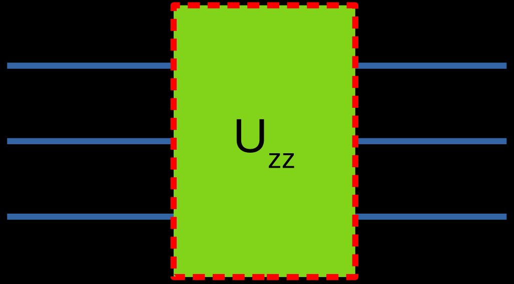
$$U = e^{iHt}$$

$$U = e^{-i\gamma_e B_0 S_z t}\, e^{-i\gamma_n B_0 I_{z1} t}\, e^{-i\gamma_n B_0 I_{z2} t}\, e^{iA_1 S_z I_{z1} t}\, e^{iA_2 S_z I_{z2} t}$$

Gate$_1$ Gate$_2$ Gate$_3$ Gate$_4$ Gate$_5$







- Clock dynamics analysis showed that we can implement U_{zz} gate ~9.5 times faster than traditional ZZ parity check circuit

- Clock dynamics analysis showed that we can implement U_{zz} gate ~9.5 times faster than traditional ZZ parity check circuit

  GRAPE:

- Clock dynamics analysis showed that we can implement U_{zz} gate ~9.5 times faster than traditional ZZ parity check circuit

GRAPE:

- Modulated control Hamiltonian

- Clock dynamics analysis showed that we can implement U_{zz} gate ~9.5 times faster than traditional ZZ parity check circuit

GRAPE:

- Modulated control Hamiltonian
- Drift Hamiltonian

- Clock dynamics analysis showed that we can implement U_{zz} gate ~9.5 times faster than traditional ZZ parity check circuit

GRAPE:

- Modulated control Hamiltonian
- Drift Hamiltonian
- Target unitary

- Clock dynamics analysis showed that we can implement U_{zz} gate ~9.5 times faster than traditional ZZ parity check circuit

GRAPE:
- Modulated control Hamiltonian
- Drift Hamiltonian
- Target unitary
- Time resolution

- Clock dynamics analysis showed that we can implement U_{zz} gate ~9.5 times faster than traditional ZZ parity check circuit

GRAPE:

- Modulated control Hamiltonian
- Drift Hamiltonian
- Target unitary
- Time resolution
- Initial state

- Clock dynamics analysis showed that we can implement U_{zz} gate ~9.5 times faster than traditional ZZ parity check circuit

  GRAPE:
  - Modulated control Hamiltonian
  - Drift Hamiltonian
  - Target unitary
  - Time resolution
  - Initial state
  - Gate evolution time

- Clock dynamics analysis showed that we can implement U_{zz} gate ~9.5 times faster than traditional ZZ parity check circuit

  GRAPE:
  - Modulated control Hamiltonian
  - Drift Hamiltonian
  - Target unitary
  - Time resolution
  - Initial state
  - Gate evolution time
  - Min – max control amplitude

- Clock dynamics analysis showed that we can implement U_{zz} gate ~9.5 times faster than traditional ZZ parity check circuit

GRAPE:
- Modulated control Hamiltonian
- Drift Hamiltonian
- Target unitary
- Time resolution
- Initial state
- Gate evolution time
- Min – max control amplitude
- Etc...

- Clock dynamics analysis showed that we can implement U_{zz} gate ~9.5 times faster than traditional ZZ parity check circuit

  GRAPE:
  - Modulated control Hamiltonian
  - Drift Hamiltonian
  - Target unitary
  - Time resolution
  - Initial state
  - Gate evolution time
  - Min – max control amplitude
  - Etc...
- GRAPE found single step parity check gate

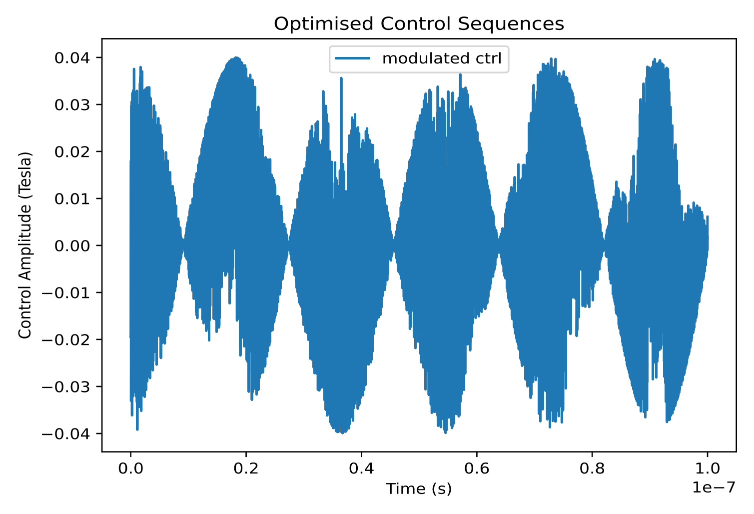
- Clock dynamics analysis showed that we can implement U_{zz} gate ~9.5 times faster than traditional ZZ parity check circuit

GRAPE:

- Modulated control Hamiltonian
- Drift Hamiltonian
- Target unitary
- Time resolution
- Initial state
- Gate evolution time
- Min – max control amplitude
- Etc...

- GRAPE found single step parity check gate with a 10 times shorter gate evolution time than the traditional parity check circuit

- Clock dynamics analysis showed that we can implement U_{zz} gate ~9.5 times faster than traditional ZZ parity check circuit



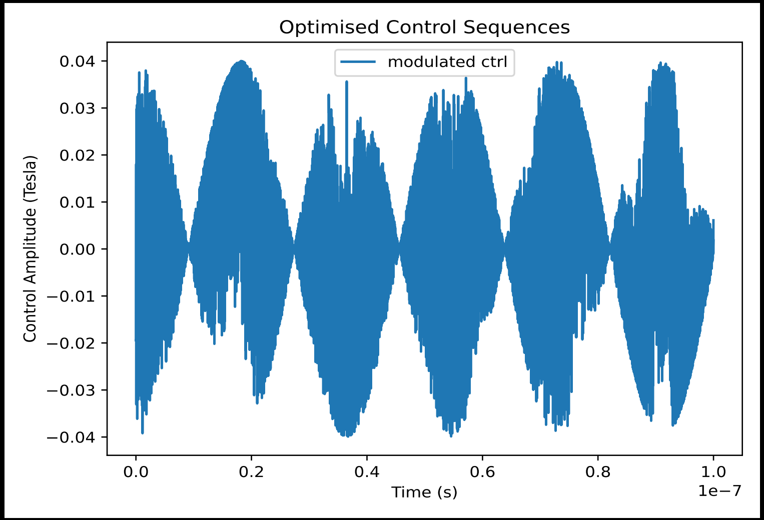
GRAPE:

- Modulated control Hamiltonian
- Drift Hamiltonian
- Target unitary
- Time resolution
- Initial state
- Gate evolution time
- Min – max control amplitude
- Etc...

- GRAPE found single step parity check gate with a 10 times shorter gate evolution time than the traditional parity check circuit

- Clock dynamics analysis showed that we can implement U_{zz} gate ~9.5 times faster than traditional ZZ parity check circuit

GRAPE:
- Modulated control Hamiltonian
- Drift Hamiltonian
- Target unitary
- Time resolution
- Initial state
- Gate evolution time
- Min – max control amplitude
- Etc...



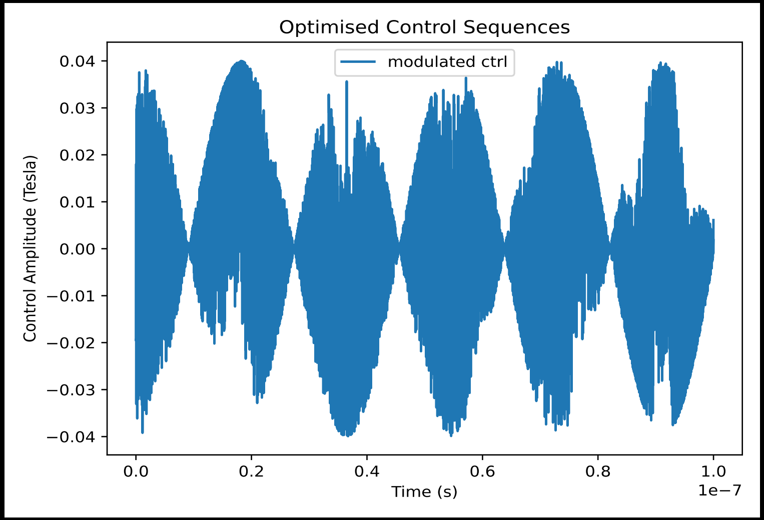
Simulated fidelity error is < 10^{-5}

- GRAPE found single step parity check gate with a 10 times shorter gate evolution time than the traditional parity check circuit

- Clock dynamics analysis showed that we can implement U_{ZZ} gate ~9.5 times faster than traditional ZZ parity check circuit

GRAPE:

- Modulated control Hamiltonian
- Drift Hamiltonian
- Target unitary
- Time resolution
- Initial state
- Gate evolution time
- Min – max control amplitude
- Etc...

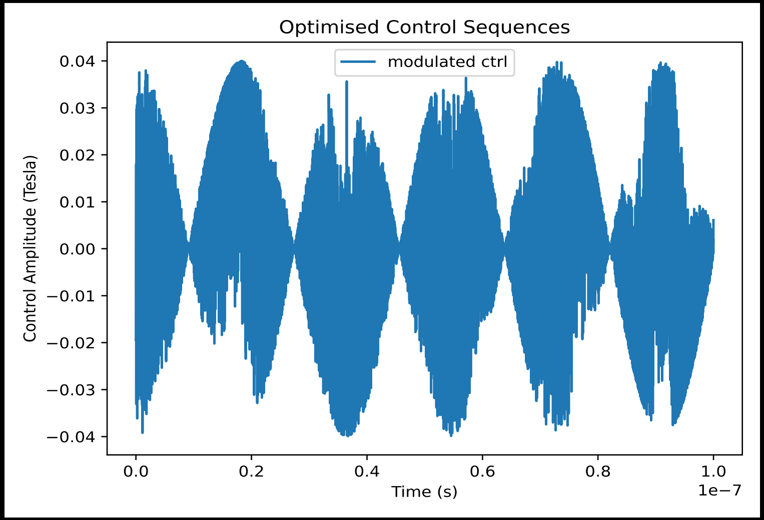


Simulated fidelity error is < 10^{-5}

- GRAPE found single step parity check gate with a 10 times shorter gate evolution time than the traditional parity check circuit
- Analytic solution and GRAPE matches

- Clock dynamics analysis showed that we can implement U_{zz} gate ~9.5 times faster than traditional ZZ parity check circuit

GRAPE:
- Modulated control Hamiltonian
- Drift Hamiltonian
- Target unitary
- Time resolution
- Initial state
- Gate evolution time
- Min – max control amplitude
- Etc...



Simulated fidelity error is $< 10^{-5}$

- GRAPE found single step parity check gate with a 10 times shorter gate evolution time than the traditional parity check circuit
- Analytic solution and GRAPE matches

Hence: M_{pp2} is one of single step parity check gate + a measurement

6. NEXT STEPS

6. NEXT STEPS

1. Experiment :)

6. NEXT STEPS

1. Experiment :)
2. Possibly more pulse generation in case of noise

6. NEXT STEPS

1. Experiment :)
2. Possibly more pulse generation in case of noise
3. Implementing the new gate set for different systems like antimony or 2P2e systems

7. CONCLUSION

7. CONCLUSION

1. We modified the universal gate set for Quantum Error Correction

7. CONCLUSION

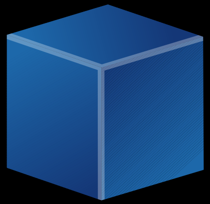
1. We modified the universal gate set for Quantum Error Correction

7. CONCLUSION

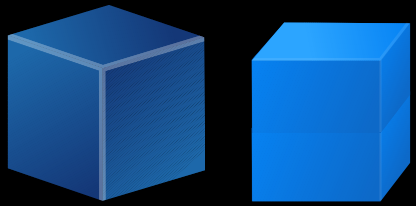
1. We modified the universal gate set for Quantum Error Correction

7. CONCLUSION

1. We modified the universal gate set for Quantum Error Correction

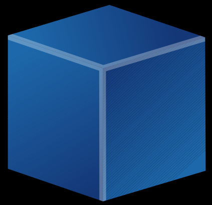


7. CONCLUSION



1. We modified the universal gate set for Quantum Error Correction
2. We found M_{pp2} :)

7. CONCLUSION

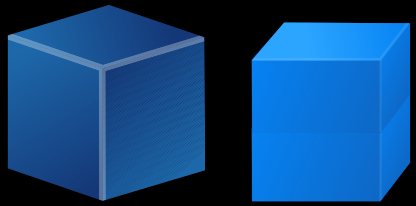



1. We modified the universal gate set for Quantum Error Correction
2. We found M_{pp2} :)
3. We analytically showed our systems are able to implement Single Step Parity Check Gate Set

7. CONCLUSION

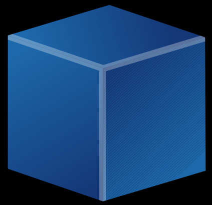
1. We modified the universal gate set for Quantum Error Correction
2. We found M_{pp2} :)
3. We analytically showed our systems are able to implement Single Step Parity Check Gate Set
4. We numerically showed that our systems are able to implement Single Step Parity Check Gate Set

7. CONCLUSION

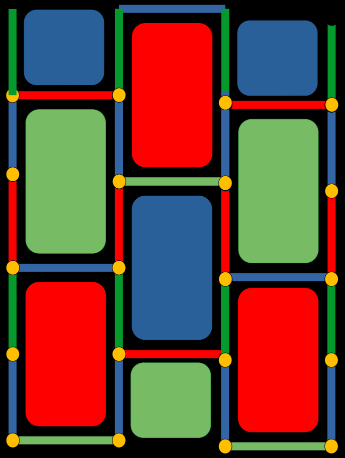


1. We modified the universal gate set for Quantum Error Correction
2. We found M_{pp2} :)
3. We analytically showed our systems are able to implement Single Step Parity Check Gate Set
4. We numerically showed that our systems are able to implement Single Step Parity Check Gate Set
5. The first milestone on the road of implementing Honeycomb code

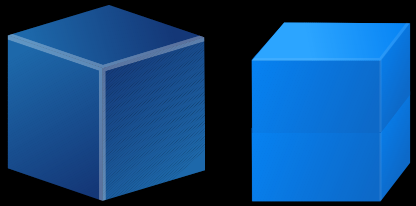
7. CONCLUSION



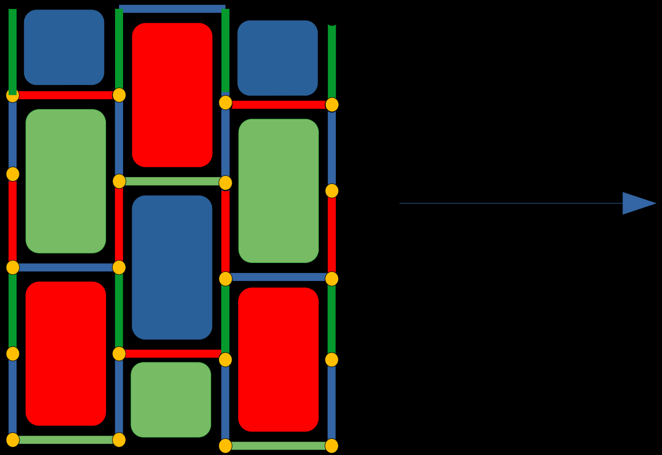
1. We modified the universal gate set for Quantum Error Correction
2. We found M_{pp2} :)
3. We analytically showed our systems are able to implement Single Step Parity Check Gate Set
4. We numerically showed that our systems are able to implement Single Step Parity Check Gate Set
5. The first milestone on the road of implementing Honeycomb code



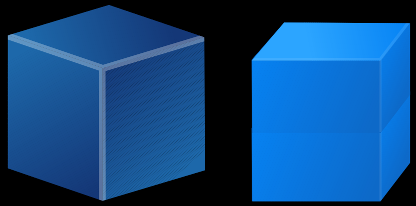
7. CONCLUSION



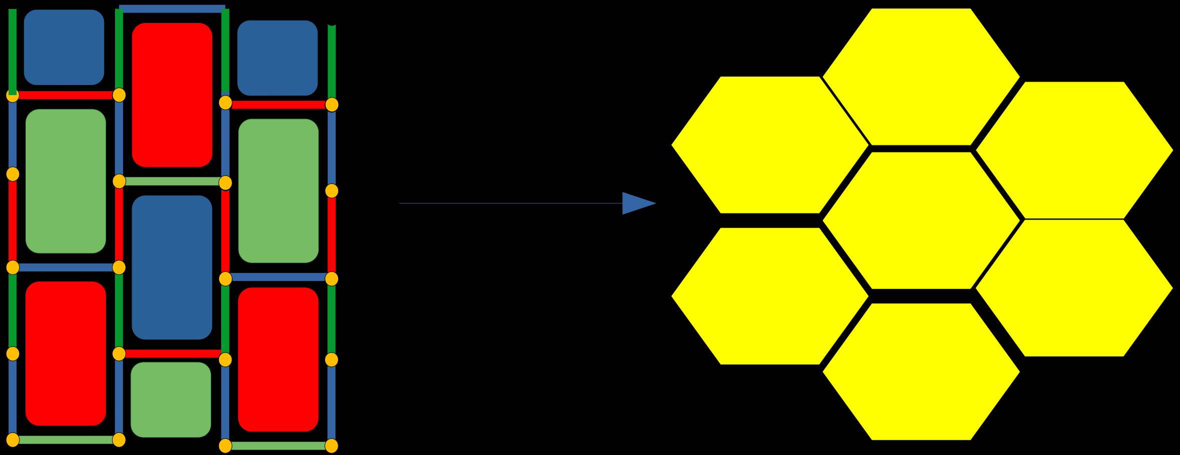
1. We modified the universal gate set for Quantum Error Correction
2. We found M_{pp2} :)
3. We analytically showed our systems are able to implement Single Step Parity Check Gate Set
4. We numerically showed that our systems are able to implement Single Step Parity Check Gate Set
5. The first milestone on the road of implementing Honeycomb code



7. CONCLUSION



1. We modified the universal gate set for Quantum Error Correction
2. We found M_{pp2} :)
3. We analytically showed our systems are able to implement Single Step Parity Check Gate Set
4. We numerically showed that our systems are able to implement Single Step Parity Check Gate Set
5. The first milestone on the road of implementing Honeycomb code



Prof. Andrea Morello

Prof. Andrea Morello

Prof. Andrea Morello

Dr. Simon Devitt

Prof. Andrea Morello

Dr. Simon Devitt

Prof. Andrea Morello

Dr. Simon Devitt

My lovely team :)

Prof. Andrea Morello

Dr. Simon Devitt

My lovely team :)

THANK YOU FOR LISTENING