# On the Addressability Problem in CSS Codes

Jérôme Guyot*, Samuel Jaques†
*Université Paris-Saclay, ENS Paris-Saclay, France
Email: jerome.guyot@ens-paris-saclay.fr
†University of Waterloo, Canada
Email: sejaques@uwaterloo.ca

***Abstract*—Quantum computers are highly susceptible to noise, necessitating error-correcting codes for reliable computation. Among these, Calderbank-Shor-Steane (CSS) codes provide a robust framework. However, the addressability problem—the ability to apply fault-tolerant operations to specific logical qubits—remains a critical challenge, especially in high-rate codes. This paper explores impossibility results for addressable gates in CSS codes and discusses potential paths forward, offering insights into the trade-offs between code performance and computational efficiency.**

## I. Motivation

Quantum computers are particularly vulnerable to noise, and so the most promising path to large-scale quantum computing is to use error-correcting codes. In these codes, many *physical* qubits are combined into one or more *logical* qubit(s), such that the logical qubits are long-lived and error-resistant.

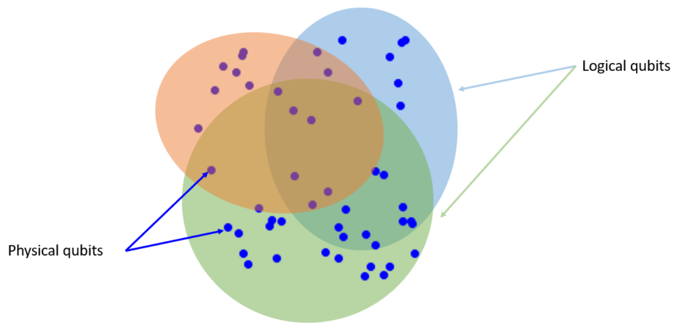


Fig. 1. Visualization of an error correcting code

A drawback of quantum error correction is that, by design, it becomes difficult to modify the encoded logical state. Unlike with classical error-correcting codes, we cannot decode the state to compute on it, as it is unlikely to remain coherent long enough for any operation. Thus, we need fault-tolerant quantum computation: not only should we have a method to encode the data, but we should also be able to operate on it *while* it is encoded.

A powerful tool in constructing fault-tolerant quantum computation is a *transversal* gate. Strictly speaking, this is any physical circuit that is guaranteed not to propagate errors between qubits of the code, and more commonly we require that it enacts some specific action on the logical state as well.

As an example, in a self-dual CSS code, applying an $H$ (Hadamard) gate to all physical qubits in the code will not only preserve the codespace, it will effectively apply an $H$ gate to all *logical* qubits in the code. However, in most quantum circuits we need more precision than this. We need to be able to apply an $H$ gate to *only one* qubit in the code. We need our fault-tolerant operations to be *addressable*.

This distinction does not matter for surface codes, which (depending on the precise description) encode only one logical qubit. A large-scale surface code computation is best seen as a *product* of codes, each working independently. Any transversal gate can be targeted to a single logical qubit (or pair of qubits for a CNOT) by simply applying the gates only to those physical qubits corresponding to the desired logical qubit.

However, this strategy fails for more complicated codes that encode many qubits, where the notion of a correspondence between physical qubits and logical qubits is ill-defined. For good performance, the logical qubits are not spatially localized in this way. If we apply a gate to one physical qubit, it will impact many logical qubits.

This addressability problem has become relevant recently, with the development of asymptotically good quantum codes [1], [2]. As the size of the code increases, the number of logical qubits in these codes approaches exactly the number of physical qubits, while maintaining good code distance, which means logical qubits cannot be spatially localized. Thus, it is a challenging problem to find physical circuits which can address specific logical qubits for some desired logical gate, and that is what we address in this paper. Furthermore, those asymptotically good codes belong to the family of CSS codes [3], hence we chose to restrict ourselves to CSS codes.

While considerable work has been done to find valid transversal implementations and study their actions on the code, very little has been done on the addressability problem. Addressability is mentioned when a transversal gate allows it, such as in [4], but is rarely a goal itself. In [5] the authors find implementations using state injection for some targeted CNOT and CZ and single qubit $H$ and $P$ gate, thus getting partial addressability results. Using completely different methods, [6] constructs explicit Clifford implementation of logical Clifford gates on hypergraph product codes, thus getting addressability on hypergraph product code. We also mention [7], which focuses on the implementation of targeted CCZ gates on sheaf codes.

## II. Methods

### A. Addressability

Let us first define a bit more formally what we mean by addressability. To illustrate the different definitions we will use a visual example where the meshed shapes correspond to the application of physical gates on physical qubits inside the shape. We also consider a unitary $U$ that sends respectively :

- $U($ orange $)=$ red
- $U($ green $)=$ pink
- $U($ blue $)=$ purple

We say that a circuit $G$ is targeting on $I$ if it acts only on the logical qubits that are in the subset $I$, see Fig. 2. Intuitively, for a $p$-qubits unitary $U$ to be addressable it means that we are able to implement the logical action $U$ on any subset of $p$ logical qubits. Then using the composition of the circuits, we would be able to apply $U$ in parallel on disjoint subsets.

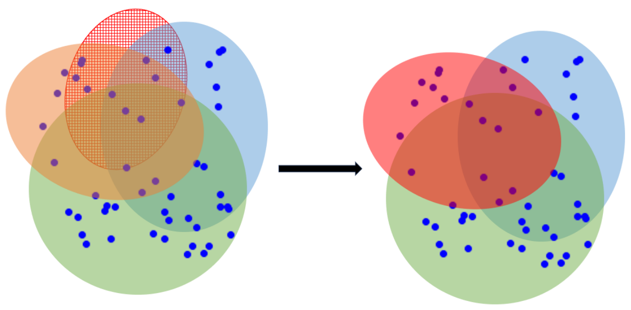


Fig. 2. Targeting circuit on $I=\{$ Orange logical qubit $\}$

**Definition 1.** *Addressability*

*A $p$-qubits unitary $U$ is addressable if for any subset of $p$ logical qubits, there is a targeting circuit on this subset, acting as $U$.*

*Furthermore, let $P$ be a property on circuits, we say that $U$ is $P$-addressable if we can implement those actions with circuits satisfying $P$.*

**Example 1.** *We could take $P$ to be :*

- *The circuit uses only Clifford gates.*
- *The circuit has depth less than $n$.*
- *The circuit act independently on all physical qubits.*
- *The circuit uses less than $n$ gates.*

Some of these properties $P$ (like depth less than $n$) are not closed under composition. To see why this matters, suppose we had an addressable $T$ gate and we wanted to apply $T$ gates to logical qubits 1 and 2. At the logical level, these gates *should* commute with each other and we should be able to apply them in parallel, but there is no guarantee that composing the physical circuits will preserve property $P$. That is, the depth of applying the circuit to two disjoint qubits might increase from applying it to just one. We thus give a stronger definition for when such parallelism is possible:

**Definition 2** (Full Addressability)**.** *Let $U$ be a $p$-qubit unitary, we say that $U$ is fully addressable on $C$ if for any disjoint union of subsets of $p$ logical qubits $I$, there exists a circuit $G$ implementing $U$ and targeting $I$.*

*Furthermore, let $P$ be a property on circuits, we say that $U$ is $P$-fully addressable if we can implement those actions with circuits satisfying $P$.*

The previous definitions of addressability required one to be able to target any subset of $p$ logical qubits. However, we are also interested in cases where we might not have all of those subsets, or maybe cases where we can only do two subsets together but not each alone. In fact, it is often easier to apply the same logical action to all logical qubits, and more generally, it might be easier to preserve the structure of the code by acting on multiple subsets of $p$ logical qubits. Hence we need to capture the cases where we are able to apply some logical $U$ on disjoint subsets at the same time, but we do not want to apply on all logical qubits, as this has no property of targeting some particular qubits.

**Definition 3** (Partial Addressability)**.** *Let $U$ be a $p$-qubit unitary, we say that $U$ is partially addressable on $C$ if there exists a disjoint union of subsets of $p$ logical qubits $I$ that is not $\emptyset$ or $[\![k]\!]$ such that there exists a circuit $G$ implementing $U$ and targeting $I$.*

*Furthermore, let $P$ be a property on circuits, we say that $U$ is $P$-partially addressable if we can implement this action with a circuit satisfying $P$.*

**Example 2.** *Fig. 2 shows that $U$ is partially addressable on this code since there is a targeting circuit implementing $U$ on $I$ ={Orange logical qubit}.*

We want to highlight that the study of addressability is only interesting when restricting the set of possible implementations, since we always have the (impractical) option to decode, apply the gates physically on the qubit we want, then encode again. Hence we will always consider some property $P$ and study the $P$-addressability.

In this work we mostly consider what we call single qubit Clifford addressability, which means that we take $P$ to be "The circuit is made of single qubit Clifford gates". We also consider circuits that act as permutations of physical qubits, which we call permutation addressability. However, other restrictions would be interesting for future work. For example we could try to consider bounded depth circuits, to capture which gates are “efficiently” addressable. We could also consider bounded width circuits, or bounded partition size (in the formal transversality definition) which would mean bounding the error propagation in the circuits. Finally, we might also care about the number of gates used, as each gate might be subject to errors, so fewer gates means a less faulty implementation.

**Lemma 1.** *Let $U$ be a $p$-qubits unitary and $P$ a property on circuit that is preserved by composition. Then $U$ is $P$-addressable iff $U$ if $P$-fully addressable.*

**Remark 1.** *For a property $P$ that is preserved by composition, we will just write $P$-addressability since the two definitions are equivalent. Since all the properties we consider in this work are preserved by composition, we will use this terminology.*

**Example 3.** *Fig. 3 illustrates that $U$ is addressable on the code since we can find a targeting circuits for all $I$ made of only 1 logical qubit. Using Lemma 1 we also get that $U$ is fully addressable.*

When defining addressability, we mentioned that the targeting circuit needs to act as some fixed unitary $U$ on the targeted logical qubits. However, the logical action of a physical circuit depends on the basis we are taking for the logicals. In order to have positive result about addressability, one would have to fix one basis of logicals. With a different basis, the same unitary might not have a transversal implementation. This poses little restriction for our impossibility results, as generally we show that a given circuit cannot preserve the code at all, regardless of its logical action.

A second issue with basis changes is that any gate can be implemented "in software" by changing the basis. That is, instead of performing a logical $H$ gate, we could re-define the basis of the code such that we swap $X$ and $Z$ stabilizers. Of course, this would require us to modify all future gates, equivalent to commuting the $H$ gate through all subsequent gates in the circuit. While this trick can be extremely useful (for example, all $X$ and $Z$ gates in the surface code are commuted in this way), we cannot efficiently compute this for *all* gates in a quantum circuit, or else quantum circuits would be efficiently classical simulatable! Thus, we restrict ourselves to a single basis.

## *B. Splitting codes*

We consider the problem of efficient physical circuits to enact logical gates addressably. As noted earlier, there are two trivial ways to make addressable gates: the first is to decode, apply the gate, then re-encode, and the second is to use a product of smaller codes which admit transversal gates.

The first trivial method is problematic because it is not fault tolerant. The quantum state is unprotected after decoding. Thus, we consider gate sets that will not alter the distance of code. We take three approaches for this: first, single qubit circuits; second, circuits made of SWAPs or other permutations; third, circuits made from a depth-1 CNOT circuit.

The second trivial method forbids us from having high-rate codes. One can readily see that if multiple codes are run in parallel, and we treat them as one larger code, the larger code's distance is at most the minimum distance from one of the subcodes. Thus, to have asymptotically good codes, they must not "split" into subcodes like this. Moreover, even if specific instances of the codes in [1], [2] turn out to split, they must contain a smaller non-splitting code in order to achieve the distances they do.

We formally define the notion of splitting as follows :

**Definition 4.** *Let $C = CSS(A, B)$, we say that $A$ splits on some non-empty support $h \subsetneq \{1, \dots, n\}$ if the basis of $A$ can be written up to permutation of the columns as $\begin{pmatrix} A_1 & 0 \\ 0 & A_2 \end{pmatrix}$ with $h$ being the support of $A_1$.*

*If $A$ and $B$ both split on some support $h$, we say that the code $C$ splits on $h$. This is equivalent to saying that $C$ splits into two independent codes $C_1, C_2$ where $C_1$ is made of the qubits in $h$ and $C_2$ of the rest of the qubits. In this case $C_1, C_2$ are both CSS codes.*

**Example 4.** *Let us highlight the splits of $A, B$ with red boxes and the splits of $C$ with green boxes.*

$$A = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

$$B = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

We also notice that if we take any CSS code and apply any circuit of single-qubit gates, we will obtain another code with the same distance and rate. However, this new code may not be easy to work with and it may not have any of its own efficient fault-tolerant operations. Thus, we further require that our circuits preserve the original code space.

Indeed, if we are willing to change the code, an even simpler way to apply any gate is to simply change the interpretation of the basis, which we already ruled out.

Ultimately, we are considering operators that preserve the code space, for CSS codes that do not split. This is actually enough to study the addressability on any CSS code as we can deduce addressability property of splitting codes from the addressability properties of the subcodes it is made of.

## III. Results

We show a series of impossibility results for addressable gates under the given restrictions.

We start by analyzing single-qubit Clifford circuits. Any such circuit can be re-written up to phase as a gate from $\{I, H, HP, PH, PHP\}$, where $P$ is the phase gate adding a phase of $i$, followed by some Paulis. By definition, a Clifford circuit will only preserve the code space of a CSS code if it is in the normalizer of the code's stabilizer group. Since Pauli gates commute (up to phase) with the stabilizer group, we focus only on the gates in $\{I, H, HP, PH, PHP\}$ and assume the phases can be corrected.

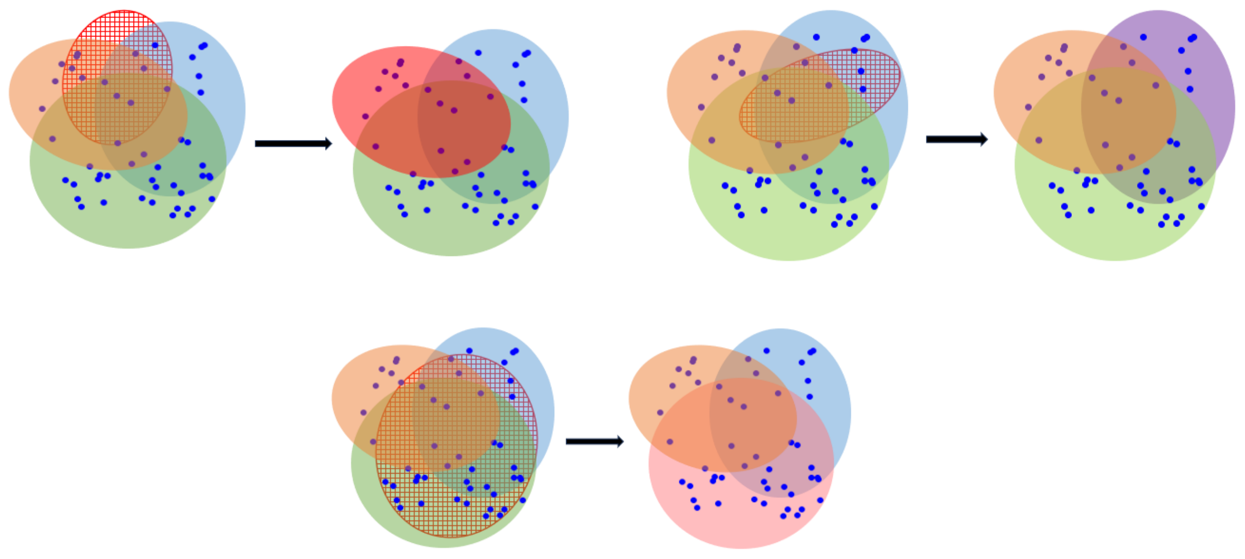


Fig. 3. $U$ is addressable on the code

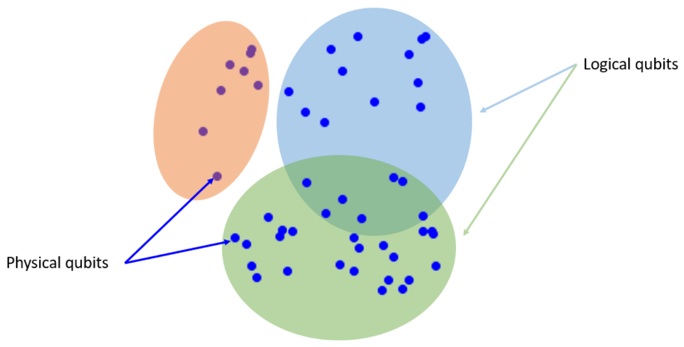


Fig. 4. Visualization of a splitting code

We first show that if such a circuit applies any physical $H$, $PH$, or $HP$ gates to some qubits in the code, it must apply such gates to *all* physical qubits in the code. This is quite restrictive, as we also show that these gates necessarily modify the logical state in *all* the logical qubits. In short, no addressable gate can be constructed from these physical gates.

Using this, we go on to show that if we want the logical action of an $H$, $HP$, $PH$, or CNOT gate, addressed to any strict subset of qubits, it cannot be done with single-qubit Clifford gates. To show this, we note from the above that we could not use a circuit of physical $H$, $HP$, or $PH$ gates to enact these logical operators, and then we show that $P$ and $PHP$ do not have the correct structure to enact $H$, $HP$, $PH$, or CNOT. Using this non-partial adressability result on non-splitting CSS codes, we derive a weaker result : non addressability on general CSS codes with rates greater than $1/7$.

**Theorem 1.** *Let $C$ a non-splitting CSS code. Then $PH, HP, H$ are not single-qubit Clifford partially addressable on $C$.*

**Corollary 1.** *Let $C$ be an $[\![n,k,d]\!]$ CSS code, if its rate is greater than* $\min(\frac{1}{2d-1}, \frac{1}{7})$ *then $H, HP$ and $PH$ are not single-qubit Clifford addressable on $C$.*

**Remark 2.** *In particular, PK codes [1] and quantum tanner codes [2] do not have single qubit Clifford addressable $H, HP, PH$.*

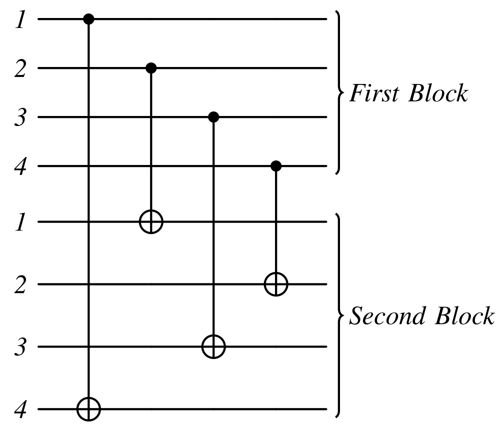
Our second set of results concerns circuits built from physical swaps. It is a restrictive condition to permute the physical qubits in a code but preserve the codespace. We give a counting argument that upper bounds the possible number of such permutations. We then show that this upper bound is less than the number of possible permutations of logical qubits *if* the rate of the code is asymptotically greater than $\frac{1}{3}$. Since SWAP gates generate all permutations, this means that no CSS code with rates this high can have addressable logical SWAP gates formed out of physical SWAP gates. This result stays valid even when considering splitting CSS codes.

**Theorem 2.** *SWAP is not permutation-addressable on families of CSS codes having an asymptotical rate greater than $\frac{1}{3}$.*

As a corollary, for these codes one cannot make addressable logical CNOT gates out of physical SWAP gates, since CNOT gates can generate SWAP gates.

Finally, we consider circuits built out of CNOTs between two codes. That is, we have two codes running in parallel, each using the same stabilizer group on their respective physical qubits, and we apply CNOTs with controls from one code and targets on the other. We further require that all physical qubits are the target or control of some CNOT.

**Example 5.** *Consider the following unitary :*



*where control and targets are from two different blocks of the same type of code made of* 4 *physical qubits.*

With such circuits, we cannot construct a set of addressable logical CNOTs. We show this by showing that the commutation relations of CNOTs with Pauli strings imply that such circuits form an automorphism of the code, and using our previous permutation upper bound, there are simply not enough such automorphisms to accommodate all logical CNOTs.

As an additional result, we give an algorithm running in $\mathcal{O}(n^2)$ that detects when a code splits into parallel subcodes and returns the supports of the splits as well as the subcodes. This could be helpful for future work into addressable codes, but also might be useful as a quick check for whether the distance of a randomly-constructed LDPC code has high distance (since a splitting code has a smaller distance).

## Conclusion

Ideally, we would answer the question of addressability, by either giving a method to perform addressable gates on high-performance codes, or definitively proving that this is impossible. Instead, we have only some impossibility results. However, our results suggest what routes will be necessary if addressable gates are possible, highlight new proof techniques for considering these problems, and emphasize some of the restrictions me might need in considering the addressability problem.

For example, [6] and [5] seem to contradict our results by providing an addressable $H$ gate. However, as these papers point out themselves, their techniques do not necessarily preserve distance. [6] involves enacting a linear transformation on the stabilizer vectors by applying a physical CNOT from each physical qubit in the code to *an unprotected* auxiliary qubit. This means any phase error on this qubit will propagate up into the code. Hence, distance-preserving techniques remain an important consideration.

One easy fix might be to encode the auxiliary qubit in a different code (say, a surface code). However, it is not clear that there is such a targeted CNOT *between* such different codes. This is an open question that we hope can be resolved with techniques similar to those we employ in this work.

Another method to escape our restrictions would be to allow the physical circuit to modify the code. For example, maybe there is a family of codes that can all be reached from each other by depth-1 CNOT circuits. This would be a special structure to have, so we assumed it did not exist, but finding such a structure would open up many possibilities for addressable gates.

Overall, we hope our results motivate more consideration of addressability and that our techniques can be taken further, either for constructive results or impossibility theorems. This work also shows that we should not take it for granted that, because one quantum error-correcting code is able to encode logical qubits more efficiently than another one, it will be overall more efficient for computation.

## References

[1] P. Panteleev and G. Kalachev, "Asymptotically good quantum and locally testable classical ldpc codes," in *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, ser. STOC 2022. New York, NY, USA: Association for Computing Machinery, 2022, p. 375–388. [Online]. Available: https://doi.org/10.1145/3519935.3520017

[2] A. Leverrier and G. Zemor, "Quantum tanner codes," in *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*. Los Alamitos, CA, USA: IEEE Computer Society, nov 2022, pp. 872–883. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/FOCS54457.2022.00117

[3] A. R. Calderbank and P. W. Shor, "Good quantum error-correcting codes exist," *Phys. Rev. A*, vol. 54, pp. 1098–1105, Aug 1996. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.54.1098

[4] G. Zhu, S. Sikander, E. Portnoy, A. W. Cross, and B. J. Brown, "Non-clifford and parallelizable fault-tolerant logical gates on constant and almost-constant rate homological quantum ldpc codes via higher symmetries," *ArXiv*, vol. abs/2310.16982, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:264490902

[5] A. O. Quintavalle, P. Webster, and M. Vasmer, "Partitioning qubits in hypergraph product codes to implement logical gates," *Quantum*, vol. 7, p. 1153, Oct. 2023. [Online]. Available: https://doi.org/10.22331/q-2023-10-24-1153

[6] A. Patra and A. Barg, "Targeted clifford logical gates for hypergraph product codes," 2024. [Online]. Available: https://arxiv.org/abs/2411.17050

[7] T.-C. Lin, "Transversal non-clifford gates for quantum ldpc codes on sheaves," 2024. [Online]. Available: https://arxiv.org/abs/2410.14631