EPJ Quantum Technology
a SpringerOpen Journal

RESEARCH Open Access

Soft syndrome iterative decoding of quantum LDPC codes and hardware architectures

Nithin Raveendran[1*], Javier Valls[2], Asit Kumar Pradhan[1], Narayanan Rengaswamy[1], Francisco Garcia-Herrero[3] and Bane Vasić[1]

*Correspondence: nithin@arizona.edu
[1]Department of Electrical and Computer Engineering, The University of Arizona, Tucson, USA
Full list of author information is available at the end of the article

Abstract
In practical quantum error correction implementations, the measurement of syndrome information is an unreliable step—typically modeled as a binary measurement outcome flipped with some probability. However, the measured syndrome is in fact a discretized value of the continuous voltage or current values obtained in the physical implementation of the syndrome extraction. In this paper, we use this "soft" or analog information to benefit iterative decoders for decoding quantum low-density parity-check (QLDPC) codes. Syndrome-based iterative belief propagation decoders are modified to utilize the soft syndrome to correct both data and syndrome errors simultaneously. We demonstrate the advantages of the proposed scheme not only in terms of comparison of thresholds and logical error rates for quasi-cyclic lifted-product QLDPC code families but also with faster convergence of iterative decoders. Additionally, we derive hardware (FPGA) architectures of these soft syndrome decoders and obtain similar performance in terms of error correction to the ideal models even with reduced precision in the soft information. The total latency of the hardware architectures is about 600 ns (for the QLDPC codes considered) in a 20 nm CMOS process FPGA device, and the area overhead is almost constant—less than 50% compared to min-sum decoders with noisy syndromes.

Keywords: Quantum error correction; Syndrome-based decoding; FPGA devices

1 Introduction

Quantum error correction (QEC) is a necessary step in realizing scalable and reliable quantum computing. For fault-tolerant QEC, quantum low-density parity-check (QLDPC) codes are advantageous over surface codes, the current leading candidate [1–5], in terms of the scaling of qubit overhead and minimum distance. Recent breakthroughs related to various product constructions of QLDPC codes provided constructions of "good" QLDPC code families, i.e., codes with finite asymptotic rate and relative minimum distance [4, 6–9]. Hardware-efficient fault-tolerant quantum computation demonstrations using high-rate QLDPC codes are exciting and also compatible with recently demonstrated experimental capabilities [10–12]. Similarly, there has been significant progress in improving the iterative decoding performance of finite-length QLDPC codes using post-processing and heuristic techniques [13–17]. However, the QLDPC decoding problem still has unanswered questions and, in particular, faster decoders for QLDPC codes are needed to meet the stringent timing constraints in hardware. The problem is even more challenging than classical LDPC code decoding since we have to overcome syndrome measurement errors in addition to the errors on physical qubits. To address these errors without repeating the measurements, data-syndrome codes and "single-shot" error correction codes have been proposed [18–21]. The idea of *single-shot* decoding is to devise a decoder for a given code such that even with one round of noisy measurements it can correct errors very well. However, noisy syndrome measurement has not been analyzed under iterative decoding, which is of great interest while decoding these good QLDPC codes.

Quantum errors are primarily detected and corrected using the measurement of syndrome information which itself is an unreliable step in practical error correction implementations. Typically, such faulty or noisy syndrome measurements are modeled as a binary measurement outcome flipped with some probability. However, the measured syndrome is in fact a discretized value of the continuous voltage or current values obtained in the physical implementation of the syndrome extraction.

The first, and primary, challenge is that when a decoder uses a noisy syndrome as input, it significantly and non-trivially alters the iterative decoding dynamics. Secondly, in the absence of a referent "true" syndrome, it is difficult to formulate the halting condition, and an iterative decoder can, in principle, run until reaching the maximum number of iterations without finding the error pattern. A naïve binary quantization of the syndrome would lead to an estimated error that is possibly matched to a wrong syndrome.

In the literature, noisy measurements are typically modeled by a simple phenomenological error model in which binary measurement outcomes are flipped with some probability. However, the measurement is typically more informative since it corresponds to continuous voltages, currents, or discrete photon counts based on the specific physical implementation. Instead of quantizing this analog information to a binary outcome and losing crucial information, one can leverage the soft syndrome in the iterative decoder and modify the update rules accordingly. In the work by Pattinson *et al.* [22], syndrome measurement outcomes beyond simple binary values were considered. The decoders used for surface codes, such as minimum weight perfect matching and union-find, were modified accordingly to obtain higher decoding thresholds.

In this work, we model a realistic syndrome measurement by a perfect measurement with an ideal bipolar (± 1) outcome followed by a noisy channel that leads to a continuous soft outcome. We restrict our attention to the simple setting of a symmetric Gaussian noise on each of the ideal syndrome measurements. In the context of iterative decoding, we *modify* the min-sum algorithm (MSA) based decoder (a hardware-friendly variant of the belief propagation (BP) decoder) to use these soft syndrome values as additional information to drive the message passing algorithm towards convergence [42]. This iterative decoder for QLDPC codes that employs *soft syndrome information* is novel and is one of the main contributions of this paper. It is also unprecedented in the setting of iterative decoding of classical LDPC codes, although faulty components in decoders have been studied [23–25]. Furthermore, even if the code does not have the single-shot prop-

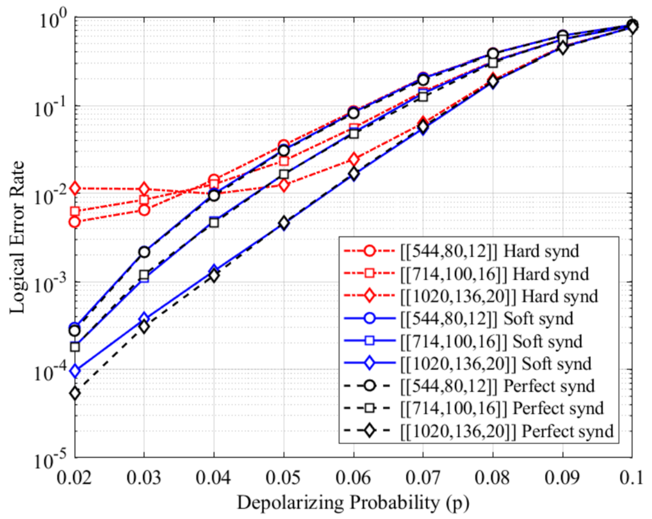


Figure 1 Two-threshold phenomenon observed for 'Hard Syndrome' (which uses standard MSA), but the 'Soft Syndrome' setting (which uses modified MSA) is devoid of this issue

erty, our strategy allows one to perform effective decoding *without repeating syndrome measurements*, which is attractive practically.

We consider three scenarios for decoding:

1. *Perfect Syndrome*: In this case, there is no noise added to the syndrome and we use the standard MSA decoder.
2. *Hard Syndrome*: Here, there is Gaussian noise added to the syndrome, but this continuous value is thresholded to input a hard syndrome to the standard MSA decoder.
3. *Soft Syndrome*: Now, as in the hard syndrome case, the syndrome is noisy and it is thresholded to produce a sign (± 1), but the magnitude is converted into a log-likelihood ratio (LLR) to also provide soft information. The decoder in this case is our new modified MSA, where the check node update rules are modified and, specifically, the LLR corresponding to the syndrome is utilized and also updated during the check node update.

As one might expect, the threshold for the 'Hard Syndrome' case is worse than the 'Perfect Syndrome', since the decoder is unchanged. But, more interestingly, for the 'Hard Syndrome', when we fix the variance of the Gaussian noise affecting the syndrome and sweep the qubit depolarizing rate, below what we identify as the usual decoding threshold, there arises a "second (inverted) threshold"; below this (lower) threshold, the logical error rate creeps up again because the syndrome error dominates the qubit error much more strongly (see Fig. 1). This suggests that, under realistic settings, it is insufficient to only optimize the threshold, and one needs to avoid such problematic behavior below the threshold.

Next, we propose the modified MSA decoder with justification for our new rules for updating nodes as well as the syndrome LLR. We show that the resulting decoder does not suffer from this spurious "second threshold" phenomenon. Besides, the decoding threshold is now almost as good as that for the 'Perfect Syndrome' setting. Hence, our work paves the way for efficient and effective decoding of good QLDPC codes under a physically motivated phenomenological noise model, without the use of repetitive syndrome measurement rounds.

Finally, we provide an efficient architecture and the results derived from FPGA implementations for different QLDPC codes (from 442 physical qubits to 1428) showing how latency can be kept almost constant and very close to standard MSA decoders, around 600 ns for 30 iterations, with similar power consumption.

The paper is organized as follows. In Sect. 2 we introduce the necessary notations and explain the syndrome-based MSA and its variants. In Sect. 3 we motivate the soft syndrome decoders and explain our modification to correct the noisy syndromes. Section 4 presents simulation results showing the benefits of our modified MSA. Section 5 introduces the derived hardware architecture and the results and comparisons of FPGA implementations for different QLDPC codes. Finally, Sect. 6 concludes this paper with a discussion on future work.

2 Preliminaries

2.1 Pauli operators and CSS codes

The single-qubit Pauli operators are $\mathrm{I}, \mathrm{X}, \mathrm{Z}, \mathrm{Y} = \imath \mathrm{XZ}$ and the n-qubit Pauli group $\mathcal{P}_n$ consists of tensor products of single-qubit Paulis up to a quaternary global phase $1, \imath, -1$, or $-\imath$, where $\imath = \sqrt{-1}$. There is a useful binary representation $[a, b] \in \{0, 1\}^{2n}$ of Pauli operators, e.g., $\mathrm{X} \otimes \mathrm{Z} \otimes \mathrm{Y} \mapsto [a, b] = [101, 011]$, so that entries of a (resp. b) indicate which qubits are acted upon by X (resp. Z), and $a_i = b_i = 1$ (resp. 0) indicates a Y (resp. I). An $[\![n, k, d]\!]$ CSS code [26] encodes k logical qubits into n physical qubits, and d is the minimum distance of the code that indicates the minimum number of qubits on which the error channel must act to disturb the logical information. The CSS code is defined by n-qubit Pauli operators, called *stabilizers*, some of which are purely X-type, $[a, 0]$, and the others are purely Z-type, $[0, b]$. Hence, the stabilizers can be represented using two binary parity-check matrices H_{X} (whose rows map to $[a, 0]$) and H_{Z} (whose rows map to $[0, b]$) that must satisfy $H_{\mathrm{X}} H_{\mathrm{Z}}^{\mathrm{T}} = 0$ so that these Pauli operators commute.

2.2 Noise model

The single-qubit depolarizing channel, also known as the memoryless Pauli channel, is a widely studied error model characterized by the depolarizing probability p: the channel randomly introduces a Pauli error according to the probabilities $P_{\mathrm{X}} = P_{\mathrm{Y}} = P_{\mathrm{Z}} = p/3$ and $P_{\mathrm{I}} = 1 - p$. For the physical data qubits, we assume an i.i.d. depolarizing error model, i.e., each physical qubit is affected by an independent depolarizing error. Let $\mathbf{e} = [\mathbf{e}_{\mathrm{X}}, \mathbf{e}_{\mathrm{Z}}]$ be the binary representation of a Pauli error acting on the n qubits. The corresponding syndrome is computed as $[\mathbf{s}_{\mathrm{X}},\ \mathbf{s}_{\mathrm{Z}}] = [H_{\mathrm{Z}} \mathbf{e}_{\mathrm{X}}^{\mathrm{T}}, H_{\mathrm{X}} \mathbf{e}_{\mathrm{Z}}^{\mathrm{T}}]^{\mathrm{T}} \pmod 2$. Since we consider decoding of H_{X} and H_{Z} separately, we avoid using subscripts X and Z in syndrome $\mathbf{s}$.

2.3 Syndrome noise model

As mentioned in Sect. 1, we model a realistic syndrome measurement by a perfect measurement with an ideal bipolar (± 1) outcome followed by i.i.d. symmetric Gaussian noise on each component of the syndrome. After introducing symmetric Gaussian noise $n_i \sim \mathcal{N}(0, \sigma^2)$ to the ideal syndrome (component) $s_i \in \{\pm 1\}$, we have a noisy soft syndrome $\tilde{r}_i = s_i + n_i$ which when conditioned on s_i is distributed normally: $[\tilde{r}_i \mid s_i = 1] \sim N(+1, \sigma^2)$ and $[\tilde{r}_i \mid s_i = -1] \sim N(-1, \sigma^2)$. Other relevant asymmetric syndrome measurement models are discussed in [22, Sect. 1.4].

Based on the normally distributed soft syndromes, we can compute the syndrome log-likelihood ratios (LLRs) as

$$\gamma_i := \log \frac{\Pr(\tilde{r}_i|s_i = +1)}{\Pr(\tilde{r}_i|s_i = -1)} = \frac{2\tilde{r}_i}{\sigma^2}. \tag{1}$$

Since the noise is zero-mean, the sign of the soft syndrome, $\operatorname{sgn}(\gamma_i)$, indicates if the parity-check i is satisfied ($\operatorname{sgn}(\gamma_i) = +1$) or unsatisfied ($\operatorname{sgn}(\gamma_i) = -1$). This is exactly the same as the measured syndrome s_i used in a conventional min-sum decoder. However, its magnitude, $|\gamma_i|$, provides valuable soft information about the reliability of the measured syndrome.

2.4 QLDPC codes and syndrome min-sum algorithm (MSA)

A CSS-QLDPC code has sparse stabilizer generators and is represented by a pair of bipartite (Tanner) graphs whose biadjacency matrices are the stabilizer matrices H_X and H_Z. The Tanner graphs have n variable nodes, denoted by $j \in \{1, 2, \dots, n\}$, represented by circles in Fig. 2, and m check nodes, denoted by $i \in \{1, 2, \dots, m\}$, represented by squares in Fig. 2. If the (i,j)th entry of the binary parity-check matrix is not zero, then there is an edge between variable node i and check node j. The sets of neighbors of variable node j and check node i are denoted by $\mathcal{N}(j)$ and $\mathcal{N}(i)$, respectively.

Like classical LDPC codes, QLDPC codes also employ decoders based on iterative BP or message-passing algorithms. The fundamental difference between the decoders of QLDPC codes and classical LDPC codes is that the decoders for the former take as input the syndrome obtained after the stabilizer measurements, whereas decoders for the latter take as input a noisy version of the codeword. The QLDPC decoder's goal is to output an error pattern whose syndrome matches the measured syndrome. Let us first consider the case when these stabilizer measurements are perfect, meaning there is no noise in the syndrome measurement process.

Suppose that the coded qubits are corrupted by an X-type or Z-type error, corresponding to the binary error vector $\mathbf{e} = [e_1, e_2, \dots, e_n]$ whose entries are realizations of independent Bernoulli random variables with parameter p. Given binary syndrome $\mathbf{s}$ vector of length m, the decoder performs a finite number of message-passing iterations over the Tanner graph to compute *a posteriori probabilities* $\Pr(e_j|\mathbf{s})$, for $j \in \{1, \dots, n\}$, corresponding to error bit e_j conditioned on the measured syndrome $\mathbf{s}$.

Next, we briefly describe the min-sum algorithm (MSA) based decoder. For this purpose, we first introduce the required notations. Denote by $\mu_{i,j}$ the message sent from check node i to variable node j. Similarly, the message sent from variable node j to check node i is denoted by $\nu_{i,j}$. The measured syndrome value is $s_i \in \{\pm 1\}$, $i \in \{1, 2, \dots, m\}$. Since the error e_j, $j \in \{1, 2, \dots, n\}$, at variable node j is a realization of Bernoulli random variable with pa-

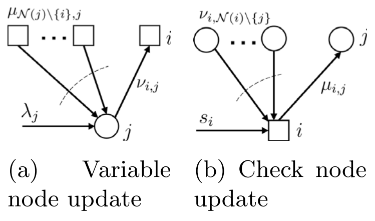


Figure 2 Node updates for MSA based on Eqs. (2) and (3)

rameter p, the log-likelihood ratio (LLR), denoted by λ_j, corresponding to the jth variable node is given by $\lambda_j = \ln(\frac{1-p}{p})$.

In the first decoding iteration, all the outgoing messages from variable node j, for all j, are initialized to λ_j. In subsequent iterations, for $i \in \mathcal{N}(j)$, message $\nu_{i,j}$ is computed] using the update rule[1]

$$\nu_{i,j} = \lambda_j + \sum_{i' \in \mathcal{N}(j) \setminus \{i\}} \mu_{i',j}. \tag{2}$$

For $j \in \mathcal{N}(i)$, the message $\mu_{i,j}$ is computed using the rule

$$\mu_{i,j} = \left(s_i \cdot \prod_{j' \in \mathcal{N}(i) \setminus \{j\}} \operatorname{sgn}(\nu_{i,j'}) \right) \cdot \left(\min_{j' \in \mathcal{N}(i) \setminus \{j\}} |\nu_{i,j'}| \right). \tag{3}$$

The sign function is defined as $\operatorname{sgn}(A) = -1$ if $A < 0$, and $+1$ otherwise. The first and second parenthesis in Eq. (3) give the sign and the magnitude of the computed check-to-variable message $\mu_{i,j}$, respectively. Iteration will be indicated as a superscript of $\mu_{i,j}$ and $\nu_{i,j}$ when required. Min-sum update rules are less complex than that of BP decoders (which are never used in practice), but as the min-sum approximation overshoots the BP messages, the MSA is typically modified using a normalized MSA [27], wherein check node messages are multiplied (in Eq. (3)) by a scalar β as a correction factor, where $0 < \beta < 1$.

A min-sum decoder employs the update functions in Eq. (2) and Eq. (3) in each iteration $\ell \leq \ell_{\max}$ to determine the error at each variable node. The error estimate $\hat{x}_j$ at the variable node j is computed using

$$\hat{x}_j = \hat{\Phi}(\lambda_j, \mu_{i,j}) = \operatorname{sgn}\left(\lambda_j + \sum_{i \in \mathcal{N}(j)} \mu_{i,j} \right), \tag{4}$$

for all $j \in \{1, 2, \dots, n\}$, i.e., it is based on the sign of the *decision update function* output, $\hat{\Phi}(\lambda_j, \mu_{i,j})$, that uses the LLR value and *all* incoming check node messages to the variable node j, where $i \in \mathcal{N}(j)$.

Halting Criterion: The output of the decoder at the ℓth iteration, denoted by $\hat{\mathbf{x}}^{(\ell)} = (\hat{x}_1^{(\ell)}, \hat{x}_2^{(\ell)}, \dots, \hat{x}_n^{(\ell)})$, is used to check whether all parity-check equations are matched, i.e., whether the syndrome at ℓth iteration $\hat{\mathbf{s}}^{(\ell)} := \hat{\mathbf{x}}^{(\ell)} \cdot \mathbf{H}^{\mathrm{T}}$ is equal to the measured syndrome $\mathbf{s}$, in which case iterative decoding terminates and outputs $\hat{\mathbf{x}}^{(\ell)}$ as the error vector. Otherwise, the iterative decoding steps continue until a predefined maximum number of iterations, denoted by $\ell_{\max}$, is reached. The decoding is deemed successful if the true error pattern is found, i.e., $\hat{\mathbf{x}}^{(\ell)} = \mathbf{e}$. Otherwise, the decoding is said to have failed. Miss-correction as referred to in the literature of classical coding theory occurs when the post-correction step with the estimated error pattern results in a logical error. This is also classified as an error correction failure.

Note that the MSA decoder is exactly the "Perfect Syndrome" decoder, described in Sect. 1, when the syndrome measurement process is noiseless. When we have a noisy observation of the syndrome, recall that an estimate of the syndrome is obtained by passing the observations to a thresholding function. This estimated syndrome is fed to the MSA decoder to get the 'Hard Syndrome' decoder defined earlier.

[1]Note that the messages can be scaled by an α factor in order to improve the error correction performance of the decoder.

3 Soft syndrome decoder

As discussed above, under noisy syndrome, the MSA decoder does not use the soft information available in the measured syndrome and treats the thresholded syndrome as the correct syndrome. Due to this reason, the decoder typically obtains error pattern estimates only close to the true error pattern, but it does not converge to a correct pattern. Also, note that the halting condition is never met with the standard MSA decoder when there are noisy syndromes as inputs, so the decoder runs until the maximum number of iterations and declares a decoding failure. To sum up, using a noisy syndrome has two disadvantages with the standard MSA decoder: i) drives the decoder to produce error estimates matching the measured (noisy) syndrome, even when it is unreliable; and ii) it makes the iterative decoder's halting condition ambiguous, even after the decoder converges to the true error estimate. As it will be described next both problems can be solved separately, which, for some cases, can be useful, i.e. hardware implementations are constrained by latency for the maximum number of iterations, so including extra resources for a halting condition may not introduce any advantage when it comes to area, timing and power consumption of the hardware implementation of the decoder.

In this section, we propose a modified MSA decoder [42] that uses the soft information available in the measured syndrome to address these issues: i) to converge to an error pattern only based on reliable information and ii) to update the measured syndromes to implement a halting condition. Note that the first issue is necessary to make the decoder converge to a valid error pattern, but the second one is necessary to reduce the average number of iterations by means of applying the halting criterion. As mentioned in Sect. 1, we refer to the proposed decoder as 'Soft Syndrome' (MSA).

3.1 Node update with soft information from the observed syndrome

To use the soft information from the observed syndrome, we modify the Tanner graph by adding another type of node, called "virtual variable nodes", in addition to the existing variable and check nodes. A virtual variable node is added corresponding to each of the m check nodes. There is an edge between a virtual variable node and the corresponding check node. The connections between the variable and check nodes are not altered. Hence, the standard MSA decoder still holds.

The initialization and update rules at the variable nodes for the 'Soft Syndrome' decoder are the same as that of the MSA decoder described in Sect. 2. We now describe the update rules at the check nodes and virtual variable nodes for the 'Soft Syndrome' decoder. In addition to the measured syndrome $\mathbf{s}$, a binary vector of length m, the 'Soft Syndrome' decoder receives the respective reliability (i.e., syndrome LLR) values γ_i as defined in Eq. (1). The goal of this 'Soft Syndrome' iterative decoder in the noisy syndrome setting is to find an error pattern that matches the perfect syndrome $\tilde{\mathbf{s}}$ using the syndrome reliability γ and the qubit LLR λ.

The magnitude of the soft syndrome LLR value determines the reliability of the check information. To deploy the 'Soft Syndrome' decoder, we determine a *cutoff* above which the soft syndrome value is deemed reliable and below which it is not. If the syndrome is reliable, i.e., if the $|\gamma_i|$ corresponding to check node i is greater than the predefined cutoff, then we use the conventional check node update rule given in Eq. (3). Choosing this cutoff value denoted by Γ is important and it can be optimized to improve the decoding performance. There are two main changes to the minimum rule used at the check nodes.

With the soft syndrome γ_i available from the virtual variable node, first, we compute the check to variable message as the minimum of magnitude of the extrinsic variable node messages as well as the soft syndrome magnitude $|\gamma_i|$. For $j \in \mathcal{N}(i)$, the magnitude and sign of message $\mu_{i,j}$ are given by

$$|\mu_{i,j}| = \begin{cases} (\min_{j' \in \mathcal{N}(i) \setminus \{j\}} |\nu_{i,j'}|) & \text{for } |\gamma_i| > \Gamma, \\ (\min_{j' \in \mathcal{N}(i) \setminus \{j\}} (|\nu_{i,j'}|, |\gamma_i|)) & \text{otherwise;} \end{cases} \tag{5}$$

$$\operatorname{sgn}(\mu_{i,j}) = \left(s_i \cdot \prod_{j' \in \mathcal{N}(i) \setminus \{j\}} \operatorname{sgn}(\nu_{i,j'}) \right). \tag{6}$$

These two equations allow the decoder converge to a correct error pattern, even when the measured syndromes are unreliable.

3.2 Updating the halting condition

Second, we update the soft syndrome, both magnitude and reliability, based on the magnitude and sign of all the incoming messages to the check node from its neighboring variable nodes, thus effectively pushing the estimated syndrome towards the correct syndrome. By applying Eq. (7) and (8) we avoid an ambiguous halting condition, reducing the average number of iterations of the algorithm. In the ℓth iteration, the estimated sign and reliability of syndrome corresponding to check node i is denoted by $\tilde{s}_i^{(\ell)}$ and $\tilde{\gamma}_i^{(\ell)}$, respectively. In the first iteration, we initialize $\tilde{s}_i^{(1)} = s_i$ and $\tilde{\gamma}_i^{(1)} = |\gamma_i|$. In the subsequent iterations, the reliability $\tilde{\gamma}_i^{(\ell)}$ of the estimated syndrome corresponding to the check node i is updated using

$$\tilde{\gamma}_i^{(\ell)} = \min_{j' \in \mathcal{N}(i)} |\nu_{i,j'}| \tag{7}$$

if $\min_{j' \in \mathcal{N}(i)} |\nu_{i,j'}| > |\tilde{\gamma}_i^{(\ell-1)}|$ and $\tilde{s}_i^{(\ell-1)} = \prod_{j' \in \mathcal{N}(i)} \operatorname{sgn}(\mu_{i,j'})$, otherwise $\tilde{\gamma}_i^{(\ell)} = \tilde{\gamma}_i^{(\ell-1)}$.

The sign of the estimated syndrome corresponding to the check node i is updated using

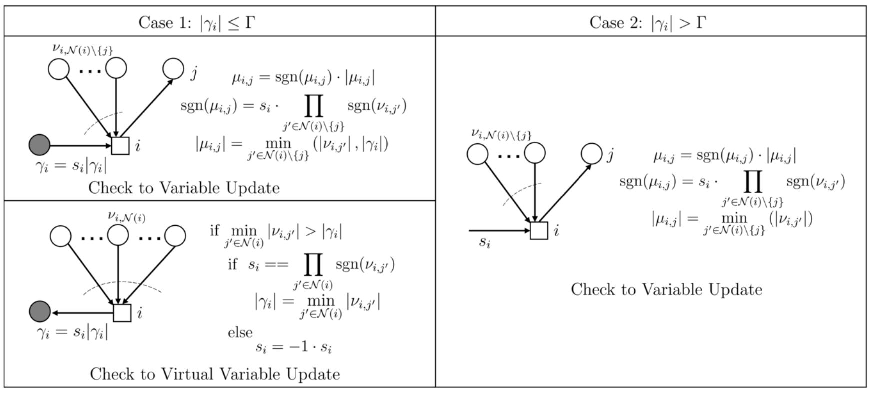
$$\tilde{s}_i^{(\ell)} \mapsto (-1) \cdot \tilde{s}_i^{(\ell-1)} \tag{8}$$

if $\min_{j' \in \mathcal{N}(i)} |\nu_{i,j'}| > |\tilde{\gamma}_i^{(\ell-1)}|$ and $\tilde{s}_i^{(\ell-1)} \neq \prod_{j' \in \mathcal{N}(i)} \operatorname{sgn}(\mu_{i,j'})$, otherwise $\tilde{s}_i^{(\ell)} = \tilde{s}_i^{(\ell-1)}$.

The halting criterion for the 'Soft Syndrome' decoder is whether all parity-check equations are matched to these updated signs $\tilde{s}_i^{(\ell)}$ $\forall i \in \{1, 2, \dots, m\}$, i.e., whether the syndrome at ℓth iteration, $\hat{\mathbf{x}}^{(\ell)} \cdot \mathbf{H}^{\mathrm{T}}$ is equal to the $\tilde{\mathbf{s}}$, in which case decoding terminates and outputs $\hat{\mathbf{x}}^{(\ell)}$ as the error vector. The modified decoding rules are illustrated in Fig. 3.

Though in the simulation results in Sect. 4, we use the 'Soft Syndrome' discussed above, we also identify another variant where one does not update the reliability or the sign of the estimated syndrome given in Eq. (7) and (8), this variant is described in Sect. 5 showing similar logical error rate and less complexity in terms of hardware resources.

Note that we are not using an explicit syndrome code for protecting the syndrome bits as in [19, 28], instead relying on the message passing algorithm to infer what the syndromes should be. Using this modified MSA decoder, we hope to avoid the overhead of repeated measurements as we can identify the measurement errors in the instances where this decoder converges. Since the approach can be used for any message-passing variant



Case 1: $|\gamma_i| \leq \Gamma$
Case 2: $|\gamma_i| > \Gamma$
$\nu_{i,\mathcal{N}(i)\backslash\{j\}}$
j
i
$\mu_{i,j} = \mathrm{sgn}(\mu_{i,j}) \cdot |\mu_{i,j}|$
$\mathrm{sgn}(\mu_{i,j}) = s_i \cdot \prod_{j' \in \mathcal{N}(i)\backslash\{j\}} \mathrm{sgn}(\nu_{i,j'})$
$|\mu_{i,j}| = \min_{j' \in \mathcal{N}(i)\backslash\{j\}} (|\nu_{i,j'}|, |\gamma_i|)$
$\gamma_i = s_i|\gamma_i|$
Check to Variable Update
$\nu_{i,\mathcal{N}(i)}$
if $\min_{j' \in \mathcal{N}(i)} |\nu_{i,j'}| > |\gamma_i|$
if $s_i == \prod_{j' \in \mathcal{N}(i)} \mathrm{sgn}(\nu_{i,j'})$
$|\gamma_i| = \min_{j' \in \mathcal{N}(i)} |\nu_{i,j'}|$
else
$s_i = -1 \cdot s_i$
Check to Virtual Variable Update
s_i
$|\mu_{i,j}| = \min_{j' \in \mathcal{N}(i)\backslash\{j\}} (|\nu_{i,j'}|)$

Figure 3 Modification of the check node update for the 'Soft Syndrome' decoder based on a predefined cutoff, Γ. In addition to the 'minimum' update, check to virtual variable updates (shaded variable node) are updated conditionally

of iterative decoder, we can also deploy our solution jointly even with post-processing approaches such as an ordered-statistics decoder [13]. Beyond the scope of this paper are alternative decoders like small-set-flip decoders for quantum Tanner codes [29] which will need adaptation of our approach to appropriately weigh the inputs and perform decoding over quantum Tanner codes.

4 Monte-Carlo simulation results

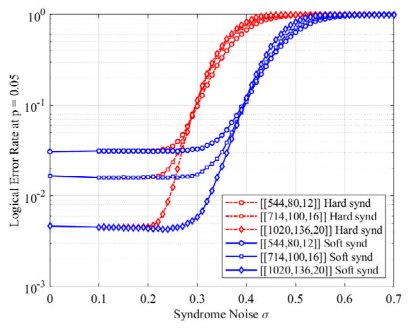
4.1 QLDPC codes and simulation setup

We use *lifted product* (LP) QLDPC codes proposed by Panteleev and Kalachev [30] which provide flexibility in constructing finite length QLDPC codes from good classical (and quantum) *quasi-cyclic (QC)* [31] LDPC codes. We choose QC-LDPC codes as constituent classical LDPC codes to construct LP code families for demonstrating our simulation results. For example, from the $[155, 64, 20]$ Tanner code [32] with quasi-cyclic base matrix of size 3×5 and circulant size $L = 31$, we obtain the $[\![1054, 140, 20]\!]$ *LP Tanner code* [33]. To show the decoding threshold plots, we also chose the LP code family described in [33, Table II]. LP-QLDPC codes with increasing code length and minimum distance ($d = 12, 16, 20, 24$) are chosen to demonstrate the thresholds. The syndrome-based MSA is chosen with a normalization factor of β set to 0.75 empirically, and has a preset maximum of $\ell_{\max} = 100$ iterations. For the Monte-Carlo simulations, specifically for the threshold plots, we collect at least 10,000 logical errors (sufficient to avoid statistical errors). The Cutoff Γ is set to 10 for Fig. 1 and to 5 for the rest of the simulation plots.

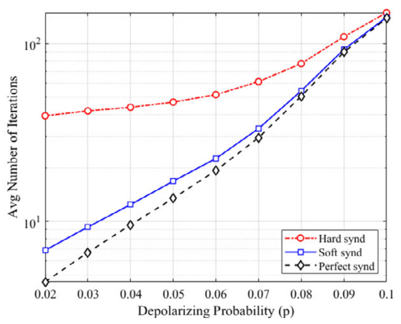
Different scenarios of simulation setups—'Perfect', 'Hard', and 'Soft' syndrome decoding as described in Sect. 1—are considered, with depolarizing noise as the underlying noise for the physical qubits. We also denote the respective MSA decoders as 'perfect synd', 'hard synd', and 'soft synd'.

4.2 Threshold plots

Threshold on Syndrome Noise: To observe the syndrome noise threshold, we plot in Fig. 4a the logical error rate for a fixed depolarizing probability of $p = 0.05$ against different values of the syndrome noise standard deviation σ. In this setting, the 'Hard Syndrome' decoder



(a) Logical Error Rate for a fixed depolarizing probability of $p = 0.05$ is plotted against the syndrome noise standard deviation σ for the QLDPC code family with increasing code length and distance. Observe that σ threshold increased from $\approx$ 0.25 for 'Hard Syndrome' to ≈ 0.4 for 'Soft Syndrome'.



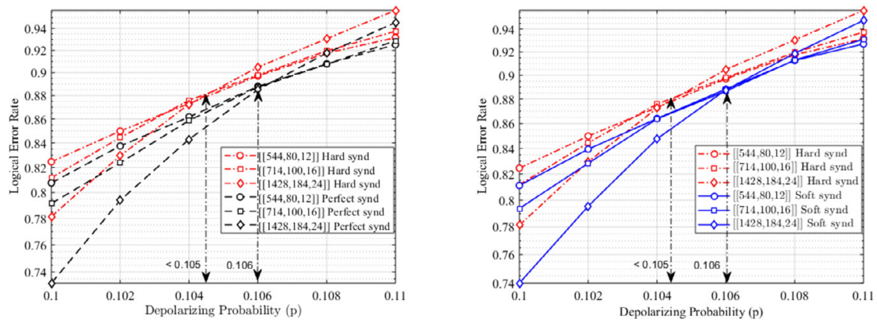
(b) For LP Tanner code with $\sigma = 0.3$, average number of iterations while considering decoding over both H_X and H_Z matrices shows that 'Soft Syndrome' decoder requires far less decoding iterations to correct both data and syndrome errors compared to the 'Hard Syndrome' decoder.

Figure 4 Benefits of the 'Soft Syndrome' decoder: (a) syndrome noise threshold, (b) average number of iterations

produces an early threshold at syndrome noise $\sigma \approx 0.25$. Above this syndrome noise level, a hard decoder cannot improve the decoding performance even by increasing the code length arbitrarily. However, with the 'Soft Syndrome' decoder we see that the threshold point is at a higher syndrome noise value of $\sigma \approx 0.4$. This indicates that the modified MSA decoder is more robust and can tolerate noisier syndromes.

This experiment also gives us the range of syndrome noise where 'Soft Syndrome' decoder performs close to the 'Perfect Syndrome' decoder at $p = 0.05$ (from Fig. 1). Interestingly, at low levels of syndrome noise ($\sigma < 0.25$), the 'Soft Syndrome' decoder is able to perform at least as good as the 'Perfect Syndrome' decoder. This is a particularly encouraging noise regime where the soft information about noisy syndromes can be dealt with the modified MSA intrinsically, without repeating measurements or requiring single-shot property in the code. For our subsequent simulations, we use the syndrome noise $\sigma = 0.3$ to observe the decoding threshold and the logical error rate suppression with respect to qubit depolarizing probability.

Threshold on Depolarizing Noise: Using the same QLDPC code family and a fixed syndrome noise standard deviation $\sigma = 0.3$, we plot the logical error rate against depolarizing probability for the decoders of interest in Fig. 5. One can see the transition from error suppression to error enhancement with increasing p which signifies the existence of the error threshold. As expected, since we use the same standard MSA decoder, we see in Fig. 5a that the threshold for the 'Hard Syndrome' case is worse than 'Perfect Syndrome'. More interestingly, when we look at lower p in Fig. 1, below what we identify as the usual decoding threshold, there exists another crossover point for the 'Hard Syndrome' case where the logical error rate rises up again because the syndrome error starts to dominate the qubit error much more strongly. However, the 'Soft Syndrome' curves in both Fig. 1 and Fig. 5b show that, besides achieving the same decoding threshold as 'Perfect Syndrome', the modified MSA is also devoid of the spurious "second threshold" phenomenon.



(a) Thresholds for the LP-QLDPC codes from LP118 code family in [33]: 'Perfect Syndrome' vs 'Hard Syndrome'

(b) Thresholds for the LP-QLDPC codes from LP118 code family in [33]: 'Soft Syndrome' vs 'Hard Syndrome'

Figure 5 The set of curves in figures (a) and (b) correspond to the use of different decoders in the perfect and noisy syndrome setting. The transition of the curves with increasing p signifies an error threshold

4.3 Faster decoder convergence

As we discussed in Sect. 1 and Sect. 3, the stopping criterion of an iterative decoder in the presence of measurement errors is ambiguous. To understand the correct decoding performance of these decoders in the presence of syndrome errors, we do not automatically declare a logical error if the decoder exhausts the maximum number of iterations and fails to find an error matching the (evolved) syndrome. Instead, we declare a logical error if the true error 'plus' the error estimate produced by the decoder has a non-trivial syndrome or results in a logical error. This gives a fair chance to the iterative decoders in the presence of syndrome errors.

Even in the presence of noisy syndrome, the 'Soft Syndrome' decoder corrects the syndrome and data errors and converges to the true error pattern. The faster convergence of 'Soft Syndrome' decoder is shown in Fig. 4b, which clearly shows that it requires an average number of decoding iterations very close to that of the 'Perfect Syndrome' decoder. Hence, under moderate syndrome noise, besides avoiding repeated syndrome measurements and single-shot property, the modified MSA also does not suffer from a high penalty in decoding latency, as we will show in the next section—an attractive feature for hardware implementation.

4.4 Simulation for other code constructions

To explore the versatility of our proposed decoder, we conducted simulations on other codes with different construction methods: *hypergraph codes* and *bicycle codes*. Two well-known hypergraph codes, B1 and C2, as well as the A2 bicycle code from [34] were simulated under the same conditions described at the beginning of this section. The results, as depicted in Figs. 6, 7 and 8, reveal a consistent impact of the proposed algorithm across all simulated codes, irrespective of their construction method. Notably, it can be observed more substantial improvements for the B1 and C2 codes, where an early error floor emerges if our proposal is not applied and noisy syndromes are present. Conversely, the A2 code, exhibits suboptimal performance in the waterfall region even without noisy syndromes (perfect syndromes), only commencing error correction at a physical error rate of 10^{-2}, so, in that case, the proposed decoder does not have room for improving the behavior, as it can be seen if we compare our solution to the perfect syndrome performance.

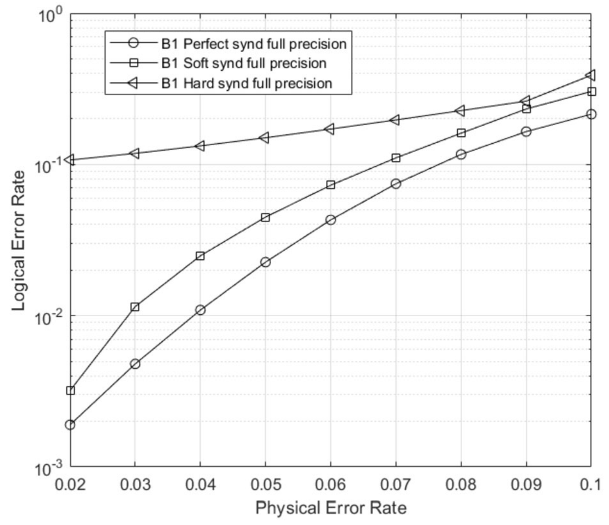


Figure 6 Performance for the B1 hypergraph code

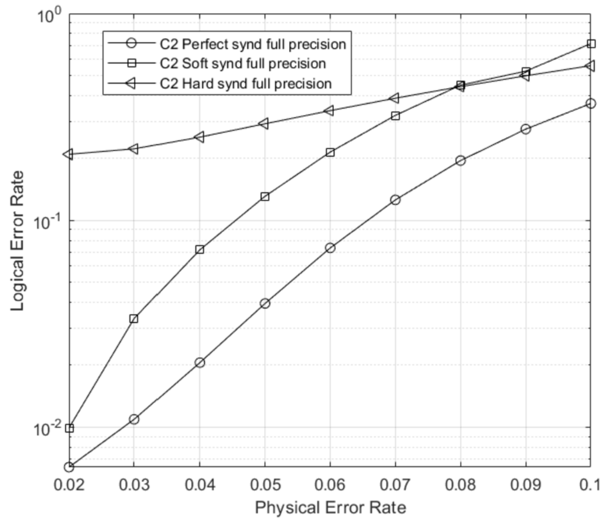


Figure 7 Performance for the C2 hypergraph code

5 Hardware implementation

We first discuss some of the salient points we adopt and consider when implementing a finite precision iterative decoder in hardware. We ensure that the precision and performance of the 'Soft Syndrome' decoder is similar to the existing standard MSA decoder ('Hard Syndrome' decoder). In the second part of this section, the focus is on the derived architecture and how hardware complexity is reduced by simplifying some parts of the algorithm. In the final subsection, we compare the differences in timing, power, and area of the 'Soft Syndrome' decoder architecture, implemented for several QLDPC codes, and the extra cost in area and speed compared to the standard MSA decoder ('Hard Syndrome' decoder) architecture for the same codes.

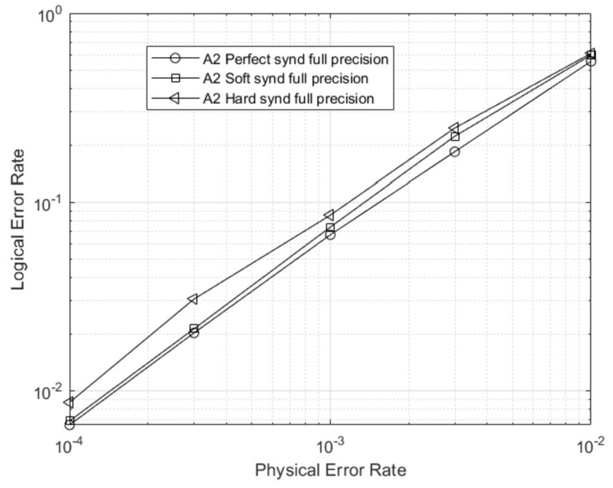


Figure 8 Performance for the A2 bicycle code

5.1 Finite precision analysis

One of the main simplifications of the standard 'Hard Syndrome' MSA decoder from a hardware point of view was initializing the channel LLR value λ_j to 1 instead of $\lambda_j = \ln(\frac{1-p}{p})$. This simplification has two important advantages: first, it was not necessary to quantize λ_j and consider it as an input for the architecture (reducing buffers and pin-out); second, the growth of the messages exchanged in the algorithm is smaller, which leads to similar error correction performance with less number of bits for the quantization. These allowed us to quantize the architectures, whose implementation results are detailed in Table 1, with only 6-bit precision messages and with negligible differences compared to the full-precision analysis. This also reduces the wiring and the dynamic power consumption and increases the speed of the final decoders that we derived.

However, the 'Soft Syndrome' decoder relies upon comparisons between the λ_j messages and the soft information from the syndromes γ_i. (It is important to remark that λ_j is used in the computation of $\nu_{i,j'}$, see Eq. (2).) For this reason, λ_j cannot be simplified, because the difference in reliability between the messages in the variable nodes and the syndrome information is important to update the check-node messages and make a decision on which update rule needs to be applied (case 1 or case 2 from Fig. 3). To measure with enough precision these two reliability values (λ_j and $\nu_{i,j'}$), some extra bits are required compared to the 'Hard Syndrome' decoder. This increases the wiring and the complexity of the operations involved (additions and comparisons) and, hence, reduces the maximum clock frequency achievable. As can be seen in Fig. 9, the finite precision versions require two extra bits compared to the 'Hard Syndrome' decoder to obtain a similar logical error rate as the full-precision decoders.

For the codes we tested: ($[\![442,68,10]\!]$, $[\![714,100,16]\!]$ and $[\![1020,136,20]\!]$), the 'Hard Syndrome' decoder is not able to improve the logic error rate beyond 10^{-2}, while the 'Soft Syndrome' decoder performs in all cases close to the 'Perfect Syndrome' scenario (syndrome without noise). Note that $[\![442,68,10]\!]$ and $[\![714,100,16]\!]$ perform well with 8 bits as can be seen in Fig. 9. It is interesting to see that error correction performance would suffer some negligible loss when the code grows, e.g., $[\![1020,136,20]\!]$ and beyond for the 'Soft Syndrome' decoder.

Table 1 Summary of implementation results for standard MSA decoder ('Hard Syndrome' decoder) architecture with 6-bit quantization (xcvu095 FPGA device, 20 nm CMOS process)

Code $[\![n,k,d]\!]$	$[\![442,68,10]\!]$	$[\![544,80,12]\!]$	$[\![714,100,16]\!]$	$[\![1020,136,20]\!]$	$[\![1428,630,24]\!]$
Clock period	8 ns	8 ns	8 ns	8 ns	10 ns
Total latency (@30it)	480 ns	480 ns	480 ns	480 ns	600 ns
Lookup Table (LUT)	69,586 (13%)	85,487 (15.9%)	16112,042 (20.8%)	159,564 (29.7%)	221,022 (41.1%)
Flip Flop (FF)	15,801 (1%)	19,427 (1.8%)	25,472 (2.4%)	36,339 (3.4%)	50,834 (4.7%)
Power	1.77 W	2.7 W	3.02 W	4.43 W	4.64 W

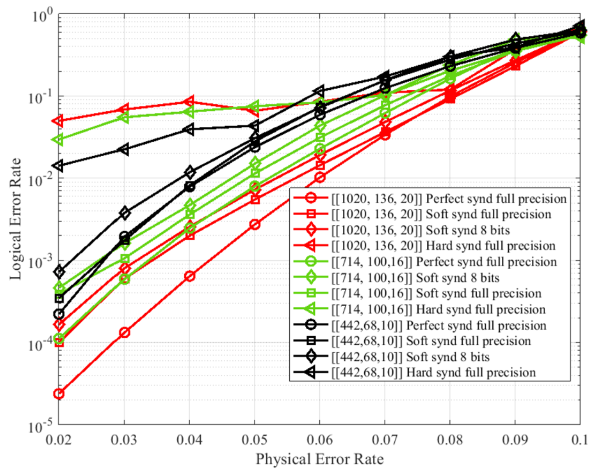


Figure 9 Error correction performance of standard MSA decoder ('Hard Syndrome' decoder) and 'Soft Syndrome' decoder (quantize and full-precision versions) with noisy syndromes and 'Perfect-syndrome' decoding with 30 iterations. The 'Soft Syndrome' decoder does not implement Eq. (7) and (8) for the syndrome update and the halting condition

To sum up, even if we increase the number of bits to infinity with other algorithms, like the 'Hard Syndrome' decoder, we would not be able to perform any error correction in the presence of noise in the measurement of the syndromes. These algorithms only work well for scenarios that are similar to 'Perfect Syndrome' decoding.[2] So, we can conclude that, in terms of quantization, a few extra bits is the price to be paid to allow the 'Soft Syndrome' decoder to work in a quantum processor with noisy syndrome measurements.

Finally, one important effect is the extra error correction can be reached by quantization. Similar to what happened in classical decoding, the introduction of non-linearity in the quantized messages may improve the error correction in message-passing algorithms [35]. Further research is required to exploit this property.

5.2 Hardware architecture and simplification of the algorithm

Before starting with the hardware architecture it is necessary to understand intuitively what makes the algorithm work under noisy conditions in the measurement of the syndromes. The main changes are in case 1, when the syndrome is not reliable enough ($|\gamma_i| \leq \Gamma$). Case 2 is just the 'Hard Syndrome' decoder that treats the syndrome as reliable, similar to the 'Perfect Syndrome' scenario.

[2] 8-bit finite precision analysis for 'Perfect syndrome' decoders is omitted in the figure for simplicity as it achieves the same performance as the full precision version.

As detailed in Sect. 3, for case 1 we can distinguish two different updates, those that affect the magnitude of the messages $\mu_{i,j}$ (Eq. (5) and (6)) and those that affect the syndrome s_i (Eq. (7) and (8) to compute the *revised* syndrome). For the former, the algorithm needs to choose between the minimum magnitude of $\nu_{i,j'}$ and γ_i, which means that the magnitude (i.e., reliability) of the generated messages will be determined by the least reliable information at the check node, which can be the input soft syndrome (γ_i) or the LLRs ($\nu_{i,j'}$) from variables connected to that check node. The new update rules for the check node messages $\mu_{i,j}$ determine the convergence of the algorithm to a correct error pattern.

On the other hand, for the latter, the change of the sign of the syndrome is useful to *revise* the incoming (measured) hard syndromes and allow the decoder to implement the halting criterion, i.e., the estimated error matches the revised syndrome. Note that if the measured syndromes are noisy and no update has been performed, then the syndromes obtained by the iterative process will never match the measured syndromes and, hence, no halting will be implemented, as explained in Sect. 3. However, for quantum processors, the limiting parameter is latency in the worst-case scenario and not average latency. So, including or not a halting criterion will not improve the final hardware characteristics of the decoder. On the contrary, it will add extra hardware (extra registers or memories to store old syndrome values, extra wiring to feedback information from one iteration to another, etc.), which is not necessary for the convergence of the algorithm. For this reason, the derived architecture will not add the extra hardware required to implement the syndrome update and it will just focus on the convergence of the algorithm by introducing the new update rule of $|\mu_{i,j}|$ described in case 1. In other words, the decoder still uses the syndrome magnitude to decide whether to update the messages or not, but it does not *revise* the syndrome itself. Following this reasoning, when the syndrome is very noisy and case 1 is applied, the decoder will end when the maximum number of iterations is reached. When the syndromes are considered perfect, the decoder works just as the 'Hard Syndrome' decoder.

In Fig. 10 we can see the circuit scheme for the check node unit of the 'Soft Syndrome' decoder, including the above simplification. The extra inputs compared to the 'Hard Syn-

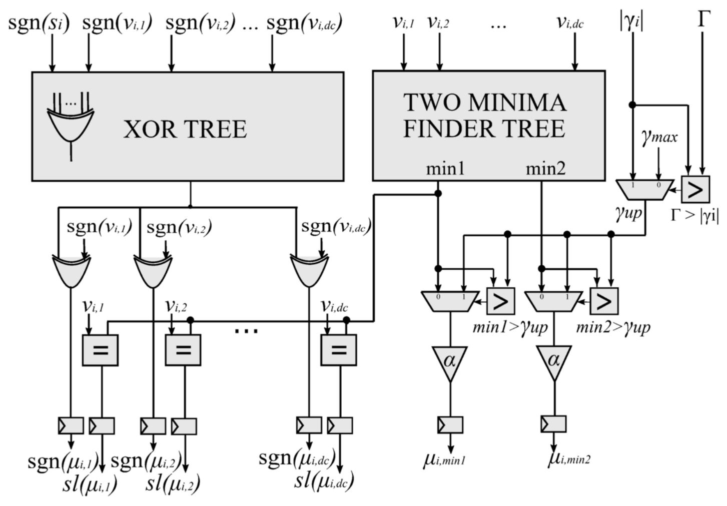


Figure 10 Architecture for a check-node unit for the 'Soft syndrome' decoder. *sl* bits indicate if the connected VNU contains the first minimum or not. γ_{up} is the maximum value of $|\gamma_i|$ that can be quantized

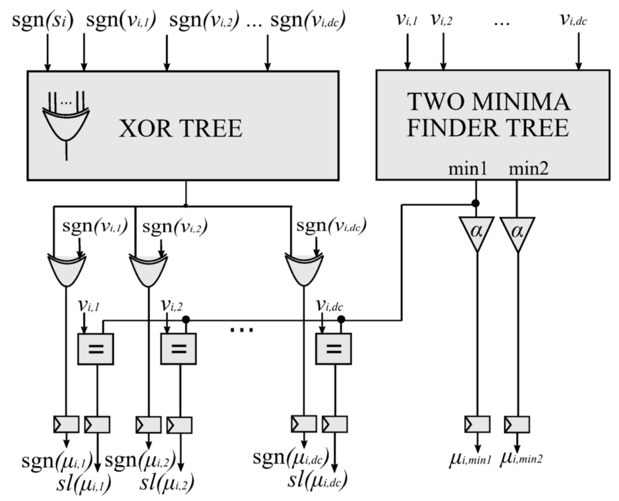

sgn(si) sgn(vi,1) sgn(vi,2) ... sgn(vi,dc)
vi,1 vi,2 ... vi,dc
XOR TREE
TWO MINIMA FINDER TREE
min1 min2
α
sgn(μi,1) sl(μi,1) sgn(μi,2) sl(μi,2) sgn(μi,dc) sl(μi,dc)
μi,min1 μi,min2

Figure 11 Architecture for check-node unit for the 'Hard syndrome' decoder

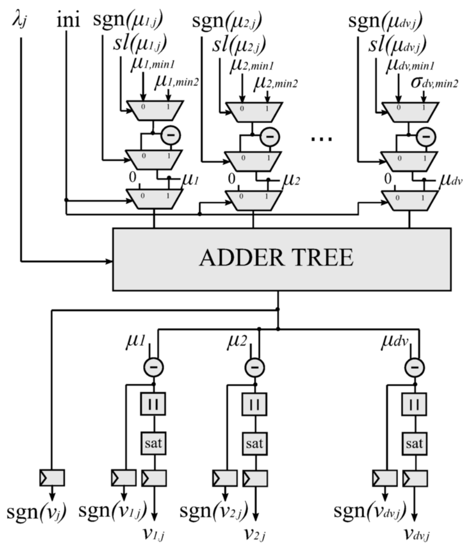

λj ini sgn(μ1,j) sl(μ1,j) μ1,min1 μ1,min2
sgn(μ2,j) sl(μ2,j) μ2,min1 μ2,min2
sgn(μdv,j) sl(μdv,j) μdv,min1 σdv,min2
ADDER TREE
sat
sgn(vj) sgn(v1,j) v1,j sgn(v2,j) v2,j sgn(vdv,j) vdv,j

Figure 12 Architecture for variable-node unit for the 'Soft syndrome' decoder

drome' decoder in Fig. 11 are $|\gamma_i|$ and $|\Gamma|$. The additional operations include the comparison necessary to decide if the syndrome is reliable enough, and the comparisons to update the messages $\nu_{i,j'}$ with the magnitude of the syndrome or the less reliable message received at the check node.

For the variable node, the main difference (see Fig. 12 and Fig. 13) is that the incoming messages for the initialization are no longer constants, one input is required to include the quantized information of $\lambda_j = \ln((1-p)/p)$. The second difference, not included in the figure but mentioned in the previous subsection, is that more bits are required to quantize λ_j. The complexity of the operations (adders in the variable-node unit and comparisons in the check-node unit) is larger, occupying more area and increasing the critical path.

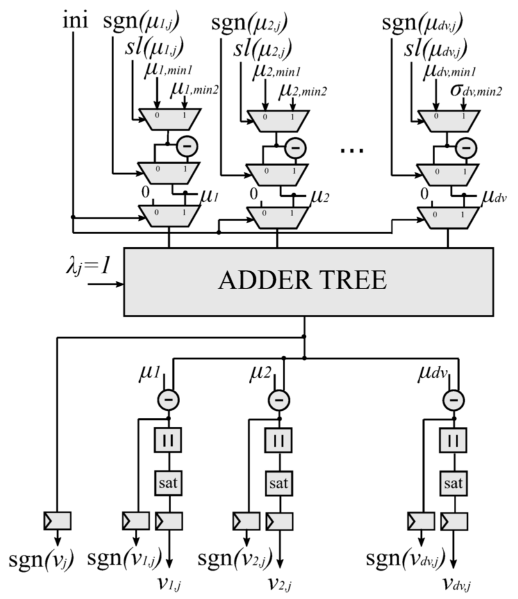


Figure 13 Architecture for variable-node unit for the 'Hard syndrome' decoder

5.3 Comparison of architectures and codes

In this section, we compare the full-parallel architecture for the 'Hard Syndrome' decoder from [36] to the 'Soft Syndrome' decoder introduced in this paper. The decoder was implemented for five different codes of different lengths and minimum distances ($[\![442,68,10]\!]$, $[\![544,80,12]\!]$, $[\![714,100,16]\!]$, $[\![1020,136,20]\!]$, and $[\![1428,630,24]\!]$) in order to see the effect of scaling the number of qubits in hardware results. All the architectures were implemented in a Xilinx FPGA xcvu095 (20 nm technology[3]) to allow a fair comparison. The 'Hard Syndrome' decoders were implemented with 6-bit quantization and the 'Soft Syndrome' decoders were implemented with 8-bit quantization to obtain a similar performance comparable to the full-precision versions. All the decoders were implemented with a maximum number of iterations equal to 30.

An FPGA was prioritized to GPUs or ASICs, as a platform to implement all the experiments for the following key aspects:

1. FPGAs have been extensively studied in environments with cryogenic temperatures, akin to those in which superconducting technology operates. References such as [37] demonstrate that although they exhibit different behavior at these temperatures compared to room temperature, they do not experience significant changes in terms of latency, which is a critical parameter.
2. Additionally, these devices are already integrated within the architecture of quantum computers for control tasks and have demonstrated robust performance, so they can be considered a realistic evaluation platform.
3. Power consumption is another crucial factor. FPGA devices are preferred over GPUs for decoding architectures. While a specific study comparing power consumption has not been conducted for quantum LDPC codes, other surveys have reported such comparisons for classical codes. FPGA devices are considered more efficient

[3]https://www.xilinx.com/products/silicon-devices/fpga/virtex-ultrascale.html.

Table 2 Summary of implementation results for the 'Soft Syndrome' decoder architectures with 8-bit quantization (xcvu095 FPGA device, 20 nm CMOS process)

Code $[\![n,k,d]\!]$	$[\![442,68,10]\!]$	$[\![544,80,12]\!]$	$[\![714,100,16]\!]$	$[\![1020,136,20]\!]$	$[\![1428,630,24]\!]$
Clock period	9.5 ns	10 ns	10 ns	10 ns	12 ns
Total latency (@30it)	570 ns	600 ns	600 ns	600 ns	720 ns
Lookup Table (LUT)	104,482 (19.4%)	128,004 (23.8%)	168,150 (31.3%)	237,422 (44.2%)	335,109 (62.3%)
Flip Flop (FF)	24,063 (2, 3%)	30,257 (2.814%)	39,684 (3.7%)	48,657 (4.5%)	79,267 (7.4%)
Power	1.67 W	1.94 W	2.39 W	4.26 W	4.59 W

compared to GPUs in power, area, and timing, as shown in [38]. Additionally, FPGA devices implement a customized architecture, reducing the exchange of information between the processing units of the decoder and the memory/registers. GPU architectures are more constrained in this regard. Even achieving similar timing results with GPUs would entail higher power consumption costs compared to FPGA devices.

4. While FPGAs may exhibit higher latency and power consumption in comparison to ASICs, the shorter design and verification time represent significant advantages. Given the current stage, where there isn't a definitive consensus on the optimal quantum LDPC code or decoder, we favored real FPGA implementation results over ASIC synthesis. The reconfigurable nature of FPGAs facilitates modifications to implemented designs and the testing of various configurations, encompassing different codes and decoders. This abbreviated design time, combined with FPGA reconfigurability, empowered us to implement and evaluate five different codes, each with two different decoders. Conducting such a comprehensive implementation analysis would have been impractical within the scope of ASIC design. To the best knowledge of the authors, ASIC implementation will become more critical as the number of protected qubits surpasses ten thousand. Until that juncture, satisfactory results can be attained through full custom FPGA design to perform fair comparisons of hardware implementations of QEC decoders.

The results are summarized in Tables 1 and 2.

For the timing results, it can be seen that the decoders from [36] have an almost constant delay, except when the number of check nodes grows significantly ($[\![1428,630,24]\!]$), in which case the delay increases due to the limitations of the internal routing of the FPGA device. Similar behavior can be observed for the 'Soft Syndrome' decoders, with the delay remaining almost constant for the three shorter codes and an increase of 2 ns for the denser code due to the routing. The difference of about 2 ns between both architectures is due to the extra bits needed to take into account the soft information in both syndromes and initialization messages. It is also because of the addition of comparators and multiplexers to implement the new update rules in case 1 of the algorithm (see the differences already mentioned between Fig. 10 and 11). Regarding the total latency of the decoders, we can see how 'Soft Syndrome' is beyond the most restrictive timing requirement reported in [39] for superconducting quantum processors, between 400 ns and 500 ns. To solve this issue, several options can be followed: the reduction of the number of iterations, the implementation in a more modern device, or the implementation over ASIC. The same happens with the largest code for the architecture described in [36].

Comparing the power consumption results, we can see that the 'Soft Syndrome' decoder consumes less power than the 'Hard Syndrome' decoder. This is due to the fact that the frequency of operation of our proposed decoder is smaller than the one that can be employed for the decoder from [36], reducing the overall power consumption. However, in all the cases the power consumption is beyond 1 W, which was reported as a possible power budget in [40] if the FPGA was set as close as possible to the quantum chip. To improve these results, other technology improvements and deeper analysis should be explored.

Finally, if we compare area results, we can see that the 'Soft Syndrome' decoder needs around 50% more LUTs than the 'Hard Syndrome' decoder in all cases. This comes from the increase in the number of bits and the extra hardware required to implement the magnitude update in case 1. The increase in Flip Flops (FFs)is almost negligible compared to the overall resources available in the FPGA. An important point to note is that no extra memories or feedback wiring is required, simplifying the whole architecture. If larger QLDPC codes need to be processed, then it would be necessary to migrate to larger devices as the interconnections between cells would be saturated with an occupation of more than 60%.

6 Conclusions and future directions

Despite recent advancements in QLDPC code constructions, one still needs to develop fast and effective decoders for these codes under practical settings where the syndrome measurement process can itself be noisy. In this work, we considered the MSA-based decoder for QLDPC codes. We showed that standard MSA is affected severely by syndrome noise, and proposed a modified MSA that is robust to a reasonable amount of syndrome noise even without repeated measurements. In this paper, we demonstrated how it is possible to obtain efficient hardware implementations of this new decoder with almost constant delay and a moderate increase of area resources compared to standard MSA decoders. The results are promising for practical applications even for the most restrictive technologies such as superconducting quantum processors. In future work, we will extend our analysis to further investigate systematic ways [41] to understand iterative decoding under noisy syndromes, and also consider realistic circuit-level noise models.

Funding
This work is generously supported by the National Science Foundation (NSF) under grants: CCF-2100013, CCF-2106189, CCSS-2027844, CCSS-2052751, ERC-1941583, CIF-1855879, NASA under the SURP Program, and by the U.S. Department of Energy, Office of Science, National Quantum Information Science Research Centers, Superconducting Quantum Materials and Systems Center (SQMS) under the contract No. DE-AC02-07CH11359. This work is also supported by the QuantERA grant EQUIP with the grant number PCI2022-132922, funded by Agencia Estatal de Investigación, Ministerio de Ciencia e Innovación, Gobierno de España, MCIN/AEI/10.13039/501100011033, and by the European Union "NextGenerationEU/PRTR".

Abbreviations
BP, belief propagation; CMOS, complementary metal-oxide-semiconductor; CNU, check-node unit; CSS, Calderbank, Shor, Steane; FPGA, field-programmable gate array; LLR, log-likelihood ratio; LP, lifted product; MSA, min-sum algorithm; QEC, quantum error correction; QC, quasi-cyclic; QLDPC, quantum low-density parity-check; VNU, variable-node unit.

Availability of data and materials
The models used during the current study are available from the corresponding author upon reasonable request.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare no competing interests.

Author contributions
Nithin Raveendran, Asit Kumar Pradhan, Narayanan Rengaswamy, and Bane Vasic conceptualized the algorithm and were responsible for its development and simulation results in software. Nithin Raveendran, Asit Kumar Pradhan, Francisco Herrero, and Narayanan Rengaswamy contributed equally to writing the algorithm description, while Javier Valls Coquillat and Bane Vasic provided critical feedback and revised the manuscript. Javier Valls Coquillat and Francisco Herrero were responsible for the hardware development, which included designing and building the experimental FPGA setup including data collection and analysis. All authors have read and approved the final version of the manuscript.

Author details
[1]Department of Electrical and Computer Engineering, The University of Arizona, Tucson, USA. [2]Instituto de Telecomunicaciones y Aplicaciones Multimedia, Universitat Politecnica de Valencia, Valencia, Spain. [3]Department of Computer Architecture and Automatics (DACYA), Complutense University of Madrid, Madrid, Spain.

Received: 5 May 2023 Accepted: 8 October 2023 Published online: 19 October 2023

References
1. MacKay DJC, Mitchison G, McFadden PL. Sparse-graph codes for quantum error correction. IEEE Trans Inf Theory. 2004;50(10):2315–30. https://doi.org/10.1109/TIT.2004.834737.
2. Babar Z, Botsinis P, Alanis D, Ng SX, Hanzo L. Fifteen years of quantum LDPC coding and improved decoding strategies. IEEE Access. 2015;3:2492–519.
3. Gottesman D. Fault-tolerant quantum computation with constant overhead. Quantum Inf Comput. 2014;14(15–16):1338–72.
4. Leverrier A, Tillich J, Zémor G. Quantum expander codes. In: Proc. IEEE 56th ann. symp. on foundations of computer science, Berkeley, CA, USA. 2015. p. 810–24.
5. Breuckmann NP, Eberhardt JN. Quantum low-density parity-check codes. PRX Quantum. 2021;2(4):040101. https://doi.org/10.1103/PRXQuantum.2.040101.
6. Hastings MB, Haah J, O'Donnell R. Fiber bundle codes: breaking the $n^{1/2}$ polylog(n) barrier for quantum LDPC codes. In: Proc. of the 53rd ann. ACM SIGACT symp. on theory of computing, New York, NY, USA. 2021. p. 1276–88. https://doi.org/10.1145/3406325.3451005.
7. Breuckmann NP, Eberhardt JN. Balanced product quantum codes. IEEE Trans Inf Theory. 2021;67(10):6653–74. https://doi.org/10.1109/TIT.2021.3097347.
8. Panteleev P, Kalachev G. Asymptotically good quantum and locally testable classical LDPC codes. In: ACM symposium on theory of computing, Rome, Italy: ACM; 2022. p. 375–88.
9. Leverrier A, Zémor G. Quantum Tanner codes. arXiv preprint. 2022. arXiv:2202.13641.
10. Bravyi S, Cross AW, Gambetta JM, Maslov D, Rall P, Yoder TJ. High-threshold and low-overhead fault-tolerant quantum memory. arXiv preprint. 2023. arXiv:2308.07915.
11. Tremblay MA, Delfosse N, Beverland ME. Constant-overhead quantum error correction with thin planar connectivity. Phys Rev Lett. 2022;129(5):050504. https://doi.org/10.1103/physrevlett.129.050504.
12. Xu Q, Ataides JPB, Pattison CA, Raveendran N, Bluvstein D, Wurtz J, Vasić B, Lukin MD, Jiang L, Zhou H. Constant-overhead fault-tolerant quantum computation with reconfigurable atom arrays. 2023. arXiv:2308.08648.
13. Panteleev P, Kalachev G. Degenerate quantum LDPC codes with good finite length performance. Quantum. 2021;5:585.
14. Kuo K-Y, Lai C-Y. Refined belief propagation decoding of sparse-graph quantum codes. IEEE J Sel Areas Inf Theory. 2020;1(2):487–98. https://doi.org/10.1109/jsait.2020.3011758.
15. Liu YH, Poulin D. Neural belief-propagation decoders for quantum error-correcting codes. Phys Rev Lett. 2019;122:200501. https://doi.org/10.1103/PhysRevLett.122.200501.
16. Rigby A, Olivier JC, Jarvis P. Modified belief propagation decoders for quantum low-density parity-check codes. Phys Rev A. 2019;100:012330. https://doi.org/10.1103/physreva.100.012330.
17. Raveendran N, Vasić B. Trapping sets of quantum LDPC codes. Quantum. 2021;5:562. https://doi.org/10.22331/q-2021-10-14-562.
18. Ashikhmin A, Lai CY, Brun TA. Correction of data and syndrome errors by stabilizer codes. In: Proc. IEEE intl. symp. inf. theory. 2016. p. 2274–8.
19. Ashikhmin A, Lai C-Y, Brun TA. Quantum data-syndrome codes. IEEE J Sel Areas Commun. 2020;38(3):449–62.
20. Bombín H. Single-shot fault-tolerant quantum error correction. Phys Rev X. 2015;5(3):031043.
21. Campbell ET. A theory of single-shot error correction for adversarial noise. Quantum Sci Technol. 2019;4(2):025006.
22. Pattison CA, Beverland ME, da Silva MP, Delfosse N. Improved quantum error correction using soft information. arXiv preprint. 2021. arXiv:2107.13589.
23. Vasić B, Ivanis P. Error Errore Eicitur: a stochastic resonance paradigm for reliable storage of information on unreliable media. IEEE Trans Commun. 2016;64(9):3596–608.
24. Vasić B, Ivanis P, Declercq D. Approaching maximum likelihood performance of LDPC codes by stochastic resonance in noisy iterative decoders. In: Inf. theory and applications workshop, San Diego. 2016.
25. Raveendran N, Nadkarni PJ, Garani SS, Vasić B. Stochastic resonance decoding for quantum LDPC codes. In: Proc. IEEE intl. conf. commun. 2017. p. 1–6.
26. Calderbank AR, Shor PW. Good quantum error-correcting codes exist. Phys Rev A. 1996;54(2):1098–105. https://doi.org/10.1103/physreva.54.1098.
27. Chen J, Dholakia A, Eleftheriou E, Fossorier MPC, Hu X-Y. Reduced-complexity decoding of LDPC codes. IEEE Trans Commun. 2005;53(8):1288–99.
28. Kuo K-Y, Chern I-C, Lai C-Y. Decoding of quantum data-syndrome codes via belief propagation. In: Proc. IEEE intl. symp. inf. theory. 2021. p. 1552–7.

29. Fawzi O, Grospellier A, Leverrier A. Efficient decoding of random errors for quantum expander codes. In: Proc. 50th ann. ACM SIGACT symp. on theory computing, Los Angeles, CA, USA. 2018. p. 521–34. https://doi.org/10.1145/3188745.3188886.
30. Panteleev P, Kalachev G. Quantum LDPC codes with almost linear minimum distance. IEEE Trans Inf Theory. 2022;68(1):213–29. https://doi.org/10.1109/TIT.2021.3119384.
31. Fossorier MPC. Quasicyclic low-density parity-check codes from circulant permutation matrices. IEEE Trans Inf Theory. 2004;50(8):1788–93. https://doi.org/10.1109/TIT.2004.831841.
32. Tanner RM, Sridhara D, Sridharan A, Fuja TE, Costello JDJ. LDPC block and convolutional codes based on circulant matrices. IEEE Trans Inf Theory. 2004;50(12):2966–84. https://doi.org/10.1109/TIT.2004.838370.
33. Raveendran N, Rengaswamy N, Rozpedek F, Raina A, Jiang L, Vasić B. Finite rate QLDPC-GKP coding scheme that surpasses the CSS Hamming bound. Quantum. 2022;6:767. https://doi.org/10.22331/q-2022-07-20-767.
34. Panteleev P, Kalachev G. Degenerate quantum LDPC codes with good finite length performance. Quantum. 2021;5:585. https://doi.org/10.22331/q-2021-11-22-585.
35. Declercq D, Danjean L, Li E, Planjery SK, Vasic B. Finite alphabet iterative decoding (FAID) of the (155,64,20) Tanner code. In: 6th intl. symp. on turbo codes and iterative information processing. 2010. p. 11–5. https://doi.org/10.1109/ISTC.2010.5613861.
36. Valls J, Garcia-Herrero F, Raveendran N, Vasić B. Syndrome-based min-sum vs OSD-0 decoders: FPGA implementation and analysis for quantum LDPC codes. IEEE Access. 2021;9:138734–43. https://doi.org/10.1109/ACCESS.2021.3118544.
37. Homulle H, Visser S, Patra B, Ferrari G, Prati E, Sebastiano F, Charbon E. A reconfigurable cryogenic platform for the classical control of quantum processors. Rev Sci Instrum. 2017;88(4):045103.
38. Ferraz O, Subramaniyan S, Chinthala R, Andrade J, Cavallaro JR, Nandy SK, Silva V, Zhang X, Purnaprajna M, Falcao G. A survey on high-throughput non-binary LDPC decoders: ASIC, FPGA, and GPU architectures. IEEE Commun Surv Tutor. 2022;24(1):524–56. https://doi.org/10.1109/COMST.2021.3126127.
39. Das P, Pattison C, Manne S, Carmean D, Svore K, Qureshi M, Delfosse N. A scalable decoder micro-architecture for fault-tolerant quantum computing. arXiv preprint. 2020. https://doi.org/10.48550/ARXIV.2001.06598. arXiv:2001.06598.
40. Tannu SS, Myers ZA, Nair PJ, Carmean DM, Qureshi MK. Taming the instruction bandwidth of quantum computers via hardware-managed error correction. In: 2017 50th ann. IEEE/ACM intl. symp. on microarchitecture (MICRO). 2017. p. 679–91.
41. D'Anjou B. Generalized figure of merit for qubit readout. Phys Rev A. 2021;103:042404. https://doi.org/10.1103/PhysRevA.103.042404.
42. Raveendran N, Rengaswamy N, Pradhan AK, Vasić B. Soft syndrome decoding of quantum LDPC codes to correct of data and syndrome errors. In: IEEE intl. conf. on quantum computing and engineering (QCE). 2022. https://arxiv.org/abs/2205.02341.

Publisher's Note