貪欲法を用いた表面符号向けエラー訂正復号器の FPGA・ASIC 実装

青山 連[1,a)]　門本 淳一郎[2]　小林 和淑[1]

概要：実用的な計算をできる量子コンピュータの実現には誤り訂正機能が必要である．表面符号は代表的な誤り訂正手法のひとつであり高い誤り訂正機能を持つ．本研究では誤り訂正機能の一部である復号器をハードウェア記述言語である Verilog HDL を用いて設計した．設計にあたって，貪欲法と呼ばれるアルゴリズムを使用した．設計した FPGA を対象としたものと ASIC を対象としたものの 2 種類の復号器を 180nm プロセス，22nm プロセスを用いて論理合成を行った．FPGA を対象とした合成では使用リソースとして LUT と FF，ASIC 向けの合成では面積と消費電力を求めた．この結果を同様に反復貪欲法を用いた先行研究と比較した．比較の結果，FPGA 対象の合成において LUT は 229%，FF は 19%となり，ASIC 対象の合成において面積は 0.872%，消費電力は 2.96%となった．

FPGA and ASIC Implementations of a Surface Code Error Correction Decoder Using a Greedy Algorithm

Aoyama Ren[1,a)]　Kadomoto JUNICHIRO[2]　Kobayashi KAZUTOSHI[1]

Abstract: Error correction is mandatory to realize a quantum computer that can perform practical calculations. Surface codes are one of the error correction methods and have high error correction capability. In this study, we designed a decoder, which is a part of the error correction function, using Verilog HDL. An algorithm called the greedy method was used in the design. The decoder was synthesized using 180-nm process and a 22-nm cell libraries. Two types of synthesis were performed, one for FPGAs and the other for ASICs: LUTs and FFs were used as resources for synthesis for FPGAs, and area and power consumption were obtained for synthesis for ASICs. The results were compared with those of previous studies that also used the greedy method. The results showed that the LUT and FF used 229% and 19% of the resources for FPGA synthesis, respectively, while the area and power consumption for ASIC synthesis were 0.872% and 2.96%, respectively.

1. はじめに

現在の半導体技術はムーアの法則に従い微細化，発展してきた．しかし，微細化による性能向上にも限界が指摘されてきた．これに対してコンピュータの応用範囲は拡大し，求められる計算量も増加している．これに対応するための新たなコンピュータとして量子コンピュータが注目されている．量子コンピュータとは量子もつれや重ね合わせ状態など，量子力学的性質を利用したコンピュータである．その特性を利用することにより高速な計算を可能にするとされている．しかし，現在の量子コンピュータのエラー率は実用的な計算を行うには非常に高いため，エラー訂正機能が必須である．

エラーを訂正するための量子誤り訂正符号の中でも量子ビットを 2 次元平面上に並べた表面符号は誤り訂正しきい値が高く注目されている [1]．表面符号におけるエラー訂正ではエラーを検出した量子ビットの位置の情報をもとに正しいエラー位置を推定する必要がある．その処理を行う部分を復号器と呼ばれ，研究が行われている [2]，[3]，[4]．

本稿では表面符号を対象として，貪欲法を用いた復号器

[1] 京都工芸繊維大学
Kyoto Institute of Technology
[2] 東京大学
The University of Tokyo
[a)] raoyama@vlsi.es.kit.ac.jp

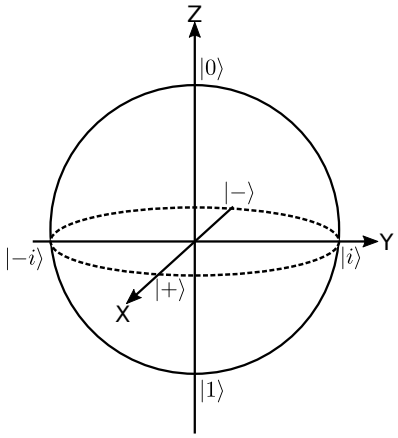

図 1　ブロッホ球

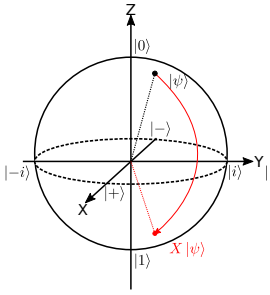

図 2　X エラー

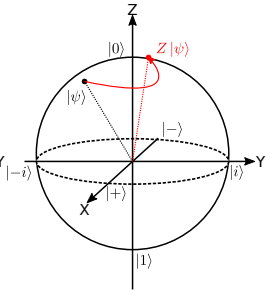

図 3　Z エラー

を設計し，論理合成を行った結果について述べる．第 2 節では量子誤り訂正について述べる．第 3 節では設計した復号器について述べる．第 4 節では復号器を論理合成した結果について述べる．第 5 節で結論を述べる．

2. 量子誤り訂正

2.1 量子コンピュータにおけるエラー

量子コンピュータにおける情報の基本単位は量子ビットである．量子ビットの状態 $|\psi\rangle$ は二つの基底状態 $|0\rangle, |1\rangle$ と複素数 α, β を用いて $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ と表され $|\alpha|^2 + |\beta|^2 = 1$ である．量子ビットの状態は**図 1** に示すブロッホ球を用いて表される．

量子コンピュータには古典コンピュータにもみられるビット反転エラー以外のエラーが存在する．X エラーは古典コンピュータと同様のビット反転エラーであり，量子ビットの状態は $X|\psi\rangle = \alpha|1\rangle + \beta|0\rangle$ と表される．Z エラーは量子コンピュータ特有のもので，位相反転エラーと呼ばれ，$Z|\psi\rangle = \alpha|0\rangle - \beta|1\rangle$ と表される．それぞれのエラーを**図 2**，**図 3** に示す．この他に，Y エラーと呼ばれるものもあり，これは X エラーと Z エラーの組み合わせで表される．

2.2 表面符号

表面符号は，物理量子ビットを 2 次元平面上に配置し 1 つの論理量子ビットを構成する誤り訂正符号である．表面符号の構成を図 4 に示す．符号を構成する物理量子ビット

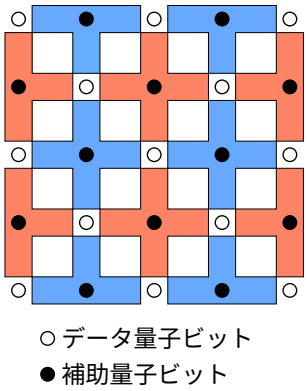

図 4　表面符号の概略図

は情報を保持するデータ量子ビットと，データ量子ビットに発生したエラーを検知する補助量子ビットに分けられる．補助量子ビットは隣接したデータ量子ビットを観測し，エラーの有無を調べる [5]．これにより得られる測定値をエラーシンドロームと呼び，この情報をもとに訂正処理が行われる．量子ビットには 2 種類のエラーがあるため，補助量子ビットも 2 種類に分けられる．

表面符号の復号方法について述べる．補助量子ビットは隣接した奇数個のデータ量子ビットにエラーが発生すると，1 を示す．この 1 を示した補助量子ビットをアクティブノードと呼ぶ．量子コンピュータが計算している間，一定間隔毎に補助量子ビットの観測が行われる．この間隔をコードサイクルと呼ぶ．

表面符号における復号とは，補助量子ビットから得られるエラーシンドロームをもとにデータ量子ビットのエラー位置を推定し訂正することである．この推定するという問題には，近似的に解くことができる方法を用いることが一般的である．代表的なものとして，アクティブノードを頂点とし，アクティブノード間のマンハッタン距離を重みとしたグラフの最小重みマッチング問題として扱う手法である．復号器ではアクティブノード間のマンハッタン距離が最小となるようなアクティブノードの組を決定し，そのノード間にエラーが起きたデータ量子ビットが存在すると推定する．

エラーを推定する際，推定し訂正するエラーはデータ量子ビットのみではなく補助量子ビットの測定エラーについても行う必要がある．測定エラーについては，複数回補助量子ビットを観測することで対応できる．ある時間の観測結果とそのあとの観測結果，その二つの観測結果の XOR 値を重ねた 3 次元グラフ上でマッチング問題を解くことで測定エラーの検知，訂正を行うことができる．

3. 設計した復号器

3.1 使用アルゴリズム

今回設計した復号器では貪欲法と呼ばれるアルゴリズム [6] を用いた．貪欲法はスケーラビリティと小さなレイ

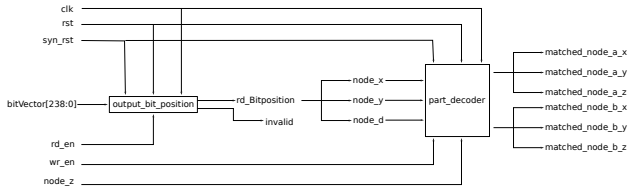

図 5 復号器の概略図

テンシを示すことが可能であり，先行研究でも用いられている [3], [4], [7]．ただし，得られる解が大域的最適解ではなく局所的最適解となる可能性がある．

先行研究 [7] では同様に貪欲法を用いた復号器が設計されている．先行研究で用いられている貪欲法は反復貪欲法と呼ばれる異なったアルゴリズムである．実装が容易で先に述べたような性質を持つため本研究では通常の貪欲法を用いている．

3.1.1 貪欲法の動作

貪欲法の動作の概要を以下に示す．

(1) ノードを 1 つ選択する．
(2) 選択したノードとそれ以外の全ノードとのマンハッタン距離を求める．
(3) 最も距離の短いノードの組を出力する．
(4) 2 つ目以降のノードにも同様の処理を行う．

3.1.2 反復貪欲法の動作

反復貪欲法の動作の概要を以下に示す．

(1) 許容コストを 1 とする．
(2) 貪欲法と同様の手順でマッチングを行う．このペアのコストが許容コストより大きければマッチング不成立とする．
(3) 許容コストを増やし手順（2）を繰り返す．
(4) すべてのノードがマッチングされるまで繰り返す．

3.2 復号器の構成

復号器の構成の概要を**図 5** に示す．表面符号上のエラーを示す情報 (bitVector) を入力とする．bitVector の各ビットが表面符号上の補助量子ビットに対応しており，エラーを検出した補助量子ビットに対応したビットが 1 となる．bitVector のビット数はほ場量子ビットの数に依存する．今回の設計では符号距離 15 を想定している．これは物理量子ビットのエラー率が $p = 10^{-4}$，論理量子ビットのエラー率が $p_L = 10^{-15}$ の場合，十分な符号距離であるとされているからである [8]．bitVector は図 5 内の output_bit_position でエラーの起きたビットの桁数 (rd_Bitposition) に変換する．また，このときしきい値以上のエラーが検出された場合，復号不可として invalid が 1 となる．ra_Bitposition は対応した座標へと変換され，part_decoder に入力される．part_decoder では入力された座標感のマンハッタン距離を求め，最短となる座標の組み合わせを求める．この最短距離を求める際に貪欲法を用いている．

3.2.1 output_bit_position の動作

output_bit_position の動作の概要を次に示す．

(1) bitVector の各桁が 0 か 1 かを判定する．
(2) 1 だった場合，その桁数を保存する．ただし，エラー数がしきい値を超えた場合 invalid を 1 として終了．
(3) 順次，桁数を読み出して出力する．

3.2.2 part_decoder の動作

part_decoder 内部の構成を**図 6** に示す．part_decoder は [9] で設計したものを用いている．入力したノードを格納する node_stack とマンハッタン距離を計算し，最短距離となるノードの組を推定する decoder に分かれる．part_decoder の動作の概要を次に示す．

(1) アクティブノードの座標を入力とし，node_stack へ格納する．
(2) node_stack から 2 つのノードを読み出し，ノード間のマンハッタン距離を計算する．
(3) 手順 2 で求めたマンハッタン距離が最短であればそのノードの組を出力とする．
(4) 全ノードの組み合わせでこの処理を行い，最短距離となるノードの組を決定する．
(5) ノードが決定するとその情報を出力すると同時に，そのノードが格納されていた stack のアドレスにフラグを立てる．
(6) フラグを立てたアドレスの stack からはノードが読み出されない．
(7) これにより stack 内のフラグの立っていないアドレスからノードを読み出し，最短距離となるノードの組み合わせを求める．
(8) stack 内のノードすべての組が決まるまで処理を繰り返す．

マンハッタン距離の計算方法について述べる．ノード 0 の座標 (x_0, y_0, z_0)，境界までの距離 d_0, ノード 1 の座標 (x_1, y_1, z_1)，境界までの距離 d_1 とする．ノード間のマンハッタン距離 d_m と境界を超えたときのノード間の距離 d_b はそれぞれ次のように求められる．d_m と d_b のより短いほうが最終的なノード間の距離 d として記録される．

$$d_m = |x_0 - x_1| + |y_0 - y_1| + |z_0 - z_1|$$
$$d_b = d_0 + d_1$$
$$d = \min(d_m, d_b)$$

node_stck の概略図を図 7 に示す．node_stack は入力されたノードを格納する stack と各 stack のアドレスに対応する flag の 2 種類の配列からなる．格納可能なノードの数をエントリ数と呼ぶ．flag が 1 となっているアドレスの stack からは読み出しが行われない．

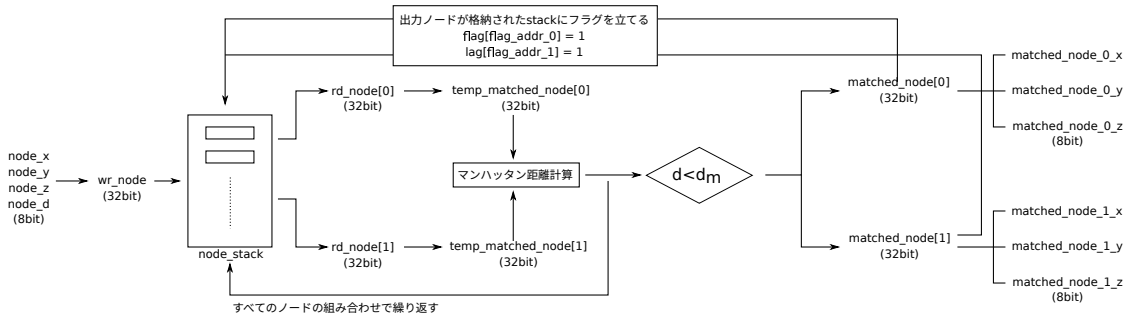

図 6　part_decoder の構成

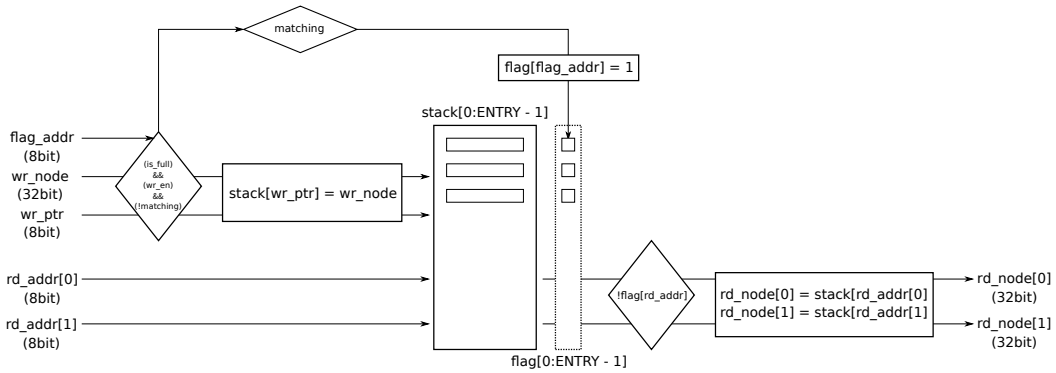

図 7　node_stack の概略図

4. 復号器の論理合成

4.1 合成の詳細

本研究では，設計した復号器を FPGA，ASIC を対象としてそれぞれ論理合成を行った．その結果について反復貪欲法を用いた先行研究 [7] と比較を行う．

4.1.1 FPGA 向けの合成

FPGA を対象として論理合成を行った．ターゲットとした FPGA は zcu104 評価ボードである．合成結果の比較項目として使用リソースを確認した．

表 1 に先行研究で設計された復号器と本研究で設計した復号器の使用リソースを示す．先行研究に対し，本研究の復号器は LUT は 229%，FF は 19%の割合となった．

4.1.2 ASIC 向けの合成

設計した復号器を ASIC に向けて Design Compiler を用いて論理合成を行った．そのときの電力と面積を確認し，比較を行った．使用したライブラリは，先行研究との比較をするために 180nm プロセスと，本研究における ASIC 化に向け，22nm プロセスを利用している．

表 2 に先行研究と本研究の合成結果を示す．先行研究に対して電力は 19.0%，面積は 27.3%となった．

本研究で設計した復号器を使用するライブラリを変更し論理合成を行った結果について述べる．電力は 2.96%，面積は 0.872%であった．

表 1　FPGA 使用リソース

	LUT	FF
先行研究	14191	9111
本研究	32557	1766

表 2　ASIC を対象とした合成結果の比較

	プロセス	周波数 MHz	電力 mW	面積 mm^2
先行研究	180nm	N/A	130	6.83
本研究		33	24.7	1.87
	22nm	100	0.731	0.0163

4.2 復号器の評価

FPGA 向けの合成について，反復貪欲法は for 文によるループ回数が多くなっている．このため，表 1 に示すように FF の数が多くなっていると考える．

ASIC 向けの合成において 180nm プロセスと 22nm プロセスについて，プロセスの変化以上の面積の減少が確認できる．それぞれのライブラリを確認すると，配線層の数が 180nm プロセスでは 5 層，22nm プロセスでは 9 層となっている．また，インバータのセルの大きさに顕著な差が見られた．180nm プロセスでは 12 ピッチで 0.8μm であった．セルの大きさを式 1 のように求められる．

$$\begin{aligned} w \times h &= 2.4\mu\mathrm{m} \times 9.6\mu\mathrm{m} \\ &= 23.04\mu\mathrm{m}^2 \end{aligned} \tag{1}$$

22nm プロセスのライブラリでは 7 ピッチで 0.1μm であるので式 2 のように求められる．

$$w \times h = 0.28\mu\text{m} \times 0.7\mu\text{m} = 0.196\mu\text{m}^2 \quad (2)$$

(1) と (2) を比較する 22nm プロセスのものは 180nm プロセスの 0.85%となり面積の差と近い値となる．このことからプロセスの違いによる面積の差はライブラリのピッチによる差であると言える．

5. 終わりに

本校では表面符号を対象とした貪欲法を用いた復号器の論理合成について述べた．論理合成は FPGA を対象としたものと ASIC を対象とした 2 種類を行った．ASIC 対象としたものは 180nm プロセスと 22nm プロセスを用いて行った．FPGA を対象とした合成の結果は先行研究に比べ，LUT は 229%，FF は 19%であった．180nm プロセスを使用した合成の結果は先行研究に比べ，電力は 19%，面積は 27.3%となった．設計した復号器の 180nm プロセスと 22nm プロセスを用いた合成結果を比較すると，電力は 2.96%，面積は 0.872%となった．これらの差の要因について述べた．

謝辞 本研究は，JST ムーンショット型研究開発事業グラント番号 JPMJMS226A の支援を受けたものです．

参考文献

[1] Fowler, A. G., Mariantoni, M., Martinis, J. M. and Cleland, A. N.: Surface codes: Towards practical large-scale quantum computation, *Phys. Rev. A*, Vol. 86, p. 032324 (online), DOI: 10.1103/PhysRevA.86.032324 (2012).

[2] Das, P., Pattison, C. A., Manne, S., Carmean, D., Svore, K., Qureshi, M. and Delfosse, N.: A scalable decoder micro-architecture for fault-tolerant quantum computing, *arXiv preprint arXiv:2001.06598* (2020).

[3] Ueno, Y., Kondo, M., Tanaka, M., Suzuki, Y. and Tabuchi, Y.: Qecool: On-line quantum error correction with a superconducting decoder for surface code, *2021 58th ACM/IEEE Design Automation Conference (DAC)*, IEEE, pp. 451–456 (2021).

[4] Liao, W., Suzuki, Y., Tanimoto, T., Ueno, Y. and Tokunaga, Y.: WIT-Greedy: Hardware System Design of Weighted ITerative Greedy Decoder for Surface Code, *Proceedings of the 28th Asia and South Pacific Design Automation Conference*, pp. 209–215 (2023).

[5] 徳永裕己ほか：量子コンピュータ: 3. 量子コンピュータの誤り訂正技術-物理に即したトポロジカル表面符号，情報処理， Vol. 55, No. 7, pp. 695–701 (2014).

[6] Drake, D. E. and Hougardy, S.: A simple approximation algorithm for the weighted matching problem, *Information Processing Letters*, Vol. 85, No. 4, pp. 211–213 (online), DOI: https://doi.org/10.1016/S0020-0190(02)00393-9 (2003).

[7] 中村徹舟， 宮村信，井上弘士，川上哲志，阪本利司，多田宗弘，谷本輝夫：極低温不揮発 FPGA を対象とした誤り耐性量子コンピュータ向け表面符号復号器の RTL 設計，情報処理学会論文誌コンピューティングシステム（ACS），Vol. 17, No. 1, pp. 13–25 (2024).

[8] Suzuki, Y., Sugiyama, T., Arai, T., Liao, W., Inoue, K. and Tanimoto, T.: Q3DE: A fault-tolerant quantum computer architecture for multi-bit burst errors by cosmic rays, *2022 55th IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pp. 1110–1125 (online), DOI: 10.1109/MICRO56248.2022.00079 (2022).

[9] 青山連, 門本淳一郎, 小林和淑：誤り耐性量子コンピュータに向けた 22nm バルクプロセスによる表面符号用エラー訂正復号器の設計，電子情報通信学会技術報告 (VLSI 設計技術)，No. VLD2023-38, pp. 49–53 (2023).