SCIS 2013 The 30th Symposium on Cryptography and Information Security
Kyoto, Japan, Jan. 22 - 25, 2013
The Institute of Electronics,
Information and Communication Engineers

グリッチ PUF を用いた鍵生成 LSI の試作
LSI Implementation of Device Key Generator using Glitch PUFs

鈴木 大輔* Daisuke SUZUKI
清水 孝一* Koichi SHIMIZU
菅原 健* Takeshi SUGAWARA
汐崎 充† Mitsuru SHIOZAKI
藤野 毅‡ Takeshi FUJINO

あらまし 近年，セキュリティチップで実現されている耐タンパ性を ASIC や FPGA などの汎用 LSI で実現可能な技術：Physical Unclonable Functions (PUF) が注目されている．これまでに我々は，ある１つのゲートにおける遅延のばらつきがグリッチの形状に大きな影響を与えることを利用した PUF の構成法 Glitch PUF (GPUF) を提案している．本稿では, 65nm CMOS プロセス上で試作した GPUF による鍵生成 LSI の性能評価結果を報告する．特に，本報告では GPUF におけるグリッチ生成回路の構成と PUF の基本性能であるユニーク性と環境変化に対するロバスト性の関係を中心に報告を行う．

キーワード Physical Unclonable Function，LSI implementation，Evaluation

1 はじめに

金融取引や入退室管理などセキュリティが求められる様々な場面では，IC カードや暗号ボードに代表される高度な暗号機能を持つ暗号ハードウェアが広く利用されている．暗号ハードウェアが広く利用される理由の１つに，そのセキュリティレベルの高さが挙げられる．例えば，IC カードの主要コンポーネントであるセキュリティチップには，暗号技術を用いるために不可欠である秘密情報 (鍵情報) への不正なアクセスを検知し，情報漏えいを防止する技術 “耐タンパ技術” が適用されている．

近年，前述したセキュリティチップで実現されている耐タンパ性を ASIC や FPGA などの汎用 LSI でも実現可能な技術：Physical Unclonable Function (PUF) [1] が注目されている．PUF は LSI に代表される個々の人工物が持つ物理的な特徴量を応じて，与えられたチャンレンジ（挑戦）に対し，レスポンス (応答) を返すように設計されたシステムである．その人工物が持つ特徴量は製造ばらつきによって発生し，特徴量が同じになる人工物を複製することは困難であるとされている．加えて，ノイズを含む特徴量から安定した秘密情報を生成する Fuzzy Extractor [2] (FE) を PUF の応答に対して実行することにより，複製困難なデバイス固有鍵を生成することが可能となる [3]. この鍵情報は，セキュリティチップのように不揮発性のメモリに格納せずとも再生成することで利用可能なため，チップを開封して直接格納されたデータを読み取るような解析に対して耐性を持つ．特に，汎用 LSI が持つ特徴量を用いた PUF の実現方式やデバイス固有鍵生成に関する研究 [4, 5, 6, 7, 8, 9, 10, 11] が盛んに行われている．現在，LSI 上での PUF の実現は SRAM-PUF [5] に代表されるメモリセルの特徴量を用いる方式と，Arbiter PUF [6] のような回路遅延のばらつきを特徴量とする方式に分けられる．特に後者の方式は総称して Delay-PUF と呼ばれている．

SRAM-PUF に関しては，FE で必要な誤り訂正符号や汎用ハッシュ関数の実装についても最適化されており [10, 11], 現在最も実現性の高い PUF の１つとして知られている．しかしながら，SRAM-PUF や Butterfly-PUF [8] のようなメモリセルの電源投入直後における不安定状態を利用した PUF は，一旦消去した鍵を再現できるタイミングが限られるという問題がある．また，FPGA のように，予め入手可能なデバイスであれば特徴量が持つ情報量を評価可能であるが，ASIC への実装を

* 三菱電機株式会社 情報技術総合研究所 〒 247-8501 神奈川県鎌倉市大船 5-1-1. Mitsubishi Electric Corporation, Information Technology R & D Center, 5-1-1 Ofuna Kamakura, Kanagawa, 247-8501, Japan.
† 立命館大学総合理工学研究機構 〒 525-8577 滋賀県草津市野路東 1-1-1. Research Organization of Science & Engineering, Ritsumeikan University, 1-1-1 Nojihigashi, Kusatsu, Shiga, 525-8577, Japan.
‡ 立命館大学大学院理工学研究科 〒 525-8577, 滋賀県草津市野路東 1-1-1, Graduate School of Science and Technology, Ritsumeikan University, 1-1-1 Nojihigashi, Kusatsu, Shiga, 525-8577, Japan.

想定した場合，動作モデルのみが提供される SRAM セルの特徴量を設計段階で評価することは難しいと考えられる．

そこで我々はこれまでに文献 [12, 13] において鍵の再現に関する制約がなく, 一般的な論理回路設計法で実装可能な Glitch PUF を提案している．本稿では，文献 [14] で報告した Glitch PUF を用いた鍵生成回路について，その評価結果を報告する．特に，本報告では Glitch PUF におけるグリッチ生成回路の構成と PUF の基本性能であるユニーク性と環境変化に対するロバスト性の関係を中心に報告を行う．

以下, 本稿の構成について述べる．第 2 章では, Glitch PUF の基本動作について述べる. 第 3 章では, 設計・試作した Glitch PUF のテストチップについて説明し，第 4 章及び第 5 章ではテストチップの評価方法と評価結果をそれぞれ示し, 第 6 章で本稿のまとめを行う．

2 Glitch PUF

2.1 Glitch PUF の動作原理

Glitch PUF（以下, GPUF）は，論理回路を構成する各ゲートの入出力信号間の遅延関係によって発生するグリッチと呼ばれる現象を利用した PUF の一構成法である．

以下, 図 1 に示す簡単な論理回路でその原理を説明する．図 1 のような，複数の入力信号に対して，AND や XOR などの論理演算を行う回路においては，一般に，各信号の遅延差によって，グリッチと呼ばれる信号の過渡遷移が発生する．図 1 では，入力信号 $(x1, x2, x3)$ が全て 0 から 1 に変化する場合，$x1, x2$ の信号変化の時間差によって，まず XOR ゲートの出力に凸状のグリッチが発生する．そして次に，$x3$ の変化が，このグリッチよりも早く AND ゲートに到達すれば，このグリッチは AND ゲートの出力に伝播する．逆に，$x3$ の変化が，このグリッチよりも遅く AND ゲートに到達すれば，このグリッチは AND ゲートの出力には伝播しない．さらに，もし $x3$ の方が早い場合であっても，AND ゲートの PATHPULSE 特性 [15] 次第で，幅の短いグリッチが出力に伝播しないといったことも起こりうる．ただし，十分に幅の長いグリッチに限定すれば，そのグリッチ形状は，Arbiter-PUF[6] などの Delay-PUF と同様に，遅延の相対関係によって確定し，その形状は動作環境が変化しても維持されることが期待できる．

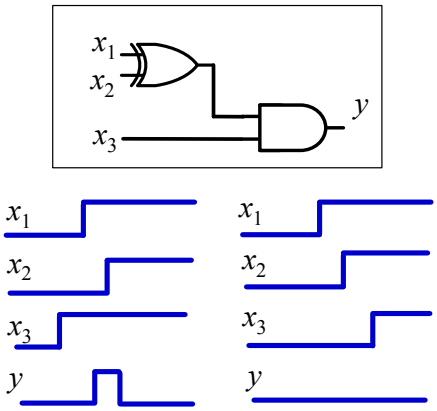

図 1: 遅延値の違いによる回路の挙動

Algorithm 1 鍵生成処理における GPUF の動作

Setting: m ビット入力 n ビット出力のランダムロジック f に対して入力信号に状態遷移 $x'_i \to x_i$ によって発生する各出力信号での立ち上がりエッジの偶奇数 $b_i = \mathrm{G2R}_f(x'_i \to x_i)$,
n ビットデータ $b_i = (b_{i,1}, b_{i,2}, \cdots, b_{i,n})_2$.

Input: $l \cdot m$ ビットデータ $X = (x_1, x_2, \cdots, x_l)$,
繰り返し回数 $\mathsf{cnt_{re}}$, 安定性判定閾値 $\mathsf{th_{err}}$.

Output: $(W, S_{\mathrm{mask}}) \leftarrow \mathrm{GPUF}_{\mathsf{Gen}}(X, \mathsf{cnt_{re}}, \mathsf{th_{err}})$,
$l \cdot n$ ビット PUF 応答 $W = (w_1, w_2, \cdots, w_l)$,
$l \cdot n$ ビット補助データ $S_{\mathrm{mask}} = (s_1, s_2, \cdots, s_l)$.

```
for i = 1 to l do
    cnt[1 : n] ← all 0
    for j = 1 to cnt_re do
        b_i = G2R_f(0 → x_i)
        for k = 1 to n do
            cnt[k] ← cnt[k] + b_{i,k}
        end for
    end for
    for j = 1 to n do
        if (cnt[j] ≤ th_err) then
            w_{i,j} ← 0, s_{i,j} ← 1
        else if (cnt_re − 1 − th_err ≤ cnt[j]) then
            w_{i,j} ← 1, s_{i,j} ← 1
        else
            w_{i,j} ← 0, s_{i,j} ← 0
        end if
    end for
end for
return W, S_mask;
```

我々は, GPUF の基本コンセプトを文献 [12] で提案し, その後改良を加えたバージョンを文献 [13] で示した．ここで, Alg. 1 及び Alg. 2 として文献 [13] における GPUF の動作を擬似コード化した記述を示す．Alg. 1 は鍵生成時における GPUF の動作を表す．鍵生成時は, PUF 応答のエラーレートを下げるために, 同一の入力状態遷移に対してエッジの偶奇判定を複数回行い, その出力が安定的かを検査する処理を行う．具体的には, 繰り返し回数 $\mathsf{cnt_{re}}$ で規定される回数分繰り返し処理を行う．その後,

Algorithm 2 鍵再現処理における GPUF の動作

Setting: m ビット入力 n ビット出力のランダムロジック f に対して入力信号に状態遷移 $x'_i \to x_i$ によって発生する各出力信号での立ち上がりエッジの偶奇数 $b_i = \mathrm{G2R}_f(x'_i \to x_i)$,
n ビットデータ $b_i = (b_{i,1}, b_{i,2}, \cdots, b_{i,n})_2$.

Input: $l \cdot m$ ビットデータ $X = (x_1, x_2, \cdots, x_l)$,
$l \cdot n$ ビット補助データ $S_{\mathrm{mask}} = (s_1, s_2, \cdots, s_l)$,
繰り返し回数 $\mathsf{cnt_{re}}$.

Output: $W \leftarrow \mathrm{GPUF}_{\mathsf{Rep}}(X, S_{\mathrm{mask}}, \mathsf{cnt_{re}})$,
$l \cdot n$ ビット PUF 出力 $W = (w_1, w_2, \cdots, w_l)$.

1: **for** $i = 1$ to l **do**
2: $\quad \mathrm{cnt}[1:n] \leftarrow$ all 0
3: $\quad$ **for** $j = 1$ to $\mathsf{cnt_{re}}$ **do**
4: $\quad\quad b_i = \mathrm{G2R}_f(0 \to x_i)$
5: $\quad\quad$ **for** $k = 1$ to n **do**
6: $\quad\quad\quad \mathrm{cnt}[k] \leftarrow \mathrm{cnt}[k] + b_{i,k}$
7: $\quad\quad$ **end for**
8: $\quad$ **end for**
9: $\quad$ **for** $j = 1$ to n **do**
10: $\quad\quad$ **if** $\mathrm{cnt}[j] < \mathsf{cnt_{re}}/2$ **then**
11: $\quad\quad\quad w_{i,j} \leftarrow 0$
12: $\quad\quad$ **else**
13: $\quad\quad\quad w_{i,j} \leftarrow 1$
14: $\quad\quad$ **end if**
15: $\quad$ **end for**
16: $\quad w_i \leftarrow w_i \cap s_i$
17: **end for**
18: **return** W;

出力ビット毎に安定性判定閾値 $\mathsf{th_{err}}$ を基準としてビットの安定性に関する検査を行い, 基準を満たさないビットはそのビットに対応する補助データ S_{mask} のビット値を 0 とすることで鍵生成及び再現時に 0 として扱う処理を行う. これに対し, Alg.2 に示す鍵再現時における Glitch PUF の動作では, 繰り返し処理は生成時と同様に実施するが, 最終的な応答は多数決判定と S_{mask} によるマスク処理で決める.

2.2 Glitch PUF の回路構成

図 2 に GPUF の回路構成を示す. 図中 “Glitch count regiser” は, 立ち上がりエッジの偶奇数を判定する信号をクロックとするフリップ・フロップ (FF) である. この FF は, 判定処理を行う直前に図中 “clear” 信号によってリセットされる.GPUF の情報量やビット誤り率は, “Random logic f” の回路構成に依存する. 過去の FPGA 上における GPUF の評価から, ビット誤り率は論理段数が深くなるほど上昇すると予想されるため, 本稿で述べる試作チップでは論理段数の異なる複数の f を選択可能な構成としている. 繰り返し処理によるビット誤り率の低減や利用しないビットの決定は図中 “Counter & Comparator” で実行される. これらの計算に必要なレジスタサイズは $\mathsf{cnt_{re}}$ によって決定される.3.1.3 項に挙げた例である $\mathsf{cnt_{re}} = 7$ であれば, b_i の各ビットに対してそれぞれ 3 ビットのレジスタを用いることで実装できる.

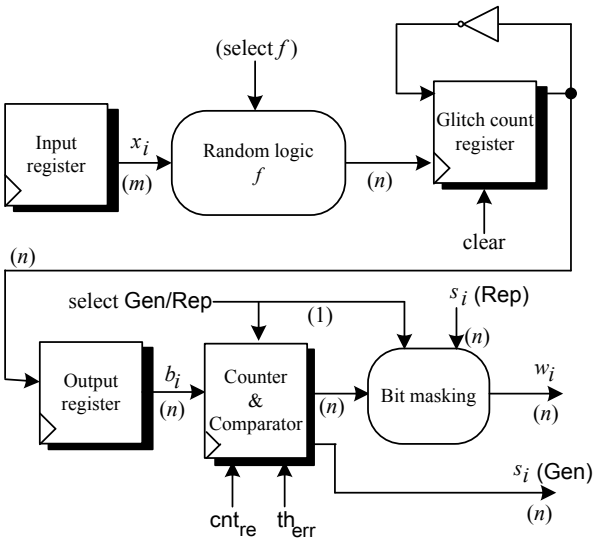

図 2: GPUF 回路

3 テストチップの設計及び試作

本章では, 富士通の 65nm CMOS スタンダードセル・ライブラリを用いて試作した, テストチップの概要と GPUF の回路設計について述べる.

3.1 テストチップの全体構成

図 3 にテストチップの全体構成を示す. テストチップは, I2C slave のインターフェースを持つチップとして動作する. また, 本チップは GPUF による FE による鍵生成機能を備える. 鍵生成処理に必要な補助データは, 図中の SRAM に格納され, 鍵生成回路がマスタとなってリード/ライトすることで鍵生成及び鍵再現に必要なデータのハンドリングを行う.

GPUF は, ランダムロジックの構成と PUF としての性能の関連性を実測で評価するために, 複数のランダムロジックを搭載し, 外部コマンドによって使用するランダムロジックを切り替えられる構成としている. また, ランダムロジックとして, 評価の目的毎に **TYPE 1** と **TYPE 2** の 2 種類を実装している. 加えて, チップ間及びチップ内での PUF としての性能を評価する目的で, 同一モジュール, 同一レイアウトの GPUF を種類毎に 4 つ搭載し, 物理的なモジュールを選択できる構成としている. また, 繰り返し回数 $\mathsf{cnt_{re}}$ や安定性判定閾値 $\mathsf{th_{err}}$ についても外部からパラメータを選択できる構成としている.

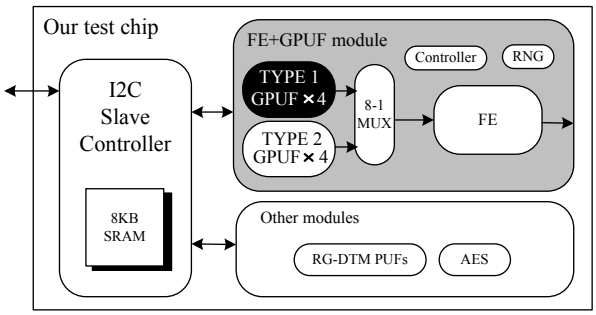

図 3: テストチップの全体構成

表 1: GPUF のデザインパラメータ

デザイン パラメータ	テストチップの 設定値
入力ビット長	$n = 32$（固定）
出力ビット長	$m = 32$（固定）
レスポンス数	$l \in \{4, 8, \cdots, 512\}$ (可変)
繰り返し回数	$\mathsf{cnt_{re}} \in \{1, 3, 7, \cdots, 255\}$（可変）
安定性判定閾値	$0 \leq \mathsf{th_{err}} \leq \lfloor \mathsf{cnt_{re}}/2 \rfloor - 1$（可変） 但し, $\mathsf{cnt_{re}} = 1$ の場合, $\mathsf{th_{err}} = 0$ のみ利用可

その他のモジュールを含めたデザインパラメータを表 1 に示す.

このほか, 本テストチップには, 文献 [17] で提案されている RG-DTM PUF やサイドチャネル評価用の AES などが実装されており, 多目的テストチップとして全体設計が行われている.

3.2 グリッチ生成回路

前述のようにテストチップでは GPUF のグリッチ生成回路として **TYPE 1** と **TYPE 2** の 2 種類を実装している. 紙面の関係上, ここでは本稿で採り上げる **TYPE 1** について説明を行う.

図 4 に示される回路構成を持つ **TYPE 1** は，我々が文献 [16] で提案するブロック暗号の処理，物理乱数発生及び PUF の機能を統合したコプロセッサでのグリッチ生成回路を想定したものであり, AES のデータパスを利用した回路構成である. AES として動作する場合は図 4 の “chain logic” と示される AND ゲートはすべて 0 を出力するように制御する. 一方，GPUF として動作する場合は, 図 4 の chain logic をオンにすることで変更することができる. 例えば，図中左端の chain logic をオンにすると左端の SubBytes 出力のパスがその右側の SubBytes 入力として合流することになるため, およそ SubBytes 一つ分のロジック段数が増加することになる. 定性的には,

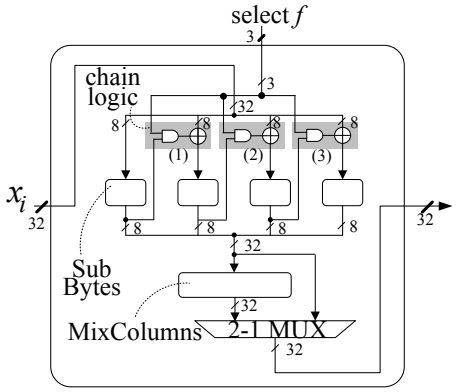

図 4: **TYPE 1** のグリッチ生成回路

表 2: グリッチ生成回路の種類

グリッチ 生成回路	chain logic (1)	(2)	(3)	データ パス
GG1	OFF	OFF	OFF	1 S + 1 M
GG2	ON	OFF	OFF	2 S + 1 M
GG3	ON	ON	OFF	3 S + 1 M
GG3	ON	ON	ON	4 S + 1 M

S: SubBytes, M: MixColumns

論理段数が増えることで遅延ばらつきによるグリッチの振る舞いはより複雑化され, 結果として PUF としてのユニーク性が増加することが期待できる. 一方で, グリッチの振る舞いが複雑になれば, 温度や電圧に対してセンシティブなパスが増加することと予想され,PUF のとしてのロバスト性が低下することが予想される.

第 4 章及び 5 章では, このトレードオフについて chain logic のオン/オフと，PUF の諸性能の関係を評価する. 尚，本稿では chain logic のオン/オフとグリッチ生成回路の対応を，表 2 に示される表記を用いる.

3.3 レイアウト

テストチップのレイアウトを図 5 に示す. チップサイズは 2.5mm × 2.5mm であり，電源電圧は 1.2V でメタルは 7 層である. レイアウトは Cadence Encounter Digital Implementation System を用いて実施している. 図中, 中央に配置された大きさの異なる 2 種類の正方形区画は，同一レイアウトでマクロ化して配置した 8 つの GPUF 回路である. GPUF は他のモジュールとは独立に合成する. また, レイアウト実行時はフロアプランされた領域に GPUF 単体で配置配線を行い, その結果をマクロ化する. このマクロをフロアプランで確保された同一面積の複数領域にマップする. 以上のフローによって, 合成時の仮負荷, 及びレイアウト後の実負荷抽出によって得られる遅延情報ファイルは, 同じタイプの GPUF 間で同一となる.

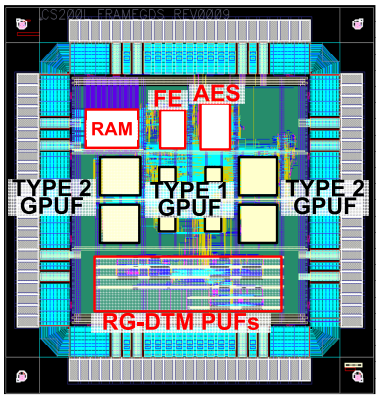

図 5: テストチップのレイアウト

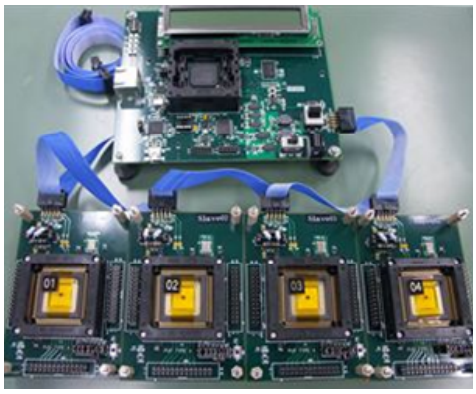

図 6: テストチップの評価環境

尚，図 5 で正方形区画のうち，回路面積が小さい区画に **TYPE 1** のグリッチ生成回路が実装されている．

4 評価方法

4.1 評価環境

図 6 にテストチップの評価環境を示す．テストチップは PC から図中上側の FPGA ボード経由で制御される．また，温度変化や電圧変化に関するテストは恒温槽と FPGA ボードに搭載されたポテンショメータの制御によって行う.FPGA とテストチップ間の接続は耐熱ケーブルで接続し，テストチップを搭載したボードのみ温度変化を与える．評価における温度, 電圧の設定は共にデータロガーによる計測値で自動調整する．以下では特に断りがない限り, 温度はテストチップの表面温度を指し，電圧はテストチップへ供給するコア電圧を指す．

4.2 評価項目

まず，以下で使用する基本的な表記法を説明する．$\mathsf{HD}(A, B)$ はビット列 A, B のハミング距離を表し，$A \cap B$ はビット毎の AND 処理を表す．$\mathsf{HW}(A)$ は A のハミング重みを意味する．

テストチップに対する評価項目を表 3 にまとめる．評価は評価項目 5 を除き，テストチップ 16 石に対して実施している．表 3 で，評価対象の欄は，チップ間で同一の位置に実装された GPUF 間の比較を単にチップ間と表記する．また，同一チップ内で同一レイアウトにより実装された GPUF 間の評価をチップ内と表記し，(チップ間, チップ内) を (Y,N) と表記した場合，チップ内で同じ位置に実装された GPUF をチップ間で比較することを意味する．また，(N,Y) と表記された評価項目は，チップ内の同一レイアウトの GPUF 間での比較をチップの数だけ行うこと意味する．計測回数の欄は，評価対象あたりに同一のチャレンジに対するレスポンスを取得する回数を意味する．チャレンジはすべてランダムなビット列を発生させている．

評価項目 1　表 3 の評価項目 1 では，温度変化範囲 (0°C ∼ 85°C) 及び電圧変化範囲 (1.20V±5%) におけるすべてのコーナケースでのビットエラーレート BER を評価する．BER は Alg.1 の表記法を用いた場合，次のように表される．

$$\mathrm{BER} := \mathsf{HD}(W^{\mathrm{n}} \cap S^{\mathrm{n}}_{\mathrm{mask}}, W^{\mathrm{c}} \cap S^{\mathrm{n}}_{\mathrm{mask}}) / \mathsf{HW}(S^{\mathrm{n}}_{\mathrm{mask}})$$

ここで，A^{n} は常温・常電圧下 (27°C, 1.20V) における GPUF の出力を表し，A^{c} は同じ GPUF のコーナーケースにおける出力を表す．コーナーケースは 4 通り存在するが，FE における誤り訂正符号の訂正能力を決定するために重要となる BER の最大値を評価する．

評価項目 2　評価項目 2 では，常温・常電圧下において，Alg.1 における S_{mask} の各ビットが有効になる確率

$$P_r[s_{i,j} = 1] := \mathsf{HW}(S^{\mathrm{n}}_{\mathrm{mask}}) / (l \cdot n)$$

を評価する．すなわち，生成したレスポンスのうち，利用可能なレスポンスの発生効率を示す指標となる．

評価項目 3　評価項目 3 では，常温・常電圧下における，チップ間の平均ハミング距離

$$\mathrm{AHD} := \mathsf{HD}(W^{i} \cap S^{i}_{\mathrm{mask}}, W^{j} \cap S^{i}_{\mathrm{mask}}) / \mathsf{HW}(S^{i}_{\mathrm{mask}}) \quad (i \neq j)$$

を評価する．ここで，A^{i} は i 番目の GPUF におけるデータを表す．この評価は一般に PUF のユニーク性と呼ばれる指標であり，同一チャレンジに対して，チップ間でどの程度異なるレスポンスがが返されるかを示す指標である．

評価項目 4　評価項目 4 は，評価項目 3 の評価を，チップ内の GPUF 間で行う．チップ 1 石あたり 4 つの GPUF が存在するため，1 石あたり 12 通りの組み合わせに対して平均ハミング距離を評価する．この目的は，GPUF のユニーク性がウェハ内で近接していても，保証されるかを評価することにある．

表 3: 評価項目

評価項目	評価内容の概要	評価対象 チップ間　チップ内	計測回数
1	環境変化に対する BER の最大値	(Y,N)	100
2	利用可能なレスポンスの発生効率	(Y,N)	1
3	チップ間での GPUF 間のレスポンスのユニーク性	(Y,N)	1
4	チップ内での GPUF 間のレスポンスのユニーク性	(N,Y)	1
5	経年劣化による BER の最大値	(Y,N)	100

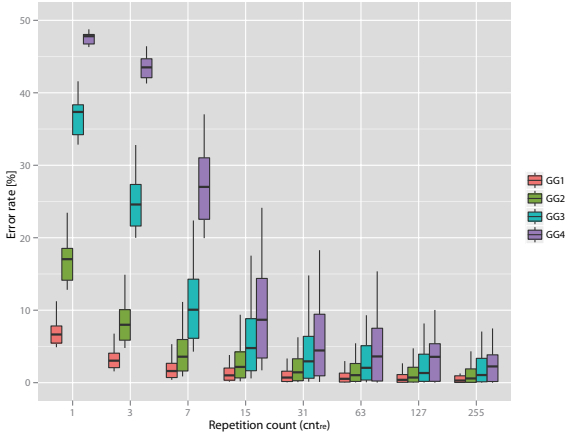

図 7: 環境変化に対する BER の最大値

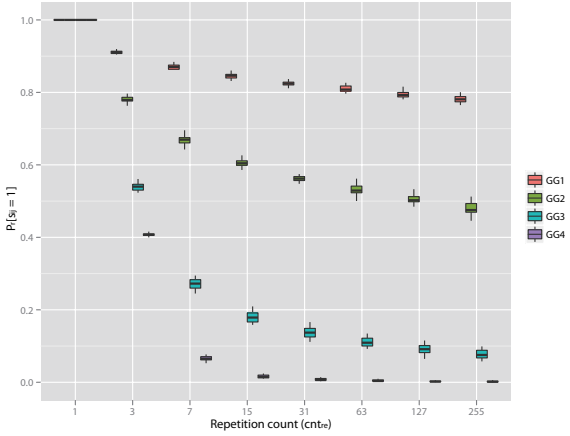

図 8: 利用可能なレスポンスの発生効率

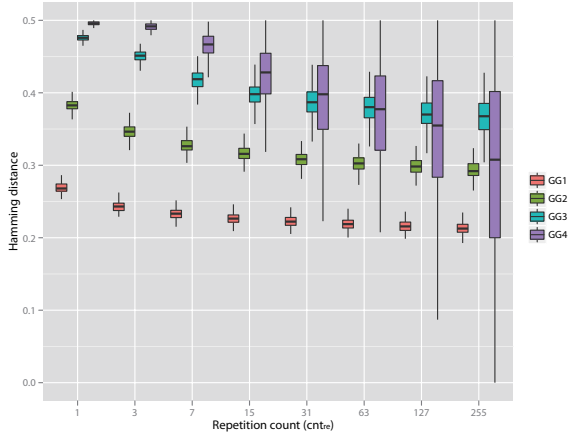

図 9: チップ間での GPUF 間のレスポンスのユニーク性

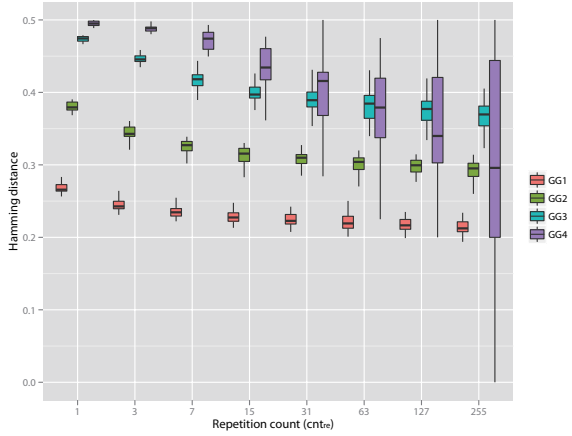

図 10: チップ内での GPUF 間のレスポンスのユニーク性

評価項目 5　評価項目 5 では，高温 (125°C) 及び高電圧 (1.32V) 下で GPUF の連続運転を行うことで劣化の加速を行い，加速前後のチップに対して常温・常電圧下におけるレスポンスの BER を評価する．高温・高電圧下での動作は計 336 時間まで実施している．本評価に用いたチップは評価項目 4 までに用いた 16 石とは異なる個体を用いており，2 石に対して実施している．

加速試験は様々な方法があるが，ここでは温度と電圧を加速要因とした一般的な定ストレス法を用いる．加速のモデル式は以下のような関係が知られている．

$$\text{温度加速モデル式 : 寿命 } L_1 = A_1 \cdot \exp(E_a/kT)$$
$$\text{電圧加速モデル式 : 寿命 } L_2 = A_2 \cdot \exp(-\beta/V)$$

ここで，E_a は活性化エネルギー,k はボルツマン定数，T は絶対温度，β は電圧加速係数,V は印加電圧であり，A_1，A_2 は定数である．一例として $E_a = 0.5$ や $\beta = 0.8$ とし，使用温度を 60°C とするならば，加速係数は 37.84 となり，336 時間の試験はおよそ 1.5 年に相当する計算となる．

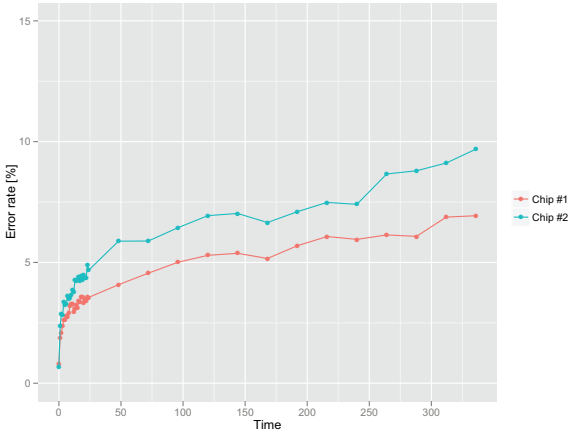

図 11: 経年劣化による BER の最大値

5 評価結果

評価項目 1 図 7 に評価項目 1 の結果である環境変化に対する BER の最大値を示す．図 7 は 16 石に対する BER の最大値をボックスプロットしたものである．横軸は Alg.1 における繰り返し回数 $\mathrm{cnt_{re}}$ であり，縦軸が BER である．図 7 からグリッチ生成回路の論理段数が増加するにつれて BER が増加する．逆に $\mathrm{cnt_{re}}$ の増加に従い，BER が低減することがわかる．一方で，$\mathrm{cnt_{re}} = 7$ あるいは $\mathrm{cnt_{re}} = 15$ までは BER が急速に低減するのに対して，それ以上は低減率が低いことがグラフから読み取れる．FE での鍵生成を想定した場合，利用可能なリソースにもよるが，15% 程度を BER の上限とするならば，$\mathrm{cnt_{re}} = 7$ では GG1 及び GG2 がグリッチ生成回路として利用可能であることがわかる．

一方で，文献 [16] で我々が提案する物理乱数発生器としての GPUF の応用を考えるならば，BER が高いことが望ましい．この場合，$\mathrm{cnt_{re}} = 1$ で 50% 付近の BER を持つ GG4 がグリッチ生成回路としてふさわしい選択であると言える．

評価項目 2 図 8 に評価項目 2 の結果である利用可能なレスポンスの発生効率を示す．横軸は図 7 と同じ設定である．図 8 から利用可能なレスポンスの発生効率は，グリッチ生成回路の依存性が非常に高く,GG3 と GG4 では生成したレスポンスのほとんどのビットが Alg.1 で不安定で利用できないビットと判定される．一方，GG1 及び GG2 では，評価項目 1 での繰り返し回数の効率を考慮にいれるならば，$\mathrm{cnt_{re}} = 7$ における発生効率は GG1 はおよそ 9 割，GG2 では 7 割程度のレスポンスが利用できる．従って，レスポンスの発生効率の観点からもグリッチ生成回路としては GG1 及び GG2 が適切な選択であると言える．

評価項目 3 図 9 に評価項目 3 の結果であるチップ間での GPUF 間に対するレスポンスのユニーク性を示す．横軸は図 7 と同じ設定である．図 9 からグリッチ生成回路の論理段数が増加するにつれて平均ハミング距離 AHD がより理想値である 0.5 に近づくことがわかる．評価項目 1 及び 2 の結果を考慮に入れた場合，GG2 が GG1 よりもおよそ 1 割程度平均ハミング距離が高く,$\mathrm{cnt_{re}} = 7$ では 33% 程度となる．ここで，評価項目 1 の結果から，$\mathrm{cnt_{re}} = 7$ における GG2 のエラーレートはコーナケースを考慮しても 11% 程度であるため，チップをユニークに特定することが可能である．しかし，理想値である 50% にはならないため，今後改良が必要な項目であると考える．

評価項目 4 図 10 に評価項目 4 の結果であるチップ内での GPUF 間のレスポンスに対するユニークを示す．図 9 及び図 10 の比較から，チップ内，チップ間に依存せず，同一のユニーク性が得られることがわかる．

評価項目 5 図 11 に評価項目 5 の結果である経年劣化による BER の最大値を示す．ここでは，GG2 における $\mathrm{cnt_{re}} = 7$ の結果のみを示す．チップ間で特性にばらつきがあるが，初期段階の加速で急速にエラーレートが増加したあとはゆるやかに増加することがわかる．また，336 時間の加速では 15% を超えずに動作可能であることがわかる．一方で，図 11 からは，4 章で述べた加速係数が適切であれば，10 年などの長期間にわたり，鍵の再生成なしで同一の鍵を使い続けることは困難であることがわかる．

6 おわりに

本稿では, 65nm CMOS プロセス上で試作した GPUF による鍵生成 LSI の性能評価結果を報告した．特に，本報告では GPUF におけるグリッチ生成回路の構成と PUF の基本性能であるユニーク性と環境変化に対するロバスト性の関係を中心に報告を行った．

我々の評価結果から，本試作で用いたプロセス上においては，GPUF のグリッチ生成回路として AES の SubBytes 回路 2 段分と MixColums 回路がユニーク性とロバスト性の観点から利用可能であることがわかった．

また，GPUF のパラメータの一つである安定ビットを判別するための繰り返し回数とレスポンスの生成効率についてもその関係性を定量的に示した．

本稿では，評価の目的のために GPUF のデザインパラメータを外部から設定できる構成で LSI を開発したが，この方法は実用時にも有用であると考える．回路情報の IP 化を想定した場合，製造プロセス毎にグリッチ生成回路を調整することは開発コストが大きい．本構成のように暗号回路などのモジュールをグリッチ生成回路として利用することで，回路規模の増加を抑えつつ多様なプロセスに対応可能な IP とすることが可能と考える．

今後は他のプロセスでの試作評価や, ユニーク性の改善及び設計段階における PUF のシミュレーション精度向上を目的とした実測との誤差要因などを調査していく予定である.

謝辞

本研究の一部は, JST-CREST「ディペンダブル VLSI システムの基盤技術」の研究の一環として実施しました. また, 本チップ試作は株式会社トッパン・テクニカル・デザインセンター, (株) イー・シャトルおよび富士通株式会社の協力で行われたものです. 関係各位に感謝いたします.

参考文献

[1] R. S. Pappu: Physical One-way Functions. Ph.D. Thesis, M.I.T., `http://pubs.media.mit.edu/pubs/papers/01.03.pappuphd.powf.pdf`, 2001

[2] Y. Dodis, M. Reyzin, and A. Smith: Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. Eurocrypt 2004, LNCS 3027, pp. 523-540, Springer-Verlag, 2004.

[3] P. Tuyls and L. Batina: RFID-Tags for Anti-Counterfeiting. CT-RSA 2006, LNCS 3860, pp. 115-131, Springer-Verlag, 2006.

[4] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas: Silicon Physical Random Functions. Proc, of the 9th ACM Conference on Computer and Communications Security (CCS 2002), pp. 148-160, 2002

[5] J. Guajardo, S. S. Kumar, G.-J. S̃chrijen and P. Tuyls: FPGA Intrinsic PUFs and Their Use for IP Protection. CHES 2007, LNCS 4727, pp. 63-80, Springer-Verlag, 2007.

[6] J.-W. Lee, D. Lim, B. Gassend, G. E. Suh, M. van Dijk, and S. Devadas: A Technique to Build a Secret Key in Integrated Circuits for Identification and Authentication Applications. Proc. of the IEEE VLSI Circuits Symposium, pp 176-179, 2004.

[7] G. E. Suh and S. Devadas: Physical Unclonable Functions for Device Authentication and Secret Key Generation. Proc. of the 44th annual Design Automation Conference (DAC 2007), pp. 9-14, 2007.

[8] S. S. Kumar, J. Guajardo, R. Maes, G.-J. S̃chrijen and P. Tuyls: Extended Abstract: The Butterfly PUF: Protecting IP on every FPGA. Proc. of the IEEE International Workshop on Hardware-Oriented Security and Trust 2008 (HOST 2008), pp. 67-70, 2008.

[9] M. Majzoobi, F. Koushanfar and M. Potkonjak, : Lightweight secure PUFs. Proc. of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD 2008), pp. 670-673, 2008.

[10] C. B̈osch, J. Guajardo, A.-R. Sadeghi, J. Shokrollahi and P. Tuyls : Efficient Helper Data Key Extractor on FPGAs. CHES 2008, LNCS 5154, pp. 181-197, Springer-Verlag, 2007

[11] R. Maes, P. Tuyls, and I. Verbauwhede : Low-Overhead Implementation of a Soft Decision Helper Data Algorithm for SRAM PUFs. CHES 2009, LNCS 5747, pp. 332-347, 2009.

[12] D. Suzuki and K. Shimizu : The Glitch PUF: A New Delay-PUF Architecture Exploiting Glitch Shapes. CHES 2010, LNCS 6225, pp. 366-384, 2010.

[13] K. Shimizu and D. Suzuki : Glitch PUF: Extracting Information from Usually Unwanted Glitches. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, Vol. E95-A, No. 1, 2012.

[14] D. Suzuki, K. Shimizu, T. Tsurumaru, T. Sugawara, M. Shiozaki and T. Fujino : Device Key Generator using Glitch PUFs. SCIS2012, 4C2, 2012 (in Japanese).

[15] Standard Delay Format Specification version 3.0. `http://www.eda.org/sdf/sdf_3.0.pdf`, 1995.

[16] K. Shimizu, D. Suzuki, T. Tsurumaru, T. Sugawara, M. Shiozaki and T. Fujino : Unified Coprocessor Architecture for Secure Key Storage and Challenge-Response Authentication SCIS2013, 2012 (in Japanese).

[17] K. Fruhashi, M. Shiozaki, A. Fukushima, T. Murayama and T. Fujino : The arbiter-PUF with high uniqueness utilizing novel arbiter circuit with Delay-Time Measurement. Proc. of the IEEE International Symposium on Circuits and Systems 2008 (ISCAS 2008), pp. 2325 - 2328, 2008.