博士論文

組込みプロセッサに対するレーザーフォールト攻撃と対策に関する研究

A Study on Laser Fault Attacks and Countermeasures in Embedded Processors

国立大学法人 横浜国立大学
環境情報学府

坂本 純一
Junichi SAKAMOTO

2020 年 3 月

あらまし

スマートフォンやクレジットカードなど，重要な情報を格納するデバイスが我々の身近に数多く存在する．これらのデバイスは，紛失や盗難時にも内部の秘密情報を漏らすことのないよう暗号化やアクセス制御技術によってデータの秘匿性を担保している．これらのセキュリティ技術はある仮定の下でセキュアになるよう慎重に設計されており，論理的な世界でこれらを攻撃することは困難である．しかしながら，これらの技術は物理的な実体を持つデバイス上に実装され動作するため，実世界では論理的な世界では考慮されていなかった攻撃（物理的な攻撃）が可能である．例えば，多くのセキュリティ技術はデバイスやアルゴリズムが正しく動作するという仮定の元でセキュリティが保証されているため，物理的なデバイスの動作に誤りが発生した場合想定されたセキュリティが達成できない場合がある．攻撃者がデバイスに意図的に誤りを注入することも考えられるため，誤りが発生する前提でセキュリティ設計を行うことが望ましい．デバイスの動作に注入される誤りをフォールトと呼び，フォールトを利用した攻撃をフォールト攻撃と呼ぶ．フォールト攻撃はデバイスに想定外の動作を引き起こすため対策が難しく，さらに対策部分の動作すら誤る恐れがあることから強力な攻撃だと見なされている．

デバイスのセキュリティを保証するスキーム（ISO/IEC 15408 など）では，デバイスに対する物理攻撃耐性を評価するために，既知の物理攻撃を評価対象に実際に実行することで脆弱性が検証される．脆弱性検証の結果，攻撃成功に要するコストに応じたセキュリティレベルが保証される．このようなセキュリティ保証スキームの問題点は，未知の攻撃に対するセキュリティが考慮されないことであり，考えられる攻撃法を早急に洗い出し，攻撃が表面化する前に対策をとることが重要である．また前述の保証スキームでは攻撃が成功する場合でもそのコストが十分に高ければ安全という判断になるため，攻撃のコストを正確に見積もるということも重要である．本論文ではフォールト攻撃の中でも特に強力だとされている，レーザー照射によるフォールト注入に着目する．レーザー照射は回路のごく狭い範囲のみに誤りを注入できるためフォールト注入の自由度が高い一方で，高価な装置やレーザーを照射するためのデバイスの事前加工，さらに攻撃対象デバイスの回路レイアウトに対する深い知識などの必要性により攻撃に要するコストが高いとみなされてきた．いくつかのデバイスはレーザー攻撃にかかるコストが現実的ではないという理由でレーザー攻撃の脅威を考慮しておらず，もしも従来考えられていたよりも低コストでレーザー攻撃が実行できたとすれば多くのデバイスに対する重大な脅威となる恐れがある．本論文はデバイス上のフラッシュメモリにレーザーを照射することで“命令改変攻撃”と呼ばれる新たな攻撃が可能なことを示す．また命令改変攻撃は使用する機器の値段や攻撃者が持つ知識などの面で従来のレーザー攻撃よりも低コストで実行可能であることを明らかにする．さらに命令改変攻撃に適切な制限を課すことによりソフトウェアによる低コストな対策手法を提案し，命令改変攻撃が現実世界で悪用されることを事前に防ぐ．

本論文ではまず，フォールト攻撃とそれを取り巻く関連研究を 1 章で紹介する．IoT 社会のように大量のデバイスが身の回りにある環境においては，攻撃者のデバイスに対する物理的なアクセスが容易になり，デバイスが実装攻撃の脅威に晒される機会が増加する．セキュリティ保証スキームの枠組みで言えば

これはデバイスへのアクセスという攻撃のコストが下がることを意味しており，IoT 機器が従来と同程度のセキュリティを達成するためにはより高い耐タンパー性が求められる．本論文で着目するレーザー照射によるフォールト攻撃は，非常に強力な一方で攻撃コストが高すぎるため現実的ではないとされてきた．本論文ではレーザー照射攻撃に精密さを求めるよりも低コストで攻撃に利用できるフォールトを注入することが大きな脅威につながるのではないかと考え，フラッシュメモリへのレーザー照射を利用した命令改変という手法を提案する．攻撃のコストを攻撃に要する時間・装置の価格・攻撃者の持つ専門知識・攻撃者の持つ攻撃対象の知識・の 4 項目で評価し，提案手法が従来のレーザー攻撃よりも低コストである事を明らかにする．

2 章ではデバイスにレーザーを照射するための機器について紹介する．レーザーフォールトに対する耐性を調査する目的で市販されているレーザー装置を初めに解説する．市販のレーザー装置は非常に精密な制御が可能であるが，非常に高価であり一部の攻撃者しか利用できないものである．本研究では独自にレーザー装置の部品を選定し，制御ソフトウェアを開発して安価なレーザーフォールト注入装置を構築する．構築したレーザー装置は市販品に劣らない精度を持ちながら価格を抑えることに成功しており，装置の価格という面で低コストなレーザー照射攻撃を可能にする．さらなる発展として，本研究ではダブルスポットのレーザー装置も構築する．レーザー照射は回路のごく一部にしか影響を与えないことが強みであるが，一方でそれが制限になる場合もあるため回路上の 2 点を同時に照射することができればより強力な攻撃が可能である．

3 章ではレーザーフォールト攻撃の対象となり得るデバイスの構成が解説される．レーザー攻撃は実装攻撃の一種であるから，攻撃の効果は攻撃対象デバイスの物理的な実装に依存する．本研究では特にソフトウェア実装の暗号を攻撃対象とするため，ソフトウェアを駆動するためのプロセッサのアーキテクチャが紹介される．また本研究ではペアリング暗号と呼ばれる先端暗号システム高速化のための専用コアを開発した．公開鍵暗号のようなリッチな暗号システムを利用する組込みプロセッサには，計算時間を削減する目的で専用回路という形のハードウェアアクセラレータが搭載される場合がある．このようなアクセラレータもレーザーフォールト攻撃の対象となり，またアクセラレータに直接フォールト攻撃せずとも，ROM にレーザーを照射することで命令改変攻撃が可能であることを示す．さらにレーザー攻撃を行うコストを決定する要素として，デバイスのパッケージングがある．普通 IC チップは何らかのパッケージに収められており外部に露出していないため，IC チップにレーザー照射を行うにはパッケージ開封によって IC チップを外部に露出させる必要がある．しかしながら近年のデバイスは容易にパッケージ開封可能であるものがあり，デバイスの加工という観点でレーザー攻撃のコストは低くなっていることを示す．

4 章では 2 章で紹介した装置を使って 3 章の攻撃対象を攻撃した際にどのような効果が得られるかを解説する．レーザーの照射場所として，多くの組込みプロセッサでプログラム格納領域として利用されている NOR フラッシュメモリに着目する．フラッシュメモリはチップ上の面積割合が大きく，またアレイ状の規則的な構造を持つことから，レイアウト情報がなくとも目視にて容易にその場所を特定することができる．さらに NOR フラッシュメモリは微細化の限界に近づいていることが知られており，将来にわたってレーザー攻撃の脅威に晒される恐れがある．組込みプロセッサ上のフラッシュメモリにはプロセッサが実行するプログラムが格納されているから，フラッシュメモリへのレーザー照射は実行される命令を“改変”することができる．従来の研究では命令改変のようなフォールトは精密な制御が必要であるため実現不可能であると結論づけられていたが，実験により安価なレーザー装置でも命令改変攻撃が可能であることを示す．また命令改変使った攻撃例として従来のソフトウェアベースフォールト攻撃対策が攻撃できることや，コプロセッサを使ったリッチな公開鍵暗号が攻撃できることを示す．また命令改変攻撃の難易度は攻撃者が攻撃対象のプログラムを知っているかに依存するため，レーザーを使って ROM に格納された

プログラムを抽出する攻撃を提案する．

5 章ではレーザー攻撃に対する対策手法を述べる．レーザー攻撃の対策としては，レーザー照射を検知する回路を追加するようなハードウェアによる対策が本質的だとされているが，コストがかかることや既存のデバイスには適用できないという問題がある．IoT 社会のエッジノードなどに利用される低コストデバイスにも導入できるような，ソフトウェア上で低コストなレーザーフォールト対策にニーズがあると考える．4 章で紹介した命令改変攻撃に対に適切な制約を与えることでソフトウェア上での対策を提案し，シミュレーションによる効果の検証を行う．

最後に 6 章を総括とし，本研究をまとめる．本論文では命令改変攻撃と呼ばれる強力なフォールト注入が，低コストなレーザー照射で実現できることを示し，さらに命令改変攻撃に対するソフトウェア上での対策を提案した．市販のレーザー攻撃評価用装置はなるべく精密なレーザー照射を行うことを目指しているため非常に高価だが，攻撃方法によってはそこまでの精密さが要求されずより低コストに実行可能な場合がある．適切なセキュリティ評価を行うには攻撃に必要なコストの下限を知ることが重要であり，本論文の成果はそれを達成する大きなステップである．

目次

第 1 章

序論

1.1　組込みデバイスにおける物理セキュリティの重要性

近年の著しい情報化社会の発展に伴い，我々の身の回りにある電子機器の数は増え続けている．身の回りの電子機器の数の参考としてのために，2018 年の総務省情報通信白書 [66] で示されている世界の IoT (Internet of Things) デバイス[*1]数の推移及び予測[*2]を参照する．これによれば世界の IoT デバイスは 2014 年の時点で 170 億台を超えており，2020 年には 400 億台に到達する見込みである．このような急激な電子機器の増加に伴い，セキュアにすべき情報資産も増加していくことが予想される．電子機器の数が増えると言うことはそれだけ攻撃者の攻撃対象になり得るデバイスが増えるということであり，これまで以上に攻撃に対するセキュリティ対策を考慮する必要がある．スマートフォン内の個人情報、クレジットカード内の暗証番号や通信機機内のネットワーク暗号鍵等は攻撃対象になり得る情報の代表である．これらのデバイスは “組込みデバイス” と呼ばれ比較的小さな物体である．サーバーや個人 PC のような比較的大きなデバイスと異なり組込みデバイスは攻撃者の手に触れる機会が多く，時には攻撃者がそのデバイスを持ち去って攻撃を行うような状況が考えられる．例えばクレジットカードを店員に渡す際や，スマートフォンを紛失した際である．このような状況で攻撃者は，そのデバイスの物理的特徴に着目した “実装攻撃” と呼ばれる攻撃を行うことができる．

実装攻撃とは，デバイスに実装された機能の実装方法に依存する脆弱性を利用した攻撃を指す．暗号技術やそれを利用した通信プロトコル等のセキュリティ機能の多くは，論理的なレイヤーで，ある仮定を満たすという条件の下でセキュリティが保証されている．しかしながらその機能がデバイス上に実装された場合，論理的なレイヤーでは考慮されない情報漏洩や攻撃注入の経路が発現する．このような経路は “サイドチャネル” と呼ばれる．攻撃に利用できる程度のサイドチャネルが存在するかどうかはその機能のソフトウェア上の実装方法やデバイスの物理的な実装方法に依存しており，論理的なレイヤーでは検証が難しいためそれぞれの実装に対して脆弱性を検証することが求められる．

IoT 時代における実装攻撃シナリオの例を図 1.1 に示す．この図は IoT アプリケーションの代表例とされるセンサネットワークから情報がインターネットクラウド上へ吸い上げられる様子を示している．センサネットワークがセンスするデータは温度や湿度などのデータであり，それ自体には高い秘匿性が要求されない場合があるが，それをビッグデータとして扱う場合にはそのデータの “質” が求められる．よって少なくとも，そのデータが正しいノードから送信されたものなのか，改ざんされていないかといった事を保証するための暗号技術の利用が望まれる．センサネットワークに利用されるノードは非常に数が多いた

[*1] 固有の IP アドレスを持ちインターネットに接続が可能な機器及びセンサーネットワークの末端として使われる端末等
[*2] （出典）IHS Technology

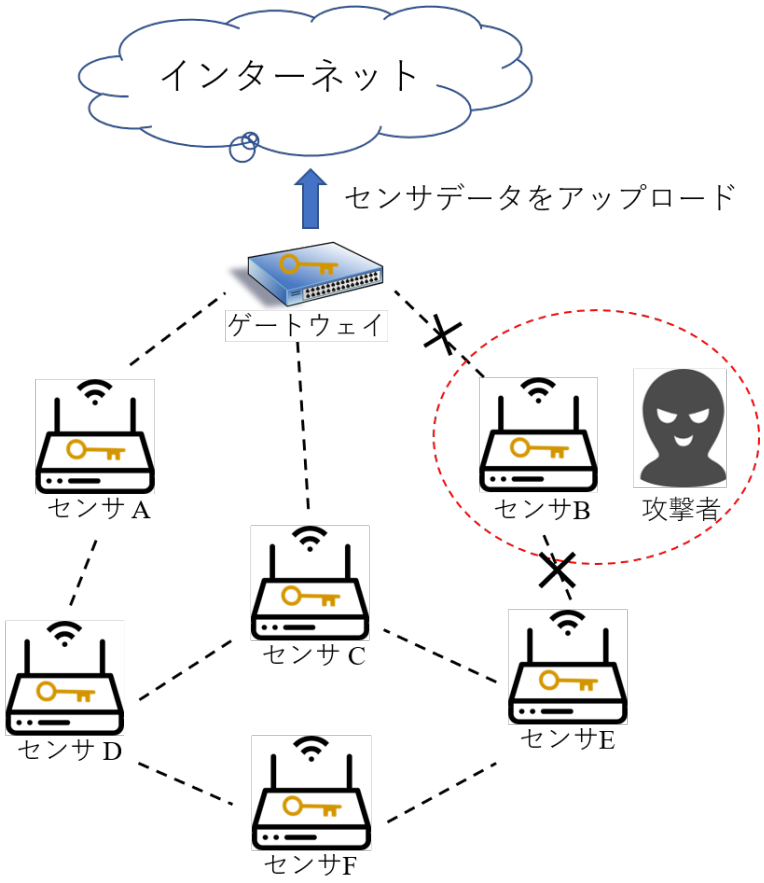


図 1.1: IoT 時代における実装攻撃シナリオの例．センサネットワークが共通鍵を使って暗号通信を行っているような場合，実装攻撃はネットワーク全体に対する重大な脅威になる．

め，各ノードにかけられるコストは低くリッチな公開鍵暗号や耐タンパー性を実装できない場合がある．このような場合には共通鍵暗号方式を使ってネットワーク上のデータを守るという方法が現実的であり，ネットワーク内のすべてのノードで共通のグローバルマスターキーを扱う．あるノードからのグローバルマスターキー流出がネットワーク全体のリスクに繋がるため，このような運用にはノードのコストを抑えつつも高いセキュリティが求められる．またメッシュタイプのような冗長性の高いトポロジでネットワークが構築されていた場合，攻撃者が一つのデバイスを持ち去ったとしてもネットワークは機能を維持することができ，攻撃者の介在に気づけないかもしれない[*3]．現実世界では運用上の理由で上記のようなグローバルマスターキーを使ったネットワークがセンサネットワーク以外にも多数存在すると考えられる．そのような場合には，一つのノードよりもネットワーク全体のリスクを考慮して耐タンパー性を実装することが重要となる．

[*3] 山間部などの保守点検が容易ではない場所のネットワークの場合，故障だと判断されたノードが再設置されるまでには相当のタイムラグがあると考えられる．

1.2　実装攻撃

1.2.1　実装攻撃の概要

AES や RSA 暗号のように，理論上の重大な脆弱性が見つかっておらず数十年以上運用されている暗号システムはセキュアだと信じられている．セキュアだと信じられている暗号システムの多くは最良の攻撃を行ったとしても 2^{100} 以上の計算量を解読に要するため，現代の計算機では現実的な時間内に解読することができない．このような暗号は “計算量的に安全な暗号” と呼ばれ，実用化されている暗号の多くはこのクラスに属する．暗号の安全性に関するもう一つの重要なクラスが，“情報理論的に安全” な暗号である．情報理論的に安全な暗号は無限の計算量を持つ攻撃者に対してもセキュアであることが証明された暗号であり，実行速度等の面で課題が残っているが，高いセキュリティを要求するアプリケーションでは実用化が期待されている．いずれにしても，世の中で実用化されている暗号システムは計算量的/情報理論的に安全性が証明されたものであり，ある仮定の元では攻撃が困難であると信じられている．

上記で述べた様に，実用化されている暗号システムを正攻法で（安全性証明の仮定を満たす状態で）攻撃することは困難である．そこで攻撃者にとって，暗号システムの実装上の脆弱性を突く攻撃が有効な手段となる．図 1.2 にそのような攻撃の例を示す．ユーザー（攻撃者）がクレジットカードの 4 桁の暗証番号を探索したいとする．クレジットカードは入力された暗証番号を認証し，Accept/Reject の応答を返すものとする．ここでクレジットカードは暗証番号認証を 1 桁ごとに行うよう実装されている．すると暗証番号入力から応答までの時間が正解桁数に応じて変化するから，攻撃者はこの時間情報を利用することで暗証番号を 1 桁ずつ当てていくことが可能である．結果として本来の暗証番号空間 10^4 が 4×10 に削減されたことになる．このような攻撃（“サイドチャネル攻撃” と呼ばれる）は暗号システムにも適用可能である事が知られており，実用の暗号システムに対する重大な脅威とみなされている．上記の例は実行時間という情報を用いて攻撃を行っているが，サイドチャネル攻撃に利用される情報（サイドチャネル情報

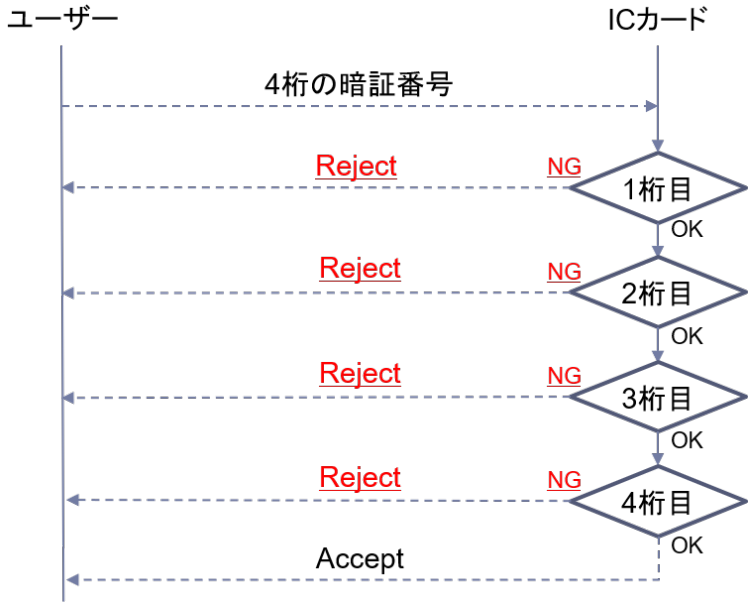


図 1.2: 実装上の脆弱性の例．応答までの時間という情報を利用することで 10^4 個の暗証番号が 4×10 個に絞り込まれる．

と呼ばれる）は他にも，消費電力，放射電磁波，音響ノイズなどがある．

また実装攻撃のもう一つの重要なクラスとして “フォールト攻撃” が知られている．サイドチャネル攻撃における攻撃者はサイドチャネル情報を静的に観測するだけであった一方、フォールト攻撃における攻撃者は暗号システムに対してレーザーや電磁波を照射するなどと動的に働きかける．

1.2.2　サイドチャネル攻撃（SCA）

サイドチャネル攻撃は静的な実装攻撃であり，1999 年に Kocher らによって初めて導入された手法である [30]．暗号アルゴリズムはふつう，十分な量の平文と暗号文の組が利用できるとして，暗号化に使われた秘密鍵が解読できないように設計されている．しかしながらひとたび暗号アルゴリズムが物理的に実装されると，その暗号はサイドチャネル情報と呼ばれる情報を漏えいさせつつ動作することになる．サイドチャネル情報としては，暗号デバイス内で処理中の値と相関のある物理量が利用され，例えば消費電力，電磁波，音波などがある．攻撃者は，本来なら得られるはずのない（アルゴリズムの）中間値をサイドチャネル情報から推定することで，秘密鍵を容易に計算することが可能となる．

図 1.3 にサイドチャネル情報とサイドチャネル攻撃の関係を示す．図中ではあるハードウェア上に暗号処理が実装されており，暗号処理は入力された平文を秘密情報に従って暗号化して出力する．古典的な暗号解読では，攻撃者は平文と暗号文の組（メインチャネル情報）から秘密情報を特定できるかが議論されてきた．しかしながら暗号処理がハードウェア上で行われる以上、暗号処理の挙動はハードウェアの物理的なふるまい，すなわち消費電力や放射電磁波などのサイドチャネル情報を変化させる．サイドチャネル情報には秘密情報と相関のある情報が含まれている恐れがあり，攻撃者はメインチャネルの情報以外にサイドチャネル情報を使うことで古典的暗号解読よりも効率的に秘密情報を取得できる可能性がある．

1.2.3　フォールト攻撃（FA）

実装攻撃の重要なクラスであるフォールト攻撃 [9, 60] は，機器の動作や計算の途中値に注入した意図的なエラー（フォールト）によって機器内の秘密情報を取得する攻撃である．機器に対するフォールトはクロックグリッチ [8, 24, 59]・電磁波照射 [37, 45, 36]・レーザー照射 [54, 46]・ボディバイアシング [11] 等様々な手法によって注入される．フォールト攻撃は機器の動作を予期できないものに変えてしまうか

図 1.3: サイドチャネル攻撃の概要図．暗号処理を行うハードウェアからは平文・暗号文の他に消費電力や電磁波などのサイドチャネル情報が漏洩しており，サイドチャネル情報を観測することで攻撃者は秘密情報を効率的に解読できる可能性がある．

図 1.4: フォールト攻撃の概要図．攻撃者は暗号処理が実装されたハードウェアに意図的な計算誤り（フォールト）を注入することで誤り暗号文を取得し，誤り暗号文から秘密情報を特定できる．

ら，うまくフォールトを注入することができれば非常に強力な攻撃が可能になる．例えば，出力データのアドレスを誤らせることで正規の出力ポートから秘密情報を出力させる攻撃が報告されている他，暗号文の計算パスに誤りを注入し誤りを含む暗号文と正常な暗号文の組から秘密鍵を推測する Differential Fault Analysis (DFA) が多くの暗号アルゴリズムに対して提案されており [26, 14, 12]，非常に強力な攻撃として知られている．

図 1.4 にフォールト攻撃（特に DFA）の概要図を示す．図中ではあるハードウェア上に暗号処理が実装されており，暗号処理は入力された平文を秘密情報に従って暗号化して出力する．暗号処理はあらかじめ決められた手順に従って暗号化処理を行うが，攻撃者がハードウェアに何らかの手法で意図的な計算誤りを注入することで処理内容が変化し，結果として誤りを含む暗号文を出力する場合がある．誤り暗号文は正規の暗号アルゴリズムが出力すべき値とかけ離れているため，秘密鍵の情報を含んでいる場合がある．このような誤り暗号文と平文の組を多数収集し，それらの関係から秘密鍵を推測するのが DFA である．

1.2.4　複合攻撃

SCA と FA を組み合わせたより巧妙かつ強力な攻撃である複合攻撃が報告されている．複合攻撃は 2 つのクラス，フォールトを利用して SPA を可能にするもの [2] と，フォールト注入手法（特にレーザー照射）によって DPA の効率を上げるもの [22] に分類できる．レーザー照射は照射位置の電流量を増加させるから，回路全体のうち一部の消費電力を選択的に際だたせることができる．Di-Battista ら [22] は FPGA 上に実装した DES の S-Box にレーザーを照射することで，レーザー照射しない時と比べて半分の波形数で DPA による秘密鍵取得に成功している．ここで特徴的なのが，この手法はレーザー照射によって論理的なエラーを注入しないという点である．これは光センサを追加する等コストのかかる方法でしか検出できないため，論理的なエラーを注入する手法と比べて対策が困難である．

1.3　物理セキュリティの評価方法

デバイスやシステムのセキュリティを評価・認証する制度の一つに CC(Common Criteria for Information Technology Security Evaluation)[62, 63, 64] がある．CC は ISO/IEC 15408 で標準化されており，各国の政府調達基準などに利用される重要な制度である．CC によって保証されるものは，評価対象がある水準のセキュリティを持っているかどうかではないことに注意が必要である．CC では行われた

評価のレベル（EAL, Evaluation Assurance Level）が保証される．7 段階の EAL が存在するが，これは評価機関によって行われた評価の厳密さや量を示すものであり，高い EAL が安全性の高さを直接保証するわけではない．日本では IPA（情報処理推進機構）が EAL の認証を行っているが，異なる評価機関が行った場合でも評価結果を均質にするため，EAL ごとにどのような評価を実施し，実施結果をどのようにスコア付けすべきかということが CEM（Common Methodology for Information Technology Security Evaluation）[65] にまとめられている．CEM も ISO/IEC 18045 として標準化されており，国際的に認められた規格である．CEM に記載された評価方法のほとんどはデバイスの仕様書がセキュリティ要件を満たしているかどうかを検査するためのものであり，デバイスの実際のセキュリティ強度を測るものではないが，唯一の例外が AVA_VAN という項目である．AVA_VAN は評価対象（TOE, Target of evaluation）に対して実際に脆弱性検証を行うことを求めており，AVA_VAN の項目を満たすためには評価機関が TOE に対して実際に攻撃を行い，攻撃成功までのコストが十分大きいことが求められる．

AVA_VAN の中でも特に実装攻撃の評価方法については CC サポート必須技術文書の一つである AAPS[34] に記載されている．AAPS では攻撃のコストをどのように算定するかが記述されており，攻撃成功に要する 6 項目

- 所用時間：攻撃成功までに要した時間
- 使用機器：攻撃に使用する機器
- TOE アクセス：攻撃成功までに必要な TOE の数.
- 専門知識：攻撃者の持つ暗号やハードウェア，オシロスコープやレーザー装置等の機器に関する知識
- TOE 知識：攻撃成功に要する TOE 知識．公開されたマニュアルで十分なのか，ソースコード・回路設計情報が必要かなど
- オープンサンプル：識別フェーズで利用できる TOE のサンプルがどの程度公開されているか

の合計点が攻撃のコスト（高い方が攻撃困難）となる．なお各項目は識別フェーズ（どのようにすれば攻撃できるかのパスを考え，そして攻撃できるまで）・悪用フェーズ（識別フェーズで構築したパスに従って攻撃できるまで）に分かれている．高い EAL には攻撃コストが高い攻撃に対してもセキュアであることが求められ，例えば EAL6 には攻撃コスト 31 以上，EAL3 なら攻撃コスト 16 以上が求められる．脆弱性検証は既知の攻撃方法を使って行われるが，具体的にどの攻撃方法を評価するというのは公開されておらず，またもちろん未知の攻撃法に対するセキュリティは検証できない．

1.4　これまでのフォールト攻撃に関する研究と本研究の目的

本論文では実装攻撃の中でも特にフォールト攻撃に着目するため，フォールト攻撃の先行研究を掘り下げる．フォールト攻撃研究の目的は，攻撃者がどのような攻撃が可能かを事前に知ることでそれに対する対策を講じるためである．フォールト攻撃はデバイスの動作に誤りを注入するものであるため強力なフォールト注入が可能な攻撃者はデバイスに実装された対策技術そのものの動作を誤らせることも可能であり，対策には十分な検討を要する．対策の検討とは，デバイスが守るべき資産の価値を考慮し，それに応じた攻撃者を適切に考慮しなければならないということである．本来ならすべてのデバイスに強力な対策を実装することが望ましいが，強力な対策にはそれに応じたコストがかかってくるため，想定する攻撃者に対して適切な対策をとるということが IoT のような低コストのデバイスを守るために望まれる．

フォールト注入されるレベル	フォールトモデル	有効な対策
アルゴリズム	Ifスキップ	検算
命令	命令スキップ	命令多重化
レジスタ	セット/リセット/フリップ	開封検知やレーザーセンサ
トランジスタ	ON/OFF/フリップ	

図 1.5: フォールト注入の段階とフォールトモデルに応じた対策の関係

1.4.1　フォールトモデル

フォールト攻撃に晒されるデバイスの挙動は信用できないため，フォールト攻撃対策を考える上で何らかの信頼の起点を仮定する必要がある．この目的のため，攻撃者の注入できるフォールトに何らかの制限を課して抽象化した “フォールトモデル”[56] が定義される．想定する攻撃者に応じて適切なフォールトモデルを考えることにより，そのフォールトモデルの範囲内の攻撃者に対してはセキュアな対策を講じることができる．様々なレベルのフォールトモデルを定義することが可能であるが，強力なフォールトモデルを対象とした対策には高いコストがかかることから，求められるセキュリティレベルに応じたフォールトモデルの設定が重要である．代表的なフォールトモデル及びフォールト注入の段階，またフォールトモデルに応じた対策の関係を図 1.5 に示す．ほとんどの論理回路が CMOS ロジックで製造されているため，トランジスタレベルが最も抽象度の低いフォールト注入の段階となる．もし任意のトランジスタに自由にフォールトを注入することができるならば，論理的に対策するのはほぼ不可能だと考えられる．トランジスタレベルのフォールトモデルとしては，トランジスタの状態を ON や OFF にできるものや，フリップさせるものがある．トランジスタレベルのフォールトには，レーザー照射やプロービングといった数千万円オーダーの装置による精密な注入が求められる．また攻撃者が攻撃したいトランジスタがどこにあるかを知っていなければならいという意味でも，攻撃にかかるコストは高いといえる．トランジスタレベルの上位のフォールトにレジスタレベルのフォールトがあるが，これも高価な装置を要する．また回路上から攻撃したいレジスタの場所を知るのもコストがかかる．より上位のフォールトレベルが命令レベルやアルゴリズムレベルであり，命令スキップや if スキップのようなフォールトはクロックグリッチのような安価な装置で注入可能なことが知られている．命令スキップモデルにおける攻撃者は，実行中のアセンブリ命令をスキップ（実行させなく）することができるとモデル化される．命令スキップモデルのフォールトが実際に生じることが多くのフォールト注入手法 [8, 59, 37, 45, 54]，対象デバイスで報告されており，命令スキップモデルは現実的な攻撃者のモデルとして活発に研究されている．また命令スキップによって DFA による導出が可能であることも確認されており [59, 31, 20, 48]，フォールト攻撃の脅威にさらされる暗号デバイスには命令スキップへの耐性が求められる．いくつかの考察から，命令スキップを連続した命令に注入することは難しいと考えられており，この仮定を利用したソフトウェアによる対策が提案されている [38, 10].

フォールトモデル		攻撃コスト			
		所要時間	専門知識	使用機器	TOE知識
クロックグリッチによる	Ifスキップ	低	高	低	低
	命令スキップ				
レーザー照射による	レジスタ セット/リセット/フリップ	高	高	高	高
	トランジスタ ON/OFF/フリップ				

図 1.6: フォールトモデルごとの相対的な攻撃コスト

1.4.2　従来のフォールトモデルごとの攻撃コスト

図 1.5 で示したフォールトモデルについて，フォールトモデルごとの攻撃コストを AAPS で規定される項目に従って見積もる．AAPS による攻撃コストの算定は本来ならある TOE が特定されて上で行われるが，ここでは TOE を特定せずある同一のデバイスにそのフォールトモデルを注入した際の相対的な攻撃コストを考える．同一のデバイスを考えるためオープンサンプル・TOE アクセスの 2 項目には相対的な差がないと考え検討しない．クロックグリッチによる命令スキップと，レーザー照射によるトランジスタフリップ間の攻撃コストを比較した結果を図 1.6 に示す．まず攻撃までの所要時間に関してはトランジスタフリップモデルによる攻撃の方が成功まで長くかかると考えられる．これはクロックグリッチのパラメータがグリッチタイミングとグリッチ幅の二つなのに対し，レーザー照射はそれに加え照射位置が加わるためである．続いて専門知識については，どちらも暗号やオシロスコープの扱いなどで高い水準が必要になると考えられる．レーザー照射には光学の知識も必要になる可能性があるが，それほどの差はないと考える．一方使用機器については大きな差がある．クロックグリッチは数十万円程度の機器で注入可能だが，トランジスタを狙うほどに高精度なレーザー照射には数千万円の装置が必要である．TOE の知識に関しても，レーザー照射のほうが高い水準が必要になると考えられる．クロックグリッチには照射場所というパラメータが存在しないが，レーザー照射の場合攻撃者が照射したいトランジスタやレジスタがどこに存在するのかという情報が必要になる場合があり，回路レイアウトなどの機密情報が必要になる．

1.4.3　本研究の目的

図 1.6 からわかるとおり，これまでのフォールトモデルにおける攻撃コストには大きなギャップが存在する．もしこの間に未知の攻撃方法が存在し，それが低コストで実行できるならば，攻撃のコストが高すぎるため安全だとみなされていたデバイスに対する大きな脅威となる可能性がある．本論文では図 1.7 に示すように，実装攻撃の中でも特に強力なレーザーフォールト攻撃について，より低コストな機器によって命令スキップよりも強力な “命令改変” と呼ばれるフォールトが注入できることを示す．本論文の寄与は主に次の 3 つである．

1. デバイス上のフラッシュメモリにレーザーを照射することで命令改変と呼ばれる新たな攻撃が可能である事．

フォールトモデル		攻撃コスト			
		所要時間	専門知識	使用機器	TOE知識
クロックグリッチによる	Ifスキップ	低	高	低	低
	命令スキップ				
低コストレーザー照射による	命令改変	中	高	中	低
レーザー照射による	レジスタ セット/リセット/フリップ	高	高	高	高
	トランジスタ ON/OFF/フリップ				

図 1.7: 本論文で明らかにすること．低コストなレーザー装置を使って，これまで未知の攻撃手法が実行可能である事を示し，またそれに対するソフトウェアによる対策を提案する．

2. 命令改変攻撃が従来のレーザー攻撃と比べて低コストに実行できること
3. 命令改変攻撃を行う攻撃者の能力を適切にモデル化する事でソフトウェアによる対策が可能であること．

命令改変モデルにおける攻撃者は，実行されるアセンブリ命令を別の命令へ置き換えることができるとモデル化される．NOP 命令への改変は命令スキップとみなせる [38] ため，命令スキップモデルは命令改変モデルのサブセットとみなせる．多くのデバイスとフォールト注入手法で命令改変が生じることが報告されているが，制御が難しく攻撃に利用できないと考えられていたことから，これまであまり研究されてこなかった [8, 37, 31]．一方最新の研究 [18][Pub. 1] では，レーザー照射のような精密なフォールト注入手法によって命令改変を制御できることが示されている．ここではチップ上のフラッシュメモリにレーザーが照射されており，照射する位置によってフラッシュから読み出されるアセンブリ命令の特定の 1 ビットに誤りが注入される．またモノスポットレーザーでは 1 ビットの誤りしか引き起こせないことや，ビットセット片方向のみの誤りであるという制限が述べられているが，この制限下でも命令改変は非常に強力である．

さらに本論文では命令改変フォールトにレーザースポット数という制約を加えることで，ソフトウェアによう低コストな対策が可能であることを示す．提案対策は従来の命令スキップに対する対策と比べても大きくないコストで実装可能であり，この対策を実装することでこれまで命令スキップまでしか対策できなかった対象をよりセキュアにすることが可能である．

1.5　本論文の構成

本論文ではどれだけのコストで，どういう対象に対して，どういったフォールト注入が可能なのかという 3 つの観点を明らかにする．本論文の全体像を図 1.8 に示す．1 章ではフォールト攻撃とそれを取り巻く関連研究を紹介した．IoT 社会のように大量のデバイスが身の回りにある環境においては，攻撃者のデバイスに対する物理的なアクセスが容易になり，デバイスが実装攻撃の脅威に晒される機会が増加する事を述べた．本論文では実装攻撃の中でもフォールト攻撃，特にレーザー照射によるものに着目し，従来よりも低コストで強力な攻撃が達成できる可能性を述べた．

2 章ではデバイスにレーザーを照射するための機器について紹介する．レーザーフォールトに対する耐

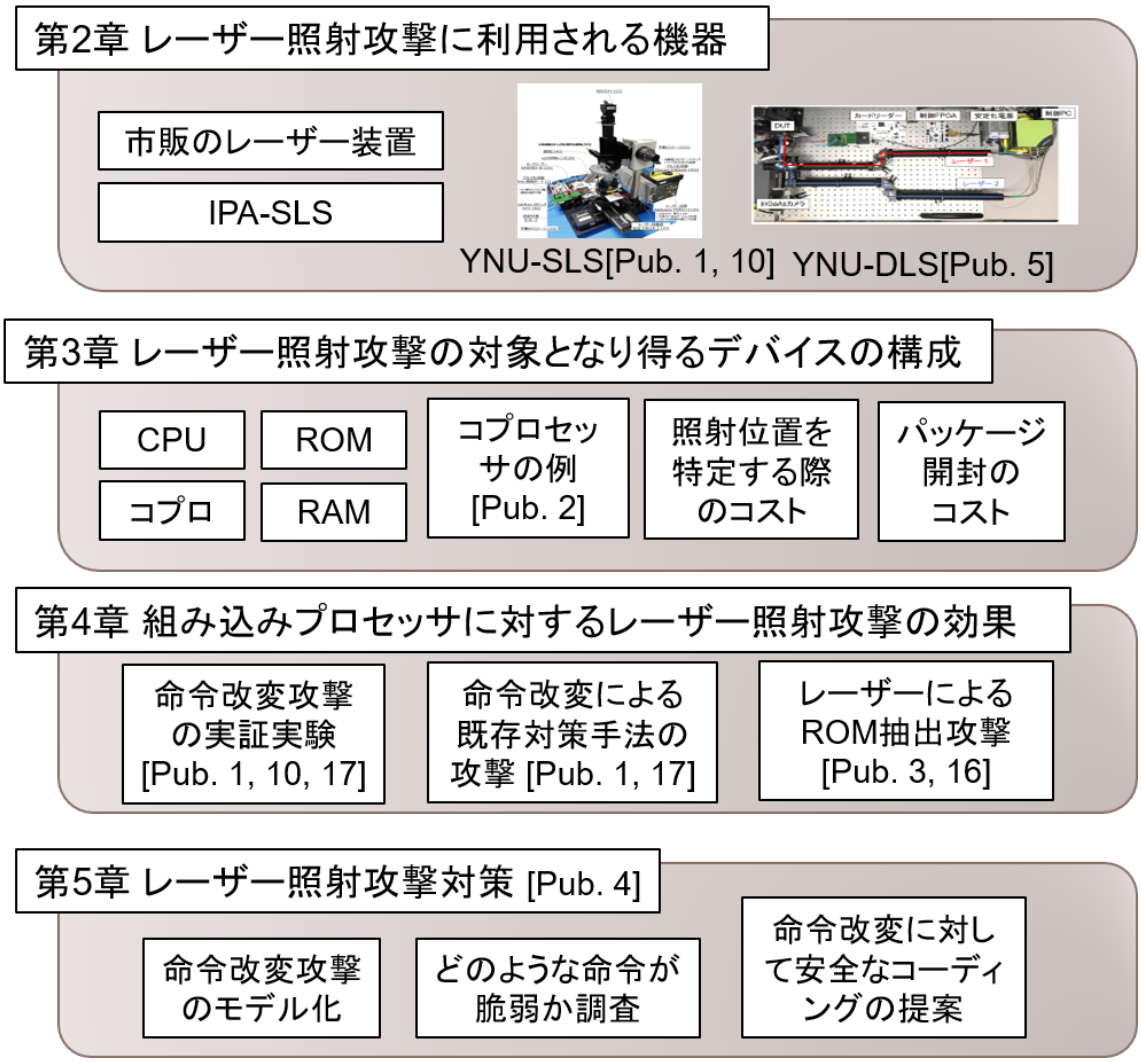


図 1.8: 本研究の全体像と本論文の構成

性を調査する目的で市販されているレーザー装置を初めに解説する．市販のレーザー装置は非常に精密な制御が可能であるが，非常に高価であり一部の攻撃者しか利用できないものである．本研究では独自にレーザー装置の部品を選定し，制御ソフトウェアを開発して安価なレーザーフォールト注入装置を構築する．構築したレーザー装置は市販品に劣らない精度を持ちながら価格を抑えることに成功しており，装置の価格という面で低コストなレーザー照射攻撃を可能にする．さらなる発展として，本研究ではダブルスポットのレーザー装置も構築する．レーザー照射は回路のごく一部にしか影響を与えないことが強みであるが，一方でそれが制限になる場合もあるため回路上の 2 点を同時に照射することができればより強力な攻撃が可能である．

3 章ではレーザーフォールト攻撃の対象となり得るデバイスの構成が解説される．レーザー攻撃は実装攻撃の一種であるから，攻撃の効果は攻撃対象デバイスの物理的な実装に依存する．本研究では特にソフトウェア実装の暗号を攻撃対象とするため，ソフトウェアを駆動するためのプロセッサのアーキテクチャが紹介される．また本研究ではペアリング暗号と呼ばれる先端暗号システム高速化のための専用コアを開発する．公開鍵暗号のようなリッチな暗号システムを利用する組込みプロセッサには，計算時間を削減する目的で専用回路という形のハードウェアアクセラレータが搭載される場合がある．このようなアクセ

ラレータもレーザーフォールト攻撃の対象となり，またアクセラレータに直接フォールト攻撃せずとも，ROM にレーザーを照射することで命令改変攻撃が可能であることを示す．さらにレーザー攻撃を行うコストを決定する要素として，デバイスのパッケージングがある．普通 IC チップは何らかのパッケージに収められており外部に露出していないため，IC チップにレーザー照射を行うにはパッケージ開封によって IC チップを外部に露出させる必要がある．しかしながら近年のデバイスは容易にパッケージ開封可能であるものがあり，デバイスの加工という観点でレーザー攻撃のコストは低くなっていることを示す．

4 章では 2 章で紹介した装置を使って 3 章の攻撃対象を攻撃した際にどのような効果が得られるかを解説する．レーザーの照射場所として，多くの組込みプロセッサでプログラム格納領域として利用されている NOR フラッシュメモリに着目する．フラッシュメモリはチップ上の面積割合が大きく，またアレイ状の規則的な構造を持つことから，レイアウト情報がなくとも目視にて容易にその場所を特定することができる．さらに NOR フラッシュメモリは微細化の限界に近づいていることが知られており，将来にわたってレーザー攻撃の脅威に晒される恐れがある．組込みプロセッサ上のフラッシュメモリにはプロセッサが実行するプログラムが格納されているから，フラッシュメモリへのレーザー照射は実行される命令を“改変”することができる．従来の研究では命令改変のようなフォールトは精密な制御が必要であるため実現不可能であると結論づけられていたが，実験により安価なレーザー装置でも命令改変攻撃が可能であることを示す．また命令改変使った攻撃例として従来のソフトウェアベースフォールト攻撃対策が攻撃できることや，コプロセッサを使ったリッチな公開鍵暗号が攻撃できることを示す．

5 章ではレーザー攻撃に対する対策手法を述べる．レーザー攻撃の対策としては，レーザー照射を検知する回路を追加するようなハードウェアによる対策が本質的だとされているが，コストがかかることや既存のデバイスには適用できないという問題がある．IoT 社会のエッジノードなどに利用される低コストデバイスにも導入できるような，ソフトウェア上で低コストなレーザーフォールト対策にニーズがあると考える．4 章で紹介した命令改変攻撃に対に適切な制約を与えることでソフトウェア上での対策を提案し，シミュレーションによる効果の検証を行う．

最後に 6 章を総括とし，本研究をまとめる．本論文では命令改変攻撃と呼ばれる強力なフォールト注入が，低コストなレーザー照射で実現できることを示し，さらに命令改変攻撃に対するソフトウェア上での対策を提案した．市販のレーザー攻撃評価用装置はなるべく精密なレーザー照射を行うことを目指しているため非常に高価だが，攻撃方法によってはそこまでの精密さが要求されずより低コストに実行可能な場合がある．適切なセキュリティ評価を行うには攻撃に必要なコストの下限を知ることが重要であり，本論文の成果はそれを達成する大きなステップである．

第 2 章

レーザー照射攻撃に利用する装置の低コスト構築

本章では，レーザーフォールト注入に利用される装置について解説する．ある攻撃が現実的な脅威かどうかを評価するために，その攻撃にかかるコストを算定することが重要であり，フォールト攻撃の場合攻撃に利用する装置の価格は攻撃コストの大きな部分を占める要素である．一般的にレーザーフォールト攻撃は高価な装置が必要な一方で強力な攻撃が可能だとされており，攻撃による漏洩情報の価値が低いようなデバイスはレーザーフォールト攻撃に晒される可能性を考慮しない場合が多い．しかしながら，安価な装置でレーザーフォールト攻撃（ないし光学的なフォールト攻撃）を行う研究 [27, 50] も存在しており，レーザーフォールト攻撃に対する体系的な知見を得るためには，どれくらいの価格の装置でどれだけの攻撃を行うことができるのかを評価する必要がある．文献 [42] にて，レーザーフォールト関連研究でどのような装置が利用され，どの対象に対してどのようなフォールトが得られたかがまとめてあるが，レーザー装置の価格というところまでは言及されていない．本研究では独自にレーザー装置を構築することで，どれくらい安価にレーザー注入が可能かを評価する．

2.1　市販のレーザーフォールト評価用装置

設計したデバイスのレーザーフォールト耐性を評価する目的で，いくつかの企業からレーザーフォールト装置が販売されている．文献 [41] で示されているように，市販のレーザー装置で有名なものには Riscure 社の Laser Station2（図 2.1）や，Alphanov 社の Single Laser Microscope Station (S-LMS)（図 2.2）がある．各社とも価格をオープンにしていないため正確な価格を記すことはできないが，文献 [16] で用いられたスタンダードなレーザー装置は 150k ユーロ ＝ 約 1800 万円と述べられている．確かに，レーザー攻撃を行うのに 1800 万円の装置が必要ともなれば，ほとんどの状況においてレーザー攻撃は現実的な脅威とはならない．しかしながら市販のレーザー装置には保守費用やサポートなどのソフト的な価格が含まれていると考えられ，攻撃装置のハードウェアだけに着目すればより安価にレーザー攻撃を行うことができる可能性がある．

2.2　IPA におけるレーザーフォールト評価用装置

4.4 章では，情報処理推進機構（IPA）にあるレーザー装置をお借りして実験を行っている．IPA の装置の一部には特別仕様の機器が含まれており，公開できない部分があるがここではできる限りの説明を

図 2.1: Riscure 社 Laser Station2[44]

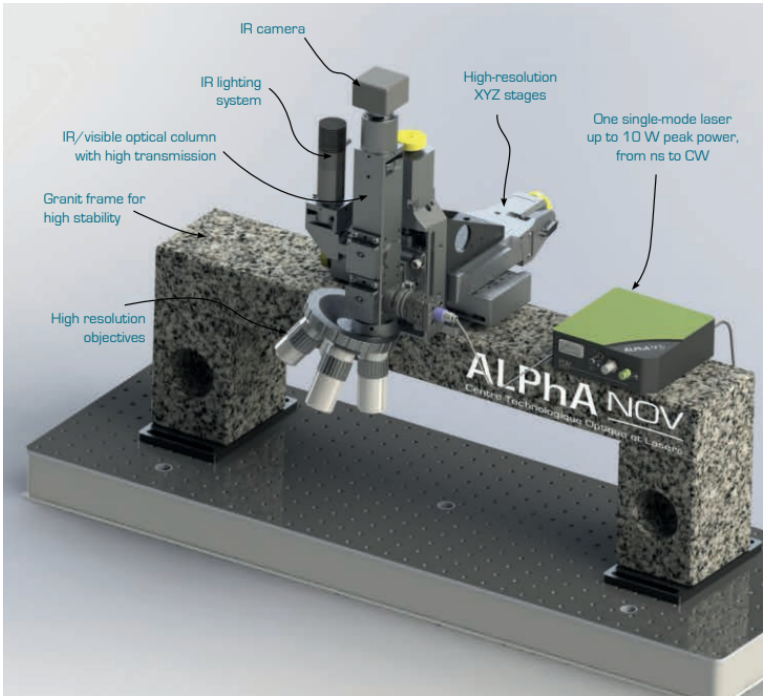


図 2.2: Alphanov 社 Single Laser Microscope Station (S-LMS)[1]

行う．

IPA におけるレーザー装置の構成図を図 2.3 に示す．このセットアップを用いてレーザー照射の 4 つのパラメータと DUT の制御を行うことができる．実験に用いたレーザーは裏面のシリコン基板を通過するために波長 976nm のものを用いている．レーザーは安定電源から最大で 1.7 W を供給されており，これはレーザー強度を制御している．レーザー照射の時間と時刻は FPGA で生成されるトリガパルスによって制御される．さらにレーザーは XYZ ステージにマウントされており，1 µm の精度で照射位置を制御可能である．また DUT のグラウンドラインには 10 Ωのシャント抵抗が挿入されており，アクティブプローブを使って消費電力を記録している．測定は 1G Sample/s で行われ，フィルタ等は入っていない．最後に，我々のセットアップは光学系を通して照射の様子を画像として得ることができるが，我々は裏面照射で実験を行うため，可視光画像では照射位置の情報を得られない．そこで我々はハロゲン光源と可視光フィルタを使って，赤外画像から照射位置の情報を得る．このレーザー装置の価格は市販品と同程度と仮定し，約 1800 万円程度と予想される．以下ではこのレーザー装置のことを，IPA Single-Spot Laser Station(IPA-SLS) と呼ぶ．

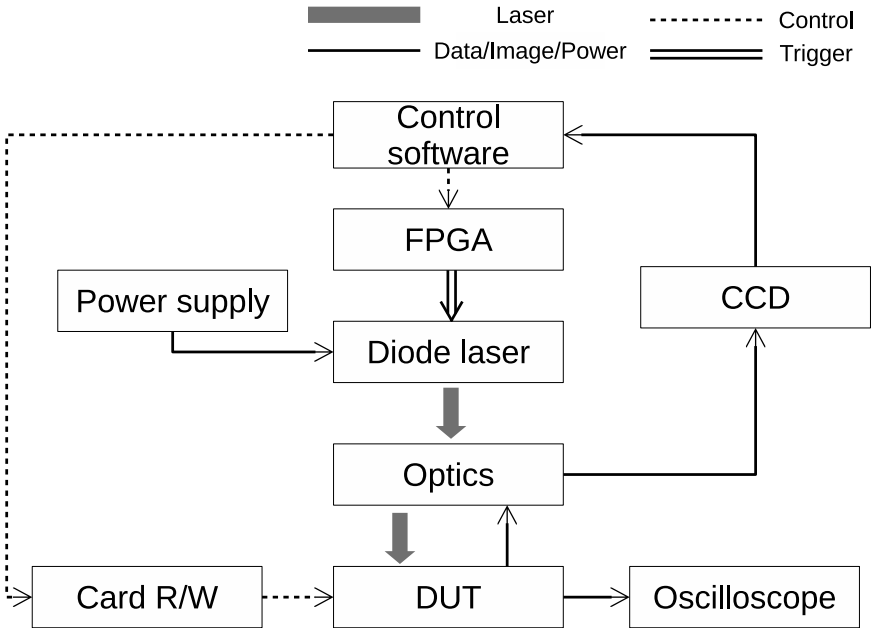


図 2.3: レーザー照射環境

2.3　シングルスポットレーザー照射装置の構築

IPS-SLS の構成を参考にして，独自の裏面レーザー照射用装置を図 2.4 のように構築した．裏面照射は IC チップのシリコンを透過する必要があるため，赤外レーザー (Alphanov PDM+, 1064 nm) がレーザー光源として選ばれた．レーザーはシンプルな光学系に取り付けられており，20 倍の対物レンズ (SIGMA KOKI PAL-20-NIR-LC00) を通して DUT に照射される．対物レンズを通った後のレーザースポットは 30–40 µm と見積もられている．これは光学限界である 1 µm よりもかなり大きく, より低価格のレーザー光源や光学系でも達成可能だと考えられる．光学系は特別に設計された赤外リング照明（波長 1060 nm）を含んでおり，また近赤外領域に感度を持つ NIR カメラを利用することで，チップ裏面からシリコンを透過して配線層の回路レイアウトを観察しながらレーザー照射を行うことができる．レイア

ウト観察しながらの照射はレーザー攻撃のために必須ではないが，攻撃者がどこにレーザーを照射すればよいかを容易に決定することができるようになる．

このレーザー装置は 4 つのレーザーパラメータである照射位置，照射強度，照射時刻，照射時間を制御できる．DUT は XY ステージにマウントされており，ステージを手動で動かすことによってレーザー照射位置を制御する．安定化電源 (松定 P4K18-2) はレーザー照射強度の制御を担っている．またパルス信号がトリガ生成器で生成され，レーザー照射時刻及び照射時間を制御する．パルス幅とパルス生成タイミングがレーザー照射時間と照射時刻にそれぞれ対応しており，パルスのパラメータは PC 上から制御できる．トリガ生成器には DUT（Device Under Test）の消費電力のパターンをマッチングして信号を生成する機器 (Riscure icWaves[43]) が含まれており，DUT から直接照射タイミングのトリガがとれない場合でも消費電力は計からトリガが生成できる．このようなパターンマッチング機器はより現実的な攻撃に使われるものである．

レーザー装置の構成に利用した機器のリストを表 2.1 に示す．機器の合計価格は 1154 万円となり，市販装置の価格の 6 割強の価格でレーザー装置を構築できた．実際には，オシロスコープ及び icWaves はレーザー攻撃に必須の装置ではないためオプションとして扱える．今回利用したオシロスコープは，攻撃対象に対してかなりオーバースペックであり，より安価で低性能のオシロを用いても同じ実験を行うことができると考えられる．またレーザー照射タイミングのトリガを通信インタフェースや攻撃対象から直接とることができる場合には icWaves は不要である．これら二つの機器を抜いたときの価格は 394 万円となり，かなり低価格でレーザー装置を構築可能であることがわかる．最後に，これらの機器を用いて構築したレーザー装置の完成後の様子を図 2.5 に示す．以下ではこのレーザー装置のことを，YNU Single-Spot Laser Station(YNU-SLS) と呼ぶ．

表 2.1: シングルレーザー装置の価格（2018 年 12 月時点）

使用目的	装置名	値段 [万円]
安定化電源	松定 P4K18-2	10
レーザー	Alphanov PDM+	130
CCD カメラ	IDS UI-1240SE-NIR-GL	12
光学系	シグマ光機 NIR レンズチューブ	140
	シグマ光機 PAL-20-NIR-LC00	20
	日進電機 1064nm リングライト	30
	シグマ光機 Z ステージ	15
XY ステージ	シグマ光機 TSD-605C	10
パルス生成器	Riscure icWaves	450
	Xilinx Arty FPGA board	2
カードリーダー	SASEBO-W	25
オシロスコープ	Lecroy HDO6104A	310

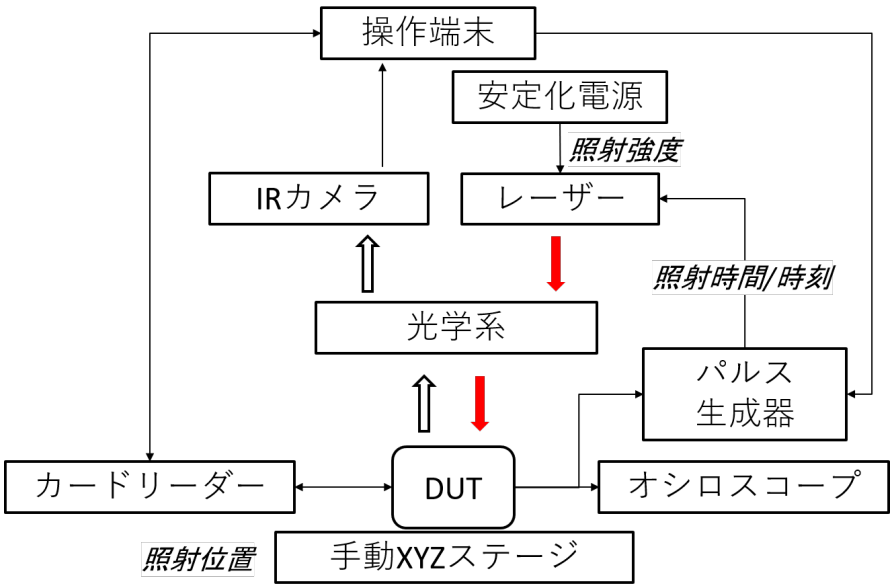


図 2.4: シングルスポットレーザー装置の構成

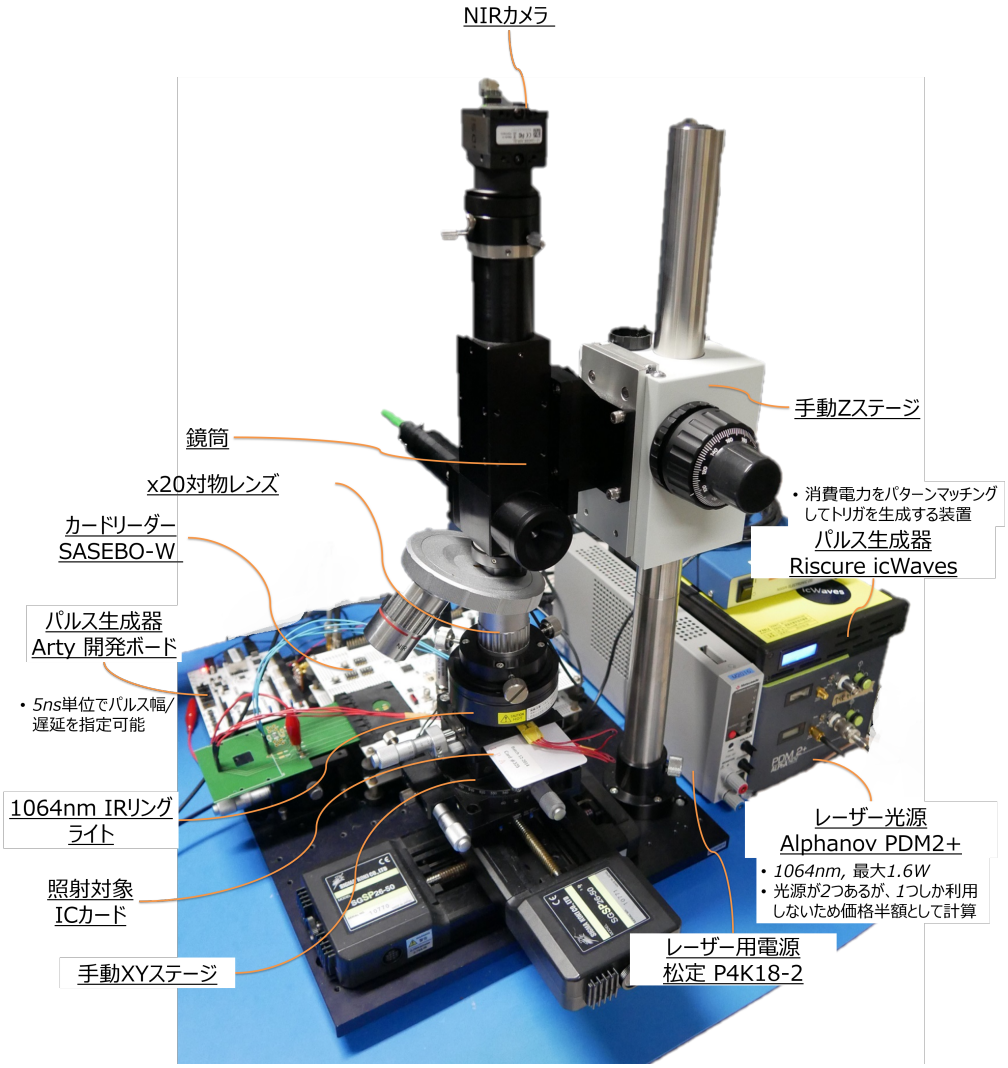


図 2.5: 構築したシングルスポットレーザー装置

2.4　ダブルスポットレーザー照射装置の構築

より強力なレーザー照射環境の構築を目指し，図 2.6 に示すダブルレーザー照射環境を構築する．光学系以外の構成は YNU-SLS とほぼ同一であり，レーザーと安定化電源の数が増えただけである．. ダブルレーザー装置における光学系の詳細を図 2.8 に示す．波長 1064nm の 2 本のレーザーが光学系で結合され，対物レンズを通して攻撃対象に照射される．この光学系は二本のレーザー光をターゲット上で独立に移動するために，二本のレーザーそれぞれの光軸の角度を照射位置設定用ミラーで調整できる．ビームエキスパンダーは取り替え可能であり，レンズを変更することで倍率を調整することが可能である．倍率 4 倍が最大倍率となり，このときに 20 倍の対物レンズを使うと理論上は約 4 µm のレーザースポットを得られる．光軸の結合には偏光ビームスプリッターを用いており，直線偏光のレーザーを二本利用する場合は偏光状態を直交させることでビームの強度を保ったまま照射可能である．結合されたレーザーはダイクロイックミラーにて反射され，対物レンズを通してターゲットに照射される．このダイクロイックミラーは 1200nm 以上の波長を透過するものであり，1200 nm 以上の光を投光することでターゲットの像を IR カメラ側に転送する．IR カメラには 1200nm 帯の波長に感度を持つ InGaAs カメラを利用している．加えてそれぞれのレーザーは照射タイミングを独立に設定する事ができるため，シングルレーザーの実験装置として使用することもできる．

本ダブルレーザー照射装置のセットアップにかかった費用はおよそ 1,000 万円以下であった（IcWaves, オシロ含まず）．以下ではこのレーザー装置のことを，YNU Double-Spot Laser Station(YNU-DLS) と呼ぶ．

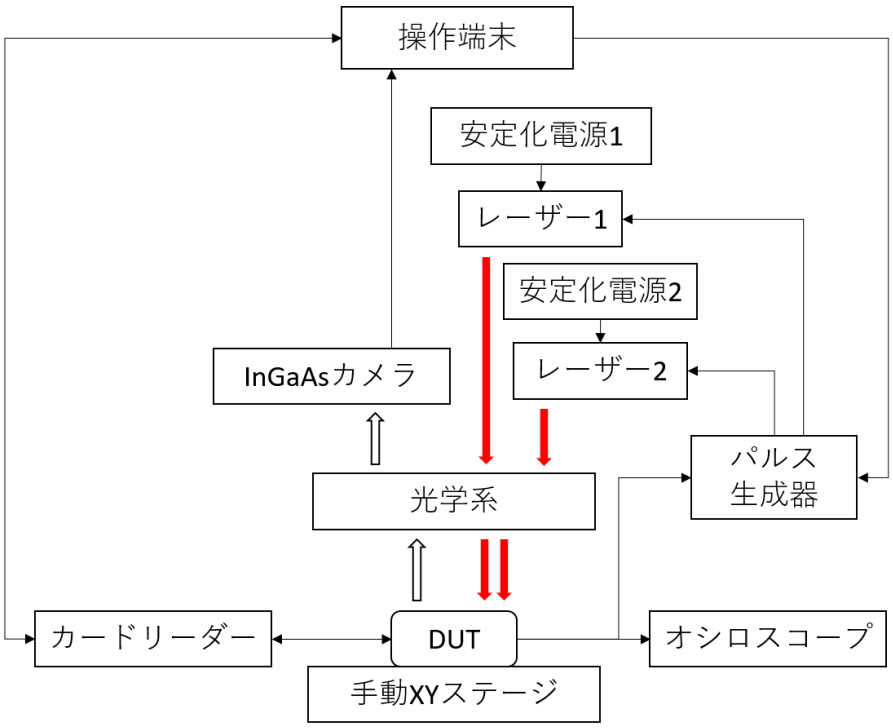


図 2.6: ダブルスポットレーザー装置の構成

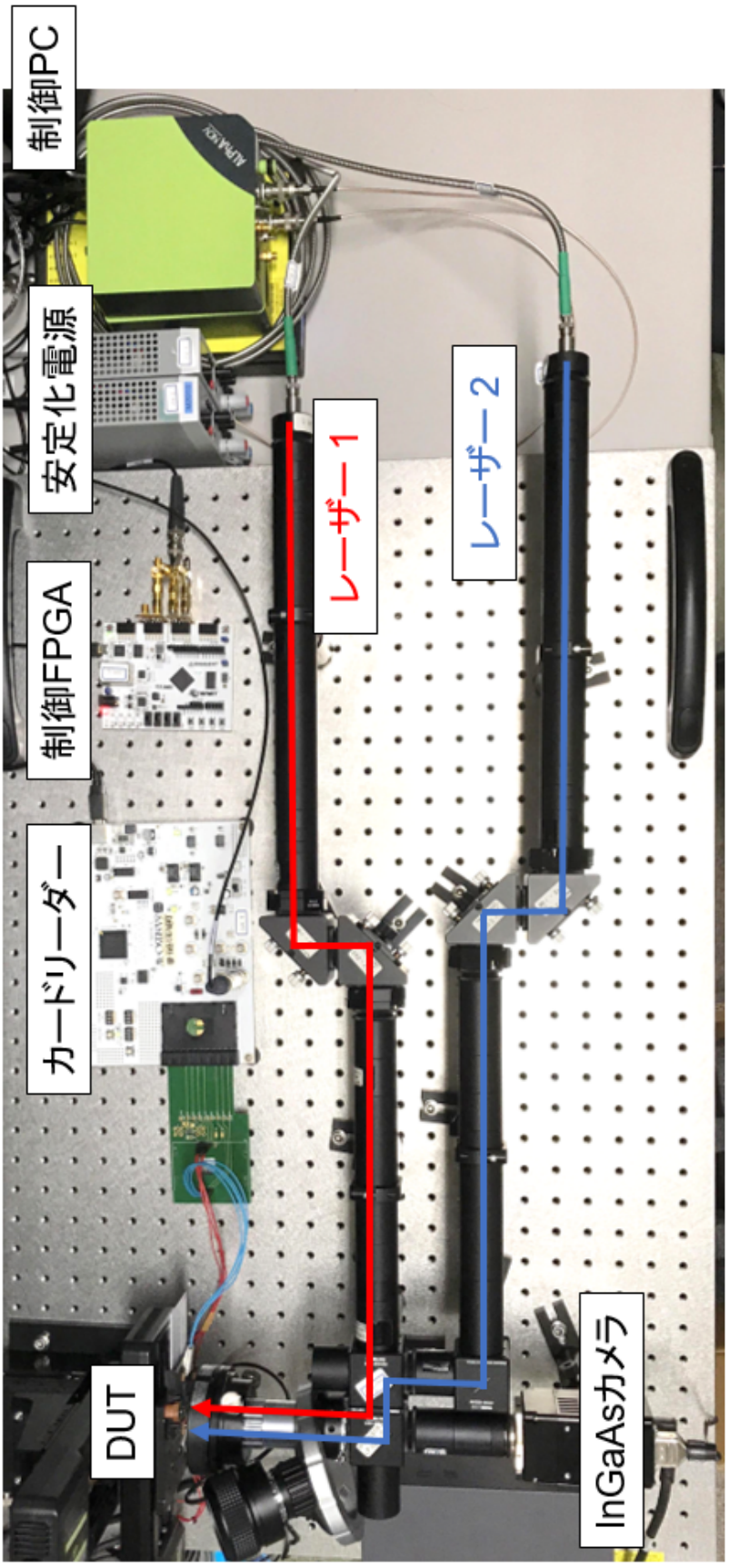


図 2.7: 構築したダブルスポットレーザー装置

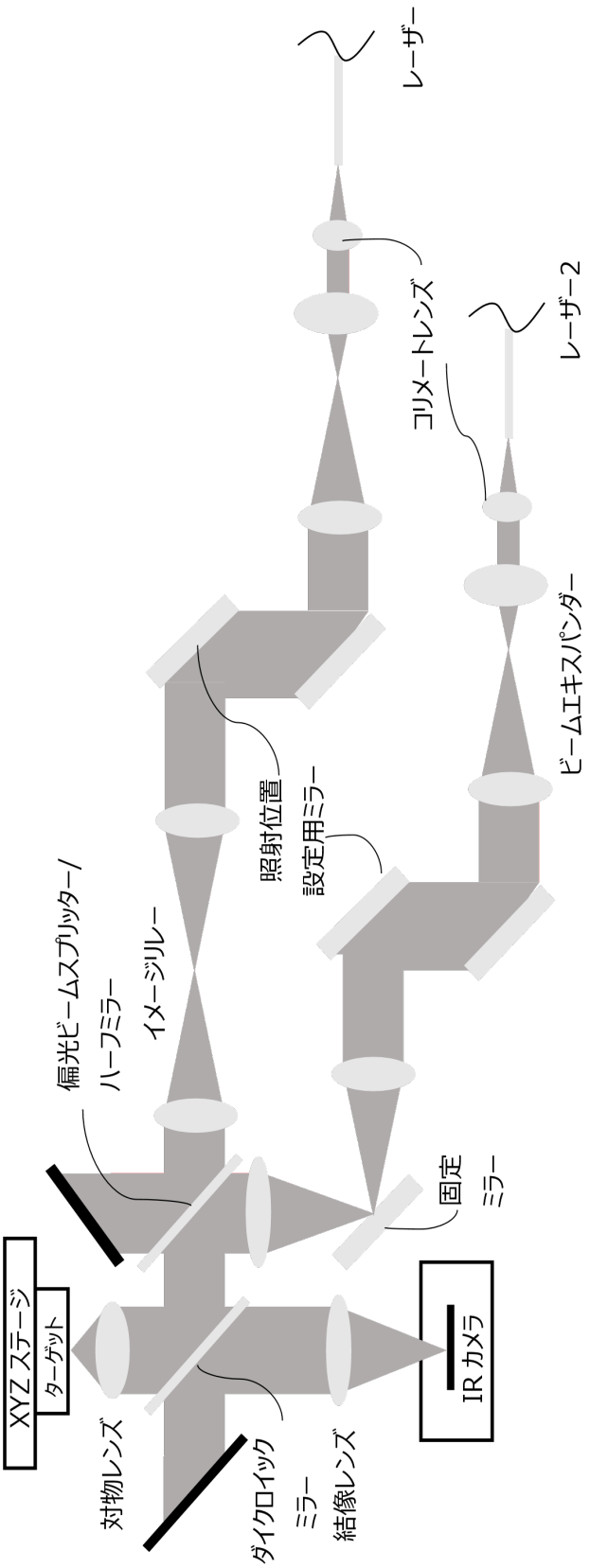


図 2.8: ダブルレーザー装置の光学系

2.5　トリプルスポット以上のレーザー照射装置構成の可能性

同時に照射可能なレーザーのスポット数は，レーザーフォールト攻撃の能力を制限する大きな要因のひとつである．現在市販されているレーザー装置は我々の知る限りではダブルスポットまでだが，Alphanov 社や ST Microelectronics 社の協同で，4-spot レーザー装置が開発されていることが報告されている [39]．図 2.7 で示した構成を拡張することにより，コストをかければレーザーのスポット数を増やすことは容易である．YNU-DLS では偏光ビームスピリッターを使って二つの光軸を一本にまとめているため，レーザー強度を落とすことなく光軸を結合できたが，3 本以上の結合はハーフミラーなどが必要になるため強度の減衰は避けられないと考える．しかしながら強度が小さくてもフォールト注入可能なデバイスに対してはこの強度低下は問題にならず，また高価で高強度なレーザーを利用することで解決できる．また，スポット数が増加したとしてもそれぞれのスポットの照射タイミングが同時でも良いのであれば光源の数は一つでもよい．この場合も各スポットのレーザー強度が低下するが，十分に低い強度でもフォールト注入可能なデバイスであれば問題にならないと考えられる．

第 3 章

レーザー照射攻撃の対象となり得るデバイスの構成

本章ではレーザーフォールト攻撃の対象となり得るデバイスの構成が解説される．レーザー攻撃の対象となるデバイスの構造を図 3.1 に示す．本研究ではデバイスとはパッケージングされた IC チップのことを指し，IC チップ内で CPU やメモリがバスで接続されている．守るべき秘密情報がチップ内に格納されており，通常は不揮発性の ROM に保存されている．しかしながら秘密情報の利用時には ROM から読み出され RAM や CPU のレジスタに展開されるため，デバイス動作時には ROM 以外の場所にも存在する可能性がある．またデバイスは CPU の他にもある演算に特化した専用コプロセッサを持つ場合がある．近年ではエッジデバイスにも公開鍵暗号のようなリッチな暗号機能が求められるようになっており，公開鍵暗号の計算に必要となる多倍長乗算器のニーズが高い．もちろんコプロセッサ内に秘密情報を持つ場合があり，コプロへのフォールト注入が情報漏洩を引き起こすことも考えられる．

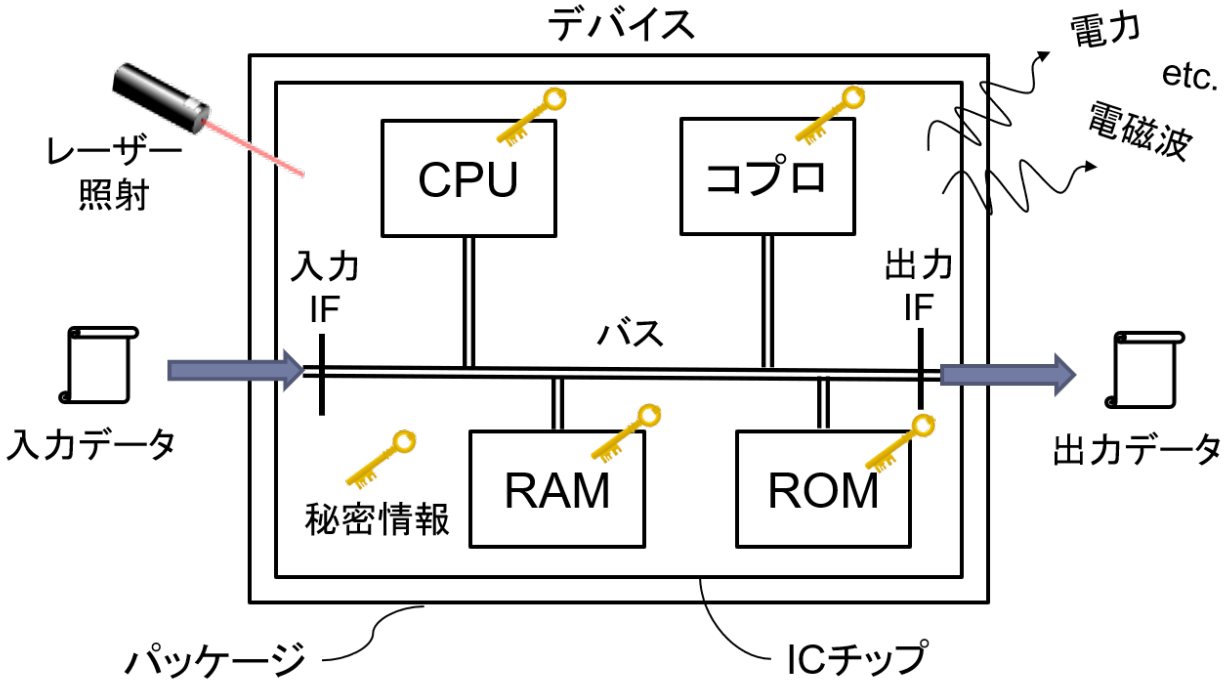


図 3.1: レーザーフォールト攻撃の対象となり得るデバイスの構造

攻撃者は 4 つのチャネルを使ってデバイスにアクセスすることができると考えられる．まずバスの入出力インタフェースを通して攻撃者はデバイスのデータを入出力することができる．このチャネルをメインチャネルと呼ぶ．また攻撃者は，消費電力や漏洩電磁波といったサイドチャネル情報を観測することができる．サイドチャネル情報はデバイスの動作や内部データと相関があることがわかっているため，サイドチャネル情報から重要な情報が漏洩する恐れがある．さらに攻撃者はデバイスに対してレーザー照射を行うことができる．レーザーは IC チップに照射することで何らかのフォールトを発生させるから，レーザーフォールトを行う攻撃者はまずパッケージを開封して IC チップを外部に露出させなければならない．レーザー照射によるフォールトは，メインチャネルからは書き換えられない場所のデータを書き換える行為だと考えることができるため，一種の入力チャネルとみなせる．このような電力や電磁波の観測及びレーザー照射の二つのチャネルを，サイドチャネルと呼ぶ．従って，攻撃者がデバイスから情報を得るためには，

- データ入力してデータ出力を観測する．
- データ入力してサイドチャネルを観測する．
- レーザー照射してデータ出力を観測する．
- レーザー照射してサイドチャネルを観測する．

の 4 つのパターンがあることがわかる．もちろんこれらの 4 つを自由に組み合わせて利用することもできる．セキュアなデバイスを実現するためにはこれら 4 つのパターンでの情報漏洩をすべて検討する必要がある．

以下では，本研究で使用するデバイス上のプロセッサアーキテクチャとして AVR, ARM の 2 つを紹介する．またコプロセッサの例としてペアリング演算専用回路を FPGA 上に実装した結果を示す．さらにデバイスのパッケージングの例をいくつか紹介し，本研究での実験に利用するためにパッケージ開封を行った結果を示す．

3.1　プロセッサアーキテクチャ

3.1.1　AVR アーキテクチャ

ATMega163

AVR は ATmel 社が開発した 8 ビットプロセッサアーキテクチャである．ATMega163 は ALU や ROM，RAM などの基本的な素子を備えたマイコンである．これらはすべて 8 ビットのバスで接続されており，外部との入出力は SPI や UART 等のインタフェース及びポートを通してやりとりされる．AVR は 2 段のパイプラインを持っており，命令の実行及び次の命令のフェッチが同時に行われている．また RISC ライクな設計であり 32 本の汎用レジスタを持ちほとんどの命令が 1 サイクルで実行される．AVR はハーバードアーキテクチャを採用しており，命令とデータのアドレス空間が完全に分離している．従って ROM が ROM 自身を書き換えるというようなことは難しい．ATMega163 の場合 ROM は 8k ワード，RAM は 1kB を搭載している．AVR の特徴の一つに，データパスが 8 ビットの一方で命令は 16 ビットというのがある．16 ビット=1 ワードである．書籍 [61, page 28] によると ATMega163 は第一世代のシリーズであり 600 nm というレガシープロセスで製造されている．

ATMega163 を実装した IC カードタイプのデバイスが 4.2 章の実験対象となる．

ATMega8515

本研究における実験では, もう一つの AVR マイコンとして ATMega8515[6] を利用する．ATMega8515 のアーキテクチャは ATMega163 とほぼ変わらないが，ROM が 4k ワード，RAM が 512 バイトに縮小されている．また ATMega8515 は第 2 世代の ATMega シリーズであり 350 nm プロセスで製造されている．

4.4 章では，ATMega 8515 が実装された IC カードに対して実験を行う．我々はまず，ATMega 8515 card のパッケージを開封しレイアウトの観察を行った．後述するが，IC カードタイプのパッケージングは容易に開封可能である．図 3.2 に示すように，フラッシュのレイアウトが簡単に特定できる．フラッシュメモリ部を拡大したものを図 3.3 に示す．Cell matrix は 16 の列から構成されており，これは AVR が 16bit を 1 ワードとするためである．このことから，n 番目の列には n 番ビット目のデータが格納されていることが予測できる．

実験に用いる DUT は，基板裏面からレーザーを照射するためにコンタクト電極の中央をナイフで取り除き基板を露出させている．これに加えて 4.4 章の実験では，裏面からの赤外観察のために基板を 50 μ m まで研磨した DUT を利用する．しかしながら，赤外照明や InGaAs カメラを利用すればこの研磨は不要であることを記しておく．

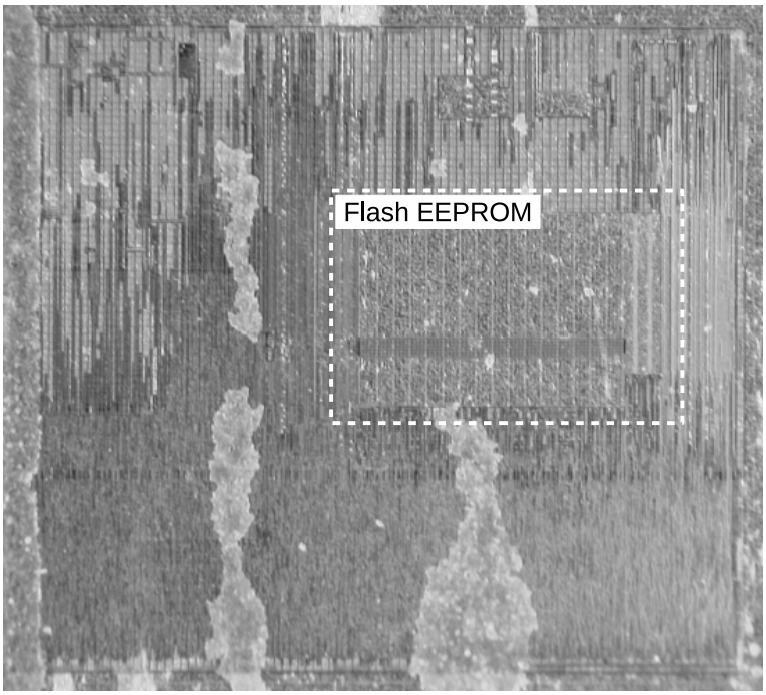


図 3.2: ATMega8515 のダイイメージ（表面可視光画像）

3.1.2　ARMv4 アーキテクチャ

ARM アーキテクチャは 16 本の 32-bit 汎用レジスタを持つ 32bit アーキテクチャであり，32 ビットの内部バスで ROM などの周辺回路を接続している．ARM 社はプロセッサの IP を販売しており，それを購入した企業は ARM プロセッサに独自のコプロセッサを追加してデバイスを運用することができる．ARMv4 アーキテクチャは年代的に 90–130nm プロセス で製造されたものと推測される．ARM7 ファ

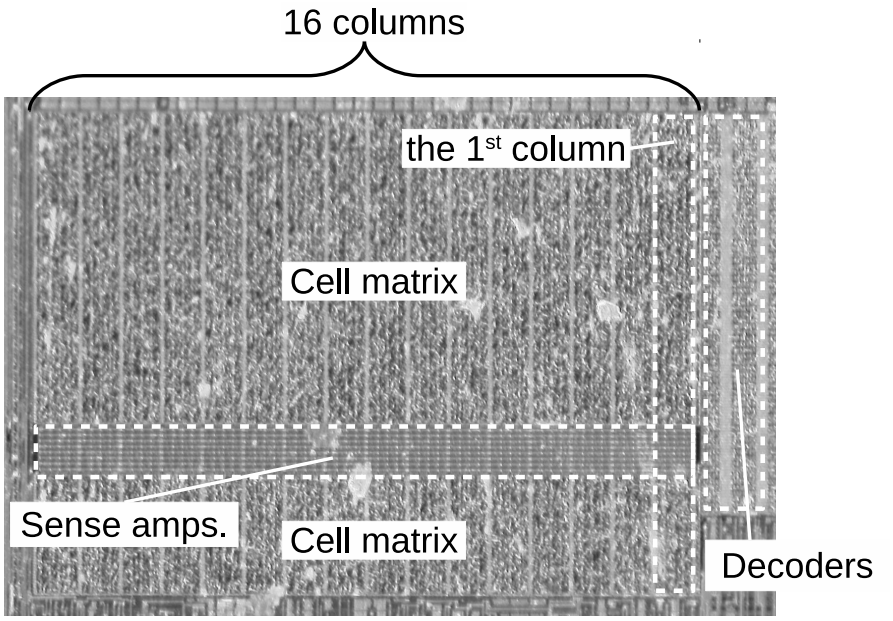


図 3.3: ATMega8515 のフラッシュ部分拡大図と構造の推測図（表面可視光画像）

ミリのコアはフォン・ノイマンアーキテクチャを採用しており プログラム空間とデータ空間が同一アドレス空間上にある． キャッシュメモリやメモリ保護ユニットは搭載していない．またフェッチ・デコード・実行の三段パイプラインを採用しており，ほとんどの ALU 命令は 1 サイクルで実行される．

IPA から提供された TVC（Test Vehicle Card）は ARMv4 ファミリの SC100 コアを実装した IC カードタイプのデバイスであり，4.2 章及び 4.3 章の実験で DUT として扱われる．TVC の動作には内蔵発振のクロックが使われており，オシロスコープでの 計測から周波数は 9MHz 程度と見積もった．TVC はある程度耐タンパー性が考慮されたデバイスであり，異常周波数検知回路及び異常電圧検知回路が実装されている．実験はこれらを有効にした状態で行われる．

また 5.3 章で提案される命令改変攻撃対策は ARM 16-bit Thumb 命令セット [5, Chapter 3] を対象とする．命令長が長くなるほど命令改変の挙動が複雑になり対策が困難になるため，16 ビットの命令セットを対象とした．ARM は世界で最も普及しているプロセッサの一つであり，ARM プロセッサ向けに書かれたアプリケーションソフトウェアは多岐にわたる．ARM プロセッサは伝統的に 32-bit の命令セットを備えているが，コード量や消費電力の削減のため ARMv4T 以降のプロセッサには新たに 16-bit の Thumb 命令セットが追加された．Thumb 命令セットは Thumb-1 と呼ばれることもあるが，本論文では単に Thumb と呼ぶ．ARMv7, ARMv8 のような最新のプロセッサはより洗練された可変長の命令セット Thumb-2 を実装しているが，Thumb-2 は Thumb のスーパーセットであるため Thumb で書かれたプログラムは Thumb-2 上でも動作する．このように Thumb 命令セット上で対策手法を提案することによりそのアプリケーションは多岐に渡る．ARM プロセッサは 16 個のレジスタ（r0-r15）を備えているが，Thumb 命令セットからはそのうちの 8 つ（r0-r7）にしかアクセスできない．また検算処理を行う上で重要となるのが条件分岐命令である．Thumb の条件分岐は cpsr (current program status register) 上の 5 つの条件フラグ，N (negative), Z (zero), C (carry), V (overflow), Q(saturation) の状態によって制御される．例えば beq (branch equal) 命令は $Z = 1$ のときに分岐し，bhi (branch unsigned higher) 命令は $C = 1$ かつ $Z = 0$ のときに分岐（ジャンプ）する．これらの条件フラグはプロセッサの演算結果の状態に従って随時セット/クリアされる．例えば検算処理の際には比較命令によって Z フラグを更新す

るのが一般的である．

3.2　コプロセッサ

エッジノードのような計算資源の少ないデバイスにおいても，公開鍵暗号のようなリッチな暗号機能を利用したいというニーズが高まっている．特に近年では公開鍵暗号の機能をより発展させた “高機能暗号” と呼ばれる暗号技術が IoT 時代の通信量増加などの課題を解決するものとして期待されている．しかし高機能暗号の計算は従来の公開鍵暗号よりもさらに計算コストのかかる “ペアリング” と呼ばれるプリミティブ演算が必要であり，計算資源の少ないデバイスにおいては実用的な時間内に計算終了しない．このような場合，デバイス内の CPU とは別に暗号処理専用に設計されたコプロセッサが搭載される場合がある．本節ではコプロセッサの例として低遅延ペアリング演算回路を FPGA 上に実装した結果を示す．独自に設計したコプロセッサはそのアーキテクチャの実装部分の詳細までわかっているから，レーザーフォールト注入時の影響を容易に解析できる．

3.2.1　ペアリング演算用コプロセッサの概要

図 3.4 に，開発したペアリング演算用コプロセッサの概要を示す．コプロセッサ中のメイン回路は 18 段パイプラインで構成された多項式演算用ユニット（Pipelined Polynomial Arithmetic Unit, PPAU）である．PPAU はプリアダー，13 段の剰余乗算器，定数倍演算器，ポストアダーで構成されており，特に以下のような 2 次拡大体の乗算

$$z_0 + z_1 i = (x_0 + x_1 i) \cdot (y_0 + y_1 i) = x_0 y_0 - x_1 y_1 + ((x_0 + x_1) \cdot (y_0 + y_1) - x_0 y_0 - x_1 y_1) i \quad (1)$$

を高速化するために設計されている $(z_0, z_1, x_0, x_1, y_0, y_1 \in F_p)$.

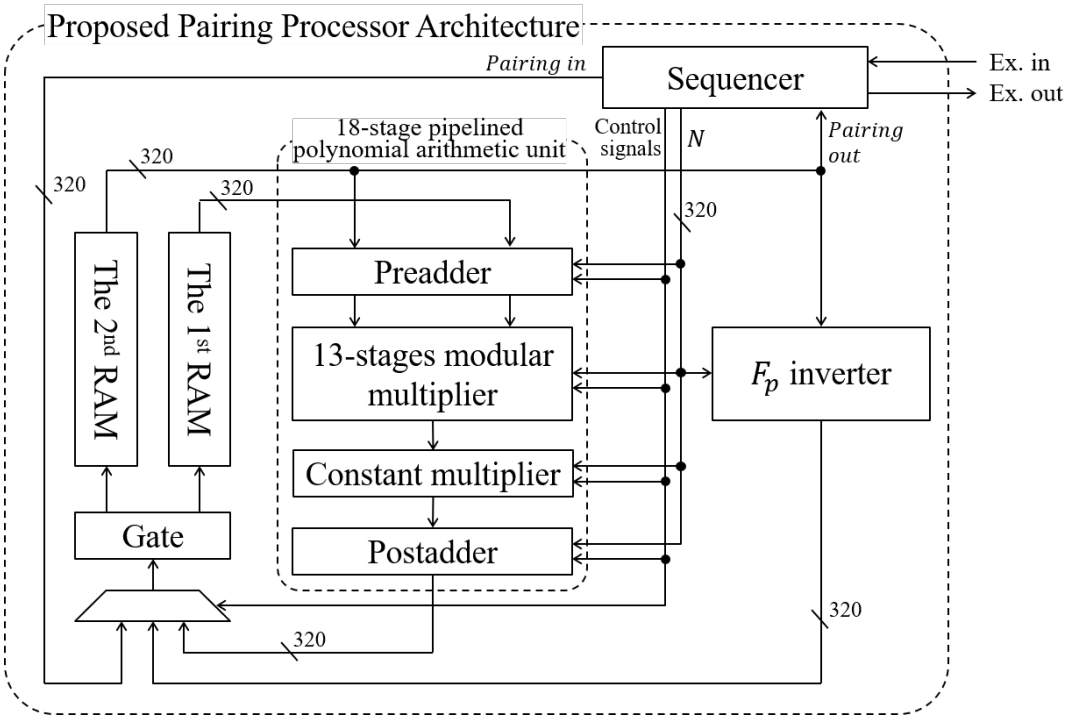


図 3.4: ペアリング演算用コプロセッサの概要

このようにペアリング演算には大きな標数上の有限体乗算が大量に必要であるため，コプロセッサの回路リソースの大部分は剰余乗算器に割かれている．またコプロセッサは有限体上の逆元演算を高速化するための専用回路も備えている．PPAU と逆元演算器はシーケンサによって制御され，RAM のデータを受け取って RAM にデータを出力する．今回の実装ではシーケンサと RAM はコプロ内に実装しているが，このコプロセッサを外部の CPU とバスで接続して動作させる際にはシーケンサはデバイス上の ROM に，RAM はデバイス上の RAM に置き換わる構成が考えられる．またシーケンサ内の ROM にはコプロセッサの動作を制御するプログラムとも言うべき制御信号が格納されているため，コプロセッサも後に説明する命令改変攻撃の対象になり得ると考えられる．

3.2.2　プリアダー

プリアダーは剰余乗算器の入力となる値を計算するための回路である．図 3.5 にプリアダー回路の概要図を示す．この回路は文献 [58] を参考に修正を加えたものである．プリアダーは続く剰余乗算器の入力となる被乗数 $A \bmod N$ 及び乗数 $B \bmod N$ を計算するための二つのアキュムレータを持っている．各アキュムレータには 3 つの動作モードが存在する．左のアキュムレータは動作モードによって出力を $X_{t-1}, X_{t-1} + X_{t-2}, X_{t-1} + Y_{t-1},$ の 3 つから選択することができる．ここで X_t は時刻（サイクル）t でのアキュムレータの入力データを指す．右のアキュムレータは動作モードによって出力を $Y_{t-1}, X_{t-1} + Y_{t-1}, Y_{t-1} + Y_{t-2}$ の 3 つから選択できる．F_{p^2} 乗算を計算する際には次のような流れでプリアダーは動作する．まず 2 つのデータ x_0 (または y_0) 及び x_1 (または y_0) がそれぞれ左（右）のアキュムレータに入力される．続いて x_0 (または y_0), x_1 (または y_1), $x_0 + x_1$ (または $y_0 + y_1$) がそれぞれ左（右）のアキュムレータから 3 サイクルかけて出力される．これらの値が後段の剰余乗算器の入力となる．

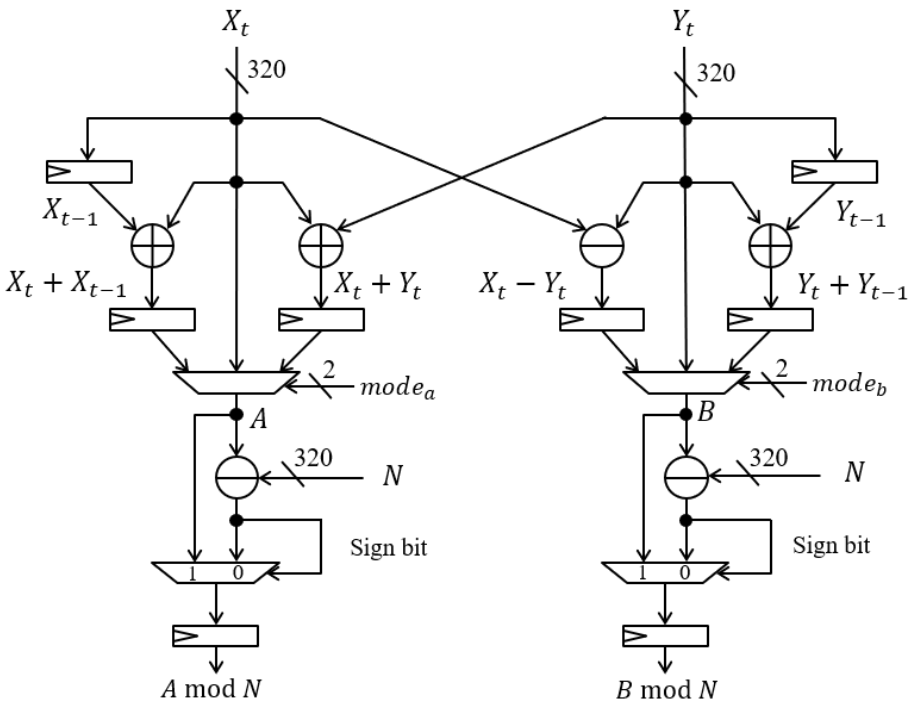


図 3.5: プリアダーの構成

3.2.3　剰余乗算器

剰余乗算はペアリング演算の計算量のほとんどを占めているため，高速な剰余乗算器は低遅延なペアリングコプロセッサを開発するために不可欠である．いくつかの剰余乗算アルゴリズムが提案されているが，本研究のペアリングコプロセッサでは quotient pipelining Montgomery multiplication (QPMM) algorithm [40] を採用する．QPMM のアルゴリズムを Algorithm 1 に示す．今回の実装ではパラメータ $k = 32, d = 1, n = 10$ とした．OPMM はアルゴリズムレベルで計算依存性が低く，パイプライン化したときにクリティカルパスが短くなるように設計できることが利点である．一方で OPMM は明らかな欠点も持っており，ループ段数の増加やデータパス幅の増加は無視できない影響を及ぼす．QPMM は，通常のモンゴメリ乗算を冗長に計算することで計算依存性を削減しているため，256 ビットの剰余乗算を行うのに 320 ビットの計算が必要になる．また QPMM 内のループ回数は乗算計算のビット幅に依存するため，256 ビットのとき 8 段のループで済んでいたのが 320 ビットだと 10 段のループが必要になる．しかしながら，これらの影響を差し引いても QPMM は高速に剰余乗算を実行でき，回路規模の増加よりも高速化が重要とされる場面では QPMM が利用される．

Algorithm 1: Quotient Pipelining Montgomery Multiplication [40]

Input: 基数:k; 遅延定数:d; 分割数:n; A; B; N; $N > 2$, $\gcd(N, 2) = 1$, $(-NN') \bmod 2^k = 1$,
$\tilde{N} = (N' \bmod 2^{k(d+1)})N$, $4\tilde{N} < 2^{kn} = R$, $N'' = (\tilde{N} + 1)/2^{k(d+1)}$, $0 \le A, B \le 2\tilde{N}$,
$B = \sum_{i=0}^{n+d} (2^k)^i b_i$, $b_i \in \{0, 1, \cdots, 2^k - 1\}$, $b_i = 0$ for $i \ge n$.

Output: $S_{n+d+2} \equiv ABR^{-1} (\bmod N), 0 \le S_n \le 2\tilde{N}$.

$S_0 := 0$; $q_{-d} := 0; \cdots; q_{-1} := 0$;
for $i := 0$ **to** $n + d$ **do**
　$L_1 : q_i \quad := S_i \bmod 2^k$;
　$L_2 : S_{i+1} := S_i/2^k + q_{i-d}N'' + b_i A$;
end
$S_{n+d+2} := 2^{kd} S_{n+d+1} + \sum_{j=0}^{d-1} q_{n+j+1} 2^{kj}$;
return S_{n+d+2};

QPMM アルゴリズムを図 3.6 に示すような回路として実装した．低遅延を達成するための基本的なアイデアは，QPMM アルゴリズムを完全にアンロールして実装することである．このようなアプローチは軽量暗号の一種である低遅延暗号 [15] の構築に用いられている．アンロール実装のメリットの一つは，不要な部分の回路を大きく削減できることである．Algorithm 1 は明らかに不要な計算，例えば $i = 0, 1$ における $q_{i-d}N''$ や $i = 10, 11$ における $b_i A$ を含んでおり，アンロール実装ではこの部分の回路を削減できる．図 3.6 にはこのような不要計算は図示されておらず，i 番目のパイプラインステージが $i - 1$ 番目のループに対応している．2^k の剰余及び除算はビットマスク及びシフト演算によって簡単に計算できるため，QPMM アルゴリズムの中でこれら以外の部分，すなわち乗算 $q_{i-d} \times N''$, $b_i \times A$，及び加算 $S_i/2^k + q_{i-d}N'' + b_i A$ を高速計算する回路が必要である．これらの計算は Partial Products (PPs) generator 及び Partial Sums (PSs) generator 回路で計算される．

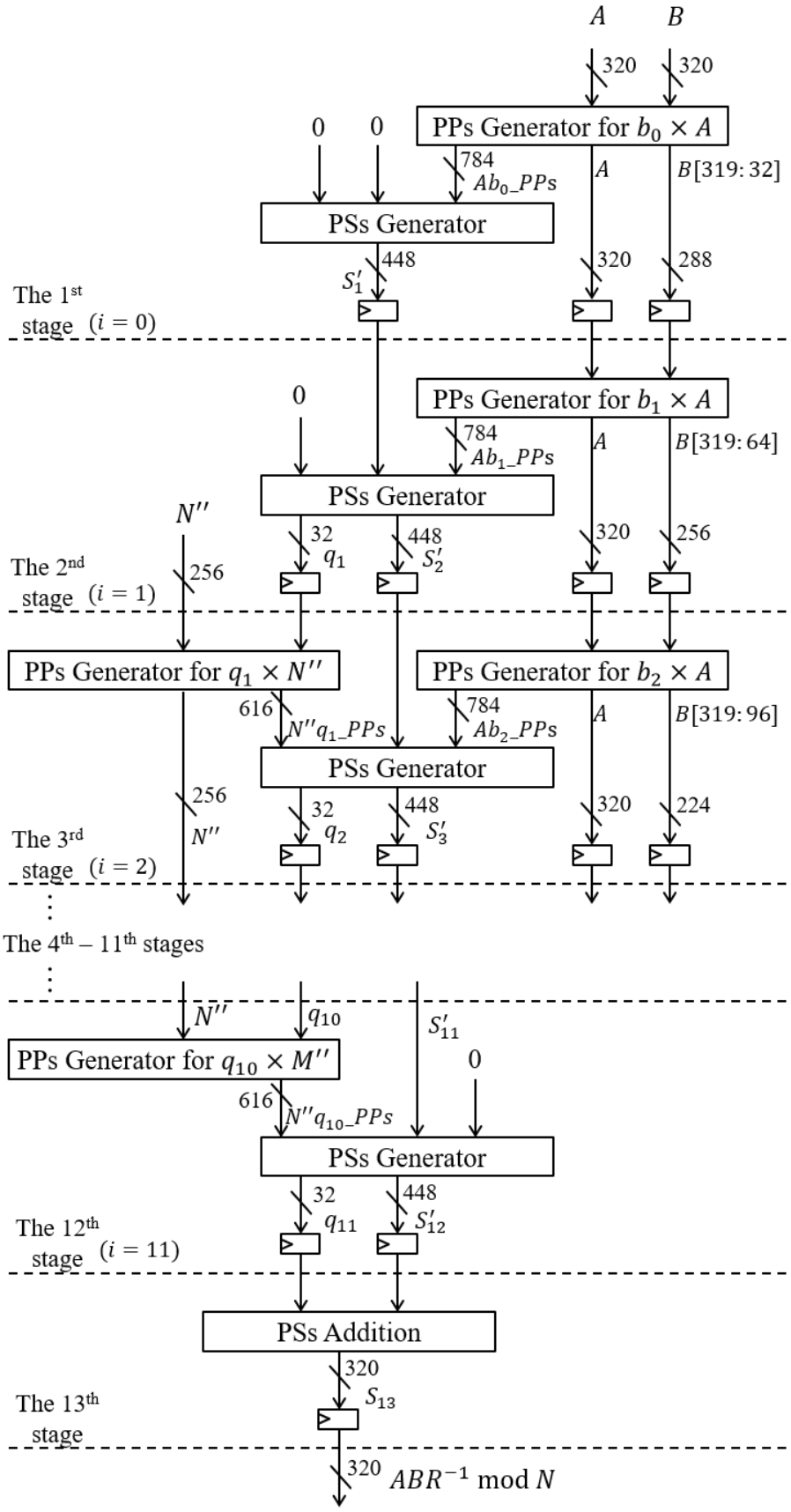


図 3.6: 剰余乗算器の構成

■PPs generator　PP generator（図 3.7）は乗算 $q_{i-d} \times N''$ 及び $b_i \times A$ を計算するためのユニットである．これらの乗算は 24×32bit の部分積の和として計算されるが，加算部分はこのユニットでは計算しない．このユニットでは乗算を完全には計算しないので乗算器ではなく PP generator という呼び方をしている．各 PP は FPGA 上の専用回路である Digital Signal Processor (DSP) を使って計算される．Xilinx Virtex-6 FPGA の DSP の場合は 24×17 ビットの乗算を効率的に計算できる．DSP には 17 ビットのシフタや乗算結果に値を足し込むポストアダー回路が含まれており，これらの機能を利用して二つの DSP をひとつの 24×32 bit PP 計算ユニットとする．32×320 bit 乗算を計算するためには $\lceil 320/24 \rceil = 14$ 個の PP が必要である．PPs generator の出力は $784 = 56 \times 14$ ビットとなる．

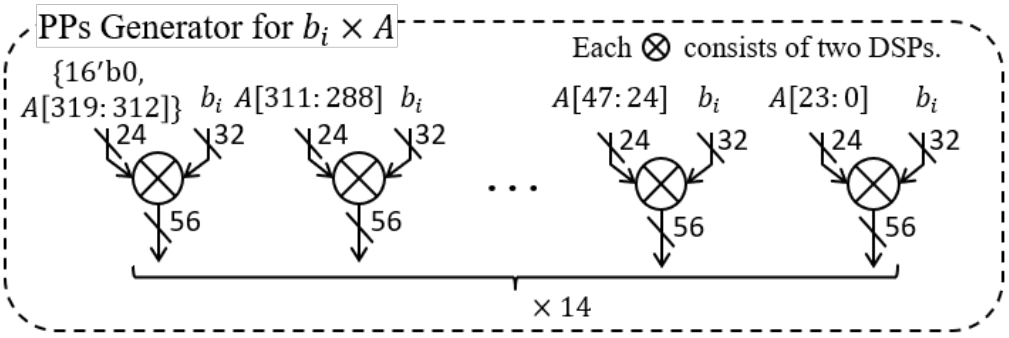


図 3.7: PPs generator の構成

■PSs generator　PSs generator（図 3.8）は $S_i/2^k + q_{i-d}N'' + b_iA$ の加算部分を計算するユニットである．この計算は PP gernerator によって計算される二つの乗算結果に依存するため，この部分をできるだけ高速に計算することがペアリング全体の高速化につながる．PPs generator はこの加算部分を 32 ビットごとに各 PS で並列に計算し，それぞれの PS はキャリーを伝搬しない．キャリーはすべて 13 段目にある PSs Addition ユニットで足し合わせられる．

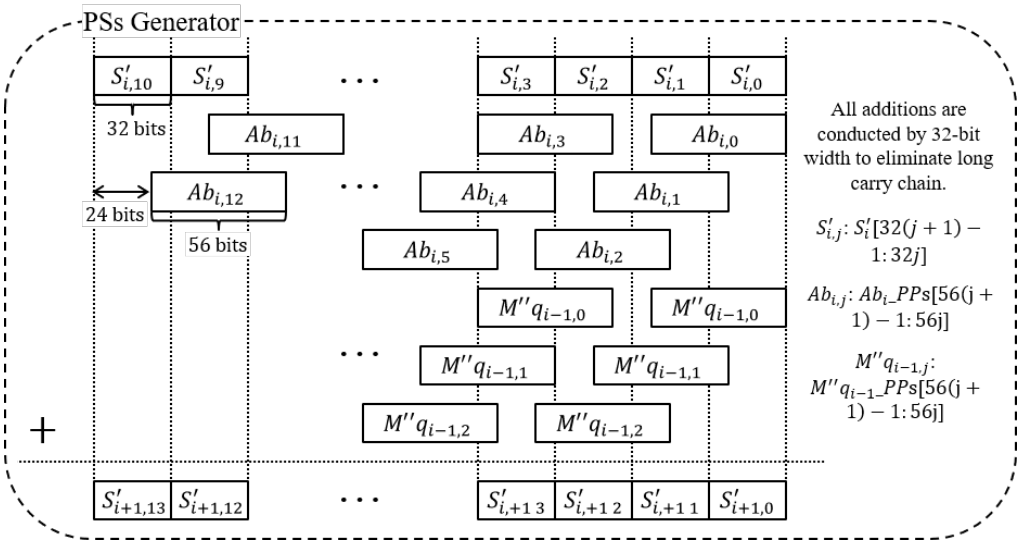


図 3.8: PSs generator の構成

3.2.4　定数倍演算器

楕円曲線上の加算などで拡大体の多項式計算を行う際に発生する定数乗算は定数倍演算器によって計算される．乗じる定数は演算ごとに選択することができ，1 倍 (そのまま出力)，2 倍，3 倍，4 倍，6 倍を出力する機能をもつ．これらの小さな係数による乗算はシフタと加算器で実現できるため，乗算器を持つよりも効率的に計算可能である．

3.2.5　ポストアダー

ポストアダーは剰余乗算後の値をアキュムレートするユニットである．ポストアダーの構成図を図 3.9 に示す．ポストアダーは 3 つのアキュムレータ及びそれらの一つを出力として選択するためのセレクタ回路から成る．各アキュムレータは F_p 加減算器と 320 ビット幅の RAM を持っている．図に示されている $mode$ 信号によってアキュムレータ動作は制御されており，6 パターンの動作を行うことができる．

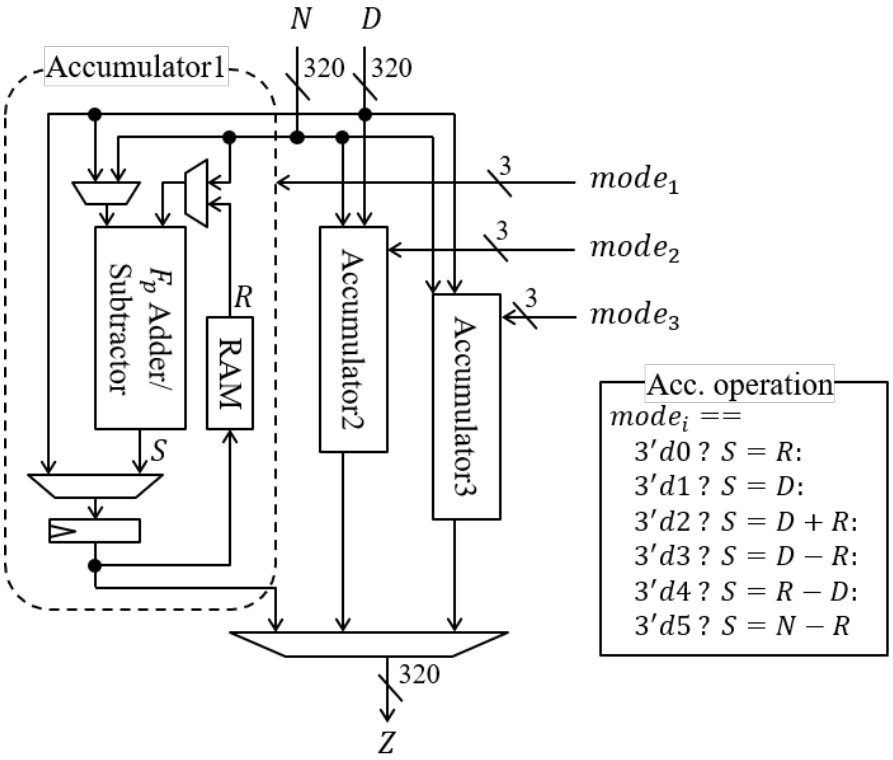


図 3.9: ポストアダーの構成

3.2.6　逆元演算器

F_p 上の逆元演算は，ペアリング計算を構成する演算の中でも特に計算量が多い．従ってペアリング計算アルゴリズムの多くはできるだけ F_p 逆元演算の回数を減らすように設計されており，本研究で実装するペアリングアルゴリズムでも F_p 逆元演算は 1 回しか登場しない．F_p 上の逆元を計算するアルゴリズムの一つにフェルマーの小定理を用いるものがあり，これは PPAU で計算できるがフェルマーの小定理は計算の依存性が高く PPAU のようなパイプライン段数の長い演算器で計算するとパイプライン空きが

大量に発生してしまう．p が 254 ビットの場合 F_p 上の逆元計算には少なくとも 254 回の F_p 乗算が現れるため，PPAU で計算した場合逆元演算には少なくとも $18 \times 254 = 4572$ サイクルかかる．これはペアリング全体のサイクル数の 2–3 割を占めており，無視できないほどのボトルネックとなっている．逆元演算のボトルネックを避けるために，提案コプロセッサには逆元演算専用のユニットが実装されている．このユニットは拡張ユークリッドアルゴリズムを使って逆元を平均 350 サイクルで計算することができ，逆元演算を大きく高速化することができる．拡張ユークリッドアルゴリズムはいくつかの加算器とシフタで構成できるため，ペアリング回路全体で見たときの回路規模は小さい．

3.2.7　スケジューリング

提案するコプロセッサを使ってペアリング計算を行うためには，コプロセッサの動作を制御する制御信号の列を組み上げる必要がある．提案コプロセッサは長いパイプラインを持っているため，制御信号のスケジューリングの仕方によってパイプライン効率が大きく変化し，高速なペアリング演算のためには洗練された制御信号スケジューリングが必要である．

提案コプロで利用する制御信号スケジューリングの大部分は Yao らの文献 [58] を参考にしている．この文献では 18 段パイプラインを持つペアリングプロセッサが提案されており，それに適したペアリング計算方法が示されている．PPAU も 18 段パイプラインを持つことから，彼らと同じスケジューリングによって効率のよい計算が可能である．しかしながら，彼らのスケジューリングがミラーループ部の計算で 98.7% というパイプライン効率を示している一方で最終ベキ部分は 78.2% と改善の余地がある．最終ベキ計算は easy part $f^{((p^6-1)(p^2+1))}$ の計算及び hard part $f^{((p^4-p^2+1)/r)}$ 計算の二つに分けることができ，その名が示すとおり hard part の計算量が大部分を占める．これまでの研究 [3] では hard part 計算に圧縮自乗算 [29] 使うことによってペアリング全体の計算量を減らすことができると示されている．しかしながら，PPAU のようなパイプラインを持つ回路で計算を行う場合，計算量が多くなったとしても計算依存性の少ないアルゴリズムを使った方が全体の計算時間を削減できる可能性がある．圧縮自乗算アルゴリズムにもいくつか種類があり，Yao らの実装では最も計算量の少ない SQR_{2345} が利用されていたが，考察の結果 PPAU には SQR_{12345} がより適していることがわかった．SQR_{12345} は SQR_{2345} よりも計算量が増加するが，計算依存性が少なく，パイプライン空きを減らすことができる．

本研究でのスケジューリングによるパイプライン効率及びペアリング計算にかかるサイクル数を表 3.1 に示す．本研究の実装は Yao らの実装 [58] と比べて最終べき部分のパイプライン効率が大きく向上しており，ペアリング全体のパイプライン効率も 97.6% という高効率を達成している．

表 3.1: ペアリング計算にかかるサイクル数とパイプライン効率の比較

Design	ML	ML Occu. Rate	FE	FE Occu. Rate	Total	Total Occu. Rate
ours	9700	99.5%	8751	93.6%	18151	96.7%
[58]	37976	98.7%	56124	64.4%	94100	78.2%

3.2.8　実装結果

評価プラットフォーム

2 種類の FPGA ボード，ML605 及び VCU118 上に提案ペアリングコプロセッサを実装し，性能評価を行う．ML605 ボードには比較的古いプロセスで製造された 40 nm Virtex-6 xc6vlx240t-3 が実装されている．ペアリングコプロセッサ開発の先行研究の多くがこの FPGA を使っているため，比較を行うために利用する．一方で VCU118 ボード上には先端プロセスで製造された 16 nm Virtex Ultrascale+ xcvu9p-2 が実装されており，最新のプラットフォーム上での性能が評価される．

実装結果

FPGA 上での実装結果及び先行研究との性能比較を表 3.2 に示す．提案ペアリングコプロセッサは大量に回路リソースを消費する代わりに最も高速にペアリングを計算できることがわかる．

表 3.2: FPGA 上に実装されたペアリング計算回路の性能比較

	デバイス	回路リソース	動作周波数	サイクル数	計算時間	消費電力
本研究	VCU118 実機	5421 CLBs, 460 DSPs	288 MHz	17580	61 µs	NA
	シミュレーション		173 MHz		102 µs	1.754 W
	ML605 実機	11672 slices, 500 DSPs	115 MHz		153 µs	NA
	シミュレーション		89 MHz		198 µs	1.227 W
[25]	xc6vlx240t-3	5163 slices, 144 DSPs	166 MHz	62166	375 μs	NA
[57]	xc6vlx240t-2	7032 slices, 32 DSPs	250 MHz	166027	664 µs	NA
[58]	xc6vlx240t-1	5237 slices, 64 DSPs	210 MHz	77769	409 µs	NA
[28]	65nm LP CMOS	323k gates	633 MHz	330053	554 µs	NA
[33]	65nm CMOS	354k gates	800 MHz	512541	640 µs	NA

3.3　IC のボンディング方法及びパッケージ開封方法

シリコン上に製造された IC を外部と接続するためには，何らかの方法でボード基板上の配線と接続する必要がある．このような配線の手法を “ボンディング” と呼び，ワイヤボンディング手法がこれまで主流だったがフリップチップボンディングが近年増加している．また，IC チップがボンディングされた後 IC チップは物理的な衝撃などから守るために樹脂などで封入されるが，このプロセスを “パッケージング” と呼ぶ．レーザー攻撃を行うためにはパッケージを開封して IC チップを露出させる必要があり，攻撃を行うまでの難易度がパッケージングやボンディング手法によっても様々である．ここでは代表的なボンディング手法と，それを開封する方法について解説する．

3.3.1　ワイヤボンディング

ワイヤボンディングによるボンディングを図 3.10 に示す．最も一般的なボンディング手法であり，ほとんどの IC チップはワイヤボンディングによって基板へ接続されている．ワイヤボンディングには，リードを介して基板と接続する SMD 型と，リードを介さず直接ワイヤが基板にボンディングされる COB 型があるが，多くのデバイスで利用されているのは前者である．ワイヤボンディングの場合ボンディングだけだとシリコンは基板に固定されず，またワイヤは非常に繊細であるため物理的な衝撃から保護する必

要がある．よってチップとワイヤは樹脂などのパッケージに封入されており，これが我々がよく目にする Dual In-line Package (DIP) や Small Outline Package (SOP) タイプのディスクリート IC である．パッケージングされた状態ではレーザー照射のような侵襲攻撃を行うことができないため，パッケージングは侵襲攻撃の簡単な対策になっているといえるが，攻撃者はいくつかの方法でパッケージを開封し IC チップを（動作する状態で）外部に露出させることが可能である．

ワイヤボンディングされたパッケージングを開封するには，チップの上面（配線層）側を露出させる場合と，チップの裏面（シリコン）側を露出させる場合がある．チップ上面から開封を行う際には発煙硝酸のような強酸によって樹脂パッケージを溶かすのが主流である．ブログ [53] にて，DIP タイプの PIC18F1320 を硝酸で開封した結果が示されている．図 3.11 に，我々が硝酸で開封した IC の画像を示す．このブログでは，開封した IC に UV ライトを照射して PIC マイコンのセキュリティヒューズを書き換えることに成功している．

この様に個人レベルでもワイヤボンディングタイプの IC チップの上面からパッケージを開封し，攻撃を行うことが可能だが，硝酸のような薬品を扱うのはそれなりにコストが高いと考えられる．一方で，ワイヤボンディングタイプの IC チップを裏面から開封する際には薬品を利用しなくてもよい．裏面からの開封はドリルなどを使って樹脂に穴を開ける手法が一般的である．上面からの開封時にはドリルを使うとワイヤを傷つける恐れがあるが，裏面にはワイヤのようなセンシティブな部品がないため，慎重にドリリングを行うことで IC チップを破壊せずに露出させることができる．裏面開封の場合，露出したシリコンは可視光域ではほぼ鏡面に見えるが，赤外線カメラなどを用いることでシリコンを透過して配線層の構造を見ることも可能である．

IC カードのような，非常に薄いパッケージングを要求される場合 COB タイプのボンディングが利用される．このような場合基板も非常に薄い場合が多く，カッターナイフなどで簡単に基板を切除できる場

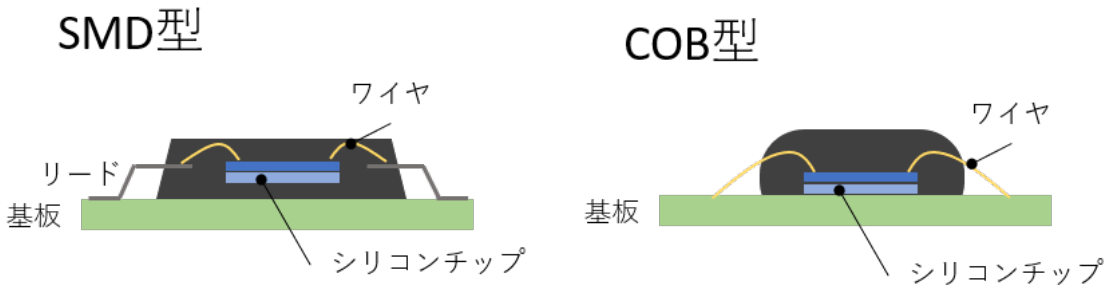


図 3.10: ワイヤボンディングによる基板への実装

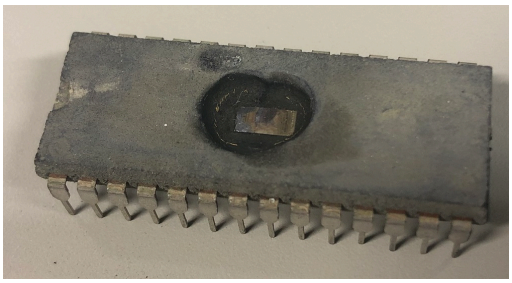


図 3.11: DIP パッケージの上面を薬品で開封

合がある．チップと基板の間には通常樹脂などが封入されておらず，接着剤のようなものでチップを基板に貼り付けてある場合が多い．この接着剤もカッターや溶剤で簡単に除去できるため，COB タイプのチップの裏面露出は容易に行える．

3.3.2　フリップチップボンディング

近年のデバイスの小型化のニーズに伴い，パッケージを含めた IC チップにも物理的な小ささが求められるようになっている．従来のワイヤボンディングによる基板との接続では，どうしてもワイヤ分パッケージの大きさが大きくなってしまう．そこでワイヤを使わずにバンプと呼ばれる突起を使ってボンディングを行う手法が開発されている．バンプを使うパッケージングには Ball Grid Array (BGA) や Pin Grid Array (PGA) などがあるが，バンプを IC チップの表面に直接取り付け，チップの配線層側を基板に向けてボンディングすることを，フリップチップボンディングと呼ぶ．図 3.12 に示すように，フリップチップはシリコン基板面が上向きに実装されており，また樹脂封入が成されていない場合もありレーザー裏面照射が容易に行える．図 3.13 のように一部のフリップチップは金属製のふたでシリコンを覆っているものもあるが，蓋は簡単に取り外すことができ（いわゆる殻割り），このような場合も簡単にレーザー攻撃を行うことができる．

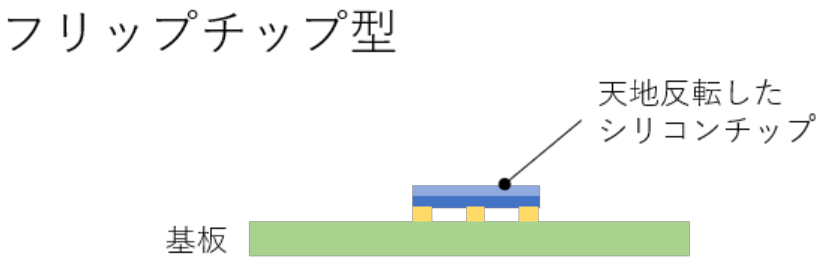


図 3.12: フリップチップボンディングによる基板への実装

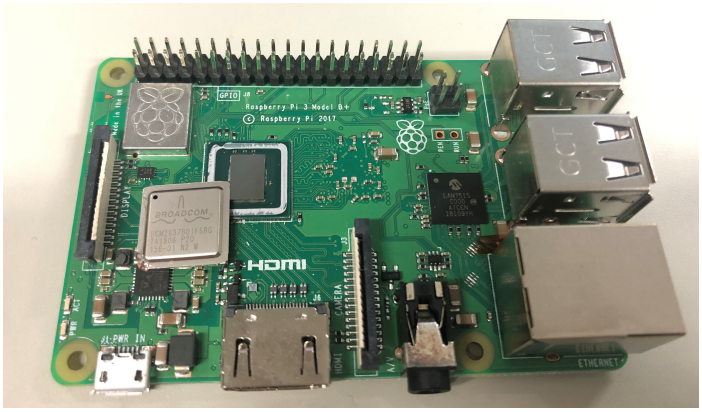


図 3.13: 殻割りした Raspberry Pi 3 Model B+ 上 CPU の様子

第 4 章

組み込みプロセッサに対するレーザー照射攻撃の効果

本章では，ここまでの章で導入したレーザー装置を使ってレーザー攻撃対象となるデバイスにレーザー照射を行った際に，どのような影響があるかを示す．まず先行研究におけるレーザー照射のデバイスへの影響を紹介する．先行研究で示されているレーザー照射の影響のほかに新たに，本研究ではレーザー照射が照射位置の局所的な消費電力を示すこと，及び実行中の命令の 1 ビットを改変できることを示す．

4.1　レーザー照射攻撃の先行研究

4.1.1　IC へのレーザー照射によるフォールト生起の原理

IC チップへレーザーを照射した際の物理的な挙動から，フォールトの原理を解説する．図 4.1 に，IC チップ上のトランジスタにレーザーが照射された際の様子を示す．シリコンのバンドギャップを超えるエネルギーを持ったレーザー光などの強力な放射線がシリコンに照射された際，放射線の軌跡に沿って光電効果による電子正孔対が生成される．もし軌跡が pn 接合面を通過していた場合，キャリアとなる粒子が電界に引かれドリフト電流となる．このときトランジスタの閾値電流を超える電流が流れると，OFF 状態だったトランジスタが ON 状態になる．IC チップのほとんどはトランジスタで構成されているため，トランジスタの状態が変化すれば，それは IC 動作の故障（フォールト）として観測される．一般的には，レーザー照射の影響はドレイン/ソースからグラウンドへの電流源としてモデル化される．

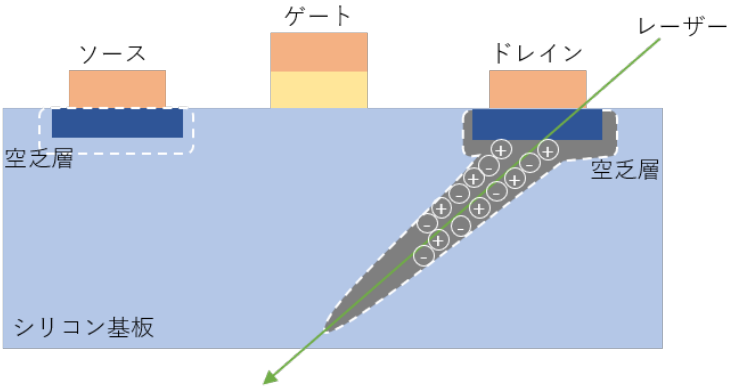


図 4.1: IC へのレーザー照射によるフォールト生起の原理

4.1.2　レーザー照射によるフォールト攻撃

初めてレーザーをフォールト注入に使ったのは Skorobogatov らであり，マイコン中の SRAM の任意のビットを反転できることが示された [50]．Skorobogatov はその後もレーザーを利用したフォールト注入の研究を続けており，文献 [49, 52] では EEPROM へレーザーを照射する攻撃を提案している．これらの攻撃はメモリに保存されている値の抽出を目的としておりデータフローへの攻撃といえる．レーザー照射を利用したプログラムフローへの攻撃としては，文献 [55] でレーザー照射によってセキュアマイコン上に実装した DES を攻撃した結果が示されている．ここではレーザー照射によって命令スキップが生じたと結論付けられている．

4.1.3　レーザー照射によってサイドチャネル情報を増加させる攻撃

組込み機器の多くは不揮発性メモリに秘密鍵やパスワード，実装プログラム等の秘密情報を格納しており，これらの守秘性は十分に保たれなければならない．製造者はふつうセキュリティヒューズを用いて許可のない外部からのメモリ読み出しを禁じているし，より高いセキュリティが必要とされる場面ではそもそも外部からのメモリ読み出し機能を実装していないことさえある．このような状況下では，攻撃者は正規経路外すなわちサイドチャネルからの情報を用いて不揮発性メモリの内容を取得しようとする．

図 4.2 に示されるように，サイドチャネル情報を使ったメモリのデータ抽出攻撃は Invasive, Semi-invasive, Non-invasive の 3 つのクラスに分類することができる．中でも Semi-invasive が最も現実的かつ効率的な手法と考えられており，我々はこのクラスの新たなメモリ抽出攻撃を提案する．UV light や電磁波照射，そして特にレーザーを利用するものがこのクラスに分類され, , 図 4.2 の Semi-invasive クラスの手法も全てレーザーを利用している．これらの手法を比較したものを表 4.1 に示す．従来の研究で論理的なエラーを注入して EEPROM の値を抽出する手法が提案されているが，このようなエラーは ECC

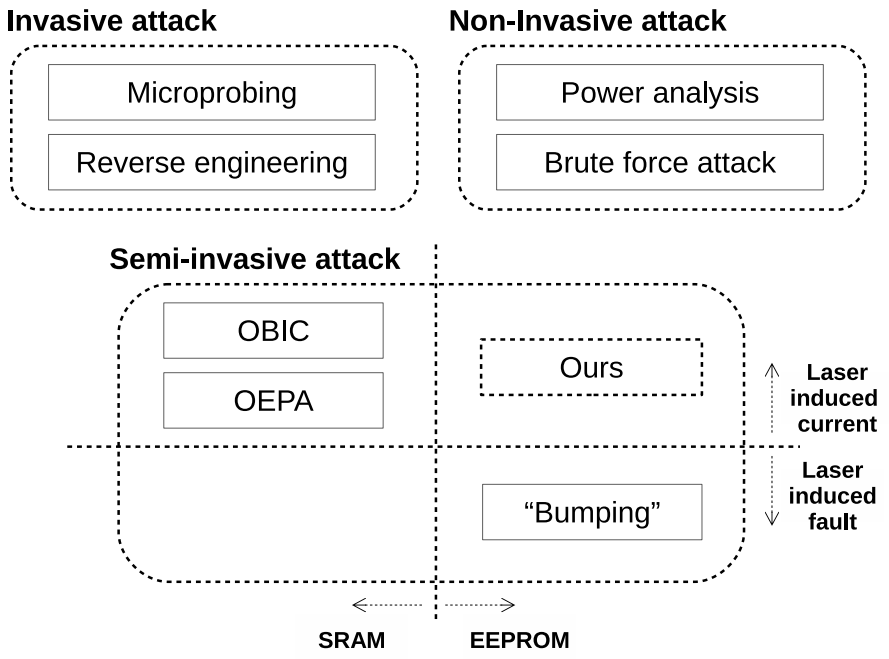


図 4.2: サイドチャネル情報を用いてメモリの値を抽出する手法の分類

表 4.1: Semi-invasive なメモリデータ抽出攻撃の比較

	OBIC [47]	OEPA [51]	“Bumping” [52]	Ours
No logical faults	Yes	Yes	No	Yes
Target memory	SRAM	SRAM	EEPROM	EEPROM
Target value	Cell’s state	Cell’s state	Contents	Read bits
Target location	Cell	Cell	Control logic	Sense amp.
Real-time measurement	Limited	Yes	No	Yes

や簡単な比較検証で検知可能であるから重大な脅威にはなりえない．よって我々は，論理的なエラーを注入することなく EEPROM のデータを抽出する手法を提案する．4.4 章では ATMega8515 マイコン上のフラッシュメモリを攻撃対象として，そこから逐次読み出されている値（命令フェッチされる機械語）を電力波形から取得した結果を示す．

Semi-invasive attack クラスの攻撃はレーザーを使うものが多く，Skorobogatov はレーザー照射によってメモリから読み出される値が 0/1 にマスクされることを利用した手法を提案している [52]．外部からのメモリ読み出しが実装されていない状況であっても，アップデートのための verification 機能は実装されていることが多く，これを利用してメモリ内容を総当りで取得することが可能である．これにはふつう膨大な時間がかかるため現実的ではないが，彼はレーザー照射によってフラッシュメモリから読み出される値を既知の値（すべて 1 もしくは 0）にすることで，2^{100} の探索空間を 2^{15} 程度にまで減らしている．しかしながら，上で述べたように論理的なフォールトを注入する手法は容易に対策可能であり大きな脅威とはなり得ない．一方で，論理的なフォールトを注入せずに SRAM セルの状態を取得する手法が提案されている [47, 51]．SRAM セルが消費する電力はふつう回路全体の消費電力に比べて小さすぎて見えないが，彼らはレーザーを使ってセルに流れる電流を増幅することで回路の消費電力からターゲットした SRAM セルの状態を取得することに成功している．

4.2　フラッシュメモリへのレーザー照射を用いた命令改変攻撃

先行研究において，多くのアーキテクチャ，フォールト注入手法で命令スキップモデルのフォールトが発生することが確認されている．一方で命令スキップを一般化した概念である命令改変モデルのフォールトは制御が難しいため攻撃に利用できないとされてきた．本節では，フラッシュメモリにレーザー照射を行うことで命令改変モデルのフォールト注入が可能である事を示す．

4.2.1　フラッシュメモリセルを狙ったレーザー照射による命令改変

実験環境

リスト 4.1 のようなプログラムを動作中の ATMega163 に対し，IPA-SLS を用いたレーザー照射実験を行う．リスト 4.1 は 2 つのレジスタの初期値 0x01 を 0xFF に上書きするプログラムである．4, 5 行目実行中にレーザー照射を行い，実行終了後のレジスタの値を本来の結果 0xFF と比較することでどのようなフォールトが起きたかを推測する．なお 4, 5 行目の実行タイミングを消費電力波形から識別しやすくするため，その上下に nop 命令を 100 個ずつ挿入している．レーザー強度は 0.85W，照射時間は 100 ns

に設定し，図 4.3 の破線で囲まれた約 2500 µm x 2000 µm の領域をレーザーで走査する．また表面を開封したチップに走査領域を照らし合わせた画像を図 4.4 に示す．この画像から，走査領域の下部にフラッシュメモリの一部が位置していることがわかる．

Listing 4.1: r24 と r25 の初期値 0x01 を 0xFF に上書きするプログラム

```
1 ldi   r24, 0x01
2 ldi   r25, 0x01
3 nop x100
4 ldi   r24, 0xFF
5 ldi   r25, 0xFF
6 nop x100
```

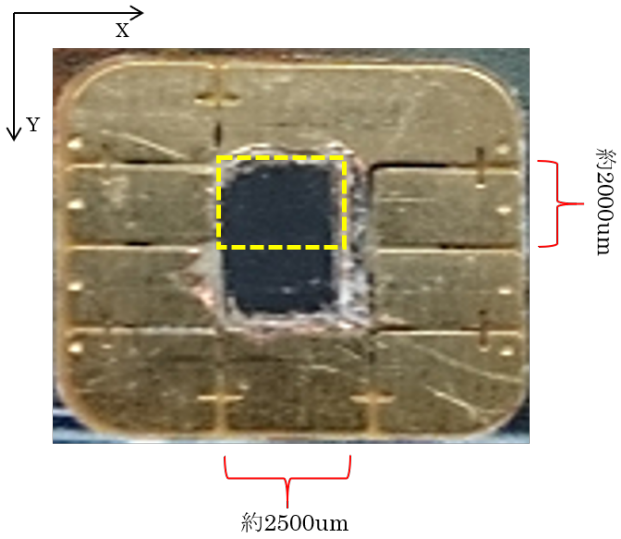


図 4.3: レーザースキャン領域

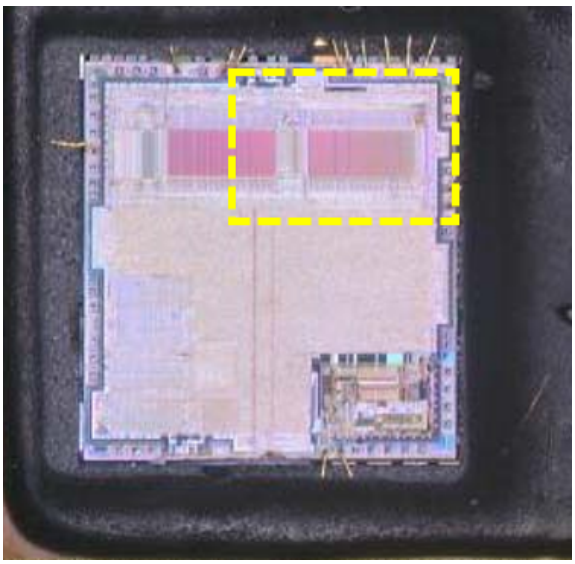


図 4.4: レーザースキャン領域下にある回路

実験結果

実験の結果得られた照射位置とフォールトの関係を図 4.5 に示す．正常にプログラムが動作した場合，2 つのレジスタ共に出力として 0xFF が得られるはずであるが，レーザーの照射位置を変えながらプログラムを実行しているため，照射位置によってさまざまな出力が得られた．この図から，同じフォールトがある程度固まった照射位置において得られることがわかった．

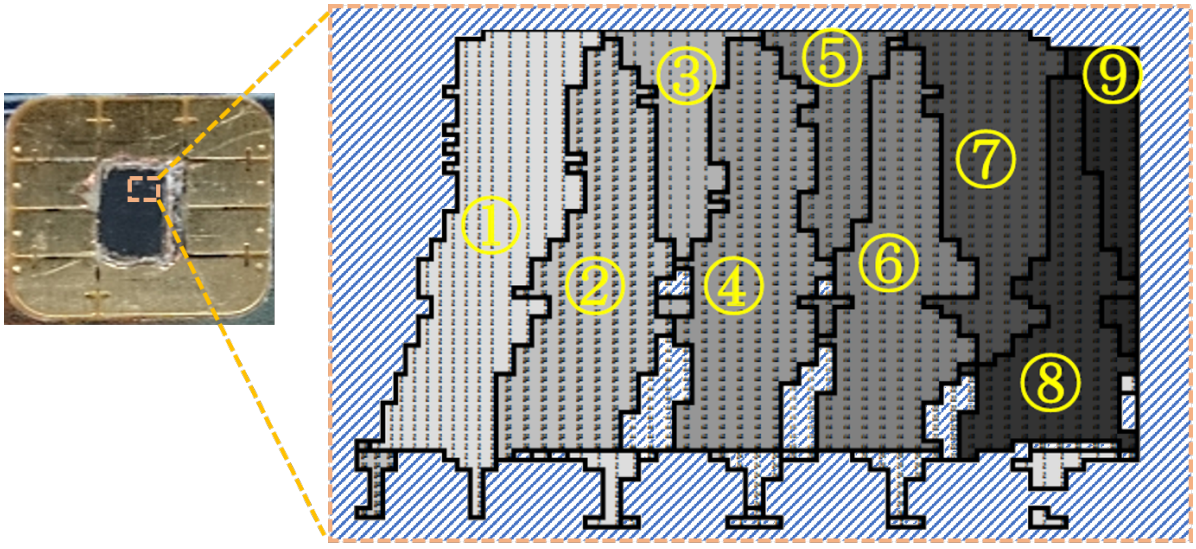
考察の結果，図 4.5 で得られたフォールトは実行される命令のある 1 ビットまたは隣接する 2 ビットが 0 に改変されて実行された結果だと考える事でうまく説明できることがわかった．AVR 命令セットでの ldi 命令の機械語の書式を図 4.6 に示す．この書式に従うと，リスト 4.1 の 4, 5 行目の命令は，

ldi r24, 0xFF: 1110 1111 1000 1111
ldi r25, 0xFF: 1110 1111 1001 1111

というマシンコードで実行されるはずである．ここで 12 ビット目と 13 ビット目が 0 にクリアされたとすると，

ldi r24, 0xF7: 1110 1111 1000 0111
ldi r24, 0xF7: 1110 1111 1000 0111

のような命令に改変される（下線部は改変されたビット箇所及びそれによって生じる命令改変を示している）．これらの命令が実行されると r24 は 0xF7 に設定され，r25 を操作する命令は実行されないため r25 は初期値 0x01 のままになる．図 4.5 中のフォールト②がこれに対応するため，②のフォールトが得られた照射箇所においては本来実行されるコードの 12, 13 ビット目がクリアされると考えることができ



	①	②	③	④	⑤	⑥	⑦	⑧	⑨
r24	0xFF	0xF7	0xF7	0xF3	0xFB	0xF9	0xFD	0xFC	0xFE
r25	0x01	0x01	0xF7	0xF3	0xFB	0xF9	0xFD	0xFC	0xFE

図 4.5: ATMega163 上のフラッシュメモリにレーザー照射した際に生じるフォールトと照射位置の関係

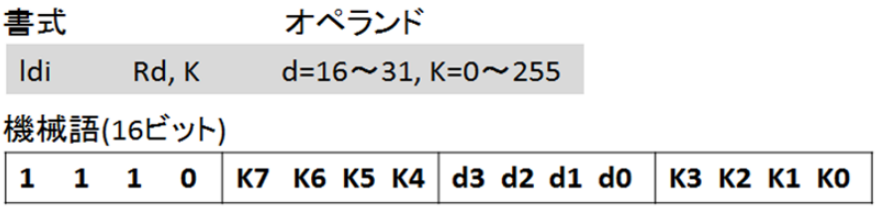


図 4.6: AVR 命令セット上の ldi 命令フォーマット

	①	②	③	④	⑤	⑥	⑦	⑧	⑨
クリアされたと考えられるビット位置	12	12, 13	13	13, 14	14	14, 15	15	15, 16	16

図 4.7: 各照射位置において 0 クリアされたと考えられるビット位置

る．同じように考えることで①から⑨までのすべての出力が説明できる．①から⑨について，それぞれどのビットがクリアされて実行されたと考えられるかを図 4.7 にまとめる．フォールト照射位置が右に行くに従って，クリアされたと考えられるビット位置も右にずれて（増加して）おり，フラッシュメモリのある列に照射することで，その列が保存しているビットに対応したビットがクリアされていると推測できる．また 1 ビットクリアされるものと 2 ビットクリアされるものが交互に現れていることがわかるが，これはフラッシュメモリの列の境界にレーザーを照射した際に，隣接する 2 ビットが影響を受けたためだと考えられる．このほか add 命令や cp 命令といった ldi 以外の命令実行中にレーザーを照射した場合でも，同じ照射位置で図 4.7 と同じ位置のビットがクリアされたときの結果が得られた．また ATMega8515 に対して IPA-SLS を使った場合，TVC に対して YNU-SLS を使った場合にも同様の結果が得られたため，フラッシュメモリセルを狙った命令改変は攻撃対象や実験装置によらずある程度一般性を持ったフォールトであることがわかる．

フラッシュメモリのセル上へのレーザー照射よる命令改変の原理

筆者らが 2015 年に命令改変の報告をした後 [Pub. 17]，2018 年によく似た命令改変攻撃の報告がなされている [18]．そこでは ARM Cortex-M3 コアが攻撃対象とされ，シリコン裏面からフラッシュメモリ上へレーザーを照射することで実行される命令の 1 ビットないし隣接する 2 ビットを 1 にセットすることに成功している．論文上ではフラッシュメモリセル上のレーザー照射による命令改変の原理が次のように説明されている．

図 4.8 にフラッシュメモリセルの模式図と，そこにレーザーを照射した際の影響を示す．フラッシュメモリはフローティングゲートトランジスタのマトリクスで構成されているが，図ではそのうちの 1 列だけを示している．フラッシュメモリがあるセルの値を読み出すとき，まず読み出しセルのビット線（BL）がプリチャージされる．その後ワード線（WL）がアサートされ，対象セルのフローティングゲートに電荷がたまっている場合はプリチャージされた電荷がビット線上に残り，電荷がたまっていない場合はプリチャージ電荷が読み出しセルを通じてグラウンドへと流れる．図 4.8 では左側のセルが読み出された際の状況が示されており，左側のセルには電荷がたまっているためプリチャージ電荷がビット線上に残らなければならない．ここで，読み出しセルと同じビット線上の別のセルにレーザーが照射されたとする．レーザーはトランジスタを短絡させるため，セルの状態によらずプリチャージ電荷をグラウンドへ逃がしてし

まう．この原理に従うと読み出し対象のセルに電荷がたまっていなかった場合は読み出し動作はフォールトしないため，片方向の誤りしか引き起こせないと考えられる．実験結果からも片方向のフォールトしか観測されていないためこの原理は実際のフォールトをよく説明できている．ATMega163 の実験結果でビットクリアしか観測されなかった一方で，Cortex-M3 の実験結果 [18] ではビットセットしか観測できなかったのは，電荷がたまっている状態を 1 と判断するか 0 と判断するかの違いによるものと考えている．またこのフォールトは読み出し対象のセル “以外” のセルにレーザー照射することでフォールトが発生するというのが大きな特徴である．従来のレーザー照射攻撃は，特定のトランジスタをフォールトさせるためにごく限られた領域を狙う必要があったが，この原理ではむしろ広い領域に当てた方がフォールトしやすく，攻撃にかかる所要時間や TOE 知識といったコストが小さいといえる．

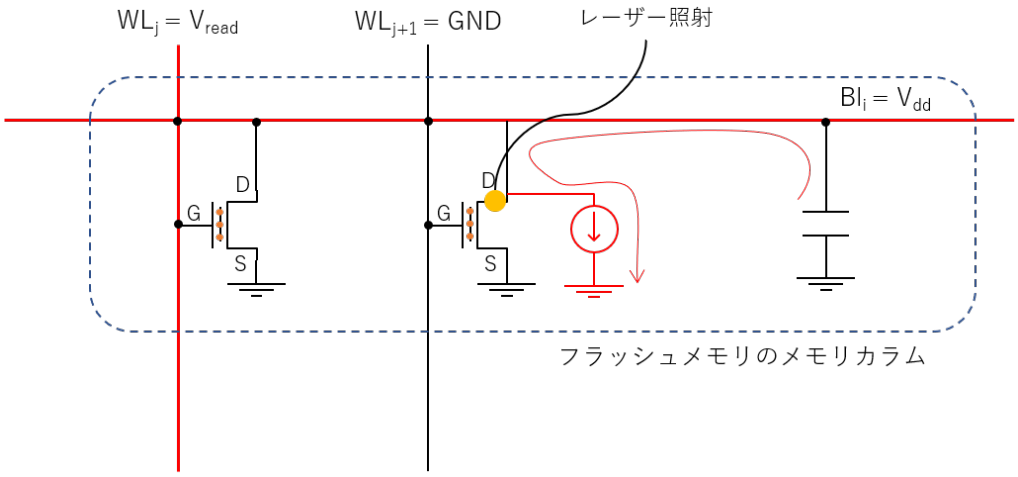


図 4.8: フラッシュメモリセル上のレーザー照射による命令改変の原理

4.2.2　センスアンプを狙ったレーザー照射による命令改変

攻撃原理

前項で紹介したセル上へのレーザー照射による命令改変は，片方向にしかフォールトを注入できないという制限があった．本研究では，フラッシュメモリ上のセンスアンプへレーザーを照射することによる命令改変攻撃を提案する．本手法は照射位置によってビットセット/クリア両方向の改変が可能である．

提案手法の説明に入る前に，フラッシュメモリがどのように値を読み出すか解説する．デバイス上に実装されたフラッシュメモリの構造を図 4.9 に示す．このフラッシュは w ビットのバスに接続されており，命令長は w ビットである．この場合フラッシュメモリのセルアレイは w 個の列に分割されていることが多い．フラッシュに格納されたプログラムのあるビット位置が対応する列に格納されている．フラッシュの読み出し動作はまずワード線（WL）及びビット線（BL）をアサートし，各列から読み出すセルを選択するところから始まる．選択されたセルが電荷を保持していた場合，その電荷がゲート電界によるチャネル生成を妨げるためセルに電流は流れず，プリチャージされている電荷は BL 中に残る．一方で選択されたセルが電荷を保持していない場合，通常のトランジスタと同じようにドレインからソースへチャネルが生成され BL にプリチャージされた電荷はセルを通って GND へ流れる．BL に残った電荷はセンスアンプで増幅され，流れた電流量によってセルに電荷が貯まっていたかどうかを判断する．典型的なセンスアンプは電流電圧 (I/V) 変換回路及び比較器で構成されており，比較器は I/V で変換された電圧をリファ

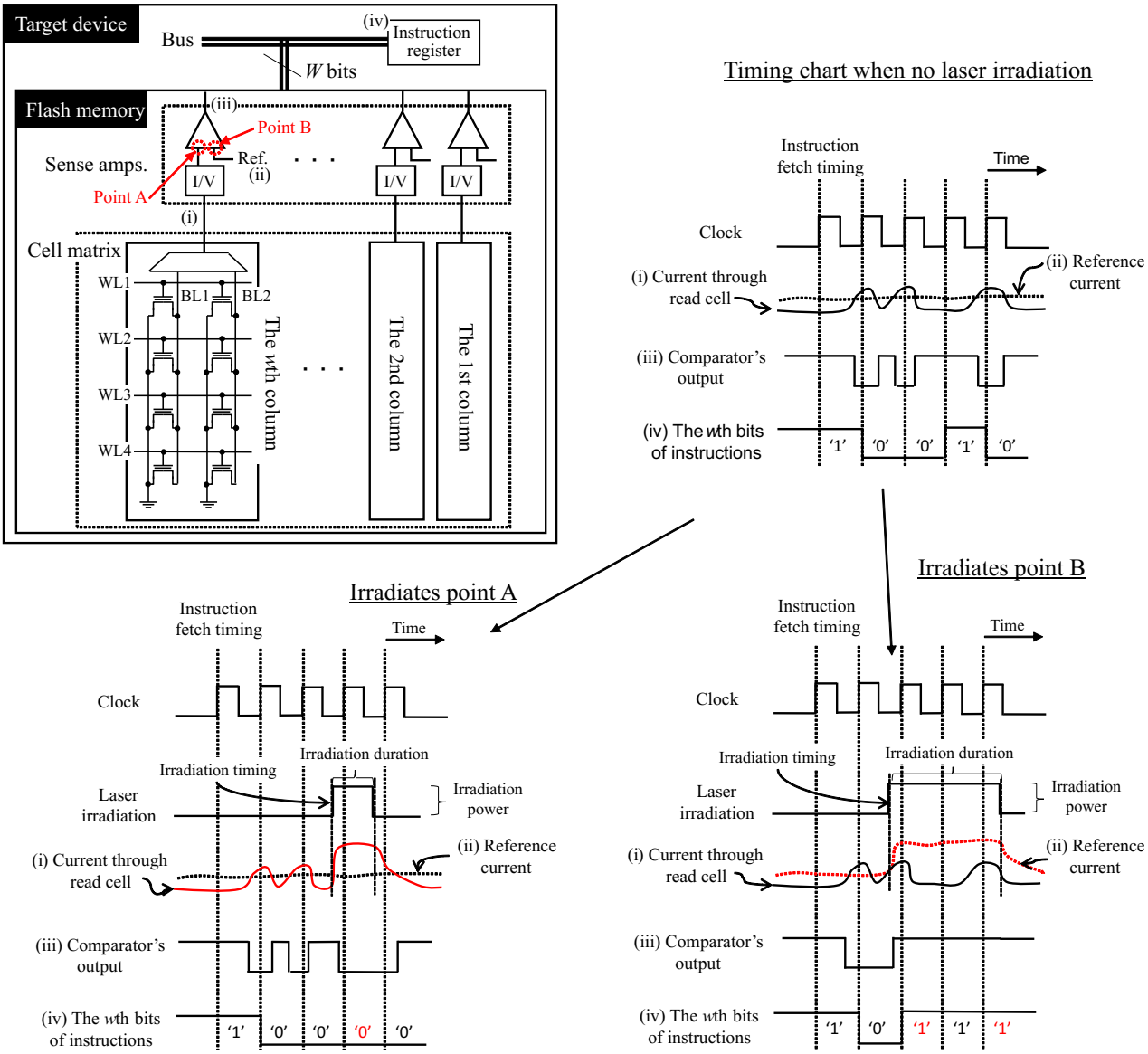


図 4.9: フラッシュメモリの構造と，レーザー照射時のタイミングチャート

レンス電圧と比較して 0/1（BL に電荷が残っているかどうか）を判定する．比較器が ‘1’ を出力する場合リファレンス電圧の方が大きく，‘0’ を出力する場合は増幅された電圧の方が大きい．比較器は差動増幅器として実装されている場合が多く，微弱な電圧を 100 倍程度増幅し論理回路で利用できるレベルにする．このような増幅効果があることから，センスアンプへのレーザー照射は，論理回路上へのレーザー照射よりも敏感であることが予想される．

続いて，どのように提案手法が命令改変を注入しそれを制御するのか説明する．図 4.9 の右上の図はレーザー非照射時のセンスアンプ動作のタイミングチャートを示している．ここではビット列 “10010” が w 番目の列から読み出されている．このビット列は実行される命令列の w ビット目に対応している．

図中のポイント A（w 番目の比較器の入力部分）にレーザーを照射した状況を考える．このときのタイミングチャートを図 4.9 の左下に示す．4 番目のクロックの立ち上がり直前でレーザー照射が開始されており，5 番目の直前で照射が終わっている．レーザー照射によって生成された電子正孔対は，セルに流れる電流を増加させることから，比較器の入力電圧がリファレンスよりも大きくなり，間違った値を出力す

るようになる．比較器出力の誤りが命令フェッチのタイミングに被ってしまうと本来 ‘1’ と判断されるべきビットが ‘0’ という誤った値として命令レジスタにラッチされる．4 サイクル目のフェッチタイミングにレーザー照射時間がかぶっていた場合には 4 サイクル目の命令が改変されることから，提案手法はレーザー照射タイミングによってどの命令を改変するかを制御可能である．

次に図中のポイント B（w 番目の比較器のリファレンス入力）にレーザーを照射した際のことを考える．このときのタイミングチャートを図 4.9 の右下に示す．今回のレーザー照射は 3 番目のクロックの立ち上がり直前から開始されており，5 番目のクロックの立ち上がり直後に終わっている．この場合のレーザー照射はリファレンス電圧を増加させ，従って今回の例では本来 ‘0’ のはずの値が ‘1’ として読み取られる．また今回は複数クロックに渡ってレーザー照射が行われており，最終的に得られるビット列は ‘110111’ となる．レーザー照射が 3 クロックに渡っている一方でフォールトした値は 2 カ所だけであり，当然だが元から 1 の場所は 1 に改変されてもフォールトとはならない．このように提案手法は，レーザー照射時間を延ばすことで改変する命令の数を制御することができる．

さらに，n 番目のセンスアンプを狙うことで命令の n ビット目を改変することができ，またセンスアンプの参照電圧/セル電圧を狙うことで改変方向を選ぶことができる．もしも攻撃者が複数スポットを照射することができるなら攻撃者はある命令の複数ビットを改変することができるため，レーザーのスポット数は攻撃者の能力を制限する上で重要なファクターとなる．

攻撃者が命令 Ins_{src} を Ins_{dst} に改変したい場合を考える．Ins_{src} を Ins_{dst} に改変するために必要なレーザーのスポット数 N は次の式

$$N = HD(bin(Ins_{src}), bin(Ins_{dst})) \tag{4.1}$$

で表される．ここで $HD(,)$ は 2 つの引数のハミング距離を返す関数であり，$bin(ins)$ は ins のバイナリ表現である．N が大きいほど攻撃者の能力は上がるが，それに準じて攻撃コストも増加することが予想される．2 章で解説したように，現在市販されている市販のレーザー装置は高々 $N = 1$ もしくは $N = 2$ あるため，現時点では大きな N での攻撃は現実的でない．

まとめると提案手法は

1. 照射位置によって命令中の何ビット目を改変するか，及び 0/1 のどちらに改変するかを制御
2. 照射時間によってフォールトを注入する命令の数を制御
3. 照射時刻によってどの命令を改変するかを制御

することが可能である．以下の実験では，YNU-SLS 及び YNU-DLS を用いて $N = 1$ 及び $N = 2$ の際の提案手法の実現結果を示す．

実験結果

YNU-SLS を使い，センスアンプを狙ったレーザー照射による命令改変を TVC 上で実証する．レーザー照射のための 4 つのパラメータ（照射位置，照射強度，照射時間，照射時刻）が命令改変にどのように影響するかを実験によって確かめる．

■照射位置の影響　図 4.10 は TVC 上のフラッシュメモリの構造を示しており，フラッシュメモリはこのような規則的かつ大きな構造を持っていることから事前知識がなくとも回路上の位置の特定が容易である．TVC のフラッシュメモリは 2 つのメモリプレーンから構成されているが，図では一枚のプレーンのみを示している（実際には画像の上にもう一枚同様のプレーンが存在する）．各プレーンは 32 の列を持っており，ARM が 32-bit アーキテクチャである事との関連が予想される．列デコーダ付近にあると予想されるセンスアンプにレーザーが照射され，どの照射位置によってどのようなフォールト結果が得られるかを調査する．

リスト 4.2 が TVC のフラッシュメモリに保存されている．このプログラムは MOV 命令によって即値 0x00 を 6 つのレジスタ（R5–R10）に転送するものであり，正常動作時には全てのレジスタが 0x00 になっているはずである．例えば，もしプログラム実行後に 6 つのレジスタの値が 0x01, 0x01, 0x01, 0x00,

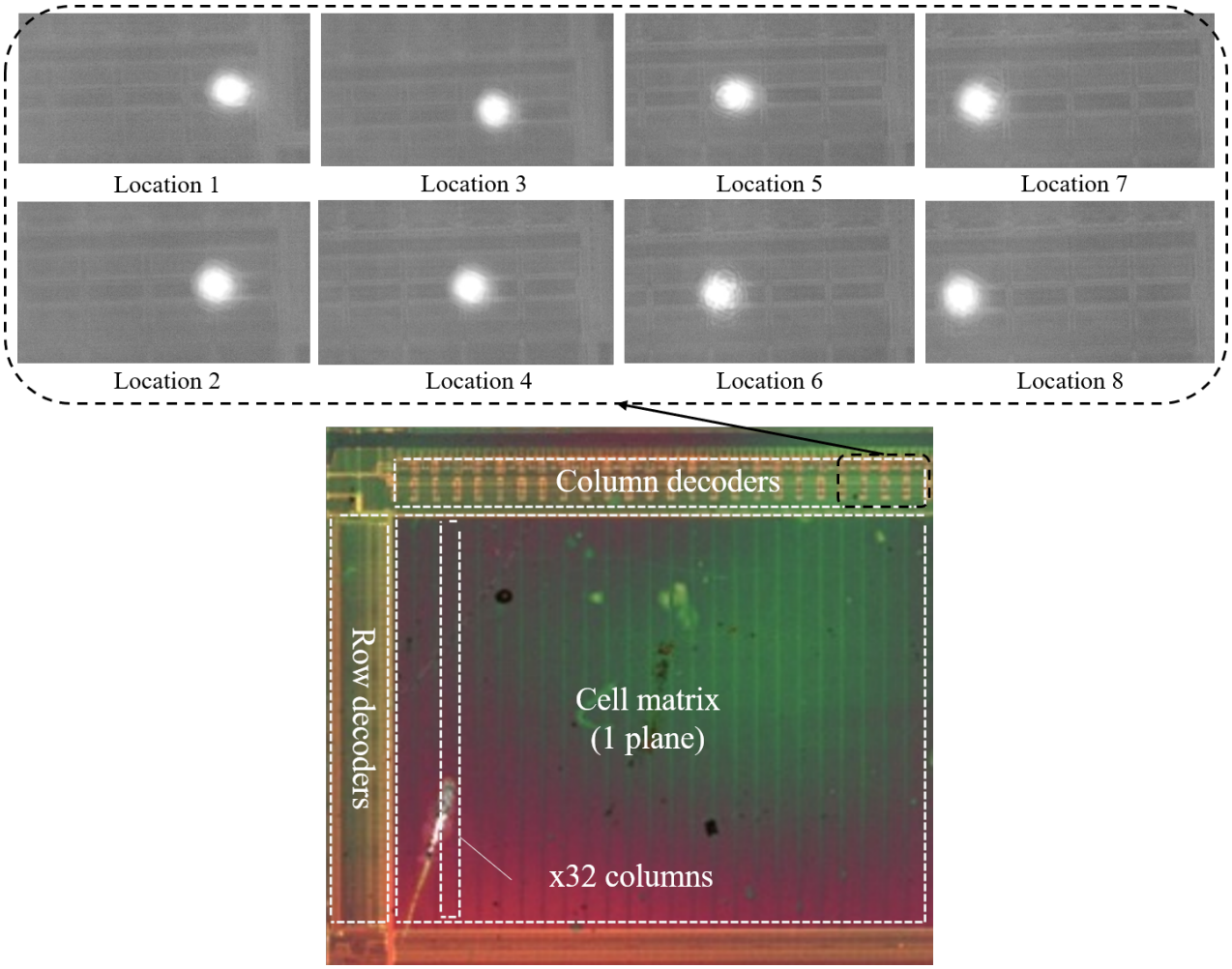


図 4.10: TVC のフラッシュメモリ構造と，列デコーダ上の 8 つの照射位置

0x00, 0x00 になっていたとすれば，最初の 3 つの MOV 命令の即値部分の 1 ビット目が ‘1’ に改変されたのだと推測することが可能である (MOV 命令のフォーマットについては ARM Reference Manual[4] を参照).

Listing 4.2: 照射位置の影響を確認するためのプログラム．値 0x00 をレジスタ R5 から R10 に格納する．

```
1 Initialize registers R5 to R10 with 0x00.
2 NOP x20
3 MOV R5, 0x00     ;0xe3a05000
4 MOV R6, 0x00     ;0xe3a06000
5 MOV R7, 0x00     ;0xe3a07000
6 MOV R8, 0x00     ;0xe3a08000
7 MOV R9, 0x00     ;0xe3a09000
8 MOV R10, 0x00    ;0xe3a0a000
9 NOP x20
10 Output R5 to R10
```

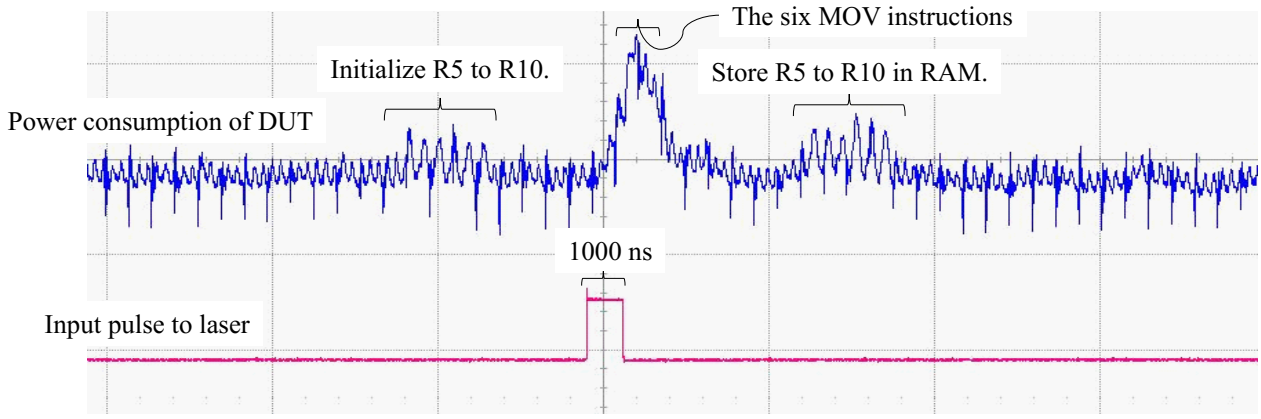


図 4.11: レーザー照射時の TVC の消費電力

図 4.11 はセンスアンプ上にレーザーを照射した際の TVC の消費電力を示している．このときレーザー強度は 600 mV，照射時間は約 9 サイクルに相当する 1000 ns に設定されている．照射タイミングは MOV 命令群（リスト 4.2 の 3 行目）の実行前に設定されており，全ての MOV 命令実行中にレーザー照射時間がかぶるようになっている．レーザー照射直後に消費電力が急上昇しており，MOV 命令の実行に何らかの影響を与えていることがうかがえる．図 4.10 に示した 8 つの照射位置に対する出力結果を表 4.2 に示す．正常実行時の値 0x00 と比べて，3 つのレジスタ R5, R7, R9 の値は 1 ビット異なっており，照射点ごとに誤りビットの位置が異なっている．この結果はセンスアンプのレーザー照射により実行される命令の機械語のある 1 ビットが ‘1’ に改変された事を示しており，また照射位置によってどのビットを改変するかを制御できることを示している．例えば，照射位置 1 は 32 ビット機械語のうちの 5 ビット目を ‘1’ に改変している．R5, R7, R9 にフォールトが注入されている一方で，図 4.10 の 8 つの照射位置では R6, R8, R10 にはフォールトが注入されなかった．他の照射位置に対しても同様の実験を行うことで，R6, R8, R10 にだけ影響する照射位置が存在することがわかり，命令の格納されているアドレスの奇数/偶数によっても照射位置が異なることがわかった．

続いて，同様の実験が別のプログラム（リスト 4.2 の即値 0x00 箇所を 0xFF に変更したもの）に対しても行われた．照射位置 1 から 8 に対しては何のフォールトも得られず，これはこの位置のフォールトが

表 4.2: 8 つの照射位置で得られたフォールト値

Location	R5	R6	R7	R8	R9	R10
1	0x10	0x00	0x10	0x00	0x10	0x00
2	0x20	0x00	0x20	0x00	0x20	0x00
3	0x80	0x00	0x80	0x00	0x80	0x00
4	0x40	0x00	0x40	0x00	0x40	0x00
5	0x01	0x00	0x01	0x00	0x01	0x00
6	0x02	0x00	0x02	0x00	0x02	0x00
7	0x08	0x00	0x08	0x00	0x08	0x00
8	0x04	0x00	0x04	0x00	0x04	0x00

ビットフリップではなく単にビットセットするだけである事を示している．一方で，照射位置 1 から 8 の少し上の位置において，ビットクリアとなる照射位置が確認された．このような実験を繰り返すことで，32bit 全てのビットに対して 1 もしくは 0 になる照射位置を特定することができた．また MOV 命令以外の別の命令 (ALU, メモリ, 分岐命令等) に対しても，MOV 命令の時と同じ照射位置で同じ位置のビットが改変できることを確認した．

図 4.12 はこれらの結果をまとめたものである．ここで照射位置は X 座標及び Y 座標の組 (X, Y) として表現されている．図のように，右 16 列の列デコーダ上に対するレーザー照射は偶数アドレスの命令を改変し，一方で左 16 列は奇数アドレスの命令を改変することができた．このような結果になった理由はバススクランブルによるフォールト攻撃対策が考えられるが，本質的な対策とはなっていない．従ってフラッシュメモリの読み出し動作高速化のために 2 命令を異なる場所から同時フェッチしているためだと考えている．左右の 16 列にはそれぞれレーザーに対してセンシティブな点（1 に改変する照射位置 32 箇所及び 0 に改変する照射位置 32 箇所）が 64 箇所ずつあった．次の説明では右の 16 列のみにフォーカスするが，左右の 16 列でパターンは同じであるため，左の 16 列についても同じ事がいえる．

XY 座標で表現されるレーザー照射位置に対して，Y 座標は改変の方向がセットなのかクリアなのかを決定しており，$Y = 0$ の位置ではビットクリアが生じることを，$Y = 1$ の位置ではビットセットが生じることを示している．照射位置の X 座標は 32 ビット命令のうちの何ビット目にフォールトを注入するかを決定しており，$X = 5$ の位置では 5 ビット目が改変されることを示している．

■照射強度による影響　照射強度による影響を調査するため，リスト 4.2 の修正版（0x00 が 0xFF に変更されたバージョン）がフラッシュメモリに格納された．ある照射強度において座標 (8, 0)[*1]に 100 回レーザー照射を行い，照射強度ごとに注入されるフォールトが変化するのかを確認する．照射時間は 500 ns，照射時刻は 6 つの MOV 命令の実行直前で固定した状態でレーザーへの供給電圧を 360 mV から 600 mV まで 10 mV ずつ変化させる．

図 4.13 はレーザー強度とそのレーザー強度の時に得られたフォールト値の頻度の関係を示している．フォールト未注入時の値は R5 から R10 までそれぞれ 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF となる．図からわかるとおり，3 通りのフォールト値 (1) 0x7F 0xFF 0xFF 0xFF 0xFF 0xFF, (2) 0x7F 0xFF 0x7F, 0xFF, 0xFF, 0xFF 及び (3) 0x7F 0xFF 0x7F, 0xFF, 0x7F, 0xFF が観測された．座標 (8, 0) へ

[*1] 8 ビット目を 0 に改変する照射位置

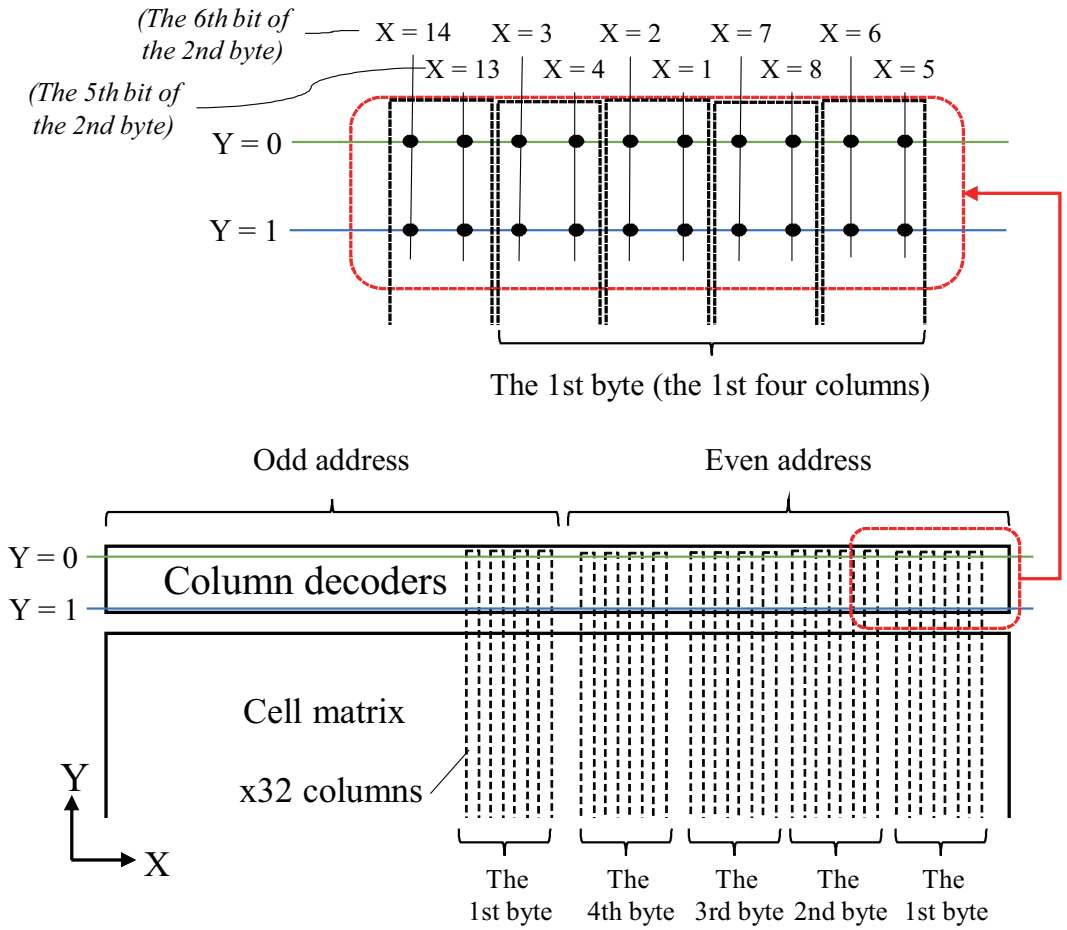


図 4.12: レーザー照射位置とフォールトが注入されるビット位置の関係．座標 (X, Y) へのレーザー照射は 32 ビット命令の X ビット目を Y に改変する．照射位置とフォールトが注入されるビットの位置は 1 バイトごとにパターンがあり，図中では最下位バイトのみを示している．

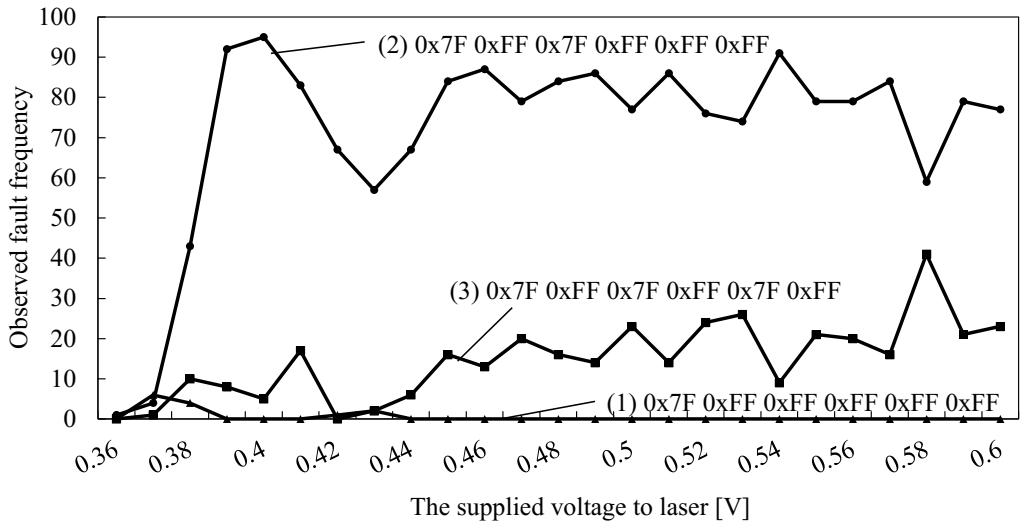


図 4.13: レーザー強度と注入されるフォールトの関係

のレーザー照射は実行される機械語の 8 ビット目を 0 に改変するため，命令 MOV regX, 0xFF は MOV regX, 0x7F のように変化する．フォールト値に 0x7F が現れているのはこのためである．3 つのフォールト値全てで 0x7F が連続して現れないのは座標 (8, 0) へのレーザー照射が偶数アドレスの命令にしか影響しないためである．

レーザー強度が強くなるに従い (3) の出現率が多くなっており，合わせて (1) 及び (2) の出現率は減少傾向にある．この結果はレーザー強度が強くなるほど，レーザー照射の影響が長く続くことを示している．またフォールト値 (2) の出現率が 370 mV を超えた時点で急激に上昇しており，命令改変の注入にはある一定の閾値以上のレーザー強度が必要だと考えられる．

■照射時間による影響　レーザー照射時間による影響を調査するため，リスト 4.3 が TVC 上のフラッシュメモリに書き込まれた．このプログラムは値 0x01 を R2 に 255 回加算するものであり，正常動作時には R2 の値は 0xFF になる．プログラム実行中に座標 (1, 0) の位置にレーザーを照射する．ADD 命令の 1bit 目は加算する即値の 1bit 目であるから，この位置のレーザー照射によって ADD R2, R2, 0x01 命令は ADD R2, R2, 0x00 命令に改変される．照射時間を長くするほどより多くの命令が改変される事が期待されるため，照射時間を長くするほどプログラム実行終了時の R2 の値が小さくなるはずである．レーザーには 600 mV が供給され，照射時間を 250 から 5000 ns まで 250 ns（約 2 サイクル）ずつ増加させる．クロックジッタ等の影響により照射タイミングにブレが生じるため，各照射時間で 100 回実験を行い，それぞれの実験終了後の R2 の値の平均値を代表値としてグラフにプロットする．

実験結果を図 4.14 に示す．照射時間 250 ns 及び 500 ns において値 255 が観測されており，これらの照射時間の時レーザー照射時間が ADD 命令に達しておらず命令改変がなかったことを意味している．ADD 命令の直前には NOP 命令が配置されているが，NOP 命令の 1 ビット目は 0 であるから座標 (1, 0) へのレーザー照射によって改変されない．照射時間 750 ns 以上の時，プログラム 4.3 実行後の R2 の値は徐々に線形に減少して行っており，250 ns ごとに概ね 1 ずつ減少している．250 ns というのは約 2 サイクル分に相当するが，(1, 0) へのレーザー照射は偶数アドレスの命令にしか影響しないためこのような結果となる．

Listing 4.3: 照射時間の影響を確認するためのプログラム．R2 に値 0x01 を 255 回足す．

```
1 MOV R2, 0x00          ;0xe3a02000
2 NOP x20
3 ADD R2, R2, 0x01      ;0xe2822001
4 | x256 times
5 ADD R2, R2, 0x01      ;0xe2822001
6 NOP x20
7 Output R2
```

■照射時刻による影響　照射時刻による影響を調査するため，リスト 4.2 のプログラムが TVC に書き込まれた．このプログラムは正常似動作した際 0x00 0x00 0x00 0x00 0x00 0x00 を出力する．この実験では照射時間を 1 クロックサイクルよりも短い 50 ns に固定し，命令改変が 2 命令以上に生じないように設定する．照射位置を (1, 1)，照射強度 600 mV に固定し，このとき命令改変により 0x01 と言う値が観測されると期待される．

照射時刻 t_0 においてレーザーを照射した際，フォールト出力 0x01 0x00 0x00 0x00 0x00 0x00 が観測された．t_0 から照射時刻を 10 ns ずつ増加させていくと，3 つのフォールト出力 (1)0x01 0x00 0x00 0x00 0x00 0x00, (2)0x00 0x00 0x01 0x00 0x00 0x00, (3)0x00 0x00 0x00 0x00 0x01 0x00 が時刻 t_0 か

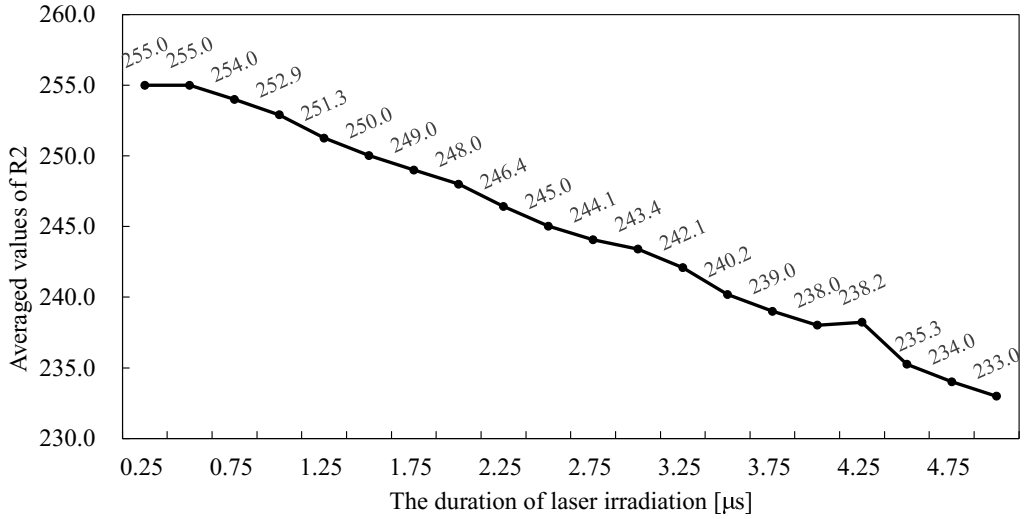


図 4.14: 照射時間と命令改変数の関係

ら $t_0 + 50$ ns まで，$t_0 + 200$ から $t_0 + 250$ ns まで，そして $t_0 + 400$ から $t_0 + 450$ ns まででそれぞれ観測された．このように照射時刻を変化させることで，命令改変する対象の命令を制御することが可能である．

考察

■フォールト注入可能な命令数（照射時間）の制御性　提案手法であるレーザー照射による命令改変は，レーザー照射のタイミングが命令フェッチのタイミングと重なっている際に誤りを注入する．したがって照射時間を延ばすことで，複数の連続する命令にフォールトを注入し続けることが可能である．強力なレーザーを連続で発振するためには放熱を十分に行うなどの対策が必要になるが，連続発振可能な CW レーザーが市販されているように長い時間レーザーを照射し続けるのは比較的容易である．一方で，照射可能なパルス幅の下限には制限がある場合が多い．短パルスを照射可能なレーザーはその分高価になる傾向にあり，レーザー照射装置のコストに制限のある攻撃者はフォールト注入可能な命令数に下限があると考えられる．フラッシュメモリのフェッチ周期を T_{fetch}，攻撃者が照射可能な最小のパルス幅を T_{pulse_min} としたとき，フォールト注入可能な最小の命令数 N_{ins_min} は

$$N_{ins\_min} = \begin{cases} \lceil \frac{T_{pulse\_min}}{T_{fetch}} \rceil & (0 \leq T_{pulse\_min} \leq T_{fetch}) \\ \lceil \frac{T_{pulse\_min}}{T_{fetch}} - 1 \rceil & (T_{fetch} < T_{pulse\_min}) \end{cases} \tag{4.2}$$

と表せる．

■どの命令を改変するか（照射時刻）の制御性　ある命令を狙ってフォールトを注入するためには，対象の命令が実行されたタイミングで正確にレーザー照射を行わなければならない．レーザー照射タイミングは外部からのトリガによって決定されるため，タイミングの精度はトリガ生成器の精度に依存する．数万円程度の安価な FPGA ボードでも数百 MHz 程度のクロックで動作するものが市販されており，トリガ生成の精度は容易に担保できると考えている．より困難なのは対象の命令が実行されているタイミングをどのように知るかということである．このような場合、攻撃者は攻撃対象の消費電力を観察するなどして現在実行中の命令を推測することが可能である．特に分岐命令のような重要な命令は分岐の可否で消費電力が大きく変化するため，容易に実行タイミングを特定することが可能である．また，対象となる命令の

実行タイミングが正確にわからずとも，攻撃者は照射タイミングを総当たり的に走査するというアプローチをとることも可能である．どの命令に照射されているか攻撃者がわからずとも，一度でも攻撃対象命令に照射できれば攻撃が成功するという状況がいくつか（秘密情報を直接出力させる攻撃など）存在する．

■提案手法の適用範囲　提案手法はあるデバイスの中のフラッシュメモリにレーザー照射するものである．組込みデバイスのプログラムメモリはそのランダムアクセスのしやすさから NOR タイプのフラッシュメモリが利用される場合が多く，NOR タイプフラッシュをプログラム格納領域として実装しているデバイスは提案手法で攻撃可能だと考えている．ARM Cortex-A シリーズのようなアプリケーションプロセッサはチップ上に ROM 領域を持たず外部に ROM を実装しているが，このような場合も攻撃対象になり得ると考えている．スマートカードのような省電力が要求されるデバイスではオンチップに ROM を内蔵した組込みプロセッサが利用される場合が多く，提案手法で攻撃可能であると考えている．また NOR タイプのフラッシュメモリは微細化の限界に近づいてきたことが示唆されており，45nm より小さいプロセスのロードマップが白紙になっているものもある [19]．一般的にレーザー照射攻撃はプロセスの微細化によって困難になると言われているが，提案手法にはこの効果が働かない恐れがある．

4.2.3　ダブルレーザー装置による命令改変攻撃

ここまでの実験で，TVC は奇数/偶数アドレスに配置された命令を別々の場所から読み込んでいることがわかった．従って，命令を 2 重化することでどちらかの命令は必ず実行され，それが対策になる可能性がある．しかしながら攻撃者が 2 以上のスポットを同時に照射可能なレーザー装置を利用できる場合このような対策はとれない．奇数/偶数アドレス読み出しのそれぞれの場所にレーザーを照射することで連続する命令にもフォールト注入可能だと考えられる．YNU-DLS 装置を使って奇数/偶数アドレスの 13bit 目を改変したときの様子を図 4.15 に示す．ダブルレーザー装置は二つの照射位置を同時に照射できており，これらの照射間隔をずらすことで連続する命令にもフォールト注入可能であることを確認した．また攻撃者がダブルレーザーを利用可能な場合，ある命令の任意の 2 ビットを改変可能であることも実験で確認した．

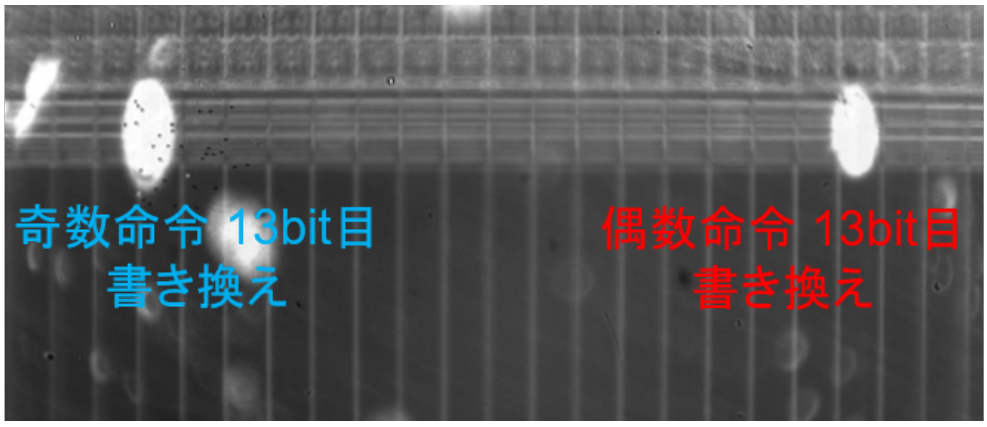


図 4.15: ダブルレーザー装置による照射の様子

4.3　命令改変を利用した攻撃

4.3.1　命令スキップ攻撃対策に対する命令改変攻撃

これまでのソフトウェアに対するフォールトモデルは，攻撃者がある命令をスキップすることができるという命令スキップモデルが一般的だった．よって命令スキップに対する対策手法がいくつか提案されているが，それらは命令改変モデルに対する耐性を考慮していないため，命令改変によって簡単に攻撃できる可能性がある．ほとんどの命令スキップ対策はソフトウェアに何らかの冗長性を持たせることによって攻撃を防いでおり，図 4.16 に示すものはアルゴリズムレベル冗長性による対策の代表的な例である．

この対策は，暗号化した結果を復号し，それを元の平文と比較することでフォールトを検知する．DFA を行う攻撃者は誤り暗号文が必要となるから暗号化時にフォールトを注入しており，誤り暗号文を復号したものは元の平文と一致しないためフォールト注入を検知できる．しかしながら，このような検算手法を単純に実装してしまうと命令スキップに対して脆弱である事が知られている．図 4.16 (a) がその単純な実装であり，5 行目の分岐命令をスキップすることで攻撃者は検算処理をバイパスし誤り暗号文を得ることが可能である．遠藤ら [24] によって上記のような命令スキップ攻撃に対する実装方法が提案されており（図 4.16 (b)），この実装は分岐命令がスキップされた場合エラー処理に至るため命令スキップに対してセキュアだとされている．

図 4.16 (c) は，遠藤らの命令スキップ対策を命令改変で攻撃する例である．1 ビットの改変によって BREQ 命令 (比較した値が等しいときに分岐する命令) を BRNE 命令 (比較した値が等しくないときに分岐する命令) にすることができ，このような攻撃はソフトウェア上の制御フローを完全に書き換えることができるため非常に強力である．TVC 上に図 4.16 (b) を実装し，YNU-SLS によって上記の命令改変攻撃を実証した結果を図 4.17 に示す．図 4.17 (a) はレーザーが照射されていないときの消費電力であり，検算後に暗号文が出力されているのがわかる．図 4.17 (b) は暗号化中にレーザーを照射し，暗号文に誤りを注入した際の消費電力である．このとき検算によってフォールトが検知されており，暗号文出力関数が実行されていない．図 4.17 (c) は暗号化中及び BREQ 命令実行中にレーザーを照射し，BRNE 命令に改変した際の消費電力である．これは図 4.17 (a) の波形と似ており，暗号文出力が行われていることがわ

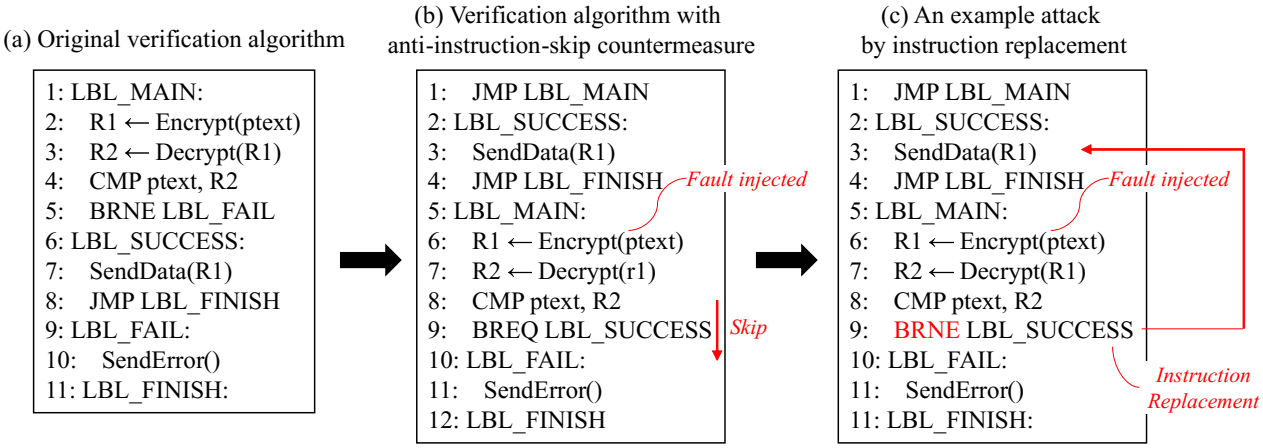


図 4.16: アルゴリズムレベル冗長性を使った命令スキップ対策であるデフォルトフェイル付き検算 [24] を命令改変で攻撃する例．検算対策は命令スキップには有効だが，命令改変に対しては脆弱である．

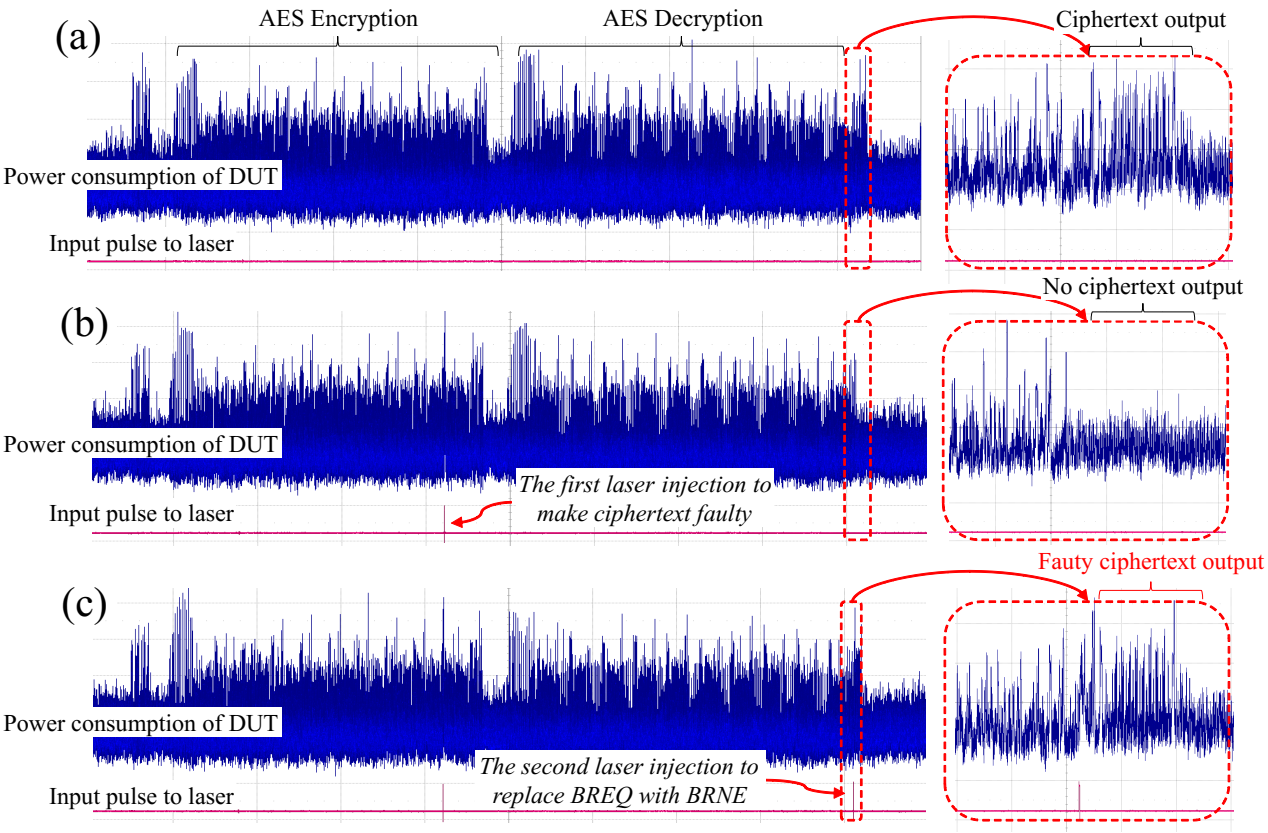


図 4.17: 命令スキップ対策を命令改変で攻撃した際の消費電力波形．(a) フォールト未注入時の波形．AES 暗号化及び復号の後、正しい暗号文が出力されている．(b)AES 暗号化中にフォールトが注入されたときの波形．フォールトが検知されたため暗号文は出力されていない．(c)AES 暗号化中にフォールトが注入され，かつ検算部分の BREQ 命令が BRNE 命令に改変された際の波形．暗号化中にフォールトが注入されているにもかかわらず誤り暗号文が出力されている．

かる．また実際に出力暗号文に誤りが含まれていることも確認し，従来の命令スキップ対策は命令改変に対して安全はないことが実証された．

4.3.2　ペアリング暗号に対する命令改変攻撃

公開鍵暗号へのレーザーフォールト攻撃の適用例として，TVC 上にソフトウェア実装されたペアリング暗号を命令改変攻撃した結果を示す．関連する先行研究としては，まず 2014 年に Lashermes らが，Cortex M3 上に実装したペアリング演算に対して電磁波照射 によるフォールト解析実験を行っている [32]．彼らは電磁波照射による命令スキップによって [23] の手法で解析を行っているが，彼らの攻撃対象ペアリングには最終ベキ部分が実装されておらず，最終ベキを含めた実装の攻撃を今後の課題としている．同年に Blomer らは最終ベキを含めたペアリング実装に対するフォールト攻撃を行っている [13]．彼らは AVR XMEGA マイコン上に実装されたペアリング演算に対してクロックグリッチによって命令スキップフォールトを注入している．彼らはまずミラーループ実行中に命令スキップを引き起こして中間値に誤りを発生させ，続いて最終ベキ関数への分岐命令をスキップすることで最終ベキ関数の影響を無効化している．このように，ペアリング演算に対するフォールト攻撃の研究は命令スキップモデルを前提としたものが多い一方で命令改変モデルを対象にしたものは報告されていない．またペアリング演算の結果は

ハッシュ関数を通して暗号として利用される場合が多い．ハッシュ関数を通した後の値ではフォールト解析を実行できない [17] ため，ペアリング暗号を攻撃する際はペアリング演算部の攻撃に加えてハッシュ関数部の攻撃も必要になると考える．

これらのことより，ペアリング暗号の攻撃には (1) ミラーループ部，(2) 最終ベキ部，(3) ハッシュ関数部への 3 つのフォールト注入が必要なことがわかる．これまでの研究では (1) と (2) へのフォールト注入が命令スキップによって達成されている．本研究では，命令改変によっても (1) と (2) へのフォールト注入が可能であることを示し，更に (3) の影響を受けない攻撃が命令改変によって可能であることを示す．

実験環境

レーザー照射装置として YNU-SLS 装置を用い，攻撃対象は TVC とする．TVC はユーザー領域として 64kB の ROM, 4kB の RAM を利用可能だが，特に RAM の制限のため公開されているペアリングの実装は動作しないものが多く，RAM を極力利用しないペアリングをコーディングすることにした．TVC に搭載されている RSA 高速化用の HW 演算器を利用することで，本ペアリング実装は約 2 秒で BN254 optimal Ate ペアリングを実行できる．また本実装は TVC 上の乗算器コプロセッサをフラッシュから制御する方法でプログラミングしており，公開鍵暗号のようなリッチな暗号をハードウェアで高速化する際にも命令改変が脅威になることが示される．

ペアリングに対する命令改変攻撃

命令改変を用いて，TVC 上に実装したペアリング演算を攻撃する．解析手法には Bae らの手法 [7] 用いるため，ミラーループ中の if 文スキップが攻撃目標となる（アルゴリズム 2 の 5 行目）．さらに最終ベキの影響を回避するため，最終ベキのスキップも同時に行う．

Algorithm 2: BN 曲線上の Optimal Ate ペアリング [3]

Input: $P \in G_1$, $Q \in G_2$, $s = |6u + 2|$
Output: $a_{opt}(P, Q)$

1 $T \leftarrow Q$, $f \leftarrow 1$
2 **for** $i := \lfloor log_2 s \rfloor - 1$ **to** 0 **do**
3 　$f \leftarrow f^2 \cdot l_{T,T}(P), T \leftarrow 2T$
4 　**if** $s_i = 1$ **then**
5 　　$f \leftarrow f \cdot l_{T,Q}(P), T \leftarrow T + Q$
6 　**end**
7 **end**
8 $Q_1 \leftarrow \pi_p(Q), Q_2 \leftarrow \pi_p^2$
9 **if** $u < 0$ **then**
10 　$T \leftarrow -T, f \leftarrow f^{-1}$
11 **end**
12 $f \leftarrow f \cdot l_{T,Q_1}(P), T \leftarrow T + Q_1$
13 $f \leftarrow f \cdot l_{T,-Q_2}(P), T \leftarrow T - Q_2$
14 $f \leftarrow f^{p^{12}-1/r}$
15 **return** f

ミラーループ関数中の if 文をコンパイルした結果，リスト 4.4 のようなアセンブラが得られた．cmp, beq 命令によって，r0 の値が 0 のときに分岐を行っており，r1 の値が 1 のときに関数 EC_addition, Fp12_sparse_mul が呼ばれる．命令改変によってこの 2 つの関数の実行をスキップする方法はいくつかあるが，今回は bl 命令の条件フィールドを改変することを目指す．ARM 命令セットの命令は最上位の 4bit が条件フィールドと呼ばれる形式になっており，これによって命令の実行を条件付きにすることができる．例えば beq 命令の条件フィールドは 0x0 になっており，これはゼロフラグがセットされている際に実行されることを意味する．一方でプログラム 1 中の cmp, bl 命令の条件フィールドは 0x0e となっておりこれは無条件実行を意味する．今回は bl 命令の条件フィールドを 0x6（オーバーフローフラグが 1 のときに実行）に改変する．bl 命令の直前にある引数準備の段階でオーバーフローフラグがセットされることはほぼないと思われるため，これによって bl 命令を実行させなくすることが可能である．照射位置はフラッシュメモリ上の 32bit 目を 1 にセットするとした．また照射時間は 1 µs としプログラム 1 のアドレス 0x431b4 から 0x431cc にかかるようにした．

実験の結果を図に示す．図 4.18 にはミラーループ実行中の TVC の消費電力が示されており，nop 実行時（点線丸で示した部分）に消費電力が下がっていることがわかる．ミラーループのループ構造による nop は繰り返し現れており，レーザー非照射時には初回の nop と nop の間隔が長くなっている．これは if 文の中を実行しているためである．レーザー照射時には nop の間隔が短くなっており，if 文がスキップされたことがわかる．またこのとき出力された値は想定したものと一致していた．ペアリング演算の攻撃のためにはミラーループの攻撃に続いて最終ベキをスキップする必要がある．最終ベキ部分のアセンブル結果を見たところ，プログラム 1 の関数呼び出しと同じように bl 命令にて最終ベキ関数の呼び出しを行っていた．従ってミラーループの攻撃と同様に条件フィールドを改変することで最終ベキをスキップすることが可能である．これも実験で確認し，出力した値が想定と正しいことを確認した．

Listing 4.4: ミラーループ関数の if 文部分のコンパイル結果（一部）

```
1  |  :            nop  x200
2 4319c: e3500000 cmp  r0, #0
3 431a0: 0a00000a beq  431d0
4  |  : 引数の準備
5 431b4: ebfff497 bl  40418 <EC_addition>
6  |  : 引数の準備
7 431cc: ebfff8f0 bl  41594 <Fp12_sparse_mul>
```

ペアリング暗号に対する命令改変の脅威

命令改変によってペアリング演算を攻撃できることがわかったが，ペアリング暗号の攻撃ではさらにハッシュ関数の攻撃が必要となる．本章では，命令改変を使ってハッシュ関数の影響を受けずに秘密情報を取得する方法を解説する．リスト 4.5 に，実装したペアリング演算（暗号）の出力部のアセンブル結果を示す．ここで，res_out 関数は第一引数が示すアドレスから 256bit を外部へ出力する関数である．今回実装したペアリングの出力は 256bit の元が 12 個であるから res_out 関数は計 12 回呼ばれる．リスト 4.5 は初めの 3 回部分を示している．ARM EABI では第一引数は r0 レジスタで渡されることが決まっており，sp のアドレスに出力すべき値が格納されていることがわかる．プログラム 2 は sp の値を一度 r4 に保存し，r4 に 32 を加算していくことで 256bit×12 の値を順次出力している．ここで問題になるのは，例えば add 命令の即値にフォールトを注入することで，本来出力してはならない領域の値が出力される恐れがあることである．

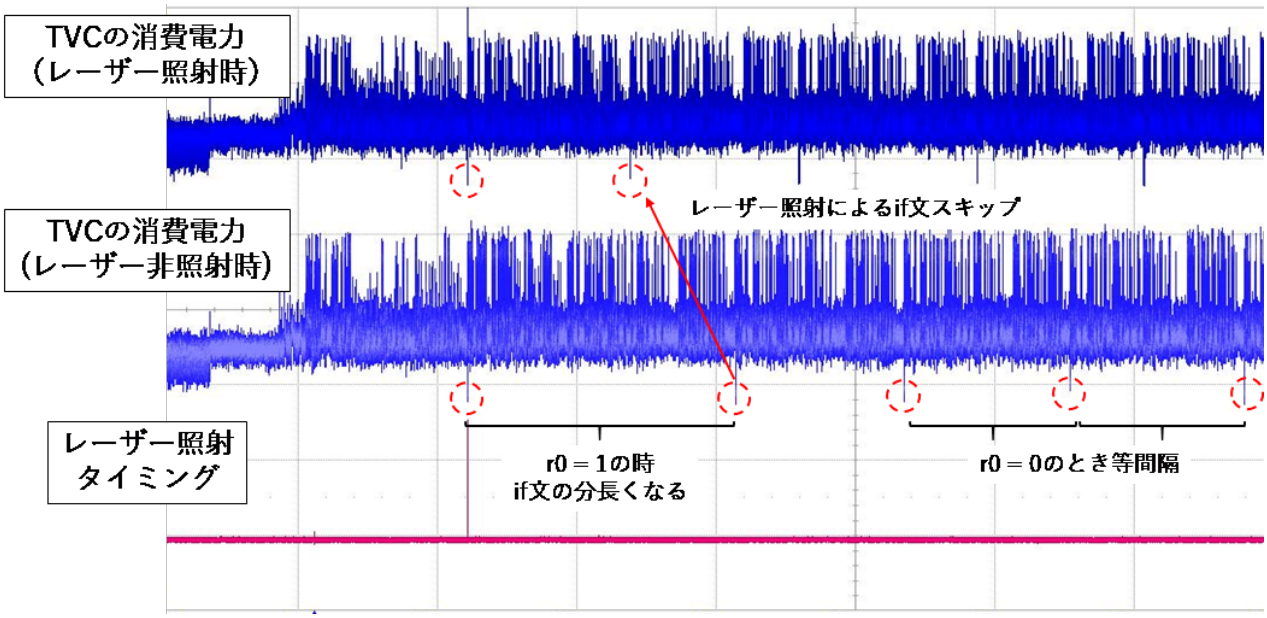


図 4.18: 命令改変による if スキップ

Listing 4.5: ペアリング演算出力部分のコンパイル結果（一部）

```
1 43650: e1a0000d mov     r0, sp
2 43658: ebfffd0d bl      42a94 <res_out>
3 43660: e1a0400d mov     r4, sp
4 43664: e28d0020 add     r0, sp, #32
5 4366c: ebfffd08 bl      42a94 <res_out>
6 43674: e2850040 add     r0, r4, #64
7 4366c: ebfffd08 bl      42a94 <res_out>
```

今回の実装ではペアリングの出力 f, 秘密情報 Q, 公開情報 P がこの順番でスタック領域に格納されており，すなわち sp を増加させることで Q を出力させることができる．これにはいくつか方法が考えられるが，add 命令の書き込みレジスタ r0 を r4 に改変することを考える．ARM 命令セットから，この改変は 15 ビット目を 1 にセットする照射位置で達成できる．例えば，アドレス 0x43664 の add 命令を改変することで，次の res_out 関数による出力はおかしなものになるが，r4 が sp+32 に更新されるため続く res_out 関数の出力からは 256bit ずつずれたものが出力されるようになる．この結果，最後の res_out 関数呼び出し時には本来出力されるはずのない，f の次の領域すなわち Q の最初の 256bit が出力される．同様のフォールトを別の add 命令に引き起こすことで，全ての Q を出力させることができる．この攻撃は追加の解析を必要としないことから，フォールト出力を得たあとの解析の計算が必要ない．また最終べキやハッシュ関数の影響も受けないため，既存の解析手法よりも強力な攻撃であるといえる．

4.4　レーザー照射によって ROM の値を抽出する攻撃

ここまでに示した命令改変攻撃は，攻撃者が ROM に格納されているプログラムを知っているという前提のもとで行われてきた．このような前提は決して非現実的なものではないが，我々はレーザー照射を用いて ROM 内の値を読み出す手法を新たに提案する．

提案手法には以下のような特徴がある．

■EEPROM に対する論理的なフォールトを注入しない攻撃：　提案手法は，論理的なフォールトを注入しない程度のレーザーを用いて EEPROM の値を抽出した初めての例である．これは ECC 等のデジタルなレイヤでは検知できないから，光センサ等のアナログ対策しか有効でなくデジタルな対策に比べて大きなコストがかかる．

■センスアンプへのレーザー照射：　OBIC や OEPA といった，論理的なエラーを注入しないレーザーを用いて SRAM セルの値を抽出する手法が発表されている．提案手法はこれらの手法とよく似たものであるが，従来手法があるセルにレーザーを照射していたのに対して提案手法はセンスアンプを狙う．センスアンプは非常に微小な電流を感知するものであるからレーザー照射に脆弱だとされているにもかかわらず，実際にセンスアンプを狙って攻撃した事例は我々の知る限り報告されておらず本研究が初めての試みである．

■セルの状態ではなく読み出し中の値を取得：　OBIC や OEPA はある SRAM セルにレーザーを照射してそのセルの状態や状態の変化を得ることを目的としており，あるレーザー照射位置に対して対象セルの値しか得ることができない．一方で提案手法はセンスアンプを通るメモリ読み出し値を取得するものであるから，一度照射位置を特定すれば位置の移動をすることなく複数のセルの値を得ることができる．

■Partial reverse engineering：　提案手法はメモリから読み出されている値を取得するものであるから，メモリの内容全体を得ることはできない．しかしながらこれは逆に攻撃者に有利に働くかもしれない．例えば，EEPROM からの秘密鍵のロードは暗号処理の直前に行われると推測できるから，SPA を用いて暗号処理の実行タイミングを知っている攻撃者は秘密鍵読み出し中の値のみを得ることができる．このように，提案手法は取得したい情報だけを選択的に得ることができる．

4.4.1　提案手法

提案手法を導入する前に，EEPROM の代表的な構造と読出し動作を解説する．本手法は EEPROM を対象としたものであるが，本紙では特に NOR-type Flash EEPROM に対する結果が示される．近年の多くの組込み機器はプログラム格納領域に Flash EEPROM を用いており，これは大容量のため回路面積が大きく容易に位置を特定することができる．

図 4.19(a) に，NOR-type flash EEPROM の構造図を示す．w ビットワードのフラッシュは w 列から成るセルアレイを持っており，n 列目にはワードの中の n ビット目が保存されている．フラッシュメモリの読出し動作時にはまず，ワード線とビット線がアサートされることにより各セル列から一つのセルが選択される．このとき選択されたセルが値 ‘0’ を格納しているならばそのセルに電流が流れる．一方で値 ‘ 1 ’ を格納している際には電流が流れない．この時の電流は，各列に接続されているセンスアンプによって増幅され，論理値として出力される．従って，センスアンプは 0/1 の電流によって 0/1 を出力する 2 状態の回路だとみなすことができる．ここで，w 列目から論理値 “1010” が読み出されている場合を考える．w 番目のセンスアンプは，図 4.19(b) (i) のような論理値を出力するはずである．ここで横軸は時間経過を示している．このとき，(ii) のようなタイミングで w 番目のセンスアンプにレーザーを照射した際したとする．センスアンプを構成する回路の一部にレーザーが照射されているはずだが，センスアンプが 2 状態回路である以上その一部分もまた 2 状態回路となっているはずである．複数のトランジスタから構成される回路にレーザーを照射した際の影響を予測することは一般には困難とされるが，2 状態回路である以上はレーザーを照射した際の影響も 2 状態を持つことが予測される．提案手法ではこのときの影響の差を

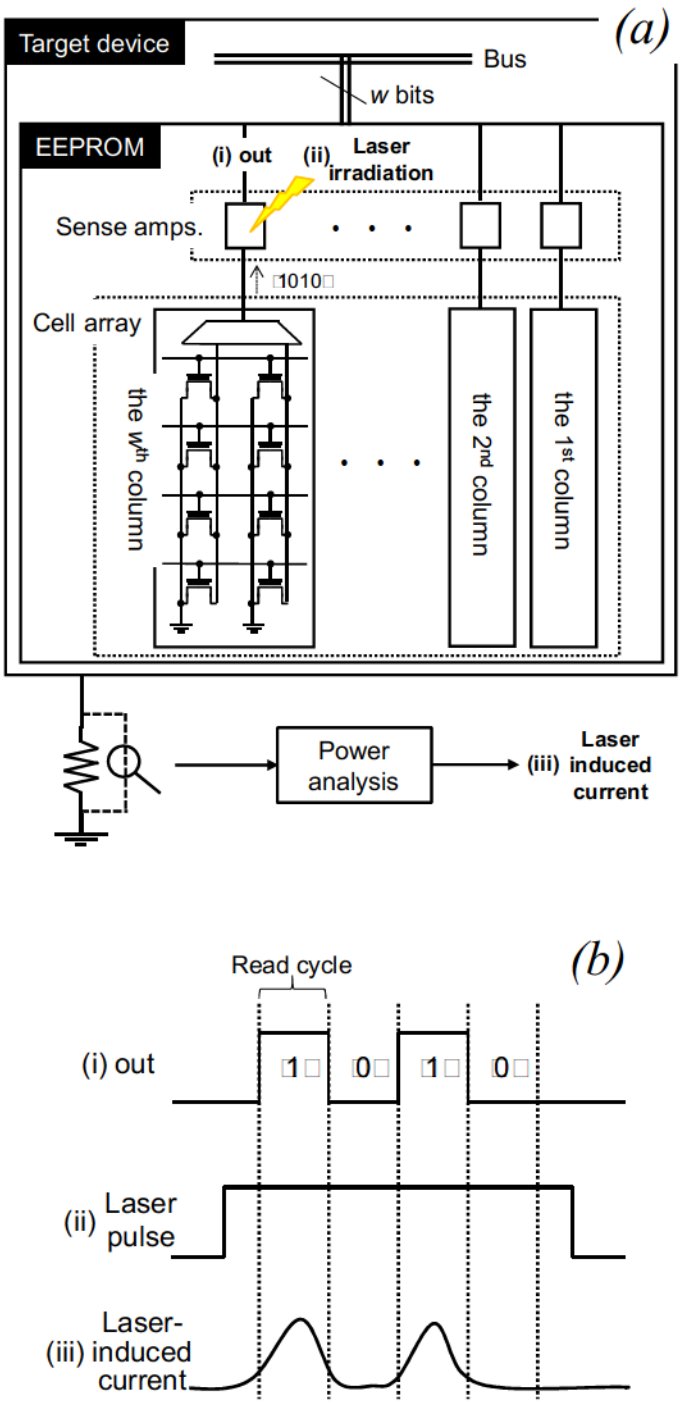


図 4.19: (a)NOR Flash EEPROM の構造及び (b) 提案手法の原理

電流量の差として観測する．ひとつの CMOS トランジスタにレーザーを照射した時，そのトランジスタの状態によって増加する電流 (Laser-Induced Current, LIC) の量が変化することが知られており，この影響は複数のトランジスタで構成される回路に対しても拡張できるはずである．すなわち，観測される LIC の量は，センスアンプの状態 (出力の値) によって変化することが期待され，(iii) のようになると思われる．また，我々が観測可能な消費電力はふつう回路全体の電力を含んでおり，(iii) のように LIC のみを抜き出す必要がある．提案手法では次の章に示す電力解析手順を踏むことで，LIC の抽出に成功した．

提案手法は OBIC や OEPA のようなひとつのセルを狙ってレーザーを照射する手法と異なり，センスアンプに対して照射を行う．従ってメモリに保存されている値を抽出するものというよりはむしろ，メモリ読出し値を抽出するものである．提案手法はすべてのセルの値を読み出しことはできないかもしれない．しかし従来の手法では複数のセルの値を抽出するためにレーザーの移動が必要であったのに対し，提案手法はレーザーを移動することなく複数セルの値を得ることができる．EEPROM は頻繁に書き換わるものではないと考えられるから，これは非常に効率的な手法であると考える．

4.4.2　AVR プロセッサに対する提案手法によるプログラム抽出結果

提案手法は論理的なエラーを注入しない程度のレーザーを使うから，我々はまずレーザー強度とフォールト感度の関係を調べる実験を行った．IPA-SLS を用い、DUT は ATMega8515 とする．DUT のフラッシュを値 ‘1’ で初期化し，図 4.20 の照射位置 P に 41 mW のレーザーを照射した．この間にフラッシュ内の全 4k ワードを読み出したところ，本来 ‘FFFF’ の値が読み出されるはずのところの 1 ビット目が誤った値 ‘FFFE’ が全ワードで読み出された．同様の実験をレーザー強度を少しずつ弱くして行ったところ，図 4.21 のような関係が得られた．図の横軸はレーザー強度，縦軸はそのレーザー強度の時に 4k ワードのうち誤りのあったワードの割合を示している．約 35 mW 以上のレーザーは全てのワードに誤りを注入し，約 33 mW 以下は誤りを注入していないことがわかる．レーザー強度が高いほど誘起される電流量は増加すると予測されるため，続いて行う実験では 33 mW のレーザーを利用することに決定した．

提案手法によりフラッシュ EEPROM の読出し値を抽出する実験を行う．対象のフラッシュにはリスト 4.6 のようなテストプラグラムが書き込まれており，実行中にはこれらが 5 MHz で 1 命令ずつ読み出される．実験は次の手順で行われる：

1. レーザーを照射しない状態でテストプログラムを 50 回動作させ，この際の電力波形を平均化したものを T_{ref} とする．
2. 図 4.20 の位置 P に 33 mW のレーザーを照射し，その間にテストプログラム 1 を動作させる．これを 50 回繰り返し得られた電力波形を平均化したものを T_{laser} とする．
3. T_{laser} と T_{ref} の差分を求め T_{diff} とする．

上の手順の結果を図 4.22 (a) に示す．図 4.22 (a) の一番上に T_{laser} と T_{ref} が重ねて表示されており，ちょうど時間軸の中央付近でテストプログラムが実行されている．これらの波形には一見して大きな差があるように見えないが，レンジを拡大して表示した T_{diff} には明らかなスパイクがいくつか見て取れる．さらにこれらの波形を 1 クロックサイクルごとに区切ると，スパイクがその間にぴったりと収まっている．クロックサイクルの中にスパイクがある部分を ‘1’ ない部分を ‘0’ とすると，T_{diff} からビットシーケンス ‘…01101010…’ が得られた．これはテストプログラム 1 の 1 ビット目に対応しており，提案手法によってフラッシュから読み出されている値を抽出することができた．また，照射位置を図 4.20 の Q に変えて同様の実験を行った結果を図 4.22 (b) に示す．今度は T_{diff} からビットシーケンス ‘…01011010…’ が得られており，これはテストプログラムの 9 ビット目に対応している．同様の実験を全 16 列に繰り返し，テストプログラム 1 を完全に復元することができた．このときの T_{diff} に信号処理を行い，0/1 を識別させた結果を図 4.23 に示す．それぞれの列でユニークなビットシーケンスが現れていることがわかる．またこの実験は全体で 30 分以下で終了した．

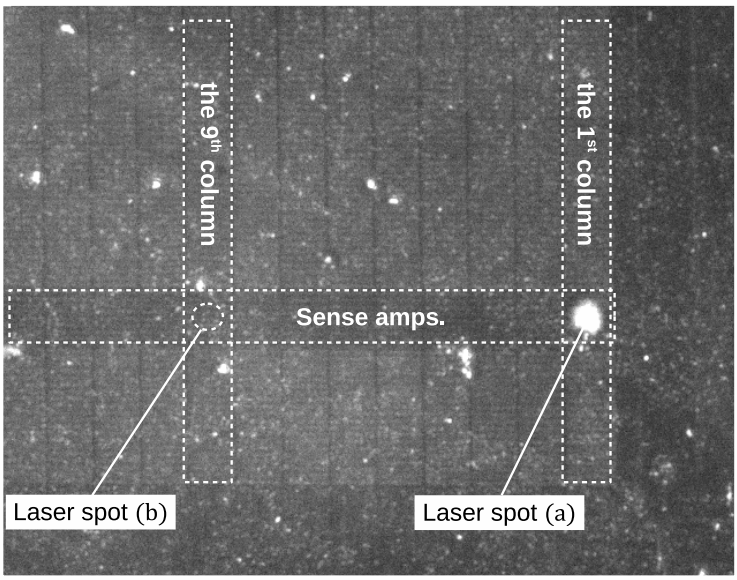


図 4.20: レーザー照射位置（裏面赤外線画像）

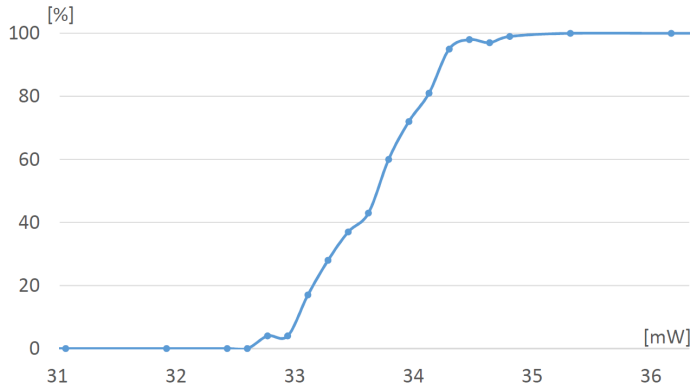


図 4.21: レーザー強度とフォールト感度の関係

Listing 4.6: テストプログラム

```
1  .func Test
2
3  Test:
4      nop                  ;0000 0000 0000 0000
5      add r27, r25         ;0000 1111 1000 1001
6      subi r26, 0x01       ;0101 0000 1010 0001
7      and r27, r28         ;0010 0011 1011 1100
8      ori r24, 0x11        ;0110 0001 1000 0001
9      eor r25, r24         ;0010 0111 1001 1000
10     inc r24              ;1001 0101 1000 0011
11     nop                  ;0000 0000 0000 0000
12
13 .endfunc
```

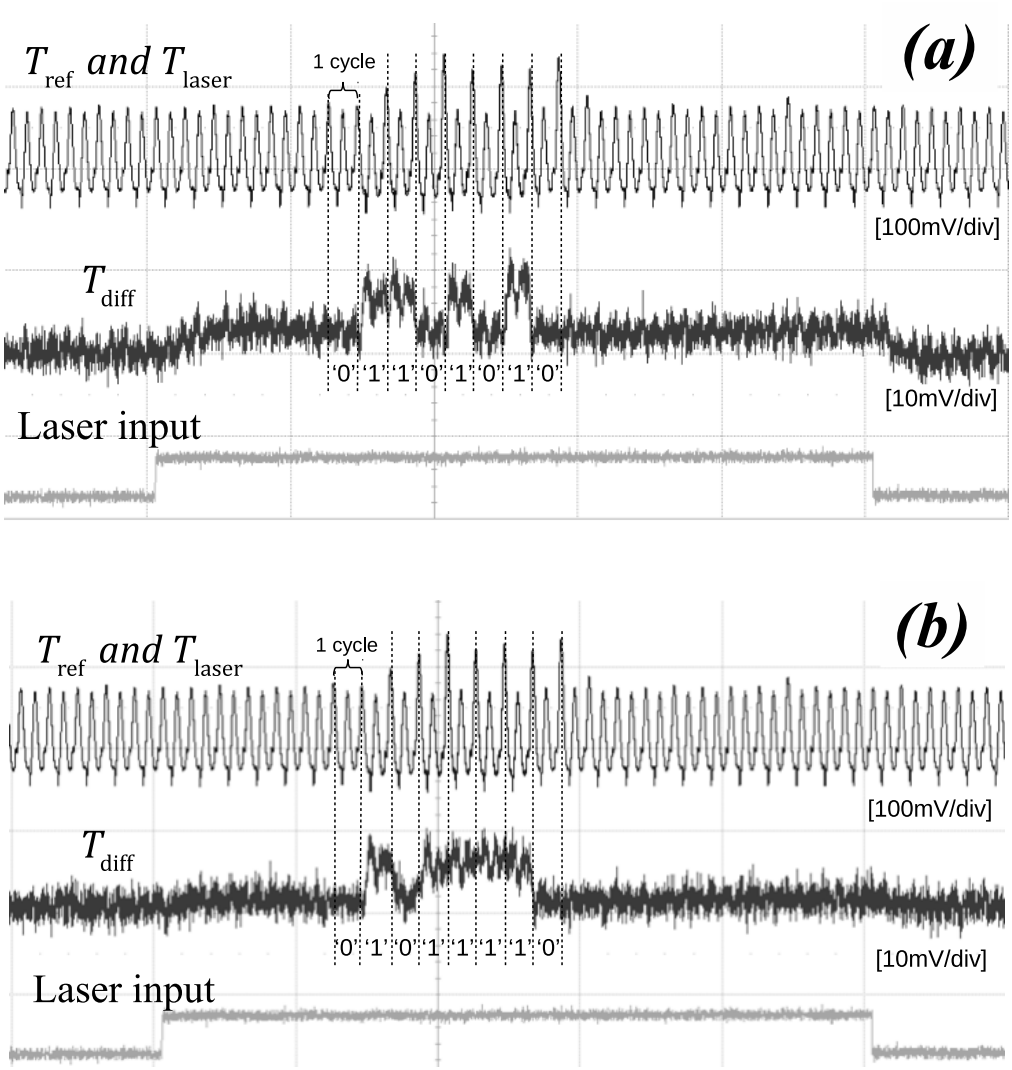


図 4.22: テストプログラムの抽出結果

4.4.3　実験結果の考察

我々はテストプログラム以外にも分岐命令やロードストア命令を含むプログラムに対して実験を行い良好な結果を得た．特筆すべきは，フラッシュからのロード命令 lpm によりフラッシュから読み出される値も取得できたことである．このことから，提案手法はフラッシュからフェッチされる機械語だけでなくロードされる値を全て取得できることがわかる．暗号処理の秘密鍵等，秘密情報をフラッシュからロードして利用するような場面では，それらの守秘性が提案手法により脅かされる危険がある．また本紙の最初で述べたように，提案手法はまさに読み出されている最中の値を取得するものであるからメモリの内容全体を抽出することはできない．しかしながら，例えば秘密鍵のロードは暗号処理の直前に行われると推測できるから，SPA を用いて暗号処理の実行タイミングを知っている攻撃者は秘密鍵読み出し中の値のみを選択的に取得することが可能である．

さらに，提案手法はセンスアンプを狙うものであるから，センスアンプを利用する他のタイプのメモリにも適用できるかもしれない．今回は特に NOR タイプフラッシュを実験対象とたが，NAND フラッシュや EEPROM, SRAM に対しても提案手法は有効であると考えられ，今後の結果報告が望まれる．

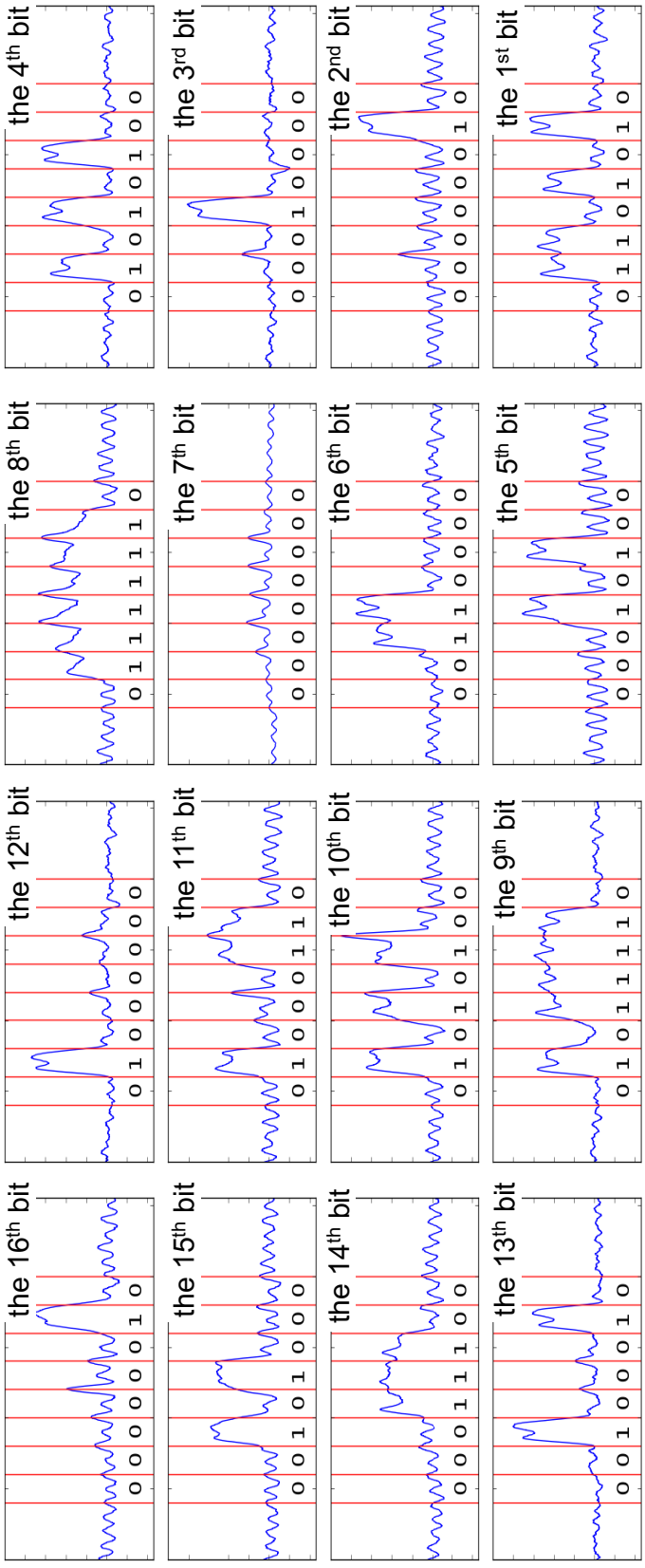


図 4.23: 全 16 ビットの抽出結果

第 5 章

レーザー照射攻撃対策

5.1　ハードウェアによる対策

レーザー照射に対するハードウェアレベルの対策として，光センサによるレーザー光検知が提案されている [21]．この手法はアナログレベルでレーザーを検知できるため強力な対策といえるが，レーザーセンサを配置していない場所にはレーザー照射が可能であり，またセンサを高密度にすればするほどコストが上昇するといった欠点がある．より洗練されたレーザーセンサとして，レーザーが照射された際の基板電流を監視するものが提案されている [35]．この対策は基板に流れる電流を検知するため，ある程度の範囲を持った領域のレーザー照射を検知することができ，低密度でも効果の高い対策をとることが可能である．フラッシュメモリへのレーザー照射に限定したハードウェアレベル対策のとしては，フラッシュメモリの暗号化やパリティチェックの追加などが提案されている．しかしながら，毎サイクルリードされるフラッシュからの値を毎サイクル復号するのは難しいため，単純な XOR スクランブルのような方法しか実現していないようである．この場合命令改変は可能だと考えられる．

5.2　ソフトウェアによる対策

5.2.1　命令スキップ対策

これまでに提案されているソフトウェアベースの命令スキップ対策手法を紹介する．命令スキップ対策の多くは，冗長計算を行いその計算結果が等しいかどうか検算するものが多い．ここでは命令レベル，アルゴリズムレベルの二つの検算対策を紹介する．

命令ダブリング（Instruction doubling, ID）

命令レベルでの命令スキップ対策である命令ダブリング（ID）の例をリスト 5.1 に示す．このプログラムはロード命令に冗長性持たせており，同じアドレスから異なるレジスタに値を読み出し，その値を検算することでフォールトを検知する．このプログラムを命令スキップで攻撃するためには二つの方法がある．r1 と r3 それぞれ同じフォールト値を注入するか，レジスタにフォールトを注入しかつ分岐命令をスキップするかである．この対策は短い間隔で命令スキップを注入することは困難という仮定に基づき安全とされている．このプログラムで KEY はアドレスを表しており，今後アドレスに格納された値を言及したい際は [KEY] のように表記する．

デフォルトフェイル付き平文検算 (Plaintext integrity verification with defaultfail, PIVD)

アルゴリズムレベルでの命令スキップ対策であるデフォルトフェイル付き平文検算をリスト 5.2 に示す．AES 暗号化実行後に暗号文が復号され，平文と復号結果を比較することで，フォールト注入を検出する．この対策は，プログラムがふつう下方向に進むことから，命令スキップによって上方向には移動できないことを利用している．リスト 5.1 の対策のように，フォールト検出時に分岐する方法は分岐命令スキップ時に正常なプログラムフローへ進んでしまうが，PIVD のようにフォールト非検出時に正常フローへ分岐することで，分岐命令スキップによる攻撃を対策することができる．リスト 5.2 の対策は比較前にゼロフラグをクリア することで，9 行目の比較処理をスキップされた際に偶然分岐してしまう可能性を防いでいる．

Listing 5.1: Instruction duplication.

```
1  .extern FaultHandler
2  .func ID
3  .thumb
4
5  ID:
6      ldr r1, =KEY
7      ldr r3, =KEY
8      cmp r1, r3
9      bne FaultHandler
10
11 .endfunc
```

Listing 5.2: Plaintext integrity verification with defaultfail.

```
1  .extern FaultHandler
2  .extern OutCTXT
3  .func PIVD
4  .thumb
5                      ;OutCTXT function is located at a lower address ↩
                           from here.
6  PIVD:
7      b LBL_MAIN
8  LBL_CMP:
9      cmp r3, r1
10     beq OutCTXT
11     b FaultHandler
12 LBL_MAIN:
13     b EncAndDec     ;Encrypt PTXT to CTXT, and decrypt CTXT to PTXT'.
14     ldr r1, [r3, #0]
15     ldr r3, [r3, #4]
16     mov r5, #1      ;Init Z flag to 0.
17     bne LBL_CMP
18                     ;FaultHandler function is located at an upper ↩
                           address from here.
19 .endfunc
```

5.3　命令改変攻撃対策の提案

命令スキップは実行中のアセンブリ命令の実行をスキップさせるフォールトであり，これまでの実験結果から多くの組込みプロセッサに対して注入可能なことがわかっている．このため命令スキップの注入は一般的な攻撃者が持つ能力として認識されており，命令スキップに対して安全な対策がいくつか提案されている．一方で，実行中のアセンブリ命令を別の命令に改変して実行させる命令改変を組込みプロセッサに注入できることも報告されている．命令スキップは何もしない命令（NOP 命令）への改変と捉えることもできるため，命令スキップは命令改変の一部分だと考えることができ，従って命令改変は命令スキップよりも強力なフォールトである．命令改変は多くのプロサッセ上及び多くのフォール注入手法で報告されているが，どのような命令に改変するか制御することが難しいため攻撃に利用できないとの見解が多かった．しかしながら最新の研究では，レーザー照射のような精密なフォールト注入手法によってある程度は命令改変の制御が可能であることが示され，さらに既存の命令スキップ対策を実際に命令改変で攻撃した結果が報告されている．このように攻撃手法の研究は日々進展しており，それに合わせた対策手法の検討が必要である．本論文では，命令改変に対して安全に値の検算を行う手法を提案する．既存のソフトウェアによるフォールト攻撃対策の多くは計算途中の結果にフォールトが注入されているかどうかを検算することに基づいており，安全な検算はフォールト攻撃対策に必須の要素である．

5.3.1　対象とする命令セット：16-bit Thumb

命令改変攻撃がデバイスに与える影響は命令セットアーキテクチャ（ISA）の構造に大きく依存するから，命令改変攻撃対策を提案する上でまず対象とする ISA を決定する必要がある．我々は ARM 16-bit Thumb ISA を対象として，命令改変攻撃に対する対策を提案する．命令長が長くなるほど命令改変の挙動が複雑になり対策が困難になるため，16 ビットの命令セットを対象とした．ARM は世界で最も普及しているプロセッサの一つであり，ARM プロセッサ向けに書かれたアプリケーションソフトウェアは多岐にわたる．ARM プロセッサは伝統的に 32-bit の命令セットを備えているが，コード量や消費電力の削減のため ARMv4T 以降のプロセッサには新たに 16-bit の Thumb 命令セットが追加された．Thumb 命令セットは Thumb-1 と呼ばれることもあるが，本研究では単に Thumb と呼ぶ．ARMv7, ARMv8 のような最新のプロセッサはより洗練された可変長の命令セット Thumb-2 を実装しているが，Thumb-2 は Thumb のスーパーセットであるため Thumb で書かれたプログラムは Thumb-2 上でも動作する．このように Thumb 命令セット上で対策手法を提案することによりそのアプリケーションは多岐に渡る．ARM プロセッサは 16 個のレジスタ（r0-r15）を備えているが，Thumb 命令セットからはそのうちの 8 つ（r0-r7）にしかアクセスできない．また検算処理を行う上で重要となるのが条件分岐命令である．Thumb の条件分岐は cpsr (current program status register) 上の 5 つの条件フラグ，N (negative), Z (zero), C (carry), V (overflow), Q(saturation) の状態によって制御される．例えば beq (branch equal) 命令は $Z = 1$ のときに分岐し，bhi (branch unsigned higher) 命令は $C = 1$ かつ $Z = 0$ のときに分岐（ジャンプ）する．これらの条件フラグはプロセッサの演算結果の状態に従って随時セット/クリアされる．例えば検算処理の際には比較命令によって Z フラグを更新するのが一般的である．

5.3.2　想定するフォールトモデル

第 4 章で示したシングルスポットレーザーによる命令改変攻撃の結果を受け，我々は攻撃者の注入できるフォールトに以下の制限を課す．

(i) 攻撃者は，実行される機械語の 1 ビットをセットすることができる．
(ii) 攻撃者は，任意の複数命令にフォールトを注入することができる．
(iii) 複数命令にフォールトを注入する場合，注入ビットの位置を変えることができない．

以上の能力を持った攻撃者を One-bit-set instruction manipulation model（OSIM）呼び，フォーマルな定義を以下に示す．

Definition 1 (命令)**.** 命令長を w とすると，ある（機械語）命令 ins は w 次元行ベクトル,

$$ins = (b_{w-1}, b_{w-2}, \ldots, b_0), b_i \in \{0, 1\}$$

として表現される．

Definition 2 (実行中のプログラム)**.** 実行中のプログラム P は n 個の命令から成る $n \times w$ 行列,

$$P = \begin{pmatrix} ins_1 \\ ins_2 \\ \vdots \\ ins_n \end{pmatrix} = \begin{pmatrix} b_{1,w-1}, b_{1,w-2}, \ldots, b_{1,0} \\ b_{2,w-1}, b_{2,w-2}, \ldots, b_{2,0} \\ \vdots \\ b_{n,w-1}, b_{n,w-2}, \ldots, b_{n,0} \end{pmatrix}, b_{i,j} \in \{0, 1\}$$

として表現される．

P は ROM に格納されたプログラムではないことに注意せよ．命令改変フォールトは命令レジスタに生じるものであり，ROM に保存された値を書き換えるわけではない．

Definition 3 (One-bit-set instruction manipulation (OSIM) モデル)**.** フォールトベクトル fv を n 次元列ベクトル,

$$fv = \begin{pmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{pmatrix}, e_i \in \{0, 1\}$$

とする．c ビット目への OSIM フォールトは次のように表現される;

$$P \oplus_c fv = \begin{pmatrix} b_{1,w-1}, b_{1,w-2}, \ldots, b_{1,c} | e_1, \ldots, b_{1,0} \\ b_{2,w-1}, b_{2,w-2}, \ldots, b_{2,c} | e_1, \ldots, b_{2,0} \\ \vdots \\ b_{n,w-1}, b_{n,w-2}, \ldots, b_{n,c} | e_1, \ldots, b_{n,0} \end{pmatrix} = P',$$

ここで演算子 $\oplus_c$ は，右オペランドと左オペランドの c 列目の要素ごとの OR 演算を表しており，$|$ はビット OR 演算子である．

c はレーザーが照射しているフラッシュメモリ上の列の位置，すなわち攻撃対象のビット位置を示しており，$e_i = 1$ の箇所でレーザー照射が行われたことを示している．例えば，e_1, e_3, e_n が場合 1，3，n 番目の命令フェッチタイミングでレーザーが ON になる．オリジナルのプログラム P とは異なる P' が実行されることで，本来意図しない動作が発生し，重大な情報漏洩が引き起こされる恐れがある．

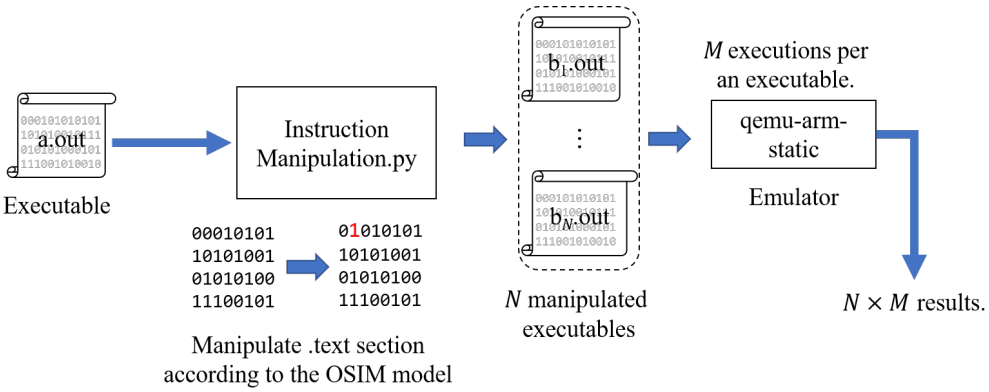


図 5.1: OSIM モデルシミュレーション環境

5.3.3　OSIM モデルシミュレート環境

図 5.1 のような OSIM モデルをシミュレートする環境を構築し，それによってリスト 5.2 を攻撃することでどのような命令改変が情報漏洩を引き起こすのか調査する．この環境は Thumb 命令で書かれた実行形式ファイルを入力とし，パイソンプログラムが OSIM モデルに従って，考え得る全ての命令改変パターンで実行形式ファイル中の.text セクションを改変する．命令長 w の命令が n 命令実行形式ファイルに含まれるとき，w 個の位置に対してそれぞれ 2^n 個のフォールトベクトルが考えられるから，全ての命令改変パターンは $N = w \times 2^n$ 個となる．パイソンプログラムはそれぞれの命令改変を行った N 個の実行形式ファイルを生成し，それぞれが qemu 上で引数を変えながら M 回実行される．得られる $N \times M$ 個の実行結果を解析し，どのような改変パターンが攻撃成功に繋がるかを調査することで対策の構築を行う．

5.3.4　命令改変による PIVD 攻撃シミュレーション結果

図 5.1 の環境を使って，リスト 5.2 の命令スキップ対策が命令改変によってどのように攻撃されるかを調査した結果を表 5.1 にまとめる．ここではあらかじめ PTXT と PTXT′ には異なる値が格納されており，暗号化処理中にフォールト注入が行われたことを想定している．パラメーター $n = 8$[*1], $N = 16 \times 2^8 = 4096$, $M = 1$ である．攻撃結果から，全ての改変パターンのうち約 3 % (112 / 4096) が攻撃成功（誤り暗号文出力）に繋がっていることがわかる．ほとんどの命令改変は検知されるか，例外処理に進んでいる．ここで EXP4, EXP11, EXP19 はそれぞれ未定義命令，セグメンテーションフォールト，タイムアウトを示す．この結果は攻撃成功率が低すぎるように思われるかもしれないが，実際には攻撃者はこのうちの成功する命令改変のみを使うことができるため，どの改変が攻撃成功に繋がるかわかっている攻撃者は高い確率で誤り暗号文を得ることができる．

誤り暗号文出力に至った 112 の命令改変パターンを分析したところ，大きく分けて 4 つのカテゴリにクラス分けできることがわかった．次でそれぞれのクラスがどのように攻撃を成功させるか解説し，それに対する対策手法を提案する．

[*1] EncAndDec 関数の実行中にすでにフォールトが注入された状況を再現しているため，13 行目の分岐命令は命令改変対象から除いている．

表 5.1: シミュレータによる PIVD 攻撃結果

攻撃対象	誤り暗号文出力	フォールト検知	EXP4	EXP11	EXP19	n	N
Original PIVD	112	1056	2080	847	1	8	4096

5.3.5　4 つの命令改変攻撃クラス及びそれらに対する対策手法

オペランドレジスタ改変 (Operand register manipulation, ORM)

図 5.2 のように 9 行目の cmp 命令のオペランドレジスタが改変される攻撃である．Thumb 命令セットの cmp 命令はソースレジスタ Rs 及びディスティネーションレジスタ Rd の二つをオペランドとして持ち，オリジナルの PIVD では異なるレジスタ r1 及び r3 がそれぞれ Rs 及び Rd として利用されている．これらの二つのレジスタの値が異なる場合，PIVD はフォールトを検知しフォールトハンドラ関数へ至るはずだが，図 5.2 のような改変によって常に同じ値が比較されるようになり，完全性検証プロセスは常にバイパスされる．

■ORM に対する対策手法　上記のような攻撃を防ぐために，cmp 命令の Rs 及び Rd を適切に選択する手法を提案する．OSIM モデルの攻撃者は命令の 1 ビットしか改変できないため，Rs と Rd が次のような関係を満たす場合に攻撃者は Rs==Rd とすることができない;

$$HD(bin(Rs), bin(Rd)) > 1,$$

ここで $HD(,)$ は二つの引数のハミング距離を返す関数であり，$bin(reg)$ はレジスタ reg のバイナリ表現を示している．上記の手法を適用した，ORM-secure PIVD 実装の例をリスト 5.3 に示す．9 行目の cmp 命令のオペランドがオリジナルの PIVD と異なる．それと併せて，ロード命令で使われるレジスタも変更している．

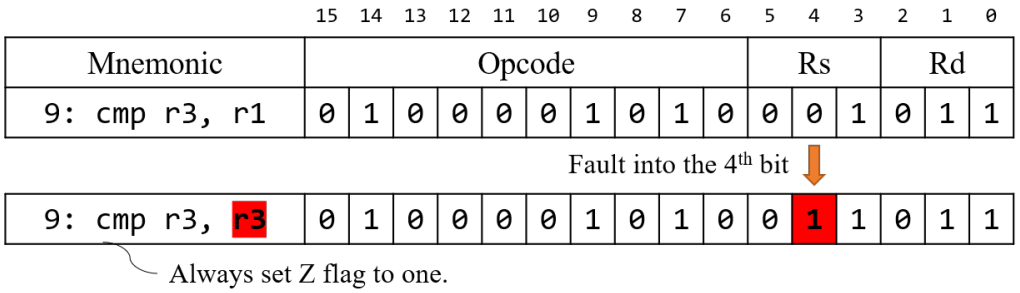


図 5.2: Cmp 命令の機械語フォーマットと完全性検証プロセスをバイパスする改変の例

Listing 5.3: ORM-secure PIVD 実装.

```
1 .extern FaultHandler
2 .extern OutCTXT
3 .func ORM_secure_PIVD
4 .thumb
5                       ;OutCTXT function is located at a lower address ↩
                            from here.
```

```
 6 ORM_secure_PIVD:
 7     b LBL_MAIN
 8 LBL_CMP:
 9     cmp r3, r5
10     beq OutCTXT
11     b FaultHandler
12 LBL_MAIN:
13     b EncAndDec      ;Encrypt PTXT to CTXT, and decrypt CTXT to PTXT'.
14     ldr r5, [r3, #0]
15     ldr r3, [r3, #4]
16     mov r1, #1       ;Init Z flag to 0.
17     bne LBL_CMP
18                      ;FaultHandler function is located at an upper ↩
                            address from here.
19 .endfunc
```

分岐アドレスの改変（Branch address manipulation, BAM）

11 行目の無条件分岐命令のアドレスを改変することで強制的に OutCTXT 関数を実行する攻撃である．図 5.3 のように，Thumb 命令セットの無条件分岐命令は分岐先を 2 の補数による pc 相対アドレスで表現している．PIVD は命令スキップを防ぐために FaultHandler 関数を完全性検証よりも上位のアドレスに配置しており，従って FaultHandler 関数への分岐命令の符号ビット（10 ビット目）は 0 になっている．この符号ビットへの OSIM フォールトは分岐先を完全性検証部よりも下位のアドレスに改変する．デフォルトフェイルに従うと OutCTXT 関数は完全性検証部よりも下位のアドレスに配置されているため，偶然分岐先が OutCTXT に一致する場合に誤り暗号文が出力されてしまう．

■BAM に対する対策手法　上記のような BAM による攻撃を防ぐために，FaultHandler を OutCTXT と完全性検証部の間に配置する手法を提案する．FaultHandler 関数を完全性検証部よりも下位アドレスに配置することで分岐命令の符号ビットはもとから 1 になるため，OSIM モデルの攻撃者は符号ビットへのレーザー照射によって命令改変を引き起こせなくなる．分岐アドレスの他の位置（9 から 0 ビット目）への改変に関しても，分岐先をより上位にすることしかできないため，下位アドレスに配置してある OutCTXT へ分岐してしまうことを防げる．上記の手法を ORM-secure PIVD に適用した，ORM-BAM-secure PIVD 実装の例をリスト 5.4 に示す．リスト 5.3 と比べて 11 行目の分岐アドレス及

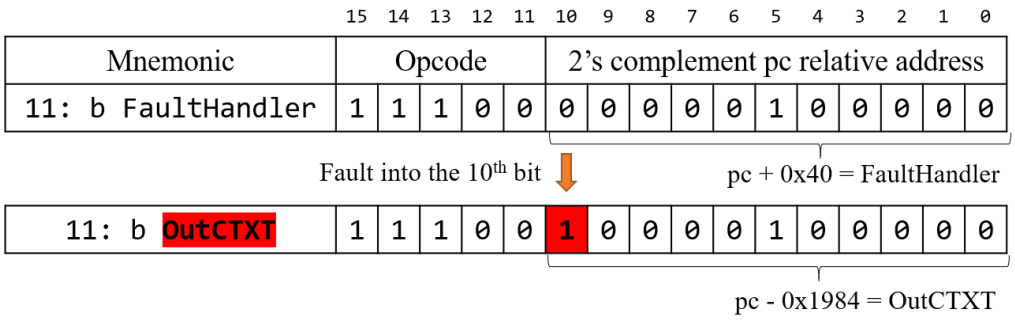


図 5.3: 無条件分岐命令の機械語フォーマット及び OutCTXT 関数を強制的に実行させる命令改変の例

び，FaultHandler 関数の位置が異なっている．

Listing 5.4: ORM-BAM-secure PIVD 実装

```
1  .extern FaultHandler
2  .extern OutCTXT
3  .func ORM_BAM_secure_PIVD
4  .thumb
5                       ;OutCTXT function is located at a lower address ↩
                            from here. FaultHandler function is located at ↩
                            an upper address from here.
6  ORM_BAM_secure_PIVD:
7      b LBL_MAIN
8  LBL_CMP:
9      cmp r3, r5
10     beq OutCTXT
11     b FaultHandler
12 LBL_MAIN:
13     b EncAndDec      ;Encrypt PTXT to CTXT, and decrypt CTXT to PTXT'.
14     ldr r5, [r3, #0]
15     ldr r3, [r3, #4]
16     mov r1, #1       ;Init Z flag to 0.
17     bne LBL_CMP
18
19 .endfunc
```

ロードアドレスの改変（Load address manipulations, LAM）

LAM には二つのタイプがあった．ここではそれぞれ LAM1, LAM2 と呼ぶ．図 5.4 は完全性検証される値をロードする ldr 命令（14 行目）のロード元アドレス部を改変することによる攻撃例を示している．このロード命令はベースレジスタ Rb の値にオフセットを足したアドレスの値をディスティネーションレジスタ Rd に読み込む．いま PTXT の値が r3 が示すアドレスに格納されているが，PTXT と PTXT′ が隣接するアドレスに格納されていた場合，図 5.4 のような改変を行うことで r5 及び r3 に同じ値が読み込まれ，完全性検証を常にバイパスすることができる．

図 5.5 はロード命令に対する命令改変攻撃のもう一つの例 LAM2 を示している．LAM2 の特徴は二つの連続する命令に改変が行われることである．OSIM の仮定により，攻撃者は同じビット位置なら任意の複数の命令に改変を行うことができる．改変は二つの ldr 命令の 7 ビット目（オフセットフィールド）に注入される．このとき二つの命令は PTXT, PTXT′ とは全く関連のないアドレスを参照しているが，それらのアドレスが未使用領域であった場合，qemu 環境だと 0 がロードされることがわかった．実デバイス上で同様の攻撃が可能かどうかは検討の余地が残るが，大きな配列を 0 で初期化している場合などはこのような状況があるかもしれない．図 5.5 の例は 7 ビット目にフォールトを注入しているが，他のオフセットフィールドやベースレジスタを改変することによっても同様の攻撃の可能性がある

■LAM に対する対策　ORM の対策と同じように，適切なオフセット値を選択することを LAM1 への対策として提案する．LAM1 のような，ロードアドレスの改変によって二つのロード命令のロード元アドレスが同じに改変される攻撃を防ぐためには，ロード命令のオフセット値が次の関係を満たす必要がある；

$$HD(Offset_{PTXT}, Offset_{PTXT'}) > 1,$$

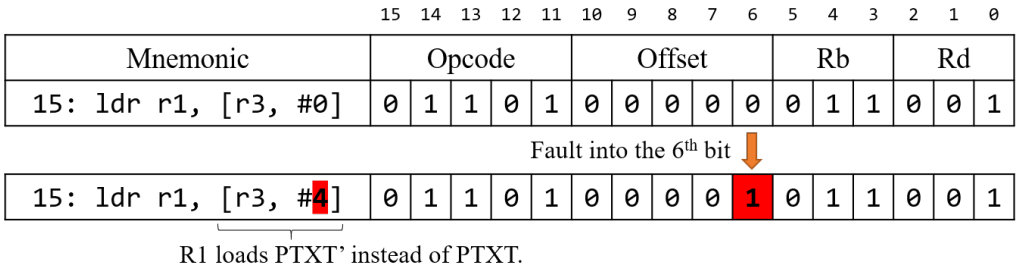


図 5.4: ロード命令の機械語フォーマットと，LAM1 による完全性検証バイパスための改変の例

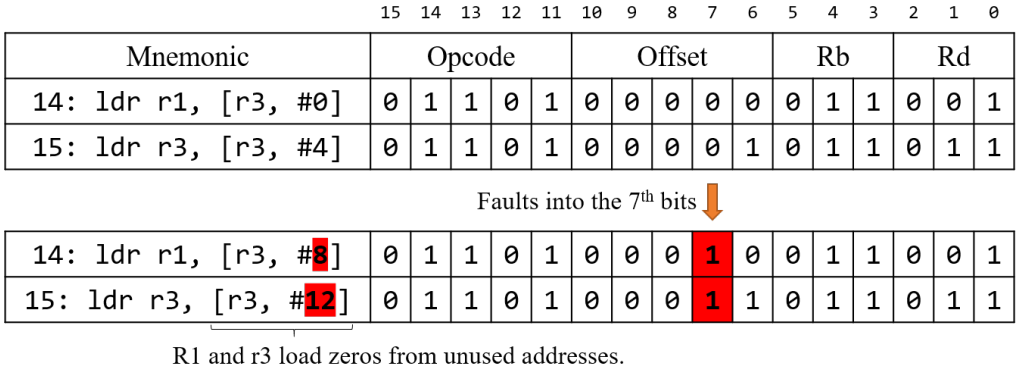


図 5.5: ロード命令の機械語フォーマットと，LAM2 による完全性検証バイパスための改変の例

ここで $HD(,)$ は二つの引数のハミング距離を返す関数であり，$Offset_{PTXT}$ は PTXT のベースアドレスからのオフセット値を示している．

LAN2 のような攻撃を防ぐために，未仕様のメモリ領域は乱数で初期化する対策を提案する．こうすることによって，ロード命令の読み出しアドレスが改変された場合に値が偶然一致する確率を小さくすることができる．

Listing 5.5: ORM-BAM-LAM-secure PIVD 実装

```
1  .extern FaultHandler
2  .extern OutCTXT
3  .func ORM_BAM_LAM_secure_PIVD
4  .thumb
5                      ;OutCTXT function is located at a lower address ↩
                           from here. FaultHandler function is located at ↩
                           an upper address from here.
6  ORM_BAM_LAM_secure_PIVD:
7      b LBL_MAIN
8  LBL_CMP:
9      cmp r3, r5
10     beq OutCTXT
11     b LBL_FAIL
12 LBL_MAIN:
13     b EncAndDec      ;Encrypt PTXT to CTXT, and decrypt CTXT to PTXT'.
14     ldr r5, [r3, #0]
```

```
15      ldr r3, [r3, #12]
16      mov r1, #1        ;Init Z flag to 0.
17      bne LBL_CMP
18
19  .endfunc
```

分岐条件の改変（Branch condition manipulation, BCM）

10 行目の beq 命令が bne 命令などに改変される攻撃である．条件分岐命令のフォーマットは図 5.6 に示すように 4bit の cond フィールドを持っており，cond=0x0 のとき Z=1 で分岐する beq 命令になる．cond フィールドの 1 ビット目がセットされることで beq 命令は例えば Z=0 の時に分岐する bne 命令に改変される．この結果，本来 PTXT == PTXT′ の時に LBL_SUCCESS へ分岐する処理のはずが，PTXT != PTXT′ のときに分岐するようになる．このような攻撃は分岐条件改変攻撃（BCM）と呼ばれており，すべての分岐条件に対する攻撃が可能であることがわかっている．

■BCM に対する対策　BCM は既に説明した命令改変攻撃と異なり，オペコード部分を改変する攻撃である．よって BCM に対する対策は少々トリッキーな方法が必要となる．BCM に対策を施した ORM-BAM-LAM-BCM-secure PIVD 実装をリスト 5.6 に示す．この実装はこれまでの対策と異なる，オリジナルの PIVD にいくつかの命令が追加されている．15–17 行目の命令は cmp 命令の代わりに利用される比較のための命令列であり，その結果に従って 18 行目の分岐命令が検算の成否を決定する．この分岐命令で比較が成功した場合 8 行目へ分岐するが，9–13 行目に BCM 攻撃を検知する命令列が配置されている．

BCM によって 18 行目の beq 命令が bne 命令に改変されたと仮定すると，Z=0 にも関わらず 9 行目へ至ってしまうが，このとき 9 行目の bne 命令の分岐条件も満たすため FaultHandler 関数へ分岐する．もし 9 行目実行中にレーザーが照射されたとしても，9 行目の 8 ビット目はもとから ‘1’ であるため改変とならず，正常に実行される．同じ事が他の条件分岐命令にもいえるため，9–12 行目に，18 行目の分岐

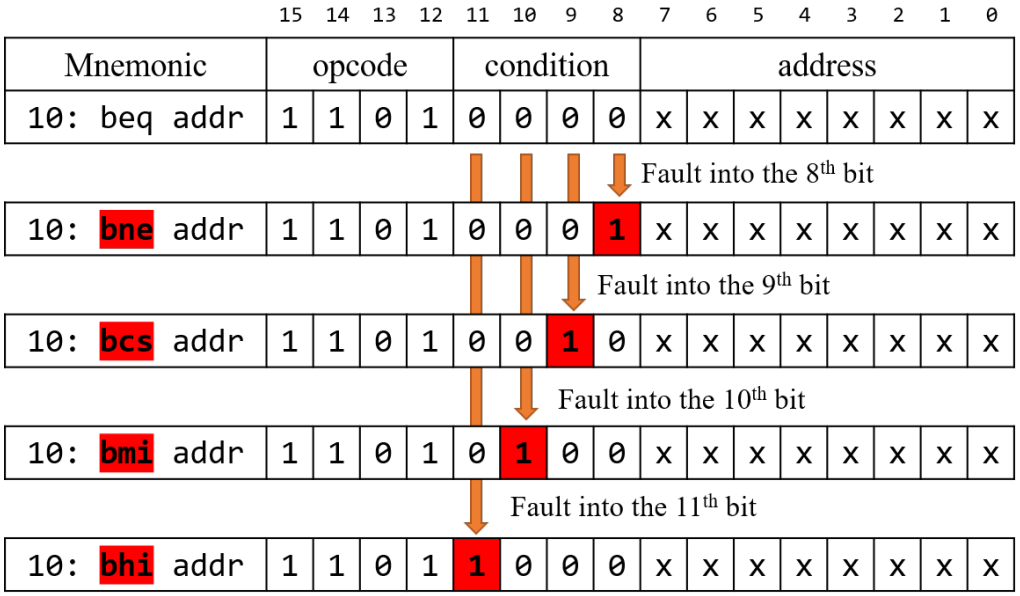


図 5.6: 条件分岐命令の機械語フォーマット及び分岐条件の改変による攻撃

命令が改変されうる全ての分岐命令を配置して FaultHandler 関数へ分岐させている．

続いて cmp 命令を用いて比較を行わない理由を説明する．cmp 命令による比較はレジスタの値が等しい場合 (Z, N, C) に (1, 0, 1) をセットするが，9-12 行目に複数の分岐命令を配置したことにより，(Z, N, C) = (1, 0, 0) のときのみ 13 行目が実行され OutCTXT へ至る．よってレジスタの値が等しい場合にのみ (Z, N, C) = (1, 0, 0) をセットする処理が必要である．eor 命令はこの目的に合うフラグレジスタの更新を行うが，オペコードの改変に脆弱であったため採用を見送った．リスト 5.6 は複数命令（17 − 19 行目）を使って値が等しいときに (Z, N, C) = (1, 0, 0) となる比較処理を実現している．この命令列は 1 の補数を足したものの 1 の補数を計算しており，1 の補数が桁上がりの発生しない最大の数であることを考えると値が等しい時のみ 0 になる．

Listing 5.6: ORM-BAM-LAM-BCM-secure PIVD 実装

```
1  .extern FaultHandler
2  .extern OutCTXT
3  .func ORM_BAM_LAM_BCM_secure_PIVD
4  .thumb
5                      ;OutCTXT function is located at a lower address ↩
                           from here. FaultHandler function is located at ↩
                           an upper address from here.
6  ORM_BAM_LAM_BCM_secure_PIVD:
7      b LBL_MAIN
8  LBL_COND_BRANCH:
9      bne FaultHandler
10     bcs FaultHandler
11     bmi FaultHandler
12     bhi FaultHandler
13     beq OutCTXT
14 LBL_CMP:
15     mvn r3, r3          ;|Only if r3 == r5, |
16     add r3, r5          ;|                   |
17     mvn r3, r3          ;|set Z flag to 1.  |
18     beq LBL_COND_BRANCH
19     b FaultHandler
20 LBL_MAIN:
21     b EncAndDec         ;Encrypt PTXT to CTXT, and decrypt CTXT to PTXT'.
22     mov r5, #1          ;Init Z flag to 0.
23     ldr r5, [r3, #0]
24     ldr r3, [r3, #12]
25     bne LBL_CMP
26
27 .endfunc
```

5.3.6　シミュレータによる命令改変対策の検証

前項で提案した命令改変対策を適用した各 PIVD 実装に対してシミュレータによる命令改変攻撃を行った結果を表 5.2 に示す．ここで EXP4, EXP11, EXP19 はそれぞれ未定義命令，セグメンテーションフォールト，タイムアウトを示す．誤り暗号文出力数は対策を重ねるに従い減少していることがわかり，

表 5.2: OSIM 対策を適用した PIVD 実装に対するシミュレータによる命令改変攻撃結果

攻撃対象	誤り暗号文出力	フォールト検知	EXP4	EXP11	EXP19	n	N
Original PIVD	112	1056	2080	847	1	8	4096
ORM-secure PIVD	64	1137	2080	815	0	8	4096
ORM-BAM-secure PIVD	48	1008	2112	928	0	8	4096
ORM-BAM-LAM-secure PIVD	20	1092	2044	940	0	8	4096
ORM-BAM-LAM-BCM-secure PIVD	0	129739	286396	107286	851	15	524288

それぞれの対策が有効である事を示している．ORM-BAM-LAM-BCM-secure PIVD に至っては N が巨大であるにもかかわらず誤り暗号文が出力されなくなっており，有効な対策だといえる．

5.3.7　命令改変攻撃対策のコスト

レーザー攻撃に対する本質的な対策はレーザーセンサなどのハードウェアを追加することだが，ハードウェアの修正を要する対策はコストが高く，また既存のデバイスには適用できないという問題がある．本論文で提案した命令改変攻撃対策はソフトウェアのコーディング方法のみに基づいているため，低コストで実装可能だと考える．

ORM, LAM1, BAM に対する対策は命令のオペランドの選び方のみを修正するためほとんどコストなしで実装可能である．利用レジスタの量がわずかに増え，また関数を配置するアドレスに制約を課すが，ほぼ無視できるコストである．一方で，BCM, LAM2 に対する対策はコストの増加が考えられる．BCM に対する対策はオリジナルの PIVD にいくつかの命令を追加するため，実行速度・プログラムフットプリントの面でコストが増える．しかしながらこれらの増加は暗号処理の実行時間やフットプリントに比べてかなり小さいと考えられる．

LAM2 に対する対策である未使用メモリ領域の初期化は，領域の広さにもよるが実行時間という面で相応のコストがかかると考えられる．しかしながらこの対策が必要かどうかは，デバイスが未使用の RAM 領域をどう扱っているかというデバイス依存であり，一般的にデバイスの起動時の SRAM の状態は不定になるため，qemu のように 0 初期化されるという状況は少ない可能性がある．もしメモリランダム化が必要な場合でも，格納する値に高いランダム性は必要なく，異なる値が格納されるだけで十分であり，デバイスの起動時に一度だけ実行するなどでオーバーヘッドを抑えることができると考えている．

第 6 章

総括

本論文では，IoT 社会に増加する実装攻撃の中でも特に大きな脅威となり得るレーザーフォールト攻撃に関して，利用されるレーザー装置とそのコスト，攻撃対象となるデバイス，デバイスにレーザーを照射した際の影響，レーザーフォールトに対する対策の 4 つを体系的に示した．コモンクライテリアのようなセキュリティ評価・認証スキームにおいて，評価対象の評価レベルを保証するため既知の攻撃が評価対象デバイスに実行される．よってセキュリティ評価をより強固なものにするためには，未知の強力な攻撃を攻撃者に先んじて発見し評価スキームに組み込むことが重要である．この目的のため，命令改変という未知の攻撃の存在を示したことが本論文の成果の一つである．またコモンクライテリアにて高い評価レベルを得るためには，攻撃が成功するまでに要したコストが十分高いことが求められる．従って，ある攻撃手法が現実的かを評価するためには，その攻撃を行うのに必要なコストの下限を適切に見積もることが重要となる．本論文で提案した命令改変攻撃は一般的には高コストだといわれるレーザー照射を利用するが，低コストで構築可能なレーザー装置を使い，回路レイアウトのような重要な情報を知らずとも容易に注入できることを示したのが二つ目の成果である．さらに，命令改変攻撃に対するソフトウェア上での対策手法を提案し，ハードウェアによる耐タンパー機構を実装できないような低コスト機器のセキュリティレベルを引き上げた事が三つめの成果である．

本論文の各章の内容を以下にまとめる．

第 1 章では，フォールト攻撃とそれを取り巻く関連研究が導入され，IoT 時代における実装攻撃の脅威を解説した．実装攻撃のなかでもフォールト攻撃、特にレーザーフォールト攻撃が強力であるであることを延べ，攻撃対策の必要性を説いた．

第 2 章ではデバイスにレーザーを照射するための機器について紹介した．4 つのレーザー照射装置が解説され，市販品が約 1800 万円である一方，本研究で構築したレーザー装置は 400 万円弱で精密なレーザー照射が可能であることがわかった．またレーザーを同時に 2 カ所に照射可能なダブルレーザー装置も構築され，これは 1000 万円弱で構築できた．

第 3 章ではレーザーフォールト攻撃の対象となり得るデバイスの構成が解説された．抽象化された組込みデバイスは CPU, ROM, RAM, コプロセッサがバスで接続された構成になっており，これらにレーザーが照射されることを述べ，CPU のアーキテクチャの例として AVR 及び ARM アーキテクチャを説明した．またコプロセッサの例としてペアリング演算専用プロセッサを開発し，このような専用コプロセッサもレーザー攻撃の対象になり得ることを示した．さらに，IC チップのパッケージング技術についても述べられた．レーザー攻撃を行うためにはパッケージを開封して IC チップを露出させる必要があるが，フリップチップ ya COB タイプの裏面開封はほとんど開封の手間がないことを述べた．

第 4 章では，ここまでで導入された攻撃対象及びレーザー装置を使ってレーザー照射の影響を調査し

た．従来の研究で多くのフォールト注入手法はプロセッサで実行されている命令をスキップできることが示されていたが，より強力な命令改変フォールトが AVR ATMega163 及び ARM SC100 上で生じることを示した．またレーザーのパラメータを調整することで命令改変を制御することができ，ペアリング暗号のような高度な公開鍵暗号や，既存のフォールト対策実装を命令改変で攻撃した結果を示した．さらに命令改変攻撃はプログラムを知っている攻撃者にとってより現実的な攻撃方法となるため，レーザー照射によって ROM 中の値を読み出す攻撃も提案された．

第 5 章ではレーザー攻撃に対する対策が述べられた．ハードウェアによる対策は効果的だがコストがかかり，低コスト IoT 機器のようなデバイスにはソフトウェアによる低コストな対策のニーズがある．これまでのレーザー攻撃に対する対策はハードウェアによるものが多かったが，命令改変攻撃に対してはレーザースポット数などの制限を与えることによってソフトウェアでの命令改変攻撃対策を提案した．提案対策は命令改変をシミュレートする環境で評価され，従来の対策手法と比べて誤り暗号文を出力する回数が減ったことが示された．

本論文によって，以下のような今後の発展性が明らかになった．本論文で構築したレーザー装置は市販品に比べると低コストであるが，それでもまだコスト削減可能な箇所があると考えている．例えばレーザー光源として Aphanov 社の PDM レーザーを利用しているが，これはフォールト解析用の専用品でありかなり高品質・高強度なレーザーである．しかしながら本論文の結果から命令改変を注入するには PDM レーザーの最大強度の数割程度の強度で十分である事がわかった．より強度が弱い光源を利用することでコストを抑えた命令改変が実行可能になる可能性がある．本論文では，ハードウェアによるレーザー攻撃が実装されていない対象に対して低コストで強力なレーザー攻撃が可能である事を示した．ハードウェアによるレーザー攻撃対策が実装されているような高価値なデバイスを攻撃するには，より高価なレーザー装置や攻撃者の知識が求められると考えられる．

本論文では低レイテンシペアリングアーキテクチャを提案したが，このアーキテクチャのパイプライン段数を減らすなどして回路量を減らすことで低コストな IoT 機器にも搭載できないかと考えている．低コスト IoT 機器は性能上の制約から公開鍵暗号のようなリッチな機能を実装できず，ネットワーク全体で同一の鍵を共有して運用している場合があり，一つのデバイスからの鍵流出がネットワーク全体の脅威になる．もし低コストなデバイスで公開鍵暗号を利用できるようになり，それぞれのデバイスが異なる鍵を持つようになればサイドチャネル攻撃によるリスクを低下させることができると考える．

謝辞

本研究を進めるにあたり日頃から有益なご指導，アイデアを頂きました横浜国立大学大学院環境情報学府 松本勉指導教授に謝辞を述べます．松本教授からは研究へ取り組む姿勢からわかりやすい図や発表の作り方まで，およそ研究に関わるすべての部分でご指導いただきました．学部 4 年次から 6 年間の長きにわたり未熟な私をご指導、時にはご鞭撻いただいたおかげで，本研究をまとめることができました．深く感謝致します．

本研究を進めるにあたり多大なご意見とご助言を頂きました，横浜国立大学大学院環境情報学府 四方順司教授，吉岡克成准教授に感謝致します．お二人からは IPS 研究会の場で有益なコメントを頂き，またその他の場所でも気にかけて頂きました．特に四方先生には私が四方研究室に席を置いていたこともあり大変お世話になりました．吉岡准教授には学部時代のコンタクト教員の頃から長きにわたり，非常勤講師として雇用していただくなど大変お世話になりました．

本論文の審査に加わってくださり，貴重なご意見を頂きました横浜国立大学大学院環境情報学府 森辰則教授，白川真一講師に深く感謝致します．特に森教授には，私が本学へ編入した際の単位認定などでお世話になり，教授のおかげでここまで問題なく進学することができました．

本研究を進めるに当たって議論や助言を頂きました横浜国立大学 情報・物理セキュリティ研究拠点 藤本大介客員助教，吉田直樹特任助教に感謝致します．藤本助教からは本論文に関わる多くの部分において技術的なご助言を頂きました．吉田助教からは研究室の良い先輩として，学部 4 年次からの長きにわたり研究に対するご意見や研究室生活のサポートをして頂きました．

さらに毎日の研究室でお世話になりました成松未央秘書，石舘知子技術補佐員，川村恵美子技術補佐員に感謝致します．皆様のサポートなくして研究を進めることはできませんでした．

また毎週の輪講でご助力，ご協力頂きました松本勉研究室，四方研究室，吉岡研究室の皆様に感謝致します．特に本研究の初期の方針を支えてくださった大野仁先輩，中田量子先輩に深く感謝致します．

最後に，これまで私を育て上げ，博士後期課程までの研究生活を応援してくれた両親，家族に深く感謝します．

Fundings and Supports

本研究の一部は，セコム科学技術振興財団の研究助成を受けて行われた．同財団の支援に感謝します．

本研究の一部は，内閣府が進める戦略的イノベーション創造プログラム（ＳＩＰ）「ＩｏＴ社会に対応したサイバー・フィジカル・セキュリティ」（管理法人：NEDO）によって実施されたものである．

本研究の一部は，国立研究開発法人新エネルギー・産業技術総合開発機構（NEDO）の委託業務の結果得られたものである．同機構の支援に感謝します．

本研究の実験の一部については情報処理推進機構 (IPA) の設備を使って行われた．関係者のサポート及び研究への助言について深く感謝します．

本論文で示した実験の対象デバイス TVC は情報処理推進機構 (IPA) セキュリティセンターから貸与されたものである．関係各位のご協力に感謝します．

参考文献

[1] Alphanov, "Single laser fault injection microscope - s-lms," 2019/12/12. [Online]. Available: https://www.alphanov.com/en/news/alphanov-has-designed-four-point-laser-rig-laser-fault-injections-integrated-circuits

[2] F. Amiel, K. Villegas, B. Feix, and L. Marcel, "Passive and active combined attacks: Combining fault attacks and side channel analysis," in *Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC 2007)*, Sep. 2007, pp. 92–102.

[3] D. F. Aranha, P. S. L. M. Barreto, P. Longa, and J. E. Ricardini, "The realm of the pairings," in *Selected Areas in Cryptography – SAC 2013*, T. Lange, K. Lauter, and P. Lisoněk, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 3–25.

[4] ARM, *ARM Architecture Reference Manual*, https://cs.nyu.edu/courses/spring18/CSCI-GA.2130-001/ARM/arm_arm.pdf.

[5] ARM, *ARM Architecture Reference Manual Thumb-2 Supplement*, http://class.ece.iastate.edu/cpre288/resources/docs/Thumb-2SupplementReferenceManual.pdf.

[6] ATMEL, *8-bit AVR Microcontroller with 8K Bytes In-System Programmable Flash ATmega8515 ATmega8515L*, 2010. [Online]. Available: http://ww1.microchip.com/downloads/en/devicedoc/doc2512.pdf

[7] K. Bae, S. Moon, and J. Ha, "Instruction fault attack on the miller algorithm in a pairing-based cryptosystem," in *2013 Seventh International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, July 2013, pp. 167–174.

[8] J. Balasch, B. Gierlichs, and I. Verbauwhede, "An in-depth and black-box characterization of the effects of clock glitches on 8-bit mcus," in *2011 Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2011, Tokyo, Japan, September 29, 2011*, 2011, pp. 105–114. [Online]. Available: https://doi.org/10.1109/FDTC.2011.9

[9] H. Bar-El, H. Choukri, D. Naccache, M. Tunstall, and C. Whelan, "The sorcerer's apprentice guide to fault attacks," *Proceedings of the IEEE*, vol. 94, no. 2, pp. 370–382, Feb 2006.

[10] A. Barenghi, L. Breveglieri, I. Koren, G. Pelosi, and F. Regazzoni, "Countermeasures against fault attacks on software implemented AES: effectiveness and cost," in *WESS*. ACM, 2010, p. 7.

[11] N. Beringuier-Boher, M. Lacruche, D. El-Baze, J.-M. Dutertre, J.-B. Rigaud, and P. Maurine, "Body biasing injection attacks in practice," in *Proceedings of the Third Workshop on Cryptography and Security in Computing Systems*, ser. CS2 '16. New York, NY, USA: ACM,

2016, pp. 49–54. [Online]. Available: http://doi.acm.org/10.1145/2858930.2858940

[12] I. Biehl, B. Meyer, and V. Müller, "Differential fault attacks on elliptic curve cryptosystems," in *Advances in Cryptology — CRYPTO 2000*, M. Bellare, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 131–146.

[13] J. Blömer, R. G. d. Silva, P. Günther, J. Krämer, and J. Seifert, "A practical second-order fault attack against a real-world pairing implementation," in *2014 Workshop on Fault Diagnosis and Tolerance in Cryptography*, Sep. 2014, pp. 123–136.

[14] D. Boneh, R. A. DeMillo, and R. J. Lipton, "On the importance of checking cryptographic protocols for faults," in *Advances in Cryptology — EUROCRYPT '97*, W. Fumy, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, pp. 37–51.

[15] J. Borghoff, A. Canteaut, T. Güneysu, E. B. Kavun, M. Knezevic, L. R. Knudsen, G. Leander, V. Nikov, C. Paar, C. Rechberger, P. Rombouts, S. S. Thomsen, and T. Yalçın, "Prince – a low-latency block cipher for pervasive computing applications," in *Advances in Cryptology – ASIACRYPT 2012*, X. Wang and K. Sako, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 208–225.

[16] J. Breier and D. Jap, "Testing feasibility of back-side laser fault injection on a microcontroller," in *Proceedings of the WESS'15: Workshop on Embedded Systems Security*, ser. WESS'15. New York, NY, USA: ACM, 2015, pp. 5:1–5:6. [Online]. Available: http://doi.acm.org/10.1145/2818362.2818367

[17] S. Chatterjee, K. Karabina, and A. Menezes, "Fault attacks on pairing-based protocols revisited," *IEEE Transactions on Computers*, vol. 64, no. 6, pp. 1707–1714, June 2015.

[18] B. Colombier, A. Menu, J.-M. Dutertre, P.-A. Moëllic, J.-B. Rigaud, and J.-L. Danger, "Laser-induced single-bit faults in flash memory: Instructions corruption on a 32-bit microcontroller," Cryptology ePrint Archive, Report 2018/1042, https://eprint.iacr.org/2018/1042, 2018.

[19] Cypress, *Cypress Roadmap: Flash Memory Q2 2019*, https://www.cypress.com/file/206951/download.

[20] A. Dehbaoui, A.-P. Mirbaha, N. Moro, J.-M. Dutertre, and A. Tria, "Electromagnetic glitch on the aes round counter," in *Constructive Side-Channel Analysis and Secure Design*, E. Prouff, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 17–31.

[21] O. Derouet, "Secure smartcard design against laser fault injection," in *4th Workshop on Fault Diagnostic and Tolerance in Cryptography, Vienne, Autriche*, 2007, p. 87.

[22] J. Di-Battista, J.-C. Courrege, B. Rouzeyre, L. Torres, and P. Perdu, "When failure analysis meets side-channel attacks," in *Cryptographic Hardware and Embedded Systems, CHES 2010*, S. Mangard and F.-X. Standaert, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 188–202.

[23] N. El Mrabet, "Fault attack against miller's algorithm," *IACR Cryptology ePrint Archive*, vol. 2011, p. 709, 01 2011.

[24] S. Endo, N. Homma, Y. Hayashi, J. Takahashi, H. Fuji, and T. Aoki, "A multiple-fault injection attack by adaptive timing control under black-box conditions and a countermeasure," in *COSADE*, ser. Lecture Notes in Computer Science, vol. 8622. Springer, 2014, pp. 214–228.

[25] S. Ghosh, I. Verbauwhede, and D. Roychowdhury, "Core based architecture to speed up optimal

ate pairing on fpga platform," in *Pairing-Based Cryptography – Pairing 2012*, M. Abdalla and T. Lange, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 141–159.

[26] C. Giraud, "Dfa on aes," in *Advanced Encryption Standard – AES*, H. Dobbertin, V. Rijmen, and A. Sowa, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 27–41.

[27] O. M. Guillen, M. Gruber, and F. De Santis, "Low-cost setup for localized semi-invasive optical fault injection attacks," in *Constructive Side-Channel Analysis and Secure Design*, S. Guilley, Ed. Cham: Springer International Publishing, 2017, pp. 207–222.

[28] J. Han, Y. Li, Z. Yu, and X. Zeng, "A 65 nm cryptographic processor for high speed pairing computation," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 4, pp. 692–701, April 2015.

[29] K. Karabina, "Squaring in cyclotomic subgroups," *IACR Cryptology ePrint Archive*, vol. 2010, p. 542, 01 2010.

[30] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Advances in Cryptology — CRYPTO' 99*, M. Wiener, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 388–397.

[31] S. V. D. Kumar, S. Patranabis, J. Breier, D. Mukhopadhyay, S. Bhasin, A. Chattopadhyay, and A. Baksi, "A practical fault attack on arx-like ciphers with a case study on chacha20," in *FDTC*. IEEE Computer Society, 2017, pp. 33–40.

[32] R. Lashermes, M. Paindavoine, N. El Mrabet, J. J. A. Fournier, and L. Goubin, "Practical validation of several fault attacks against the miller algorithm," in *2014 Workshop on Fault Diagnosis and Tolerance in Cryptography*, Sep. 2014, pp. 115–122.

[33] Y. Li, J. Han, S. Wang, D. Fang, and X. Zeng, "An 800mhz cryptographic pairing processor in 65nm cmos," in *2012 IEEE Asian Solid State Circuits Conference (A-SSCC)*, Nov 2012, pp. 217–220.

[34] J. I. Library, "Application of attack potential to smartcards and similar devices," 2019. [Online]. Available: https://www.sogis.eu/documents/cc/domains/sc/JIL-Application-of-Attack-Potential-to-Smartcards-v3-0.pdf

[35] K. Matsuda, T. Fujii, N. Shoji, T. Sugawara, K. Sakiyama, Y. Hayashi, M. Nagata, and N. Miura, "A 286f2/cell distributed bulk-current sensor and secure flush code eraser against laser fault injection attack," in *2018 IEEE International Solid - State Circuits Conference - (ISSCC)*, Feb 2018, pp. 352–354.

[36] A. Menu, S. Bhasin, J.-M. Dutertre, J.-B. Rigaud, and J.-L. Danger, "Precise spatio-temporal electromagnetic fault injections on data transfers," 08 2019, pp. 1–8.

[37] N. Moro, A. Dehbaoui, K. Heydemann, B. Robisson, and E. Encrenaz, "Electromagnetic fault injection: towards a fault model on a 32-bit microcontroller," *CoRR*, vol. abs/1402.6421, 2014. [Online]. Available: http://arxiv.org/abs/1402.6421

[38] N. Moro, K. Heydemann, E. Encrenaz, and B. Robisson, "Formal verification of a software countermeasure against instruction skip attacks," *J. Cryptographic Engineering*, vol. 4, no. 3, pp. 145–156, 2014.

[39] A. News, "Alphanov has designed a four-point laser rig for laser fault injections on integrated circuits." [Online]. Available: https://www.alphanov.com/en/products-services/

single-laser-fault-injection

[40] H. Orup, "Simplifying quotient determination in high-radix modular multiplication," in *Proceedings of the 12th Symposium on Computer Arithmetic*, July 1995, pp. 193–199.

[41] D. Petryk, Z. Dyka, and P. Langendoerfer, "Optical fault injections: Most often used setups," 09 2018.

[42] Petryk, Dmytro and Dyka, Zoya and Langendoerfer, Peter, "Optical fault injections: a setup comparison," 06 2018.

[43] Riscure, "icwaves." [Online]. Available: https://www.riscure.com/product/icwaves/

[44] Riscure, "Laser station 2." [Online]. Available: https://getquote.riscure.com/en/quote/2101106/laser-station-2.htm

[45] L. Riviére, Z. Najm, P. Rauzy, J. Danger, J. Bringer, and L. Sauvage, "High precision fault injections on the instruction cache of armv7-m architectures," *CoRR*, vol. abs/1510.01537, 2015. [Online]. Available: http://arxiv.org/abs/1510.01537

[46] J. Rodriguez, A. Baldomero, V. Montilla, and J. Mujal, "Llfi: Lateral laser fault injection attack," in *2019 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC)*, Aug 2019, pp. 41–47.

[47] D. Samyde, S. Skorobogatov, R. Anderson, and J. . Quisquater, "On a new way to read data from memory," in *First International IEEE Security in Storage Workshop, 2002. Proceedings.*, Dec 2002, pp. 65–69.

[48] J. Schmidt and C. Herbst, "A practical fault attack on square and multiply," in *2008 5th Workshop on Fault Diagnosis and Tolerance in Cryptography*, Aug 2008, pp. 53–58.

[49] S. Skorobogatov, "Optical fault masking attacks," in *2010 Workshop on Fault Diagnosis and Tolerance in Cryptography*, Aug 2010, pp. 23–29.

[50] S. Skorobogatov, "Semi-invasive attacks-a new approach to hardware security analysis," *Technical report*, 01 2005. [Online]. Available: https://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-630.pdf

[51] S. Skorobogatov, "Optically enhanced position-locked power analysis," in *Cryptographic Hardware and Embedded Systems - CHES 2006*, L. Goubin and M. Matsui, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 61–75.

[52] S. Skorobogatov, "Flash memory 'bumping' attacks," in *CHES*, ser. Lecture Notes in Computer Science, vol. 6225. Springer, 2010, pp. 158–172.

[53] B. studios, "Hacking the pic 18f1320," 2007. [Online]. Available: https://www.bunniestudios.com/blog/?page_id=40

[54] E. Trichina and R. Korkikyan, "Multi fault laser attacks on protected CRT-RSA," in *2010 Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2010, Santa Barbara, California, USA, 21 August 2010*, 2010, pp. 75–86. [Online]. Available: https://doi.org/10.1109/FDTC.2010.14

[55] J. G. J. van Woudenberg, M. F. Witteman, and F. Menarini, "Practical optical fault injection on secure microcontrollers," in *2011 Workshop on Fault Diagnosis and Tolerance in Cryptography*, Sep. 2011, pp. 91–99.

[56] I. Verbauwhede, D. Karaklajic, and J. Schmidt, "The fault attack jungle - a classification model

to guide you," in *2011 Workshop on Fault Diagnosis and Tolerance in Cryptography*, Sep. 2011, pp. 3–8.

[57] G. Yao, J. Fan, R. C. Cheung, and I. Verbauwhede, "A high speed pairing coprocessor using rns and lazy reduction." *IACR Cryptology ePrint Archive*, vol. 2011, p. 258, 01 2011.

[58] G. X. Yao, J. Fan, R. C. C. Cheung, and I. Verbauwhede, "Faster pairing coprocessor architecture," in *Pairing-Based Cryptography – Pairing 2012*, M. Abdalla and T. Lange, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 160–176.

[59] B. Yuce, N. F. Ghalaty, H. Santapuri, C. Deshpande, C. Patrick, and P. Schaumont, "Software fault resistance is futile: Effective single-glitch attacks," in *2016 Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2016, Santa Barbara, CA, USA, August 16, 2016*, 2016, pp. 47–58. [Online]. Available: https://doi.org/10.1109/FDTC.2016.21

[60] F. Zhang, X. Lou, X. Zhao, S. Bhasin, W. He, R. Ding, S. Qureshi, and K. Ren, "Persistent fault analysis on block ciphers," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2018, no. 3, pp. 150–172, Aug. 2018. [Online]. Available: https://tches.iacr.org/index.php/TCHES/article/view/7272

[61] 山根彰，*AVR* マイコン・リファレンス・ブック 第 *4* 版． CQ 出版, 2011.

[62] 独立行政法人情報処理推進機構 技術本部 セキュリティセンター情報セキュリティ認証室，"情報技術セキュリティ評価のためのコモンクライテリアパート 1: 概説と一般モデル バージョン 3.1 改訂第 5 版"，2017. [Online]. Available: https://www.ipa.go.jp/security/jisec/cc/documents/CCPART1V3.1R5-J1.0.pdf

[63] 独立行政法人情報処理推進機構 技術本部 セキュリティセンター情報セキュリティ認証室，"情報技術セキュリティ評価のためのコモンクライテリアパート 2: セキュリティ機能コンポーネント バージョン 3.1 改訂第 5 版"，2017. [Online]. Available: https://www.ipa.go.jp/security/jisec/cc/documents/CCPART2V3.1R5-J1.0.pdf

[64] 独立行政法人情報処理推進機構 技術本部 セキュリティセンター情報セキュリティ認証室，"情報技術セキュリティ評価のためのコモンクライテリアパート 3: セキュリティ保証コンポーネント バージョン 3.1 改訂第 5 版"，2017. [Online]. Available: https://www.ipa.go.jp/security/jisec/cc/documents/CCPART3V3.1R5-J1.0.pdf

[65] 独立行政法人情報処理推進機構 技術本部 セキュリティセンター情報セキュリティ認証室，"情報技術セキュリティ評価のための共通方法 バージョン 3.1 改訂第 5 版"，2017. [Online]. Available: https://www.ipa.go.jp/security/jisec/cc/documents/CEMV3.1R5-J1.0.pdf

[66] 総務省，"平成 30 年版情報通信白書"，2018. [Online]. Available: http://www.soumu.go.jp/johotsusintokei/whitepaper/ja/h30/pdf/n1100000.pdf

研究内容の公表

本研究を構成する公表論文

■査読付き論文誌および国際会議論文

[Pub. 1] J. Sakamoto, D. Fujimoto and T. Matsumoto, “Laser-Induced Controllable Instruction Replacement Fault Attack,” IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences, Vol.E103-A, No.01, Jan. 2020.

[Pub. 2] J. Sakamoto, Y. Nagahama, D. Fujimoto, Y. Okuaki and T. Matsumoto, “Low-Latency Pairing Processor Architecture Using Fully-Unrolled Quotient Pipelining Montgomery Multiplier,” in Proc. of 2019 IEEE Asian Hardware-Oriented Security and Trust (AsianHOST), Xi’an, 2019.

[Pub. 3] J. Sakamoto, D. Fujimoto and T. Matsumoto, “Laser irradiation on EEPROM sense amplifiers enhances side-channel leakage of read bits,” in Proc. of 2016 IEEE Asian Hardware-Oriented Security and Trust (AsianHOST), Yilan, 2016.

■査読無し論文

[Pub. 4] 林俊吾，坂本純一，松本勉，“シングルスポットレーザーによる命令改変攻撃に耐えられるインテグリティ検証プログラムの構成法，” 2020 年暗号と情報セキュリティシンポジウム SCIS2020 予稿集 3E2-3，2020.

[Pub. 5] 鈴木朋郎，坂本純一，松本勉，“ダブルレーザー照射装置を用いた TVC に対する命令置換フォールト攻撃，” 信学技報, vol. 119, no. 143, HWS2019-33, pp. 221-225, 2019 年 7 月.

[Pub. 6] 山崎満文，坂本純一，奥秋陽太，松本　勉，“パイプライン型剰余乗算器を用いたペアリング計算 FPGA のサイドチャネルセキュリティ評価”，信学技報, vol. 119, no. 143, HWS2019-24, pp. 151-156, 2019 年 7 月.

[Pub. 7] 坂本純一，衣川昌宏，藤本大介，林　優一，松本　勉，“無線モジュールの送信波が誘発する情報漏洩の分析”，信学技報, vol. 119, no. 2, HWS2019-5, pp. 25-29, 2019 年 4 月，ハードウェアセキュリティ研究会若手優秀賞.

[Pub. 8] 山崎満文，坂本純一，松本　勉，“ソフトウェア実装ペアリング計算における乱数マスク型サイドチャネル攻撃対策の評価”, 暗号と情報セキュリティシンポジウム 2019, 3D3-3.

[Pub. 9] 宮地遼，坂本純一，松本　勉，“数論変換を用いた Ring-LWE 暗号のソフトウェア実装に対する高次相関電力解析攻撃”, 暗号と情報セキュリティシンポジウム 2019, 2D3-5.

[Pub. 10] 坂本純一，松本勉，“命令置換レーザーフォールト攻撃の ARM プロセッサ上ペアリング実装に対する効果，” 信学技報, vol. 118, no. 272, HWS2018-54, pp. 41-46, 2018 年 10 月,

ハードウェアセキュリティ研究会若手優秀賞.

[Pub. 11] 坂本純一，宮地遼，松本　勉，“ソフトウェア実装ペアリング暗号に対する電磁波解析攻撃の評価”，暗号と情報セキュリティシンポジウム 2018, 1D2-5.

[Pub. 12] 奥冨賢哉，坂本純一，松本　勉，“希出現波形に着目したテンプレート攻撃とその評価”, 暗号と情報セキュリティシンポジウム 2018, 3D2-5.

[Pub. 13] 奥冨賢哉，坂本純一，松本　勉，“すべての部分鍵候補のテンプレートを利用する自己プロファイリング型サイドチャネル攻撃”, 信学技報 ISSN 2432-6380.

[Pub. 14] 奥冨賢哉，坂本純一，吉田直樹，藤本大介，松本　勉，“ガウス過程に基づくテンプレート攻撃の AES 実装に対する評価”，2017 年暗号と情報セキュリティシンポジウム SCIS2017 3C1-1，2017.

[Pub. 15] 坂本純一，奥冨賢哉，松本　勉，“攻撃対象暗号モジュールを直接プロファイリングするサイドチャネル攻撃”，2017 年電子情報通信学会総合大会 AS-3-10.

[Pub. 16] 坂本純一，藤本大介，松本勉，“レーザー照射と電力解析によるメモリ読出し値の抽出”，2016 年電子情報通信学会ソサイエティ大会 予稿集 A-7-10，2016.

[Pub. 17] 坂本純一，大野仁，土屋遊，中田量子，松本勉，“命令置換フォールト攻撃とその対策，”2015 年暗号と情報セキュリティシンポジウム SCIS2015 予稿集 2F3-5，2015.

その他

■査読付き論文誌および国際会議論文

[Pub. 18] A. Prasitsupparote, Y. Watanabe, J. Sakamoto, J. Shikata, and T. Matsumoto, “Implementation and Analysis of Fully Homomorphic Encryption in Resource-Constrained Devices”, International Journal of Digital Information and Wireless Communications (IJDIWC), Volume 8, Issue 4, pp. 288-303, 2019.

[Pub. 19] K. Shirai, T. Kiyokawa, J. Sakamoto, T. Toyama, T. Matsumoto, “Real-time Electrical Data Forgery in In-vehicle Controller Area Network Bus,” in Proc. of escar Asia 2018 world leading automotive security conference, 2018.

■査読無し論文

[Pub. 20] 奥秋陽太，坂本純一，藤本大介，松本　勉，“パイプライン型剰余乗算器を用いたペアリング暗号の FPGA 実装 ～ 集約署名の場合 ～”，信学技報, vol. 119, no. 143, HWS2019-25, pp. 157-162, 2019 年 7 月.

[Pub. 21] 外山　拓，坂本純一，吉田直樹，松本　勉，“USB 機器の電圧変化による個体識別の可能性”，信学技報, vol. 119, no. 143, HWS2019-56, pp. 391-396, 2019 年 7 月.

[Pub. 22] 奥秋陽太，坂本純一，吉田直樹，藤本大介，松本　勉，“パイプライン型剰余乗算器を用いたペアリング計算器の FPGA 実装における定数倍演算器の改良”，暗号と情報セキュリティシンポジウム 2019, 1D1-4.

[Pub. 23] 奥秋陽太，坂本純一，吉田直樹，藤本大介，松本　勉，“パイプライン型剰余乗算器を用いたペアリング計算器における圧縮自乗算の高速化”，信学技報, vol. 118, no. 272, HWS2018-50, pp. 19-24, 2018 年 10 月.

[Pub. 24] 奥秋陽太，坂本純一，吉田直樹，藤本大介，松本　勉，“パイプライン型剰余乗算器を用いたペアリング計算器の FPGA 実装に対する圧縮自乗算の高速化”，2018 年電子情報通信学会総合ソサイエティ大会，A-20-9.

[Pub. 25] 長浜佑介，藤本大介，坂本純一，松本　勉，“パイプライン型剰余乗算器を用いたペアリング計算器の FPGA 実装による消費エネルギー評価”，信学技報, vol. 118, no. 3, HWS2018-5, pp. 23-28, 2018 年 4 月.

[Pub. 26] 白井和樹，清川貴仁，坂本純一，松本　勉，“CAN における置換型電気的データ改ざんとその評価”，暗号と情報セキュリティシンポジウム 2018, 3E3-4.

[Pub. 27] 相馬一樹，坂本純一，藤本大介，松本　勉，“測距対象への光照射を用いたパルス LIDAR の計測セキュリティ評価方法”, 暗号と情報セキュリティシンポジウム 2018, 2D3-3.

[Pub. 28] 遠山毅，清川貴仁，白井和樹，坂本純一，小熊　寿，松本　勉，“自動車向けポータブルセキュリティテストベッド v1.1 ～ メッセージ改ざん対策の導入と検証 ～”，信学技報 ISSN 2432-6380.

[Pub. 29] 吉田直樹, 福島和彦, 宮内成典, 坂本純一, 藤本大介, 松本　勉, “TEE システムアーキテクチャとそのオープンソース実装のセキュリティ評価”，信学技報, vol. 117, no. 125, ISEC2017-36, pp. 267-274, 2017 年 7 月.

[Pub. 30] 清川貴仁，中山淑文，坂本純一，藤本大介，松本　勉，“ＣＡＮにおける電気的データ改竄に対するソフトウェア検知方式”，2017 年暗号と情報セキュリティシンポジウム SCIS2017 1E2-4，2017.

[Pub. 31] J. Sakamoto, T. Matsumoto, “Symmetric Ciphers Software-implementable in Energy Harvesting Wireless Sensor Modules,”11th International Workshop on Security IWSEC2016, 招待講演.

[Pub. 32] 坂本純一，松本　勉，“環境発電センサモジュールにソフトウェア実装可能な共通鍵暗号”，2016 年暗号と情報セキュリティシンポジウム SCIS2016 予稿集 2C4-1，SCIS 論文賞.

[Pub. 33] 大石和臣，吉田直樹，渡邉直紀，坂本純一，松本　勉，“自己破壊的耐タンパーソフトウェアの試験実装”，2015 年 暗号と情報セキュリティシンポジウム SCIS2015, 2B1-4.

[Pub. 34] 大石和臣，吉田直樹，渡邉直紀，坂本純一，松本　勉，“自己破壊的耐タンパーソフトウェアの試験実装と評価”，信学技報, vol. 114, no. 489, ICSS2014-99, pp. 217-222, 2015, ICSS 研究賞.