HAW HAMBURG

BACHELOR THESIS
Rene Herthel

Evaluation der Eigenschaften von SRAM PUF in einem Testbed unter Verwendung von RIOT mit Low Power Modes

FAKULTÄT TECHNIK UND INFORMATIK
Department Informatik

Faculty of Engineering and Computer Science
Department Computer Science

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN HAMBURG
Hamburg University of Applied Sciences

Rene Herthel

Evaluation der Eigenschaften von SRAM PUF in einem Testbed unter Verwendung von RIOT mit Low Power Modes

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang *Bachelor of Science Technische Informatik*
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Thomas Schmidt
Zweitgutachter: Prof. Dr. Franz Korf

Eingereicht am: 07.10.2022

Rene Herthel

Thema der Arbeit

Evaluation der Eigenschaften von SRAM PUF in einem Testbed unter Verwendung von RIOT mit Low Power Modes

Stichworte

IoT, RIOT, PUF, SRAM, Stromsparmodus, Zufallszahlengenerator

Kurzzusammenfassung

Static Random Access Memory (SRAM) Physically Unclonable Functions (PUF) leiten unvorhersehbare Informationen aus SRAM-Zellen ab. Diese können für kryptografische Zwecke oder als Seed verwendet werden. In dieser Arbeit wird die Eignung von SRAM basierten PUF unter der Verwendung von Low-Power-Modes im Betriebssystem RIOT untersucht. Dafür werden die PUF-Eigenschaften Zuverlässigkeit und Einzigartigkeit evaluiert.

Rene Herthel

Title of Thesis

Evaluation of SRAM PUF properties in a testbed using RIOT with low power modes

Keywords

IoT, RIOT, PUF, SRAM, Low-Power-Mode, Random-Number-Generator

Abstract

Static Random Access Memory (SRAM) Physically Unclonable Functions (PUF) derive unpredictable information from SRAM cells. These can be used for cryptographic or seed purposes. In this work, the suitability of SRAM-based PUFs is investigated using low-power modes in the RIOT operating system. For this purpose, the PUF properties reliability and uniqueness are evaluated.

Inhaltsverzeichnis

Abbildungsverzeichnis **vi**

Listings **viii**

1 Einleitung **1**
1.1 Related Work 2
1.2 Organisation 3

2 Übersicht zum Internet of Things und RIOT-OS **4**
2.1 Internet of Things 4
2.2 Betriebssystem RIOT 5

3 Static Random Access Memory **10**

4 Physical Unclonable Functions **12**
4.1 SRAM basierte PUF 13
4.1.1 SRAM als Entropiequelle 13
4.1.2 Auslesen der Startwerte 14
4.2 Die Eigenschaften Einzigartigkeit und Zuverlässigkeit 15
4.3 Anwendungsbeispiele 16
4.3.1 Echte- und Pseudozufallszahlengeneratoren 17
4.3.2 Zufallszahlengeneratoren für allgemeine und kryptografische Zwecke 17
4.3.3 Identifikation und Authentifizierung 20
4.4 Fuzzy-Extraktor 21
4.4.1 Enrollment 21
4.4.2 Rekonstruktion 22

5 Der PUF Seed Generator in RIOT **24**

6 Experimente mit der IoT-Testplattform FIT IoT-LAB **26**
6.1 Die Testplattform FIT IoT-LAB 26
6.2 Das IoT-LAB-M3 Entwicklungsboard 27
6.3 Ausmessen von SRAM-Startwerten mit aktiviertem Low-Power-Mode . . . 29
6.4 Erzeugen von Identitäten mit aktiviertem Low-Power-Mode 31

7 Evaluation von SRAM PUF mit Low-Power-Modes **33**
7.1 Die Hamming-Distanz und das Hamming-Gewicht 33
7.2 Berechnung der Metriken in Python 34
7.3 Bias-Test mit dem Hamming-Gewicht 35
7.4 Intra-Chip Test mit der Hamming-Distanz 36
7.5 Inter-Chip Test mit der Hamming-Distanz 37
7.6 Inter-Chip Test mit erzeugten IDs von 194 Testgeräten 38

8 Zusammenfassung und Ausblick **40**
8.1 Zusammenfassung 40
8.2 Ausblick 41

Literaturverzeichnis **42**

A Anhang **46**
A.1 Ergebnisse des Bias-Test mit erzeugten Startmustern von 355 Testgeräten 46
A.2 Ergebnisse des Intra-Chip Test mit erzeugten Startmustern von 355 Testgeräten 54
A.3 Ergebnisse des Inter-Chip Test mit erzeugten Startmustern von 355 Testgeräten 63
A.4 Ergebnisse des Inter-Chip Test mit erzeugten IDs von 194 Testgeräten . . 71
Selbstständigkeitserklärung 77

Abbildungsverzeichnis

2.1 Die Definition des IoT in seiner einfachsten Form [22] 5
2.2 Die Softwareebenen und Module von RIOT [2] 7
2.3 Übersicht des Bootvorgangs in RIOT [23] 9

3.1 Eine Anordnung von SRAM-Zellen die aus zwei kreuzgekoppelten Invertern bestehen. Die Struktur einer 6T-SRAM-Zelle im Detail und die Schwellspannung V_{th} der zugehörigen Transistoren [24] 11

4.1 Ein 64-Bit-Fingerabdruck in der jede Zelle die Wahrscheinlichkeit P hat den Zustand 1 anzunehmen [25] 14
4.2 Ablauf der Enrollment-Phase [18] 22
4.3 Ablauf der Rekonstruktions-Phase [18] 22

5.1 PUF random seeder in RIOT [15] 24

6.1 IoT-LAB M3 mit verschiedenen verbauten Sensoren und einer STM32 MCU [6] 28
6.2 IoT-LAB M3 mit einem NOR-Flashspeicher [6] 28
6.3 Architektur des IoT-LAB-M3 Boards [6] 29
6.4 Ablaufdiagramm für das Ausmessen von SRAM-Startwerten mit LPM . . 30
6.5 Ablaufdiagramm der Testfirmware 32

7.1 Histogramm des Intra-Chip Experiments des Hamming-Gewichts von 355 Testgeräten mit je 100 SRAM-Startmuster. 35
7.2 Histogramm des Intra-Chip Experiments des Hamming-Gewichts von 355 Testgeräten mit je 100 SRAM-Startmuster. 36
7.3 Histogramm des Inter-Chip Experiments der Hamming-Distanz mit 355 Testgeräten mit je 100 Startmustern 38

7.4 Histogramm des Inter-Chip Experiments der Hamming-Distanz mit 195 Testgeräten mit je 100 IDs 39

Listings

7.1 Berechnung der Hamming-Distanz in Python 34
7.2 Berechnung des Hamming-Gewichts in Python 35

A.1 Datensatz mit 355 Testgeräten, jeweils 100 erzeugte Startmuster und das dazu berechnete Hamming-Gewicht. 46
A.2 Datensatz mit 355 Testgeräten, jeweils 100 erzeugte Startmuster und die dazu berechnete Hamming-Distanz. 54
A.3 Datensatz mit 355 Testgeräten, jeweils 100 erzeugte Startmuster und der Hamming-Distanz zwischen einem PUF-A und den anderen PUFs. 63
A.4 Datensatz mit 194 Testgeräten, jeweils 100 erzeugte IDs und der Hamming-Distanz zwischen einem Testgerät A und den anderen Testgeräten. 71

1 Einleitung

Die Anzahl der Geräte, die an einem IP-Netzwerk angeschlossenen sind, wird bis 2023 mehr als dreimal so hoch sein wie die Weltbevölkerung. Es soll durchschnittlich 3,6 vernetzte Geräte pro Kopf geben, gegenüber 2,4 vernetzten Geräten pro Kopf im Jahr 2018 [3]. Bis 2025 wird erwartet, dass es mehr als 30 Milliarden Geräte im Internet of Things (IoT) geben wird, also durchschnittlich fast 4 IoT-Geräte pro Person [13]. Diese Geräte sind häufig als eingebettete Systeme im IoT überall im Alltag wiederzufinden. Dadurch verlassen sich auch immer mehr Menschen auf Integrierte Schaltungen (IC), um sensible Aufgaben durchzuführen. So wird zum Beispiel ein RFID-Chip häufig als Schlüsselkarte für die Zugangskontrolle zu Gebäuden verwendet oder Mobiltelefone beinhalten sensible Daten, wie vertrauliche Dokumente oder persönliche E-Mails. Für ICs ist es deshalb sehr wichtig Sicherheitsoperationen ausführen zu können. Dazu gehört das Authentifizieren eines Gerätes, das Schützen von vertraulichen Informationen und das Gewährleisten von sicherer Kommunikation in einem kostengünstigen hoch-sicheren Weg. Dafür benötigen die ICs für gewöhnlich teure Sicherheitshardware die aus Platz- und Kostengründen im IoT häufig nicht verfügbar sind. Für Sicherheitsoperationen wird ein unvorhersehbares Geheimnis benötigt, dass in jedem IC physisch vorhanden ist. Auf dieses Geheimnis kann nicht zugegriffen werden und es ist nicht duplizierbar [27].

In der aktuell besten Vorgehensweise wird ein geheimer Schlüssel in nicht-flüchtige Speicher abgelegt, wie beispielsweise EEPROM. Um vertrauliche Informationen zu schützen, wird ein Gerät mit kryptografischen Operationen authentifiziert. Zu kryptografischen Operationen zählen beispielsweise digitale Signaturen und Verschlüsselungen. Bei dieser Vorgehensweise ist das Verwalten von Geheimnissen im Speicher mit Mehraufwand und Kosten verbunden. Nicht-flüchtige Speichertechnologien sind anfällig für invasive Attacken, weil Geheimnisse immer in digitaler Form verwaltet werden müssen. Auch Batterie gestützte RAM-Speicher können ausgelesen werden, selbst wenn der Schlüssel über einen längeren Zeitraum gespeichert wurde. Um ein hohes Maß an physischer Sicherheit zu erreichen, muss der IC durch eine teure Schaltung zur Erkennung von Manipulationen geschützt werden. Diese muss ständig batteriebetrieben sein [27].

Für dieses Problem stellen Physical Unclonable Functions (PUF) eine innovative Lösung dar. Diese können Geheimnisse aus komplexen physikalischen Eigenschaften einer Integrierten Schaltung (IC) ableiten. So kann beispielsweise ein flüchtiges Geheimnis aus den zufälligen Verzögerungseigenschaften von Schaltkreisen und Transistoren generiert werden. Da eine PUF die zufälligen Schwankungen während eines IC-Fertigungsprozesses ausnutzt, ist das Geheimnis schwer vorherzusagen oder zu extrahieren [27]. Für das Ableiten von Geheimnissen hat sich der SRAM-Speicher als praktikable Entropiequelle herausgestellt. SRAM-Speicher sind auf vielen Plattformen vorhanden [15].

Um ein zuvor abgeleitetes Geheimnis einer PUF rekonstruieren zu können, wird häufig ein sogenannter Fuzzy-Extraktor eingesetzt. Mit diesen lassen sich vom Rauschen beeinflusste neue PUF-Messungen mit Hilfsdaten korrigieren und zuvor erzeugte Geheimnisse rekonstruieren [18]. In dem Kontext vertrauenswürdiger Firmware-Updates von IoT-Geräten werden die Hilfsdaten eines IoT-Gerätes typischerweise auf einem Server berechnet. Dazu wird vom Server bei der Bereitstellung eines IoT-Gerätes zuerst die PUF-Antwort ausgemessen, um mit dieser die Hilfsdaten zu erzeugen. Die Hilfsdaten werden danach auf einen nicht-flüchtigen Speicher des IoT-Gerätes übertragen [21].

In dieser Arbeit wird die Eignung von SRAM-basierten PUFs unter Verwendung von Low-Power-Modes im Betriebssystem RIOT evaluiert. Dafür werden die PUF-Eigenschaften Zuverlässigkeit und Einzigartigkeit unterschiedlicher IoT-Geräte analysiert und evaluiert.

1.1 Related Work

Im Jahr 2001 wurde von Pappu [19] das Konzept von Physical Unclonable Function unter dem Namen Physical One-Way Function vorgestellt. Bei dieser Technologie wird eine Antwort erhalten, wenn eine mit Luftbläschen gefüllte Epoxid-Scheibe mit einem Laser getroffen wird. Aus dieser Messung wird ein Streubild abgeleitet. Im Jahr 2002 wurde von Gassend et al. [7] dieses Prinzip in Silicon Physical Random Functions weiterentwickelt. Diese Funktionen nutzen die Variationen im Herstellungsprozess eines ICs, um diese einzigartig charakterisieren zu können. Für diesen Zweck wird die Frequenz von Ring-Oszillatoren gemessen. Mit dieser Methode, auch bekannt als Ring-Oszillator-PUF, können ICs charakterisiert werden. Im Jahr 2004 wurde von Lee et al. [17] mit der Arbiter PUF ein weiteres Konzept vorgestellt, welches auf verzögerten Signallaufzeiten basiert. ICs können sowohl mit den PUFs charakterisiert werden, die Verzögerungen

messen, als auch mit den speicherbasierten PUFs. Speicherbasierte PUFs basieren auf der Messung der Einschaltzustände von Speicherzellen. Sie inkludieren SRAM PUF, welche von Guajardo et al. [9] 2007 vorgestellt wurden [25]. Der Aufbau von Static Random Access Memory (SRAM) und die Funktionsweise einzelner SRAM-Zellen werden hier beschrieben: [24].

PUFs benötigen geeignete Entropiequellen für Anwendungen wie das Erzeugen von Seeds für Zufallszahlengeneratoren (PRNG). Dies, sowie die Generierung von Zufall im Internet of Things und die Verwendung von SRAM als Entropiequelle bis zur IoT-Anwendung werden hier beschrieben: [16].

1.2 Organisation

Die Arbeit hat die folgende Struktur. Kapitel 2 ab Seite 4 bietet eine Übersicht zum Internet of Things und das Betriebssystem RIOT. Kapitel 3 ab Seite 10 erläutert den Aufbau eines SRAMs und die Nutzung von SRAM als Entropiequelle. Kapitel 4 ab Seite 12 beinhaltet die Grundlagen zu den PUFs. Hier werden insbesondere SRAM basierte PUF, PUF-Eigenschaften, Anwendungsbeispiele und der Fuzzy-Extraktor vorgestellt. In Kapitel 5 ab Seite 24 wird der PUF Seed Generator in RIOT vorgestellt. Kapitel 6 ab Seite 26 beschreibt die durchgeführten Experimente auf der IoT-Testplattform FIT IoT-LAB. In Kapitel 7 ab Seite 33 wird die Evaluation von SRAM PUF mit Low-Power-Modes vorgestellt. Kapitel 8 ab Seite 40 hält eine Zusammenfassung und einen Ausblick bereit.

2 Übersicht zum Internet of Things und RIOT-OS

Die erste Sektion in diesem Kapitel gibt eine Definition und einen Überblick über das Internet of Things. Es beschreibt die Hauptmerkmale des Internet of Things und welche Dinge in diesem miteinander verbunden sind. Die zweite Sektion stellt das Betriebssystem RIOT im Allgemeinen vor und den modularen Aufbau der Softwarearchitektur sowie verwendete Module und den Ablauf des Bootvorgangs.

2.1 Internet of Things

Das Internet of Things (IoT) kann in seiner einfachsten Form als ein Netzwerk von physischen Elementen betrachtet werden, das durch folgende Merkmale befähigt wird:

- **Sensoren** zum Sammeln von Informationen
- **Identifikatoren** zur Identifizierung der Datenquelle
- **Software** zur Analyse der Daten
- **Internetkonnektivität** zur Kommunikation und Benachrichtigung

Das IoT ist also das Netzwerk der Dinge mit eindeutiger Elementidentifikation, eingebettet in Software, mit Sensoren und allgegenwärtiger Konnektivität mit dem Internet. Durch das IoT wird es Dingen oder Objekten ermöglicht über die Telekommunikationsinfrastruktur des Internets Informationen mit dem Hersteller, dem Betreiber oder anderen angeschlossenen Geräten auszutauschen. Es ermöglicht die Erfassung physischer Objekte (zur Bereitstellung spezifischer Informationen) und die Fernsteuerung über das Internet. Dadurch werden Möglichkeiten für eine direktere Integration zwischen der physischen Welt und computergestützten Systemen geschaffen. Dies führt zu einer höheren

Effizienz, verbesserten Genauigkeit und einem wirtschaftlichen Nutzen. Jedes Ding ist durch sein eingebettetes Computersystem eindeutig identifizierbar und kann mit der bestehenden Internetinfrastruktur interagieren. Das IoT in seiner einfachsten Form kann als Schnittpunkt von Internet, Dingen und Daten betrachtet werden, wie in Abbildung 2.1 dargestellt [22].

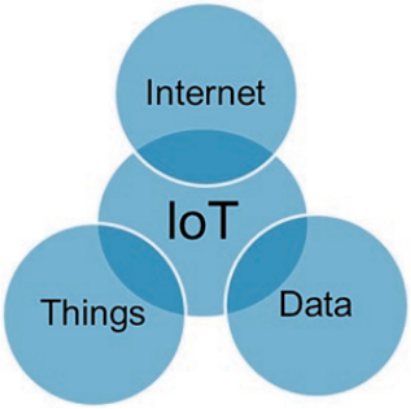

Abbildung 2.1: Die Definition des IoT in seiner einfachsten Form [22]

Das Hauptmerkmal des IoT ist die physische Verbindung von allem und jedem, wie zum Beispiel Sensoren, Geräten, Maschinen, Menschen, Tieren und Bäumen. Des Weiteren werden Prozesse über das Internet zur Überwachung oder Steuerung miteinander verbunden. Die Verbindungen beschränken sich nicht auf Informationsseiten, sondern sind tatsächliche und physische Verbindungen. Diese ermöglichen es den Benutzern sogenannte Dinge zu erreichen und bei Bedarf die Kontrolle zu übernehmen. Das Ziel ist das Sammeln von Informationen aus diesen Objekten, um Produkte und Dienstleistungen zu verbessern [22].

2.2 Betriebssystem RIOT

RIOT ist ein Betriebssystem, welches nach den Bedürfnissen des IoT entworfen wurde. Der Quellcode ist Open-Source und wird auf GitHub [1] unter einer LGPLv2.1 [8] bereitgestellt. Die Lizenz erlaubt es, RIOT mit proprietärer Software zu verknüpfen und stellt sicher, dass RIOT von den Endnutzern modifiziert werden kann. Dabei wird das Betriebssystem von einer weltweiten Gemeinschaft von Entwicklern weiterentwickelt [2].

[1] https://github.com/RIOT-OS/, abgerufen am 01-10-2022

RIOT ist gegenüber Low-End Geräten besonders ressourcenschonend, was unter anderem die Entwicklung gängiger 8-Bit, 16-Bit und 32-Bit Architekturen erleichtert. Bei Netzwerkstandards liegt der Fokus auf offenen, standardisierten Netzwerkprotokollen, wie die IETF Protokolle. Weiterhin wird ein eigener IP-Stack namens GNRC unterstützt, mit dem sich Geräte im IoT vernetzen lassen. Weitere Merkmale des Betriebssystems sind Echtzeit- und Multithreading-Fähigkeiten, Energie-Effizienz, ein geringer Speicherbedarf sowie eine einheitliche, von der zugrunde liegenden Hardware unabhängige Programmierschnittstelle (API) [2].

Eine der Entwurfsprinzipien von RIOT ist die modulare Softwarearchitektur, die um einen minimalistischen Kernel herum aufgebaut ist. Somit ist RIOT in Software-Modulen strukturiert, die erst zur Kompilierzeit konfiguriert und verwendet werden. Die Module sind abgekapselt vom Kern des Betriebssystems und bieten minimalistische Funktionalitäten an. Mit diesem Ansatz wird das System in wohldefinierte Programmcode-Einheiten aufgebaut, wobei nur die für den Anwendungsfall benötigten Module inkludiert werden [2].

Auch die Hardware-Treiber sind in RIOT modular aufgebaut. Mit diesen können CPU-externe Komponenten wie Sensoren, Aktoren, Speicher oder Netzwerkschnittstellen kontrolliert werden. Für Peripheriekomponenten existieren einheitliche APIs, die auf heterogener Hardware aufbauen. Dadurch kann derselbe Programmcode auf unterschiedlicher Hardware ausgeführt werden. Zu den Peripheriekomponenten gehören unter anderem Real-Time-Clocks (RTCs), die eine akkurate Zeitangabe unterstützen. RTCs führen ihre Operationen auch dann aus, wenn die CPU durch das Power-Management des Betriebssystems in einen Energiesparmodus versetzt wurde. Die RTC wird von High-Level Treibern wie beispielsweise dem Timer-Modul XTimer genutzt. XTimer werden der Kategorie der Treiber Applikationen zugeordnet. High-Level Treiber bauen auf den Treiber der Peripheriekomponenten auf, wodurch die Portabilität der Treiber zwischen unterschiedlicher Hardware garantiert wird [2]. Die Softwareebenen und Module von RIOT werden in Abbildung 2.2 dargestellt.

Das Timer-Modul stellt eine wichtige Zuordnung zwischen der physikalischen Zeit, wie sie in der Außenwelt wahrgenommen wird und der von der CPU verwendeten internen Zeitmessung her [2]. Ein Timer-Modul ist beispielsweise die Real-Time-Clock, welche häufig in Mikrocontrollern implementiert wird. Mithilfe dieser Echtzeituhr können zum Beispiel Zeit- und Datumsfunktionalitäten dargestellt werden. Zudem kann das Modul zur Generierung von zyklischen Interrupts oder Alarmfunktionen verwendet werden. Damit

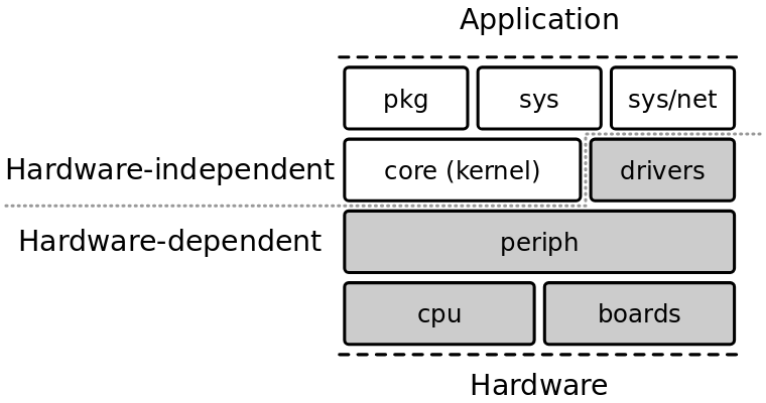

Abbildung 2.2: Die Softwareebenen und Module von RIOT [2]

die Echtzeitfunktionalität in allen Betriebsmodi, gewährleistet ist, wird das RTC-Modul, wenn es verwendet wird, nicht abgeschaltet. Es läuft in der Regel auf einer dedizierten Taktquelle weiter [12].

Das Timer-Modul in RIOT wird XTimer genannt. Hierbei handelt es sich um ein Timer-Subsystem mit einer Abstraktionsschicht auf höherer Softwareebene. XTimer bietet eine einfache API, die auf natürlichen Zeitwerten basiert und eine vollständige Abstraktion von der zugrundeliegenden Timer-Hardware bietet, um unter anderem Threads in den Ruhezustand versetzen zu können. Für die Verbindung mit RTCs nutzt XTimer die Abstraktionsschicht der Peripherie-Treiber. Auf diese Weise ist die Implementierung von XTimer unabhängig von der Hardware, während das Modul noch konfigurierbar ist. Dadurch funktioniert es mit unterschiedlichen Hardware-Umgebungen. Typische von RIOT unterstützte CPUs bieten eine Reihe von gewöhnlichen Timern und optional einen oder mehrere Real-Time-Timer (RTT) oder eine Real-Time-Clock (RTC). Threads können in den Ruhezustand versetzt werden und es können Callback-Funktionen und Event-Timer gesetzt werden, damit ein Mikrocontroller aus einem Low-Power-Modes (LPM) in den normalen Betrieb zurückkehren kann [2].

Viele Mikrocontroller definieren LPM, wie „Idle“, „Standby“ und „Stop“. Diese werden vom Power-Management-Modul in RIOT unterstützt. Das Power-Management-Modul implementiert einen Standardmechanismus, der den niedrigstmöglichen Energiezustand bestimmt. Das Modul wurde für unterschiedliche CPU-Implementierungen entwickelt und verwendet einen einfachen Konsensmechanismus namens Cascade. Cascade basiert auf einer strengen Hierarchie von Stromverbrauchsstufen, die für jede CPU definiert ist. Eine niedrigere Leistungsstufe bedeutet einen geringeren Stromverbrauch. Die Hierar-

chie ist so aufgebaut, dass bei einer Blockierung der Leistungsstufe N implizit auch alle Leistungsstufen $M \leq N$ blockiert werden, so dass die CPU nicht in eine beliebige Leistungsstufe $M \leq N$ wechseln kann. Das Power-Management-Modul definiert den niedrigstmöglichen Energiezustand als den Wert 0. Der höchstmögliche LPM ist abhängig von der Spezifikation des jeweiligen Boards und den unterstützten LPMs [2].

Als weiteres Modul nutzt RIOT das Memory-Technology-Device (MTD). Hier wurde die Schnittstelle nach dem Vorbild der entsprechenden Linux-API entwickelt. Das MTD-Modul bietet eine Abstraktionsschicht für den Zugriff auf Speicher-Bausteine, wie Flash-Speicher [2]. Die vereinheitlichte Schnittstelle vermittelt zwischen den vielfältigen hardwarespezifischen Gerätetreibern und den oberen Schichten eines Systems. Ein Vorteil von MTD ist, dass die Anwender dieser Abstraktionsschicht keine Kenntnisse über Interna der darunterliegenden Schichten besitzen müssen, wie zum Beispiel welches Dateisystem verwendet wird. Des Weiteren kann dieselbe Abstraktionsschicht beim Wechsel des Flash-Speichers wiederverwendet werden [29].

Darüber hinaus bietet RIOT einen Befehlszeileninterpreter (CLI), ähnlich einer Shell in Linux. Der Zugriff auf die CLI erfolgt in der Regel (aus der Ferne) über UART. Sie wurde entwickelt, um das Debugging und die Laufzeitkonfiguration während des Testens oder der Durchführung von Experimenten zu erleichtern. Entwickler können leicht benutzerdefinierte Befehle über dedizierte Befehlshandler hinzufügen [2].

Die Module von RIOT werden während des Bootvorgangs initialisiert. Der Bootvorgang wird in Abbildung 2.3 dargestellt. RIOT bietet alle Elemente für das Bootstrapping von IoT-Hardware vom ersten aufgerufenen Softwarebefehl im Reset-Handler bis zur eigentlichen Hauptfunktion in einem Thread-Kontext. Der Bootvorgang besteht im Wesentlichen aus den folgenden vier Schritten [2]:

1. Initialisierung des Speichers (SRAM)
2. Initialisierung von Board und CPU
3. Initialisierung der verwendeten C-Bibliotheken (optional)
4. Einrichtung und Initialisierung des eigentlichen Betriebssystems

Im ersten Schritt wird der Speicher initialisiert. Hier werden die initialisierten Variablen in den RAM (data Sektion) kopiert und alle uninitialisierten Variablen auf den Wert 0

gesetzt (bss Sektion). Die Board-Initialisierung kümmert sich dann um die Initialisierung boardspezifischer Hardware-Elemente wie On-Board-LEDs. Zu diesem Zeitpunkt wird auch die CPU-Initialisierung ausgelöst, in der unter anderem das Interrupt-System, Taktgeber und gemeinsame Peripherietreiber eingerichtet werden. Als nächstes kümmert sich die C-Bibliotheksinitialisierung um die Abbildung der C-Standard-Bibliothek auf die entsprechenden RIOT-Systemaufrufe. Dies beinhaltet unter anderem Funktionen für dynamische Speicherzuweisung oder STDIO-Zugriffe. Im letzten Schritt wird das Basissystem initialisiert und die Kontrolle an den RIOT-Kernel übergeben. Nach der Einrichtung des Idle- und Main-Threads initialisiert RIOT alle konfigurierten Systemmodule und Gerätetreiber über das AUTO_INIT-Modul. Zum Schluss wird die Hauptfunktion aufgerufen, die den Einstiegspunkt für die eigentliche Benutzeranwendung darstellt [2].

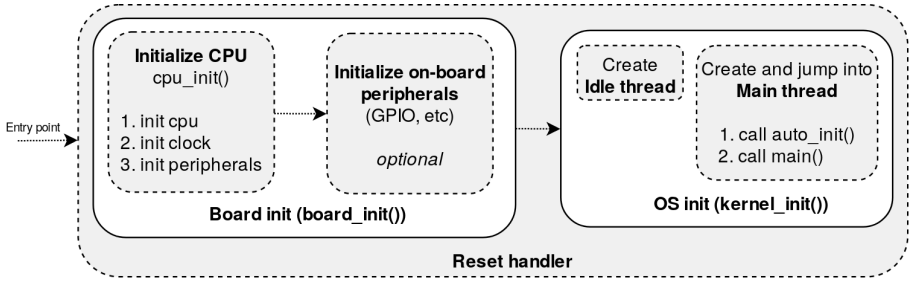

Abbildung 2.3: Übersicht des Bootvorgangs in RIOT [23]

3 Static Random Access Memory

Der Static Random Access Memory (SRAM) ist eine standardmäßig flüchtige Speichertechnologie, die Informationen für die Zeit einer anliegenden Betriebsspannung speichert. SRAM können auf unterschiedliche Weise konstruiert werden. Bei der vorherrschenden Konstruktion werden sechs Transistoren verwendet, um eine einzelne 6-Transistor-SRAM-Zelle (6T-SRAM-Zelle) zu konstruieren. Diese basiert auf der Technologie von Metall-Oxid-Halbleiter (CMOS). Bei diesem Aufbau werden vier Transistoren zur Speicherung eines einzelnen Bits verwendet, während zwei zusätzliche Transistoren den Zugriff auf die SRAM-Zelle zum Lesen oder Schreiben steuern. Die vier Speichertransistoren bilden zwei kreuzgekoppelte Inverter, die sich gegenseitig verstärken und somit eine positive Rückkopplungsschleife bilden. Sobald die Schwellenspannung (V_{th}) erreicht ist, beginnt der jeweilige Transistor zu leiten. Aufgrund von Fertigungsschwankungen gibt es in der Regel eine Diskrepanz zwischen den Schwellenspannungen der beiden kreuzgekoppelten Inverter. Eine schematische Beschreibung einer 6T-SRAM-Zelle findet sich in Abbildung 3.1. [24]. Im Detail betrachtet treiben die Inverter jeweils einen der beiden Zustandsknoten an, die mit A und B gekennzeichnet sind. Wenn der Schaltkreis stromlos ist, sind beide Zustandsknoten entladen und befinden sich in einem metastabilen Zustand ($AB = 00$). Wenn die Stromversorgung wiederhergestellt wird, findet eine Transition zu einem der stabilen Zustände 0 ($AB = 01$) oder 1 ($AB = 10$) statt. Der Zustand ($AB = 11$) ist instabil und für die Transistoren nicht erreichbar [11]. Darüber hinaus weisen Transistoren untereinander trotz gleichem logischem Aufbau unterschiedliche physikalische Eigenschaften auf, die durch Fluktuationen im Herstellungsprozess entstehen. Diese führen zu den messbaren Unterschieden der elektronischen Eigenschaften, wie der Schwellenspannung und sind über dem gesamten SRAM verteilt [25].

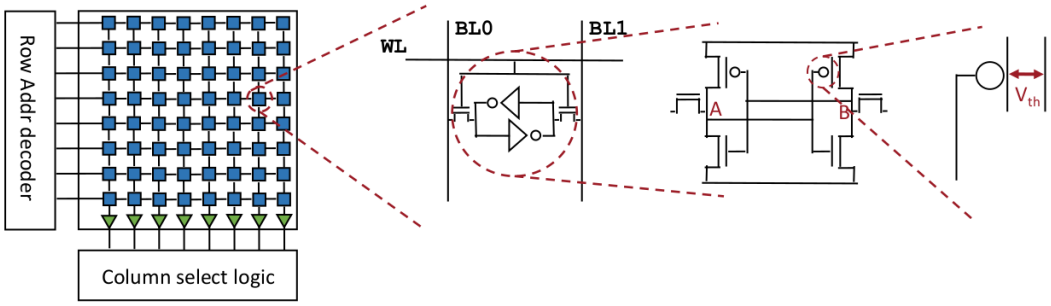

Abbildung 3.1: Eine Anordnung von SRAM-Zellen die aus zwei kreuzgekoppelten Invertern bestehen. Die Struktur einer 6T-SRAM-Zelle im Detail und die Schwellspannung V_{th} der zugehörigen Transistoren [24]

4 Physical Unclonable Functions

Eine Physical Unclonable Function (PUF) kann als eine komplexe physikalische Struktur beschrieben werden. Sie wird durch einen Stimulus c herausgefordert, um eine Antwort r zu erzeugen. Die Antwort r entspricht einer verrauschten PUF-Messung X. Die PUF-Messung X hängt sowohl von der Herausforderung c als auch von der mikroskopischen physikalischen Struktur des zugrunde liegenden Objekts ab, in der eine PUF eingebettet ist. Aufgrund winziger Abweichungen während des Produktionsprozesses ist die Struktur des physischen Objekts, das eine PUF enthält, bei jedem Exemplar einzigartig. Die Unterschiede in der physikalischen Struktur verschiedener PUFs befinden sich auf mikroskopischer Ebene. Vermutlich liegen diese außerhalb des Einflusses der derzeitigen Fertigungstechnologie. Deshalb wird davon ausgegangen, dass PUFs nicht klonbar sind und nicht einmal vom Hersteller reproduziert werden können. Das Challenge-Response-Verhalten des physikalischen Systems ist so komplex, dass die Reaktion auf eine zufällig ausgewählte Challenge nicht vorhergesagt werden kann [24]. Diese Eigenschaft hindert einen Angreifer daran, eine bestehende PUF-Instanz zu duplizieren. Wie bei algorithmischen Einweg-Hash-Funktionen sollte die Abbildung von PUFs in der Vorwärtsrichtung einfach zu berechnen, in der Rückwärtsrichtung aber nicht umkehrbar sein [11]. Da PUFs auf physikalischen Strukturen basieren, benötigen sie keine speziellen Herstellungsverfahren, Programmierungen oder weitere Prüfschritte, um zu funktionieren [27].

Eine PUF ist eine Art von Funktion, die mit physikalischen Eigenschaften eines Objekts verwoben ist. Genau wie herkömmliche mathematische Funktionen bildet eine PUF f_{PUF} Eingabeelemente (Herausforderungen) $\{c_1, c_2, ..., c_k\}$ eines Challenge-Raums $c_i \in C$ auf Ausgangselemente (Antworten) $\{r_1, r_2, ..., r_l\}$ eines Antwortraums $r_j \in R$ ab [24]:

$$f_{PUF} : C^k \rightarrow R^l.$$

Darüber hinaus wurde eine vereinfachte Notation vorgestellt [24]:

$$r \leftarrow PUF(c) \; or \; PUF(c) = r.$$

4.1 SRAM basierte PUF

SRAM PUF basieren auf der Beobachtung, dass der Einschaltzustand von SRAM-Zellen einen physikalischen Fingerabdruck offenbart. Eine SRAM-Zelle ist der erforderliche Schaltkreis für die Speicherung und den Zugriff auf einem Bit. Daher ist die SRAM-Zelle eine der kleinstmöglichen physikalischen Schaltkreise, die eine digitale Ausgabe erzeugen kann [11]. Dabei werden Fertigungsschwankungen ausgenutzt, die sich in einer Vorspannung der SRAM-Speicherzellen manifestieren. In der Einschaltphase des SRAMs werden die SRAM-Zellen aus einem metastabilen Zustand in eine ihrer bistabilen Zustände gezogen, die entweder den logischen Wert Null oder Eins darstellen. Die Tendenz, sich auf einen stabilen Zustand einzupendeln, ist auf winzige Abweichungen von der ansonsten symmetrischen Anordnung der Transistoren zurückzuführen. Die Inverter bilden eine positive Rückkopplungsschleife, die die Transistoren zu großen Unterschieden zwingt. Dadurch verlässt die SRAM-Zelle den metastabilen Zustand, was schließlich zu einem so genannten Startwert von einem der logischen Werte führt. Dabei zeigen die meisten Zellen ein stabiles Verhalten während des Einschaltens, beziehungsweise eine starke Tendenz beim Initialisieren auf einen festen Startwert. Ein Array aus mehreren Startwerten von SRAM-Zellen ergibt ein Startmuster, das als PUF-Messung X ausgewertet wird. Dieses dient als Fingerabdruck für das SRAM-Modul [24].

4.1.1 SRAM als Entropiequelle

Es wird davon ausgegangen, dass sowohl Prozessschwankungen als auch Rauschen die Verzerrung einer Zelle beeinflussen. Der Bias einer Zelle ist eine kontinuierliche Größe, die die Tendenz dieser Zelle zu 0 oder 1 beim Einschalten darstellt. Nach dem Einschalten wird der Bias durch Rauschen beeinflusst, so dass jede Zelle über viele Einschaltvorgänge durch eine Wahrscheinlichkeits-Verteilungsfunktion beschrieben werden kann. Eine Zelle mit einem Bias auf 0 oder 1 wird beim Einschalten unabhängig von den Rauschbedingungen auch auf 0 oder 1 gesetzt. Eine Zelle mit neutralem Bias hat keine starke Tendenz zu einem der beiden Zustände und kann sich entweder auf 0 oder 1 einschalten. Eine neutrale Zelle besteht nicht unbedingt aus perfekt aufeinander abgestimmten Komponenten. Stattdessen weist sie eine unbestimmte Kombination von Variationen auf, die sich beim Einschalten unter Nennbedingungen annähernd ausgleichen. Eine solche Zelle bleibt nicht unter allen Betriebsbedingungen neutral. Zur Veranschaulichung ist in Abbildung 4.1 ein 64-Bit Fingerabdruck abgebildet. Die Helligkeit der Schattierung einer Zelle

repräsentiert dabei die Wahrscheinlichkeit P, den Zustand 1 anzunehmen [11]. Durch das zufällige Rauschen und die daraus resultierende Verzerrung der SRAM-Zellen kann ein SRAM als Entropiequelle genutzt werden, um Seeds aus dem Einschaltzustand eines SRAMs zu extrahieren [28].

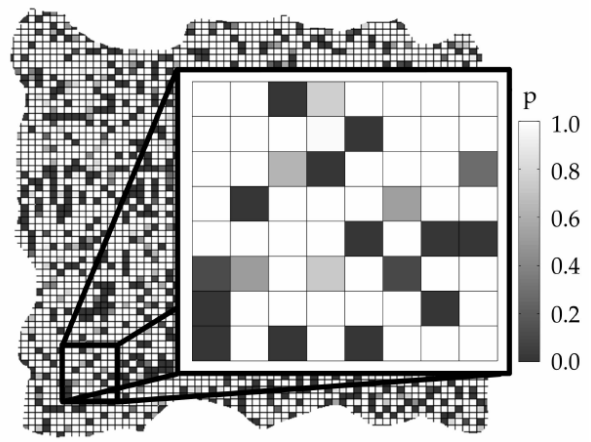

Abbildung 4.1: Ein 64-Bit-Fingerabdruck in der jede Zelle die Wahrscheinlichkeit P hat den Zustand 1 anzunehmen [25]

4.1.2 Auslesen der Startwerte

Es ist erforderlich auf das unveränderte Startmuster der SRAM-Zellen zuzugreifen, um einen Messwert aus einer SRAM-basierten PUF abzurufen. Hier wird das Attribut unverändert besonders hervorgehoben. Es bezieht sich auf den Zustand des SRAM-Arrays kurz nach dem Einschalten und bevor ein Prozess einen Schreibzugriff auf dem Speicher durchgeführt hat. Jeder Schreibvorgang auf das SRAM würde höchstwahrscheinlich das Einschaltmuster so verändern, dass die resultierenden Werte ein Artefakt des entsprechenden Softwareprozesses und nicht der zugrunde liegenden physikalischen Eigenschaften des SRAM-Moduls sind [24].

Die Logik zur Abfrage einer SRAM-basierten PUF besteht aus vier wesentlichen Schritten. Zuerst erfolgt die Einrichtung einer Kommunikationsschnittstelle zur Übertragung der Startwerte an einen Host-Rechner. Dies erfolgt in der Regel unter Verwendung von Univeral Asynchronous Receiver Transmitter (UART). Als nächstes wird Beginn und Größe des Speicherbereichs bestimmt, der als PUF abgefragt werden soll. Bei der anfänglichen Auswertung soll dieser Bereich maximiert werden, um eine vollständige Cha-

rakterisierung des SRAM-Moduls zu erhalten. Nach der Auswertung der PUF-Instanz wird jedoch nur ein minimaler SRAM-Bereich verwendet, um so wenig SRAM wie möglich zu verbrauchen. Im dritten Schritt werden die einzelnen Bytes des entsprechenden Speicherbereichs ausgelesen und an den Host-Rechner zur Auswertung oder als Eingabe für eine Fuzzy-Extraktor-Konstruktion zur Schlüsselableitung weitergeleitet. Im Letzten Schritt wird aus Sicherheitsgründen das Startmuster der jeweiligen PUF überschrieben, um für andere Prozesse unzugänglich zu sein [24].

Wenn ein dedizierter SRAM-Chip für die PUF-Nutzung genutzt werden kann, steht das gesamte SRAM für die Extraktion eines Fingerabdrucks zur Verfügung. In diesem Fall und abhängig vom Design der Stromversorgung des SRAM-Moduls kann das SRAM mit Strom versorgt und somit die PUF zu jedem beliebigen Zeitpunkt abgefragt werden. Die Nutzung des integrierten On-Chip-SRAMs kommerzieller Geräte bringt jedoch einige kritische Einschränkungen mit sich. Diese beeinflussen nicht nur die Art und Weise der Anbindung der PUF, sondern auch den Zugriffszeitpunkt auf die PUF. Wird ein SRAM-Modul neben der PUF-Nutzung in erster Linie als Hauptspeicher der eingebetteten MCU verwendet, muss auf die Extraktion eines unveränderten SRAM-Startmusters geachtet werden. Insbesondere muss die PUF-Messung unmittelbar nach dem Einschalten des SRAM-Moduls durchgeführt werden. Dies entspricht in der Regel einer frühen Phase während der Bootvorgangs des gesamten Gerätes. Außerdem kann nur ein Bruchteil des SRAM genutzt werden, da der Rest für die Ausführung der Firmware (das heißt für Stack oder Heap) verfügbar gehalten werden muss und daher vor der PUF-Messung verändert worden sein kann [24].

4.2 Die Eigenschaften Einzigartigkeit und Zuverlässigkeit

Um auf die Qualität eines PUFs Rückschlüsse ziehen zu können, werden in dieser Arbeit insbesondere die zwei Eigenschaften Einzigartigkeit und Zuverlässigkeit betrachtet. Die Eigenschaft Einzigartigkeit gilt sowohl für einzelne PUF-Antworten als auch zwischen verschiedenen Antworten. Für diesen Eigenschaften müssen zwei Bedingungen erfüllt sein. Für die erste Bedingung müssen alle Bits innerhalb einer einzelnen PUF-Antwort zufällig und unvorhersehbar sein. So liefern die Bits einer einzigen Antwort keine Information übereinander. Weiterhin muss zwischen verschiedenen Geräten ausreichend Entropie in der Quelle vorhanden sein. Das bedeutet statistisch gesehen ist jedes Gerät

einzigartig und die Wahrscheinlichkeit, dass zwei Geräte dieselbe PUF-Antwort haben, ist vernachlässigbar klein [25].

Die zweite Eigenschaft ist die Zuverlässigkeit. Unter Zuverlässigkeit versteht man, dass wann immer eine neue PUF-Antwort gemessen wird, die ursprüngliche Referenzmessung wiedererkannt wird, die während der Enrollment-Phase durchgeführt wurde. Wenn auf demselben Gerät mehrfach eine PUF-Antwort gemessen wird, wird durch zusätzliches Signalrauschen auf der Referenzmessung eine Anzahl an Fehlern (Bit-Flips) auftreten. Je nach implementierten Fehlerkorrekturcode kann durch einen Abgleich der Informationen in einem sogenannten Fuzzy-Extraktor ein bestimmtes Maß an Signalrauschen korrigiert werden [25].

4.3 Anwendungsbeispiele

Zufallszahlen werden zur Lebenszeit eines IoT-Gerätes von einer Vielzahl von Anwendungsfällen angefordert. Die Anwendungsfälle dienen entweder allgemeinen Zwecken oder sie werden für kryptografisch sichere Zufallszahlen benötigt. Für die allgemeine Verwendung sind lediglich ausreichend gut repräsentierte statistische Eigenschaften erforderlich. Kryptografisch sichere Zufallszahlen müssen auch bei böswilligen Angriffen unvorhersehbar bleiben. Während die erste Kategorie im Verhältnis leicht erreicht werden kann, ist die Bereitstellung von sicheren Zufallszahlen im eingeschränkten Umfeld schwieriger und kann je nach Angreifermodell und Stärke des Angreifers möglicherweise überhaupt nicht erreicht werden [16].

Die Erzeugung von Schlüsseln, Salts oder Nonces erfordert Zufallsdaten, die für Angreifer unvorhersehbar sind. Daher müssen Zufallszahlengeneratoren (RNGs) mit Seeds mit hoher Entropie gespeist werden. Um Zufallszahlen für kryptografische Anwendungen zu erzeugen, sollte idealerweise eine physikalische Quelle genutzt werden die echte Zufälligkeit liefert. Solche nicht-deterministischen Quellen leiten ihre Zufälligkeit von zugrunde liegenden physikalischen Eigenschaften ab, die ein unvorhersehbares Verhalten zeigen. Die meisten dieser physikalischen RNG-Konstruktionen haben jedoch zwei Nachteile. Erstens benötigen sie spezielle Hardware, um die Zufälligkeit aus den physikalischen Einheiten des Gerätes zu extrahieren. Zweitens ist der Durchsatz solcher RNGs oft zu gering für kryptografische Anwendungen, bei denen große Ströme von Zufallsbits benötigt werden. Daher wurden PUF-basierte RNGs vorgeschlagen. Diese nutzen das inhärente Rauschen einer PUF-Messung aus, um echte Zufallsdaten zu liefern [24].

4.3.1 Echte- und Pseudozufallszahlengeneratoren

Pseudozufallszahlengeneratoren (PRNG) sind deterministische Algorithmen, die Zufallszahlen mit guten Zufallseigenschaften generieren können. Diese werden durch vier Merkmale beschrieben. Dazu zählen eine lange Periode, die Effizienz, die Reproduzierbarkeit und die Portabilität. Die lange Periode beschreibt die Größe der noch nie dagewesenen Sequenz, die der Generator erzeugen kann und von der aus er sich selbst wiederholt. Die Effizienz gibt die erzeugte Menge an Zahlen an, basierend auf der Zeiteinheit und dem Energieverbrauch. Die Reproduzierbarkeit erlaubt es, dass der Generator eine zuvor erzeugte Sequenz reproduzieren kann und durch die Portabilität verhalten sich Implementierungen in verschiedenen Systemen gleich [20].

Um eine gegebene Pseudozufallssequenz zu reproduzieren, muss bei der Initialisierung eines PRNG dieselbe Seed verwendet werden. Die Seed wird vom Generator verwendet, um seinen internen Zustand vor dem Erzeugen einer Pseudozufallssequenz einzustellen. Bei der Seed handelt es sich in der Regel um eine von einem True Random Number Generator (TRNG) erzeugte Sequenz. Diese kann dem PRNG beispielsweise manuell, über eine API oder hartcodiert zum Zeitpunkt der Herstellung zur Verfügung gestellt werden [20].

Ein echter Zufallszahlengenerator (True Random Number Generator, TRNG) ist in der Regel ein Gerät, das ein physikalisches Verhalten oder einen in der Natur vorkommenden Prozess misst. TRNGs werden häufig zur Erzeugung von Seeds für Pseudozufallszahlengeneratoren (PRNGs) verwendet. Beispielsweise können das thermische Rauschen in Halbleitern oder die Photonenzählung in einem Laserstrahl gemessen werden. Durch die Abhängigkeit der Messung von einem physikalischen Phänomen müssen Sensoren verwendet werden. Die Sensoren machen das System komplexer und teurer. Außerdem sind TRNGs langsam und können eine zuvor erzeugte Sequenz nicht reproduzieren [20].

4.3.2 Zufallszahlengeneratoren für allgemeine und kryptografische Zwecke

Zufallszahlengeneratoren können für zwei unterschiedliche Zwecke genutzt werden. Darunter fallen PRNGs für allgemeine Zwecke und CSPRNGs für kryptografische Zwecke.

Die PRNGs für allgemeine Zwecke sind von Sicherheitsaspekten unabhängig. Allzweck-PRNGs sind für die meisten IoT-Geräte unverzichtbar. Sie werden in vielen Anwendungsfällen benötigt, beispielsweise für das Jittering von Netzwerkprotokoll-Timern oder Medienzugriffsprotokollen, um Kollisionen auf einem Medium zu vermeiden. Weitere Anwendungen sind zufällige Abtastung von Sensormessungen und Fuzzy-Tests. Das gewünschte Ergebnis eines PRNG ist ein gleichmäßig verteilter Strom statistisch unabhängiger Zufallszahlen. Die verwendeten Seeds zwischen logisch gleich aufgebauten Geräten müssen sich unterscheiden, um ein identisches Zufallsverhalten über alle Geräte hinweg zu vermeiden. Idealerweise sollte darauf geachtet werden bei jedem Neustart des Gerätes eine neue Seed zu verwenden. Die Implementierungen sollten schnell und effizient sein, um die Ressourcen der eingeschränkten Geräte zu schonen. Verfügbare Ressourcen werden besser für Generatoren mit hohen Sicherheitsanforderungen verwendet [16].

Zufallszahlengeneratoren die sicher für Sicherheitsanwendung eingesetzt werden können, werden kryptografisch sichere Zufallszahlengeneratoren (CSPRNGs) genannt. Diese beinhalten die Erzeugung von kryptografischen Schlüsseln, Nonces oder Salts. Es muss für CSPRNGs rechnerisch unmöglich sein, das nächste Bit der Ausgabe mit einer Wahrscheinlichkeit von mehr als 50 % vorherzusagen. Die Sicherheit kryptografischer Systeme beruht auf diesen Zufallszahlen als grundlegende Eingabe. Daher wird von CSPRNGs erwartet, dass sie höchst unvorhersehbare Zahlenfolgen ausgeben und gegen bekannte Angriffe resistent sind. Die Sicherheit einer Implementierung geht über den Umfang der rechnerischen Bemühungen zur Vorhersage künftiger Ausgaben hinaus. Sie umfasst Gegenmaßnahmen zum Schutz vor schwachen Implementierungen sowie zur Kompromittierung des Zustands durch einen Angreifer [16].

Ein CSPRNG verbraucht mehr Systemressourcen und ist komplexer in der Entwicklung als ein PRNG für allgemeine Zwecke. Er umfasst zusätzliche Bausteine wie Chiffren, kryptografische Hash-Funktionen, Laufzeittests und eine besonders robuste Seeding-Logik. Insbesondere der Speicherbedarf aber auch der Rechenaufwand dieser Bausteine stehen in potenziellem Konflikt mit den Ressourcenbeschränkungen von IoT-Knoten. Die Verfügbarkeit sicherer Zufallszahlen ist aber für eine sichere Kommunikation über das Internet unerlässlich. Die vier wichtigsten Aspekte von CSPRNGs sind statistische Zufälligkeit, Unvorhersehbarkeit und Seeds mit hoher Entropie [16].

Die statistische Zufälligkeit gibt an, dass jede statistische Verzerrung zu elementaren Angriffspunkten führt. Ein CSPRNG muss auch dann nicht von echten Zufällen unterscheidbare Sequenzen erzeugen, wenn sie aus deterministischen Algorithmen besteht.

Diese Eigenschaft beruht auf der Annahme, dass in einer Kette von pseudozufälligen Bits die Wahrscheinlichkeiten für 1 und 0 zu jedem Zeitpunkt gleich und statistisch unabhängig sind. Selbst eine sehr kleine Verzerrung muss als potenzieller Verstoß gegen die Annahme der Zufälligkeit betrachtet werden und widerspricht den kryptografischen Anforderungen an einen sicheren Generator [16].

Die Unvorhersehbarkeit bedeutet für CSPRNGs, dass sie gegen externe und interne Angriffe resistent sein müssen. Dabei wird zwischen der Resistenz einer Vorhersage und der Rückverfolgung unterschieden. Unvorhersagbarkeit bedeutet, dass ein Angreifer selbst bei vollkommen bekanntem Algorithmus nicht in der Lage ist, zukünftige Ergebnisse in der Rechenzeit zu erraten. Um dies bei einer bestimmten statistischen Qualität zu erreichen, muss der Seed völlig unvorhersehbar sein. Darüber hinaus müssen CSPRNGs auf kryptografischen Funktionen aufbauen. Dabei handelt es sich in der Regel um Einweg-Hash-Funktionen und Blockchiffren, die praktisch nicht umkehrbar sind und aus verschiedenen Eingaben keine kollidierenden Ergebnisse erzeugen. Die Resistenz der Rückverfolgung schützt nach einer Kompromittierung vor einer Rekonstruktion früherer Werte oder sogar des Seeds. Das bedeutet, dass es keine Korrelation zwischen Seed und generierter Ausgabe geben darf, womit eine perfekte Geheimhaltung in kryptografischen Protokollen gewährleistet wird. Dies wird durch die Anwendung kryptografischer Funktionen auf den internen Zustand des Generators realisiert. Falls vorhanden, kann die Geheimhaltung durch die Speicherung des Zustands in einem geschützten Speicher gehärtet werden [16].

Seeds mit hoher Entropie sind notwendig, um die Ausgabe eines CSPRNG unvorhersehbar zu machen. Als Maß für ihre Zufälligkeit wird die Shannon Entropie oder die Minimalentropie verwendet, wobei die zufälligen Bits aus physischen Ressourcen extrahiert werden. In diesem Zusammenhang werden physikalische Zufallsressourcen oft als Entropiequellen bezeichnet. Es existieren unterschiedliche physikalische Zufallsquellen. Unter anderem nutzen sie Schwankungen in elektronischen Schaltkreisen, wie Taktgeberabweichungen oder initialisierten Speicher. Weiterhin nutzen sie zufällig verrauschte Signale, wie thermisches Rauschen oder Benutzereingabesignale wie Tastenanschläge oder Mausklicks. Diese sind normalerweise nicht im IoT verfügbar [16].

4.3.3 Identifikation und Authentifizierung

In vielen Anwendungen, wie zum Beispiel im RFID-Bereich, wird die Identifizierung von Mikrochips gefordert. Der Chip schickt dabei einem Lesegerät die von der PUF generierte Zahl zu. Mit einer bei der Produktion des Chips erstellten Datenbank wird die empfangene Zahl verglichen und zugeordnet. Um die Wahrscheinlichkeit einer sich wiederholenden Zahl zur Identifizierung zu verringern, kann die Anzahl der ausgegebenen Stellen einer PUF-Antwort erhöht werden. Wenn die Anzahl der Stellen hoch genug und die Anzahl der auftretenden fehlerhaften Bits ausreichend gering ist, besteht weiterhin die Möglichkeit, den Chip mit Hilfe von fehlerkorrigierenden Codes richtig zu identifizieren. Sollten sich bei einem späteren Auslesen zu viele Bits von der Zahl in der Datenbank unterscheiden, kann die ID nicht zugeordnet werden. Im Prinzip wird mit Hilfe eines Lesegerätes der zu identifizierende Tag ausgelesen und mit den IDs aus der Datenbank verglichen. Die ID, mit dem geringsten Unterschied zur ausgelesenen ID, wird als wahre Identität definiert [10].

Bei einer Authentifizierung wird eine Identität überprüft, die behauptet echt zu sein. Bei diesem Verfahren muss eine Datenbank mit Challenge-Response-Pairs (CRP) angelegt werden, noch bevor der Chip im Umlauf ist. Dafür werden Zufallswerte (Challenges) generiert und die Ergebnisse der PUF (Responses) ausgelesen. Eine angelegte Datenbank kann nur eine begrenzte Zahl an CRPs speichern und aus sicherheitstechnischen Gründen kann jedes CRP nur einmal verwendet werden. Daher ist auch die Anzahl der möglichen Authentifizierungen begrenzt. Sollten alle Authentifizierungen aufgebraucht sein, können neue CRP generiert und in die Datenbank eingetragen werden [10].

Bei kryptografischen Operationen werden PUFs für die Generierung von Schlüsseln verwendet. Hierbei besteht die Notwendigkeit, dass der Schlüssel immer exakt derselbe ist. Sollte bereits ein Bit falsch sein, wird der Schlüssel unbrauchbar. Damit die Reproduzierbarkeit garantiert werden kann, ist eine mathematische Nachbearbeitung des Ergebnisses notwendig. Dafür werden beim erstmaligen Auslesen der PUFs Hilfsdaten zur Fehlerkorrektur erzeugt. Wenn nicht zu viele Fehler auftreten, kann mit den Hilfsdaten die erstmalige ausgelesene Zahl wiederhergestellt werden. Der Vorteil bei dieser Verwendung von PUFs ist die hohe Sicherheit des Systems. Es ist nahezu unmöglich den Schlüssel im stromlosen Zustand auszulesen, da der Schlüssel durch die von Prozessschwankungen abhängigen PUFs generiert wird [10].

4.4 Fuzzy-Extraktor

PUFs sind von Natur aus verrauscht und müssen mit Fehlerkorrekturmechanismen wie Fuzzy-Extraktoren kombiniert werden. Diese entfernen die Auswirkungen des Rauschens, bevor die PUF-Antwort in einem kryptografischen Algorithmus verarbeitet werden kann [14]. Die wichtigsten Hauptfunktionen eines Fuzzy-Extraktors sind das Verschlüsseln eines Geheimnisses in der Enrollment-Phase sowie der Informationsabgleich für die Rekonstruktion eines Geheimnisses [18]. Mit einem Fuzzy-Extraktor bieten PUFs eine signifikant höhere physische Sicherheit. Andere Methoden, die unvorhersehbare Informationen in nicht-flüchtigen Speicher halten, können jederzeit erneut ausgemessen werden. Deshalb werden sie beispielsweise auch als Speichermechanismus für kryptografische Schlüssel verwendet. Bei dieser Anwendung wird zwischen zwei Phasen unterschieden: dem Enrollment und der Rekonstruktion [25].

4.4.1 Enrollment

Der Enrollment-Prozess wird vom Hersteller oder dem Systemintegrator für jedes Gerät einmal durchgeführt. Er dient zwei Hauptzwecken: einerseits der Ableitung des gerätespezifischen Schlüssels K und andererseits der Erzeugung der so genannten Hilfsdaten W. Der kryptografische Schlüssel wird aus einem zufällig gewählten Geheimnis S abgeleitet. Das Geheimnis wird als Eingabe in eine kryptografisch sichere Hash-Funktion $H(\cdot)$ eingespeist: $K \leftarrow H(S)$. Das Geheimnis wird vom Hersteller vordefiniert und muss für jedes Gerät eindeutig sein [24].

In Abbildung 4.2 wird der Enrollment-Prozess dargestellt. Die PUF-Antwort wird als Referenzmessung (R) bezeichnet und ist der Input für den Fuzzy-Extraktor. Der Fuzzy-Extraktor (FE) leitet einen kryptografischen Schlüssel aus einem zufälligen Geheimnis ab und berechnet Hilfsdaten durch eine XOR-Verknüpfung des verschlüsselten Geheimnisses mit der Referenz-PUF-Antwort. Die Hilfsdaten werden in einem nicht-flüchtigen Speicher auf dem Gerät selbst gespeichert. Dabei repräsentieren die Hilfsdaten eine öffentlich zugängliche Information [18]. Die öffentlich zugänglichen Informationen geben dabei keine Informationen über den Schlüssel preis [24].

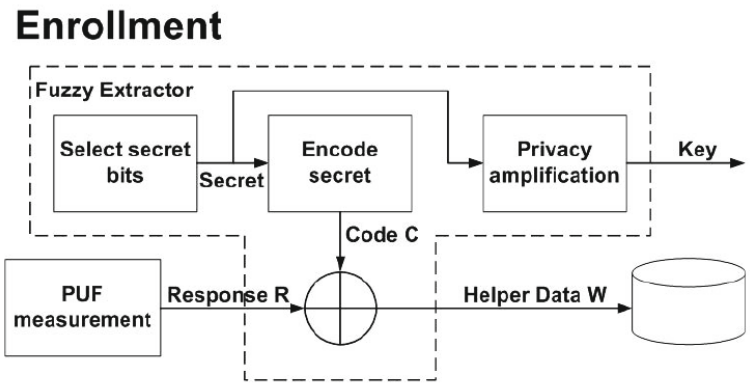

Abbildung 4.2: Ablauf der Enrollment-Phase [18]

4.4.2 Rekonstruktion

Der Rekonstruktions-Prozess wird auf der Benutzerseite durchgeführt und erfolgt jedes Mal, wenn der Benutzer das entsprechende Gerät startet [24]. In der Rekonstruktions-Phase wird, wie in Abbildung 4.3 dargestellt, die gleiche PUF erneut ausgemessen. Die PUF-Antwort wird in den Fuzzy-Extraktor eingespeist. Dabei unterscheidet sich eine neue PUF-Messung (R') geringfügig von der Referenzmessung. Der Fuzzy-Extraktor verwendet die Hilfsdaten und die neue PUF-Messung, um den während der Registrierung einprogrammierten kryptografischen Schlüssel zu rekonstruieren. Wenn die neue PUF-Messung nahe genug an der Referenzmessung liegt, kann der ursprüngliche Schlüssel durch einen Informationsabgleich erfolgreich rekonstruiert werden [18].

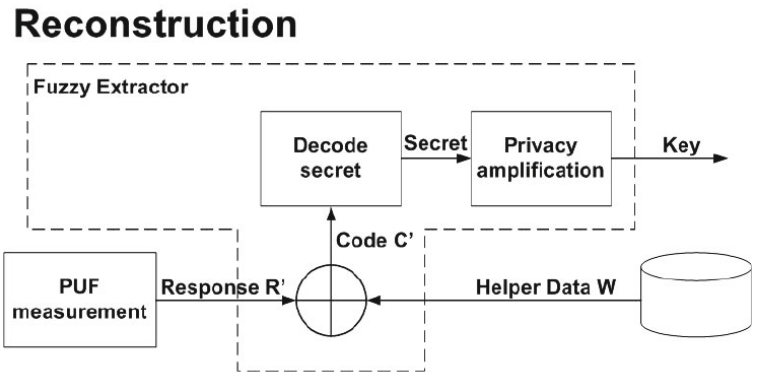

Abbildung 4.3: Ablauf der Rekonstruktions-Phase [18]

Der Informationsabgleich (Decode-Secret) wird durchgeführt, nachdem eine neue PUF-Messung mit den Hilfsdaten XOR-verknüpft wurde. Dadurch wird ein neues Geheimnis (C') erzeugt, welches sich vom verschlüsselten Geheimnis (C) an denselben Positionen unterscheidet, genauso wie eine neue PUF-Messung sich von der Referenzmessung unterscheidet. Wenn eine neue PUF-Messung und die Referenzmessung ausreichend nahe beieinander liegen, kann ein neues Geheimnis in das verschlüsselte Geheimnis korrigiert und somit auch während des Enrollment dekodiert werden [18].

5 Der PUF Seed Generator in RIOT

Das Betriebssystem RIOT stellt Standardnetzwerke für IoT-Geräte zur Verfügung, mit denen handelsübliche Mikrocontroller über das IoT mit dem globalen Internet verbunden werden können. Die Vernetzung von IoT-Geräten und kryptografische Operationen erfordern wie bei den meisten Betriebssystemen Zufallsdaten mit sehr geringer Vorhersagbarkeit. Da die meisten IoT-Geräte mit einer deterministischen Uhr laufen und fast keine Unterscheidungsmerkmale aufweisen, stellt die Generierung nicht vorhersagbarer individueller Zahlen auf Mikrocontroller eine Herausforderung dar. Für dieses Problem wurde für das Betriebssystem RIOT ein Generator für Zufalls-Seeds entwickelt, der auf PUFs basiert. Dazu wurde ein generisches PUF-Extraktor Modul[1] auf der Basis von SRAM-Speicher entwickelt. Das Modul klinkt sich direkt vor der Kernel-Initialisierung in den Startcode des Betriebssystems ein. Die Startmuster eines erhaltenen und eines nicht initialisierten SRAM-Blocks werden in eine Hash-Funktion eingespeist, um den gewünschten Seed-Wert abzuleiten, wie in Abbildung 5.1 dargestellt [15].

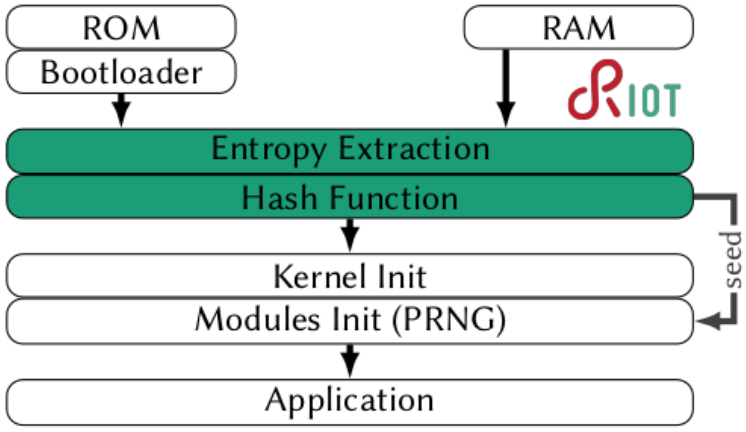

Abbildung 5.1: PUF random seeder in RIOT [15]

[1] https://doc.riot-os.org/group__sys__puf__sram.html, abgerufen am 02-10-2022

Das Modul wurde nach den Bewertungskriterien der PUF Eigenschaften Zuverlässigkeit und Einzigartigkeit evaluiert. Für die Bewertung dieser Metriken wurden wiederholt Experimente auf fünf Platinen des Atmel SAMD21[2] Mikrocontrollers durchgeführt. Von den Mikrocontrollern wurde 1 Kilobyte SRAM ausgelesen, bevor ein RIOT-spezifischer Code ausgeführt wurde. Dabei wurden sie 50-mal für 30 Sekunden ausgeschaltet, bevor sie für die nächste Messung wieder eingeschaltet wurden [15].

Für die Evaluierung der Zuverlässigkeit wurde das Hamming-Gewicht verwendet und das Ergebnis lag bei etwa $0,5$. Das deutet darauf hin, dass die SRAM-Zellen nicht auf einen bestimmten Zustand ausgerichtet sind. Für die Evaluierung der Einzigartigkeit wurde die normalisierte Hamming-Distanz mit unterschiedlichen Geräten mit einem Referenzgerät A verglichen. Dabei wurden die Abweichungen zwischen den Zellen mit derselben Speicheradresse untersucht und das Ergebnis wurde auf die untersuchte Länge von 1 Kilobyte normiert. Die Werte lagen zwischen $49,2$ % und $50,4$ %. Sie deuten auf eine geringe Korrelation zwischen den Mustern hin [15].

[2] https://ww1.microchip.com/downloads/en/DeviceDoc/SAM_D21_DA1_Family_DataSheet_DS40001882F.pdf, abgerufen am 03.10.2022

6 Experimente mit der IoT-Testplattform FIT IoT-LAB

Für die Experimente zum Ausmessen von SRAM PUF wurde ein auf GitHub bereitgestellter Test[1] für die Aktivierung von Low-Power-Modes angepasst. Mit diesem können uninitialisierte Speicherbereiche des SRAM ausgemessen werden. Durch die Aktivierung von Low-Power-Modes entfällt dabei die Notwendigkeit zusätzlicher Hardware als Schnittstelle zwischen einem Mikrocontroller und einem externen Computer. Für die Evaluierung der PUF-Eigenschaften werden Datensätze benötigt, die durch unterschiedliche Experimente erzeugt wurden. Die Experimente wurden auf einer IoT-Testplattform mit mehreren Hundert Testgeräten vom Typ IoT-LAB-M3 durchgeführt. Dabei wurden die Daten über die serielle Schnittstelle (UART) des Testgerätes ausgegeben und von einem sogenannten Serial-Aggregator zusammengefasst.

6.1 Die Testplattform FIT IoT-LAB

Das FIT IoT-LAB [5] ist eine öffentliche Testplattform, die aus über 2700 drahtlosen Low-Power-Knoten und 117 mobilen Robotern besteht. Die Plattform steht für Experimente mit drahtlosen IoT-Technologien im großen Maßstab zur Verfügung. Das IoT-LAB-Testbed wird an 6 Standorten in ganz Frankreich eingesetzt. Jeder Standort verfügt über unterschiedliche Knoten und Hardwarekapazitäten. Dabei sind alle Standorte miteinander verbunden und über dasselbe Webportal, gemeinsame REST-Schnittstellen und einheitliche CLI-Tools verfügbar. Das Ergebnis ist eine heterogene Testumgebung, die ein breites Spektrum von IoT-Anwendungsfällen und Applikationen abdeckt. IoT-LAB ist eine einzigartige Einrichtung, die es jedem ermöglicht seine Lösung in großem Maßstab zu testen, zu experimentieren und neue Netzwerkkonzepte abzustimmen. Als hochmoderne

[1] https://github.com/PeterKietzmann/RIOT/tree/pr_puf_id_gen/tests/puf_sram, abgerufen am 03-10-2022

Testumgebung ist das IoT-LAB darauf ausgerichtet, die Bedürfnisse und Anforderungen der heutigen und zukünftigen IoT-Technologie zu erfüllen [1].

Die Testplattform bietet drei Hauptmerkmale. Das Erste ist die heterogene und reichhaltige Umgebung, die für ein breites Spektrum von IoT-Anwendungen geeignet ist. Darunter fallen Hardware, Topologien, Betriebssysteme, aktuelle standardisierte Protokollstapel und Bibliotheken. Als Zweites besitzt sie die Fähigkeit laufende Experimente zu verwalten, mit ihnen zu interagieren und sie zu überwachen. Darüber hinaus bietet die Testplattform die Fähigkeit ein Experiment durch Visualisierungs- und Reproduzierbarkeitswerkzeuge zu steuern [1]. Außerdem bietet die Plattform CLI-Tools an, mit denen per Fernzugriff Experimente übermittelt und einzelne Knoten ausgeschaltet, eingeschaltet, programmiert oder zurückgesetzt werden können. Ein weiterer Teil der CLI-Tools ist der Serial-Aggregator, mit dem der Output der Testgeräte eines Experimentes über die serielle Schnittstelle (UART) aggregiert werden kann. Dies vereinfacht das Sammeln und Erzeugen von Datensätzen mit mehreren Hundert Testgeräten [2].

6.2 Das IoT-LAB-M3 Entwicklungsboard

Das für diese Arbeit verwendete Entwicklungsboard ist das IoT-LAB-M3 [6], das speziell für die IoT-LAB Testplattform entwickelt wurde. Es basiert auf einem STM32 [26] (ARM Cortex M3) Mikrocontroller mit einer ATMEL Funkschnittstelle in 2,4 GHz und vier Sensoren, wie in Abbildung 6.1 dargestellt.

Das IoT-LAB-M3 unterstützt, wie in Abbildung 6.2 zu sehen ist, einen externen 128-MBit-NOR-Flash-Speicher (N25Q128A13E1240F). Dieser ist über den SPI-Bus mit der MCU verbunden, um eine schnelle Datenübertragung zu ermöglichen. Mit diesem Speicher werden üblicherweise von Sensoren erfasste Daten gespeichert, während ein Programm läuft. Ein weiterer Anwendungsfall ist das Over-The-Air-Update für die Firmware der MCU. Einige eingebettete Betriebssysteme partitionieren den ROM-Speicher, um verschiedene Versionen der Firmware herunterzuladen und zu speichern [6].

Der Cortex-M3 ist eine 32-Bit-CPU, mit einer Taktrate von bis zu 72 MHz. Es ist in den STM32 MCU integriert und wird von STMicroelectronis hergestellt. Diese MCU ist mit 64KB RAM und 256KB ROM ausgestattet. Der Cortex-M3 unterstützt drei verschiedene LPM: Sleep, Stop und Standby. Im Sleep-Modus wird nur die CPU gestoppt.

[2] https://iot-lab.github.io/docs/tools/serial-aggregator/, abgerufen am 03-10-2022

Alle weiteren mit der CPU verbundene Peripherien können ihre Operationen fortführen. Der Stop-Modus erreicht einen niedrigen Stromverbrauch, wobei der Inhalt des SRAM erhalten bleibt. Im Standby-Modus wird auch ein niedriger Stromverbrauch erreicht. Allerdings wird der SRAM abgeschaltet, wodurch sich der Speicherinhalt verflüchtigen kann [4].

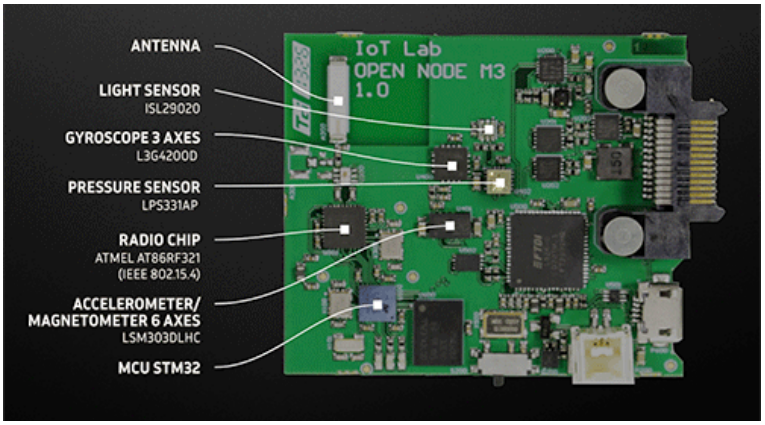

Abbildung 6.1: IoT-LAB M3 mit verschiedenen verbauten Sensoren und einer STM32 MCU [6]

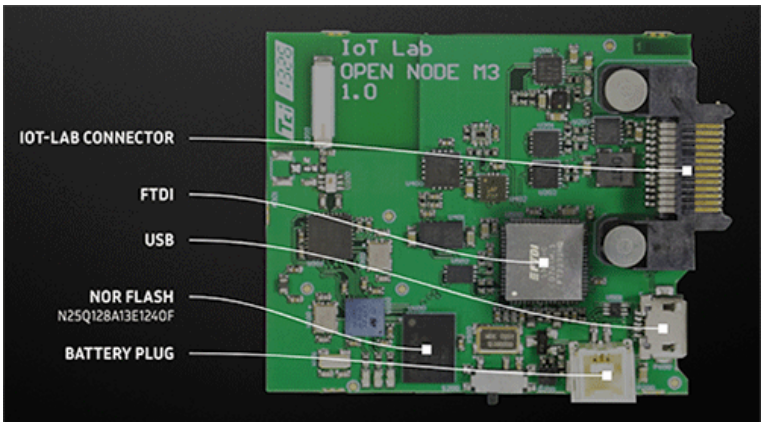

Abbildung 6.2: IoT-LAB M3 mit einem NOR-Flashspeicher [6]

Der STM32 kann, wie in der Architektur der Abbildung 6.3 zu sehen ist, über den an USB angeschlossenen FTDI2232H über JTAG zurückgesetzt, debugged und programmiert werden. Diese Komponente ermöglicht auch eine UART-Verbindung zum STM32 [6].

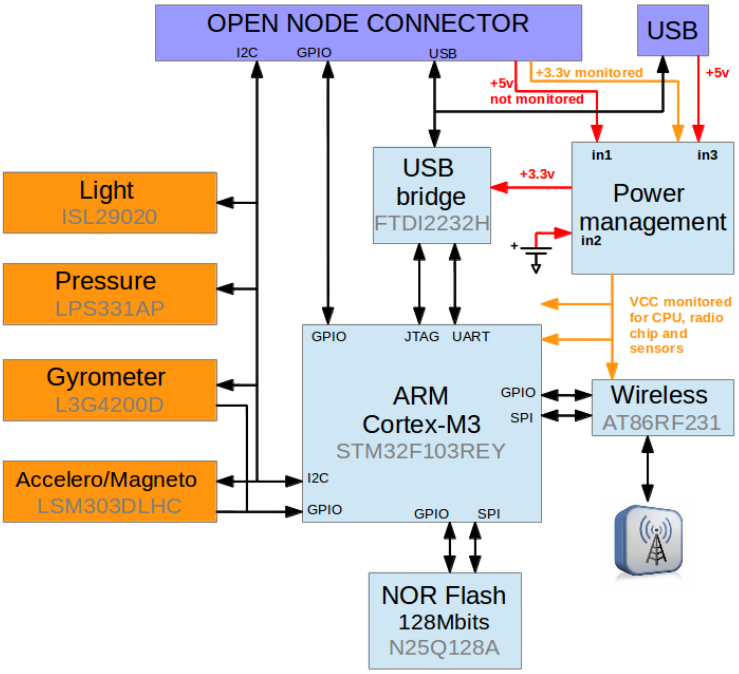

Abbildung 6.3: Architektur des IoT-LAB-M3 Boards [6]

6.3 Ausmessen von SRAM-Startwerten mit aktiviertem Low-Power-Mode

Der Ablauf zum Ausmessen von SRAM-Startwerten wird in Abbildung 6.4 dargestellt. Nach dem Einschalten des Gerätes wird der Bootvorgang gestartet. Zu Beginn des Bootvorgangs wird ein SRAM-Speicherbereich von 1 Kilobyte ausgelesen. Dies passiert noch bevor der erste Code mit Schreibzugriff auf den SRAM ausgeführt wird. Nach dem Auslesen wird der Bootvorgang bis zum Ende fortgesetzt, bis die Kontrolle an den Main-Thread übergeben wird. Nach dem Betreten der Hauptfunktion wird das zuvor ausgelesene SRAM-Startmuster über die serielle Schnittstelle ausgegeben. Danach wird mit dem RTC-Modul ein Alarm eingestellt, der nach Ablauf von 30 Sekunden einen Interrupt auslöst. Zum Schluss wird im Kontext der Hauptfunktion über das Power-Management von RIOT der LPM gesetzt, wodurch das Testgerät in den Standby-Modus übergeht. Da im Standby-Modus des IoT-LAB-M3 Mikrocontrollers die SRAM-Komponente abgeschaltet wird, verflüchtigt sich sein Speicherinhalt. Nach Ablauf der Zeit und auslösen des eingestellten RTC-Alarms, wird der Standby-Modus unterbrochen. Dadurch kehrt das Testgerät in den Normalbetrieb zurück und der Bootvorgang wird im Reset-Handler erneut

gestartet wird. Dieser Vorgang kann für eine beliebige Anzahl an Messungen durchgeführt werden. Für die Erzeugung von Datensätzen wurde dieser Vorgang in den Experimenten 100-mal ausgeführt.

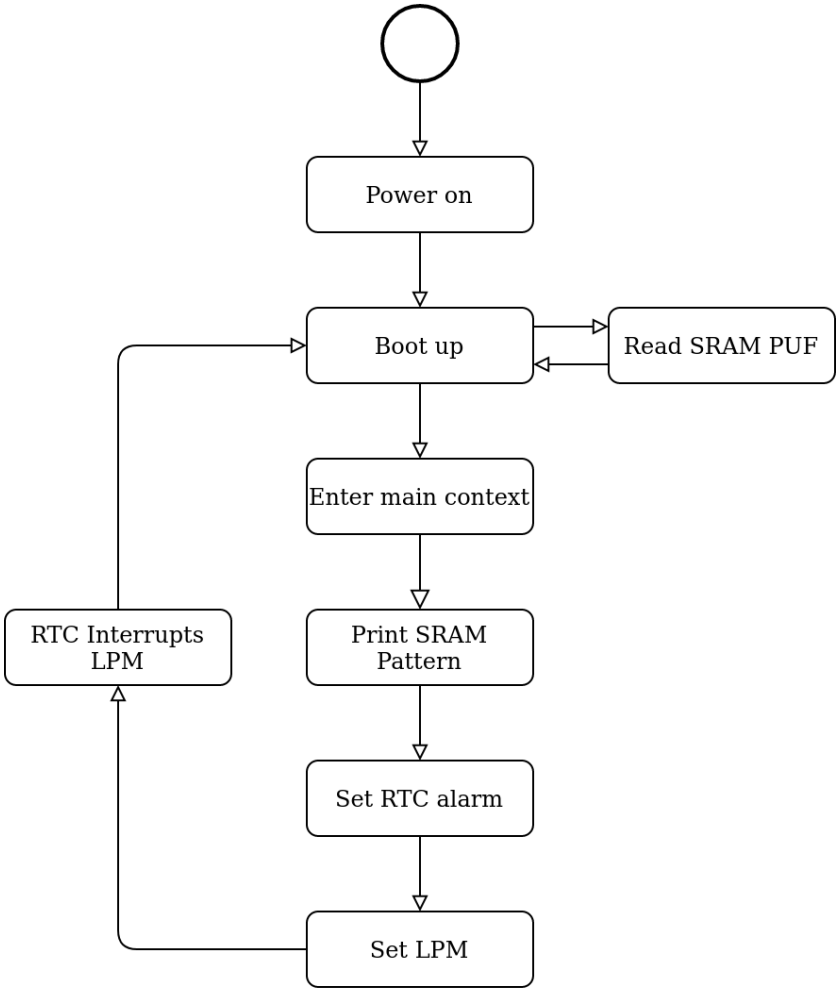

Abbildung 6.4: Ablaufdiagramm für das Ausmessen von SRAM-Startwerten mit LPM

6.4 Erzeugen von Identitäten mit aktiviertem Low-Power-Mode

Der Ablauf zum Erzeugen einer Identität (ID) mit aktiviertem LPM wird in Abbildung 6.5 dargestellt. Zuerst wird die Stromversorgung des Gerätes hergestellt, worauf der Bootvorgang gestartet wird. Auch hier wird zu Beginn des Bootvorgangs noch vor der ersten Speicherinitialisierung ein SRAM-Speicherbereich von 1 Kilobyte ausgelesen. Das Startmuster wird in einer gesonderten Datensektion im SRAM gespeichert, wodurch es später im Main-Thread wieder abgerufen werden kann. Nach Beendigung des Bootvorgangs wird die Kontrolle an den Main-Thread übergeben. Im Main-Thread wird zunächst überprüft, ob bereits Hilfsdaten erzeugt wurden. Sollten keine Hilfsdaten vorhanden sein, so wird das Enrollment ausgeführt, anderenfalls die Rekonstruktion.

In der Enrollment-Phase werden mit Hilfe des zuvor gespeicherten Startwertes die Hilfsdaten auf dem Gerät selbst generiert. Die Hilfsdaten werden anschließend mit dem MTD-Modul persistent im Flash-Speicher des Testgerätes hinterlegt. So können diese nach Verlassen eines LPM wieder abgerufen werden. Die Generierung der Hilfsdaten geschieht einmalig zur Laufzeit des Experiments des jeweiligen Testgerätes.

Die Rekonstruktions-Phase wird bei jedem Start ausgeführt, wenn das Testgerät bereits einmal die Prozedur der Enrollment-Phase durchlaufen hat. Die Hilfsdaten werden aus dem Flash-Speicher gelesen und über die API[3] des PUF-Moduls wird eine ID generiert. Nach der Enrollment- oder Rekonstruktionsphase wird die erzeugte ID über die serielle Schnittstelle (UART) ausgegeben.

Danach wird auch in diesem Experiment der Alarm der RTC eingestellt, sodass nach Ablauf von 30 Sekunden ein Interrupt ausgelöst wird. Unmittelbar nach dem Einstellen des Timers, wird über das Power-Management Modul das Testgerät in den Standby versetzt. Auch hier verflüchtigt sich der Speicherinhalt des SRAMs im Standby-Modus. Nach dem Ablauf des Timers wird der LPM unterbrochen und das Testgerät geht wieder in den Normalbetrieb zurück. Dadurch wird der Bootvorgang im Reset-Handler erneut gestartet. Dieser Vorgang kann, wie bei dem Experiment davor, ebenfalls beliebig oft wiederholt werden. Für die Erzeugung einer ID wurde dieser Ablauf für das jeweilige Testgerät 100-mal durchgeführt.

[3] https://github.com/PeterKietzmann/RIOT/blob/pr_puf_id_gen/sys/include/puf_sram.h, abgerufen am 01-10-2022

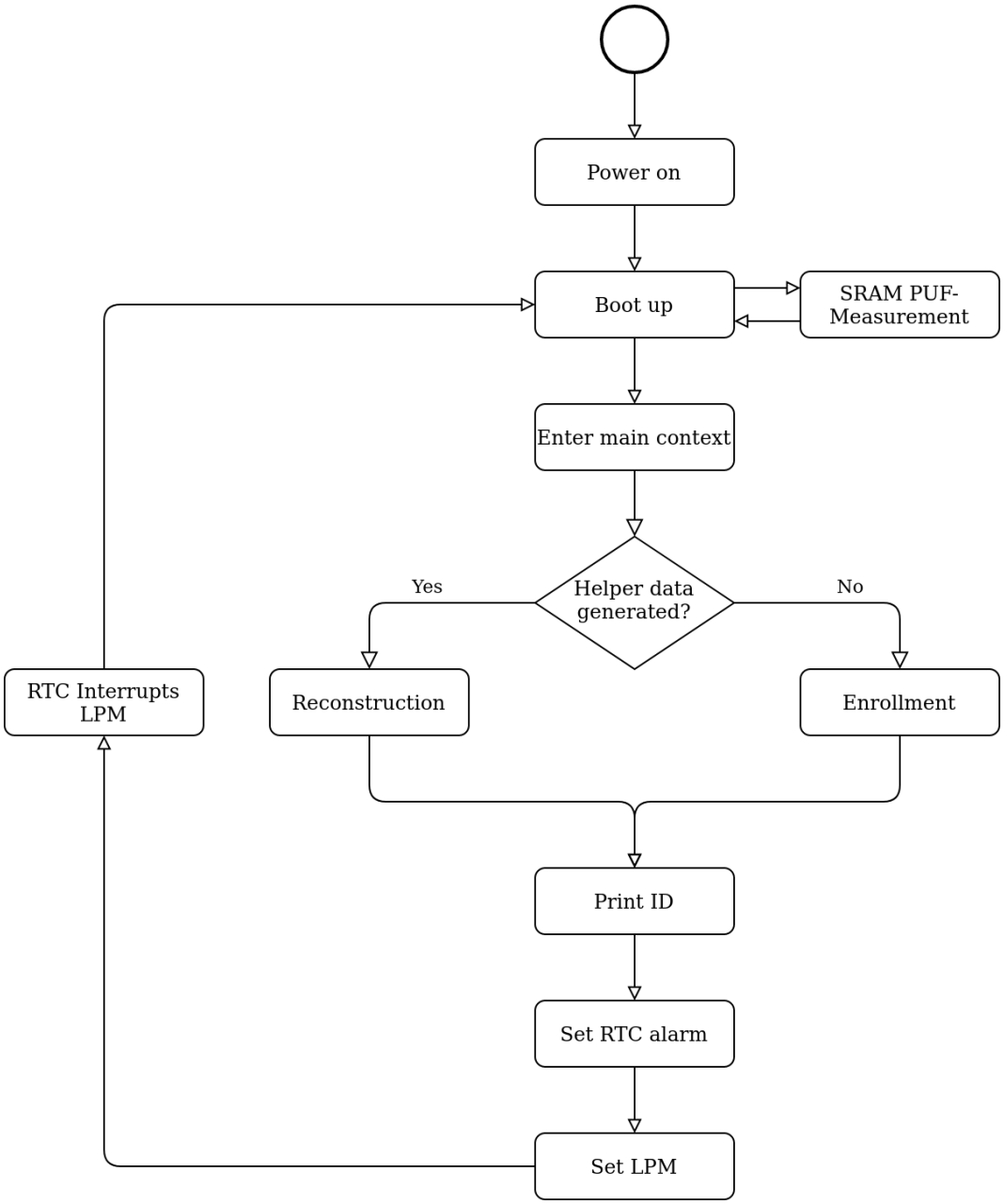

Abbildung 6.5: Ablaufdiagramm der Testfirmware

7 Evaluation von SRAM PUF mit Low-Power-Modes

Die wichtigsten Eigenschaften zum Bewerten einer PUF sind die Zuverlässigkeit und die Einzigartigkeit. Für die Zuverlässigkeit müssen erkennbare Antworten über mehrere Wiederholungen auf demselben Gerät herausgefordert werden, die oft durch Rauschen beeinflusst werden. Für die Einzigartigkeit wird die Korrelation der Ergebnisse über mehrere Geräte hinweg quantifiziert [15].

Die Evaluierung der grundlegenden PUF-Funktionen wurde mit einem geräteinternen Test (Intra-Chip) für die Zuverlässigkeit und einem geräteübergreifenden Test (Inter-Chip) für die Einzigartigkeit in einem Python-Skript durchgeführt. Für den Intra-Chip Test und den Inter-Chip Test wurde die Hamming-Distanz verwendet. Weiterhin wurde ein Bias-Test durchgeführt, um herauszufinden, ob die SRAM-Speicher der verwendeten Testgeräte gegen Eins oder Null verzerrt sind. Bei der Berechnung des Bias-Tests wurde das Hamming-Gewicht verwendet.

Für die Zuverlässigkeit wurde evaluiert, inwiefern sich die Startmuster durch das Rauschen der SRAM-Zellen bei aufeinanderfolgenden Messungen unterscheiden. Für die Einzigartigkeit wurde die Korrelation eines SRAM-Speichers von einem Testgerät A mit allen anderen Testgeräten evaluiert.

7.1 Die Hamming-Distanz und das Hamming-Gewicht

Eine PUF-Messung X, die aus einer SRAM-basierten PUF gewonnen wird, wird als Zeichenkette aus Nullen und Einsen dargestellt. Die Standardmetriken zur Bewertung SRAM-basierter PUFs basieren auf der Hamming-Distanz $HD(X_i, X_j)$, die zwischen Paaren von Messungen (X_i, X_j) berechnet wird [24].

Die Hamming-Distanz HD zwischen zwei Symbolfolgen $a = \{a_1, a_2, ..., a_n\}$ und $b = \{b_1, b_2, ..., b_n\}$ der gleichen Länge n, welche aus dem Alphabet Σ besteht, bezeichnet die Anzahl der Stellen, an denen sich die beiden Symbolfolgen unterscheiden [24]:

$$HD(a, b) \stackrel{\text{def}}{=} |\{i \in \{1, 2, ..., n\} | a_i \neq b_i\}|.$$

Das Hamming-Gewicht HW ist eine Metrik, die die Anzahl der Symbole die nicht Null sind in einer Folge von Symbolen $a = \{a_1, a_2, ..., a_n\}$ summiert. Dabei wird angenommen, dass a eine Zeichenkette ist, sodass HW die Anzahl der Elemente von a angibt, die nicht Null sind [24]:

$$HW(a) \stackrel{\text{def}}{=} |\{i \in \{1, 2, ..., n\} | a_i \neq 0\}|.$$

7.2 Berechnung der Metriken in Python

Für die Hamming-Distanz wurde, wie im Listing 7.1 dargestellt, im Python-Skript eine Funktion implementiert. Die Funktion hat zwei Parameter: A und B. Diese repräsentieren jeweils eine Zeichenkette die miteinander verglichen werden, um die Hamming-Distanz zu berechnen. Dazu wird zuerst geprüft, ob A und B die gleiche Länge besitzen. Danach wird die Summe gebildet, indem über alle Elemente von A und B iteriert wird. Dabei werden die Werte an den gleichen Positionen der beiden Zeichenketten XOR-verknüpft. Die daraus resultierenden Einsen werden aufsummiert und bilden den Rückgabewert und die ermittelte HD von zwei miteinander verglichenen Zeichenketten.

```
1 def hamming_distance(a: list, b: list) -> int:
2     assert len(a) == len(b)
3     return sum(bin(a[i] ^ b[i]).count("1") for i in range(len(a)))
```

Listing 7.1: Berechnung der Hamming-Distanz in Python

Für das Hamming-Gewicht wurde ebenfalls eine Funktion implementiert, wie in Listing 7.2 dargestellt. Die Funktion hat einen Parameter X, der eine Zeichenkette repräsentiert. Die Funktion bildet die Summe der gezählten Einsen über alle Elemente der Zeichenkette X. Die berechnete Summe stellt den Rückgabewert und den ermittelten Wert des HW dar.

```
def hamming_weight(x: list) -> int:
    return sum(bin(x[i]).count("1") for i in range(len(x)))
```

Listing 7.2: Berechnung des Hamming-Gewichts in Python

7.3 Bias-Test mit dem Hamming-Gewicht

Für den Bias-Test mit dem Hamming-Gewicht soll evaluiert werden, ob die Startwerte in Richtung des Wertes Null oder Eins verzerrt sind. Diese Metrik gibt einen ersten Hinweis über die Zufälligkeit der Startwerte. Das ideale Hamming-Gewicht folgt einer Gaußschen Verteilung mit einem Mittelwert von 50 %. Solch eine Verteilung deutet darauf hin, dass die Einschaltwerte nicht verzerrt sind [24].

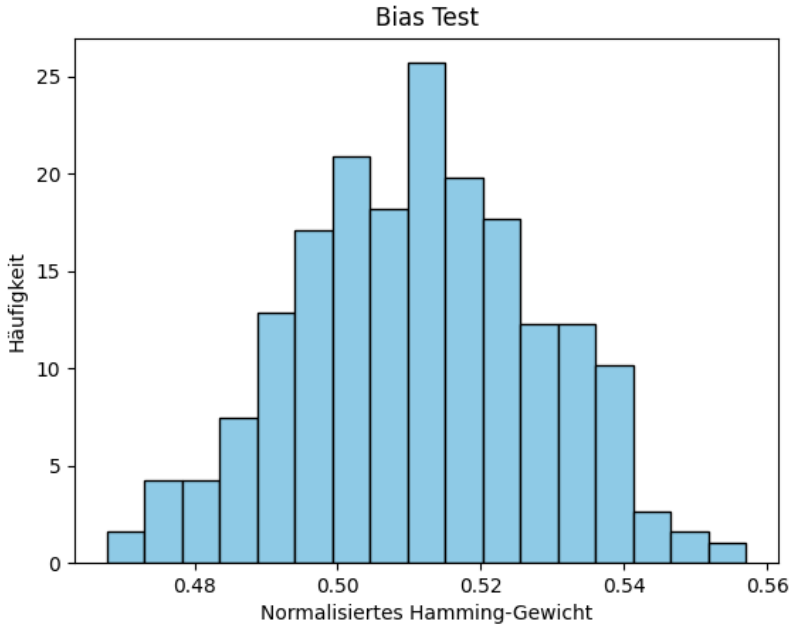

Abbildung 7.1: Histogramm des Intra-Chip Experiments des Hamming-Gewichts von 355 Testgeräten mit je 100 SRAM-Startmuster.

Das Ergebnis des durchgeführten Bias-Tests ist im Anhang A.1 angefügt. Dieses repräsentiert das Hamming-Gewicht der Startmuster von jeweils 355 Testgeräten, bei denen 100-mal ein LPM aktiviert wurde. In Abbildung 7.1 wird das Ergebnis für die Quantifizierung des Bias mit dem Hamming-Gewicht dargestellt. Die Abbildung zeigt eine

Verteilung des normalisierten Hamming-Gewichts um den Mittelwert von 50 % . Da das ideale Hamming-Gewicht einer Gaußschen Verteilung mit einem Mittelwert von 50 % entspricht, deuten die Testergebnisse darauf hin, dass die Startwerte nahezu unverzerrt sind.

7.4 Intra-Chip Test mit der Hamming-Distanz

Die Intra-Hamming-Distanz spiegelt die Stabilität wiederholter Messung einer PUF für ein einzelnes Gerät wieder, wenn diese durch eine feste Herausforderung abgefragt wird. Die Robustheit der Startwerte ist erforderlich, um ein bestimmtes Gerät zuverlässig zu identifizieren und anschließend den entsprechenden kryptografischen Schlüssel zu rekonstruieren. Die Intra-Hamming-Distanz ist eine normalisierte Anzahl von Bits, die sich zwischen aufeinanderfolgenden PUF-Messungen unterscheiden. Sie ist somit eine rationale Zahl zwischen Null und Eins [24].

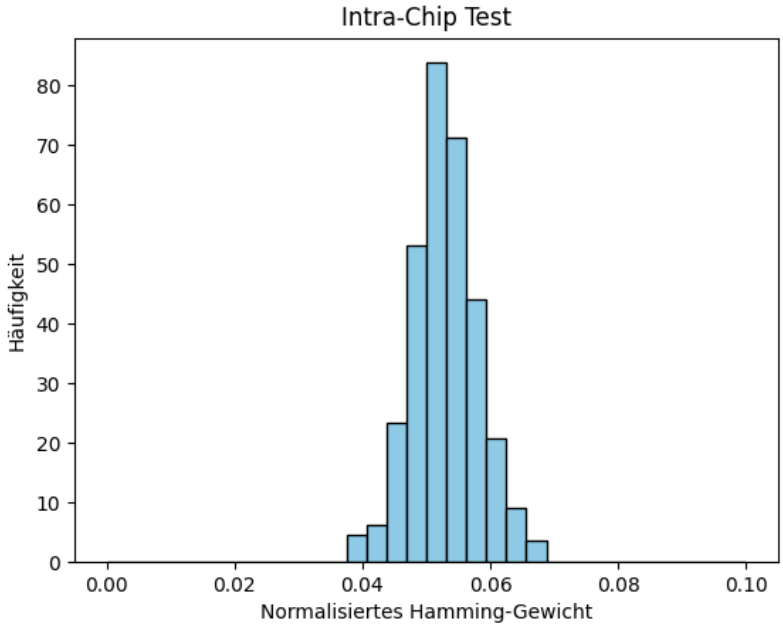

Abbildung 7.2: Histogramm des Intra-Chip Experiments des Hamming-Gewichts von 355 Testgeräten mit je 100 SRAM-Startmuster.

Ein optimaler Wert für die Intra-Hamming-Distanz liegt nahe bei null. Die meisten PUF-Messungen weisen jedoch ein gewisses Maß an Rauschen auf. Im Falle der SRAM-

basierten PUF ist dies auf die Zellen zurückzuführen, die eine eher symmetrische Anordnung der Transistoren aufweisen. Das Rauschen spiegelt sich im Umkippen der Startwerte über mehrere Versuche wieder [24]. Dies gibt Aufschluss über die Reproduzierbarkeit der PUF-Antworten [27].

Das Ergebnis des durchgeführten Intra-Chip Tests mit der Hamming-Distanz ist im Anhang A.2 angefügt. Dieses repräsentiert die Hamming-Distanz der Startmuster von jeweils 355 Testgeräten, bei denen 100-mal ein LPM aktiviert wurde. In Abbildung 7.2 wird das Ergebnis des Intra-Chip Tests dargestellt. Dieses zeigt, inwieweit sich die Startmuster durch das Rauschen der SRAM-Zellen bei aufeinanderfolgenden Messungen unterscheiden. Ein optimaler Wert für die Intra-Hamming-Distanz wird für nahe null angegeben. Bei dem durchgeführten Test liegen die Ergebnisse zwischen $0,038$ % und $0,067$ %.

7.5 Inter-Chip Test mit der Hamming-Distanz

Für den Inter-Chip Test wird geräteübergreifend die Einzigartigkeit geprüft, indem verglichen wird, wie viele PUF-Ausgangsbits sich zwischen einem PUF in einem Gerät A und einem PUF in einem Gerät B unterscheiden [27].

Wenn die PUF gleichmäßig verteilte unabhängige zufällige Bits produziert, sollte die Variation zwischen unterschiedlichen Chips durchschnittlich 50 % betragen [27]. Das heißt es soll ausgeschlossen werden, dass ein Angreifer in der Lage ist, eine Messung für ein zweites Gerät auf der Grundlage der Messungen des ersten Gerätes vorherzusagen [24].

Das Ergebnis des durchgeführten Inter-Chip Tests mit der Hamming-Distanz ist im Anhang A.3 angefügt. Dieses repräsentiert die Hamming-Distanz der Startmuster von jeweils 355 Testgeräten, bei denen 100-mal ein LPM aktiviert wurde. Dabei wurde die PUF-Messung von einem Testgerät A in Korrelation mit den PUF-Messungen der anderen Testgeräte gestellt. In Abbildung 7.3 wird das Ergebnis des Inter-Chip Test dargestellt. Bei dem Inter-Chip Test wurden Werte zwischen $44,1$ % und $51,7$ % gemessen. Dabei lag der Mittelwert bei $47,75$ %.

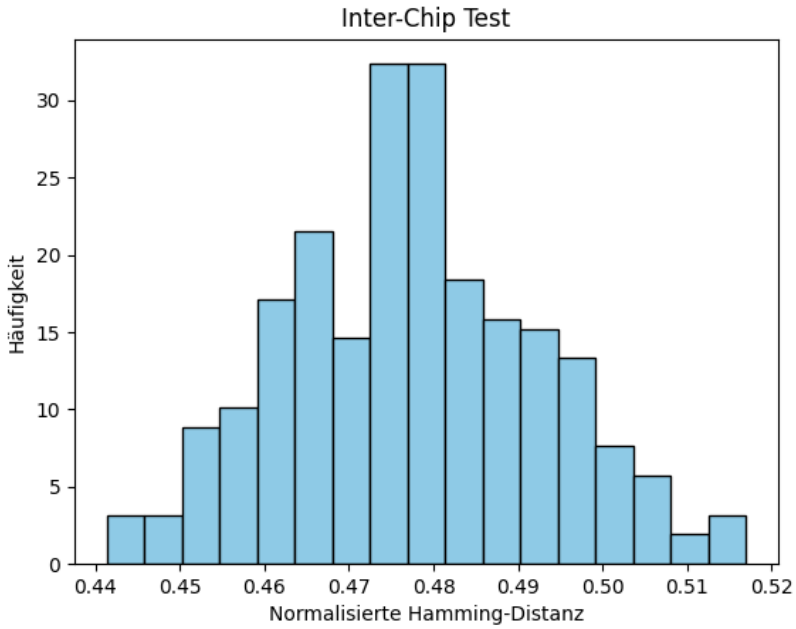

Abbildung 7.3: Histogramm des Inter-Chip Experiments der Hamming-Distanz mit 355 Testgeräten mit je 100 Startmustern

7.6 Inter-Chip Test mit erzeugten IDs von 194 Testgeräten

Im Anhang A.4 werden die Ergebnisse der Hamming-Distanz der erzeugten IDs von 194 Testgeräten aufgezeigt. Im Gegensatz zu den anderen drei Tests basiert dieser Test nicht auf den Startwerten des SRAMS. Dieser Test basiert auf den mit den Hilfsdaten generierten IDs. Durch die Aktivierung von LPM konnte evaluiert werden, dass Hilfsdaten auf dem Gerät selbst erstellt werden können. Erwartet wurde ein Ergebnis mit durchschnittlich 50 %.

Dieser Test hat gezeigt, dass die Generierung von IDs nach Erstellung von Hilfsdaten auf dem Gerät selbst funktioniert. Das Ergebnis verteilt sich um den Mittelwert von $49,7$ % liegt, was nahe an dem erwarteten Ergebnis von 50 % liegt.

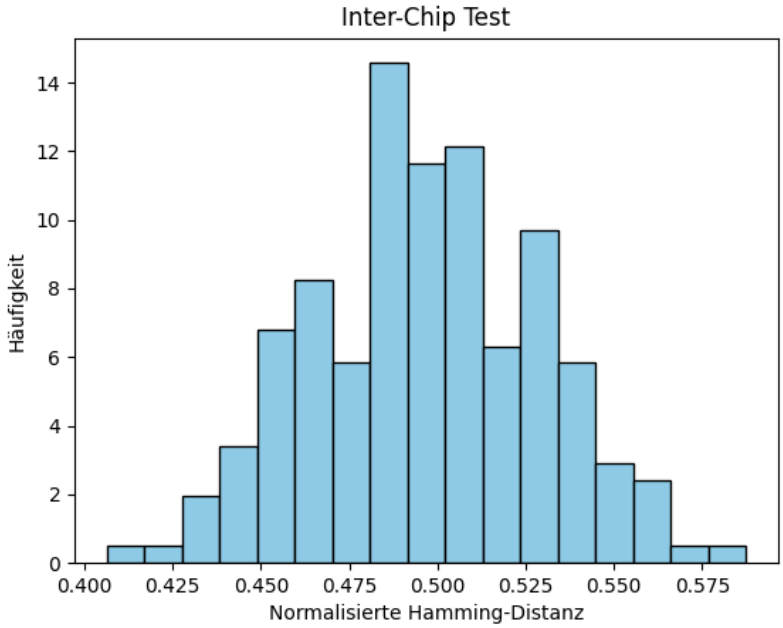

Abbildung 7.4: Histogramm des Inter-Chip Experiments der Hamming-Distanz mit 195 Testgeräten mit je 100 IDs

8 Zusammenfassung und Ausblick

8.1 Zusammenfassung

Das Ziel dieser Arbeit war die Evaluierung der Eignung von SRAM-basierten PUF mit aktivierten LPM in RIOT. Dabei sollte untersucht werden, ob die Aktivierung von LPM das Generieren von Hilfsdaten auf dem Gerät selbst ermöglicht. Die Evaluierung der durchgeführten Experimente haben die folgenden Ergebnisse erzielt.

Der Bias-Test hat gezeigt, dass mit aktiviertem LPM die SRAM-Startwerte nicht gegen Null oder Eins verzerrt sind. Der Intra-Chip Test hat mit einem Ergebnis nahe 0 % gezeigt, das wiederholte Messungen desselben SRAM PUF mit aktiviertem LPM nur geringe Variationen aufweisen. Dies deutet auf die Zuverlässigkeit eines SRAM PUF mit LPM hin. Der erste Inter-Chip Test hat mit einem Ergebnis nahe der 50 % aufgezeigt, dass unter Verwendung von LPM die Korrelation zwischen verschiedenen Geräten gering ist. Dies deutet auf die Einzigartigkeit eines SRAM PUF mit LPM hin. Der zweite Inter-Chip Test hat ergeben, dass sich die Korrelation von erzeugten IDs zwischen unterschiedlichen Geräten um den Mittelwert von 50 % verteilt. Dies lässt darauf schließen, dass die Erzeugung von Hilfsdaten auf dem Gerät selbst funktioniert. Die Verteilung um den Mittelwert deutet auf die Einzigartigkeit der erzeugten IDs zwischen unterschiedlichen IoT-Geräten hin, was sich auch in den Ergebnissen der Evaluierungen für die SRAM-Startwerte widerspiegelt. Die Erzeugung von Hilfsdaten von SRAM PUF Startwerten ist von der Qualität der Eigenschaften Einzigartigkeit und Zuverlässigkeit abhängig.

Zusammengefasst lässt sich sagen, dass mit den durchgeführten Tests festgestellt werden konnte, dass SRAM basierte PUF unter Verwendung von LPM in RIOT ähnliche Qualitätsmerkmale aufweisen, wie bei der Verwendung ohne LPM.

8.2 Ausblick

Für den industriellen Einsatz von IoT-Geräten mit SRAM-Chips und einem zweiphasigen Fuzzy-Extraktor werden Hilfsdaten benötigt, die üblicherweise auf einem externen Server berechnet werden. Die Hilfsdaten müssen anschließend auf den persistenten Speicher des IoT-Gerätes übertragen werden. Durch das Erzeugen der Hilfsdaten auf dem Gerät selbst entfällt potenziell die Notwendigkeit eines zusätzlichen Hardware-Aufbaus, der als Schnittstelle zwischen IoT-Gerät und Server dient. Je nach Anwendungsfall, zum Beispiel im Kontext von vertrauenswürdigen Firmware-Updates, kann die praktische Anwendung von SRAM PUF mit LPM erprobt und es können weitere Anwendungsfälle spezifiziert werden.

Literaturverzeichnis

[1] Adjih, Cedric ; Baccelli, Emmanuel ; Fleury, Eric ; Harter, Gaetan ; Mitton, Nathalie ; Noel, Thomas ; Pissard-Gibollet, Roger ; Saint-Marcel, Frederic ; Schreiner, Guillaume ; Vandaele, Julien ; Watteyne, Thomas: FIT IoT-LAB: A large scale open experimental IoT testbed. In: *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*, IEEE, 2015, S. 459–464

[2] Baccelli, Emmanuel ; Gündogan, Cenk ; Hahm, Oliver ; Kietzmann, Peter ; Lenders, Martine ; Petersen, Hauke ; Schleiser, Kaspar ; Schmidt, Thomas C. ; Wählisch, Matthias: RIOT: an Open Source Operating System for Low-end Embedded Devices in the IoT. In: *IEEE Internet of Things Journal* 5 (2018), December, Nr. 6, S. 4428–4440. – URL http://dx.doi.org/10.1109/JIOT.2018.2815038

[3] Cisco: *Cisco Annual Internet Report (2018–2023).* 09-03-2020. – URL https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html. – Zugriffsdatum: 2022-09-06

[4] Developer, ARM: *Cortex-M3 Technical Reference Manual.* 2010. – URL https://developer.arm.com/documentation/ddi0337/h?_ga=2.258143811.839925519.1629395464-2030874199.1629395464. – Zugriffsdatum: 2022-09-06

[5] FIT IoT-LAB: *FIT IoT-LAB.* 2020. – URL https://www.iot-lab.info/. – Zugriffsdatum: 2022-09-13

[6] FIT IoT-LAB: *IoT-LAB M3.* (o.D.). – URL https://iot-lab.github.io/docs/boards/iot-lab-m3/. – Zugriffsdatum: 2022-09-06

[7] Gassend, Blaise ; Clarke, Dwaine ; Dijk, Marten van ; Devadas, Srinivas: Silicon Physical Random Functions. In: *Proceedings of the 9th ACM Conference*

on Computer and Communications Security. New York, NY, USA : Association for Computing Machinery, 2002 (CCS '02), S. 148–160. – URL https://doi.org/10.1145/586110.586132. – ISBN 1581136129

[8] GNU Operating System: *GNU Lesser General Public License, version 2.1*. Februar 1999. – URL https://www.gnu.org/licenses/old-licenses/lgpl-2.1.en.html. – Zugriffsdatum: 2022-09-06

[9] Guajardo, Jorge ; Kumar, Sandeep S. ; Schrijen, Geert-Jan ; Tuyls, Pim: FPGA Intrinsic PUFs and Their Use for IP Protection. In: Paillier, Pascal (Hrsg.) ; Verbauwhede, Ingrid (Hrsg.): *Cryptographic Hardware and Embedded Systems - CHES 2007*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2007, S. 63–80. – ISBN 978-3-540-74735-2

[10] Hofer, M. ; Böhm, Ch. ; Bock, H.: Identifikation, Authentifizierung und Schlüsselgenerierung mittels Physical Uncloneable Functions – Übersicht und Anwendungsgebiete. In: *e & i Elektrotechnik und Informationstechnik* 127 (2010), Apr, Nr. 4, S. 72–77. – URL https://doi.org/10.1007/s00502-010-0724-3. – ISSN 1613-7620

[11] Holcomb, D. E. ; Burleson, W. P. ; Fu, K.: Power-Up SRAM State as an Identifying Fingerprint and Source of True Random Numbers. In: *IEEE Transactions on Computers* 58 (2009), Nr. 9, S. 1198–1210

[12] Hüning, Felix: *Embedded Systems für IoT*. Berlin [Heidelberg] : Springer Vieweg, 2019. – ISBN 9783662579008

[13] IoT-Analytics: *State of the IoT 2020: 12 billion IoT connections, surpassing non-IoT for the first time*. 19-11-2020. – URL https://iot-analytics.com/state-of-the-iot-2020-12-billion-iot-connections-surpassing-non-iot-for-the-first-time/. – Zugriffsdatum: 2022-09-06

[14] Katzenbeisser, Stefan ; Kocabaş, Ünal ; Rožić, Vladimir ; Sadeghi, Ahmad-Reza ; Verbauwhede, Ingrid ; Wachsmann, Christian: PUFs: Myth, Fact or Busted? A Security Evaluation of Physically Unclonable Functions (PUFs) Cast in Silicon. In: Prouff, Emmanuel (Hrsg.) ; Schaumont, Patrick (Hrsg.): *Cryptographic Hardware and Embedded Systems – CHES 2012*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2012, S. 283–301. – ISBN 978-3-642-33027-8

[15] Kietzmann, Peter ; Gündoğan, Cenk ; Schmidt, Thomas C. ; Wählisch, Matthias: A PUF Seed Generator for RIOT: Introducing Crypto-Fundamentals to the Wild. In: *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*. New York, NY, USA : Association for Computing Machinery, 2018 (MobiSys '18), S. 513. – URL https://doi.org/10.1145/3210240.3210805. – ISBN 9781450357203

[16] Kietzmann, Peter ; Schmidt, Thomas C. ; Wählisch, Matthias: A Guideline on Pseudorandom Number Generation (PRNG) in the IoT. In: *ACM Comput. Surv.* 54 (2021), jul, Nr. 6. – URL https://doi.org/10.1145/3453159. – ISSN 0360-0300

[17] Lee, J.W. ; Lim, Daihyun ; Gassend, B. ; Suh, G.E. ; Dijk, M. van ; Devadas, S.: A technique to build a secret key in integrated circuits for identification and authentication applications. In: *2004 Symposium on VLSI Circuits. Digest of Technical Papers (IEEE Cat. No.04CH37525)*, 2004, S. 176–179

[18] Leest, Vincent van der ; Preneel, Bart ; Sluis, Erik van der: Soft Decision Error Correction for Compact Memory-Based PUFs Using a Single Enrollment. In: Prouff, Emmanuel (Hrsg.) ; Schaumont, Patrick (Hrsg.): *Cryptographic Hardware and Embedded Systems – CHES 2012*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2012, S. 268–282. – ISBN 978-3-642-33027-8

[19] Pappu, Ravi ; Recht, Ben ; Taylor, Jason ; Gershenfeld, Neil A.: Physical One-Way Functions. In: *Science* 297 (2002), S. 2026 – 2030

[20] Parisot, Augusto ; Bento, Lucila M. S. ; Machado, Raphael C. S.: Testing and selecting lightweight pseudo-random number generators for IoT devices. In: *2021 IEEE International Workshop on Metrology for Industry 4.0 & IoT (MetroInd4.0&IoT)*, IEEE, 2021, S. 715–720

[21] Prada-Delgado, M. A. ; Vázquez-Reyes, A. ; Baturone, I.: Trustworthy firmware update for Internet-of-Thing Devices using physical unclonable functions. In: *2017 Global Internet of Things Summit (GIoTS)*, IEEE, 2017, S. 1–5

[22] Rayes, Ammar ; Salam, Samer: *Internet of Things From Hype to Reality - The Road to Digitization*. 2. Berlin, Heidelberg : Springer Cham, 2019. – ISBN 978-3-319-99516-8

[23] RIOT-OS: *RIOT online course*. 09-03-2022. – URL https://riot-os.github.io/riot-course/slides/03-riot-basics/#7. – Zugriffsdatum: 2022-09-28

[24] Schaller, André: *Lightweight Protocols and Applications for Memory-Based Intrinsic Physically Unclonable Functions on Commercial Off-The-Shelve Devices*. Darmstadt, Technische Universität, Dissertation, 2017. – URL http://tuprints.ulb.tu-darmstadt.de/7014/

[25] Schrijen, G. ; van der Leest, V.: Comparative analysis of SRAM memories used as PUF primitives. In: *2012 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, IEEE, March 2012, S. 1319–1324. – ISSN 1558-1101

[26] STMicroelectronics: *High-density performance line Arm ®-based 32-bit MCU with 256 to 512KB Flash, USB, CAN, 11 timers, 3 ADCs, 13 communication interfaces*. Juli 2018. – URL https://www.iot-lab.info/assets/misc/docs/iot-lab-m3/stm32f103re.pdf. – Zugriffsdatum: 2022-09-06

[27] Suh, G. E. ; Devadas, S.: Physical Unclonable Functions for Device Authentication and Secret Key Generation. In: *2007 44th ACM/IEEE Design Automation Conference*, IEEE, 2007, S. 9–14

[28] Van Herrewege, Anthony ; Leest, Vincent van der ; Schaller, André ; Katzenbeisser, Stefan ; Verbauwhede, Ingrid: Secure PRNG Seeding on Commercial Off-the-Shelf Microcontrollers. In: *Proceedings of the 3rd International Workshop on Trustworthy Embedded Devices*. New York, NY, USA : Association for Computing Machinery, 2013 (TrustED '13), S. 55–64. – URL https://doi.org/10.1145/2517300.2517306. – ISBN 9781450324861

[29] Woodhouse, David: *Memory Technology Device (MTD) Subsystem for Linux*. 25-07-2021. – URL http://www.linux-mtd.infradead.org/index.html. – Zugriffsdatum: 2022-09-24

A Anhang

A.1 Ergebnisse des Bias-Test mit erzeugten Startmustern von 355 Testgeräten

Device No.	Hamming-Weight
2	49.55 %
3	51.93 %
4	52.95 %
5	52.74 %
6	52.2 %
7	53.03 %
8	48.71 %
9	47.72 %
10	51.84 %
11	52.27 %
12	48.97 %
13	51.55 %
14	50.7 %
15	53.88 %
16	50.99 %
18	48.96 %
20	50.26 %
21	49.87 %
22	52.81 %
23	53.6 %
24	52.4 %
25	49.59 %
26	52.04 %
27	53.27 %
28	53.39 %
29	52.76 %
30	53.08 %

31	51.91 %
32	49.32 %
33	53.11 %
34	48.99 %
35	50.02 %
36	54.2 %
37	49.39 %
38	49.69 %
39	50.26 %
40	50.14 %
41	48.43 %
42	53.3 %
43	51.47 %
44	55.27 %
46	47.82 %
47	51.26 %
48	49.59 %
49	54.62 %
51	53.08 %
52	50.11 %
53	51.72 %
54	51.65 %
55	52.36 %
56	50.33 %
57	52.72 %
58	50.19 %
59	49.0 %
60	49.66 %
61	49.78 %
62	50.27 %
63	53.51 %
64	50.79 %
65	52.69 %
66	49.48 %
67	50.92 %
68	49.75 %
69	53.93 %
70	49.22 %
71	51.84 %
72	52.53 %
73	50.12 %
74	52.77 %
75	50.85 %
76	49.46 %

77	48.76 %
78	50.94 %
79	52.71 %
80	51.4 %
81	50.31 %
82	53.5 %
83	49.6 %
84	48.6 %
85	51.03 %
86	53.43 %
87	49.83 %
88	52.05 %
89	53.7 %
90	50.23 %
91	49.36 %
92	52.06 %
93	51.14 %
94	48.88 %
95	54.61 %
96	48.96 %
97	52.38 %
98	51.73 %
99	54.98 %
100	49.04 %
101	52.78 %
103	50.85 %
104	50.33 %
105	52.27 %
106	53.52 %
107	51.11 %
108	53.23 %
109	49.03 %
110	48.43 %
112	51.28 %
113	49.87 %
114	52.62 %
115	54.95 %
116	50.86 %
117	51.56 %
118	54.15 %
119	51.41 %
120	53.37 %
121	52.61 %
122	48.43 %

```
119 |       123        |     50.16 %      |
120 |       124        |     54.12 %      |
121 |       125        |     52.81 %      |
122 |       126        |     51.23 %      |
123 |       127        |     50.06 %      |
124 |       128        |     48.94 %      |
125 |       129        |     48.12 %      |
126 |       130        |     51.02 %      |
127 |       131        |     51.52 %      |
128 |       133        |     50.22 %      |
129 |       134        |     53.94 %      |
130 |       135        |     53.26 %      |
131 |       136        |     48.54 %      |
132 |       137        |     50.68 %      |
133 |       138        |     52.09 %      |
134 |       139        |     52.02 %      |
135 |       140        |     52.48 %      |
136 |       141        |     49.63 %      |
137 |       142        |     49.21 %      |
138 |       143        |     50.27 %      |
139 |       144        |     54.07 %      |
140 |       145        |     52.25 %      |
141 |       146        |     51.21 %      |
142 |       147        |     51.63 %      |
143 |       148        |      53.2 %      |
144 |       149        |     53.76 %      |
145 |       150        |     51.62 %      |
146 |       151        |     46.88 %      |
147 |       152        |     49.99 %      |
148 |       153        |     51.79 %      |
149 |       154        |     53.93 %      |
150 |       156        |     51.61 %      |
151 |       157        |     51.08 %      |
152 |       159        |     50.36 %      |
153 |       160        |      50.0 %      |
154 |       161        |     51.95 %      |
155 |       162        |     51.21 %      |
156 |       163        |     49.27 %      |
157 |       164        |     52.79 %      |
158 |       166        |     50.97 %      |
159 |       167        |     50.93 %      |
160 |       168        |     51.74 %      |
161 |       169        |      50.8 %      |
162 |       170        |     49.59 %      |
```

```
|       171        |     51.63 %      |
|       172        |      51.0 %      |
|       173        |     49.19 %      |
|       174        |     50.47 %      |
|       175        |      48.3 %      |
|       176        |     53.49 %      |
|       177        |     48.75 %      |
|       178        |     51.04 %      |
|       179        |     50.67 %      |
|       180        |     48.36 %      |
|       181        |     51.62 %      |
|       182        |     52.52 %      |
|       183        |     52.45 %      |
|       184        |     49.55 %      |
|       185        |     52.51 %      |
|       186        |     52.12 %      |
|       189        |     49.94 %      |
|       190        |      52.7 %      |
|       191        |     52.33 %      |
|       192        |     49.14 %      |
|       193        |     46.77 %      |
|       195        |     50.55 %      |
|       196        |     50.51 %      |
|       197        |     49.46 %      |
|       198        |     53.86 %      |
|       199        |     51.32 %      |
|       200        |     51.79 %      |
|       201        |     53.59 %      |
|       202        |     50.21 %      |
|       203        |     49.33 %      |
|       205        |     50.06 %      |
|       207        |     51.11 %      |
|       208        |     51.13 %      |
|       209        |     47.82 %      |
|       210        |     50.48 %      |
|       211        |     51.36 %      |
|       212        |     49.31 %      |
|       213        |     49.55 %      |
|       214        |     51.47 %      |
|       215        |     52.21 %      |
|       216        |      50.4 %      |
|       217        |     48.53 %      |
|       218        |     51.87 %      |
|       219        |     50.31 %      |
```

```
|       220        |     49.39 %      |
|       221        |     51.74 %      |
|       222        |     53.92 %      |
|       223        |     47.62 %      |
|       224        |     53.29 %      |
|       225        |     53.15 %      |
|       226        |     48.22 %      |
|       227        |      50.2 %      |
|       228        |     53.69 %      |
|       229        |     54.06 %      |
|       230        |     49.84 %      |
|       231        |     52.18 %      |
|       232        |     54.15 %      |
|       233        |     51.73 %      |
|       234        |     51.65 %      |
|       235        |     51.23 %      |
|       236        |     53.92 %      |
|       237        |     51.03 %      |
|       238        |     53.27 %      |
|       239        |     51.05 %      |
|       240        |     53.06 %      |
|       241        |     50.75 %      |
|       242        |     52.48 %      |
|       244        |     49.67 %      |
|       245        |     52.38 %      |
|       246        |      51.4 %      |
|       247        |     50.93 %      |
|       248        |      51.7 %      |
|       249        |     49.57 %      |
|       250        |     53.44 %      |
|       251        |      52.0 %      |
|       252        |     47.62 %      |
|       253        |     51.28 %      |
|       254        |     53.87 %      |
|       255        |     51.55 %      |
|       257        |     53.12 %      |
|       258        |     49.58 %      |
|       259        |     50.47 %      |
|       260        |     50.86 %      |
|       263        |     50.26 %      |
|       264        |     51.65 %      |
|       265        |     52.99 %      |
|       266        |     47.49 %      |
|       267        |     49.76 %      |
```

268	52.05 %
269	48.83 %
270	51.45 %
271	49.17 %
272	52.13 %
273	51.09 %
274	55.71 %
275	52.03 %
276	52.12 %
277	51.38 %
278	51.75 %
279	51.46 %
280	50.58 %
281	48.13 %
282	51.45 %
283	48.11 %
284	53.35 %
285	52.99 %
286	50.53 %
287	52.74 %
288	51.08 %
289	52.83 %
290	50.22 %
292	53.01 %
293	50.59 %
294	47.79 %
295	50.93 %
296	52.1 %
297	51.84 %
298	51.43 %
299	50.9 %
300	50.79 %
301	53.33 %
302	53.82 %
303	49.22 %
304	51.09 %
305	51.82 %
306	50.39 %
307	51.5 %
308	53.62 %
309	50.18 %
310	51.11 %
311	51.48 %
312	51.38 %

313	52.2 %
314	50.21 %
316	49.81 %
317	50.44 %
318	51.7 %
319	51.33 %
320	52.51 %
321	49.61 %
322	50.77 %
323	51.22 %
324	50.44 %
325	50.16 %
326	51.07 %
327	49.16 %
328	53.85 %
329	51.81 %
330	50.93 %
332	49.41 %
333	47.95 %
334	51.22 %
335	48.36 %
336	51.4 %
337	49.63 %
338	51.3 %
339	51.05 %
340	52.01 %
341	50.23 %
342	49.02 %
343	54.79 %
344	49.86 %
345	52.53 %
346	53.69 %
347	50.86 %
348	52.16 %
349	50.55 %
350	51.31 %
351	52.16 %
353	47.89 %
354	49.85 %
355	50.27 %
356	51.66 %
357	50.86 %
358	51.5 %
359	48.26 %

```
|        360         |      50.02 %       |
|        361         |      49.77 %       |
|        362         |      51.96 %       |
|        363         |      53.98 %       |
|        365         |      50.71 %       |
|        366         |      48.36 %       |
|        367         |      49.59 %       |
|        368         |       50.1 %       |
|        369         |      47.75 %       |
|        370         |      50.34 %       |
|        371         |      50.94 %       |
|        372         |      53.52 %       |
|        373         |      47.08 %       |
|        374         |      52.36 %       |
|        375         |      51.84 %       |
|        376         |      50.97 %       |
|        377         |      48.71 %       |
|        378         |      51.22 %       |
|        379         |      49.63 %       |
|        380         |      50.35 %       |
| ------------------ | ------------------ |
|        AVG         |      51.14 %       |
|        MIN         |      46.77 %       |
|        MAX         |      55.71 %       |
| ------------------ | ------------------ |
|  Devices total     |        355         |
+--------------------+--------------------+
```

Listing A.1: Datensatz mit 355 Testgeräten, jeweils 100 erzeugte Startmuster und das dazu berechnete Hamming-Gewicht.

A.2 Ergebnisse des Intra-Chip Test mit erzeugten Startmustern von 355 Testgeräten

```
+--------------------+--------------------+
|     Device No.     | Hamming-Distance   |
+--------------------+--------------------+
|         2          |      4.48 %        |
|         3          |      5.88 %        |
|         4          |      5.73 %        |
|         5          |      5.14 %        |
```

6	4.34 %
7	5.13 %
8	4.76 %
9	6.13 %
10	5.62 %
11	5.79 %
12	5.45 %
13	5.38 %
14	4.77 %
15	4.84 %
16	5.45 %
18	6.38 %
20	4.84 %
21	5.61 %
22	5.27 %
23	4.64 %
24	5.85 %
25	4.56 %
26	4.95 %
27	5.15 %
28	4.57 %
29	4.89 %
30	4.85 %
31	5.51 %
32	4.6 %
33	4.02 %
34	5.42 %
35	5.39 %
36	6.0 %
37	4.97 %
38	5.18 %
39	5.07 %
40	5.22 %
41	5.79 %
42	5.03 %
43	4.35 %
44	5.09 %
46	4.92 %
47	6.66 %
48	5.68 %
49	4.42 %
51	5.19 %
52	5.44 %
53	4.48 %

54	4.86 %
55	5.37 %
56	5.48 %
57	5.25 %
58	5.46 %
59	5.36 %
60	4.59 %
61	5.46 %
62	5.32 %
63	4.98 %
64	5.89 %
65	5.67 %
66	5.87 %
67	5.51 %
68	5.18 %
69	5.43 %
70	5.06 %
71	5.14 %
72	6.14 %
73	5.23 %
74	5.75 %
75	5.44 %
76	5.68 %
77	5.76 %
78	5.58 %
79	6.48 %
80	5.43 %
81	5.12 %
82	4.93 %
83	5.44 %
84	5.19 %
85	5.28 %
86	5.39 %
87	5.26 %
88	5.54 %
89	4.84 %
90	4.83 %
91	5.98 %
92	5.18 %
93	5.43 %
94	4.31 %
95	5.05 %
96	5.42 %
97	5.6 %

98	5.67 %	
99	5.12 %	
100	5.03 %	
101	5.67 %	
103	5.31 %	
104	4.74 %	
105	5.13 %	
106	5.64 %	
107	5.63 %	
108	6.26 %	
109	4.91 %	
110	5.19 %	
112	5.04 %	
113	6.23 %	
114	5.34 %	
115	6.59 %	
116	6.39 %	
117	5.2 %	
118	4.65 %	
119	4.8 %	
120	5.42 %	
121	4.8 %	
122	6.04 %	
123	4.98 %	
124	5.02 %	
125	4.95 %	
126	5.28 %	
127	5.38 %	
128	5.26 %	
129	5.35 %	
130	5.58 %	
131	4.93 %	
133	5.92 %	
134	5.98 %	
135	5.13 %	
136	4.74 %	
137	5.24 %	
138	4.75 %	
139	5.37 %	
140	4.91 %	
141	4.98 %	
142	5.5 %	
143	5.17 %	
144	5.86 %	

145	5.22 %
146	4.73 %
147	4.52 %
148	5.09 %
149	4.62 %
150	5.52 %
151	4.66 %
152	5.24 %
153	5.15 %
154	5.98 %
156	5.07 %
157	6.58 %
159	5.27 %
160	4.94 %
161	6.63 %
162	5.03 %
163	4.81 %
164	5.0 %
166	6.19 %
167	5.31 %
168	5.16 %
169	5.32 %
170	4.88 %
171	5.79 %
172	6.03 %
173	5.15 %
174	4.68 %
175	5.06 %
176	4.76 %
177	4.61 %
178	4.32 %
179	5.29 %
180	5.11 %
181	5.78 %
182	5.5 %
183	4.5 %
184	5.36 %
185	5.15 %
186	4.79 %
189	5.08 %
190	6.48 %
191	5.77 %
192	5.03 %
193	4.7 %

195	4.55 %
196	4.79 %
197	5.5 %
198	5.64 %
199	4.81 %
200	5.63 %
201	5.12 %
202	5.42 %
203	6.49 %
205	5.08 %
207	5.22 %
208	5.06 %
209	5.54 %
210	5.86 %
211	4.98 %
212	4.61 %
213	5.74 %
214	5.83 %
215	5.27 %
216	5.31 %
217	4.45 %
218	4.06 %
219	5.31 %
220	5.46 %
221	4.46 %
222	4.58 %
223	5.21 %
224	5.46 %
225	5.8 %
226	6.02 %
227	5.8 %
228	5.67 %
229	5.57 %
230	4.93 %
231	4.91 %
232	4.74 %
233	5.32 %
234	5.09 %
235	5.99 %
236	5.67 %
237	5.46 %
238	5.6 %
239	4.88 %
240	6.11 %

241	4.53 %
242	5.41 %
244	6.0 %
245	5.22 %
246	4.5 %
247	5.61 %
248	5.57 %
249	5.55 %
250	5.13 %
251	5.2 %
252	5.93 %
253	4.96 %
254	5.62 %
255	5.22 %
257	5.09 %
258	6.13 %
259	4.33 %
260	4.86 %
263	4.89 %
264	6.33 %
265	4.86 %
266	4.68 %
267	5.48 %
268	5.15 %
269	5.39 %
270	5.25 %
271	5.45 %
272	5.18 %
273	5.04 %
274	6.02 %
275	5.2 %
276	5.56 %
277	5.11 %
278	4.96 %
279	6.5 %
280	5.57 %
281	4.88 %
282	5.03 %
283	5.93 %
284	5.62 %
285	6.16 %
286	4.96 %
287	5.38 %
288	5.27 %

289	4.75 %	
290	5.23 %	
292	6.0 %	
293	5.16 %	
294	5.03 %	
295	5.38 %	
296	5.66 %	
297	5.78 %	
298	5.01 %	
299	5.45 %	
300	5.58 %	
301	5.21 %	
302	5.3 %	
303	5.17 %	
304	5.45 %	
305	5.47 %	
306	5.85 %	
307	5.87 %	
308	4.39 %	
309	5.75 %	
310	5.86 %	
311	4.71 %	
312	4.75 %	
313	5.37 %	
314	6.14 %	
316	4.92 %	
317	5.42 %	
318	5.17 %	
319	5.25 %	
320	5.25 %	
321	5.98 %	
322	4.75 %	
323	5.45 %	
324	4.24 %	
325	5.68 %	
326	5.55 %	
327	5.74 %	
328	5.32 %	
329	5.66 %	
330	5.56 %	
332	5.27 %	
333	5.37 %	
334	5.23 %	
335	5.87 %	

336	4.04 %
337	5.32 %
338	5.52 %
339	5.83 %
340	4.03 %
341	4.78 %
342	5.99 %
343	4.79 %
344	5.12 %
345	4.69 %
346	5.73 %
347	5.05 %
348	4.94 %
349	5.17 %
350	5.16 %
351	4.8 %
353	6.14 %
354	6.02 %
355	5.09 %
356	4.78 %
357	4.32 %
358	5.55 %
359	5.61 %
360	3.86 %
361	6.32 %
362	5.67 %
363	5.3 %
365	5.71 %
366	4.95 %
367	5.34 %
368	4.58 %
369	5.9 %
370	5.4 %
371	4.87 %
372	5.4 %
373	4.47 %
374	5.74 %
375	5.54 %
376	5.31 %
377	6.49 %
378	5.82 %
379	5.2 %
380	5.32 %

```
|       AVG         |       5.29 %       |
|       MIN         |       3.86 %       |
|       MAX         |       6.66 %       |
| ----------------- | ------------------ |
|  Devices total    |        355         |
+-------------------+--------------------+
```

Listing A.2: Datensatz mit 355 Testgeräten, jeweils 100 erzeugte Startmuster und die dazu berechnete Hamming-Distanz.

A.3 Ergebnisse des Inter-Chip Test mit erzeugten Startmustern von 355 Testgeräten

PUF-A No.	PUF-B No.	Hamming-Distance
2	3	47.35 %
2	4	47.54 %
2	5	47.92 %
2	6	48.2 %
2	7	46.21 %
2	8	48.77 %
2	9	48.2 %
2	10	48.86 %
2	11	47.54 %
2	12	48.01 %
2	13	49.91 %
2	14	49.34 %
2	15	48.86 %
2	16	48.48 %
2	18	47.92 %
2	20	47.16 %
2	21	46.78 %
2	22	46.88 %
2	23	50.38 %
2	24	48.77 %
2	25	47.82 %
2	26	49.24 %
2	27	50.47 %
2	28	48.3 %
2	29	49.43 %

2	30	47.35 %	
2	31	48.77 %	
2	32	46.5 %	
2	33	49.62 %	
2	34	46.02 %	
2	35	48.39 %	
2	36	47.06 %	
2	37	46.12 %	
2	38	47.54 %	
2	39	46.21 %	
2	40	47.54 %	
2	41	46.21 %	
2	42	48.11 %	
2	43	46.4 %	
2	44	47.44 %	
2	46	47.73 %	
2	47	47.82 %	
2	48	45.17 %	
2	49	45.45 %	
2	51	49.05 %	
2	52	49.34 %	
2	53	47.06 %	
2	54	46.97 %	
2	55	45.36 %	
2	56	47.44 %	
2	57	47.73 %	
2	58	48.77 %	
2	59	46.69 %	
2	60	49.62 %	
2	61	48.86 %	
2	62	50.66 %	
2	63	48.2 %	
2	64	47.16 %	
2	65	46.12 %	
2	66	47.73 %	
2	67	47.16 %	
2	68	48.67 %	
2	69	45.74 %	
2	70	51.23 %	
2	71	50.28 %	
2	72	48.96 %	
2	73	51.04 %	
2	74	46.31 %	
2	75	44.79 %	

2	76	47.63 %	
2	77	49.34 %	
2	78	48.86 %	
2	79	47.82 %	
2	80	47.92 %	
2	81	48.48 %	
2	82	47.25 %	
2	83	48.86 %	
2	84	46.4 %	
2	85	47.92 %	
2	86	48.3 %	
2	87	47.25 %	
2	88	48.2 %	
2	89	47.25 %	
2	90	45.64 %	
2	91	48.01 %	
2	92	46.4 %	
2	93	49.15 %	
2	94	46.02 %	
2	95	46.31 %	
2	96	50.0 %	
2	97	46.78 %	
2	98	48.01 %	
2	99	47.35 %	
2	100	46.02 %	
2	101	48.77 %	
2	103	49.15 %	
2	104	44.79 %	
2	105	49.53 %	
2	106	47.25 %	
2	107	49.24 %	
2	108	48.39 %	
2	109	45.27 %	
2	110	46.5 %	
2	112	48.11 %	
2	113	46.59 %	
2	114	51.7 %	
2	115	45.27 %	
2	116	49.72 %	
2	117	50.0 %	
2	118	49.05 %	
2	119	45.45 %	
2	120	50.19 %	
2	121	48.11 %	

```
117 |       2       |       122       |     48.01 %     |
118 |       2       |       123       |     49.91 %     |
119 |       2       |       124       |      46.4 %     |
120 |       2       |       125       |      48.2 %     |
121 |       2       |       126       |     47.54 %     |
122 |       2       |       127       |     47.25 %     |
123 |       2       |       128       |     51.52 %     |
124 |       2       |       129       |     48.39 %     |
125 |       2       |       130       |     47.44 %     |
126 |       2       |       131       |     45.74 %     |
127 |       2       |       133       |     48.67 %     |
128 |       2       |       134       |     49.62 %     |
129 |       2       |       135       |     48.01 %     |
130 |       2       |       136       |     49.15 %     |
131 |       2       |       137       |     49.15 %     |
132 |       2       |       138       |     49.15 %     |
133 |       2       |       139       |     45.93 %     |
134 |       2       |       140       |     49.62 %     |
135 |       2       |       141       |     48.48 %     |
136 |       2       |       142       |     48.01 %     |
137 |       2       |       143       |     49.72 %     |
138 |       2       |       144       |     47.73 %     |
139 |       2       |       145       |     47.54 %     |
140 |       2       |       146       |     46.21 %     |
141 |       2       |       147       |     47.54 %     |
142 |       2       |       148       |     44.98 %     |
143 |       2       |       149       |     47.06 %     |
144 |       2       |       150       |     50.57 %     |
145 |       2       |       151       |     48.96 %     |
146 |       2       |       152       |     48.11 %     |
147 |       2       |       153       |     46.21 %     |
148 |       2       |       154       |      48.3 %     |
149 |       2       |       156       |     48.11 %     |
150 |       2       |       157       |     46.78 %     |
151 |       2       |       159       |     46.21 %     |
152 |       2       |       160       |     45.27 %     |
153 |       2       |       161       |     49.91 %     |
154 |       2       |       162       |     47.06 %     |
155 |       2       |       163       |     45.55 %     |
156 |       2       |       164       |     48.11 %     |
157 |       2       |       166       |     44.32 %     |
158 |       2       |       167       |     48.96 %     |
159 |       2       |       168       |     49.43 %     |
160 |       2       |       169       |     45.08 %     |
```

```
161 |         2          |        170         |      44.32 %       |
162 |         2          |        171         |      45.55 %       |
163 |         2          |        172         |      48.96 %       |
164 |         2          |        173         |      47.44 %       |
165 |         2          |        174         |      45.64 %       |
166 |         2          |        175         |      45.36 %       |
167 |         2          |        176         |      46.69 %       |
168 |         2          |        177         |      47.25 %       |
169 |         2          |        178         |      46.59 %       |
170 |         2          |        179         |      47.25 %       |
171 |         2          |        180         |      47.82 %       |
172 |         2          |        181         |      50.57 %       |
173 |         2          |        182         |       46.5 %       |
174 |         2          |        183         |      47.35 %       |
175 |         2          |        184         |      47.25 %       |
176 |         2          |        185         |      46.88 %       |
177 |         2          |        186         |      47.82 %       |
178 |         2          |        189         |      48.11 %       |
179 |         2          |        190         |      47.16 %       |
180 |         2          |        191         |      48.39 %       |
181 |         2          |        192         |      47.63 %       |
182 |         2          |        193         |      47.92 %       |
183 |         2          |        195         |      48.01 %       |
184 |         2          |        196         |      44.41 %       |
185 |         2          |        197         |      47.92 %       |
186 |         2          |        198         |      48.77 %       |
187 |         2          |        199         |       48.3 %       |
188 |         2          |        200         |      45.74 %       |
189 |         2          |        201         |      47.54 %       |
190 |         2          |        202         |      46.12 %       |
191 |         2          |        203         |      47.73 %       |
192 |         2          |        205         |      45.74 %       |
193 |         2          |        207         |      47.25 %       |
194 |         2          |        208         |      49.34 %       |
195 |         2          |        209         |      46.31 %       |
196 |         2          |        210         |      46.69 %       |
197 |         2          |        211         |      47.82 %       |
198 |         2          |        212         |      44.79 %       |
199 |         2          |        213         |      46.59 %       |
200 |         2          |        214         |      46.97 %       |
201 |         2          |        215         |      49.91 %       |
202 |         2          |        216         |      48.58 %       |
203 |         2          |        217         |      45.55 %       |
204 |         2          |        218         |      46.59 %       |
```

2	219	47.06 %	
2	220	50.28 %	
2	221	49.05 %	
2	222	50.66 %	
2	223	51.04 %	
2	224	46.78 %	
2	225	50.0 %	
2	226	46.5 %	
2	227	46.02 %	
2	228	44.13 %	
2	229	49.24 %	
2	230	46.97 %	
2	231	47.82 %	
2	232	48.01 %	
2	233	47.73 %	
2	234	48.96 %	
2	235	48.2 %	
2	236	48.2 %	
2	237	46.97 %	
2	238	46.4 %	
2	239	46.88 %	
2	240	46.78 %	
2	241	47.44 %	
2	242	45.83 %	
2	244	46.88 %	
2	245	48.39 %	
2	246	48.01 %	
2	247	48.67 %	
2	248	50.38 %	
2	249	50.28 %	
2	250	45.64 %	
2	251	45.93 %	
2	252	47.25 %	
2	253	47.73 %	
2	254	47.25 %	
2	255	45.36 %	
2	257	46.5 %	
2	258	45.93 %	
2	259	49.05 %	
2	260	50.19 %	
2	263	46.78 %	
2	264	47.63 %	
2	265	47.63 %	
2	266	46.69 %	

```
249 |       2       |       267       |     50.57 %     |
250 |       2       |       268       |     47.73 %     |
251 |       2       |       269       |     47.54 %     |
252 |       2       |       270       |     48.01 %     |
253 |       2       |       271       |     47.16 %     |
254 |       2       |       272       |     50.19 %     |
255 |       2       |       273       |      46.5 %     |
256 |       2       |       274       |     48.96 %     |
257 |       2       |       275       |     47.16 %     |
258 |       2       |       276       |     50.66 %     |
259 |       2       |       277       |      48.2 %     |
260 |       2       |       278       |     49.62 %     |
261 |       2       |       279       |     47.73 %     |
262 |       2       |       280       |     47.54 %     |
263 |       2       |       281       |     49.62 %     |
264 |       2       |       282       |     47.25 %     |
265 |       2       |       283       |     48.77 %     |
266 |       2       |       284       |     48.86 %     |
267 |       2       |       285       |     47.54 %     |
268 |       2       |       286       |     47.54 %     |
269 |       2       |       287       |     48.11 %     |
270 |       2       |       288       |     46.12 %     |
271 |       2       |       289       |     48.58 %     |
272 |       2       |       290       |     49.53 %     |
273 |       2       |       292       |     47.63 %     |
274 |       2       |       293       |     48.39 %     |
275 |       2       |       294       |     49.34 %     |
276 |       2       |       295       |     48.48 %     |
277 |       2       |       296       |     46.69 %     |
278 |       2       |       297       |     47.44 %     |
279 |       2       |       298       |     47.54 %     |
280 |       2       |       299       |      48.3 %     |
281 |       2       |       300       |     49.05 %     |
282 |       2       |       301       |     47.54 %     |
283 |       2       |       302       |     46.78 %     |
284 |       2       |       303       |     49.72 %     |
285 |       2       |       304       |     45.17 %     |
286 |       2       |       305       |     47.35 %     |
287 |       2       |       306       |     46.59 %     |
288 |       2       |       307       |     44.89 %     |
289 |       2       |       308       |      48.3 %     |
290 |       2       |       309       |     46.88 %     |
291 |       2       |       310       |     49.72 %     |
292 |       2       |       311       |     48.39 %     |
```

2	312	47.73 %	
2	313	47.25 %	
2	314	51.61 %	
2	316	46.78 %	
2	317	49.34 %	
2	318	49.62 %	
2	319	47.92 %	
2	320	47.92 %	
2	321	46.12 %	
2	322	45.83 %	
2	323	49.53 %	
2	324	47.63 %	
2	325	48.2 %	
2	326	46.21 %	
2	327	45.83 %	
2	328	47.44 %	
2	329	48.39 %	
2	330	48.86 %	
2	332	46.69 %	
2	333	49.43 %	
2	334	51.52 %	
2	335	48.86 %	
2	336	46.88 %	
2	337	47.92 %	
2	338	49.05 %	
2	339	47.44 %	
2	340	47.63 %	
2	341	46.31 %	
2	342	46.12 %	
2	343	45.17 %	
2	344	47.06 %	
2	345	48.11 %	
2	346	45.74 %	
2	347	47.54 %	
2	348	46.5 %	
2	349	47.54 %	
2	350	48.01 %	
2	351	45.64 %	
2	353	47.16 %	
2	354	48.86 %	
2	355	49.91 %	
2	356	47.82 %	
2	357	46.21 %	
2	358	48.01 %	

```
|         2          |        359         |      47.82 %       |
|         2          |        360         |      46.78 %       |
|         2          |        361         |      50.19 %       |
|         2          |        362         |      49.81 %       |
|         2          |        363         |      45.08 %       |
|         2          |        365         |      45.27 %       |
|         2          |        366         |      47.54 %       |
|         2          |        367         |      47.92 %       |
|         2          |        368         |      48.01 %       |
|         2          |        369         |      50.19 %       |
|         2          |        370         |      49.05 %       |
|         2          |        371         |      44.51 %       |
|         2          |        372         |      47.54 %       |
|         2          |        373         |       46.4 %       |
|         2          |        374         |      45.55 %       |
|         2          |        375         |      46.21 %       |
|         2          |        376         |      50.19 %       |
|         2          |        377         |      46.31 %       |
|         2          |        378         |       46.5 %       |
|         2          |        379         |      49.53 %       |
|         2          |        380         |      51.52 %       |
| ------------------ | ------------------ | ------------------ |
|         -          |        AVG         |      47.75 %       |
|         -          |        MIN         |      44.13 %       |
|         -          |        MAX         |       51.7 %       |
| ------------------ | ------------------ | ------------------ |
|         -          |  Devices total     |        355         |
+--------------------+--------------------+--------------------+
```

Listing A.3: Datensatz mit 355 Testgeräten, jeweils 100 erzeugte Startmuster und der Hamming-Distanz zwischen einem PUF-A und den anderen PUFs.

A.4 Ergebnisse des Inter-Chip Test mit erzeugten IDs von 194 Testgeräten

```
+--------------------+--------------------+--------------------+
|   Device-A No.     |   Device-N No.     | Hamming-Distance   |
+--------------------+--------------------+--------------------+
|         1          |         2          |       52.5 %       |
|         1          |         3          |      44.38 %       |
|         1          |         4          |      51.88 %       |
```

1	11	48.75 %	
1	12	48.75 %	
1	13	49.38 %	
1	14	51.25 %	
1	15	50.0 %	
1	17	46.25 %	
1	18	44.38 %	
1	19	50.0 %	
1	20	43.12 %	
1	21	53.75 %	
1	22	52.5 %	
1	23	49.38 %	
1	24	52.5 %	
1	25	43.12 %	
1	26	45.0 %	
1	27	56.25 %	
1	28	51.25 %	
1	29	46.88 %	
1	30	47.5 %	
1	31	50.62 %	
1	32	44.38 %	
1	33	51.88 %	
1	34	48.12 %	
1	35	45.62 %	
1	36	48.75 %	
1	37	56.25 %	
1	38	45.62 %	
1	39	45.62 %	
1	40	48.75 %	
1	41	48.75 %	
1	42	48.75 %	
1	43	50.0 %	
1	44	48.75 %	
1	45	50.62 %	
1	46	49.38 %	
1	47	49.38 %	
1	48	51.25 %	
1	49	48.75 %	
1	50	51.25 %	
1	51	48.75 %	
1	100	52.5 %	
1	101	46.25 %	
1	102	48.75 %	
1	103	52.5 %	

1	104	51.88 %	
1	105	51.88 %	
1	106	43.12 %	
1	107	51.25 %	
1	108	50.0 %	
1	109	55.0 %	
1	110	46.88 %	
1	111	50.0 %	
1	112	50.0 %	
1	114	46.25 %	
1	115	46.88 %	
1	116	45.0 %	
1	117	48.75 %	
1	118	52.5 %	
1	120	46.25 %	
1	121	52.5 %	
1	122	50.0 %	
1	123	47.5 %	
1	124	47.5 %	
1	125	45.62 %	
1	126	50.0 %	
1	128	51.88 %	
1	129	43.12 %	
1	130	48.75 %	
1	131	46.25 %	
1	132	51.25 %	
1	134	46.25 %	
1	135	54.38 %	
1	136	55.0 %	
1	137	51.25 %	
1	138	51.88 %	
1	139	47.5 %	
1	140	50.62 %	
1	141	41.88 %	
1	142	51.25 %	
1	143	51.88 %	
1	144	46.25 %	
1	145	54.38 %	
1	146	53.12 %	
1	147	48.75 %	
1	148	51.25 %	
1	149	50.0 %	
1	150	48.75 %	
1	151	56.25 %	

1	152	50.62 %	
1	153	51.88 %	
1	154	49.38 %	
1	155	47.5 %	
1	156	55.0 %	
1	157	53.75 %	
1	158	51.88 %	
1	159	54.38 %	
1	160	44.38 %	
1	161	47.5 %	
1	162	48.75 %	
1	163	56.25 %	
1	164	48.75 %	
1	165	50.62 %	
1	166	46.88 %	
1	167	48.12 %	
1	168	50.0 %	
1	169	48.75 %	
1	170	53.12 %	
1	171	47.5 %	
1	172	48.12 %	
1	173	53.12 %	
1	174	52.5 %	
1	175	45.62 %	
1	176	50.0 %	
1	177	48.75 %	
1	178	46.88 %	
1	179	53.75 %	
1	180	55.0 %	
1	181	56.25 %	
1	182	48.75 %	
1	183	50.62 %	
1	184	51.25 %	
1	185	50.62 %	
1	186	45.0 %	
1	187	53.75 %	
1	188	46.88 %	
1	189	53.12 %	
1	190	46.25 %	
1	191	45.62 %	
1	192	50.0 %	
1	193	45.62 %	
1	194	53.12 %	
1	195	50.62 %	

```
|        1        |       196       |     51.88 %     |
|        1        |       197       |      50.0 %     |
|        1        |       198       |      45.0 %     |
|        1        |       199       |     53.75 %     |
|        1        |       200       |     50.62 %     |
|        1        |       201       |      52.5 %     |
|        1        |       202       |     51.88 %     |
|        1        |       203       |     46.25 %     |
|        1        |       204       |      45.0 %     |
|        1        |       205       |     40.62 %     |
|        1        |       206       |     51.25 %     |
|        1        |       207       |      50.0 %     |
|        1        |       208       |     58.75 %     |
|        1        |       209       |      52.5 %     |
|        1        |       210       |     44.38 %     |
|        1        |       211       |     49.38 %     |
|        1        |       212       |     49.38 %     |
|        1        |       213       |      52.5 %     |
|        1        |       214       |     45.62 %     |
|        1        |       215       |     53.75 %     |
|        1        |       216       |     51.25 %     |
|        1        |       217       |     53.75 %     |
|        1        |       218       |     54.38 %     |
|        1        |       219       |     51.88 %     |
|        1        |       220       |      47.5 %     |
|        1        |       221       |      47.5 %     |
|        1        |       222       |     48.75 %     |
|        1        |       223       |     48.75 %     |
|        1        |       224       |     50.62 %     |
|        1        |       225       |     48.75 %     |
|        1        |       226       |     53.12 %     |
|        1        |       227       |     51.25 %     |
|        1        |       229       |      47.5 %     |
|        1        |       230       |     49.38 %     |
|        1        |       231       |     48.12 %     |
|        1        |       232       |     56.88 %     |
|        1        |       233       |     48.75 %     |
|        1        |       234       |     50.62 %     |
|        1        |       235       |      50.0 %     |
|        1        |       236       |     48.12 %     |
|        1        |       237       |     49.38 %     |
|        1        |       238       |     46.25 %     |
|        1        |       240       |     44.38 %     |
|        1        |       241       |      55.0 %     |
```

```
|         1          |        242         |       47.5 %       |
|         1          |        244         |      48.12 %       |
|         1          |        245         |      44.38 %       |
|         1          |        246         |      53.12 %       |
|         1          |        247         |       45.0 %       |
|         1          |        248         |       52.5 %       |
|         1          |        249         |      51.88 %       |
|         1          |        250         |       55.0 %       |
|         1          |        251         |      51.25 %       |
|         1          |        252         |      54.38 %       |
|         1          |        253         |      46.25 %       |
|         1          |        254         |       47.5 %       |
|         1          |        255         |      48.75 %       |
|         1          |        256         |      53.12 %       |
| ------------------ | ------------------ | ------------------ |
|         -          |        AVG         |       49.7 %       |
|         -          |        MIN         |      40.62 %       |
|         -          |        MAX         |      58.75 %       |
| ------------------ | ------------------ | ------------------ |
|         -          |  Devices total     |        194         |
+--------------------+--------------------+--------------------+
```

Listing A.4: Datensatz mit 194 Testgeräten, jeweils 100 erzeugte IDs und der Hamming-Distanz zwischen einem Testgerät A und den anderen Testgeräten.

Erklärung zur selbstständigen Bearbeitung

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

Ort	Datum	Unterschrift im Original