**PUF BASED AUTHENTICATION FOR**

**RESOURCE CONSTRAINED IOT DEVICES**

**J Component Project Report for the course**

**ECE4003 – EMBEDDED SYSTEM DESIGN**

*by*

**S. CHARAN (19BEC1114)**

**VARSHA JAYAPRAKASH (19BEC1233)**

**C. POOJAA (19BEC1432)**

*Submitted to*

**Dr. P. Vijayakumar**

****

SCHOOL OF ELECTRONICS ENGINEERING

VELLORE INSTITUTE OF TECHNOLOGY

CHENNAI - 600127

*December 2021*

***Bonafede Certificate***

This is to certify that the Project work titled “PUF Based Authentication for Resource Constrained IoT Devices” is being submitted by ***S. Charan* (19BEC1114)*, Varsha Jayaprakash* (19BEC1233)** and ***C. Poojaa* (19BEC1432)** for the course **Embedded System Design**, is a record of bonafide work done under my guidance. The contents of this project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University

**Dr. P. Vijayakumar**

**Associate Professor**

**School of Electronics Engineering (SENSE)**

**VIT University, Chennai**

**ACKNOWLEDGEMENT**

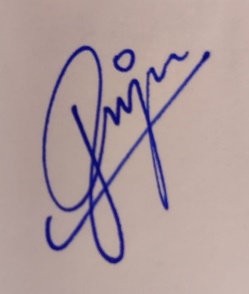
We wish to express our sincere thanks and deep sense of gratitude to our project guide, **Dr. P. Vijaykumar**, Associate Professor, School of Electronics Engineering, for his consistent encouragement and valuable guidance offered to us in a pleasant manner throughout the course of the project work.

We are extremely grateful to **Dr. A. Sivasubramanian**, Dean of School of Electronics Engineering, VIT Chennai, for extending the facilities of the school towards our project and for his unstinting support

We express our thanks to our Head of the Department **Dr Vetrivelan. P** for his support throughout the course of this project.

We also take this opportunity to thank all the faculty of the school for their support and their wisdom imparted to us throughout the course.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.

****

****

**CHARAN S VARSHA JAYAPRAKASH C POOJAA**

**ABSTRACT**

IoT based devices have a potential to be limitless in the near future, and with increased usage of these devices, a number of security issues arise. Standard encryption techniques are not suitable for these devices as they are highly resource constrained in terms of memory, speed and power consumption. Physical Unclonable Functions (PUFs) on a device represent a low-cost primitive exploiting the unique random patterns in the device and have been already applied in a multitude of applications. Once the key material is extracted, secure key generation and key agreement can be implemented on the device in order to avoid an attacker from taking over the identity of a tampered device. PUF based authentication system is developed using XOR Arbiter PUF and is implemented using MATLAB and the uniqueness and uniformity of PUF is evaluated.

***Keywords:*** *IoT, security, resource constraints, XOR Arbiter PUF, authentication, uniformity, uniqueness.*

**Table of Contents**

|  |  |  |  |
| --- | --- | --- | --- |
| **Chapter No.** | **Title** | | **Page No.** |
|  |  | |  |
|  | **Acknowledgement** | | **iii** |
|  | **Abstract** | | **iv** |
|  | **List of Figures** | | **vi** |
| **1** | **Introduction** | | **07** |
|  | 1.1 | Objectives | **08** |
|  | 1.2 | Scope | **08** |
|  | 1.3 | Application | **08** |
| **2** | **Literature Review** | | **09** |
| **3** | **Design/ Implementation** | | **11** |
|  | 3.1 | Design Approach | **11** |
|  | 3.2 | Proposed Model | **15** |
|  | 3.3 | Software Requirement | **15** |
| **4** | **Result and Analysis** | | **16** |
|  | 4.1 | MATLAB Implementation | **16** |
| 4.2 | MATLAB Output Screenshots | **17** |
| 4.3 | Security Attacks | **18** |
| **5** | **Conclusion and Future work** | | **20** |
| **6** | **Appendix** | | **21** |
| **7** | **References** | | **26** |
| **8** | **Bio Data** | | **28** |

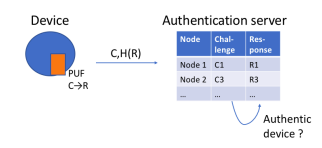
**List of Figures**

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Title** | **Page No.** |
|  |  |  |
| 1.a | PUF Procedure | 02 |
| 3.1.a | Arbiter PUF Diagram | 06 |
| 3.1.b | Ring Oscillator PUF | 07 |
| 3.1.c | SRAM PUF Diagram | 07 |
| 3.1.d | Arbiter PUF Block Diagram | 08 |
| 3.1.e | n-Order XOR Arbiter PUF | 08 |
| 3.2.a | Block Diagram | 09 |
| 4.2.a | MATLAB Output-1 | 19 |
| 4.2.b | MATLAB Output-2 | 20 |
| 4.c.c | CRP database | 20 |

1. **INTRODUCTION**

The Internet of Things (IoT) is one of the emerging technologies in today’s world to connect billions of electronic devices and provide the data security to these electronic devices while transmission is a big challenging task. Therefore, it has become urgent to secure data from attackers to preserve its integrity, confidentiality, protection, privacy and procedures required for handling it. They are exposed to various attacks like linear and differential attacks, side channel attacks, modelling and machine learning attacks etc. Most of the IoT devices electronic devices are smaller and consume less power. The conventional security algorithms are complex with its computations and not suitable for IoT environments. Lightweight cryptographic and authentication algorithms provide confidentiality in high-speed environments such as big data and cloud computing, as well as lightweight environments such as IoT devices and mobile devices.

Physical Unclonable Functions (PUFs) can be seen as the hardware equivalent of a one-way function. A silicon PUF is a physical entity that is embodied in a physical structure. Although it is easy to fabricate, it is practically infeasible to clone, even if the exact manufacturing process is produced again. Instead of using a private key that is linked to the device identity, the authentication of PUFs is based on the usage of so-called challenge–response pairs. The challenge (an electrical stimulus), is applied to the physical structure in order to react, which is called the response. This is carried out in an unpredictable manner due to the complex interaction of the stimulus with the physical micro-structure of the device, which is dependent on physical factors introduced during the manufacturing process. PUFs have relatively low hardware overhead which is helpful in the IoT context. As shown in fig 1. PUFs on devices have already been applied for device identification and authentication, binding hardware to software platforms, secure key storage, keyless secure communication, etc. It is observed that PUFs promise to achieve perfectly secure authentication without providing any cryptographic assets on the device. This also makes them the most preferred ones for resource-scarce IoT devices.



***Fig 1.a***: PUF based authentication procedure.

1. **Objectives**

With the increase in the number of IoT enabled products in the market the demand for it is also increasing. This has eventually made the physical devices used in IoT or also known as the things vulnerable to various security issues. The resource constraints associated with these devices prevent it from using heavy encryption techniques to protect the data. XOR Arbiter PUF is one of the most recommended PUF for more security and reliability. An XOR Arbiter PUF is implemented using MATLAB tools and the software efficiency is analysed. The performance is evaluated based on uniqueness and uniformity.

1. **Scope**

PUF architecture playing a major role in IoT application because of its main advantages of robustness against possible attack and easy implementation, it has varied scopes for advancement and upgradation. These technologies can be used for IC development and for authentication systems in various applications. The main advantage of XOR PUFs is the secret key generation. These features can be embedded and implemented on the FPGA boards or on the RFID tags for secure access control and for authentication purposes.

1. **Application**

An FPGA or a Field Programmable Gate Array is a hardware circuit that can be programmed to carry out one or more logical operations. It is an IC and due to its small structure (Board Space) and broad features, it is used widely in a lot of industries namely, aerospace and defence, medical electronics, digital television, cybersecurity systems and wireless communications. The main advantage of an FPGA is the ability to easily change its functionality after a product has been designed. Also, in PUF based authentication even when the key material is extracted, an attacker cannot take over the identity of the tampered device. XOR Arbiter PUFs can be used as secure key card with RFID chips for healthcare, transportation & logistics applications. These can also be used for RSA, ECC and AES algorithms for increasing the complexity. The Key zeroization is also one such unique application of the XOR arbiter PUF.

1. **RELATED WORKS**

Brisbane Ovilla-Martínez Cuauhtémoc, Mancillas-López Alberto F. Martínez-Herrera and José A. Bernal-Gutiérrez [1] et al have presented the detailed analysis of costs and performance in hardware implementations over five techniques namely COMET, ESTATE-AES /Gift, LOCUS, LOTUS, and Oribatida. The results were observed in Xilinx Artix-7 xc7a12tcsg325-3. It was indicated that it is feasible to achieve the reduction of each solution below 2000 LUTs and 2000 slices where some of them are below 850 LUTs and 600 FF when they’re included in LWC Crypto Core. Muhammad Usman, Irfan Ahmed, M. Imran Aslam, Shujaat Khan and Usman Ali Shah [2] et al discusses the lightweight encryption algorithm known as Secure IoT (SIT). The 64-bit block cipher requires a 64-bit key to encrypt the data. The architecture used is a mixture of FEISTEL and a uniform substitution-permutation network. Simulation result provided substantial security in five encryption rounds. The hardware implementation was done on a low cost 8-bit micro-controller and the results of memory utilization and encryption/decryption execution cycles are compared with benchmark encryption algorithms.

Nagham Sami, Abdelrahman Sobeih Hussein, Mohaned Khaled, Ahmed N. El-Zeiny, Mahetab Osama, Heba Yassin, Ali Abdelbaky, Omar Mahmoud, Ahmed Shawky and Hassan Mostafa [3] discussed a comparative study of eight authenticated encryption and decryption algorithms namely, ACORN, ASCON, CLOC, JOLTIK, MORUS, PRIMATEs, SCREAM and SILC, is presented. The FPGA and ASIC implementations of the eight algorithms presented are synthesized, placed and routed. Power, area, latency and throughput are measured for each of the algorithms. Finally, the results are analyzed and the best suited algorithm for IoT application is selected.Bharathi R, Parvatham [4] discusses the hardware architecture of the new Lightweight encryption algorithm (LEA) for the secured Internet of things (SIoT). This suggested model includes Encryption, decryption along with key generation process. The NLEA-SIoT is a hybrid combination of the FEISTEL and Substitution- permutation Network. The encryption/decryption architecture is a combination of Logical operations, substitution transformations, and this is designed for 64-bit data input with a 64-bit key input. The key generation process is designed with the help of KHAZAD block cipher algorithm. Here, the author has designed on the Xilinx platform and implemented on Artix-7 FPGA.

Carlos Andres Lara-Nino Arturo Diaz-Perez and Miguel Morales-Sandoval [5] explains the hardware implementations of PRESENT Technique, which is also known as the standardized lightweight cipher that is used to overcome the security issues in extremely constrained environments. The CIPHER model was reviewed and two novel designs were presented. Using the same implementation conditions, the two new proposals and three state-of-the-art designs are evaluated and compared, with respect to area, performance, energy, and efficiency. Abiy Tadesse Adebe [6] proposed an encryption and key distribution methods are implemented using reconfigurable computing which is mainly focused on targeting different optimization goals related to specific requirements of end-devices, Fog nodes, and the Cloud. The sensitive data are protected at the transit level, to ensure end-to-end addressing in terms of security requirements in IoT. Finally, Small FPGA resource utilization with high throughput results is achieved in relation to constrained IoT devices with high-speed optimizations.

Elif Bilge Kavun and Tolga Yalcin [7] discuss two different ways of implementing the FPGA lightweight cipher PRESENT model. The main design strategy for both designs is the usage of existing RAM blocks in FPGA for the storage of internal states, which eventually reduces the slice count. In the first design, S-boxes are realized within the slices, and in the second design it is integrated into the same RAM block which is used for state storage. Both applications are compared and low-cost FPGA devices are reviewed. Rajdeep Chakraborty, Jyotsna Kumar Mandal [8] discuss the advanced lightweight cryptography which is the BLOCK cipher method. The architecture and the feasibility of the system to be implemented for IoT security is assessed. The author specifies one round block cipher consisting of a Rotational Conical Cipher (RCC) and a Substitution Permutation Network (SPN). Finally, encryption of multiple blocks is done using the Cipher Block Chaining (CBC) mode. The decryption is done in a similar fashion. The proposed block cipher gets 128 bits of plaintext and this is passed through an RCC block and then finally to a Substitution Permutation Network (SPN) which gives the final ciphertext of 128 bits. The multiple blocks are encrypted with CBC mode while keeping the above structure intact. [8].

Braeken [9] discusses the key agreement scheme of a PUF based protocol, for security implementation in the IoT domain. An alternative scheme is also proposed, which is able to solve the current world issues and can provide in addition a more efficient key agreement and subsequently a communication phase between two IoT devices connected to the same authentication server. The scheme also offers identity-based authentication and repudiation, when only using elliptic curve multiplications and additions, instead of the compute intensive pairing operations. Kimmo Järvinen, Andrea Mieleeza, Reza Azarderakhsh, Patrick Longa presents a fast and a compact implementation of the Four Q on FPGAs, and the detailed demonstration of the high efficiency in new elliptic curve on reconfigurable hardware is shown clearly. The single-core and multi-core implementations can be used to compute the encryption at a rate of 6389 and 64730 scalar multiplications per second. The Xilinx Zynq-7020 FPGA, with a represent factor of 2.5 and 2 speedups are compared in correspondence with the speed of implementation on the same device. All the presented implementations were seen to exhibit regular constant-time execution protecting the timing and simple side-channel attacks. [10]

1. **DESIGN / IMPLEMENTATION**
   1. **Design Approach**

A PUF (Physical Unclonable Functions) is a physical entity which utilises the naturally occurred variations during a device manufacturing to provide a unique identity for device that makes it unclonable and unique. These architectures are chosen for the soul purpose of delivering high security for resource-constraint IoT devices. PUFs are generally implemented on any IC. The FPGA or the RFID Tags are the most common areas of implementation as they are the ones that are used in applications with high security requirements.

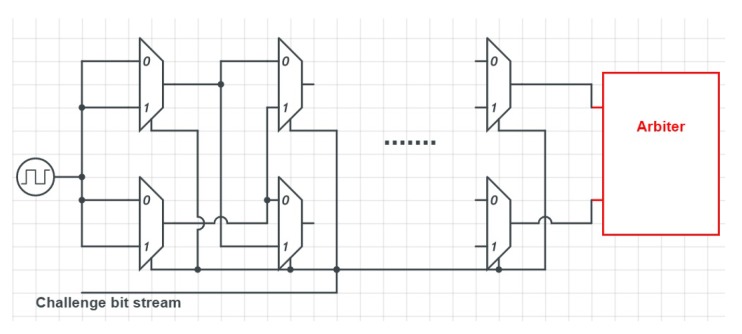
Cryptography not only sets on with encryption and decryption methods. There are a lot of security attacks that play a major role in cryptography. Information security can be analysed under the four main concepts.

1. **Confidentiality:** The main goal of this method is to ensure that the information is not provided to unauthorized access. This is archived by using encryption methods to authorize the entities with cryptographic keys.
2. **Integrity:** The absence of an information corruption for the whole life-cycle is referred to in this method. Data can be manipulated or corrupted both when stored locally or during transmission. A checksum method is used to overcome this constraint.
3. **Authenticity:** This is the assurance that the identity of a subject is the identity claimed from the source. This highly relies on integrity. Physical Unclonable Functions (PUFs) in particular is used for authentication purposes involving security concerns.
4. **Availability:** This refers to the ability to access information if needed anytime. Systems that store and process information and transmit the same have to function correctly. Apart from random faults, common security attacks are the denial-of-service attacks.

Classical authentication methods mostly rely on cryptographic algorithms which handles the secret keys stored in non-volatile memory. The main disadvantage of these methods are the unavailability of the cryptographic secret key and the expensive cryptographic algorithms. To overcome these methods, PUFs are used extensively for authentication and cryptographic secret key generation. In PUFs, the secret keys are derived from the physical characteristics of the IC instead of storing them in the memory. There are different architectures proposed for PUF usage in IoT applications. The detailed analysis of few algorithms is stated below.

**ARBITER PUF**:

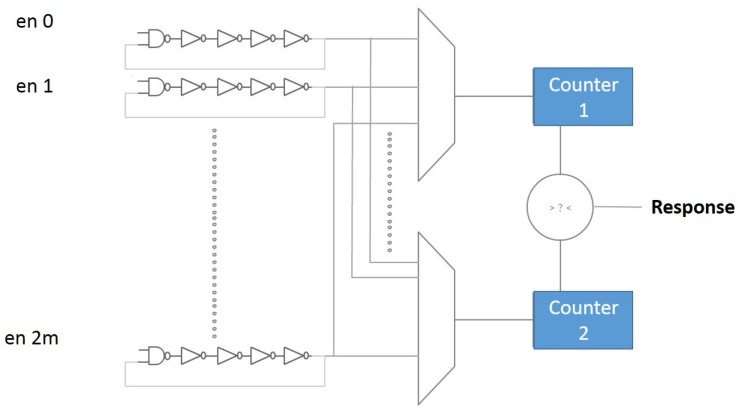
An arbitrary PUF generates a result of ‘0’ or ‘1’ by comparing two delay paths of equal lengths. These are strong PUFs which partly have resilience against machine earning attack. The challenge is used to determine the two paths that are used dynamically by using individual bits of the challenge as input into a series of interconnected multiplexers. Depending on the input, each multiplexer decides the next multiplexer to switch its output. This generates numerous combinations of possible paths. However, the delay paths here must be identical. Fig 3.1.a shows the circuit diagram of Arbiter PUF which contains multiple 2:1 mux as switching elements to produce delay and a D-Flip flop which acts as a Arbiter to decide the output response.



***Fig 3.1.a***: Arbiter PUF Diagram.

**RING OSCILLATOR PUF**:

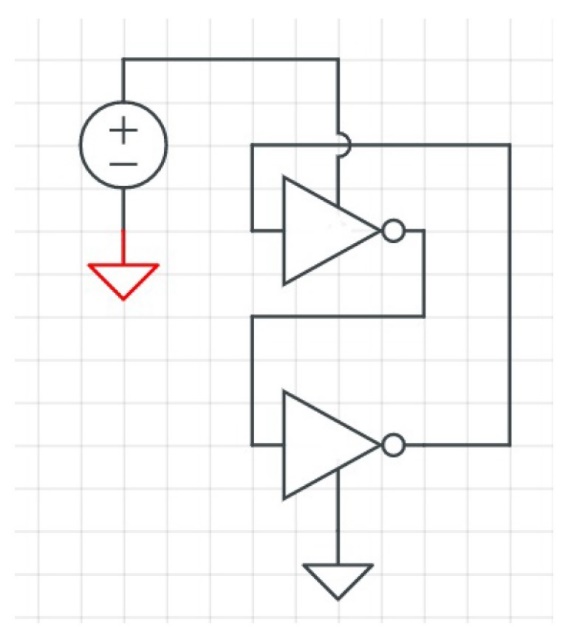
Ring oscillator PUFs contain several identical ring oscillators, which oscillate with their own frequency. As shown in fig. 3.1.b, by comparing the two Ring Oscillator frequencies, either “0” or “1” will be generated. Each incoming challenge determines the pair of ROs that is used. RO PUFs are strong PUFs, and the main reason behind using them for FPGA-based IoTs is their easy implementation on FPGA, but at the same time, they are sensitive to environmental conditions, which is a major drawback.



***Fig 3.1.b***: Ring Oscillator PUF Diagram.

**SRAM**:

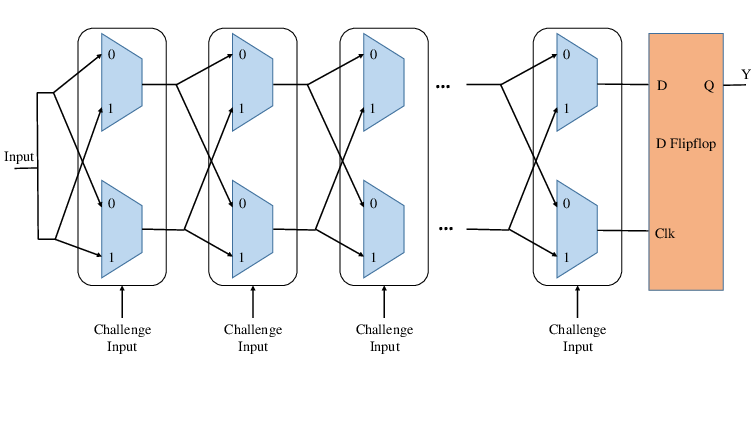
SRAM PUFs are known for their reliability and statistical properties. This architecture is based on the power-up state of SRAM blocks as shown in fig.3.1. c. When an SRAM PUF powers up, the initial value of each single SRAM cell can be either “0” or “1”. The challenge can be used as the address of an SRAM cell, and the initial value of it can be used as the SRAM PUF response. It is a weak PUF with a limited number of CRPs due to which it needs obfuscated interfaces, which cost extra security layers.



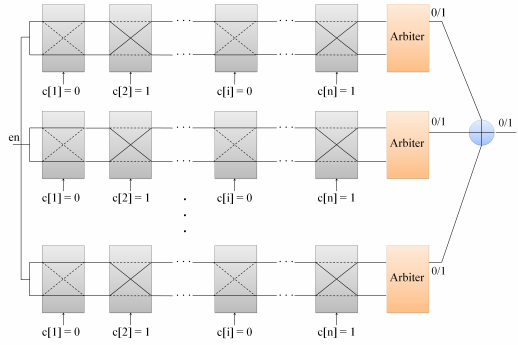
***Fig 3.1.c***: SRAM PUF Diagram.

**XOR ARBITER PUF:**

An XOR Arbiter PUFs is built from multiple Arbiter PUFs. In this type, Each Arbiter PUFs takes input signal and n-bit challenge to produce single output which is called as response. The input signal decides its path according to the challenge input as it allows input signal to decide the path in each individual multiplexers for every bit of challenge bit. Different responses from every Arbiter PUFs (shown in fig. 3.1.d) are together XORed to produce a single and final response. The block diagram of N-bit XOR Arbiter PUF is given in fig. 3.1.e. It consists of N number of challenge bits given as input to two stages of Arbiter PUF. The output contains a N bit response.



***Fig 3.1.d***: Arbiter PUF block diagram.

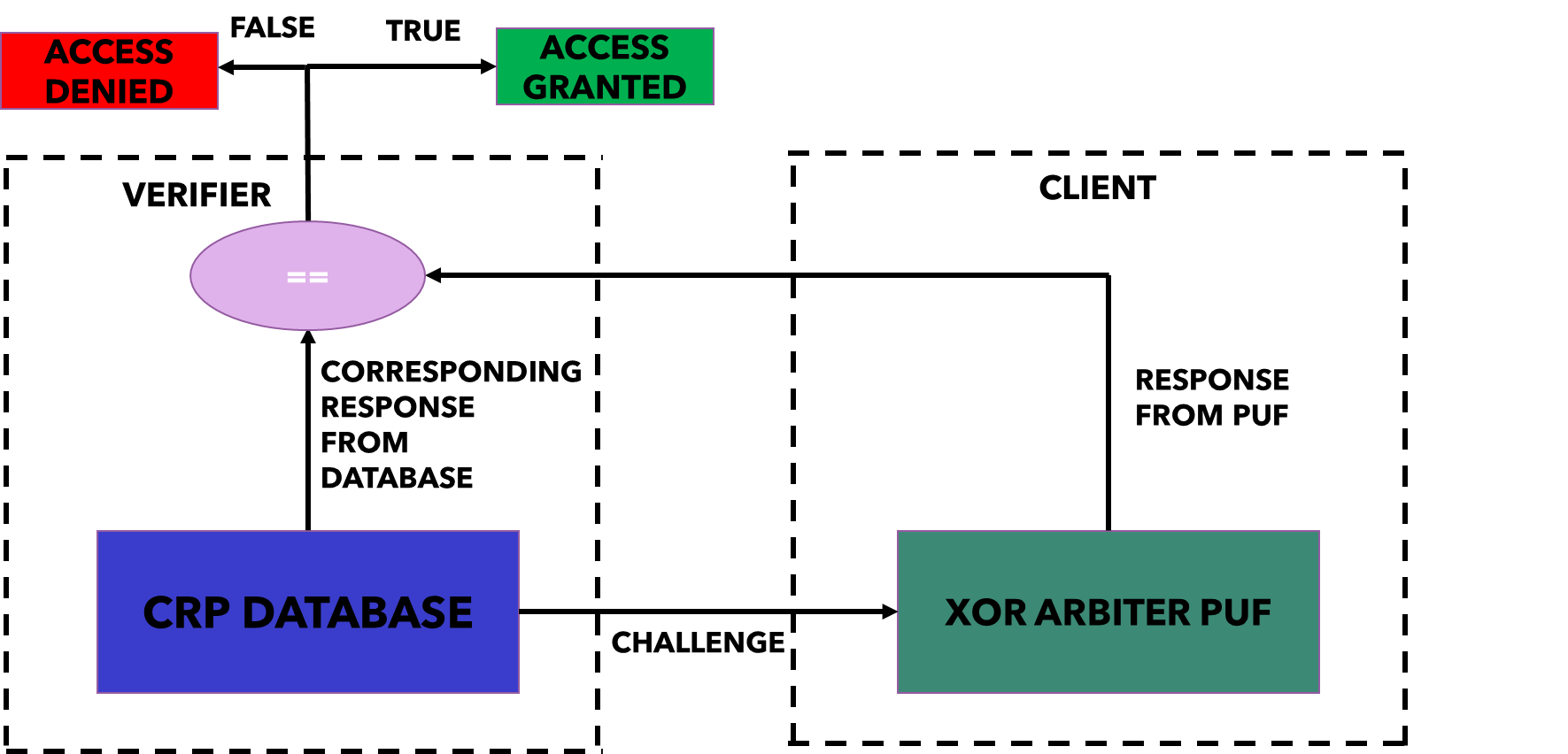


***Fig 3.1.e***: n-order XOR Arbiter PUFs Diagram.

* 1. **Proposed Model**

IoT devices are various prone to attacks. Common attacks are man-in-the-middle, Dos, Eavesdropping etc. Lightweight authentication and security schemes are required to protect IoT devices. Development of PUF based authentication technique is proposed. Software analyses based on the CRP and uniqueness and uniformity is obtained. 128-bit XOR Arbiter PUF based authentication is developed and the challenge response pairs are obtained in the client side. The challenge response pairs are stored in a database. A randomly generated challenge and response is used to compare with CRP database at verifier side. On the basis of comparison access is granted or denied. If the ID of the device doesn’t match then access is denied. Further, the uniqueness and uniformity of generated CRP database is evaluated. The block diagram of the proposed solution is given in fig. 3.2.a.

Uniqueness is a measure of the independence of PUF responses to the same challenge. The PUF uniqueness is the ratio of HD to the total response bit number. The most important metric for estimating PUF performance is the Hamming distance (HD). The intra-HD represents the uniqueness between different chips. It is the distance between two evaluations on one single PUF instantiation is the distance between the two responses, resulting from applying this challenge twice to one PUF. Uniformity is defined as the distribution of data of 1’s and 0’s in the response bits. It is calculated as a percentage of 100. It tells the percentage of 1’s and 0’s present in the response data. A uniformity of 50% makes the binary string unique and makes it difficult to trace.

****

***Fig 3.2.a***: Block Diagram.

* 1. **Software Requirements**

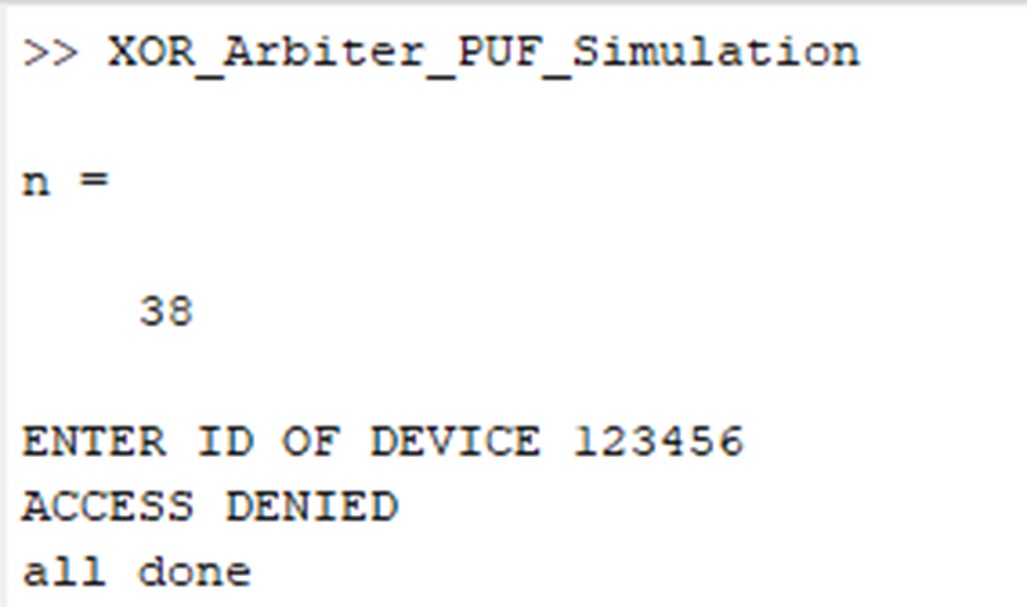
MATLAB is utilized for the implementation of PUF Based Authentication for resource constrained IoT Devices because of its varied applications and toolbox specifications. MATLAB also has various built-in functions for technical computing, graphics, and animations. This high-performance software is thus considered for the analysis of the project

1. **RESULTS AND ANALYSIS** 
   1. **MATLAB Implementation**

The required function to develop the XOR Arbiter PUF is implemented in MATLAB. The challenge is given as a random 128-bit number and random responses are obtained for it. The delay is calculated for each stage. Finally, the obtained challenge and response pairs are stored in a database. It is further extended to an authentication system where the device is given a particular ID. The user is asked to enter an ID to proceed further, If the ID is invalid then authentication is denied. Otherwise, the obtained response is compared with the CRP database to check for equality. If the responses match, then access is allowed. The obtained responses are further studied to calculate their uniqueness and uniformity.

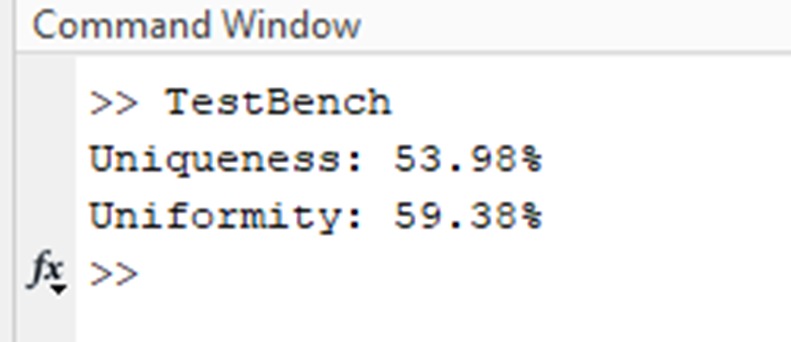
* 1. **MATLAB Output Screenshots**

The result obtained for authentication scheme is given in fig. 4.2.a. The user is expected to enter the device ID before the authentication process takes place. A random challenge is generated from the challenge response database and is given as input to the XOR Arbiter PUF to obtain the response. This response is further compared with the response stored in the database to allow or deny access.



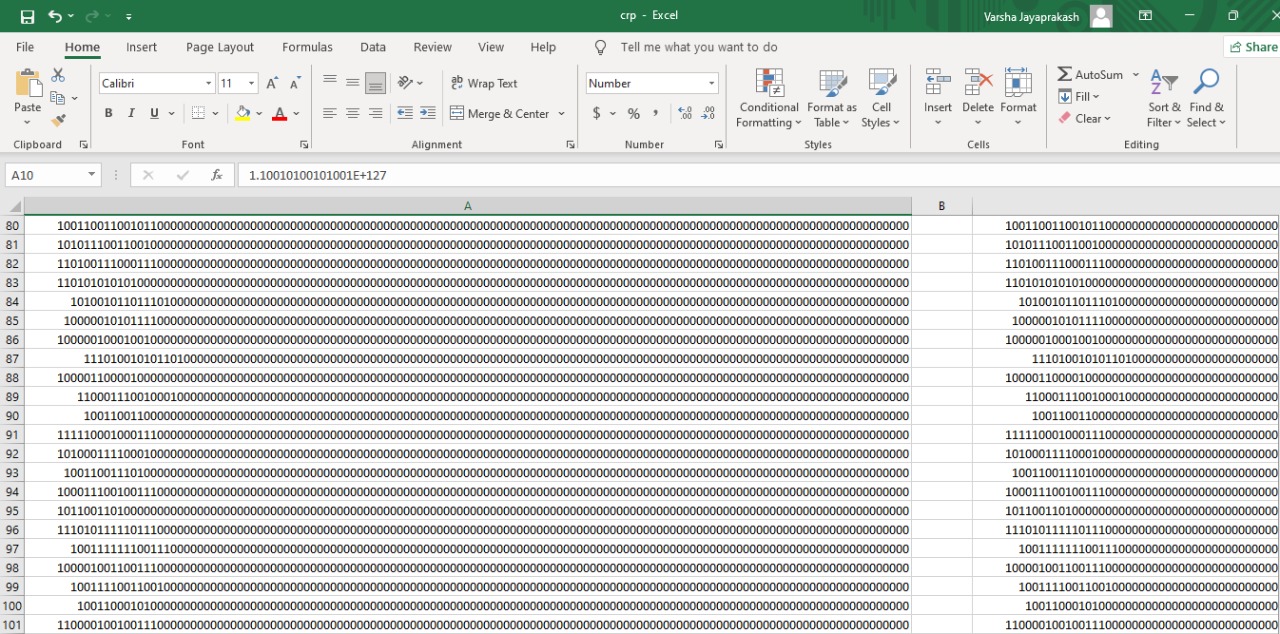
***Fig 4.2.a***: MATLAB Output-1.

The uniformity and uniqueness are calculated using the method mentioned in proposed solution. The obtained results are given in fig. 4.2.b. The obtained results show that the response generated is 53.98% unique and the uniformity of 1’s and 0’s is 59.38% which makes it most suitable for enhanced security.



***Fig 4.2.b***: MATLAB Output-2.

The obtained generated challenge and obtained response is stored in a database to use it for comparison in authentication phase. As shown in figure 4.2.c it contains 128-bit challenge and responses.



***Fig 4.2.c***: CRP database

* 1. **Security Attacks**

Although PUF are termed to be unclonable due to their strong dependence on physical parameters which makes it difficult to produce a device with exact specifications, reports have proven that some numbers of attacks have been simulated against PUF’s and were able to identify the behaviour of the PUF device. PUF is vulnerable to some of the attacks mentioned below.

A) **MODELLING AND MACHNE LEARNING ATTACKS**

The large number of challenge response pairs generated in any strong PUF makes it vulnerable to attacks that can be performed using machine learning algorithm widely used in industry. The challenge response pairs are collected and other circuital parameters are also noted down to develop a machine learning model using available algorithms like logistics regression, Support vector machines (SVM), neural networks etc. The obtained CRP dataset is used as training set to train the model and predict the performance of the any challenge given as input.

**B) INVASIVE ATTACKS**

Since PUF is a physical device, it is prone to invasive attacks where the physical components involved in the device can be damaged to gain access to the chips to study the behaviour of the device. The device needs to be taken to laboratories and studied using expensive tools to break through the device. Although the PUF model generates new set of responses every time after erasing the previous response, reports prove that partial invasive attacks have been successfully performed to study the behaviour of the circuit.

**C) ACTIVE AND PASSIVE ATTACKS**

Active attacks involve any direct modification in the parameters required for the operation of the system such as supply voltage or operational temperature. The device may be tampered upon modification of the parameters. Passive attacks are types of attacks where the performance of the device such as power consumption, temperature, response provided are noted down and reported. They can be used as information for any attacker to tamper the device.

**D) MAN-IN-THE-MIDDLE-ATTACK**

In man-in-the-middle attack an attacker positions himself between the device and server and overhears the communication between the devices. This can be used to obtain the challenge response pair database of any PUF based system and later can be trained and tested using any machine learning model to predict the responses and behaviour of the system. This attack can be performed with little effort as these devices easily connect dynamically to any previously unknown device.

1. **CONCLUSION AND FUTURE ENHANCEMENT**

**5.1 CONCLUSION:**

A PUF based authentication system is developed using XOR Arbiter PUF and is implemented using MATLAB Uniqueness and uniformity of PUF is evaluated. The different security attacks are analyzed and Lightweight authentication and security schemes are suggested to protect IoT devices. Development of PUF based authentication technique is proposed with software analyses based on the CRP and uniqueness and uniformity.

* 1. **FUTURE SCOPE:**

PUF based authentication network can be used in any of the real time applications involving FPGA or RFID tags to enhance the physical security of the device. An integrated circuit of the particular type of PUF can be developed and can be integrated with any physical device. Thy can be used to develop a real time authentication system by collecting a large number of challenge response pairs. Other types of PUF circuits other than XOR Arbiter PUF can be implemented to improve the security of the device. PUF circuits can also be used for secret key generation based on common key agreement and later be used as a key for cryptography algorithms like RSA, AES, ECC etc. to enhance the level of security. Further the performance of PUF based circuits can be studied by varying the parameters like operational voltage, temperature etc. to obtain the performance.

1. **APPENDIX**

**FUNCTION TO GENERATE CHALLENGE AND RESPONSE USING XOR ARBITER PUF**

clear all;

%clc;

tic

%PUF BASED AUTHENTICATION

ID='1234624';

n=randi(100)

c=xlsread('crp.xlsx','A:A');

r=xlsread('crp.xlsx','C:C');

c=c(n);

r=r(n);

% MODIFY HERE

numOfStages=128; %how many stages does one Arbiter PUF have

numOfXors=1; %How many XORs are used (i.e. how many Arbiter PUFs are xored)

numOfInputs=1; %How many different challenge and response paris are generated

%challengeBitsMaster=c;

challengeBitsMaster=randi(2,numOfInputs,numOfStages,'int8')-1; %Generate random challenges with one bit per entry

%load('challengeBitsMaster');

for(currInput=1:numOfInputs)

if (numOfStages == 64)

[c, seq]=LFSR(challengeBitsMaster(currInput,:),[64 63 61 60], numOfStages);

else

[c seq]=LFSR(challengeBitsMaster(currInput,:),[128 126 101 99], numOfStages);

end

challengeBitsAll((currInput -1)\*numOfStages + 1:(currInput - 1)\*numOfStages+numOfStages,:)=c;

%challengeBitsAll(1:64,:)=c;

end

numOfInputs = numOfInputs \* numOfStages;

challengeMatrixUint=zeros(numOfStages/8+1,numOfInputs,'uint8'); %Allocate memory for a more space optimized way to store the challenges with 8 bit per byte, and store it in a way to model the puf using n+1 delay values

% Transform the challenges so that you can model a PUF using n+1 parameters

for(j=1:numOfStages-1)

challengeMatrix(j,:)=1-(mod(sum(challengeBitsAll(:,j:numOfStages)'),2)\*2);

end

challengeMatrix(numOfStages,:)=1-(mod(challengeBitsAll(:,numOfStages),2)\*2);

challengeMatrix(numOfStages+1,:)=1;

% Do the same again. But this time we also store the results more efficiently

% by storing 8 challenge bits into one byte. My optimized mex function

% needs this as an input. So this is only needed for the optimized mex

% function

currIndex=1;

for(j=1:numOfStages)

challengeMatrixUint(currIndex,:)=challengeMatrixUint(currIndex,:)\*2;

challengeMatrixUint(currIndex,:)=challengeMatrixUint(currIndex,:)+uint8(1-(mod(sum(challengeBitsAll(:,j:numOfStages)'),2)));

if(mod(j,8)==0)

currIndex=currIndex+1;

end

end

challengeMatrixUint(numOfStages/8+1,:)=1;

delayVector=zeros(numOfXors,numOfStages+1,'double'); %the delay vectyors of the PUFs with n+1 parameters

%MODIFY HERE FOR DELAY

%Generate PUF instances and compute the ChallengeArray for the n+1

%simulation option

for(currXor=1:numOfXors)

% Generate a PUF using random parameters following a normal

% distribution

delayArray=double(normrnd(0,1,numOfStages,2)); %The delay difference for challenge 0 and challenge 1

% delayArray= intDelayArray;

delayArray= int16(1000\*delayArray);

%load('delayArray128')

delayArrayAllXors(currXor,:,:)=delayArray;

%transform the 2\*n parameters to n+1 parameter according to the

%formula from the paper and store it in delayVector

delayVector(currXor,1)=delayArray(1,1)-delayArray(1,2);

for(j=2:numOfStages)

delayVector(currXor,j)=delayArray(j-1,1)+delayArray(j-1,2)+delayArray(j,1)-delayArray(j,2);

end

delayVector(currXor,numOfStages+1)=delayArray(numOfStages,1)+delayArray(numOfStages,2);

%load('delayVector128')

end

%Option 1: Compute the responses using the original 2\*n delay option

%This implementation a very slow and unoptimized!

%The matrix multiplication is magnitudes faster!

outputArray=logical(zeros(numOfXors,numOfInputs)); %the final output

outputXorArray=logical(zeros(1,numOfInputs)); %the output of the individual arbiter PUFs

for(currXor=1:numOfXors) %For every Arbiter PUF in your XOR PUF constuct

for(currInput=1:numOfInputs)

currDelay=0;

for(currStage=1:numOfStages)

if(challengeBitsAll(currInput,currStage)==0)

currDelay=currDelay+delayArrayAllXors(currXor,currStage,1);

else

%the wires switched, hence the dealy is now inverted:

currDelay=currDelay\*(-1);

currDelay=currDelay+delayArrayAllXors(currXor,currStage,2);

end

currDelayPerStage(currStage)=currDelay;

end

if(currDelay>0)

outputArray(currXor,currInput)=1;

else

outputArray(currXor,currInput)=0;

end

%we use an XOR arbiter PUF we need to XOR the current PUF

%response with the PUF response from the other PUFs

% but A~=B is the same as A xor B

outputXorArray(currInput)=outputXorArray(currInput)~=outputArray(currXor,currInput);

end

end

outputArray2=logical(zeros(numOfXors,numOfInputs)); %the final output

outputXorArray2=logical(zeros(1,numOfInputs)); %the output of the individual arbiter PUFs

% An optimized implementation using the

%Generate PUF instances and compute the response

for(currXor=1:numOfXors)

%Compute the delay differences

currDelay2=delayVector(currXor,:)\*challengeMatrix(:,:);

outputArray2(currXor,:)=currDelay2>0;

% outputArray2(currXor,:)=comPUFwLUT(delayVector(currXor,:)',challengeMatrixUint(:,:));

%Xor the response bit of the current PUF with the responses of the previous PUF for an XOR PUF

outputXorArray2=outputArray2(currXor,:)~=outputXorArray2;

end

%outputXorArray should be identical with outputXorArray2 and contain the

%final responses

sum(abs(outputXorArray2-outputXorArray));

csvwrite('challenges.csv',challengeBitsMaster);

csvwrite('reference.csv',outputArray2);

resp\_gen=outputArray2;

%CHECKING FOR AUTHENTICATION

id=input('ENTER ID OF DEVICE ');

if id==ID

if r==resp\_gen

disp('AUTHENTICAL SUCCESFUL');

end

elseif id~=ID

disp('ACCESS DENIED');

elseif r~=resp\_gen

disp('ACESS DENIED')

end

disp('all done')

**LFSR TO GENERATE PSEUDO-RANDUM NUMBERS**

function[c seq]=LFSR(s,t,ln)

%--------------------------------------------------

n=length(s);

c(1,:)=s;

m=length(t);

index = 1;

for k=1:n\*(n-1) ;

b(1)=xor(s(t(1)), s(t(2)));

if m>2;

for i=1:m-2;

b(i+1)=xor(s(t(i+2)), b(i));

end

end

j=1:n-1;

s(n+1-j)=s(n-j);

s(1)=b(m-1);

if mod(k,n) == 0

c(index+1,:)=s;

index = index + 1;

end

end

seq=c(:,n)';

**FUNCTION TO LOAD RESPONSE AND EVAULATE PERFORMANCE**

R = collectResponses('res25C.dat', 30);

fprintf('Uniqueness:\t%5.2f%%\n', uniqueness(R));

% Compute the uniformity of 128 bit XOR Arbiter PUF

puf = R(2,:);

fprintf('Uniformity:\t%5.2f%%\n', uniformity(puf));

% Compute the uniformity for 18 bit XOR Arbiter PUF.

FUNCTION TO COLLECT RESPONSES FROM .dat FILE

function [data] = collectResponses(fileName, rowNum)

fileID = fopen(fileName); %Open the file

if(fileID<0)

disp('Error: No such file !');

return;

end

dataRow = fgetl(fileID); %Read one row from the .dat file

n = length(dataRow); %Calculate the length of each row

data = zeros(rowNum, n); %Pre allocating

data(1,:) = dataRow; %Save the data

for i = 2:rowNum

dataRow = fgetl(fileID);

data(i, :) = dataRow;

end

data = data - 48;

fclose(fileID); %Close the file

end

**FUNCTION TO CALCULATE UNIQUENESS**

function [uniquenessValue] = uniqueness(responseSet)

k = size(responseSet, 1); %The number of PUFs

n = size(responseSet, 2); %The number of response bits

total\_HD=0;

for i=1:k-1

for j=i+1:k

total\_HD = total\_HD + sum(abs(responseSet(i,:)- responseSet(j,:)));

end

end

uniquenessValue = (2\*total\_HD)/(n\*k\*(k-1))\*100;

end

**FUNCTION TO CALCULATE UNIFORMITY**

function [uniformityValue] = uniformity(response)

%Convert character to integer

if(response(1)>40)

response = response - 48;

end

uniformityValue = (sum(response)/length(response))\*100;

end

1. **REFERENCES**

[1] Ovilla-Martínez, Brisbane, Cuauhtémoc Mancillas-López, Alberto F. Martínez Herrera, and José A. Bernal-Gutiérrez 2020. "FPGA Implementation of Some Second Round NIST Lightweight Cryptography Candidates'' Electronics 9, no. 11: 1940. https://doi.org/10.3390/electronics9111940

[2] Muhammad Usman, Irfan Ahmed, M. Imran Aslam, Shujaat Khan, Usman Ali Shah, “SIT: A Lightweight Encryption Algorithm for Secure Internet of Things”. DOI: 10.14569/IJACSA.2017.080151

[3] Nagham Samir, Abdelrahman Sobeih Hussein, Mohaned Khaled, Ahmed N. El-Zeiny, Mahetab Osama, Heba Yassin, Ali Abdelbaky, Omar Mahmoud, Ahmed Shawky and Hassan Mostafa, “ASIC and FPGA Comparative Study for IoT Lightweight Hardware Security Algorithms”.DOI: https://doi.org/10.1142/S0218126619300095

[4] Chowdary, G. Rishika, Sheetal Neeraj, and Sai Sparsha. "MACHINE LEARNING APPROACHES FOR DIFFERENT CYBERTHREATS."

[5] Lara-Nino, Carlos Andres, Arturo Diaz-Perez, and Miguel Morales-Sandoval. "Lightweight hardware architectures for the PRESENT cipher in FPGA." IEEE Transactions on Circuits and Systems I: Regular Papers 64, no. 9 (2017): 2544-2555.

[6] Abebe, Abiy Tadesse. "Efficient FPGA Implementation of Lightweight Authenticated Encryption and Authenticated Key Distribution Methods for IoT Security." International Journal of Digital Information and Wireless Communications 9, no. 3 (2019): 165-187.

[7] Kavun, Elif Bilge, and Tolga Yalcin. "RAM-based ultra-lightweight FPGA implementation of PRESENT." In 2011 International Conference on Reconfigurable Computing and FPGAs, pp. 280-285. IEEE, 2011.

[8] Chakraborty, R. and Mandal, J.K., 2020. An FPGA-Based Advanced Lightweight Cryptography Architecture for IoT Security and Its Cryptanalysis. In Information Security and Optimization (pp. 93-105). Chapman and Hall/CRC.

[9] Braeken, A., 2018. PUF based authentication protocol for IoT. Symmetry, 10(8), p.352.

[10] Järvinen K., Miele A., Azarderakhsh R., Longa P. (2016) Four Q on FPGA: New Hardware Speed Records for Elliptic Curve Cryptography over Large Prime Characteristic Fields. In: Gierlichs B., Poschmann A. (eds) Cryptographic Hardware and Embedded Systems – CHES 2016. CHES 2016. Lecture Notes in Computer Science, vol 9813. Springer, Berlin, Heidelberg. <https://doi.org/10.1007/978-3-662-53140-2_25>.

[11] W. Che, M. Martin, G. Pocklassery, V. Kajuluri, F. Saqib, and J. Plusquellic, “A Privacy-Preserving, Mutual PUF-Based Authentication Protocol,” *Cryptography*, vol. 1, no. 1, p. 3, Nov. 2016 [Online]. Available: <http://dx.doi.org/10.3390/cryptography1010003>.

[12] Y. Yilmaz, S. R. Gunn and B. Halak, "Lightweight PUF-Based Authentication Protocol for IoT Devices," 2018 IEEE 3rd International Verification and Security Workshop (IVSW), 2018, pp. 38-43, doi: 10.1109/IVSW.2018.8494884.

[13] Zhou, Chen, Keshab K. Parhi, and Chris H. Kim. "Secure and reliable XOR arbiter PUF design: An experimental study based on 1 trillion challenge response pair measurements." In *Proceedings of the 54th Annual Design Automation Conference 2017*, pp. 1-6. 2017.

[14] Wisiol N., Becker G.T., Margraf M., Soroceanu T.A.A., Tobisch J., Zengin B. (2020) Breaking the Lightweight Secure PUF: Understanding the Relation of Input Transformations and Machine Learning Resistance. In: Belaïd S., Güneysu T. (eds) Smart Card Research and Advanced Applications. CARDIS 2019. Lecture Notes in Computer Science, vol 11833. Springer, Cham. https://doi.org/10.1007/978-3-030-42068-0\_3

[15] K. T. Mursi, Y. Zhuang, M. S. Alkatheiri and A. O. Aseeri, "Extensive Examination of XOR Arbiter PUFs as Security Primitives for Resource-Constrained IoT Devices," 2019 17th International Conference on Privacy, Security and Trust (PST), 2019, pp. 1-9, doi: 10.1109/PST47121.2019.8949070.

[16] Wisiol, N., Margraf, M. Why attackers lose: design and security analysis of arbitrarily large XOR arbiter PUFs. *J Cryptogr Eng***9,**221–230 (2019). https://doi.org/10.1007/s13389-019-00204-8.

1. **BIODATA**



**Name:** S. CHARAN

**Mobile Number:** +91 95661 90703

**E-mail:** [charan.s2019@vitstudent.ac.in](mailto:charan.s2019@vitstudent.ac.in)

**Permanent Address:** 8/15, Kamaraj Nagar, 2nd St., Chennai-93



**Name:** VARSHA JAYAPRAKASH

**Mobile Number:** +91 95001 53451

**E-mail:** [varsha.jayaprakash2019@vitstudent.ac.in](mailto:varsha.jayaprakash2019@vitstudent.ac.in)

**Permanent Address:** 1A, Aarathi flats, New Tank St., Chennai-34

 **Name:** C. POOJAA

**Mobile Number:** +91 82201 11945

**E-mail:** [poojaa.c2019@vitstudent.ac.in](mailto:poojaa.c2019@vitstudent.ac.in)

**Permanent Address:** A-401, Purva Jade, Alwartirunagar, Chennai-87