

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

**KONUM MAHREMİYETİ SAĞLAYAN BİR RFID PROTOKOLÜNÜN
ANALİZİ VE GERÇEKLENMESİ**

YÜKSEK LİSANS TEZİ

Özen ÖZKAYA

Elektronik ve Haberleşme Mühendisliği Anabilim Dalı

Elektronik Mühendisliği Programı

HAZİRAN 2013

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

**KONUM MAHREMİYETİ SAĞLAYAN BİR RFID PROTOKOLÜNÜN
ANALİZİ VE GERÇEKLENMESİ**

YÜKSEK LİSANS TEZİ

**Özen ÖZKAYA
504101220**

Elektronik ve Haberleşme Mühendisliği Anabilim Dalı

Elektronik Mühendisliği Programı

Tez Danışmanı: Doç. Dr. S. Berna Örs YALÇIN

HAZİRAN 2013

İTÜ, Fen Bilimleri Enstitüsü'nün 504101220 numaralı Yüksek Lisans Öğrencisi **Özen ÖZKAYA** ilgili yönetmeliklerin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı “**GÜVENLİ BİR RFID PROTOKOLUNUN ANALİZİ VE GERÇEKLENMESİ**” başlıklı tezini aşağıda imzaları olan jüri önünde başarı ile sunmuştur.

Tez Danışmanı : **Doç. Dr. S. Berna Örs YALÇIN**
İstanbul Teknik Üniversitesi

Jüri Üyeleri : **Prof. Dr. Ece Olcay GÜNEŞ**
İstanbul Teknik Üniversitesi

Doç. Dr. Arda YURDAKUL
Boğaziçi Üniversitesi

Teslim Tarihi : **03 Mayıs 2013**
Savunma Tarihi : **07 Haziran 2013**

Aileme ve Dostlarıma,

ÖNSÖZ

Mensubu olmaktan gurur duyduğum İstanbul Teknik Üniversitesi'nde yürüttüğüm tez çalışmam boyunca her zaman manevi desteğini hissettiğim aileme ve arkadaşım Elektronik Mühendisi Nilay TÜFEK'e, derin bilgi birikimini benimle paylaşan ve çalışmalarına yön veren değerli hocam Doç. Dr. S. Berna Örs YALÇIN'a sonsuz teşekkürlerimi sunarım.

Yine tez çalışmam boyunca her zaman yanımda olan dostlarım Abidin ALTINTAŞ ve Seyhan ÇALIŞKAN'a içten teşekkürlerimi sunarım.

Ayrıca, çalışanı olduğum şirkette edinmiş olduğum özgüveni ve profesyonel iş yürütme bilincini bana kazandıran yöneticim Elektronik Yüksek Mühendisi M. Suha GÜLERMAN'a teşekkür etmeyi kendime bir borç bilirim.

Haziran 2013

Özen Özkaya
Elektronik Mühendisi

İÇİNDEKİLER

Sayfa

ÖNSÖZ.....	vii
İÇİNDEKİLER	ix
KISALTMALAR	xi
ŞEKİL LİSTESİ.....	xv
ÖZET.....	xvii
SUMMARY	xix
1. GİRİŞ	1
2. ÇALIŞMA ALTYAPISI	3
2.1 Giriş.....	3
2.2 RFID Sistemleri.....	4
2.2.1 Etiket	6
2.2.2 Okuyucu	10
2.2.3 Veritabanı.....	11
2.3 RFID Sistemlerinin Sınıflandırılması.....	12
2.4 RFID Sistemlerinde Güvenlik ve Mahremiyet.....	13
2.4.1 Güvenlik.....	14
2.4.2 Mahremiyet	15
3. ÖZET FONKSİYONLARI	19
Genel Bilgi	19
3.1 SHA-3 Finalistleri	21
3.1.1 Blake	21
3.1.2 Grøstl.....	22
3.1.3 Jh	23
3.1.4 Skein.....	24
3.1.5 Keccak.....	25
4. KONUM MAHREMİYETİ SAĞLAYAN RFID PROTOKOLÜ	27
4.1 Protokol Tanımı.....	27
4.2 Protokolün Analizi	31
5. ÖZET FONKSİYONLARININ GERÇEKLENMESİ	33
5.1 Kullanılan Donanım Platformları.....	33
5.1.1 Rfm22b.....	33
5.1.2 Msp430.....	36
5.1.3 Stm32	37
5.1.4 Atmega128.....	39
5.2 Literatür Analizi	41
5.3 Gerçekleme Sonuçları	43
5.3.1 Blake-256	44
5.3.2 Grøstl-256	47
5.3.3 Keccak-256	50
5.3.4 Skein-256	53
5.3.5 Jh-256.....	56

5.4 Performans Karşılaştırmaları	59
5.5 Sonuç	66
6. PROTOKOLÜN GERÇEKLENMESİ	69
6.1 Geliştirmeler	69
6.1.1 Etiketten okuyucuya kuyruk isteği	71
6.1.2 Okuyucudan etikete kuyruk cevabı	72
6.1.3 Etiketten okuyucuya bağlantı isteği	72
6.1.4 Okuyucudan veritabanına bağlantı isteği	73
6.1.5 Veritabanından okuyucuya bağlantı cevabı	73
6.1.6 Okuyucudan etikete bağlantı cevabı	74
6.2 Gerçekleme Ortamı	74
7. SONUÇ VE ÖNERİLER.....	83
7.1 Çalışmanın Uygulama Alanı	84
KAYNAKLAR.....	87
EKLER.....	91
ÖZGEÇMİŞ.....	93

KISALTMALAR

ADC	: Analog Digital Converter
AE	: Associated Entry
AES	: Advanced Encryption Standard
ARM	: Acorn RISC Machine
CAN	: Control Area Network
CPU	: Central Processing Unit
CRC	: Cyclic Redundancy Check
DB-ID	: Database Identifier
DB	: Database
DMIPS	: Dhrystone Million Instructions Per Second
EAS	: Electronic Article Surveillance
EEPROM	: Electronically Erasable Programmable Read-Only Memory
FIFO	: First in first out
FRAM	: Ferroelectric Random-Access Memory
GPU	: Graphics Processing Unit
HF	: High Frequency
I2C	: Inter-Integrated Circuit
IC	: Integrated Circuit
ID	: Identifier
IFF	: Identify Friend or Foe
ISO	: International Organization for Standardization
JTAG	: Joint Test Action Group
LED	: Light Emitting Diode
LF	: Low Frequency
LST	: Last successful transaction
MAC	: Message Authentication Codes
MD5	: Message-Digest Algorithm 5
MDC	: Manipulation (or Modification) Detection Codes
MIC	: Message Integrity Codes
MIPS	: Million Instruction Per Second
MIT	: Massachusetts Institute of Technology
MSJ	: Mesaj
NIST	: National Institute of Standards and Technology
OGS	: Otomatik Geçiş Sistemi
OS	: Operating System
PWM	: Pulse Width Modulation
RAM	: Random-Access Memory
RC	: Resistor Capacitor
RF	: Radio Frequency
RFID	: Radio Frequency Identification
RISC	: Reduced Instruction Set Computing
RND	: Random
ROM	: Read-Only Memory

RSA	: Rivest Shamir Adleman
SHA	: Secure Hash Algorithm
SRAM	: Static Random-Access Memory
SWD	: Single Wire Debug
TID	: Transaction ID
UHF	: Ultra High Frequency
USART	: Universal Synchronous and Asynchronous Serial Receiver and Transmitter
USB	: Universal Serial Bus
USD	: United States Dollar

ÇİZELGE LİSTESİ

	<u>Sayfa</u>
Çizelge 2.1 : Barkod ve RFID karşılaştırması	3
Çizelge 2.2 : Aktif, yarı aktif ve pasif etiketler	7
Çizelge 2.3 : Etiketlerin fonksiyonelliklerine göre sınıflandırılması	8
Çizelge 2.4 : RFID sistemlerinin sınıflandırılması	13
Çizelge 5.1 : Algoritma gerçeklemelerinin byte cinsinden bellek kullanımı	60
Çizelge 5.2 : Atmega128 için algoritma gerçeklemelerinin byte/saat darbesi cinsinden işletim hızı	60
Çizelge 5.3 : Msp430 için algoritma gerçeklemelerinin byte/saat darbesi cinsinden işletim hızı	61
Çizelge 5.4 : Stm32 için algoritma gerçeklemelerinin byte/saat darbesi cinsinden işletim hızı	61
Çizelge 6.1 : Mesaj tanımları	71
Çizelge 6.2 : 1 nolu mesajın yapısı	71
Çizelge 6.2 : 2 nolu mesajın yapısı	72
Çizelge 6.2 : 3 nolu mesajın yapısı	73
Çizelge 6.2 : 4 nolu mesajın yapısı	73
Çizelge 6.2 : 5 nolu mesajın yapısı	74
Çizelge 6.2 : 6 nolu mesajın yapısı	74

ŞEKİL LİSTESİ

Sayfa

Şekil 2.1 : RFID patenti.	5
Şekil 2.2 : RFID sistemi.	6
Şekil 2.3 : RFID etiketleri.....	7
Şekil 2.4 : Bellekli RFID etiketi yapısı.....	9
Şekil 2.5 : Mikroişlemcili RFID etiketi yapısı.....	10
Şekil 2.6 : RFID okuyucu.	11
Şekil 3.1 : Özet fonksiyonunun sınıflandırılması	20
Şekil 3.2 : Sutun adımı ve diagonal adım	22
Şekil 3.3 : <i>Gi</i> bloğunun iç yapısı.....	22
Şekil 3.4 : Grøstl özet fonksiyonu	23
Şekil 3.5 : <i>f</i> sıkıştırma fonksiyonu	23
Şekil 3.6 : JH sıkıştırma fonksiyonu.....	24
Şekil 3.7 : Skein algoritmasının sıkıştırma fonksiyonun 4 turu.....	25
Şekil 4.1 : Sistem şeması.....	29
Şekil 4.2 : Detaylı sistem şeması.....	30
Şekil 5.1 : Rfm22b modülü.....	34
Şekil 5.2 : Rfm22b modülünün blok diyagramı	35
Şekil 5.3 : Gerçeklemede kullanılan Rfm22b modülü	35
Şekil 5.4 : MSP430 mikrokontrolörü.	36
Şekil 5.5 : Msp430 Launchpad kiti.....	37
Şekil 5.6 : Stm32 mikrokontrolörü	38
Şekil 5.7 : Mcbstm32 geliştirme kiti	39
Şekil 5.8 : Keil Ulink2 ve Mcbstm32 geliştirme kiti.....	39
Şekil 5.9 : Atmega128 mikrokontrolörü.	40
Şekil 5.10 : Stk-500 kiti.....	41
Şekil 5.11 : Blake algoritmasının “ABCDEFGH” girişi için MSP430 üzerinde benzetimi	45
Şekil 5.12 : Blake algoritmasının “ABCDEFGG” girişi için MSP430 üzerinde benzetimi	46
Şekil 5.13 : Blake algoritmasının orta uzunluktaki veri girişi için MSP430 üzerinde benzetimi	46
Şekil 5.14 : Blake algoritmasının uzun veri girişi için MSP430 üzerinde benzetimi	47
Şekil 5.15 : Grøstl algoritmasının “ABCDEFGH” girişi için Msp430 üzerinde benzetimi	48
Şekil 5.16 : Grøstl algoritmasının “ABCDEFGG” girişi için Msp430 üzerinde benzetimi	49
Şekil 5.17 : Grøstl algoritmasının orta uzunluktaki veri girişi için Msp430 üzerinde benzetimi.	49
Şekil 5.18 : Grøstl algoritmasının orta uzunluktaki veri girişi için Msp430 üzerinde benzetimi.	50

Şekil 5.19 : Keccak algoritmasının “ABCDEFGH” girişi için Msp430 üzerinde benzetimi.	51
Şekil 5.20 : Keccak algoritmasının “ABCDEFGG” girişi için Msp430 üzerinde benzetimi.	52
Şekil 5.21 : Keccak algoritmasının orta uzunluktaki veri girişi için MSP430 üzerinde benzetimi.	52
Şekil 5.22 : Keccak algoritmasının uzun veri girişi için Msp430 üzerinde benzetimi.	54
Şekil 5.23 : Skein algoritmasının “ABCDEFGH” girişi için Msp430 üzerinde benzetimi.	54
Şekil 5.24 : Skein algoritmasının “ABCDEFGG” girişi için Msp430 üzerinde benzetimi.	54
Şekil 5.25 : Skein algoritmasının orta uzunluktaki veri girişi için MSP430 üzerinde benzetimi.	55
Şekil 5.26 : Skein algoritmasının uzun veri girişi için Msp430 üzerinde benzetimi.	56
Şekil 5.27 : Jh algoritmasının “ABCDEFGH” girişi için Msp430 üzerinde simülasyonu.....	57
Şekil 5.28 : Jh algoritmasının “ABCDEFGG” girişi için Msp430 üzerinde benzetimi.	57
Şekil 5.29 : Jh algoritmasının orta uzunluktaki veri girişi için MSP430 üzerinde benzetimi.	58
Şekil 5.30 : Jh algoritmasının uzun veri girişi için Msp430 üzerinde benzetimi.	59
Şekil 5.31 : Msp430 saat darbesi/byte değerleri.	62
Şekil 5.32 : Atmega128 için saat darbesi/byte değerleri.	63
Şekil 5.33 : Stm32 için saat darbesi/byte değerleri.....	63
Şekil 5.34 : Kısa uzunluklu giriş için byte başına saat darbesi değerleri.....	64
Şekil 5.35 : Orta uzunluklu giriş için byte başına saat darbesi değerleri.	65
Şekil 5.36 : Büyük uzunluklu giriş için byte başına saat darbesi değerleri.	65
Şekil 5.37 : Gerçeklemelerin RAM kullanımı.	66
Şekil 6.1 : Çerçeve protokolün yapısı.	69
Şekil 6.2 : Gerçeklenen sistemin akış diyagramı.	75
Şekil 6.3 : Okuyucu çalışmaya başlarken.	76
Şekil 6.4 : Okuyucu bir etiketi kuyruğa eklerken.	76
Şekil 6.5 : Birinci mesaj okuyucu tarafından alınırken.	77
Şekil 6.6 : Δ TID okuyucu tarafından alınırken.....	78
Şekil 6.7 : Yeni TID okuyucu tarafından onaylanırken.....	78
Şekil 6.8 : Veritabanından okuyucuya cevap geldiğinde.....	79
Şekil 6.9 : Haberleşme tamamlandığında.	80
Şekil 6.10 : Haberleşmede hata oluştuğunda	81

KONUM MAHREMİYETİ SAĞLAYAN BİR RFID PROTOKOLÜNÜN ANALİZİ VE GERÇEKLENMESİ

ÖZET

Bu Yüksek Lisans tezi kapsamında Radyo Frekanslı Tanımlama (RFID) sistemlerinin konum mahremiyeti (location privacy) üzerinde durulmuştur. Tez kapsamında öncelikli olarak RFID’de sunulan mevcut protokollere ilişkin güvenlik ve mahremiyet analizleri yapılmış, bunlarda görülen eksiklikler kapatılarak yeni bir protokol ortaya atılmıştır. Bu protokolün ortaya atılmasındaki temel motivasyon ise konum mahremiyeti sağlayan bir RFID protokolü ortaya çıkarmaktır. Ortaya atılan bu protokol, fiziksel olarak gerçekleştirilen bir RFID sisteminde analiz edilmiştir. Bu analizlerle de anlaşılabileceği üzere, teorik ve pratik anlamda konum mahremiyeti sağlayan bir RFID protokolu elde edilmiştir. Tüm implementasyonlar, mikrodenetleyiciler üzerinde yapılmıştır. İleriki kısımda bu çalışmalar detaylıca anlatılacaktır.

RFID sistemlerinin pratik kullanımda ortaya çıkışı ikinci dünya savaşına dayanmaktadır. RFID sistemleri yardımıyla, dost savaş uçaklarının düşman savaş uçaklarından ayrılması işlemi gerçekleştirilmiştir. Bu günün dünyasında RFID sistemleri sağlık sektöründen trafiğe, hayvancılıktan ulaşımaya kadar çok farklı sektörlerde kullanılmaktadır. Günden güne artan kullanım alanları ile RFID sistemleri her geçen gün daha fazla önem kazanmaktadır.

RFID teknolojisi, herhangi bir fiziksel temasa ihtiyaç duymaksızın, etiket taşıyan bir cisim ya da kişinin kimliğinin, radyo dalgaları vasıtasıyla doğrulanmasını sağlayan bir teknolojidir. Bir RFID sisteminin temel anlamda üç bileşeni bulunmaktadır. Bunlar, kimliği tanımlanacak olan etiket (tag), etiketleri tanımlayacak olan okuyucu (reader) ve bilgilerin depolandığı (server)’dır. Kimi durumlarda depolama ünitesi ile okuyucunun birleşik olabildiği de görülmektedir. Daha da detaylandırarak olursak, RFID etiketi, okuyucudan gelen sorgulamaları alacak ve cevaplayabilecek şekilde tasarlanmış bir silikon yonga, anten ve kaplamadan meydana gelir. Silikon yonga’nın görevi etiketin üzerinde bulunduğu cisim ya da kişi ile ilgili bilgileri saklamak ve kimi durumlarda sınırlı sayısal operasyonları gerçekleştirmektir. Anten, radyo frekansındaki dalgaları toplayarak, verileri yongaya iletir. Etiketler, hafıza miktarı, okuma/yazma kapasitesi, yürütülebilen sayısal fonksiyonlar gibi kriterlere bağlı olarak farklılıklar gösterir. Etiketler, okuma sırasında kullanılan frekans aralığına bağlı olarak LF, HF, UHF olarak çeşitlendirilebilirler. Ayrıca beslenme şekillerine göre, pasif etiket, yarı-aktif etiket ve aktif etiket olarak da ayrılabilirler.

Hayatımızda her geçen gün daha fazla yer edinen RFID teknolojisi, güvenlik ve kullanıcı mahremiyeti gibi önemli kavramları da birer endişe olarak beraberinde getirmiştir. Çok sayıda araştırmacının güvenlik ve gizlilik üzerine çalışmalar yürütmesine rağmen mevcut durumda ilgili sorunlar tam olarak çözülememiştir. RFID etiketleri, üzerinde bulundukları kişilerle ilgili bazı bilgiler taşımaktadırlar. Günümüzde en yaygın olarak kullanılan RFID etiketleri, tüm sorgulara yanıt veren

yapıdadırlar ve bunu yapmak için üzerinde bulundukları kişiden bir onay almazlar. Bunun sonucu olarak RFID etiketlerini taşıyan kişilerin habersiz olarak izlenmesi ve özel hayatın ihlali konuları ortaya çıkmaktadır. Konunun ahlaki yanısıyla birlikte kötü niyetli kişilerin bu bilgileri kullanmasıyla doğabilecek çok sayıda potansiyel problem de bulunmaktadır. Tüm bu sorunlar RFID sistemlerini üzerinde çok sayıda araştırmacının çalıştığı bir konu haline getirmiştir. Endüstri de akademik dünyadaki çalışmaları büyük bir dikkatle takip etmektedir.

Bu çalışma öncelikli olarak mahremiyet özelliği sunan protokollerin analizini içermektedir. Sonraki kısımlarda ise fiziksel anlamda da mahremiyet sunan bir protokolün gerçekleştirilmesi için gerekli altyapıya değinilecek, ardından oluşturulan protokol sunulacaktır.

RFID protokollerinde güvenlik ve mahremiyetin sağlanmasında özet (hash) fonksiyonlarının önemi büyüktür. Günümüzde özet fonksiyonları ile ilgili standartları belirleyen kurum NIST'dir. NIST varolan güvenlik açıklarını kapatmak ve yeni bir özet fonksiyonu standardı ortaya koymak adına "SHA-3 Contest" adlı yarışmayı düzenlemiştir. Bu yarışmada beş algoritma finale kalmıştır. Mahremiyet sunan bir protokolün en önemli yapı taşı özet fonksiyonu olduğundan bu alanda da bu tez kapsamında çalışmalar yürütülmüş ve implementasyonlar yapılmıştır. Bu implementasyonların analizi neticesinde Blake algoritması en uygun olarak seçilmiş, ve yarışmayı Keccak kazanmasına rağmen Blake algoritması protokolde kullanılmıştır.

Yapılan testler ve deney sonuçları protokolün teorik anlamda da pratik anlamda da mahremiyet sunduğunu göstermiştir. Sonuç olarak bu çalışma ile ortaya konan çıktının, pratik olarak da kullanılabilir olduğu gösterilmiştir.

ANALYSIS AND IMPLEMENTATION OF AN RFID PROTOCOL WHICH PROVIDES LOCATION PRIVACY

SUMMARY

Location privacy of Radio Frequency Identification (RFID) systems are emphasized in the scope of this Master of Science thesis. First of all the analysis of protocols which are presented for RFID systems are made. Then the vulnerabilities of the presented protocols are removed and a new protocol is suggested and implemented. The actual motivation of the suggested protocol is to reveal such a protocol that provides location privacy for practical usage. Suggested protocol was analysed on a physically implemented RFID system. As it can be seen by the analysis that an RFID system is suggested in both theoretical and practical terms. All of the implementations are made by using the microcontrollers. These studies will be explained in the next chapters.

The emergence of practical usage of RFID systems depends on the Second World War. Operation of separating ally warcrafts from enemy warcrafts was done thanks to RFID systems. Nowadays RFID systems are used in many sectors such as health, transportation and animal husbandry. By the daily increase of usage fields of RFID systems are becoming more important day by day.

RFID is a technology that authenticates the object or the person carrying the tag without needing any physical contact by radio waves. RFID systems basically have three components. These components are the tag whose identity is identified, the reader which identifies the tags and the server that keeps the data. Server can sometimes be seen as united to reader. If we explain in detail, a tag is composed of a silicon chip, antenna and covering that takes the queries sent from reader and responds to these queries. The duty of the silicon chip is to keep the information about the object or person carried by, and to realize some restricted digital operations. The duty of the antenna is to gather the radio frequency waves and transmits them to the silicon chip. Tags differ from criterias such as the amount of memory, capability of read/write and ability to perform the numeric functions are being executed. Tag can be diversified depending on the reading frequency as LF, HF, UHF, and SHF. Moreover, tags can be separated as passive, active, semi-active according to the type of feeding source.

As RFID technology has made a place in our lives more and more every day, it also brings the user privacy and security issues as a crucial concern. Although many researchers have been performing studies about privacy and security, problems about these issues can not be solved exactly in the present case. RFID tags contain some information about the person who carries them. Today's commonly used RFID tags respond all of the queries but they do not take permission to give an answer. As a result, issues of the violation of privacy and unaware monitoring or the follow-up of the carriers emerge. Along with the moral side, there are many potential problems depending on using this information by the evil people. Therefore, these problems attract many researchers to study about RFID systems and the number of the researchers increases day by day. Industrial companies pursue the studies of the academic world with a big attention.

This study initially contains the analysis of the privacy proposing protocols. In the next chapters a basis that proposes privacy also in the physical meaning, will be mentioned. Then the protocol will be presented.

A microcontroller based system generally means limited resources. To reduce price and increase efficiency, tasks should be executed with minimum resources and in an optimized way. Also the design should be made by using minimum resources and should provide a satisfying real time performance. This trade-off is mostly a challenge in RFID systems because RFID tags have very limited resources and still need security and privacy improvements. Texas Instruments MSP430 family is an ultra-low power, ultra-low cost 16-bit microcontroller which is proposed as a current solution for this trade-off. The successful implementation of security functions on MSP430 means a big step for secure, low-power, low -cost communication systems. To sum up, MSP430 implementations are key point on the way to provide secure communication systems with lowest resources. ATmega series of ATMEL is one of the most popular 8-bit microcontrollers and lots of cryptographic implementations written for it. Because of its popularity, Atmel is the best choice among 8-bit microcontrollers. For a 32-bit high performance low cost microcontroller STM32 is one of the most popular, also. By choosing popular microcontrollers from the line between 8-bit and 32-bit, this research aims a wide sense comparison of SHA-3 finalists on different microcontroller platforms.

Platform independence of an implementation is a very important feature in practice. The devices that are using hash functions for communication are not usually based on same hardware platforms. For example, an RFID system consists of a reader and tags. Reader should have high capabilities to manage all of the tags. On the other hand, tag should be as basic as it can be and it should have minimal power consumption with smallest possible resources. This scheme shows the differences between tag and reader devices in the manner of performance. In literature, most of the implementations tested on just one hardware platform although it is not the realistic case.

Hash functions have big importance for the providing of privacy and security of RFID protocols. Today NIST determines the standarts about hash functions. NIST organized a contest named as “SHA-3” to eliminate the security issues and to present a new hash function standard. Five algorithms reached the final stage. Because hash functions are the most important building blocks of the privacy proposing protocols, many studies were performed and implementations were done. As a result of the analysis of the implementations, Blake algorithm is chosen as the most convenient protocol. Therefore, Blake algorithm is used in our protokol although Keccak won the competition.

This research covers the comparison of lightweight implementations of SHA-3 finalist algorithms on different level embedded system platforms. Implementations tested on most challenging and widely used hardware platforms by using data inputs with different length. To construct platform-independent lightweight implementations, specific optimizations are not the focus of this paper, but general embedded system considerations are covered in the next sections.

A brief explanation of each lightweight implementation is given. Then the goals, considerations and other implementation issues are argued. 256 bit length versions of the algorithms are implemented in order to settle on a common characteristics regarding hash-length and to limit the amount of code to be developed and ranked.

256 bit versions are the most suitable ones for embedded systems. Each created implementation is fully functional and a standalone application. Each implementation is tested and verified by an automatic check of the message digest. All implementations include code sections for hardware initializations, module configurations and measurement software implementations. However, these code parts are not included for performance evaluation or memory analysis. Hence, performance metrics of the algorithms are measured only. Furthermore, memory organization flexibility is also a good feature for embedded system performance. Algorithms memory usage can flexibly be distributed to Flash or RAM. Only speed based benchmarking is not enough to decide the performance rating of an embedded system. Within the embedded systems resource constraints RAM or FLASH can be individually limited. In some cases, it is needed to carry constant variables to RAM also, because FLASH memory is very limited. In the first view, it is not logical to keep constant variables to RAM but it is a reality in some RFID applications. Minimum possible RAM usage or minimum possible FLASH usage of algorithm is an important parameter for preference of the appropriate algorithm for given embedded hardware platform. Hence for each SHA-3 candidate RAM and Flash optimized (minimized) implementations are created separately. All of the memory optimizations are made without affecting the speed dramatically, because execution time of the algorithm is much more than the access speed difference between RAM and FLASH.

The test and experiment results demonstrate that our protocol proposes in both theoretical and practical terms. In conclusion, the output that is revealed by this study is also shown that it can be used practically.

1. GİRİŞ

Gelişen ticaret ve ulaşım ile birlikte nesnelerin tanımlanması bir gereklilik haline gelmiştir. Dünya ticaretinde artan hacimsel büyüme ve çeşitlilik artışı, lojistik ve envanter kontrolü maliyetlerinin öneminin bir hayli artmasına sebep olmuştur. Süpermarketler, bu ihtiyacın ilk baş gösterdiği yerler olmuş ve barkod kullanımı ile ürünlerin tanınması amaçlanmıştır. Artan ihtiyaçlarla ve yükselen teknoloji ile birlikte kimliklendirme sistemlerinden daha fazlası beklenmiş ve Radyo Frekansı Tanımlama sistemleri geliştirilmiştir.

RFID terimi yaklaşık olarak yetmiş yıldır hayatımızda olan bir terimdir. RFID sisteminin ilk kullanıldığı alan ikinci dünya savaşı olarak gösterilmektedir [1]. Burada, RFID yardımıyla hava araçları dost ve düşman olarak sınıflandırılmıştır.

RFID teknolojisi, doğrudan bir fiziksel teması ihtiyaç duymaksızın, etiket taşıyan bir cisim ya da kişinin kimliğinin, radyo dalgaları aracılığı ile doğrulanmasını sağlayan bir teknolojidir. Bir RFID sisteminin temel anlamda üç alt bileşeni bulunmaktadır. Bunlar, kimliği tanımlanacak olan etiket (tag), etiketleri tanımlayacak olan okuyucu (reader) ve bilgilerin depolandığı (server)'dır. Bazı durumlarda depolama ünitesi ile okuyucunun birleşik olabildiği görülmektedir.

RFID sistemlerinde etiketler çok kısıtlı kaynaklara sahip donanımlardır. Bununla birlikte etiketler çok düşük güç tüketimi ile çalışmalıdır. Tüm bu kısıtlar, RFID etiketlerinin güvenlik ve mahremiyet kriterlerini sağlayacak şekilde tasarlanmasını zorlaştırmaktadır. Güvenlik ve mahremiyet sağlaması için öne sürülen protokoller genelde özet(hash) fonksiyonları barındırmaktadırlar. Çoğu durumda, özet fonksiyonlarının teorik ve fiziksel güvenlik açıkları, zayıflıkları nedeni ile algoritmalar pratikte istenen mahremiyeti sağlayamaz. Bu çalışmada, bu zayıflıkları minimal düzeye indirmek için güncel anlamda SHA-3 finalistleri içinde, gömülü sistemlerde en hızlı çalışan adayı (Blake) tercih edilmiştir.

Hem maliyet hem de güç tüketimi bakımından aktif etiket tasarımında kullanılabilecek nitelikte bir mikrodenetleyici serisi olan Msp430, Texas Instruments

tarafından  retilmektedir. Bu Y ksek Lisans Tezi kapsamında yapılan  alıřmalarda etiket tasarımında Msp430 serisi mikrodeneleyicilerden yararlanılmıřtır. Okuyucu ise, y ksek performanslı olacak řekilde tasarlanmalıdır. Bu nedenden dolayı okuyucu tasarımında ST Microelectronics firmasının Stm32 ailesinden olan mikrodeneleyiciler tercih edilmiřtir.

Bu tez kapsamında tasarlanmıř olaran RFID sistemi aktif bir sistemdir. Yani etiket bir pil vasıtası ile  alıřmaktadır. Ayrıca haberleřme i in bir RF Modem kullanılmıřtır. Burada kullanılan RF modemini olabildi ince basit tutulması, hatta fiziksel anlamda daha fazla atak yapılabilmesine neden olur. Bununla birlikte sistemin maliyeti ise d řer. D ř k maliyetli ve basit bir yapıya sahip olan Rfm12b serisi RF modemler, bu tez  alıřması i in tercih edilmiřtir. HopeRF firmasının  retmiř oldu u bu modem ile haberleřen okuyucu ve etiket'te, fiziksel atakların engellenmesi ama lanmıř ve mahremiyet sa layan protokolumuz bu ortamda test edilmiřtir.

T m bu tasarım kıstasları d hilinde, hazırlamıř oldu umuz protokol n yapılan testlerin sonucunda mahremiyet sa ladı ını teyyid edilmiřtir.

2. ÇALIŞMA ALTYAPISI

2.1 Giriş

Geleneksel olarak özellikle tedarik zincirinde, kimliklendirme ve takip amacıyla barkod teknolojisi kullanılmıştır. Barkod teknolojisi yoğun olarak kullanılsa ve ihtiyacın bir kısmını giderse de, özellikle teknolojik anlamda çok sayıda limit barındırmakta ve sınırlı miktarda hareket alanı sunmaktadır.

Barkod teknolojisi, bilgiyi saklamanın düşük maliyetli ve düşük riskli bir methodu olarak ortaya atılmıştır. Öte yandan RFID vasıtası ile birçok nesneye ait bilgiler aynı anda, görüş açısı farketmeksizin toplanabilmektedir. Barkodların aksine RFID sistemlerinde çok sayıda protokol ve standard bulunmaktadır. Örnek olarak, RFID konusunda kullanılan bazı ortak frekanslar bulunmakla birlikte, çıkış güçleri, gürültü bağışıklıkları ülkeden ülkeye değişebilmektedir. Ek olarak barkodun aksine RFID sistemleri getirileriyle, özel haberleşme devrelerine, antene, bellek yönetim ve haberleşme protokollerine ihtiyaç duyarlar. Temas olmaksızın ve açı farketmeksizin kimlikleme imkânları sunan RFID teknolojisinde RFID etiketlerinin konumu hayati değilken, barkodlarda görüş açısı ve konum en büyük öneme sahiptir. Günümüzde özellikle gelişen teknoloji ve düşen etiket maliyetleriyle RFID yükselen bir teknoloji durumuna gelmiştir. Barkodlar ile RFID sistemlerinin karşılaştırıldı bir tablo Çizelge 2.1’de verilmiştir.

Çizelge 2.1 : Barkod ve RFID karşılaştırması [2].

Sistem Parametreleri	Barkod	RFID
Saklanabilir veri miktarı (byte)	1..100	16...128k
Okunabilirlik	Düşük	Çok yüksek
Yeniden yazılabilirlik	Yok	Var
Kir ve tozdan etkilenme	Çok etkilenir	Etkilenmez
Dış kaplamadan etkilenme	Çalışamaz	Etkilenmez
Maliyet	Çok düşük	Orta
Haberleşme menzili	50cm’ye kadar	10m’ye kadar
Güvenlik özellikleri	Yok	Yüksek
Çoklu okuma özelliği	Yok	Var

2.2 RFID Sistemleri

RFID, günümüzde kullanılan kablosuz otomatik kimliklendirme sistemlerinden biridir. RFID teknolojisi yardımıyla ürünler, insanlar, hayvanlar ve nesneler otomatik olarak tanımlanabilir, bunlar hakkında bilgiler toplanabilir ya da bu bilgiler doğrudan bilgisayar ortamına aktarılabilir. Günümüzdeki kimliklendirme sistemleri arasında en güvenilir olan sistem yine RFID'dir. RFID'nin kullanım alanları günümüzde çok genişlemiştir ve mevcut durumda insanlar tarafından fark edilmemektedir.

RFID teknolojisinin temelleri İkinci Dünya Savaşı'na dayanmaktadır. İkinci dünya savaşında radar teknolojisinin de yardımıyla yaklaşan uçaklar detekte edilebilmiş ancak tanımlanamamıştır. Almanlar, bu tanımlama problemini çözmek amacıyla uçaklarında radar sinyallerine cevap veren bir yapı kurmak istemişlerdir. Bu yapı aktif RFID yapısının ilk örneği olarak kabul edilir [2].

Britanya ise Almanya'nın bu hamlesine IFF'yi (Identify Friend or Foe) yaratarak cevap vermiştir [2]. Bu teknolojiye her bir uçağa bir transponder yerleştirilmiş ve yer istasyonundan alınan sinyale cevap gönderilmesi ile dost uçak tanımlaması yapılmıştır. RFID de bu temel kavram üzerine kurulmuştur. Etikete bir sinyal gönderildiğinde etiket uyanır ve uygun bir cevabı geri gönderir.

RFID sisteminin altyapısını oluşturan ve gerekliliğini sorgulayan ilk makale 1948 yılında Stockman tarafından yayınlanmıştır [3]. Bu makale aynı zaman ilk defa RFID tanımının yapıldığı yayındır. Bu sürecin ardından 1950'lerde ve 1960'larda RF haberleşme alanındaki gelişmeler de devam etmiştir. Bununla birlikte RFID teknolojisi ile ilintili birçok teknoloji geliştirilmiştir.

1960'ların sonlarına doğru RFID sistemlerinin elektromagnetik teori ile bağlantısı araştırılmış ve ilk RFID şirketi olan "Sensormatic and Checkpoint" kurulmuştur. Devam eden yıllarda RFID'ye olan ilgi iyice artmıştır. 1970'li yıllarda Los Alamos Ulusal Laboratuvarı'nda nükleer materyallerin takibine imkân veren ilk RFID etiketleri ve okuyucuları tasarlanmış ve üretilmiştir.

1980'li yıllarda RFID pratik kullanımdaki yerini arttırmıştır ve dünyada yaygınlaşmaya başlamıştır. Taşımacılık, lojistik, endüstri ve erişim izinleri gibi alanların yanında hayvanların takibinde de RFID teknolojisi öne çıkmıştır.

Devam eden yıllarda RFID teknolojisindeki ivme giderek artmıştır ve RFID günümüzdeki modern halini almıştır. Bu süreçteki en önemli katkılardan biri de IBM mühendislerinin ultra yüksek frekanslı (UHF) RFID sistemlerini tasarlaması olmuştur. UHF sayesinde daha uzun mesafelerde, daha hızlı veri aktarımı mümkün kılınmıştır [2].

RFID tabanlı sistemlerin ilk önemli patentlerinden biri Mario Cardullo [4] tarafından bulunmuştur ve bu patent Şekil 2.1’de görülebilir.

United States Patent [19]
Cardullo et al.

[11] **3,713,148**
[45] **Jan. 23, 1973**

[54] **TRANSPONDER APPARATUS AND SYSTEM**

[75] Inventors: **Mario W. Cardullo**, Rockville; **William L. Parks, III**, Bethesda, both of Md.

[73] Assignee: **Communications Services Corporation, Inc.**, Rockville, Md.

[22] Filed: **May 21, 1970**

[21] Appl. No.: **39,309**

[52] U.S. CL. 343/6.5 R, 343/6.8 R
[51] Int. Cl. G01s 9/56
[58] Field of Search 343/6.5 R, 6.5 LC, 6.5 SS, 343/6.8 R, 6.8 LC

[56] **References Cited**

UNITED STATES PATENTS

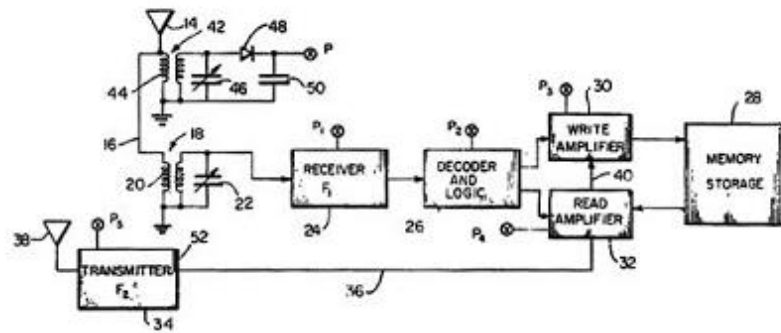
3,541,257 11/1970 McCormick et al. 343/6.5 LC X
3,144,645 8/1964 McIver et al. 343/6.5 R X

Primary Examiner—T. H. Tubbesing
Attorney—Jacobi, Lilling & Siegel

[57] **ABSTRACT**

A novel transponder apparatus and system is disclosed, the system being of the general type wherein a base station transmits an "interrogation" signal to a remote transponder, the transponder responding with an "answerback" transmission. The transponder includes a changeable or writable memory, and means responsive to the transmitted interrogation signal for processing the signal and for selectively writing data into or reading data out from the memory. The transponder then transmits an answerback signal from the data read-out from its internal memory, which signal may be interpreted at the base station. In the preferred inventive embodiment, the transponder generates its own operating power from the transmitted interrogation signal, such that the transponder apparatus is self-contained.

7 Claims, 3 Drawing Figures



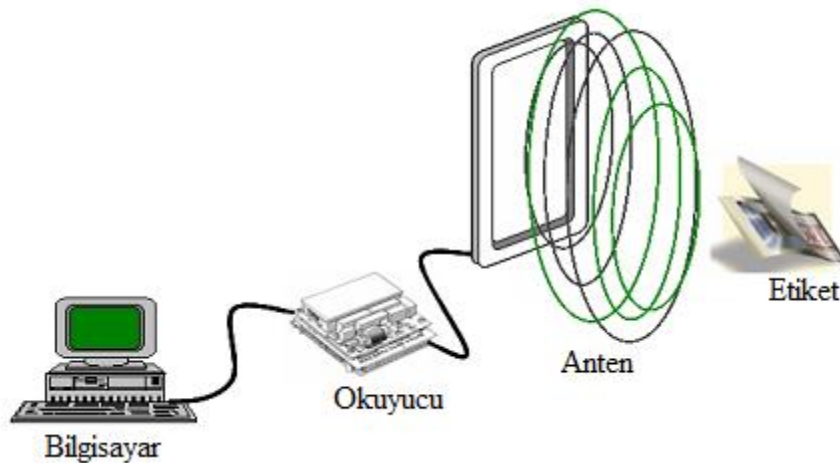
Şekil 2.1 : RFID patenti [4].

Daha önce de belirttiğimiz gibi RFID sistemleri üç ana bileşenden oluşmaktadır.

- RFID etiketi (tag, transponder), en temel anlamda üzerinde kimlik (tanımlama) bilgisini barındıran ünedir.

- RFID okuyucu (reader, transceiver), etiketten veri okuyan ya da etikete veri gönderen ünedir.
- RFID veritabanı (back-end database), etiket içeriği ile ilgili bilgileri depolayan ünedir.

Genel olarak standard bir RFID sistemi Şekil 2.2’de verilmiştir.



Şekil 2.2 : RFID sistemi [1].

2.2.1 Etiket

Bir RFID etiketi, özgün bir kimliğe sahip olan ve üzerindeki veriyi bir okuyucu isteği doğrultusunda radyo frekansında yayan bir tanımlama cihazıdır [5]. Etiket aynı zamanda transponder olarak da adlandırılabilir. Bu tez kapsamında basitlik ve yaygınlığından ötürü etiket terimi kullanılacaktır.

Bir etiketten çalışma esnasında beklenen, özel bir frekanstaki veriyi alıp otomatik olarak bir cevap mesajı göndermesidir. Etiket içerisinde minimal olarak bir entegre devre (Integrated Circuit, IC) ve bir anten yer alır. Entegre devrenin işlevi alınan veriyi işlemek, modülasyon ve demodülasyon işlemlerini yapmak ve gerekli bilgileri depolamaktır. Anten ise okuyucu ile radyo frekansında haberleşmenin sağlanması için gereklidir.

RFID sistemlerinde otomatik tanınması istenen her cisim üzerine fiziksel olarak bir etiket yerleştirilir. RFID etiketleri tipik olarak, depolama ve hesaplamaların yapıldığı bir mikroçip ile haberleşmeyi sağlayan bir anten elemanı içerir. Kimi etiketlerde ise akıllı kartlarda olduğu gibi bir temas alanı yer alır. Etiket üzerinde bulunan bellek alanı, kullanıma göre “yalnızca okunabilir”, “bir defa yazılıp – defalarca okunabilir”

ya da “tamamen okunup yazılabili” olabilir. Çok çeşitli etiket tipleri vardır. Etiket tiplerinden birkaçı Şekil 2.3’ten görülebilmektedir.



Şekil 2.3 : RFID etiketleri [1].

Etiketlerin sınıflandırılmasındaki en önemli faktörlerden biri etiketin kullandığı güç kaynağıdır. Bu bakımda etiketler; aktif, yarı-aktif ya da pasif olabilir. Aktif etiketler, etiketin içindeki bir güç kaynağı ile beslenirler. Bu besleme ünitesi bir batarya olabilir. Bu yapıdaki etiketler, haberleşmeyi diledikleri zaman başlatabilirler ve diğer etiketlerle de haberleşebilirler. Yarı aktif etiketler de üzerlerinde batarya gibi harici bir güç kaynağı taşırlar ancak yalnızca kendilerine gelen bilgileri cevaplayabilirler. Pasif etiketler ise gücün tamamını okuyucunun yaymış olduğu radyo dalgalarından alırlar ve okuyucu olmadığı sürece bir haberleşmeyi başlatamazlar.

Bir etiketin kullandığı güç kaynağı aynı zamanda onun veri aktarım mesafesini ve fiyatına etki eder. Pasif etiketler, üretmesi en ucuz etiketlerdir ancak en kısa haberleşme mesafesine sahip olan etiketlerdir. Benzer şekilde yarı aktif etikete ve aktif etikete doğru gidildikçe, üretim zorlukları ve maliyeti ile birlikte haberleşme mesafesi de artar. Üç etiket tipinin bu bağlamdaki karşılaştırması, Çizelge 2.2’de yer almaktadır.

Çizelge 2.2 : Aktif, yarı aktif ve pasif etiketler.

	Pasif	Yarı Aktif	Aktif
Güç Kaynağı	Pasif	Batarya	Batarya
Veri Aktarımı	Pasif	Pasif	Aktif
Aktarım Mesafesi	10 M	100 M	1000 M

Etiketleri, fonksiyonelliklerine göre de sınıflandırmak mümkündür. “MIT Auto-ID Center”, etiketleri fonksiyonelliklerine göre beş farklı sınıfa ayırmıştır [6]. Bu sınıflandırmaya ilişkin tablo gösterilimi ise Çizelge 2.3’de yer almaktadır.

Sınıf sıfır (Class 0) etiketler, en basit yapıya sahip olan ve yalnızca elektronik metin bilgisi (electronic article surveillance, EAS) sunan etiketlerdir. EAS etiketler yalnızca varlık bilgilerini gönderirler ve hiçbir özel kimlik verisi içermezler. Sıfır sınıf etiketler hiçbir sayısal ünite içermediklerinden maliyet bakımından çok ucuzlardır. Genel olarak kütüphanelerdeki kitaplarda ve kompakt disklerde bulunurlar.

Sınıf bir (Class 1) etiketler üzerlerinde özgün (unique) bir kimlik bilgisi depolarlar. Bellek bakımından ise “yalnızca okunabilen” ya da “bir kere yazılıp defalarca okunabilen” yapıdadırlar. Sınıf bir etiketler genel olarak pasiftirler ancak yarı aktif veya aktif olup sınıf bir fonksiyonelliği taşıyan etiketler de mevcuttur.

Sınıf iki (Class 2) etiketler okunabilir ve yazılabilir bellek taşımaktadırlar ve yapılan işlemlerle ilgili kayıt tutabilmektedirler. Bu etiketler yeniden konfigure edilerek farklı nesnelerin kimliklendirilmesinde tekrar tekrar kullanılabilirler. Yine sınıf iki etiketler, pasif, yarı aktif ya da aktif olabilirler.

Sınıf üç (Class 3) etiketler üzerlerinde kimi çevresel sensörler taşırlar. Bu tip etiketler bulundukları ortamdaki sıcaklık, ivme, hareket, basınç gibi büyüklükleri ölçebilirler. Tüm ölçümlerin, ortamda okuyucu olmasa da yapılabilmesi gerektiğinden sınıf üç etiketler yarı aktif ya da aktif olabilirler.

Sınıf dört (Class 4) etiketler diğer etiketlerle ağ kurabilirler. Bu tip etiketlerin haberleşmeyi başlatabilmeleri gerektiğinden, aktif olmaları zorunludur.

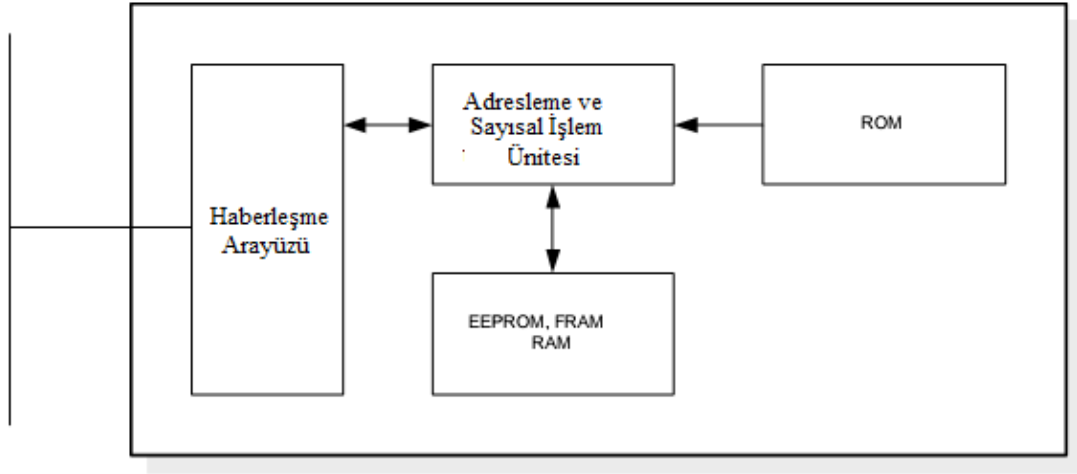
Çizelge 2.3 : Etiketlerin fonksiyonelliklerine göre sınıflandırılması

Sınıf	Hafıza	Beslenme	Özellikler
0	Yok	Batarya	Metin Saklama
1	Okunabilir	Herhangi	Kimliklendirme
2	Okunabilir/Yazılabilir	Herhangi	Veri Kaydı
3	Okunabilir/Yazılabilir	Aktif/Yarı	Çevresel Sensör
4	Okunabilir/Yazılabilir	Aktif/Yarı	Etiket Ağı

Bu tez kapsamında tasarımı yapılacak olan etiket sınıf iki bir etikettir.

Yapısal olarak etiketler, farklı donanım blokları içerebilmektedir. Bunlar arasında haberleşme ünitesi, sayısal işlem ünitesi ve bellek ünitesi bulunmak zorundadır.

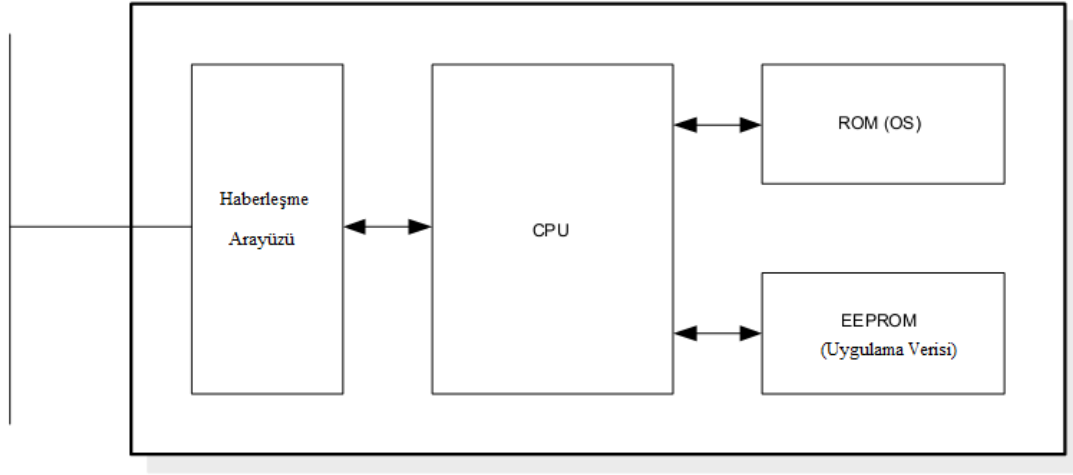
Standart olarak bellekli yapıdaki bir etiketin yapısı Şekil 2.4'te gösterilmiştir.



Şekil 2.4 : Bellekli RFID etiketi yapısı.

Yalnızca bellek barındarılan etiketler, salt-okunabilir etiketlerden kriptolojik işlemlerin yapılabildiği gelişmiş etiketlere kadar yaygın bir kullanım alanına sahiptir. Bellek olarak RAM, ROM, EEPROM ya da FRAM kullanabilirler. Haberleşme arayüzü, hem güç kaynağı olarak hem de haberleşme amacıyla kullanılır. Güvenlik fonksiyonları ve adresleme ise adresleme ve sayısal işlem ünitesi tarafından gerçekleştirilir.

Daha kompleks işlemler için etiket tasarımında sayısal işlem durum makinası yerine bir mikroişlemci de kullanılabilir. Opsiyonel olarak, işlemcinin yanına kriptografik sayısal üniteler gibi yan işlemci blokları da eklenebilmektedir. Bu sistemler mikroişlemci barındırdıklarından, yazılım bloğu barındırırlar. Kimi durumlarda mikrodenetleyici üzerinde küçük bir işletim sistemi de çalışabilmektedir. Burada işletim sistemin görevi, veri aktarımından veri yönetimi ve güvenlik fonksiyonlarının gerçekleştirilmesi işlerine kadar uzanır. Yine benzer şekilde hata deteksiyonu, fiziksel çıkış pinlerinin yönetilmesi, mesaj tamlığı yönetimi gibi konular da bu sayede yönetilebilir. Bu yüksek lisans tezi kapsamında yapılan çalışmalarda tasarlanan etiket içerisinde mikrodenetleyici barındırmaktadır. Bu sayede gelişmiş kriptorafik şemalar etiket üzerinde gerçekleştirilebilmektedir. Bu tür etiketlerin tasarımı genellikle bir güç kaynağı içerecek şekilde yapılmaktadır. Pil içeren bu tür etiketler aktif etiket adıyla sınıflandırılmaktadır. Yine bu yüksek lisans tezi kapsamında tasarlanan etiket aktif bir etikettir. Mikroişlemci barındıran etikete ilişkin blok diyagram ise Şekil 2.5'te yer almaktadır.



Şekil 2.5 : Mikroişlemcili RFID etiketi yapısı.

2.2.2 Okuyucu

RFID sistemlerinde etiket okuyucuları, etiketleri RF kanalı üzerinden sorgularlar. Ek imkânlar sunabilmek adına, tasarım esnasında RFID okuyuculara dâhili ek bellek, güç kaynağı, işlemci ünitesi ya da veritabanı bağlantısı eklenebilir. Ayrıca bir takım kriptografik hesaplama işlemleri de yine okuyucu üzerinde yapılabilir.

Okuyucudan etikete olan haberleşme kanalı genellikle “ileri” kanal olarak adlandırılır. Benzer şekilde etiketten okuyucuya olan kanal ise “geri” kanal olarak adlandırılır. Etiket ve okuyucu arasındaki etkileşimler ileriki kısımlarda daha geniş biçimde ele alınacaktır.

Pratik kullanımda RFID okuyucular elde taşınan mobil cihazlar ya da sabit şekilde konumlandırılmış cihazlar olabilir. Stok yönetimlerinde kullanılan okuyucular genellikle mobil okuyuculardır. OGS gibi sistemlerde ise okuyucunun konumu sabittir. Örnek bir RFID okuyucusu Şekil 2.6’da yer almaktadır.

Okuyucular kendilerinden beklenen fonksiyonelliğe göre birkaç USD’den yüzlerce USD’ye kadar maliyete sahip olabilir. Her durumda okuyucular etiketlerden çok daha karmaşık yapıdadırlar. Pasif etiketler söz konusu olduğunda, pasif etiketler beslenmek için ihtiyaç duydukları enerjiyi okuyucunun yaydığı enerjiden karşılarlar. Özellikle kriptografik işlemleri yerine getirecek olan okuyucular görece daha gelişmiş donanım imkânlarında sahip olmaktadır.



Şekil 2.6 : RFID okuyucu.

2.2.3 Veritabanı

Okuyucular arka plandaki veritabanı sorgusu için, sorgu anahtarı olarak genellikle etiketlerden aldıkları bilgiyi kullanır. Arka plandaki veri tabanı bu sorgu anahtarını ürün bilgisi, takip kayıtları gibi bilgilere erişmek için kullanır. Okuyucu genellikle etiketten yalnızca kimlik bilgisini okur ve bu kimlik bilgisini birincil anahtar olarak kullanarak veri tabanına sorgu yapar. Veritabanı bu anahtarın temsil ettiği satırı alır ve okuyucuya geri gönderir. Bu sayede okuyucu bu satırda bulunan ve etiketi temsil eden tüm bilgilere ulaşır. Etiket bilgilerine erişmek için birden fazla bağımsız veritabanı da kullanılabilir. Bu sayede etiket bilgilerini kullanan birden fazla uygulama geliştirilebilir.

Her zaman okuyucu ile veritabanı arasındaki bağlantının güvenli olduğu varsayılır. Protokol analizlerinde bu bağlantının azami düzeyde tümleşik olduğunu anlatmak için genel olarak okuyucu ve veritabanı aynı çerçevenin içine çizilir ve öyle temsil edilir. Aksi durumda veritabanına ulaşabilecek bir başka kanal vasıtasıyla etiket verilerinin kontrolsüz biçimde elde edilmesi ya da yönetilmesi problemleri doğar.

Birçok durumda etiketin taşıdığı verinin anlamlı olmasını sağlayan şey veritabanıdır. Veritabanının varlığı sayesinde etiketler üzerinde kimlik bilgisinden başka bir bilgi saklanmaz. Etiketler belleklerinde işaretçiler, rastgele kimlikler ya da şifrelenmiş veri barındırabilir. Bu veriler kullanılarak yeni bir veritabanı yaratılabileceği gibi, genelde etiket bilgilerinin hali hazırda yer aldığı bir veritabanı üzerinde işlemler yapmak daha

iyi bir çözümdür. Yani sistem kurulumu esnasında veri tabanı ilgili etiket bilgilerini içerecek şekilde hazırlanmalıdır.

2.3 RFID Sistemlerinin Sınıflandırılması

RFID sistemleri çok farklı şekillerde gerçekleşmiştir ve çok çeşitlidir. RFID sistemlerini inceleyebilmek adına, var olan farklı RFID sistemlerinin bazı spesifik niteliklerini inceleriz. Bu nitelikler arasında çalışma modu, çalışma frekansı, kuplaj veya bilgi işleme yetenekleri olabilir. Yine benzer şekilde, modülasyon tekniği, bilgi miktarı ve güç kaynağına göre de bir farklılaşma söz konusu olabilir.

RFID sistemleri genellikle üç parametreye göre sınıflandırılır: çalışma frekansı, menzil ve kuplaj. RFID sistemleri için çalışma frekansları 135kHz bandından 5.8 GHz mikrodalga frekansına kadar uzanmaktadır. Fiziksel kuplaj ise, elektriksel, manyetik ya da elektromanyetik olabilmektedir. Buna göre birkaç milimetreden onlarca metreye kadar menzil sağlanabilir.

Düşük menzilli yani 1cm'ye kadar menzile sahip olan RFID sistemleri genellikle yakın kuplajlı RFID sistemleri olarak adlandırılırlar. Bu durumda etiket genellikle okuyucunun içine sokulmalı ya da okuyucuya dokunacak kadar çok yaklaştırılmalıdır. Yakın kuplajlı sistemlerde fiziksel kuplaj elektriksel ya da manyetik olabilir ve 30MHz'e kadar herhangi bir frekans bu işlem için kullanılabilir. Bu senaryoda etikete çok büyük miktarda enerji aktarılabilir. Yakın kuplajlı sistemler genellikle güvenlik ihtiyacının büyük olduğu durumlarda kullanılırlar. Örnek bir senaryo olarak kapı kilitleri ve kredi kartları gösterilebilir. Mevcut durumda yakın kuplajlı RFID sistemleri için ISO 10536 standardı bulunmaktadır.

Okuma yazma mesafesi 1 metreye kadar çıktığında sistemler uzak kuplajlı sistemler olarak tanımlanmaktadır. Bu sistemlerde büyük çoğunlukla endüktif ya da manyetik kuplaj kullanılmaktadır. Kimi zaman kapasitif ya da elektriksel kuplaj da kullanılabilir. Bu senaryo için ISO 14443 ve ISO 15693 gibi birçok standard bulunmaktadır. Uzak kuplajlı RFID sistemlerinde frekans 13.56 MHz'in altındadır.

Mezilin 1 metrenin ötesinde olduğu RFID sistemleri ise uzun menzilli RFID sistemleri olarak adlandırılırlar. Tüm uzun menzilli RFID sistemleri elektromanyetik alanları kullanırlar ve UHF ya da mikrodalga frekansında çalışırlar. Bu sistemlerin çalışma frekansları genellikle UHF bandında Avrupa için 868 MHz, Amerika için ise

915 MHz, mikrodalga bandında ise 2.5 GHz ve 5.8 GHz frekanslarında olmaktadır. Pasif, pilsiz sistemlerde bu senaryoda haberleşme mesafesi 3 metreye kadar sağlanabilmektedir. Aktif sistemlerde ise bu menzil oldukça artmaktadır.

Bu bağlamda frekans tabanlı sınıflandırmanın genel anlamda gösterimi Çizelge 2.3te yer almaktadır.

Çizelge 2.4 : RFID sistemlerinin sınıflandırılması [1].

	LF	HF	UHF	Mikrodalga
Frekans	125-134kHz	13,56Mhz	868-915MHZ	2,45-5.8GHZ
Okuma menzili	1m'ye kadar	1m'ye kadar	5m'ye kadar	15m'ye kadar
Okuma hızı	Yavaş	Orta	Hızlı	Çok hızlı
Metalden Etkilenme	Negatif	Negatif	-	-
ISO standardı	11784/85 14223	11443 115693 18000	15693 18000	18000

her geçen gün daha yaygın olarak kullanılmaktadır. Buna bağlı olarak da her geçen gün yeni sorunlar karşımıza çıkmaktadır. RFID sistemlerinde karşımıza çıkan en önemli problemler ise güvenlik ve mahremiyet problemleridir.

2.4 RFID Sistemlerinde Güvenlik ve Mahremiyet

RFID sistemler her geçen gün daha yaygın olarak kullanılmaktadır. Buna bağlı olarak da her geçen gün yeni sorunlar karşımıza çıkmaktadır. RFID sistemlerinde karşımıza çıkan en önemli problemler ise güvenlik ve mahremiyet problemleridir.

RFID sistemlerinde oluşan bu güvenlik ve mahremiyet sorunları neticesinde kriptografinin en yaygın kullanım alanlarından biri de RFID olmuştur. RFID için kriptografi söz konusu olduğunda genellikle özet fonksiyonlarına ve rastgele sayı üreticilerine ihtiyaç duyulur.

RSA gibi asimetric kriptografik tekniklerin RFID etiketlerine uygulanması günümüzde maalesef çoğu durumda mümkün olmamaktadır. Bunda, RFID etiketlerin son derece sınırlı kaynaklara sahip olması da etkilidir. Bunun yerine RFID sistemlerinde genellikle simetrik kriptografi teknikleri kullanılmaktadır. Önceki yıllarda standard simetrik kriptografik algoritmalar (SHA), etiketlerde gerçekleştirilebilir durumda olmasa da, günümüzde etiketlerin ve mikroelektronik teknolojisinin gelişimiyle bu mümkün hale gelmiştir.

Genellikle güvenlik (security); kimlik doğrulama (authentication), gizlilik (confidentiality) ve ulaşılabilirlik (availability) üçlü bir sorun olarak sunulur. RFID sistemlerde güvenlik söz konusu olunca tüm haberleşme katmanları, sistemin güvenlik seviyesinin belirlenmesinde etkili olur. İleriki kısımda bu kavramların her birine değinilecektir.

Mahremiyet de RFID sistemlerin kullanımının yaygınlaşmasıyla ortaya çıkan önemli bir problemdir ve günümüzde bu problem halen tam olarak çözölememiştir.

2.4.1 Güvenlik

RFID sistemlerinde güvenlik önceden de belirtildiğı üzere üç alt başlıkta incelenir. RFID etiketleri, kimi durumlarda kullanıcılarının bir sisteme giriş yapabilmesini sağlar. Kimlik doğrulama sürecinin ardından kullanıcı ilgili sisteme giriş hakkı kazanır. Erişim kontrolü, araç immobilizer modülü, toplu taşımalarda e-bilet gibi uygulamalar bu fonksiyonelliğın mevcut gerçeklemelerindendir. Kimlik doğrulama sisteminin sağladığı güvenlik seviyesi elbette ki bu süreçte hayati öneme sahiptir. RFID sistemlerinin güvenlik seviyeleri sistemin özelliklerine oldukça bağımlıdır.

RFID etiketleri, radyo haberleşmesi ile çalışırlar. Radyo kanalı saldırılara açık bir kanaldır ve bu hatlar üzerinde tekrarlama atakları (replay attacks) yapmak oldukça basittir. Sistemin bu yapısı itibarı ile bir etiketi sabote etmek, o etiketle herhangi bir fiziksel temas kurmadan da mümkündür.

Ekonomik olarak düşük fiyatlı etiketler üzerlerinde çok fazla güvenlik önmeli barındırmazlar ve görece basit bir yapıya sahiptirler. Bu nedenden dolayı eğer etikete fiziksel erişim mümkünse etiket üzerindeki veriler kolayca elde edilebilmektedir. Farklı etiketlerin taşıdığı bilgileri toplamak ve bunlar arasında ilişki kurmak ise, sistemin birçok dinamiğı hakkında da bilgi verdiğinden güvenlik problemi kullanıcı seviyesinden, sistemin tamamına yayılabilmektedir.

Tüm bu nedenlerden ötürü güvenlik konuları, izinsiz erişimlerin önlenmesi ve etiketin taşıdığı yahut temsil ettiğı bilgilere erişilememesini içerir. Güvenlik ve mahremiyet kavramları aslında iç içe olan kavramlardır.

2.4.2 Mahremiyet

RFID etiketleri, önceden de belirtildiği üzere nesneleri, varlıkları veya insanları tanımlamak, kimliklendirmek amacıyla kullanılır ve bu tanımlama işlemi bilgi dâhilinde olmalıdır. RFID'nin sağladığı bu tanımlama imkânı vasıtasıyla bilinmeyen kaynaklar etiketin bulunduğu nesne ya da insanla ilgili birçok bilgiye ulaşabilir. Bu sayede etiketin kullanıcısının alışkanlıkları hareketleri hatta kimliği, kullanıcının isteği dışında deşifre edilebilir. Bu problem RFID sistemlerinde mahremiyet problemi olarak bilinir. Sonuç olarak RFID sistemlerinde, kullanıcı mahremiyetinin ihlal edilmesi mutlaka önlenmelidir. RFID sistemlerinde mahremiyetin korunması amacıyla önemli bilgiler şifrelenmeli ya da gizlenmelidir.

Mahremiyet kavramı yalnızca kullanıcının kimlik vb. bilgilerinin izinsiz olarak elde edilmesi anlamına gelmez. Bir çok etiketin bulunduğu bir ortamda belirli bir etiketin, bilinmeyen kaynaklarca izlenebilir olması da büyük bir problemdir. Etiket izlenmesi durumunda kullanıcının hareketleri ve kimi alışkanlıkları izlenebilir duruma gelmektedir ve bu da özel hayatın gizliliği ilkesini ihlal etmektedir.

RFID sistemlerin kullanım alanlarının yaygınlaşabilmesi için mahremiyet problemi muhakkak suretle çözülmelidir. Mahremiyet probleminin çözülebilmesi için yalnızca etiket tarafında değil, okuyucu hatta veri tabanı kısmında da önlemler alınmalıdır.

RFID sistemleri temel anlamda iki sınıf tarafından kullanılır. Birinci sınıf, sistem yöneticisinin bulunduğu aktif sınıftır. Aktif sınıf, bilgi, veri ve verinin kullanılışı üzerinde tam kontrole sahiptir. Sistem yöneticisi aynı zamanda etiketleri ve üzerlerindeki bilgiyi de yönetmektedir. Diğer taraftan, müşterilerin ya da çalışanların içinde bulunduğu pasif sınıf vardır ve genellikle pasif sınıf etiketin üzerindeki bilgi ve bu bilginin kullanılış şekli üzerinde bir etkiye sahip değildir.

Aktif sınıf açısından bakınca, RFID sisteminin tüm fonksiyonlarını yerine getirerek çalışması birinci önceliktedir. Pasif sınıf açısından bakıldığında ise, bu yeni teknolojinin getirdiği avantajların dezavantajlardan fazla olması beklentisi bulunmaktadır. Pasif sınıf, sistemin çalışma şekli ve olası açıkları konusunda bilgi sahibi olamamaktadır. Özellikle müşteri koruma organizasyonları, RFID sistemlerinin kişilerin özlük haklarını ihlal edebileceği konusunda şüpheleri bulunmaktadır. Bu özlük hakları içerisinde en çok tehdit altında olan ise mahremiyet

kavramıdır. Bu bakımdan Simson Garfinkel, bir RFID kullanım hakları bildirgesi yayınlamıştır [7]. Bu maddeler aşağıdaki gibidir:

1. Kullanılan ürünlerde RFID etiketi olup olmadığının bilinmesi hakkı.
2. Bir ürün satın alındığında RFID etiketlerinin kaldırılması ya da deaktive edilmesi hakkı.
3. Etiket taşımak zorunda olmaksızın, RFID içeren sistemleri kullanma hakkı.
4. Bir RFID etiketin içerisindeki veriye ulaşma hakkı.
5. Etiketlerin nerede ne zaman ve ne için okunacağını bilinmesi hakkı.

Mahremiyet tehdidi kullanıcı kitleri olarak pasif sınıf için söz konusu olmaktadır. Pasif parti, doğrudan etiketleri ya da üzerinde etiket taşıyan ürünleri, etiketlerdeki bilgiler üzerinde herhangi bir kontrole sahip olmaksızın kullanmaktadırlar. Bu sebeple pasif sınıfın mahremiyeti, aktif sınıf ya da üçüncü parti kimseler tarafından ihlal edilebilir durumdadır. Aktif sınıf veri üzerinde tam denetime sahip olduğundan mahremiyet tehdidi oldukça kısıtlıdır ve yalnızca üçüncü parti kimselerin oluşturduğu bir tehdit söz konusudur. Bununla birlikte aktif sınıf, istediği takdirde mahremiyet kurallarını bilinçli olarak ihlal ederek mahrem bilgileri üçüncü partilere dağıtabilir. Bu senaryo RFID güvenliğiyle alakalı değildir ve bu tezin doğrudan kapsamında değildir. İkinci bir olasılık olarak üçüncü parti, pasif sınıfın bilgilerine doğrudan erişmeye çalışabilir ya da, pasif sınıfın mahremiyetini doğrudan ihlal etmeye çalışabilir. Bu iki durumun pasif sınıf açısından sonuçları hemen hemen aynıdır çünkü pasif sınıfa ait bilgiler bu kimselerin isteği ya da bilgisi dışında alınmıştır.

Mahremiyet kavramı kavramsal olarak birçok alt başlık barındırır ve geniş senaryolarla tanımlanabilir. Ancak temel anlamda RFID sistemlerindeki mahremiyet tanımları, veri mahremiyeti ve konum mahremiyeti olarak ikiye ayrılabilir.

Eğer bir RFID etiketi kişisel bazı bilgileri içeriyorsa, pasif sınıfın mahremiyeti aşağıdaki yollarla ihlal edilebilir durumdadır:

1. Havadaki bilgileri yakalayarak ya da izinsiz okuma yaparak, bir saldırgan kişinin mahrem bilgilerine ulaşabilir.
2. Kişinin bilgilerinin doğrudan elde edilmesi yerine, potansiyel bazı yan bilgilerin elde edilmesi de saldırgan tarafından elde edilebilir. Bu bilgiler anonim bilgiler olabilir fakat yüksek bir olasılıkla bir kişiyi işaret edebilir.
3. Sanal olarak saklanan bilgilerden, kişinin isteği ya da bilgisi dışında gerçek hayatta yer alan bazı kurallar işletilebilir. Buna örnek olarak kişinin isteği olmaksızın nüfus cüzdanı kontrolünün polis tarafından yapılması gösterilebilir.

Diğer taraftan pasif sınıftaki etiket kullanıcıları, genellikle etiketleri uzun dönemler boyunca kullanırlar. Bu durumun bir sonucu olarak saldırgan, etiketler üzerindeki sabit bilgiyi takip ederek kurban hakkında bir hareket profili çıkarabilir. Bu hareket profili kişinin bir anlamda takip edilmesi, izlenmesi olarak düşünülebilir. RFID yaygınlaştıkça bu ihtimal daha korkutucu bir hal almaktadır. Konum mahremiyeti tanımı tanım itibarıyla kişisel bilgilerden konum dışındakilerle ilgilenmez. Çünkü kişilerin takip edilmesi başlı başına ciddi bir problemdir ve bu tezin temel ilgi alanlarından birini oluşturmaktadır.

RFID tabanlı izleme her ne kadar sürekli bilgi akışı sağlamasa da, diğer RF tabanlı konum belirleme yöntemlerine göre çok daha yüksek doğrulukta çalışırlar çünkü okuyucu ile etiketin arasındaki mesafe çok kısıtlıdır ve okuyucu konumu tam olarak bilinmektedir. Buna ek olarak RFID sistemi ile izleme, okuyucu ile interaksyon tipi gibi bilgileri de verdiği için birçok ek bilgi çıkarımına da imkân sunar. Örneğin eğer bir kimse aracıyla seyahat ediyorsa ve otomatik geçiş sistemi kullanıyorsa, bu kişinin konumu izlenebilir durumda olduğunda otomatik olarak araçla hareket ettiği bilgisine de ulaşılmış olur. Bu da başlı başına bir mahremiyet problemidir.

3. ÖZET FONKSİYONLARI

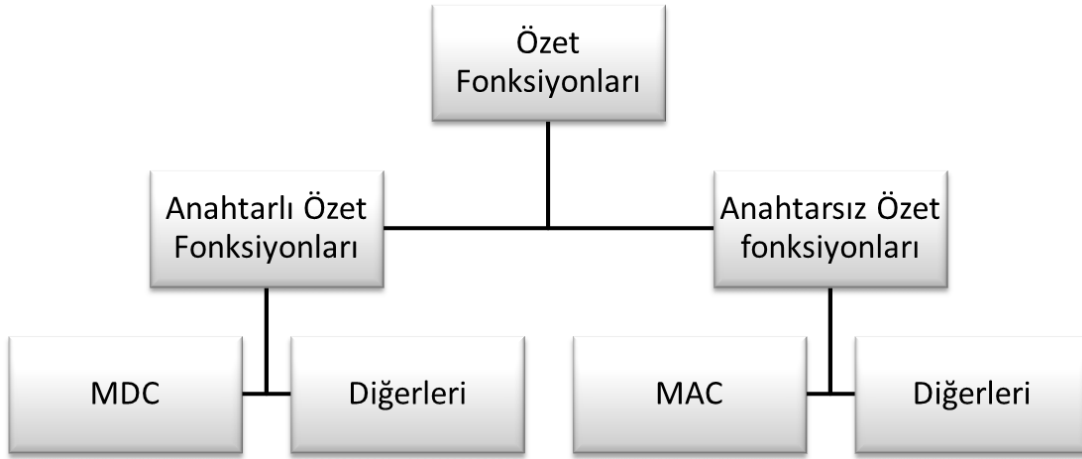
Genel Bilgi

Kriptografik özet fonksiyonları birçok uygulamada önemli bir role sahiptir. Dijital imza, veri tamlık kontrolü, otantikasyon protokolleri, şifre koruma, rastgele sayı üretimi, bu uygulama alanlarından bazılarıdır. Sonuç olarak kullanılan özet fonksiyonunun güçlü ya da zayıf yanları, uygulama alanlarını da doğrudan etkilemektedir. Uzun seneler boyunca, SHA-1, SHA-2 ve standart olmayan MD-5 özet fonksiyonu algoritmaları birçok uygulamada kullanılmıştır. Buna rağmen kriptografik ataklar, özellikle de Wang ve arkadaşlarının atakları [8, 9, 10, 11] ve gelişen teknoloji daha güvenli ve yeni özet fonksiyonlarına olan ihtiyacı doğurmuştur. Bu sebepten dolayı NIST (National Institute of Standards and Technology), herkese açık olan ve bir sonraki özet fonksiyonu standardının seçileceği SHA-3 yarışmasını başlatmıştır. Son durumda bu yarışmayı finale kalan beş algoritmadan Keccak kazanmış ve yeni standard özet fonksiyonu olmuştur.

Kriptografik özet fonksiyonları değişken uzunluklu bir giriş verisini alarak sabit uzunluklu bir çıktı üretirler. Pratikte giriş verisinin uzunluğu her zaman sınırlıdır ancak giriş verisinin azami boyutu oldukça büyük bir değer olmaktadır. Bu sayede her giriş verisinin özet değeri tek seferde bulunabilmektedir. Özet fonksiyonunun çıktısı, “özet değeri (hash value, digest value)” , “mesaj özet (message digest)” gibi terimlerle isimlendirilebilmektedir. Özet değerinin boyutu algorithmadan algoritmaya değişebilmektedir ancak genellikle 128 bit ile 1024 bit arasındadır. Eğer özet fonksiyonu n – bit çıktı üretiyorsa bu fonksiyon n -bit hash olarak adlandırılır.

Özet fonksiyonları, giriş parametrelerine göre Şekil 3,1’deki gibi adlandırılabilir. Bazı özet fonksiyonları, bir çıktı üretmek için yalnızca giriş verisine ihtiyaç duyarken bazıları giriş verisiyle birlikte bir parametreye daha ihtiyaç duyarlar. Tek parametrelili özet fonksiyonlarına anahtarsız özet fonksiyonları denir. Bu tür fonksiyonlar aynı zamanda *Manipulation (or Modification) Detection Codes*

(MDC) ya da *Message Integrity Codes (MIC)* olarak da adlandırılabilir. Adlarından da anlaşılacağı üzere tek parametrelili özet fonksiyonları yalnızca mesaj tamlığı ile ilgilenir [12]. Mesajın tamlığı, mesajın kaynaktan gönderildiği şekliyle, eksiksiz ve değişiklik olmaksızın alınması anlamını taşımaktadır. Diğer bir özet fonksiyonu tipi ise, giriş verisi ile birlikte ikinci parametre olarak bir anahtar verisini de alan, anahtarlı özet fonksiyonlarıdır. Buradaki anahtar verisi gizlidir. Bu tür özet fonksiyonları *Message Authentication Codes (MAC)* olarak da adlandırılmaktadır. MAC tipi özet fonksiyonları anahtarı da kullanarak mesajdan bir özet değeri yaratırlar. Bunun sonucunda oluşturulan özet değeri, mesajla birlikte anahtar hakkında da bilgi taşır. Bunun sonucunda MAC tipi özet fonksiyonları sadece veri tamlığı ile ilgili değil, kimlik doğrulama işleminde de katkı sağlar [12].



Şekil 3.1 : Özet fonksiyonunun sınıflandırılması

MDC tipi özet fonksiyonlarının en önemli örneklerinden biri de tek yönlü özet fonksiyonlarıdır (one way hash functions). Tek yönlü fonksiyon f , temel olarak zamanda tersi alınamayan bir fonksiyondur. Eğer $y = f(x)$ ise herhangi bir durumda y 'yi kullanarak x 'i bulmak mümkün olmamalıdır. Bu tanımdan yola çıkarak tek yönlü özet fonksiyonu H olmak üzere $H(M) = y$ denklemi ile mesajdan (M), özet değeri (y) bulunabilirken, özet değerinden (y) mesaj (M) bulunamamaktadır. Daha resmi bir tanımla, H fonksiyonu aşağıdaki koşulları sağlaması durumunda tek yönlü bir özet fonksiyonudur:

- M değişken uzunlukta olabilir ancak, fonksiyonun çıktısı sabit uzunlukta olmalıdır.

- H özet fonksiyonunun çıktısı olan y verildiğinde, M 'yi bulmak çok zor olmalı. Ek olarak verilen bir M_1 için, $H(M_1) = H(M_2)$ denklemini sağlayan bir M_2 'yi bulmak çok zor olmalı.

Zorluk koşulu, özet fonksiyonun çıkış uzunluğuna göre tanımlanmaktadır. n – bit'lik bir özet fonksiyonu için yukarıdaki iki madde 2^n özet fonksiyonu hesaplamasından daha az adımda bulunamamalıdır. Tanımlanan bu gereklilikler sağlanmalıdır ancak, özet fonksiyonları daha fazla özelliği de sağlayabilir. Örnek olarak korole olmama, yakın çakışma direnci, tümleme özgürlüğü gibi özellikler de, bir özet fonksiyonundan beklenen özellikler olabilmektedir [12].

3.1 SHA-3 Finalistleri

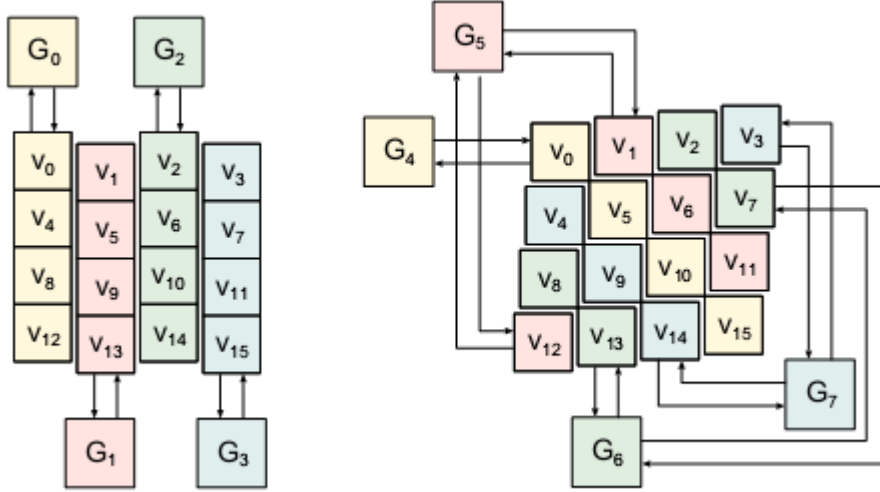
2007 yılının Nisan ayında NIST, yeni özet fonksiyonu standardını belirlemek için SHA-3 yarışmasını duyurmuştur [13]. İlk aşamada kabul edilen 64 başvuru 15'e düşürülmüş ve ilk SHA-3 konferansının ardından devam eden aylarda finale kalan 5 finalist duyurulmuştur. Nihayetinde, SHA-3 yarışmasını finalistlerden Keccak kazanmıştır. Özet fonksiyonları bu tez çalışmasında önemli bir yer tutmaktadır ve bu kısımda gerçekleştirilen özet fonksiyonları tanımlanacaktır. Tarif edilen tüm özet fonksiyonlarının çıkışları 256-bit seçilmiştir.

3.1.1 Blake

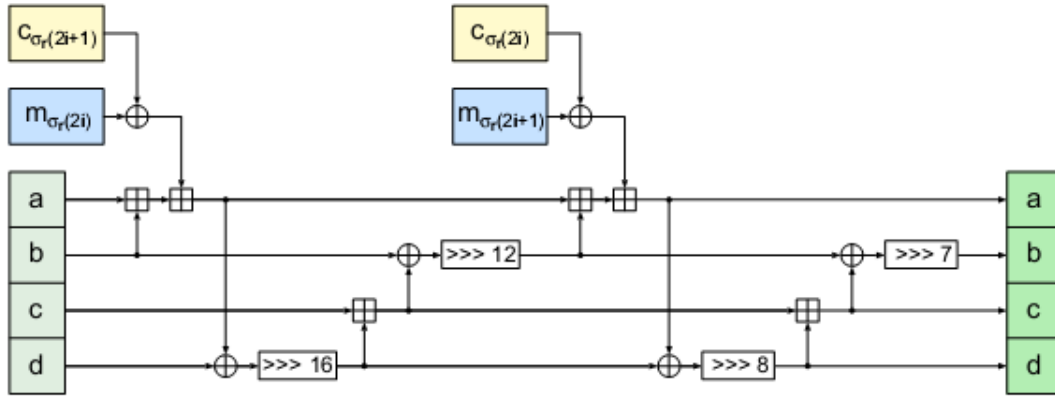
Blake, 32-bitlik bloklar üzerinde çalışan iteratif bir özet fonksiyonudur. Blake fonksiyonu, çekirdeğinde AES benzeri bir yapıdadır. Blake'de kullanılan sıkıştırma fonksiyonu ardı adına çağırılan birçok fonksiyon çağırısından oluşan, 10 tur devam eden, sütun adımları ve çapraz adımlardan oluşan bir fonksiyondur. Bu fonksiyonun çekirdeğindeki sütun adımlarının ve çapraz adımların yapısı Şekil 3.2'deki gibidir.

Her bir G_i bloğunun iç yapısı ise Şekil 3.3'teki gibidir. Şekil 3.3'te görülen a,b,c,d değerlerinden her biri, özet fonksiyonunun giriş verisinin 8-bit'lik bileşenleridir [14].

İterasyonların ardından, çıkışa gönderilecek olan değişken XOR işlemleri ile 32-bit'lik boyuta genişletilir.



Şekil 3.2 : Sütun adımı ve diagonal adım [14].

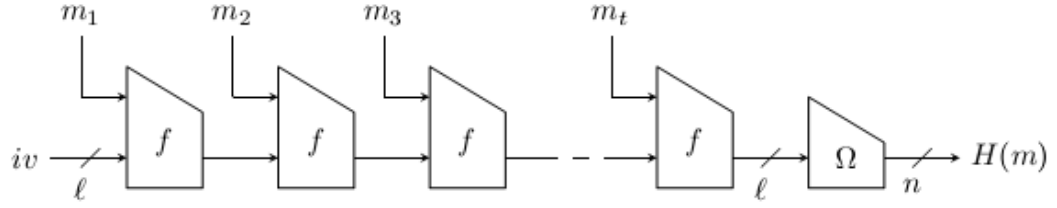


Şekil 3.3 : G_i bloğunun iç yapısı [14].

3.1.2 Grøstl

Grøstl, sıkıştırma fonksiyonu Şekil 3.4'te gösterildiği üzere iki ayrı permütasyondan oluşan iteratif bir özet fonksiyonudur. Bu iki permütasyon bloğu, geniş deneme (wide trail) tasarım stratejisi ile oluşturulmuştur ve bu sayede algoritmanın çeşitli kriptografik saldırılara karşı dayanıklılığı kolayca ifade edilebilmektedir. Ancak algoritmada bu permütasyonlar ideal kabul edilmiştir [15]

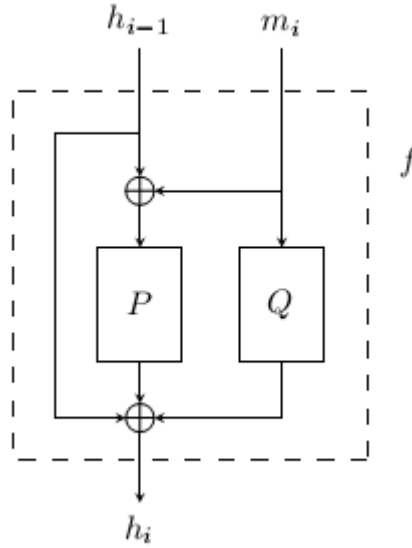
Şekil 3.4'te gösterilen iv değişkeni giriş vektörü, f sıkıştırma fonksiyonu, m_i ise mesaj bloklarıdır. f ile gösterilen sıkıştırma fonksiyonunun iç yapısı ise Şekil 3.5'te gösterilmiştir.



Şekil 3.4 : Grøstl özet fonksiyonu

Grøstl ile AES arasında birçok benzerlik bulunmaktadır. Örneğin Grøstl’de kullanılan S-box, AES’te kullanılan ile birebir aynıdır. AES’ten aldığı güçle Grøstl güvenliliğine oldukça güven vermektedir.

Grøstl algoritmasında iç durum (internal state) değişkeninin iki dalı, P ve Q adı verilen birbirlerine benzer iki dönüşümden geçirilir. Yine bu dönüşümler AES benzeri yapıdadır ve 512-bit’lik bloklar üzerinde çalışır. P ve Q işlemleri 10 tur boyunca uygulandıktan sonra, P’nin ve Q’nun çıktıları bir önceki iç durum ile XOR işlemine tabi tutularak yeni çıkış elde edilir.

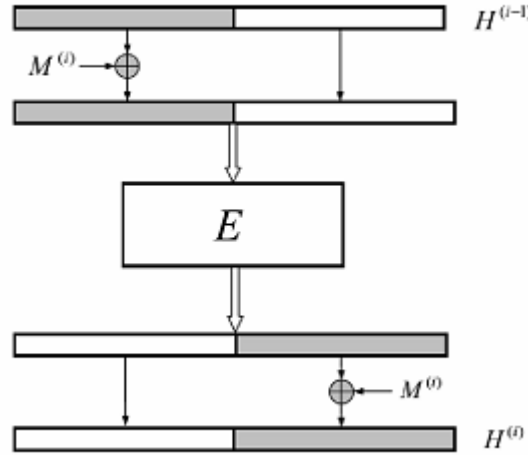


Şekil 3.5 : f sıkıştırma fonksiyonu

3.1.3 Jh

Jh, 1024 bitlik geniş bir iç durum değişkeni kullanan iteratif bir özet fonksiyonudur. Bu algorithmada, sıkıştırma fonksiyonu olarak bir blok şifreleme algoritması kullanılmaktadır. Jh da tasarımında AES’ten birçok alanda esinlenmiştir ve tasarım metodolojisi olarak AES’i genelleştirmeye çalışmıştır.

Jh'nin sıkıştırma fonksiyonunda, öncelikle mesaj, iç durum değişkeninin yarısı ile XOR işlemine tabi tutulur. Ardından bu durum 35.5 tur boyunca, E olarak adlandırılan bir bloktan geçer. En sonunda elde edilen çıktı, iç değişkenin ikinci yarısı ile XOR işlemine tabi tutulur.



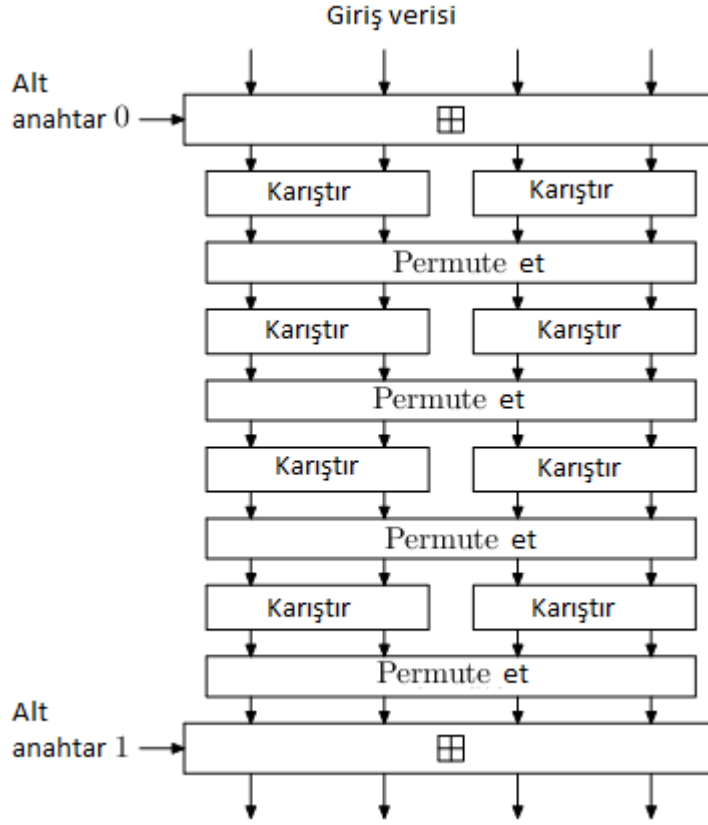
Şekil 3.6 : JH sıkıştırma fonksiyonu

Sürece ilişkin gösterim ise Şekil 3.6'da yer almaktadır [16]

3.1.4 Skein

Skein algoritması, sıkıştırma fonksiyonu olarak Threefish algoritmasını kullanmaktadır. Threefish algoritması basit bir SPN (substitution-permutation network) yapısındadır ve 512 bitlik bloklar üzerinde çalışır. Algoritma yapısal olarak yalnızca basit işlemsel operatörleri kullanır. Bunlara örnek olarak bit kaydırma, XOR işlemi ve 2^{64} modüler toplama işlemleri gösterilebilir. Threefish algoritmasının oldukça basit, hızlı ve kullanışlı olmasından dolayı, yazarlar her bir mesaj bloğu için 72 turluk bir şifreleme önermişlerdir. Skein-256 algoritmasındaki sıkıştırma algoritmasının 72 turundan 4 tanesi Şekil 3.7'de gösterilmektedir [17].

Şekil 3.7 kısaca incelendiğinde, dört tur boyunca karıştırma ve permute etme işlemlerinin yapıldığı görülebilmektedir. Burada karıştırma ifadesi ile anlatılmak istenen giriş verisinin byte düzeyindeki alt parçalarının yer değiştirmesi işlemidir. Permute etme ifadesi ile anlatılmak istenen ise byte düzeyindeki bir alt parçanın bitlerinin belirli bir düzenle yer değiştirmesidir. Bu kavramlar kriptografik sıklıkla kullanılan kavramlardır ve Skein algoritmasında da bu şekilde kullanılmışlardır.



Şekil 3.7 : Skein algoritmasının sıkıştırma fonksiyonun 4 turu

3.1.5 Keccak

Keccak, sünger (sponge) yapısında tasarlanmış bir özet fonksiyonu ailesidir [18]. Keccak, temel yapı bloğu olarak, 7 permütasyon kümesinden oluşan bir permütasyon bloğu kullanır. Sünger fonksiyonları, özet fonksiyonları içerisinde bir genelleştirmeyi ifade etmektedir. Bu genelleştirmede çıkış boyutu sonsuz olarak ayarlanabilmektedir. Keccakta kullanılan 7 permütasyon fonksiyonu $KECCAK - f[b]$ olarak adlandırılmaktadır ve bu ifadede $b = 25, 50, 100, 200, 400, 800, 1600$ değerlerini alabilmektedir. SHA-3 yarışmasına sunulan versiyonda $b=1600$ olan en geniş permütasyon öne sürülmüştür. Keccak'ta kullanılan operatörler yalnızca XOR, AND, NOT ve rotasyon operatörleriyle sınırlı tutulmuştur. Keccak 2 Ekim 2012'de SHA-3 yarışmasını kazanmış ve yeni SHA-3 standardı olmuştur. [19]

4. KONUM MAHREMİYETİ SAĞLAYAN RFID PROTOKOLÜ

1948 yılında “İnsan Hakları Evrensel Bildirgesi’nde” belirtildiği üzere, insanın mahremiyetinin korunması kişinin özlük haklarından biridir. Bu hakların RFID sistemlerinde ihlal edilme ihtimali bulunmaktadır ve kişilerin özlük hakları gereği bu ihtimal ortadan kaldırılmalıdır.

Bu amaçla yapılan çalışmalardan biri, seri numarayı etiketin üzerinden kaldırıp yalnızca üretici kimliğine ve ürün tipine yer vermektir. Bu durumda dahi etiketleri izlemek mümkün olmaktadır.

Daha karışık bir yaklaşım ise hash (özet) kilidi olarak adlandırılmaktadır. Bu düzende etiket, gönderdiği özet(hash) değerine, okuyucudan gelen doğru anahtarı alana dek üzerindeki bilgileri göndermez. Bu iletişim şemasında etiket üzerinde hash fonksiyonlarının gerçekleştirilmesine ihtiyaç duyulmaktadır. Aynı zamanda bu şemada anahtarlar arka planda yönetilmelidir. Bu yaklaşımın veri mahremiyeti sağlamasına rağmen konum mahremiyeti sağlamaması önemli bir problem ve dezavantajdır [20,21].

Daha gelişmiş bir sürüm olarak “rasgeleleştirilmiş hash kilidi” şeması ortaya atılmıştır [21]. Bu şema konum güvenliği sağlamaktadır ancak, yapısal olarak şema çok fazla sayıda etiketli sistemler için uygun değildir çünkü arka planda çok sayıda özet fonksiyonunun çalışmasına ihtiyaç duymaktadır. Ek olarak bu şemada bir gerçek sayı üreticisine ihtiyaç duyulmaktadır. Bu durumda da gerçekleştirme, zorunlu olarak daha karmaşık ve daha pahalı olmaktadır.

4.1 Protokol Tanımı

Bu kısımda, konum mahremiyeti sağlayan şema anlatılacak ve ileriki bölümlerde değinilecek olan haberleşme protokolünde yer alan bazı tanımlar yapılacak, kısaltmalar belirlenecektir. Şemanın tanıtılmasının ardından bu pratik durumda çok etiketli senaryolara da uygun şekilde şemanın nasıl gerçekleştirildiği anlatılacaktır.

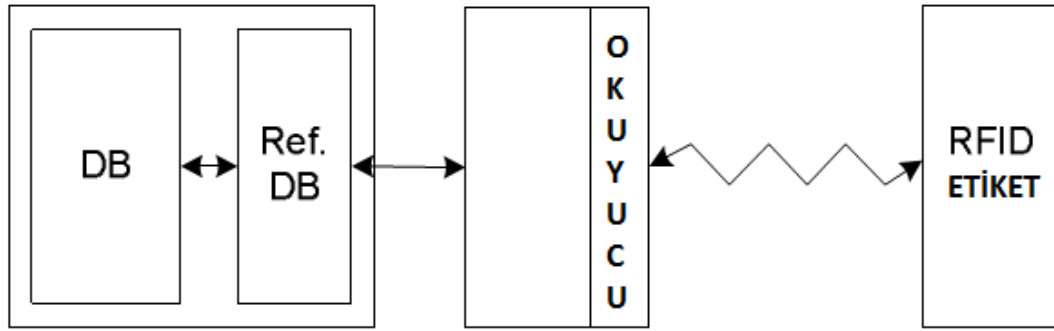
Gerçeklenen protokol “Hash Lock” (karma kilit) ile aynı önkoşula dayanarak, sadece veri gizliliği değil aynı zamanda konum gizliliğini de sağlayan bir yapıdadır. Genel anlamıyla protokolün ana fikri, her okuma girişiminde etiket kimliğini güvenli bir şekilde değiştirmektir. Güvenlikten anlaşılması gereken, dinleme, sızdırma, modifikasyon, yeniden gönderme saldırıları ve “ortadaki adam saldırısı” (man-in-the-middle attack) girişimlerine taviz verilmemesidir. Diğer yaklaşımların aksine, veri etiketten okuyucuya doğru taşındığında, okuyucu/üçüncü şahsın hiçbir şekilde güvenilir olması gerekmez. Belirtilen amaçları gerçekleştirmek için kullanılan yöntemler tek yönlü özet (hash) fonksiyonları, karşı saldırıları engellemeye yarayan işlem tanımlayıcıları ve rastgele sayı üreticilerinin yanı sıra işlem-geçişi (transaction) sayılarının her adımda farklı oluşuyla, haberleşme verileri etiket izlenmesinde kullanılamazlar. Büyük miktarda veri saklama ve hesaplama işlemleri etiket maliyetini arttırdığı için tüm bunlar okuyucu tarafına taşınmalıdır. Protokol mesaj kayıplarına da karşı esnektir.

$y = h(x)$ ‘i kriptografik tek yönlü özet fonksiyonu olarak tanımlayalım. İdealde, fonksiyonun tersinin alınmasının zor olmasının yanında, y çıktısı kullanılarak x ile ilgili önemli bir bilgi ya da öngörü ortaya atılamamaktadır. Özet fonksiyonları ile ilgili bu bilgi, genellikle zaten matematiksel kanıt olmadan da geçerli kabul edilir zira standart olarak kabul edilen güvenlik fonksiyonları bu kuralı sağlamak zorundadır. Bu sebeple maksimum güvenlik için, “güvenli anahtarlı özet fonksiyonları” veya “K-özet (K-hash) fonksiyonları” düşünülmelidir. Ancak pratikte, standart özet fonksiyonları bilgiyi yeterince gizleyebilmektedirler. Önerilen yöntemde anahtar kaynağı olarak, sınırlı geçerliliğe sahip tanımlayıcılar kullanılmaktadır. RFID-etiketindeki kapı sayısı maliyeti düşürmek için olabildiğince küçük tutulmalıdır ki bunun için de uygun bir donanım yapısı gerekmektedir. Özet fonksiyonunun yanında uygun bir birleştirme (conjunction) fonksiyonu $y = \text{conj}(a, b)$ gereklidir. Burada bahsedilen işlevi gerçekleyen “ $\circ$ ” işaret ise şu şekilde gösterilebilir: ($y = a \circ b$). Basit bir XOR fonksiyonu bu amaç için uygundur. Her bir RFID etiketinin aşağıdaki girişler için alan içermesi gereklidir:

- Veritabanı-tanımlayıcı (“DB-ID”): salt okunur, isteğe bağlı yazılabilir
- Şu anki ID (“ID”): okunur/yazılır
- İşlem numarası (“TID”): okunur/yazılır

- Son başarılı işlem numarası: (“LST”) : okunur/yazılır
- Kullanıcı verisi ya da ana anahtar için ek veriler de olabilir; fakat her durumda zorunlu değildir.

DB-ID etiketten sorumlu veri tabanının kimliğini temsil eder. Fakat etiket sahibini gizlemek için, bu veri tabanı kimliği, etiketin üreticisi ya da bağımsız bir firma tarafından sağlanmalıdır. Etiket sahibinin tüm istekleri veritabanına bu kimlik üzerinden iletilmelidir. Ayrıca gerekirse güvenlik için ara katmanlarda “gizlilik temsilcileri” (privacy proxies) kullanılabilir. Yöntemin kendisine odaklanmak adına, üreticinin veri tabanı “DB” olarak gösterilecektir. İlgili şema Şekil 4.1’de yer almaktadır.

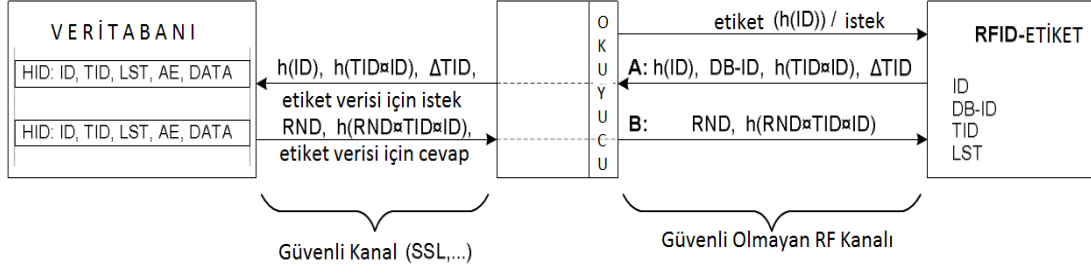


Şekil 4.1 : Sistem şeması.

DB her bir satır için aşağıdaki girdileri tabloda tutmalıdır:

- İlgili kimliğin (ID) özet değeri (hash value). Bu bilgi tablonun birincil indeksidir ve erişim anahtarıdır.
- Şu anki kimlik (ID)
- Son işlem numarası (TID)
- Son başarılı işlem numarası (LST)
- Veri tabanına bağlı girdi (AE)
- Etiket verisi/kullanıcı verisi (DATA)

İşlem numarası ve AE tekrarlı ataklarla mücadele içindir. Bunların işlevleri ileride anlatılacaktır. Haberleşme şemasının detaylı bir gösterimi ise Şekil 4.2’de yer almaktadır.



Şekil 4.2 : Detaylı sistem şeması.

Protokole göre başlangıç esnasında haberleşme aşağıdaki değerlerle başlatılır:

DB-ID'ye etiketten sorumlu veritabanı kimliği yüklenir. ID'ye rastgele bir sayı, TID ve LST'ye aynı rastgele sayılar atanır. Veri tabanında eş satırlar oluşturulur, yani satır kopyalanır. Yeni satırdaki ID, etiketin ID'si olarak atanır. Birincil satıra $h(ID)$ atanır. TID ve LST alanları etiketin TID/LST değerlerini alır. AE ise baştan set edilmez.

Bir etiket standart ikili ağaç yürüme ya da anti-çarpışma protokolü gibi yöntemlerle tekilleştirilebilmektedir. Başlangıç yapılandırmasının tamamlanmasının ardından etiket kimliğinin özet çıktısı ($h(ID)$) ile etiket sorgulanır. Bu işlem etiketi tanımlama ve adresleme için kullanılır. Sorgulandığında, etiket TID'sini bir artırır ve şu bilgileri gönderir: $h(ID)$, DB-ID, $h(TID \circ ID)$ ve ($\Delta TID = TID - LST$). Bu mesajda $h(ID)$ DB-ID'deki etiketi tanımlar. $h(TID \circ ID)$ tekrarlı ataklar için gereklidir. Her okuma teşebbüsünde kontrol edilir ve gerçek okuyucu tarafından kontrol edilir. Ayrıca etiketi de doğrular. ID ile TID zorunlu değildir; fakat TID için kullanılan bit sayısı düşükse, kullanılması oldukça yararlı olmaktadır. ΔTID , gerçek alıcı tarafından güncel TID'yi hesaplamak için kullanılır. Bu değer bir olması, etiketi izlemeye çalışan biri olmadığı konusunda bilgi vermektedir.

Etiketden gelen bilgiler okuyucunun hiç bir işine yaramadığı için, sonrakinin mesajı DB-ID ile gösterilen veri tabanına iletmesi gerekmektedir. Veri tabanında, $HID=h(ID)$ olarak atanır. LST ve ΔTID 'ler biri birine eklenir. Böylece güncel TID^* elde edilir. Bu durumda sonraki adım olarak $h(TID^* \circ ID)$ hesaplanır. Bu değer ilk mesajdaki ($h(TID \circ ID)$) ile eşleşmezse mesaj iptal edilir. Eğer mesaj geçerli ise TID^* ile TID karşılaştırılır. TID^* , TID'den büyük değilse mesaj yine iptal edilir.

TID* ve TID hakkında her şey yolunda gitti ise bir sonraki aşamaya geçilebilir. Bu aşamada rastgele bir sayı "RND" üretilecektir. Bu rastgele numara ile yeni ID (ID^*), $ID^* = RND \circ ID$ olacak şekilde tanımlanır. İlgili DB girdisi AE oluşturulmuşsa,

kaydın ID alanı ID^* olarak, HID alanı ise $h(ID^*)$ olarak güncellenir. Aksi halde, yeni bir satır ID^* ID ve $h(ID^*)$ HID olacak şekilde oluşturulur. Satırın AE girdisi ilgili alanları günceller ya da tam tersi durum gerçekleşir. TID aynı değerleri alır.

Bir sonraki aşamada, RND ve $h(RND \circ TID \circ ID)$ 'yi içeren mesaj oluşturulur ve etikete iletecek olan veritabanından okuyucuya yollanır.

Etiket $h(RND \circ TID \circ ID)$ değerini kontrol eder. Yanlış mesaj ise iptal eder ve başka mesaj kabul etmez. Aksi halde etiket, ID'yi $(RND \circ ID)$ 'nin değerine günceller. Ek olarak LST'yi TID olarak günceller. Artık etiket yeni ID ve $h(ID)$ değerlerine sahip olur ki bundan sonraki okuma girişimlerinde kullanılabilir.

İlk mesajın içeriğini anahtar olarak kullanarak, $(h(ID), DBID, h(TID \circ ID), \Delta TID)$, okuyucu veri tabanına almak istediği bilgiler için sorgu atar. Keyfi erişimler okuyucunun hak iddia edebileceği ihtimali olduğundan sınırlanır.

Konum güvenliği sağlayan protokolün işleyişi bu şekildedir.

4.2 Protokolün Analizi

Öne sürülen şema, simetrik veya asimetrik oluşundan bağımsız olarak birçok sistemde kullanılabilecek, güvenlik seviyesi yüksek bir şemadır. Şemanın ortaya koyduğu en büyük fayda, sunduğu basitliktir. Şema, yalnızca etiket kısmında bir özet fonksiyonuna, arka planda ise veri yönetimine ihtiyaç duyar. Birçok atağa karşı dayanıklı olan şema kuvvetli bir konum mahremiyeti sunmaktadır.

Şema tarafından sunulan önemli avantajlardan biri ise, etiket ve okuyucu arasındaki haberleşme kanalının güvenli olmak zorunda olmamasıdır. Aynı şekilde protokolde, üçüncü parti cihazların da güvenli olması zorunluluğu bulunmamaktadır. Ek olarak, anahtar gibi herhangi bir bilginin de uzun süre boyunca tutulması zorunlu değildir.

Tüm bunların yanında öne sürülen şema mevcut hali ile gerçeklemeye uygun değildir ve birçok teorik kabul içermektedir. Ayrıca verilerin aktarış şekli ile ilgili de herhangi bir öneri içermez. Bu bağlamda protokol şemasında haberleşme ile ilgili hemen her şey ideal kabul edilmiştir. Protokolün gerçekleşmesi için kimi değişiklikler gerekmektedir ve ileriki kısımlarda gerçekleştirme bölümünde bunlara değinilecektir.

.

5. ÖZET FONKSİYONLARININ GERÇEKLENMESİ

Bu tez kapsamındaki temel hedef, konum mahremiyeti sunan bir protokol gerçekleştirilmesi yapmaktır. Ancak konum mahremiyetinin sağlanması için ortaya sunulan protokol şeması özet fonksiyonlarına ihtiyaç duymaktadır ve bu nedenle gerçekleştirilen sistemin başarısı özellikle özet fonksiyonunun başarısına büyük oranda bağlıdır. Bu bağımlılık sebebiyle, protokol çalışması yapılmadan önce kullanılacak özet fonksiyonunun seçimi için oldukça detaylı bir çalışma yapılmıştır.

Bu kısımda gerçekleştirilen özet fonksiyonları ile ilgili geniş bilgi verilecek ve sonuçlar analiz edilecektir.

5.1 Kullanılan Donanım Platformları

Tez kapsamında yapılan çalışmada, RFID sisteminin gerçekleştirilmesi adına bazı donanım platformları kullanılmıştır. Bu donanım platformları temel anlamda, RF modem ve mikrokontrolör geliştirme kitleridir. Bu donanım platformlarının tamamı bu kısımda incelenecektir.

5.1.1 Rfm22b

Gerçekleme için en basit yapıda olan yalnızca ham veri gönderen RF modem tipi kullanılmıştır. Bu projede kullanılan RF modem Rfm22b'dir ve modem 433 MHz taşıma bandında çalışacak şekilde konfigüre edilmiştir. Modeme ilişkin bir fotoğraf Şekil 5.1'deki gibidir.



Şekil 5.1 : RFM22B modülü.

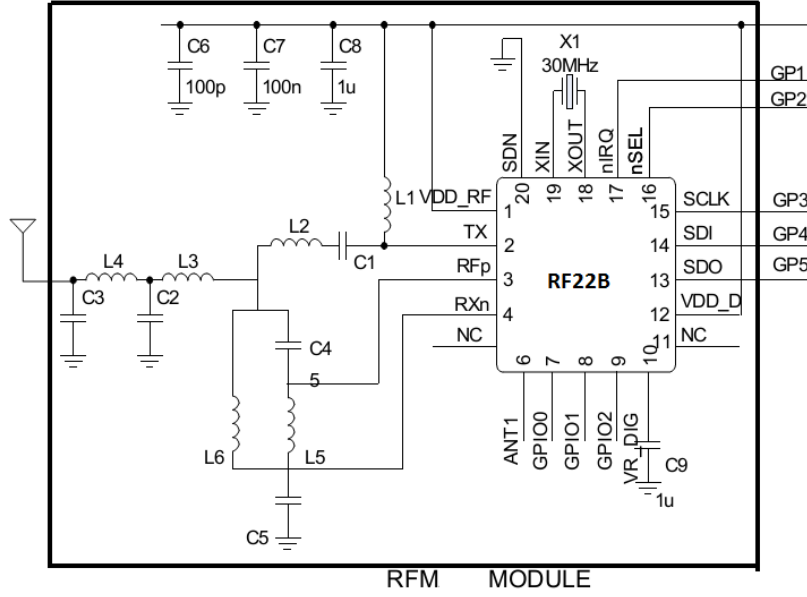
Rfm22b, HopeRF firmasının sunduğu düşük maliyetli bir RF alıcı/verici modülüdür. 433/470/868/915MHz frekans bölgelerinde haberleşme imkanı sunan bu modemler, basit yapılarından ve düşük maliyetli oluşlarından dolayı seçilmiştir.

Konum mahremiyeti sağlayan protokol, hem etiket hem de okuyucu tarafında iki yönlü veri alışverişi gerektirmektedir. Bu nedenle her iki kısımda da alıcı verici işlemleri gerçekleştirilmelidir. Ayrıca bu modemin çıkış gücü azami +20 dBm'dir ve bu da geniş mesafeden haberleşme imkânı sunmaktadır. Ayrıca bu özellik, kısa mesafede haberleşme kalitesinin yüksek olmasını, yani doğru veri aktarım oranının artmasını da sağlamaktadır.

Rfm22b'nin RF haberleşme işlevleri dışında da birçok işlevi bulunmaktadır. Uyku modu, otomatik uyanma, zamanlayıcı, düşük pil durumu detektörü, FIFO bellek, dâhili sıcaklık sensörü, analog-dijital çevirici ve giriş çıkış pinleri gibi birçok fonksiyon ve ünite de dâhili olarak Rfm22b'de bulunmaktadır. Her ne kadar bu işlevler, protokol gerçekleştirmesinde kullanılmamış olsa da, fiziksel ataklar neticesinde alınacak önlemlerin bazılarında kullanılmaları muhtemeldir.

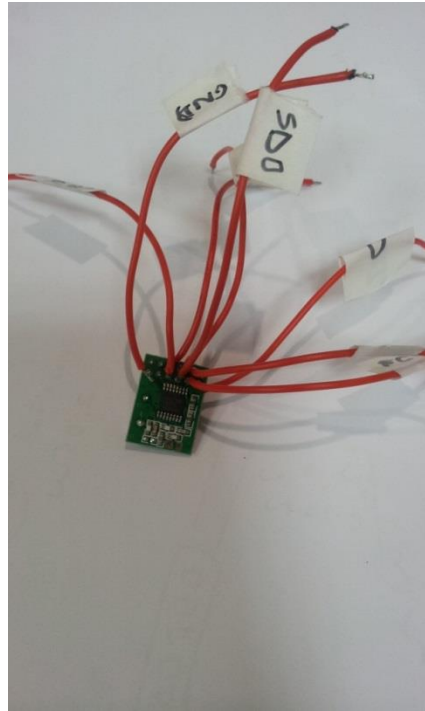
Zigbee protokolü ile çalışan vb. RF modemlerin aksine Rfm22b oldukça basit bir yapıya sahiptir ve fiziksel birçok problemin görülmesine olanak sağlar. Veri aktarım hataları, ağ kurulum sorunları gibi birçok durum bu modemlerle gözlemlenebildiğinden ve otomatik olarak çözülmediğinden, bu problemlere de özgün çözümler üretebilmek adına bu basitlikte bir modem seçilmiştir.

Rfm22b modülü, kontrolör, kuvvetlendirici ve anten gibi üniteleri içinde barındırmaktadır. Rfm22b modülünün blok diyagramı Şekil 5.2'den görülebilir.



Şekil 5.2 : Rfm22b modülünün blok diyagramı.

Gerçekleme için, Rfm22b modülünün pin çıkışları kablolar ile uzatılmıştır. Uzatılan bu kablolar çeşitli mikrokontrolörlerin ilgili pinlerine bağlanmış ve haberleşme sağlanmıştır. Bu haberleşme için hazırlanan Rfm22b modülünün bir fotoğrafı Şekil 5.3'te verilmiştir. Bu modüllerden iki adet hazırlanmıştır.



Şekil 5.3 : Gerçeklemede kullanılan Rfm22b modülü.

5.1.2 Msp430

Aktif etiketin gereklenmesinde Texas Instruments firmasının Msp430 mikrodeneleyicisi kullanılmıřtır. Ultra dūřuk gū tūketimi, dūřuk maliyeti gibi sebeplerden dolayı bu seim yapılmıřtır.

Msp430 mikrodeneleyicileri, Texas Instruments tarafından ūretilen 16-bit, RISC tabanlı, ultra dūřuk gū tūketimli bir mikrodeneleyici ailesidir. Basit tasarımıyanı sıra, zengin evreselleri, kullanım kolaylıęı, dūřuk maliyeti ve ok dūřuk gū tūketimi ile Msp430 piyasaya sūrūlmesiyle birlikte binlerce uygulamada kullanılmıř ve yeni ufuklar amıřtır.

Msp430, uyku modunda 0.1uA'den daha az akım tūketmektedir ve bu performansı ile dūřuk gūlū iřlemci aileleri iinde nemli bir yer edinmiřtir. Msp430 25MHz'e kadar kristal osilatr desteklemektedir. Ek olarak 128B ile 64 KB arasında RAM, 512B ile 521KB Flash bellek bulunduran modelleri mevcuttur. Bu anlamda bellek boyutları anlamında Msp430 ailesi ok geniř bir yelpazede seenekler sunmaktadır. Aynı zamanda Msp430 ailesi 14 pinden 113 pine kadar, otuza yakın kılıfta satılmaktadır. Bu da kullanıcıyla tasarım alanı anlamında ciddi anlamda ok sayıda alternatif sunmaktadır. MSP430 ailesinden bir mikrokontrolrūn fotoęrafı řekil 5.4'de yer almaktadır.



řekil 5.4 : MSP430 mikrokontrolrū.

Msp430 iin, Code Composer Studio ya da IAR Embedded Workbench ortamlarında yazılım geliřtirmek mmkündür. Bu tez kapsamında yapılan alıřmalarda Code Composer Studio geliřtirme ortamı kullanılmıřtır. Msp430 kitleri, aynı zamanda JTAG programlama ūnitesi de ierdięinden, bařka bir programlayıcı ya da hata ayıklama ūnitesi kullanılmamıřtır.

Bu tez kapsamında yapılan çalışmalarda Msp430 LaunchPad kiti kullanılmıştır. Kite ilişkin bir fotoğraf Şekil 5.5'te yer almaktadır.



Şekil 5.5 : Msp430 Launchpad kiti.

Msp430 LaunchPad kiti, temelde Msp430g2xx serisi mikrodeneleyiciler için tasarlanmış olup, hemen her Msp430 üyesi ile birlikte kullanılabilir durumdadır. MSP430 LaunchPad kiti üzerinde, programlama ve hata ayıklama donanımını barındırmaktadır. Kit üzerinde genel amaçlı birçok giriş/çıkış pini, LED'ler, Msp430g2xx ailesi için bir dip soket ve 2 adet Msp430 işlemci barındırmaktadır.

5.1.3 Stm32

Aktif okuyucu birçok etiket ile yaklaşık olarak aynı anda haberleşebilecek seviyede olmalıdır. Bu nedenle aktif etiket Stm32 platformu üzerinde gerçekleştirilmiştir. Stm32'nin temel bazı özellikleri aşağıdaki gibidir:

- İşlemci olarak ARM 32-bit Cortex M3 barındırır
- İşlemcisi 1.25 DMIPS/MHz ile çalışır
- 72MHz azami saat frekansında çalışabilir
- Tek saat darbesinde çarpma ve bölme işlemi yapar
- Azami 512 kilo byte Flash bellek barındırır
- Azami 64 kilo byte SRAM barındırır

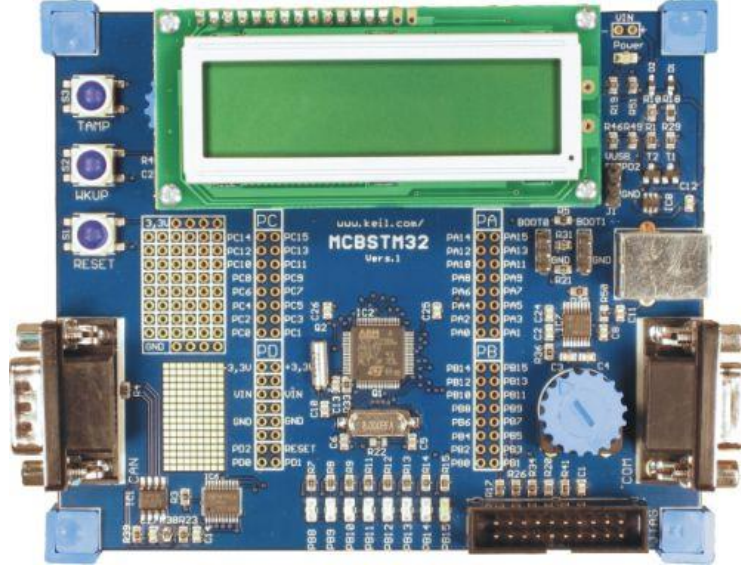
- 2.0 – 3.6 V arası gerilimle beslenebilir
- 4-16 Mhz arası harici osilatör takılabilir
- Dâhili olarak 8 MHz'lik bir RC osilatörü barındırır
- Dâhili olarak 40 kHz'lik bir RC osilatör barındırır
- Birçok düşük güç modunu destekler
- 3 adet 12-bit ADC barındırır
- Azami 112 giriş çıkış pini barındırır
- Azami 12 adet zamanlayıcı barındırır
- 4 adet zamanlayıcı barındırır
- 2 adet I2C ve 5 adede kadar USART modülü barındırır
- Azami olarak 3 adet SPI barındırır
- CAN ve USB modülleri barındırır

Stm32 işlemcisine ilişkin bir fotoğraf Şekil 5.6'te görülebilir



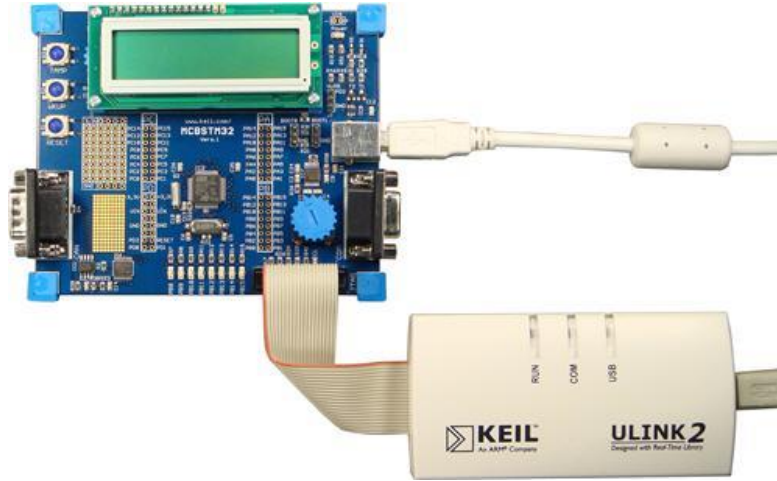
Şekil 5.6 : Stm32 mikrokontrolörü.

Bu tez kapsamında yapılan çalışmalarda Stm32 ailesinin Stm32f103rbt6 işlemcisi kullanılmıştır. Bu işlemci Stm32 ailesi içerisinde orta sınıfta yer almaktadır. İşlemcinin kullanıldığı donanım platformu Keil firması tarafından tasarlanmış bir deneme kiti olan Mcbstm32'dir. Mcbstm32 kitine ilişkin bir fotoğraf Şekil 5.7'te yer almaktadır.



Şekil 5.7 : Mcbstm32 geliştirme kiti.

Mcbstm32 kiti programlama ve hata ayıklama ünitesi olarak JTAG/SWD modülü içermektedir. Bu kit, birçok programlama ünitesi ile programlanabilmektedir. Bu tez kapsamında yapılan çalışmalarda, programlayıcı olarak Keil uLink2 kullanılmıştır. Mcbstm32 kitinin uLink2 ile programlanışına bir örnek Şekil 5.8’de yer almaktadır.



Şekil 5.8 : Keil Ulink2 ve Mcbstm32 geliştirme kiti.

5.1.4 Atmega128

Atmel firması tarafından üretilen Atmega serisi mikrodeneleyiciler sektörde en çok kullanılan mikrodeneleyicilerden biridir. Atmel firmasının ürettiği 8-bit Atmega serisi mikrokontrolörler de akademide kriptografik implementasyonların sıklıkla yapıldığı platformlardan birisidir. Bu popülerliği sebebiyle de 8-bit mikrokontrolörler içinde Atmega serisi en iyi seçim olmaktadır. Proje dahilinde yapılan

gerçeklemelerde, özet fonksiyonlarının performanslarının 8-bit veriyoluna sahip mikrokontrolörlerde görülmesi gerekmektedir ve bu amaçla Atmega serisi seçilmiştir. Atmega128 mikrodeneleyicisinin bir fotoğrafı Şekil 5.9’da yer almaktadır.



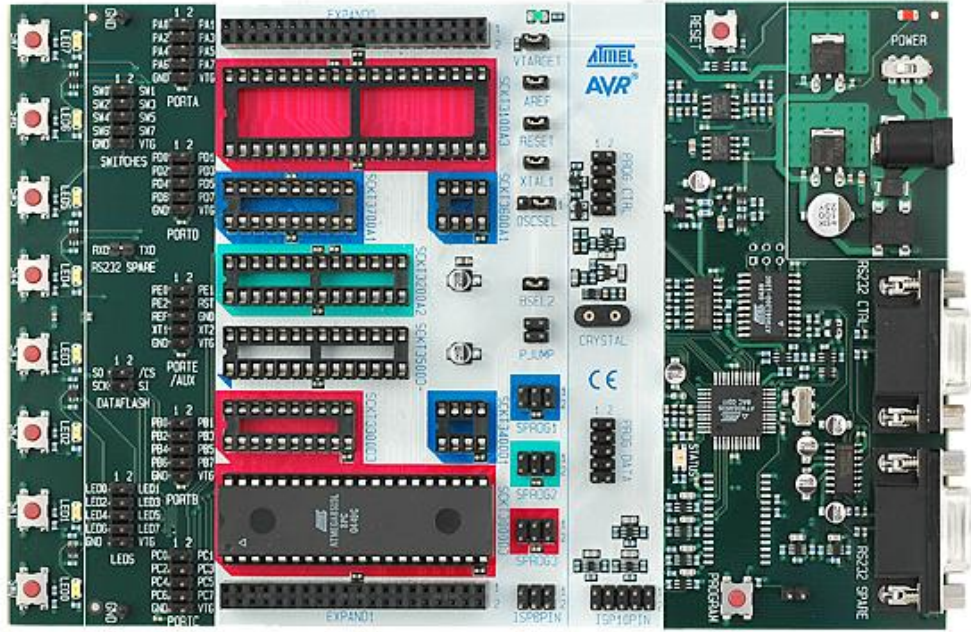
Şekil 5.9 : Atmega128 mikrokontrolörü.

Atmega serisi mikrodeneleyiciler 8-bitlik veri yoluna sahip olmalarına rağmen çok sayıda çevresel donanım ünitesi barındırırlar ve birçok uygulama için uygundurlar. Atmega serisi ile üretilen bir çok aktif RFID etiketi de bulunmaktadır. Atmega serisi mikrodeneleyicilerin öne çıkan bazı özellikleri aşağıda yer almaktadır:

- 16 MHz de 16 MIPS’e kadar çıkabilme
- Çip üzerinde 2 saat darbesinde çarpım
- 16K Byte Flash program belleği
- 512 byte EEPROM
- 1K Byte dahili SRAM
- Yaz/sil sayısı: 10,000 Flash/100,000 EEPROM
- Veri saklama: 20 yıl 85°C’de/100 yıl 25°C’ de(1)
- JTAG arayüzü
- 2 Adet 8-bit zamanlayıcı/sayıcı
- Bir adet 16-bit zamanlayıcı/sayıcı
- 4 PWM Kanalı

- 8 Kanallı, 10-bit ADC
- 32 programlanabilir giriş/çıkış pini

Atmega128 işlemcisini kullanabilmek için elbette ki bir devre gereklidir. Tıpkı kullanılan diğer mikrodenetleyicilerde olduğu gibi Atmega128 gerçeklemeleri de bir geliştirme kiti üzerinde yapılmıştır. Bu amaçla kullanılan kit Şekil 5.10’da da görülebilen Stk-500 kitidir.



Şekil 5.10 : Stk-500 kiti.

STK-500 kiti üzerinde aynı zamanda dahili bir progamlama ve hata ayıklama ünitesi de barındırmaktadır, bu nedenle ek bir progamlama ünitesi kullanılmamıştır.

5.2 Literatür Analizi

SHA-3 finalistleri, çoğunluğu donanımsal implementasyonlar olmak üzere çok sayıda akademik çalışmada analiz edilmiştir. Pratikte özet fonksiyonu gerçeklemelerinin büyük bölümü yazılım ortamında yapılmaktadır fakat SHA-3 finalistlerinin yazılım implementasyonlarının sayısı gerçekten azdır.

Her ne kadar sayıları az olsa da, SHA-3 finalistlerinin bazı gömülü sistem implementasyonları ve karşılaştırmaları bulunmaktadır. Bu çalışmalar incelendiğinde ise, bu gerçeklemeler oldukça yüksek yeteneklere sahip ARM11 gibi platformlarda yapılmıştır. Bazı araştırmalarda ise 8-bit’lik platformlar üzerinde platforma özgü

gerçekleştirmeler yapılmıştır ancak bu çalışmalar herhangi bir karşılaştırma imkânı vermemektedir. Algoritmaların gömülü sistem performanslarını inceleyebilmek için, gerçeklemeler yalnızca 8-bit mimarilerde değil, 16-bit ve 32-bit mimarilerde de yapılmalıdır.

Bir özet fonksiyonu tanım itibariyle matematiksel bir işlemdir. Bu matematiksel işlemin platform bağımsız olarak gerçekleşmesi mümkündür. Algoritma gerçeklemelerinde platform bağımsızlığı ise uygulamada büyük öneme sahiptir. Genel olarak, haberleşme esnasında özet fonksiyonu kullanan cihazlar aynı cihazlar değildir. Örneğin RFID sistemlerinde etiket ve okuyucunun aynı mimari üzerinde gerçekleşmesi, rastlanır bir durum değildir. RFID okuyucuları yüksek işlem kapasitesine sahip olmalıdır ve tüm etiketleri idare edebilmelidir. RFID etiketleri ise minimalist bir yaklaşımla tasarlanmalı ve çok az güç tüketmelidir. Literatürde gerçeklemeler yalnızca tek bir donanım mimarisi üzerinde yapılmıştır ve bu durum pratiğe uygun değildir.

Birçok platform üzerindeki gerçeklemeleri karşılaştırıldığı güzel bir çalışma Bernstein ve Lange tarafından yapılmıştır[22]. Çalışmanın odak noktası, gerçeklemelerin olabildiğince yüksek hızda çalışacak şekilde yapılmasıdır. Buradaki ölçüm metriği ise byte başına saat darbesi olarak tanımlanmıştır. Araştırmada kullanılan mimarilerin tamam yüksek kapasitelere sahip oldukça hızlı işlemcilerdir.

Davelaar ise finalistleri am64 platformunda karşılaştırmıştır [23]. Bu araştırmanın konusu ise, Java dilini kullanarak yapılacak gerçeklemelerin kolaylığıdır. Sıralama işlemi gerçekleştirme kolaylığına göre yapılmıştır.

Latif ve arkadaşları, Skein algoritması için C dili ile bir gerçekleştirme yapmıştır. Skein ailesinden Skein- 512 gerçekleştirilmiştir. Buradaki gerçekleştirilmede ise işlemci olarak Intel Core2Duo, 2GHz kullanılmıştır. Gerçekleştirme tek bir algoritma üzerinedir ve yüksek kapasiteli bir mimari üzerinde yapılmıştır.

Bos ve Stefan ise SHA-3 finalistlerinin gerçekleştirilmesini NVIDIA GPU üzerinde yapmıştır. Burada yapılan çalışmada bir gerçekleştirme de Cell platformu üzerinde yapılmıştır [24].

Eisenbarth ve arkadaşları ise, Blake, Blue Midnight Wish, Grøstl, Wish ve Shavite-3 algoritmalarının gerçekleştirilmesini 8-bit veri yoluna sahip bir mikroişlemci üzerinde yapmıştır. Bu çalışmada Blue Midnight Wish algoritmasının en yüksek hızla çalıştığı

belirtilmiştir. 8-bit mimariler için ise en dengeli algoritmanın Blake olduğu anlatılmıştır.

Bu tez kapsamında, SHA-3 finalisti olan algoritmaların hafif gerçeklemeleri, farklı seviyedeki gömülü sistemlerde gerçekleştirilmiş ve karşılaştırılmıştır. Seçilen donanım platformları en çok kullanılan ve en fazla mücadele içeren mimarilerden seçilmiştir. Yine yapılan çalışmada kısa, orta ve uzun giriş veri boyutları ile testler yapılmıştır. Tüm implementasyonlar C dilinde yapılmıştır.

Bu bölümde yapılan çalışma ile gömülü sistem üzerinde en hızlı çalışan, en az yer kaplayan ve bellek esnekliği en fazla olan algoritmalar belirlenmiş ve en iyi algoritma, konum mahremiyeti sağlayan protokol tasarımında özet fonksiyonu olarak kullanılmıştır.

5.3 Gerçekleme Sonuçları

Bu kısımda, SHA-3 finalisti algoritmaların hafif gerçeklemeleri tanımlanmaktadır. Her bir algoritmanın gerçekleme detayları açık şekilde anlatılmaktadır. Ardından hedefler, dikkat edilen hususlar ve diğer gerçekleme detayları tartışılmaktadır.

Bu tez kapsamında gerçekleştirilen özet fonksiyonlarında sabit bir özet fonksiyonu çıkış uzunluğu üzerine temel atmak adına tüm algoritmaların 256-bit sürümleri gerçekleştirilmiştir. 256-bit uzunluğunun seçilme sebeplerinden biri de bellek kullanımını mümkün olduğunca azaltmak ancak güvenlikten de feragat etmemektir.

Yapılan tüm gerçeklemeler, tam anlamıyla fonksiyonel olan ve tek başına çalışabilen uygulamalar olarak gerçekleştirilmiştir. Yani algoritmaların gerçeklemeleri birlikte değil ayrı ayrıdır. Her bir uygulama, üretilen özet çıktısının kontrol edileceği şekilde tasarlanmış ve test edilmiştir, dolayısıyla hesaplamalarda bir yanlışlık olmamaktadır. Her bir gerçeklemede, algoritma gerçeklemesinin yanı sıra, donanım konfigürasyonları ve ölçüm sistemi yazılımı da yer almaktadır. Ancak algoritmanın dışında kalan bu kısımlar, ölçümlere dâhil edilmemiştir. Sonuç olarak yalnızca algoritmaların performans metrikleri ölçülmüştür.

Bir özet fonksiyonun çalışma hızına ek olarak bellek esnekliği de gömülü sistem performansında önemli bir etkidir. Gömülü sistemlerde, başka bir bellek ünitesi olmadığından, algoritmaların bellek tüketimi RAM ve Flash bellek üzerine dağılmaktadır. Yalnızca hız performansı, bir algoritmanın gömülü sistem

gerçeklemesine uygunluđuna karar vermekte önemli deđildir. Gml sistemlerin kısıtları çerçevesinde RAM ve Flash bellek kullanımı da sınırlanabilir. Bazı durumlarda, Flash bellek çok az olduđundan, sabit deđişkenleri de RAM zerine yerleřtirmek gerekebilir. İlk bakıřta sabit deđerleri RAM zerinde tutmak mantıklı gelmeyebilir ancak bu uygulama kimi RFID uygulamalarında zorunluluk olarak karřımıza çıkmaktadır. Bir algoritmanın asgari RAM kullanımı ya da asgari Flash bellek kullanımı, pratikte uygulama seřiminde önemli bir parametredir. Bu nedenle bu tez kapsamında, her bir algoritma iin RAM kullanımını ve Flash bellek kullanımını minimize eden gereklemeler yapılmıřtır. Bu bellek optimizasyonları, iřletim hızını gzle grlr řekilde etkilememiřtir nk her bir algoritmanın iřletilme hızı, RAM ve Flash bellek alanlarına eriřim sreleri arasındaki farktan çok ok byktr.

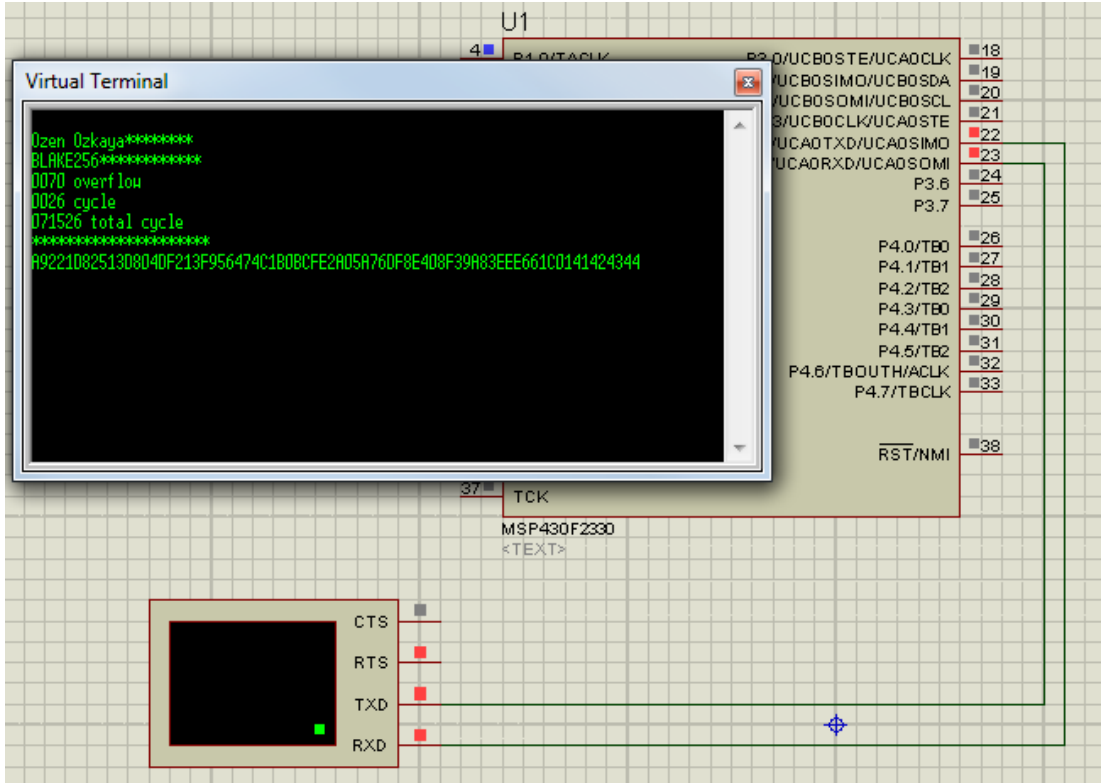
Her bir algoritma; kısa, orta ve uzun boyutlu giriř verileriyle test edilmiřtir. Kısa veri boyutu 8-byte, orta veri boyutu 64-byte, uzun veri boyutu ise 128-byte olarak belirlenmiřtir. Bu tez kapsamında gml sistem performansları incelendiđinden giriř verisi uzunluđu 128-byte olarak sınırlandırılmıřtır. Yine her bir algoritma gereklemesinde, veri tipi uzunlukları 32 bit ile sınırlandırılmıřtır. 64-bitlik iřlemler ise alt iřlem paracıklarına blnmřtr. Veri atamaları dngler iinde yapılmıř ve kod boyutu olabildiđince dřk tutulmaya alıřılmıřtır. Her bir gerekleme giriř verisi boyutu olara 8-bit ve katlarına izin vermektedir. Tm gereklemeler C dilinde yapılmıřtır. Bu sayede platform bađımsızlıđı sađlanarak adil bir karřılařtırma yapılmıřtır.

5.3.1 Blake-256

Blake [14], i durum deđiřkeni olarak 4x4 bir matris kullanmaktadır. 256 bit ıkıř uzunluđuna kadar, her bir durum paracıđı 32 bit geniřliđindedir ve uygulamanın ekirdek operasyonu 14 kez tekrarlanmıřtır. Blake algoritmasının her bir turu G fonksiyonunu 8 kez ađırmaktadır ve bunların her biri durum deđiřkeninin 4 alt paracıđını permte etmektedir. 14 turun bitiřinde ise son adım olarak, mevcut zet deđer, bir nceki zet ıktısı ve tuz(salt) deđerini kullanılarak yeni zet ıktısı oluřturulmaktadır. Burada bahsedilen G fonksiyonu, Blake algoritmasında veri transformasyonunu sađlamaktadır. G fonksiyonu; toplama, XOR ve 4 farklı 32-bit sađa dndrme operatrn kullanmaktadır. G fonksiyonu her turda farklı sırayla

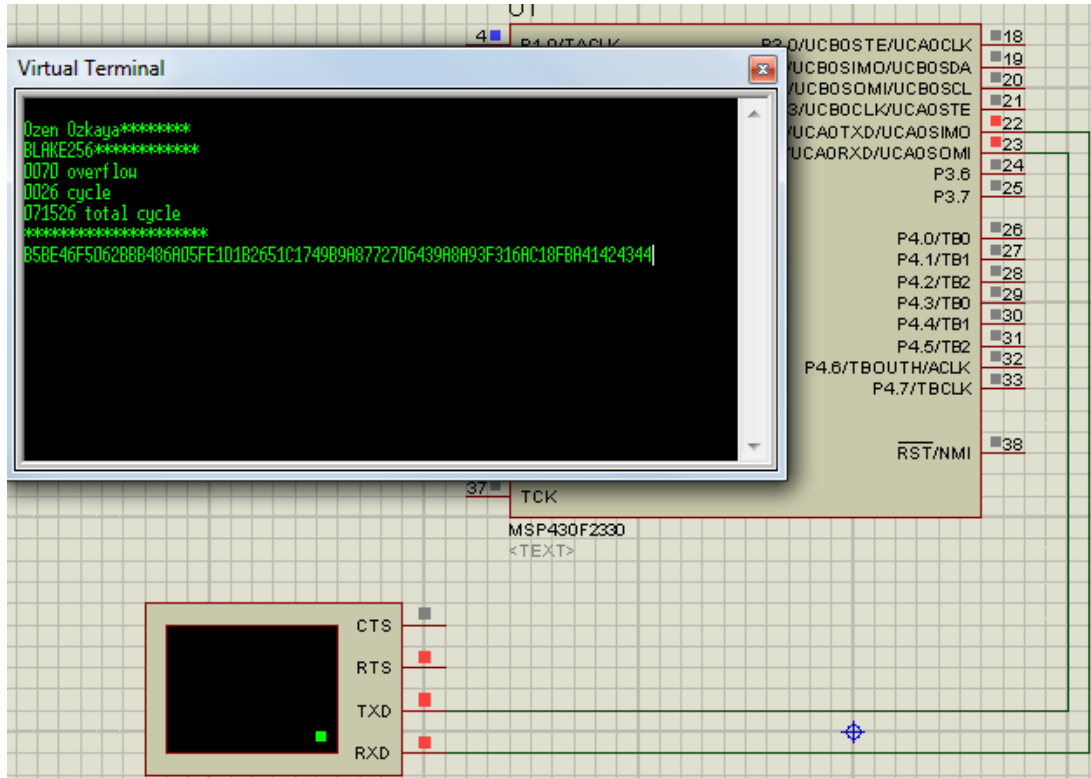
işleme dâhil edilen 16 adet 32-bitlik sabit değeri kullanmaktadır. Bu işlem sigma permütasyonu olarak da bilinmektedir. Blake algoritması hızlı ve bellek dostu bir algoritma olarak öne sürülmüştür.

Blake algoritmasının Msp430 gerçeklemesinin “ABCDEFGH” giriş verisi için ürettiği çıkış, Proteus Isis benzetim programında benzetimi yapılmıştır. İlgili benzetim çıktısı gerçek hayattaki ile birebir aynı olmuştur. Benzetim ekranına ilişkin çıktı Şekil 5.11’de yer almaktadır.



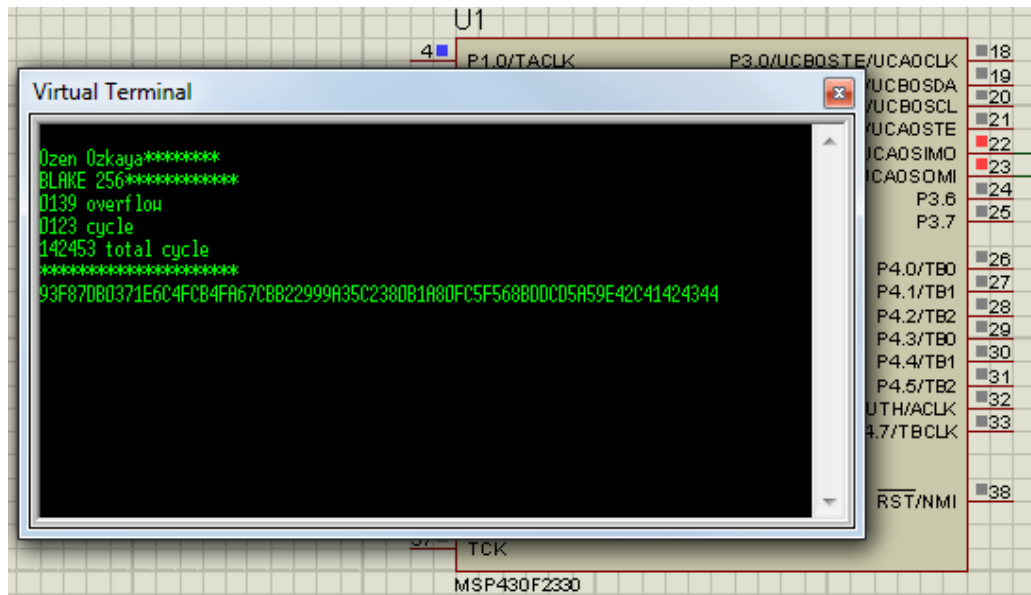
Şekil 5.11 : Blake algoritmasının “ABCDEFGH” girişi için MSP430 üzerinde benzetimi.

Giriş verisinin tek bir bitinin bile değişmesi durumunda, özet çıktısının farklı olması beklenmektedir. Bu amaca yönelik olarak giriş verisi olarak “ABCDEFGG” verilmesi durumundaki benzetim çıktısı Şekil 5.12’deki gibidir.



Şekil 5.12 : Blake algoritmasının “ABCDEFGG” girişi için MSP430 üzerinde benzetimi.

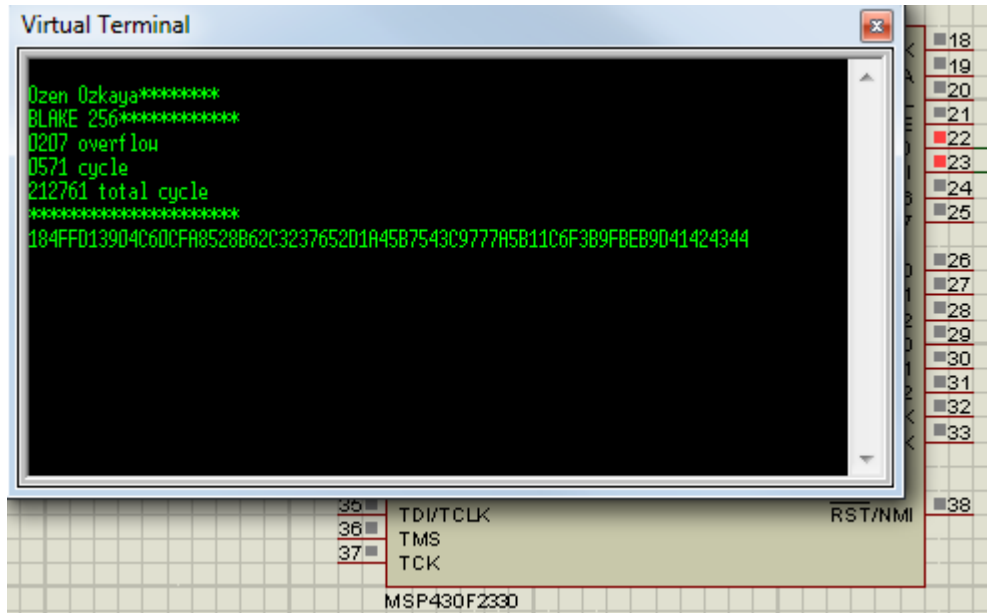
Şekil 5.11 ve Şekil 5.12’de görüldüğü üzere giriş verisindeki küçük bir değişiklik bile çıkış verisinin tamamen değişmesine yol açmaktadır. Bu da gerçeğin doğruluğunu göstermektedir.



Şekil 5.13 : Blake algoritmasının orta uzunluktaki veri girişi için MSP430 üzerinde benzetimi.

Benzer şekilde farklı giriş verisi uzunluklarında farklı özet çıktıları üretilmelidir. Şekil 5.13’de orta uzunluklu veri girişi için, giriş olarak “ABCDEFGHJKLMOPQRSTUVWXYZABCDEFGHJKLMOPQRSTUVWXYZ ABCDEFGHIJKLM” dizisi verildiğinde üretilen özet çıktısı görülmektedir. Burada dikkat edilmesi gereken iki adet nokta bulunmaktadır. Giriş verisinin boyutu değiştiğinden, Şekil 5.12’deki işletim zamanından çok daha fazla bir işletim süresiyle program tamamlanmıştır. Ayrıca özet çıktısı olarak yine bambaşka bir değer üretilmiştir.

Son olarak uzun boyutlu veri girişi için olan benzetim çıktısı Şekil 5.14’te yer almaktadır. Bu durumda algoritmanın girişine verilen karakteri dizisi "ABCDEFGHJKLMOPQRSTUVWXYZABCDEFGHJKLMOPQRSTUVWXYZ ABCDEFGHIJKLMABCDEFGHIJKLMOPQRSTUVWXYZABCDEFGHIKLMO PQRSTUVWXYZABCDEFGHIJKLM" olmaktadır. Şekil 5.14’ten görülebileceği üzere bu durumda da işletilme süresi ve özet değeri önceki değerlerden farklıdır.



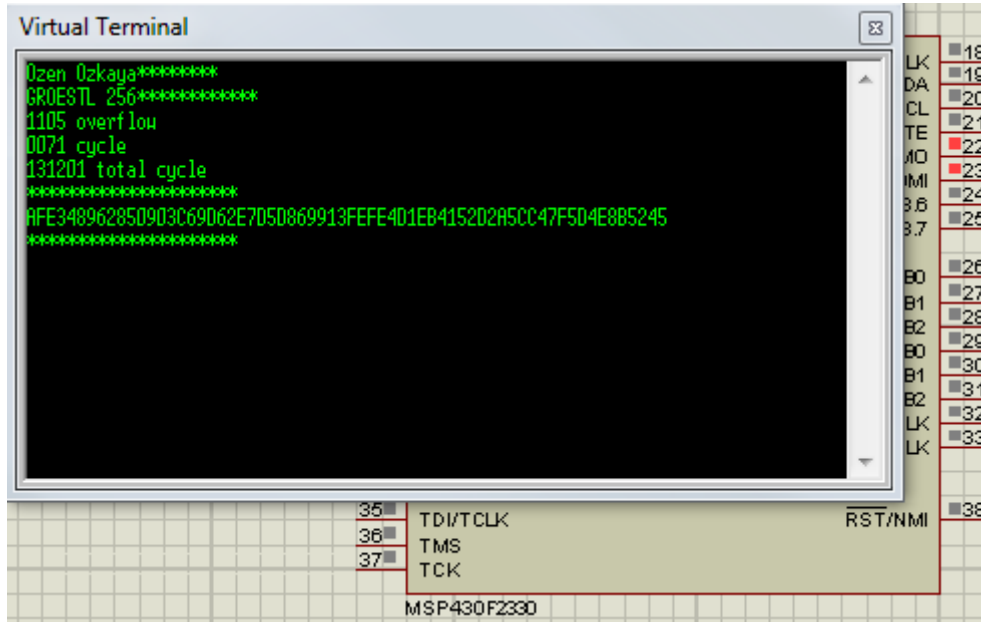
Şekil 5.14 : Blake algoritmasının uzun veri girişi için MSP430 üzerinde benzetimi.

5.3.2 Grøstl-256

Grøstl [15] fonksiyonu, sıkıştırma fonksiyonları olarak adlandırılan çekirdek üniteler içermektedir. Bu sıkıştırma fonksiyonları benzer şekilde tasarlanmış P ve Q adı verilen operasyonlarla oluşturulmaktadır. Her bir fonksiyon çağrısında P ve Q işlemleri 10 kez uygulanmaktadır. Sıkıştırma fonksiyonun ardından çıkış transformasyonu adındaki 10 P çağrısı içeren fonksiyon çalışmaktadır. Grøstl

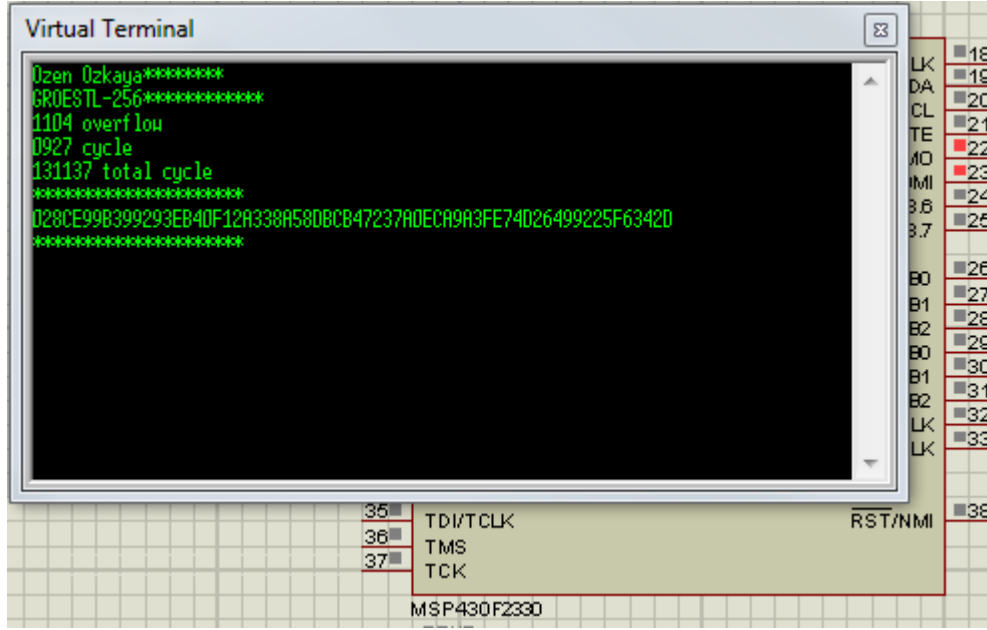
algoritmasının 256-bit versiyonunda 8-bitlik deęerler ieren 8 adet dizi bulunmaktadır. P ve Q operasyonları ilerinde AES ile benzerlik gsteren alt operatrler tařımaktadırlar. Ayrıca algoritmada kullanılan tablolar da AES'teki tablolarla benzerdir. Grstl algoritmasının gml sistem gereklemeleri hız ve bellek kullanımı bakımından iyi performans gsterememiřtir. Bu veriler ileride paylařılacaktır.

Grstl algoritmasının Msp430 gereklemesinin “ABCDEFGH” giriř verisi iin rettięi ıkıřın, Proteus Isis benzetim programında benzetimi yapılmıřtır. İlgili benzetim ıktısı gerek hayattaki ile birebir aynı olmuřtur. Benzetim ekranına iliřkin ıktı řekil 5.15’de yer almaktadır.



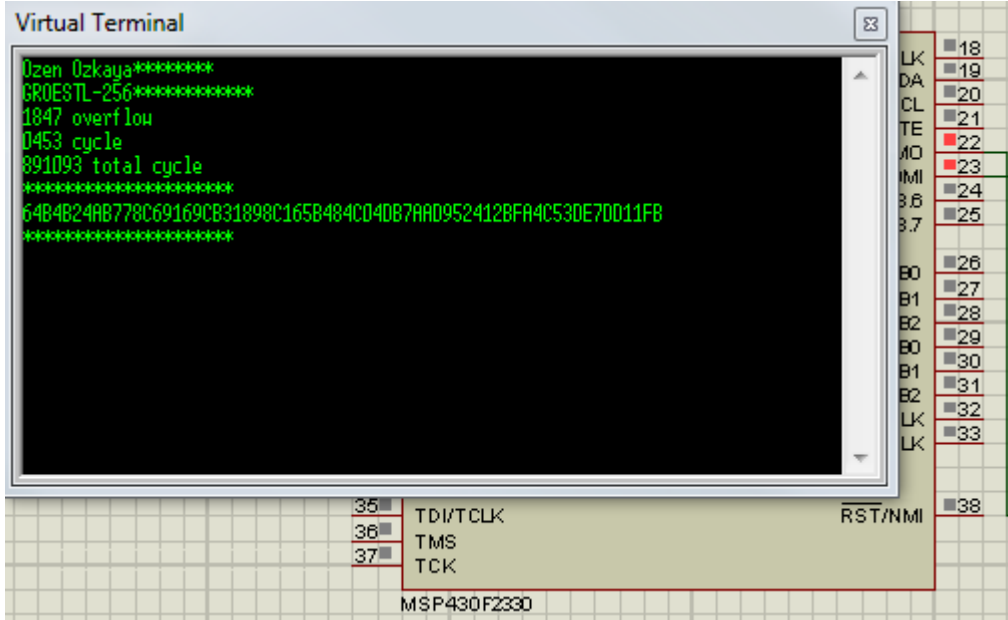
řekil 5.15 : Grstl algoritmasının “ABCDEFGH” giriři iin Msp430 zerinde benzetimi.

Giriř verisinin tek bir bitinin bile deęiřmesi durumunda, zet ıktısının farklı olması beklenmektedir. Bu amaca ynelik olarak giriř verisi olarak “ABCDEFGG” verilmesi durumundaki benzetim ıktısı řekil 5.16’daki gibidir.



Şekil 5.16 : Grøstl algoritmasının “ABCDEFGGG” girişi için Msp430 üzerinde benzetimi.

Şekil 5.15 ve Şekil 5.16’da görüldüğü üzere giriş verisindeki küçük bir değişiklik bile çıkış verisinin tamamen değişmesine yol açmaktadır. Bu da gerçeğin doğruluğunu göstermektedir.

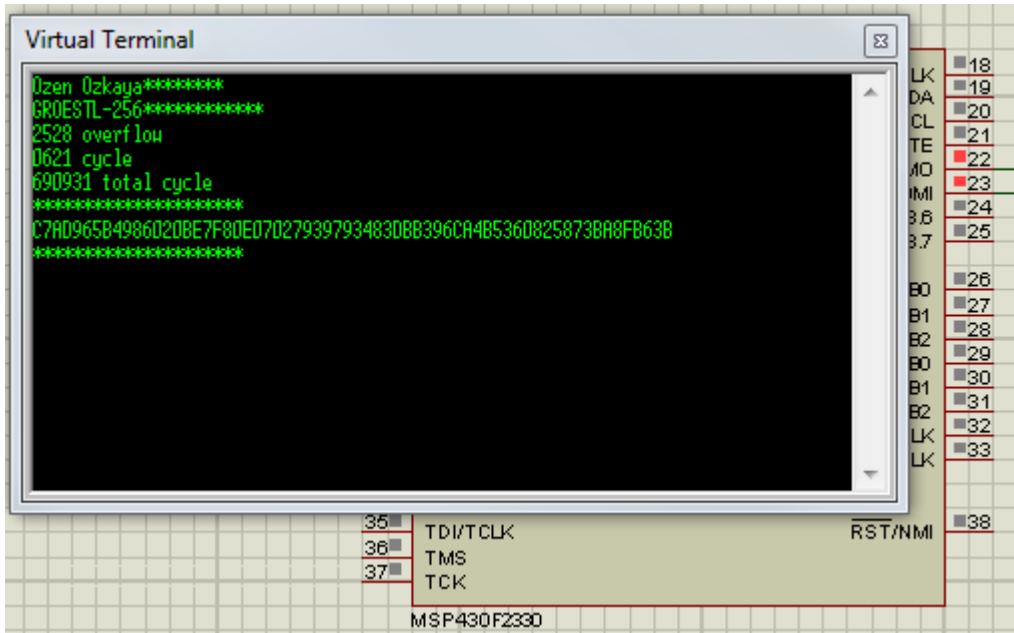


Şekil 5.17 : Grøstl algoritmasının orta uzunluktaki veri girişi için Msp430 üzerinde benzetimi.

Benzer şekilde farklı giriş verisi uzunluklarında farklı özet çıktıları üretilmelidir. Şekil 5.17’de orta uzunluklu veri girişi için, giriş olarak “ABCDEFGHJKLMOPQRSTUVWXYZABCDEFGHIJKLMOPQRSTUVWXYZ”

“ABCDEFGHIJKLM” dizisi verildiğinde üretilen özet çıktısı görülmektedir. Burada dikkat edilmesi gereken iki adet nokta bulunmaktadır. Giriş verisinin boyutu değiştiğinden, Şekil 5.16’daki işletim zamanından çok daha fazla bir işletim süresiyle program tamamlanmıştır. Ayrıca özet çıktısı olarak yine bambaşka bir değer üretilmiştir.

Son olarak uzun boyutlu veri girişi için olan benzetim çıktısı Şekil 5.18’de yer almaktadır. Bu durumda algoritmanın girişine verilen karakteri dizisi "ABCDEFGHIJKLMOPQRSTUVWXYZABCDEFGHIJKLMOPQRSTUVWXYZABCDEFGHIJKLMOPQRSTUVWXYZABCDEFGHIJKLMOPQRSTUVWXYZABCDEFGHIJKLM" olmaktadır. Şekil 5.18’den görülebileceği üzere bu durumda da işlemlenme süresi ve özet değeri önceki değerlerden farklıdır.



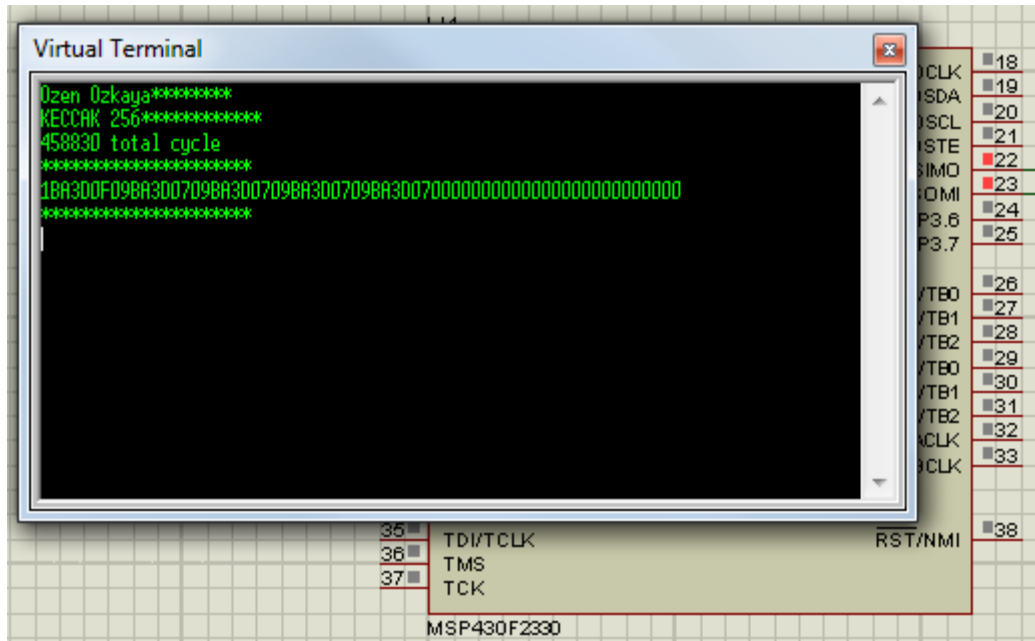
Şekil 5.18 : Grøstl algoritmasının uzun veri girişi için Msp430 üzerinde benzetimi.

5.3.3 Keccak-256

Keccak[19], giriş verisinin alt bloklarını, geniş uzunluklu durum değişkeni ile XOR operasyonunu kullanarak karıştırır. Durum değişkeni 5x5 olarak belirlenmiştir. Matrisin her bir elemanı şerit olarak isimlendirilmiştir ve 64-bit uzunluğundadır. Tek bir mesaj bloğu 24 tur boyunca işlenmektedir ve her bir tur 5 adet fonksiyon içermektedir. Bu fonksiyonlar Theta, Rho, Chi ve Iota olarak adlandırılmıştır. Theta fonksiyonu birçok XOR operasyonu içermektedir ve 64-bitlik bir kaydırma operasyonu barındırır. Gömülü ortamlarda tek seferde 64-bitlik dönüşüm yapmak

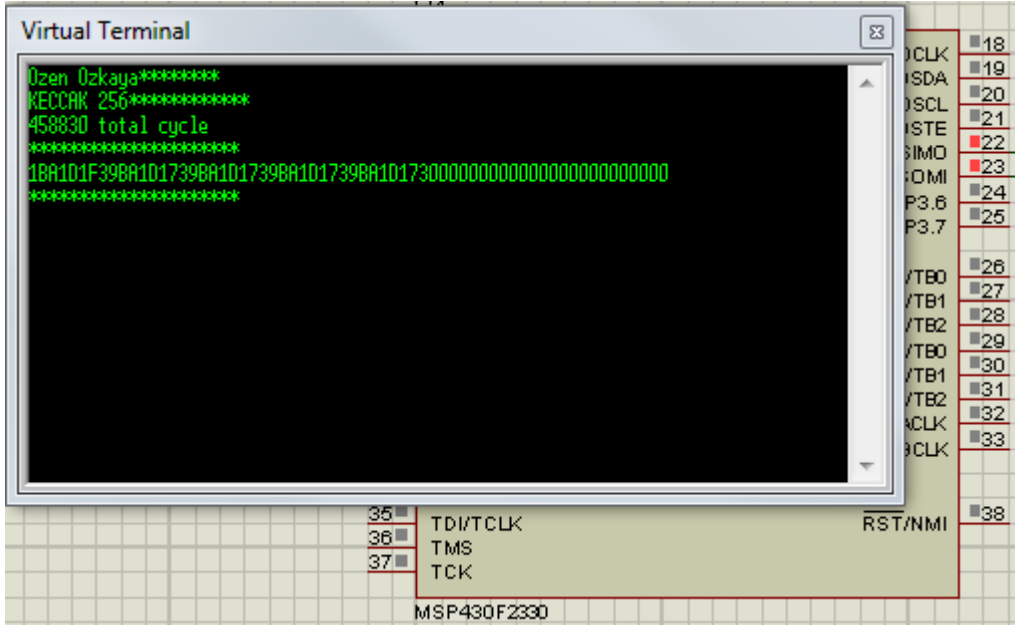
zor olduğundan bu işlem 32 -bitlik alt bloklara ayrılmıştır. Rho fonksiyonu ise, her bir tur için farklı döndürme sabitleri içermektedir. Pi ise algoritmanın permütasyon tablosunu tanımlamaktadır. Permütasyon tablosu zincirlenerek algoritmanın performansı artırılmıştır. Chi fonksiyonu ise XOR, AND ve NOT operatörlerinden başka bir operatör içermemektedir.

Keccak algoritmasının Msp430 gerçekleştirmesinin “ABCDEFGH” giriş verisi için ürettiği çıkış, Proteus Isis benzetim programında benzetime tabi tutulmuştur. İlgili benzetim çıktısı gerçek hayattaki ile birebir aynı olmuştur. Benzetim ekranına ilişkin çıktı Şekil 5.19’de yer almaktadır.



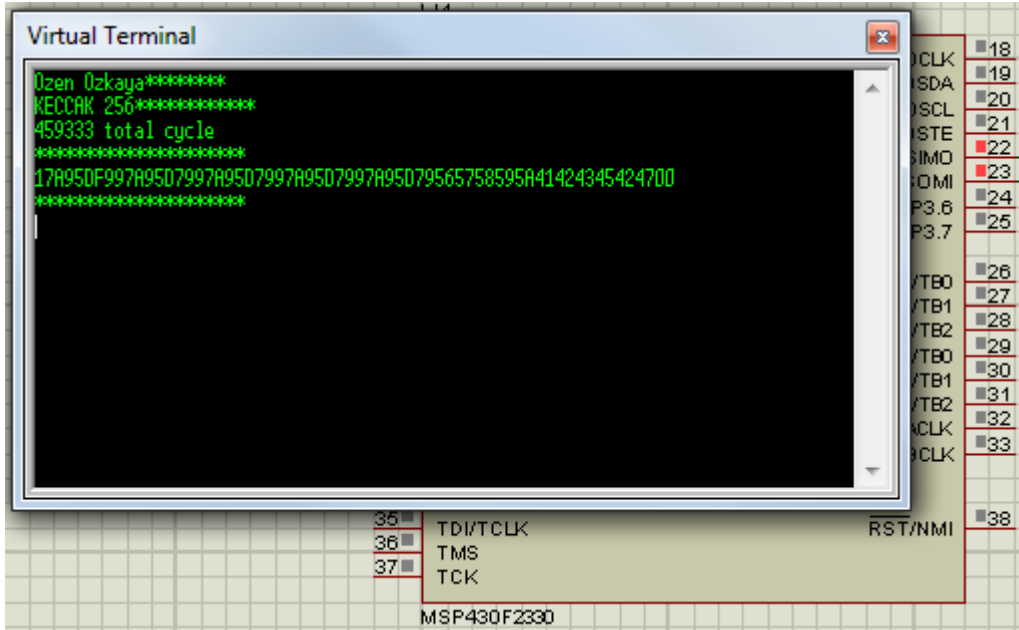
Şekil 5.19 : Keccak algoritmasının “ABCDEFGH” girişi için Msp430 üzerinde benzetimi.

Giriş verisinin tek bir bitinin bile değişmesi durumunda, özet çıktısının farklı olması beklenmektedir. Bu amaca yönelik olarak giriş verisi olarak “ABCDEFGG” verilmesi durumundaki benzetim çıktısı Şekil 5.20’deki gibidir.



Şekil 5.20 : Keccak algoritmasının “ABCDEFGG” girişi için Msp430 üzerinde benzetimi.

Şekil 5.19 ve Şekil 5.20’da görüldüğü üzere giriş verisindeki küçük bir değişiklik bile çıkış verisinin tamamen değişmesine yol açmaktadır. Bu da gerçeklemenin doğruluğunu göstermektedir.

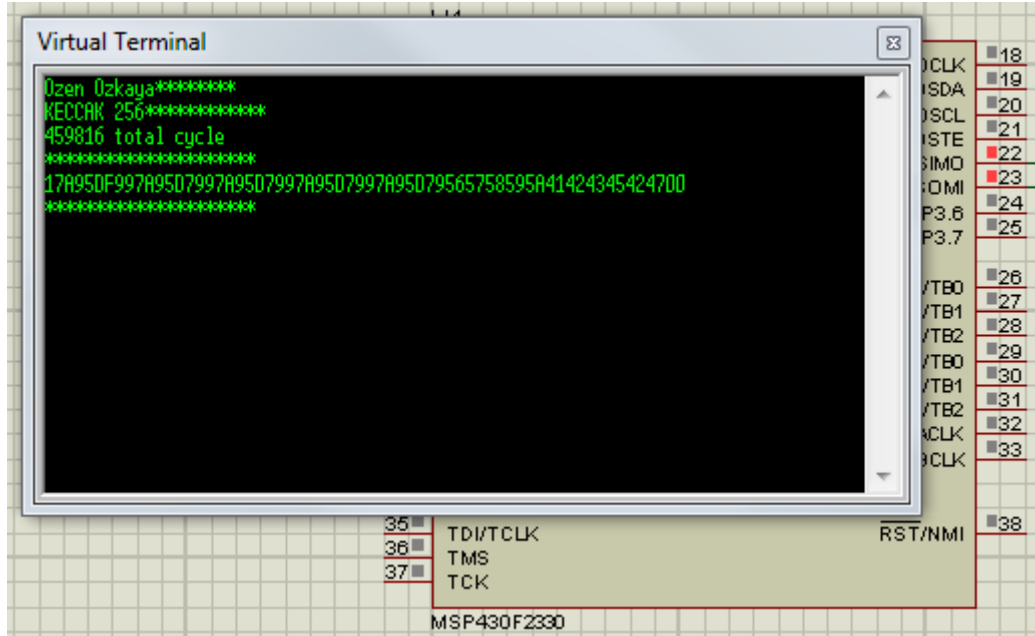


Şekil 5.21 : Keccak algoritmasının orta uzunluktaki veri girişi için MSP430 üzerinde benzetimi.

Benzer şekilde farklı giriş verisi uzunluklarında farklı özet çıktıları üretilmelidir. Şekil 5.21’de orta uzunluklu veri girişi için, giriş olarak “ABCDEFGHJKLMNOPQRSTUVWXYZABCDEFGHIJKLMOPQRSTUVWXYZ”

“ABCDEFGHIJKLM” dizisi verildiğinde üretilen özet çıktısı görülmektedir. Burada dikkat edilmesi gereken iki adet nokta bulunmaktadır. Giriş verisinin boyutu değiştiğinden, Şekil 5.20’deki işletim zamanından çok daha fazla bir işletim süresiyle program tamamlanmıştır. Ayrıca özet çıktısı olarak yine bambaşka bir değer üretilmiştir.

Son olarak uzun boyutlu veri girişi için olan benzetim çıktısı Şekil 5.22’de yer almaktadır. Bu durumda algoritmanın girişine verilen karakteri dizisi "ABCDEFGHIJKLMOPQRSTUVWXYZABCDEFGHIJKLMOPQRSTUVWXYZABCDEFGHIJKLMOPQRSTUVWXYZABCDEFGHIJKLMOPQRSTUVWXYZABCDEFGHIJKLM" olmaktadır. Şekil 5.22’den görülebileceği üzere bu durumda da işletilme süresi ve özet değeri önceki değerlerden farklıdır.



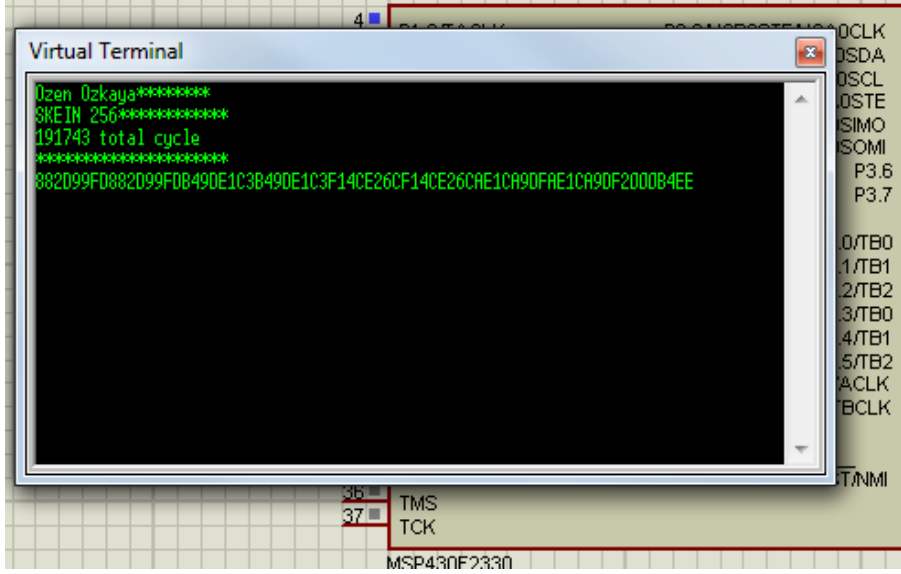
Şekil 5.22 : Keccak algoritmasının uzun veri girişi için Msp430 üzerinde benzetimi.

5.3.4 Skein-256

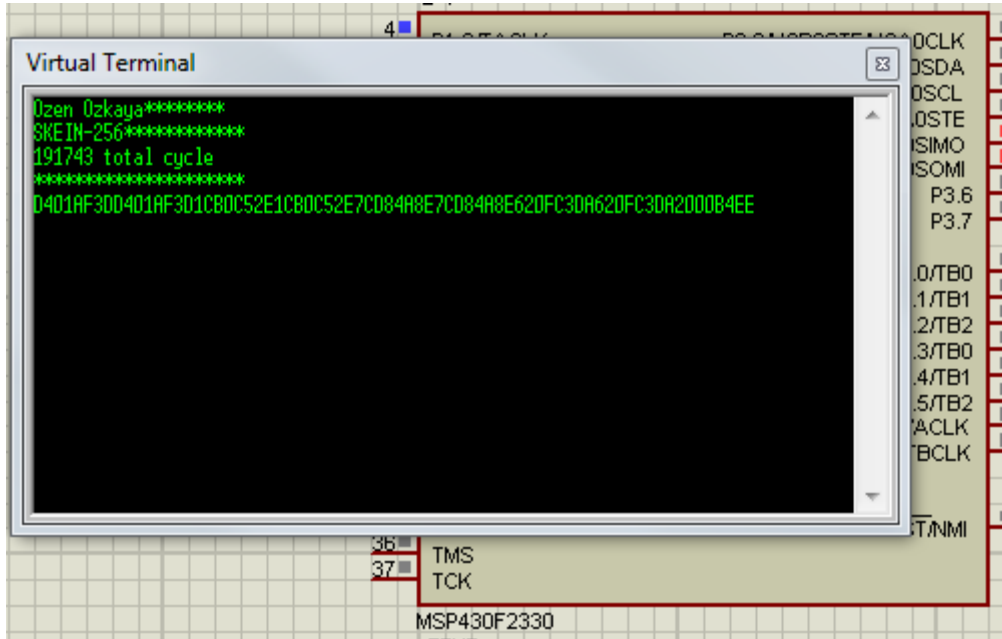
Skein [17], Threefish şifreleme algoritmasını kullanmaktadır. Bu tez kapsamında yapılan Skein gerçekleştirilmesi 256-bit durum değişkeni kullanmaktadır ve 256-bit çıkış üretmektedir. Skein algoritmasında çok sayıda tur vardır ancak her bir tur oldukça kısa ve basit tutulmuştur. Skein algoritmasının karıştırma operasyonları 72 tur boyunca uygulanır ve her bir tur 32-bitlik toplama, XOR ve döndürme operasyonlarını içermektedir. Bu sebeple Skein hızlı çalışacak şekilde tasarlanmıştır.

Skein aynı zamanda 64-bitlik bir döndürme operasyonu içerir ancak bu 32-bitlik döndürme operatörler kullanılarak gerçekleştirilmiştir.

Skein algoritmasının Msp430 implementasyonunun “ABCDEFGH” giriş verisi için ürettiği çıkış, Proteus Isis benzetim programında benzetime tabi tutulmuştur. İlgili benzetim çıktısı gerçek hayattaki ile birebir aynı olmuştur. Benzetim ekranına ilişkin çıktı Şekil 5.23’de yer almaktadır.



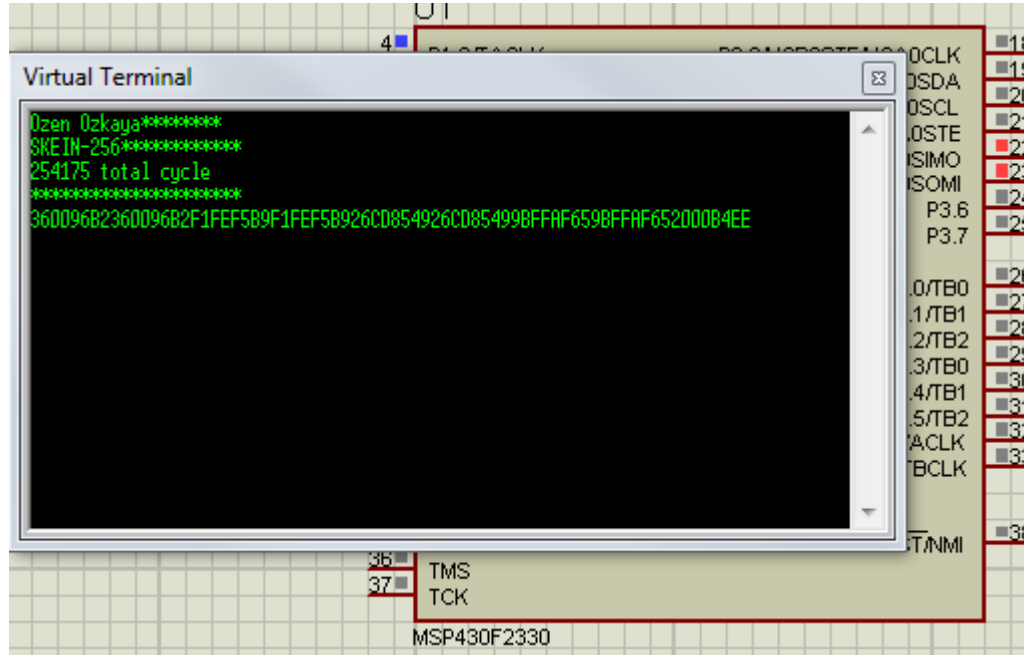
Şekil 5.23 : Skein algoritmasının “ABCDEFGH” girişi için Msp430 üzerinde simülasyonu.



Şekil 5.24 : Skein algoritmasının “ABCDEFGG” girişi için Msp430 üzerinde benzetimi.

Giriş verisinin tek bir bitinin bile değişmesi durumunda, özet çıktısının farklı olması beklenmektedir. Bu amaca yönelik olarak giriş verisi olarak “ABCDEFGGG” verilmesi durumundaki benzetim çıktısı Şekil 5.24’deki gibidir.

Şekil 5.23 ve Şekil 5.24’da görüldüğü üzere giriş verisindeki küçük bir değişiklik bile çıkış verisinin tamamen değişmesine yol açmaktadır. Bu da gerçeğin doğruluğunu göstermektedir.

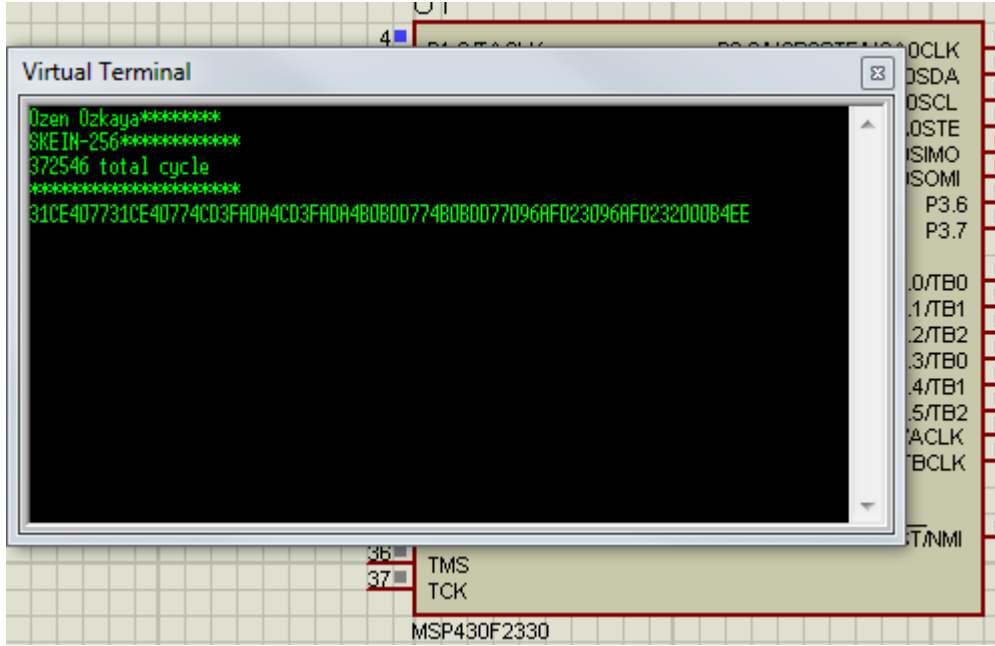


Şekil 5.25 : Skein algoritmasının orta uzunluktaki veri girişi için MSP430 üzerinde benzetimi.

Benzer şekilde farklı giriş verisi uzunluklarında farklı özet çıktıları üretilmelidir. Şekil 5.25’de orta uzunluklu veri girişi için, giriş olarak “ABCDEFGHJKLMOPQRSTUVWXYZABCDEFGHJKLMOPQRSTUVWXYZ ABCDEFGHIJKLM” dizisi verildiğinde üretilen özet çıktısı görülmektedir. Burada dikkat edilmesi gereken iki adet nokta bulunmaktadır. Giriş verisinin boyutu değiştiğinden, Şekil 5.24’deki işletim zamanından çok daha fazla bir işletim süresiyle program tamamlanmıştır. Ayrıca özet çıktısı olarak yine bambaşka bir değer üretilmiştir.

Son olarak uzun boyutlu veri girişi için olan benzetim çıktısı Şekil 5.26’de yer almaktadır. Bu durumda algoritmanın girişine verilen karakteri dizisi "ABCDEFGHJKLMOPQRSTUVWXYZABCDEFGHJKLMOPQRSTUVWXYZ ABCDEFGHIJKLMABCDEFGHIJKLMOPQRSTUVWXYZABCDEFGHIKLMO

PQRSTUVWXYZABCDEFGHIJKLM" olmaktadır. Şekil 5.26'den görülebileceği üzere bu durumda da işleme süresi ve özet değeri önceki değerlerden farklıdır.

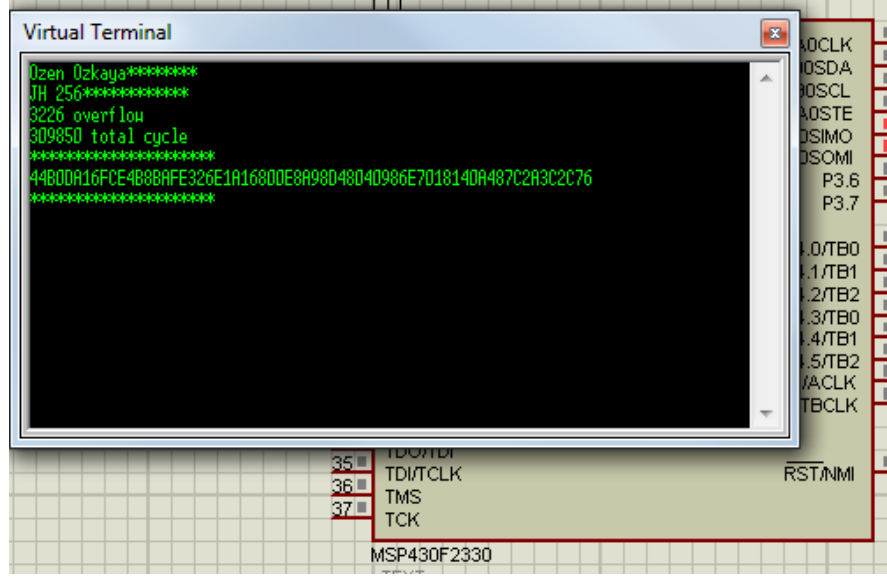


Şekil 5.26 : Skein algoritmasının uzun veri girişi için Msp430 üzerinde benzetimi.

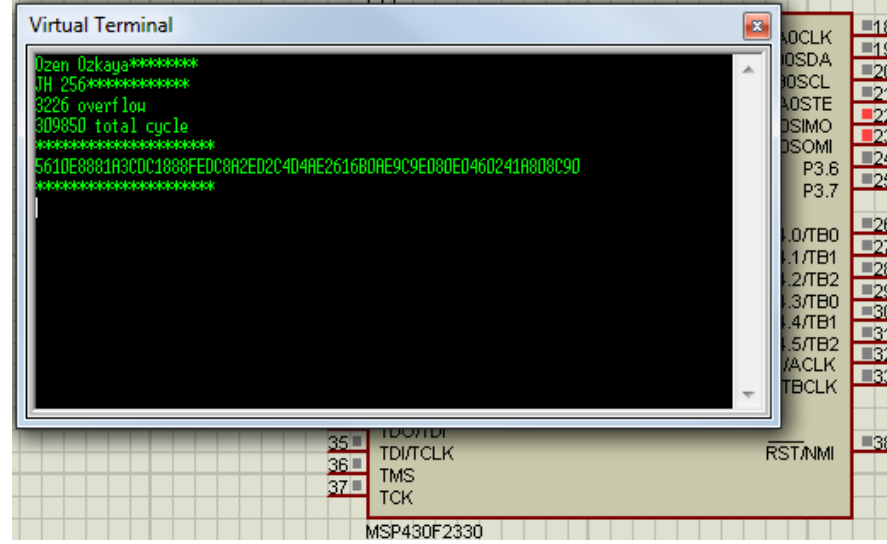
5.3.5 Jh-256

Jh [16] tarafından kullanılan iç durum değişkeni, çıkış verisi boyutundan bağımsızdır. Durum matrisinin boyutu ise 12-byte'dır. Mesaj bloğuna XOR işlemi uygulanır ve mevcut durum değişkeninin parçalarına 42 tur boyunca dönüşüm uygulanır. Jh algoritması 3 alt işlemde oluşur. Her bir tur farklı sabit değişkenler kullanır ve değişkene göre 2 adet farklı 4-bit dönüşümünden biri uygulanır. İkinci alt adım ise doğrusal dönüşümdür ve bu dönüşüm L ile simgelenmiştir. Ardından üçüncü alt adım olarak L işleminin ardından permütasyon fonksiyonu işletilir. Referans gerçeklemede Jh algoritması bit bölümlü gerçeklemelerde kolaylıklar sağlamaktadır.

Jh algoritmasının Msp430 gerçeklemesinin "ABCDEFGH" giriş verisi için ürettiği çıkışın, Proteus Isis benzetim programında benzetimi yapılmıştır. Benzetim ekranına ilişkin çıktı Şekil 5.27'de yer almaktadır.



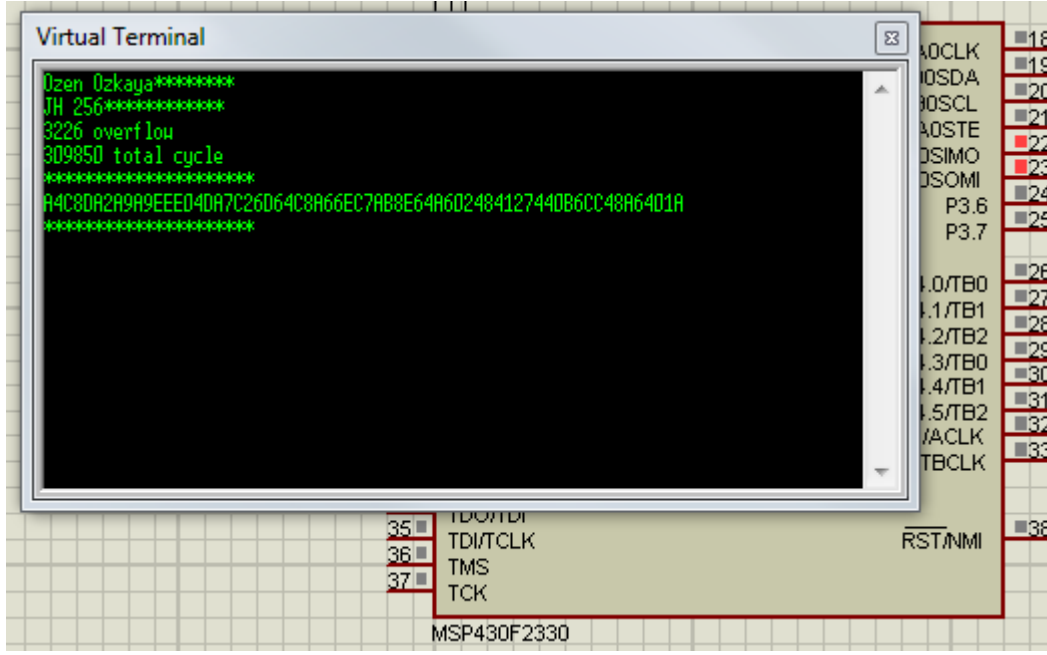
Şekil 5.27 : Jh algoritmasının “ABCDEFGH” girişi için Msp430 üzerinde benzetimi.



Şekil 5.28 : Jh algoritmasının “ABCDEFGG” girişi için Msp430 üzerinde benzetimi.

Giriş verisinin tek bir bitinin bile değişmesi durumunda, özet çıktısının farklı olması beklenmektedir. Bu amaca yönelik olarak giriş verisi olarak “ABCDEFGG” verilmesi durumundaki benzetim çıktısı Şekil 5.28’deki gibidir.

Şekil 5.27 ve Şekil 5.28’da görüldüğü üzere giriş verisindeki küçük bir değişiklik bile çıkış verisinin tamamen değişmesine yol açmaktadır. Bu da gerçeğlemenin doğruluğunu göstermektedir.



Şekil 5.29 : Jh algoritmasının orta uzunluktaki veri girişi için Msp430 üzerinde benzetimi.

Benzer şekilde farklı giriş verisi uzunluklarında farklı özet çıktıları üretilmelidir. Şekil 5.29’de orta uzunluklu veri girişi için, giriş olarak “ABCDEFGHJKLMNOPQRSTUVWXYZABCDEFGHJKLMNOPQRSTUVWXYZ ABCDEFGHIJKLM” dizisi verildiğinde üretilen özet çıktısı görülmektedir. Burada dikkat edilmesi gereken iki adet nokta bulunmaktadır. Giriş verisinin boyutu değiştiğinden, Şekil 5.28’daki işletim zamanından çok daha fazla bir işletim süresiyle program tamamlanmıştır. Ayrıca özet çıktısı olarak yine bambaşka bir değer üretilmiştir.

Son olarak uzun boyutlu veri girişi için olan benzetim çıktısı Şekil 5.30’de yer almaktadır. Bu durumda algoritmanın girişine verilen karakteri dizisi "ABCDEFGHJKLMNOPQRSTUVWXYZABCDEFGHJKLMNOPQRSTUVWXYZ ABCDEFGHIJKLMABCDEFGHIJKLMOPQRSTUVWXYZABCDEFGHIJKLMOPQRSTUVWXYZABCDEFGHIJKLM" olmaktadır. Şekil 5.30’dan görülebileceği üzere bu durumda da işletilme süresi ve özet değeri önceki değerlerden farklıdır.

Çizelge 5.1 : Algoritma gerçeklemelerinin byte cinsinden bellek kullanımı.

Algoritma	Optimizasyon	RAM	Flash
Blake	RAM	100	5522
Blake	Flash	320	5330
Grøstl	RAM	56	2056
Grøstl	Flash	372	976
Jh	RAM	100	3124
Jh	Flash	198	3020
Keccak	RAM	180	4224
Keccak	Flash	308	3928
Skein	RAM	174	3024
Skein	Flash	170	3028

Algoritma gerçeklemelerinin byte/saat darbesi cinsinden işletim hızı Atmega128 için Çizelge 5.2’de, Msp430 için Çizelge 5.3’te, Stm32 için ise Çizelge 5.4’te yer almaktadır.

Çizelge 5.2 : Atmega128 için algoritma gerçeklemelerinin byte/saat darbesi cinsinden işletim hızı.

Algoritma	Optimizasyon	Atmega128		
		Kısa	Orta	Uzun
Blake	RAM	6336	1560	1162
Blake	Flash	6336	1560	1162
Grøstl	RAM	10623	2212	1550
Grøstl	Flash	10623	2212	1550
JH	RAM	30702	3795	2531
JH	Flash	30702	3795	2531
Keccak	RAM	13052	6526	3256
Keccak	Flash	13052	6526	3256
Skein	RAM	8104	2676	1978
Skein	Flash	8104	2676	1978

Çizelge 5.2, Çizelge 5.3 ve Çizelge 5.4’te her bir gerçeklemenin işletim hızı saat darbesi /byte cinsinden verilmiştir. Buradaki her bir sonuç SHA-3 finalistlerinin 256-bit olarak gerçekleştirilmiş versiyonlarından ölçülmüştür. Bellek organizasyonlarındaki değişim işlem hızını ciddi bir şekilde etkilememektedir çünkü işlem hızı ile karşılaştırıldığında bellek erişim süresi çok düşük olmaktadır.

Çizelge 5.1’de her bir algoritmanın ne kadar Flash bellek ve RAM’in kullandığı byte cinsinden gösterilmektedir.

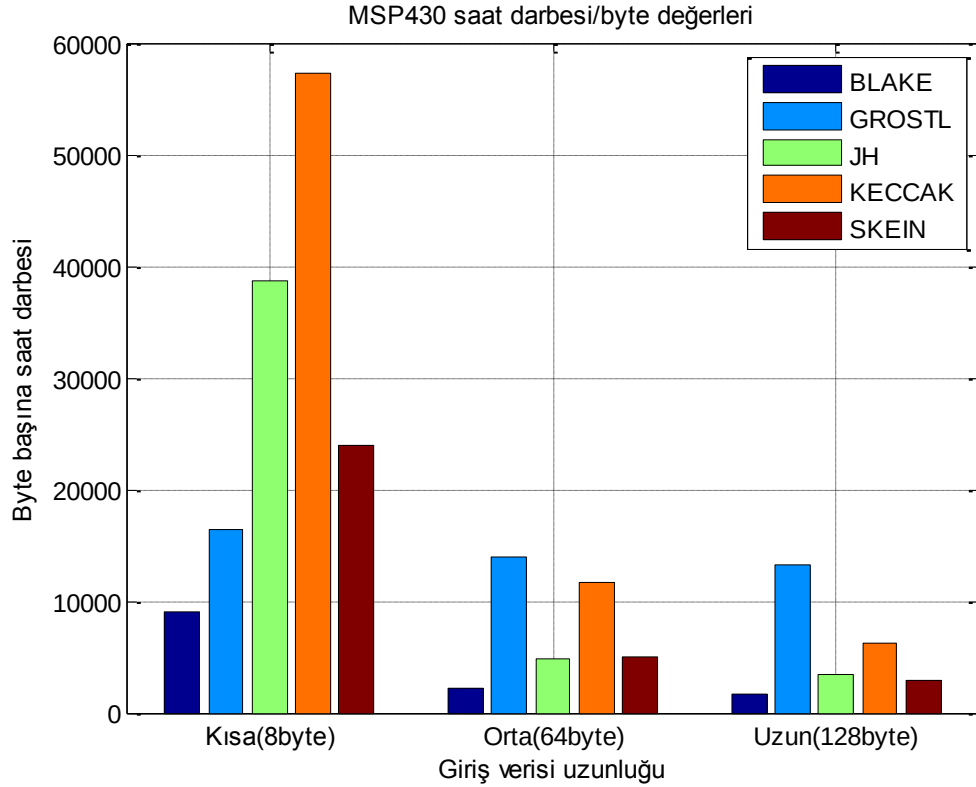
Çizelge 5.3 : Msp430 için algoritma gerçeklemelerinin byte/saat darbesi cinsinden işletim hızı.

Algoritma	Optimizasyon	Msp430		
		Kısa	Orta	Uzun
Blake	RAM	9051	2205	1718
Blake	Flash	9051	2205	1647
Grøstl	RAM	16399	13915	13192
Grøstl	Flash	16399	13915	13192
JH	RAM	38635	4841	3337
JH	Flash	38635	4841	3337
Keccak	RAM	57343	11656	6295
Keccak	Flash	57343	11656	6295
Skein	RAM	23970	3976	2902
Skein	Flash	23970	3976	2902

Çizelge 5.4 : Stm32 için algoritma gerçeklemelerinin byte/saat darbesi cinsinden işletim hızı.

Algoritma	Optimizasyon	Stm32		
		Kısa	Orta	Uzun
Blake	RAM	1896	461	1718
Blake	Flash	1896	461	1647
Grøstl	RAM	7419	869	13192
Grøstl	Flash	7419	869	13192
JH	RAM	5225	645	3337
JH	Flash	5225	645	3337
Keccak	RAM	7730	968	6295
Keccak	Flash	7730	968	6295
Skein	RAM	2989	490	2902
Skein	Flash	2989	490	2902

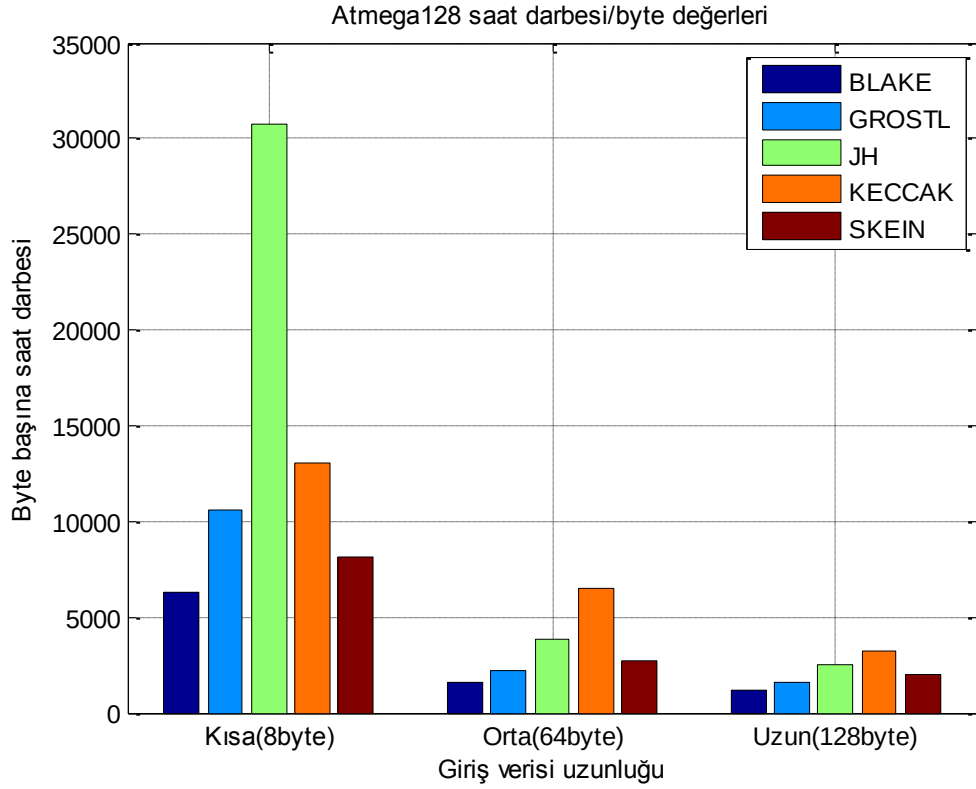
Şekil 5.31’ de Msp430’daki gerçeklemelerin saat darbesi/byte oranları görülmektedir. Bu şekil her bir gerçeklemenin kısa, orta ve uzun veri boyutlarındaki girişler için işlem hızını da içermektedir. Flash bellek optimizasyonlu ve RAM optimizasyonlu gerçeklemeler aynı hızda koştukları için, sadece algoritma adı çizdirilmiştir. Flash bellek optimizasyonlu ve RAM optimizasyonlu gerçeklemelerin aynı hızda koştuklarının sebebi, RAM ve Flash belleğe erişim süreleri arasındaki farkın, algoritmaların işletilme zamanları yanında ihmal edilebilir seviyede kalmasındandır. Gerçekte elbette ki bu iki farklı optimizasyon durumu için işletilme süresi farklıdır ancak bu farklılık ihmal edilebilir düzeyde kaldığından ölçümlere böyle yansımıştır. Şekil 5.31’e göre her bir veri uzunluğu seçeneği için en hızlı koşan Blake algoritmasıdır. Ölçümlere ait kesin değerler Çizelge 5.3’ten doğrulanabilir.



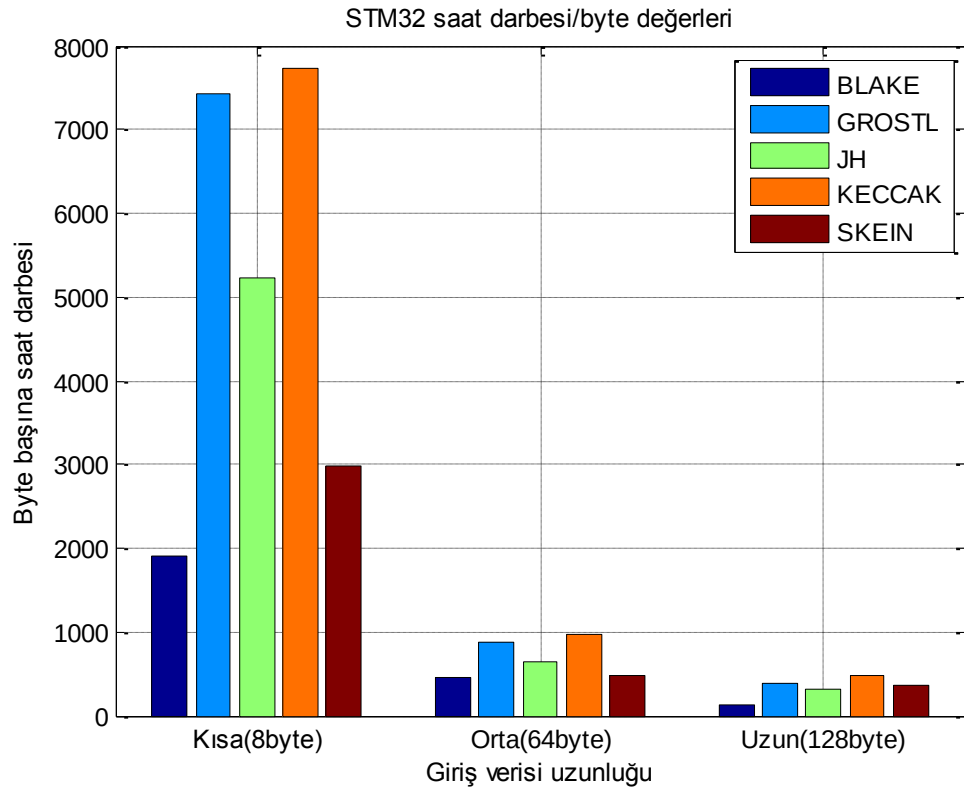
Şekil 5.31 : Msp430 saat darbesi/byte değerleri .

Şekil 5.32’ de Msp430’daki gerçeklemelerin saat darbesi/byte oranları görülmektedir. Şekil 1’de olduğu gibi, bu şekil de her bir gerçeklemenin küçük, orta ve büyük veri uzunluklarındaki girişler için işlem hızını da içermektedir. Şekil 5.31 ve Şekil 5.32 ye göre her veri uzunluğu için en hızlı algoritma Blake algoritmasıdır. Jh, küçük veri uzunluğundaki girişler için görece en yavaş olan algoritma iken, Keccak ise orta ve büyük veri uzunluğundaki girişler için en yavaş algoritmadır.

Şekil5.33’ te Stm32’deki gerçeklemelerin saat darbesi/byte oranları görülmektedir ve bu şekle göre de her veri uzunluğu için en hızlı algoritma Blake algoritmasıdır. İşlem süresi, Atmega ve Msp430’daki işlem süreleriyle karşılaştırıldığında Stm32 nin çok güçlü bir mikro kontrolör olduğu görülmektedir.

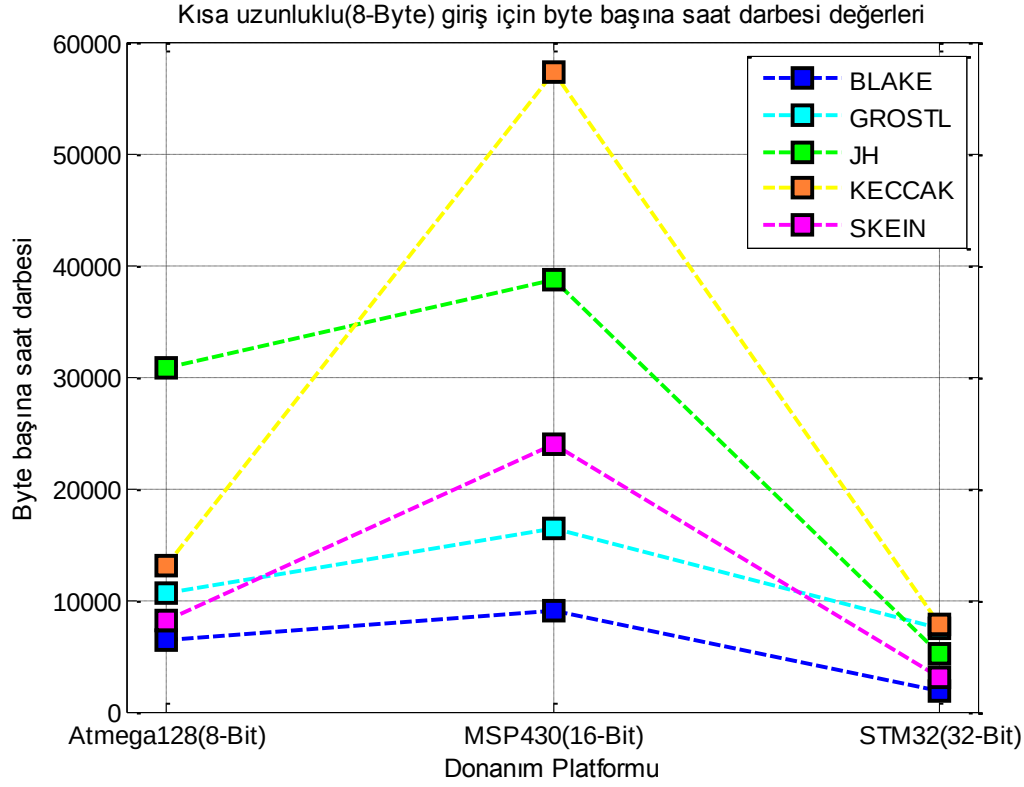


Şekil 5.32 : Atmega128 için saat darbesi/byte değerleri.



Şekil 5.33 : Stm32 için saat darbesi/byte değerleri.

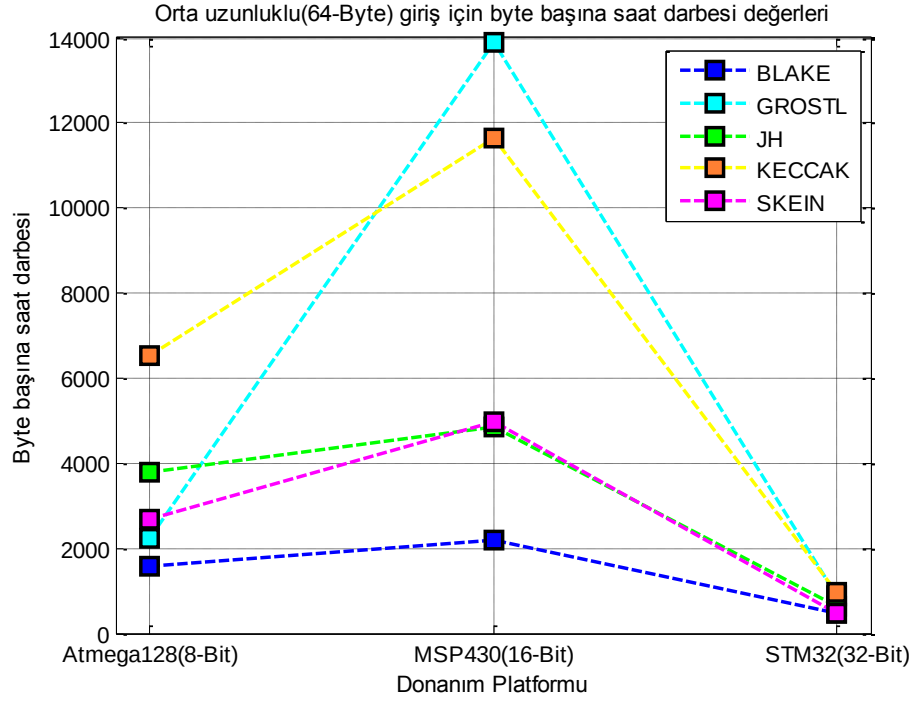
Şekil 5.14’de kısa uzunluktaki girişler için her bir algoritmanın koştugu gerekleme platformlarının karřılařtırılmasını gstermektedir. Genel olarak, beklendiđi gibi en hızlı olan Stm32 en yavaş olan ise Msp430 dır.



Şekil 5.34 : Kısa uzunluklu giriř için byte başına saat darbesi deđerleri.

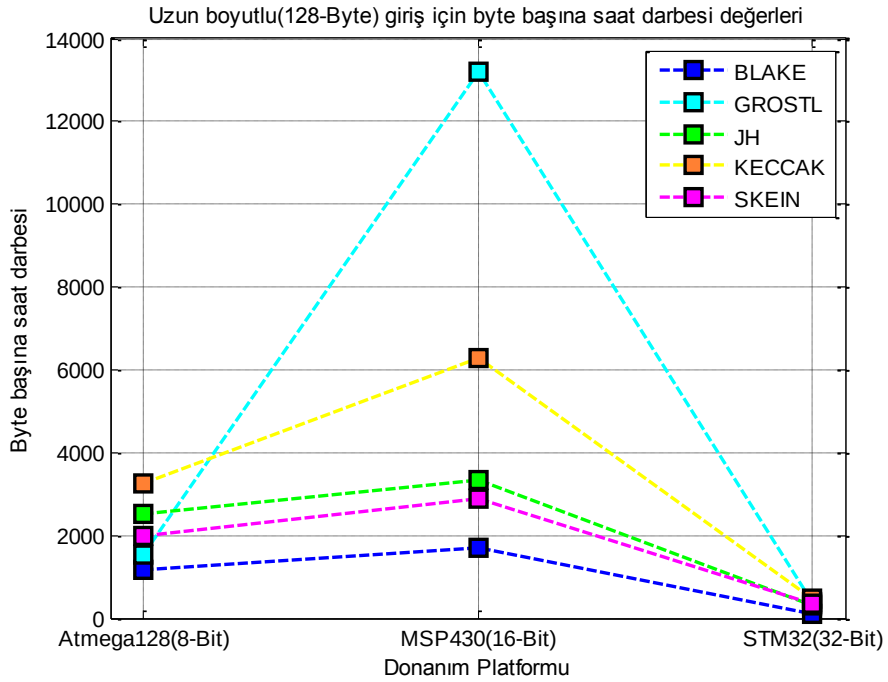
Şekil 5.35 ile orta uzunluktaki giriřler için her bir algoritmanın koştugu gerekleme platformlarının karřılařtırılmasını gstermektedir. Yine, en hızlı olan Stm32 en yavaş olan ise Msp430 dır. Bu Blake algoritmasının en hızlı olduđunun bařka bir gsterimidir.

Şekil 5.36 ile byk uzunluktaki giriřler için her bir algoritmanın koştugu gerekleme platformlarının karřılařtırılmasını gstermektedir ve burada da en hızlı olan algoritma deđiřmemiřtir. Burada byk veri uzunluđu için byte başına saat darbesi oranı yeterli bir byklktedir.

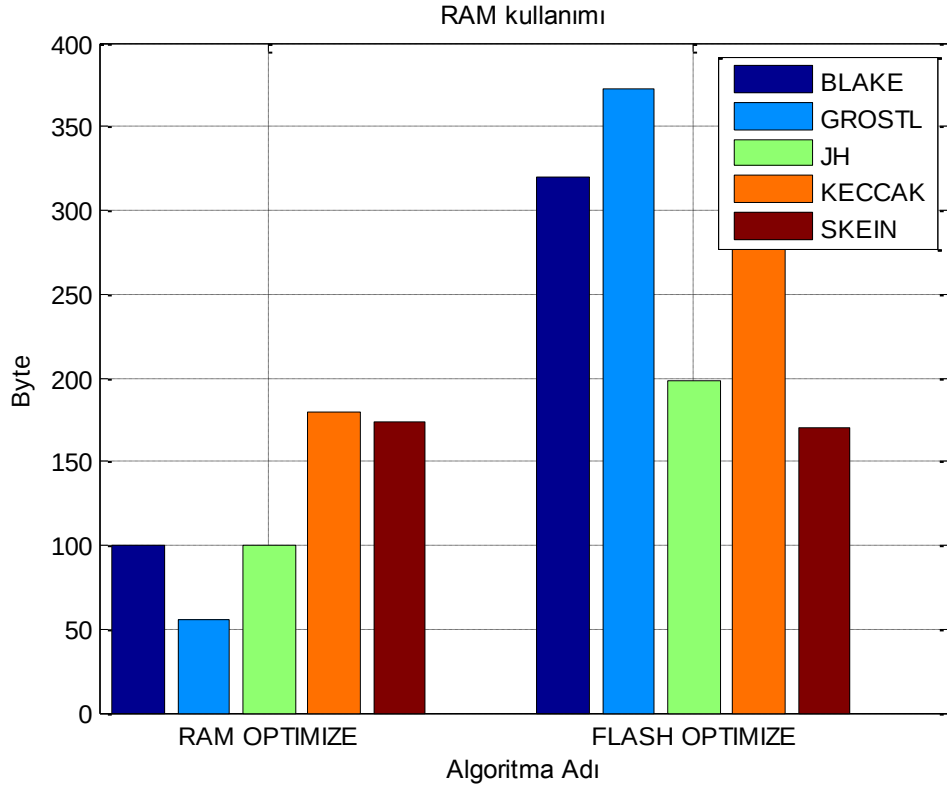


Şekil 5.35 : Orta uzunluklu giriş için byte başına saat darbesi değerleri.

Şekil 5.36 ise büyük uzunluktaki girişler için her bir algoritmanın koştugu gerçekteleme platformlarının karşılaştırılmasını göstermektedir ve burada da en hızlı olan algoritma değişmemiştir. Burada uzun veri girişi boyutu için byte başına saat darbesi oranı yeterli bir büyüklüktedir.



Şekil 5.36 : Büyük uzunluklu giriş için byte başına saat darbesi değerleri.



Şekil 5.37 : Gerçeklemelerin RAM kullanımı.

Şekil 5.37’de gerçeklemelere ilişkin RAM kullanımı görülmektedir. Burada hem RAM hem de Flash bellek optimizasyonlu versiyonlara ait sonuçlar bulunmaktadır. İlginç bir şekilde, olası en düşük RAM kullanımı performansı RAM optimizasyonlu Grøstl olmaktadır.

Şekil 5.18’de gerçeklemelere ilişkin Flash bellek kullanımı görülmektedir ve Grøstl olası en düşük Flash bellek kullanımına sahiptir. Bunun yanı sıra Blake algoritması da tatmin edici bir performans göstermektedir. Bellek kullanım ölçümleri Blake algoritması için ilginç sonuçlar vermiştir çünkü işlem, değişkenleri RAM den Flash belleğe taşırken; gerçekleştirme organizasyonu derleyiciye bağlıdır ve bu da muhtemelen sonuçları etkilemektedir.

5.5 Sonuç

Hız açısından bakıldığında, en iyi performansı Blake algoritması göstermektedir. İkinci en hızlı algoritma olan Grøstl ile karşılaştırıldığında Blake ondan üç kat hızlı çalışmaktadır. İlk bakışta Blake algoritması en düşük işlem süresi ve çok fazla işletim süresine sahip olarak görünse bile en hafif gerçeklemenin 7000 saat darbesi

sürmesi gayet normaldir. Msp430 basit bir mimaridir ve bazı modellerinde çarpıcı ve bölme alt modülleri bulunmamaktadır [25]. [25]' de verilen Tablo II' de çarpa ve bölme algoritmalarına ait ölçüm tablosu görülmektedir.

[25]'de de görüldüğü üzere, 16-bit uzunluktaki tamsayı değişkenlerin çarpımı 77 saat darbesi sürmektedir. Tamsayı tipindeki iki değişken arasındaki bölme işlemi de 183 saat darbesi sürmektedir. Tamsayı ile kesirli sayı tipindeki iki değişken arasındaki çarpım işleminin tamamlanması feci bir şekilde 427 çevrimde tamamlanmaktadır. 16-bit ile 32-bit uzunluktaki iki değişkenin çarpımına göre iki tane 32-bit uzunluktaki değişkenin çarpımı daha iyi olmakta SHA-3 aday algoritmaların hepsi 32-bitlik iki değişkenin çarpım ve bölme işlemlerini kullanmaktadır ve bunlar çok sayıda saat darbesi sürmektedir. Özetle, aday algoritmaların hız performanslarını Msp430 çarpma/bölme performansına bakarak analiz edersek özellikle Blake algoritmasının olağanüstü derecede iyi olduğunu görürüz.

Algoritmanın, belirli bir donanım üzerinde daha hızlı olarak gerçekleşmesi bu araştırmanın konusu değildir. Bu çalışma SHA-3 adaylarının hafif ölçüdeki gerçeklemelerinin karşılaştırılmasını içermektedir. Bu bildiri, her bir algoritmanın farklı mikro işlemci üzerindeki performanslarını içermekte ve onları en iyi şekilde karşılaştırmaktadır. Örneğin ARM çekirdeğinin kendi içinde bir çarpıcı modülü bulunmaktadır ve bir çarpım işlemini tek bir çevrimde, bölme işlemini ise 2-12 çevrimde yapabilir [26]. Eğer Horner çarpımını gerçeklersek ve aynı gerçeklemeyi Arm Cortex-M3 için kullanırsak, standart duruma göre daha kötü bir sonuç elde edilmiştir. Bütün bu bakış açılarından dolayı, herhangi bir aygıt üzerinde çalışabilecek hafif bir jenerik algoritma gerçekleştirilmiştir. Platformdan bağımsız gerçekleştirme tarzına rağmen, ölçüm ve sonuçlar olağanüstü derecede iyi olan performansı kanıtlamaktadır. Yapılan çalışma neticesinde en iyi gömülü sistem performansını SHA-3 yarışmasının galibi olan Keccak değil, Blake algoritması vermiştir. Bu sebeple konum mahremiyeti sağlayan protokolün gerçekleştirilmesinde de Blake algoritması kullanılacaktır.

6. PROTOKOLÜN GERÇEKLENMESİ

Önceki kısımda tarif edilen protokol, matematiksel olarak tanımlanmıştır ve yukarıda tarif edildiği haliyle yalnızca teorik bilgi özelliği taşımaktadır. Anlatılan protokolün implemente edilebilmesi için özellikle haberleşme esnasında gönderilen verilen bir çerçeve protokol dâhilinde gönderilmesi gerekmektedir. Gerçeklemeler esnasında veri uygun şekilde paketlenmiştir. Bu kısımda hem tez kapsamında yapılan teorik geliştirmeler, hem de sistemin gerçekleştirilmesi anlatılacaktır.

6.1 Geliştirmeler

Çalışma kapsamında, dördüncü kısımda anlatılan protokol gerçekleştirilmiştir. Ancak konum mahremiyeti sağlaması açısından önerilen protokol birçok anlamda idealleştirmeler içermektedir ve tanımlanan haberleşme adımlarını modelleyerek gerçekleştirme yapmak zorunludur. Bu amaca yönelik olarak bir takım mesaj tipleri tanımlanmış ve bu mesajlar ek bir çerçeve protokol ile gönderilmiştir.

Disiplin olarak haberleşme protokolü tasarımı başlı başına bir konudur ve bir çok önemli unsuru bünyesinde barındırmaktadır. Haberleşme protokollerinin genel amacı elbette ki haberleşme esnasında yaşanabilecek soruların önüne geçmektir. Bu sorunların başında mesajın doğru şekilde aktarılmaması, gönderilen mesajların iç içe geçmesi gibi sorunlar bulunmaktadır. Güvenlik ve mahremiyet işlevlerinin sağlandığı ilk protokolde bu sorunlar, diğer fiziksel etkenlerle birlikte yok sayılmıştır. Bu sebeple gerçekleştirme yapabilmek için Henrici'nin protokolünü, veri olarak taşıyan bir çerçeve protokol tasarlanmıştır. Tüm mesajlar için geçerli olan çerçeve protokol yapısı Şekil 6.1'de görülebilmektedir.

Başlangıç Karakteri	Mesaj Numarası	Veri	Checksum	Bitiş Karakteri
1Byte	1Byte	nByte	1Byte	1Byte

Şekil 6.1 : Çerçeve protokolün yapısı.

Çerçeve protokolde görüleceği üzere başlangıç ve bitiş karakterleri seçilmiştir. Her ne kadar tasarlanan protokol ikili yapıda olsa da, alınan mesajların birbirinden ayrılabilmesi için mesaj başlangıçlarının ve mesaj bitişlerinin bilinmesi gerekmektedir. Mesaj başlangıcının ve mesaj bitişinin anlaşılmasının pratikte çok sayıda yolu vardır. Mesajları donanımsal olarak senkronlamak bu methodlardan biridir. Bu yöntemde mesaj başlangıcında bir pin elektriksel olarak lojik1 (veya 0) seviyesine çekilir. Benzer şekilde mesaj bitişinde pin elektriksel olarak lojik 0 (veya 1) değerine çekilerek mesajın bittiği anlatılabilir. Ancak RFID haberleşmesinde bu tür bir imkânın olması mümkün değildir. Bu sebepten ötürü başlangıç ve bitiş yazılımsal olarak belirtilmelidir. Elbette ki bu yazılımsal tanımlar RFID haberleşmesi esnasında elektromanyetik alanın seviyesindeki değişim ile fiziksel karşılık bulmaktadır ancak senkronizasyon için ayrı bir kanal kullanılmamaktadır. Başlangıç ve bitiş karakterleri için birer byte'lık alan ayrılmıştır ve paket olabildiğince minimalist tutulmak istendiğinden bu değer seçilmiştir. Başlangıç karakteri olarak '*', bitiş karakteri olarak '#' seçilmiştir. Ascii tablosunda '*' karakteri 42 sayısına, '#' karakteri ise 35 sayısına karşılık düşer. Bu karakterler keyfi olarak seçilmiştir ve mevcut paket yapısında, paketin diğer kısımlarında bu karakterler tekrarlanabileceğinden paketin hatalı değerlendirilmesi olasılığı bulunmaktadır. Ancak paketin hatalı değerlendirilmesi durumu için de veri tamlığı kontrolü ile önlem alınmıştır. Paketin başlangıç ve bitişlerinin hatalı değerlendirilmemesi için, başlangıç ve bitiş karakterleri, karakter yerine karakter dizisi seçilebilir. Ancak bunun bir de götürüsü vardır ki; eklenen her bir fazladan karakter hem okuyucu hem de etiket tarafında fazladan işlemler yapılmasına sebep olacak ve sistemin çalışma hızını azaltacaktır. Bu sebepten dolayı mesajın, başlangıç ve bitişinin hatalı değerlendirilme olasılığı $\frac{1}{65536}$ olmaktadır ve bu oran kabul edilmiştir.

Verinin doğru aktarıldığının anlaşılabilmesi için veri tamlık kontrolü gereklidir ve bu amaca yönelik olarak çerçeve protokol dahilinde "checksum" hesaplanmaktadır. Checksum değeri, başlangıç ve bitiş karakterlerinin arasındaki tüm verilerin (her bir byte) XOR'lanması ile hesaplanır. Veri doğruluğunun kontrolü için "checksum" kullanmak mecburi değildir. Buna alternatif birçok yöntem bulunmaktadır ve bunların en bilineni CRC yöntemidir. İstendiği takdirde çerçeve protokolde, doğrulama yöntemi olarak CRC de seçilebilir ancak basitlik olması açısından protokol tasarımı veri doğrulama tercihi "checksum" yönteminden yana kullanılmıştır.

Veri alanına, Henrici tarafından öne sürülen protokoldeki mesajlar yerleştirilir. Bu mesajların her biri ileriki kısımda detaylıca anlatılacaktır. Her bir mesaj tipinin uzunluğu sabittir. Bu nedenle mesaj numarası yardımıyla, çözümlenecek mesajın uzunluğu anlaşılabilmektedir. Bu nedenle çerçeve protokolde uzunluk bilgisi göndermeye gerek yoktur. Bu da paketin minimalist tasarımının bir parçasıdır ve asgari sayıda veri gönderilmesi prensibiyle bu yapı oluşturulmuştur.

Çerçeve protokolde belirtilen mesaj numarası kısmında yer alan mesaj tanımları ve numaralandırmaları Çizelge 6.1’deki gibi yapılmıştır. İleriki kısımda mesaj tipleri tek tek detaylı olarak anlatılacaktır.

Çizelge 6.1 : Mesaj tanımları.

Mesaj Adı	Mesaj numarası
Etiketden okuyucuya kuyruk isteği	1
Okuyucudan etikete kuyruk cevabı	2
Etiketden okuyucuya bağlantı isteği	3
Okuyucudan veritabanına bağlantı sorgusu	4
Veritabanından okuyucuya bağlantı cevabı	5
Okuyucudan etikete bağlantı cevabı	6

6.1.1 Etiketden okuyucuya kuyruk isteği

Bu mesaj tipi, haberleşmenin en başında okuyucunun etiketlerin varlık yokluk durumunu anlaması işleminde kullanılır. Bu süreçte etiket, sürekli olarak rastgele bir değeri okuyucuya gönderir. Okuyucu ile etiket arasındaki haberleşme kanalının dinlenebilir durumda olduğu varsayılmaktadır. Yani bu hat güvenli bir hat olmak zorunda değildir. Bu süreçte etiketin gönderdiği mesajın yapısı tam olarak Çizelge 6.2’deki gibidir.

Çizelge 6.2 : 1 nolu mesajın yapısı

Başlangıç karakteri	Mesaj Numarası	Veri	Veri doğrulama	Bitiş Karakteri
‘*’	1	RND	CS	‘#’
1 Byte	1 Byte	4 Byte	1 Byte	1 Byte

Çizelge 6.2’de görüldüğü üzere, bu mesaj tipinin veri uzunluğu mevcut durumda sabittir ve 8 byte olmaktadır. Yöntem olarak, verinin tam anlamıyla doğru gidip gitmediğini belirleyebilmek için önceden de değinildiği üzere XOR tabanlı basit bir algoritma tercih edilmiştir. Ancak bu alanda CRC8 gibi bir algoritma da kullanılabilir. Etiket eğer pasif olarak tasarlanırsa, okuyucunun alanına girdiği anda,

aktif olarak tasarlanırsa da periyodik ya da aperiodyik aralıklarla bu mesajı yayınlar. Bu mesaj haberleşmenin ilk adımıdır ve etiket bu mesajı yayınlandıktan sonra okuyucudan bir cevap bekler.

6.1.2 Okuyucudan etikete kuyruk cevabı

Etiketden okuyucuya gönderilen kuyruğa alınma isteği, bir okuyucu bu isteği onaylayana kadar devam eder. Haberleşme mesafesinde bir okuyucu olduğunda etiketten alınan bu mesaj, okuyucu tarafından mesajın numarası değiştirilerek ve mesaja etiketin kimliği eklenerek cevap olarak geri gönderilir. Etiket gönderdiği bilgilerden etiketin kimliği elbette ki anlaşılamamaktadır ancak veritabanındaki kayıtlar ikili ağaç gibi bir yöntemle denenecek etiket tekilleştirilmektedir. Bu bir tarama işlemi gibi düşünülebilir ve etiketten cevap gelene kadar devam eder. Gönderilen bu mesaj tam olarak Çizelge 6.3'teki gibidir.

Çizelge 6.3 : 2 nolu mesajın yapısı

Başlangıç karakteri	Mesaj Numarası	Veri	Veri doğrulama	Bitiş Karakteri
'*'	2	H(ID)	CS	'#'
1 Byte	1 Byte	32 Byte	1 Byte	1 Byte

Mesajdaki özet fonksiyonunun çıktısı 32 byte olmak zorunda değildir. Ancak bu çalışma dâhilinde, gerekli güvenlik isterlerinin karşılanması dolayısıyla 32 byte uzunluğu yeterli görülmüştür. Mesajda yayınlanan bilgi aslında etiketin çerçeve protokolü bildiği bilgisidir. Yayınlanan bu özet değeri, özet fonksiyonunun tanımları gereği özgündür ve bu değer kullanılarak etiket kimliğinin doğrudan anlaşılabilmesi oldukça zordur. İlk bakışta başka bir cihazın etiketi taklit etmesiyle konum mahremiyetinin ihlal edilebileceği düşünülebilir ancak protokolün anlatıldığı bölüme değinildiği üzere, haberleşme başarılı olduğunda kimlik de dâhil olmak üzere bir çok parametre değişmektedir ve böyle bir güvenlik açığı söz konusu olmamaktadır. Bu mesajın ardından etiket, ortamda bir okuyucu olduğunu anlar ve haberleşme aşamalarına devam eder. Bu kapsamda etiket ilk olarak TID değişkenini bir artırır ve bir sonraki mesajı okuyucuya gönderir.

6.1.3 Etiketden okuyucuya bağlantı isteği

Okuyucudan kuyruk cevabı aldığı anda etiket Çizelge 6.4'teki mesajı okuyucuya iletir.

Çizelge 6.4 : 3 nolu mesajın yapısı

Başlangıç karakteri	Mesaj Numarası	Veri				Veri doğrulama	Bitiş Karakteri
'*'	3	H(ID)	DBID	H(TID°ID)	ΔTID	CS	'#'
1	1	32	L	32	4	1	1
Byte	Byte	Byte	Byte	Byte	Byte	Byte	Byte

Mesajda, tekilleştirilmiş etiketin kimliğinin özet değerinin yanısıra, veri tabanı kimliği, etiketin TID değerinin etiketin kimliği ile XOR işlemine tabi tutulmuş hali ve ΔTID değişkeni gönderilmektedir. Bu verilerin neden ve hangi işlemler neticesinde gönderildiği protokolün tanımlandığı dördüncü bölümde detaylıca anlatılmıştır.

Mesajda görüldüğü üzere, mesajın uzunluğu sabit değildir. Mesajın uzunluğunun sabit olmama sebebi ise, DB-ID değişkeninin uzunluğunun belli olmayışındır. Eğer DB-ID'nin byte cinsinden uzunluğuna L byte dersek, bu mesajın toplam uzunluğu L+72 byte olur. Mesaj çerçevesinde herhangi bir uzunluk bilgisi gönderilmemektedir ancak başlangıç ve bitiş karakterleri bellidir. Buna göre alınan mesaj uzunluğu X ise, X-72 işlemi ile DB-ID'nin de uzunluğu bulunabilmektedir. Bu durumda yine mesaj dâhilinde uzunluk bilgisinin gönderilmesine gerek kalmamıştır.

6.1.4 Okuyucudan veritabanına bağlantı isteği

Etiketten haberleşme isteği geldiğinde okuyucu, aşağıdaki bilgiler ile veri tabanına istek yapar. Bu mesajın amacı, etiketin veri tabanında var olup olmadığının anlaşılmasıdır. İlgili mesaj Çizelge 6.5'te gösterilmiştir.

Çizelge 6.5 : 4 nolu mesajın yapısı

Başlangıç karakteri	Mesaj Numarası	Veri			Veri doğrulama	Bitiş Karakteri
'*'	4	H(ID)	H(TID°ID)	ΔTID	CS	'#'
1	1	32	32	4	1	1
Byte	Byte	Byte	Byte	Byte	Byte	Byte

Bu mesajın ardından etiket, veritabanı ilgili bilgileri kullanarak, mesajdaki bilgilere sahip olan bir kayıt olup olmadığını kontrol eder.

6.1.5 Veritabanından okuyucuya bağlantı cevabı

Etiketten haberleşme isteği geldiğinde okuyucu veri tabanına istek yapar. Bu mesajın amacı, etiketin veri tabanında var olup olmadığının anlaşılmasıdır. Bu mesajın

ardından eğer veritabanında ilgili bilgilere karşılık düşen bir satır yani etiket kaydı varsa, veritabanı okuyucuya kayıt bilgilerinin varlığını doğrulamak amacıyla Çizelge 6.6'daki mesajı gönderir.

Çizelge 6.6 : 5 nolu mesajın yapısı

Başlangıç karakteri	Mesaj Numarası	Veri	Veri doğrulama	Bitiş Karakteri
'*'	5	RND	H(RND°TID°ID)	CS
1	1	4	32	1
Byte	Byte	Byte	Byte	Byte

Bu esnada görüldüğü üzere bir rastgele sayı pakete eklenmektedir. Bu rastgele sayının üretilmesiyle ilgili birçok yöntem mevcuttur ancak bu konu, çalışmamızın odak noktası değildir. Buna rağmen rastgele sayının rastgeleliği muhakkak çalışmanın kalitesini etkilemektedir. Yapılan gerçekleştirme bu rastgele sayı, mikrodeneteyicinin zamanlayıcısı kullanılarak üretilmiştir.

6.1.6 Okuyucudan etikete bağlantı cevabı

Okuyucudan gelen bağlantı sorgusundaki bilgiler, veri tabanındaki bir etiketi işaret ediyorsa, veritabanından gelen mesaj aynen etikete iletilir. İlgili mesaj Çizelge 6.7'de görülebilir.

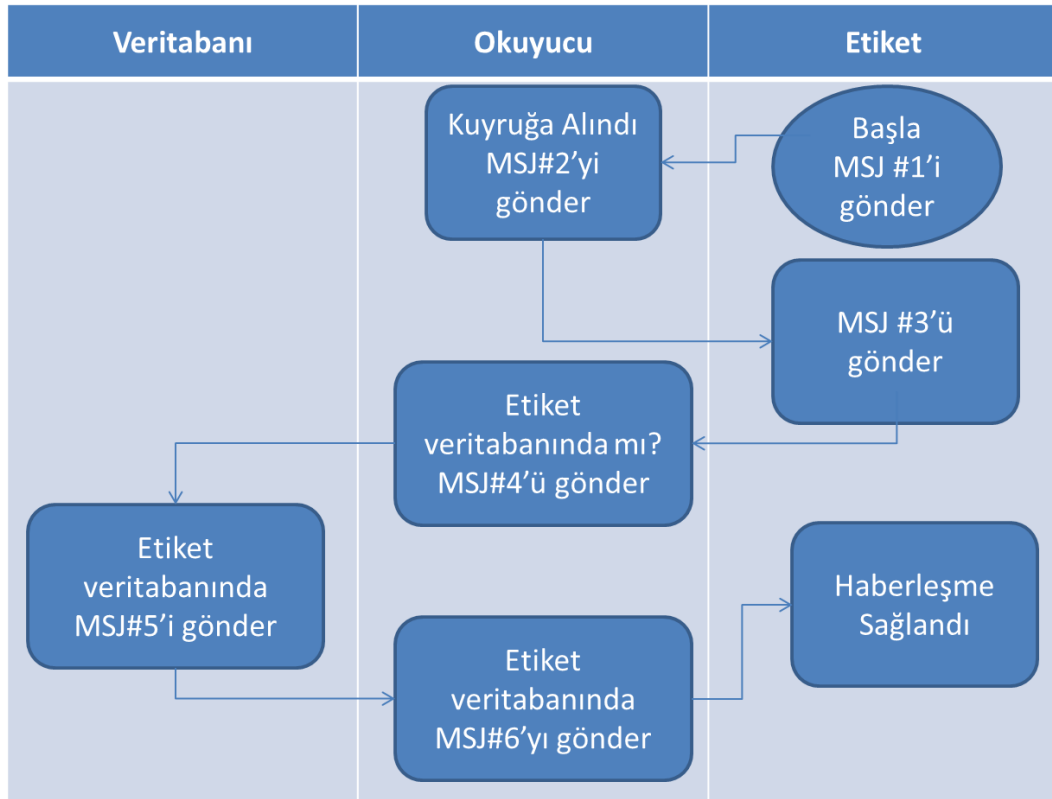
Çizelge 6.7 : 6 nolu mesajın yapısı

Başlangıç karakteri	Mesaj Numarası	Veri	Veri doğrulama	Bitiş Karakteri
'*'	6	RND	H(RND°TID°ID)	CS
1	1	4	32	1
Byte	Byte	Byte	Byte	Byte

Bu aşama tamamlandıktan sonra, konum mahremiyeti sağlanmış olur ve etiket ile okuyucu benzer şekilde haberleşmeye devam edebilir.

6.2 Gerçekleme Ortamı

Pratikte tüm sistemin akış diyagramını Şekil 6.2'deki gibi özetlemek mümkündür:

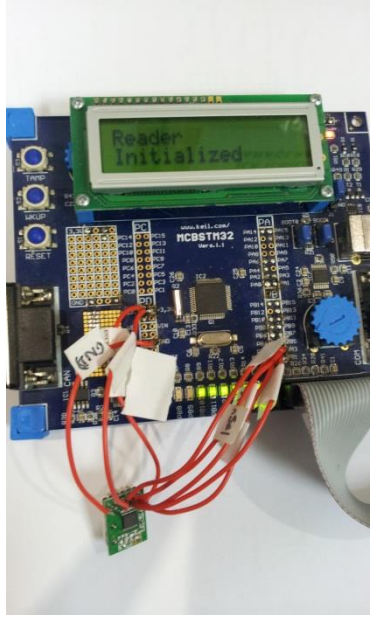


Şekil 6.2 : Gerçeklenen sistemin akış diyagramı.

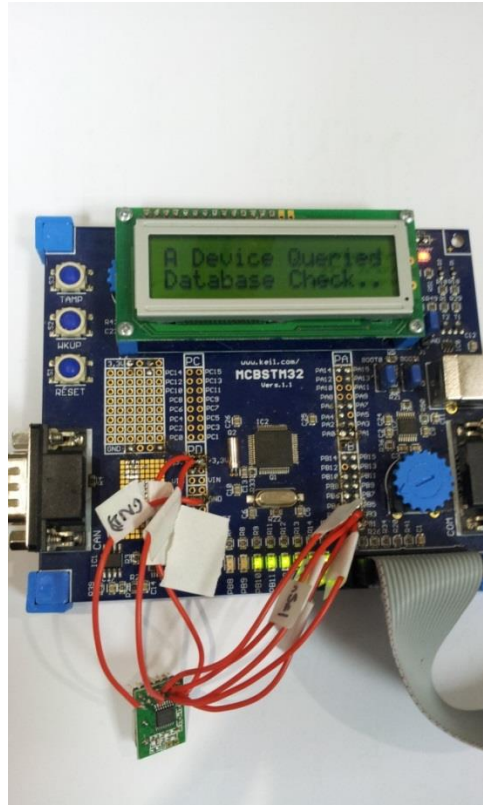
Şekil 6.2’de veritabanı, okuyucu ve etiket arasındaki etkileşim de görülebilmektedir. Çizimde kolaylık olması açısından mesaj numaraları “MSJ#” ön sembolü ile gösterilmiştir. Oklar takip edildiğinde son adım olan haberleşmenin sağlandığı durumda konum mahremiyeti sağlanmıştır ve gönderilecek ek bilgiler Şekil 6.1 ile tanımlanmış genel çerçevede, yeni bir mesaj numarası ile gönderilebilmektedir.

Gerçeklemeye ilişkin fotoğraflar yine araştırmanın bu kısmında yer almaktadır. Sistem implementasyonunda etiket Msp430 ile okuyucu ise Stm32 ile gerçekleştirilmiştir. Okuyucu kendi oturum değişkenlerini yükler ve ilk açılış konfigürasyonunu tamamlar. İlgili duruma ilişkin fotoğraf Şekil 6.3’tedir.

Okuyucu bir süre başlangıç konumunda bekler. Bu durumdayken aktif etiketimizi okuyucunun menziline soktuğumuzda etiket ile okuyucu arasında haberleşme başlar. Okuyucu etiketi hemen kabul etmez, öncelikli olarak protokolde tarif edilen haberleşme safhaları gereklidir. Etiket okuyucunun menziline soktuğumuzdaki durum Şekil 6.4’deki gibi olur.

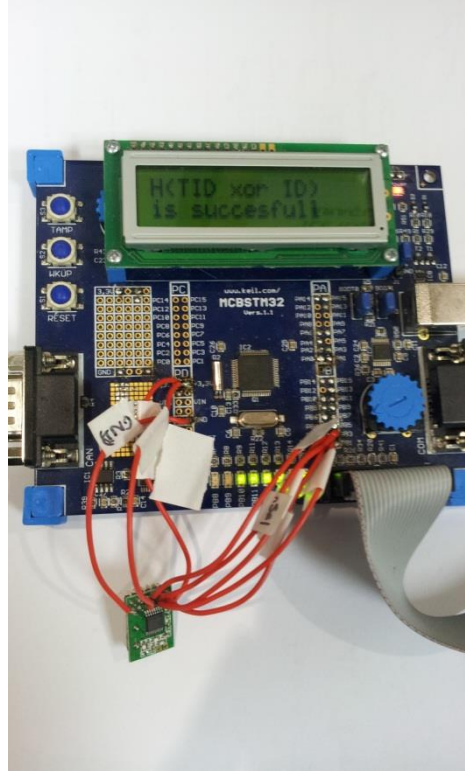


Şekil 6.3 : Okuyucu çalışmaya başlarken.



Şekil 6.4 : Okuyucu bir etiketi kuyruğa eklerken.

S Şekil 13'teki durumda, etiketin kuyruğa alındığı ve veritabanında etiketten alınan bilgilerin doğrulandığı aşama görülmektedir. Bir etiket kuyruğa alındığında TID değerini bir artırmaktadır. Bu adımdan sonra H(ID), DB-ID ve $H(TID \cdot ID)$ değerleri beklenmiş ve alınmıştır. İlgili duruma ilişkin fotoğraf Şekil 6.5'tedir.

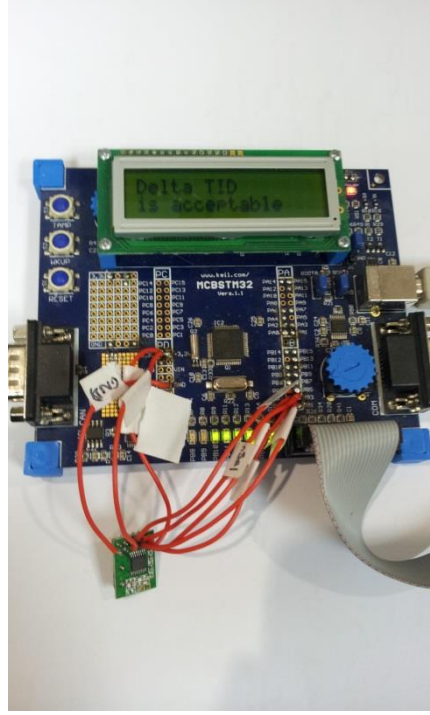


Şekil 6.5 : Birinci mesaj okuyucu tarafından alınırken.

Bu aşamada okuyucu etiketten aldığı H(ID), DB-ID ve H(TID $\circ$ ID) bilgilerini sırasıyla doğrulamaktadır. Veri dizisi içinde bulunan H(TID $\circ$ ID) parametresi tekrarlı ataklara karşı alınmış bir önlemdir ve TID değeri değişimi ile etiketlerin tekrarlı ataklarla izlenmesini önlemeyi amaçlar. Δ TID değerini alan okuyucu bundan yararlanarak TID değerini hesaplar ve doğrular. Δ TID değerinin 1 olması, etiketi taklit etmeye çalışan biri olmadığı konusunda bilgi verir.

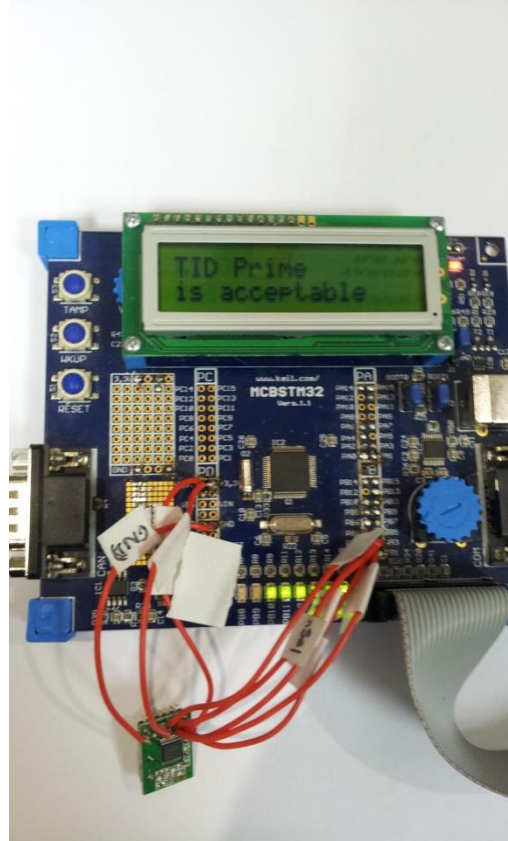
Netice itibariyle bir sonraki adıma geçebilmek için alınan değerlerin doğrulanması gerekmektedir. Bu doğrulamaların gerçekleşebilmesi için etiketteki ID, DB-ID ve TID parametrelerinin okuyucudaki ile aynı olması gerekmektedir. Buna ek olarak her iki cihazın üzerinde çalışan özet fonksiyonu implementasyonunun da aynı olması gereklidir.

Bu aşama tamamlandıktan sonra bir sonraki aşamaya geçilir. Sonraki adımda Δ TID beklenmiş ve alınmıştır. İlgili duruma ilişkin fotoğraf Şekil 6.6'dadır.



Şekil 6.6 : Δ TID okuyucu tarafından alınırken.

Bir sonraki adımda ise yeni TID' beklenir ve alınmıştır. İlgili duruma ilişkin fotoğraf Şekil 6.7'dedir.



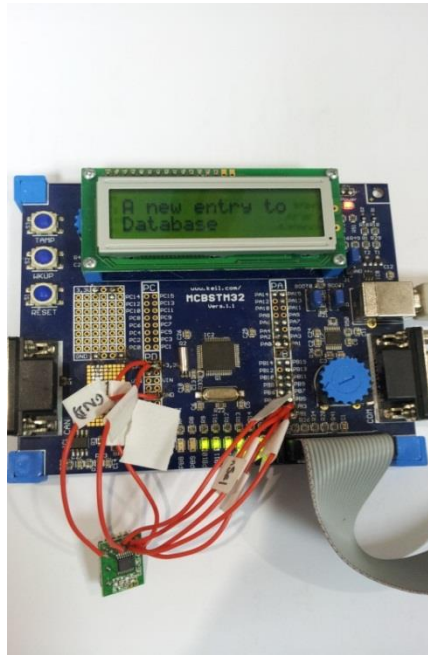
Şekil 6.7 : Yeni TID okuyucu tarafından onaylanırken.

Şekil 6.7’deki durum neticesinde etiket doğrulanarak bir sonraki aşama olan veri tabanı işlemlerine geçilmiştir. Okuyucu etiketten aldığı bilgileri tek başına değerlendiremeye muktedir değildir. Bunun sebebi, kayıtların tamamının veritabanı üzerinde olmasıdır.

Veri tabanında, $HID=H(ID)$ olarak atanır. LST ve ΔTID ’ler biri birine eklenir. Böylece güncel TID^* değeri elde edilir. Bu durumda sonraki adım olarak $h(TID^* \circ ID)$ hesaplanır. Bu değer ilk mesajdaki ($h(TID \circ ID)$) ile eşleşmezse mesaj iptal edilir. Eğer mesaj geçerli ise TID^* ile TID karşılaştırılır. TID^* , TID ’den büyük değilse mesaj yine iptal edilir.

TID^* ve TID hakkında her şey yolunda gitti ise bir sonraki aşamaya geçilebilir. Bu aşamada rastgele bir sayı “RND” üretilecektir. Bu rastgele numara ile yeni ID (ID^*), $ID^* = RND \circ ID$ olacak şekilde tanımlanır. İlgili DB girdisi AE oluşturulmuşsa, kaydın ID alanı ID^* olarak, HID alanı ise $h(ID^*)$ olarak güncellenir. Aksi halde, yeni bir satır ID^* ID ve $h(ID^*)$ HID olacak şekilde oluşturulur. Satırın AE girdisi ilgili alanları günceller ya da tam tersi durum gerçekleşir. TID aynı değerleri alır.

Bir sonraki aşamada, RND ve $h(RND \circ TID^* \circ ID)$ ’yi içeren mesaj oluşturulur ve etikete iletecek olan veritabanından okuyucuya yollanır. Etiket de bu vesileyle kimliğini değiştirmiş olur. Bu duruma ilişkin fotoğraf Şekil 6.8’dedir.

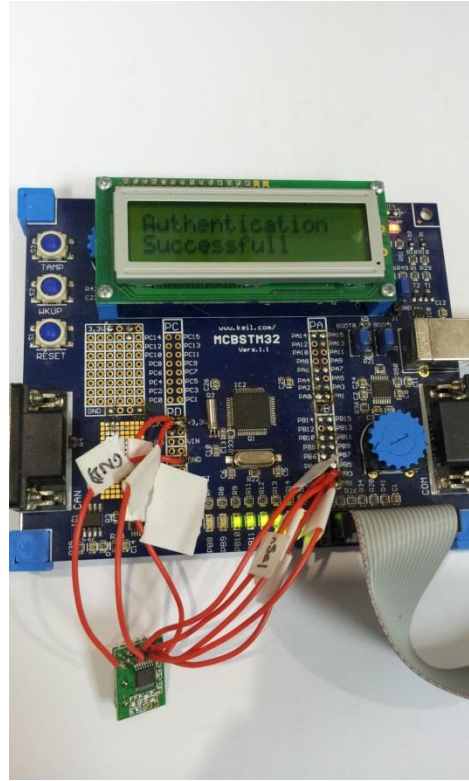


Şekil 6.8 : Veritabanından okuyucuya cevap geldiğinde.

Bir sonraki aşamada, RND ve $H(RND \circ TID \circ ID)$ 'yi içeren mesaj oluşturulur ve etikete iletecek olan veritabanından okuyucuya yollanır.

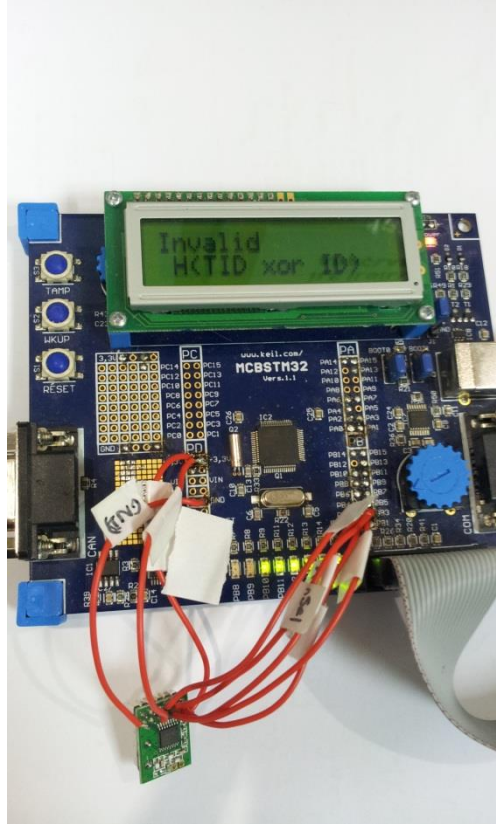
Etiket $H(RND \circ TID \circ ID)$ değerini kontrol eder. Yanlış mesaj ise iptal eder ve başka mesaj kabul etmez. Aksi halde etiket, ID'yi $(RND \circ ID)$ 'nin değerine günceller. Ek olarak LST'yi TID olarak günceller. Artık etiket yeni ID ve $h(ID)$ değerlerine sahip olur ki bundan sonraki okuma girişimlerinde kullanılabilir.

Veri tabanının güncellenmesiyle etkileşim tamamlanır ve etiket konum mahremiyeti ve güvenlik ilkelerine göre tanımlanmış olur. İlgili duruma ilişkin fotoğraf Şekil 6.9'dadır.



Şekil 6.9 : Haberleşme tamamlandığında.

Bununla birlikte sisteme veri tabanında bulunmayan bir etiketle yahut etiketin taklidiyle atak yapılmaya kalkıldığında Şekil 6.10'daki hata ile karşılaşılmaktadır ve neticede etiket kimliklenmemektedir.



Şekil 6.10 : Haberleşmede hata oluştuğunda.

Gerçekleme bir adet okuyucu ve bir adet etiket ile gerçek zamanda da test edilmiştir. Sistem fiziksel olarak konum mahremiyeti sağlamaktadır. Ancak ek olarak, haberleşme frekansının gerçek zamanlı olarak değiştirilmesi, modülasyonun değiştirilmesi gibi ek önlemlerin de alınmasıyla, bu performansı daha üst noktalara taşımak mümkündür.

7. SONUÇ VE ÖNERİLER

Bu Yüksek Lisans tezi kapsamında Radyo Frekansı Tanımlama (RFID) sistemlerinin konum mahremiyeti (location privacy) problemi üzerinde durulmuş ve bu amaca yönelik olarak olarak RFID’de sunulan mevcut protokollere ilişkin güvenlik ve mahremiyet analizleri yapılmış, bunlarda görülen eksiklikler kapatılarak mevcut bir protokolün geliştirilmesiyle gerçeklemeye uygun yeni bir protokol ortaya atılmıştır. Bu protokolün ortaya atılmasındaki temel motivasyon ise konum mahremiyeti sağlayan ve pratik kullanıma uygun bir RFID protokolü ortaya çıkarmak olmuştur ve bu amca ulaşılmıştır. Ortaya atılan yeni protokol, fiziksel olarak mikrokontrolörlerle gerçekleştirilen bir RFID sisteminde analiz edilmiştir. Bu analizlerle de anlaşılabacağı üzere, teorik ve pratik anlamda konum mahremiyeti sağlayan bir RFID protokolu elde edilmiştir.

Bu çalışma öncelikli olarak mahremiyet özelliği sunan protokollerin analizini içermektedir. Sonraki kısımlarda ise fiziksel anlamda da mahremiyet sunan bir protokolün gerçekleştirilmesi için gerekli olan altyapı dâhilinde özet fonksiyonlarında değinilmiştir. Gerçeklenen protokolün en önemli parçalarından biri özet fonksiyonu olduğundan tez kapsamında özet fonksiyonları üzerinde özellikle durulmuştur. SHA-3 adayı olarak gösterilen finalist 5 özet fonksiyonu çeşitli gömülü platformlarda gerçekleştirilmiş ve hem çalışma hızları hem de bellek kullanımları bakımında karşılaştırılmıştır. Bunun neticesinde hem hız hem de bellek kullanımı performansı olarak Blake algoritması seçilmiş, Keccak algoritması SHA-3 standardı olarak seçilmiş olsa da gerçeklemede kullanılmamıştır. Ancak istendiği takdirde kolaylıkla bu değişim de yapılabilir durumdadır çünkü kullanılan tüm platformlar için gerekli olan implementasyonlar yapılmıştır.

Konum mahremiyeti sunan protokolün gerçek hataya uygulanabilir şekilde getirilmesi açısından referans protokolün eksikleri göz önüne alınarak yeni bir çerçeve protokol oluşturulmuştur. Yapılan testler ve deney sonuçları protokolün teorik anlamda da

pratik anlamda da mahremiyet sunduğunu göstermiştir. Sonuç olarak bu çalışma ile ortaya konan çıktının, pratik olarak da kullanılabilir olduğu gösterilmiştir.

7.1 Çalışmanın Uygulama Alanı

Bu çalışma RFID sistemleri üzerinde konum mahremiyetini sağlamak üzere yapılmıştır. Dolayısıyla RFID sistemlerinin kullanıldığı her uygulamada, önerilen protokol de kullanılabilir.

Gerçeklemede kullanılan etiket aktif etikettir ve üzerinde bir besleme kaynağı barındırmaktadır. Bu nedenle yapılan gerçekleştirme kapsamında ortaya konan sistem pasif RFID sistemleri üzerinde test edilmemiştir. Ancak etiket tasarımında kullanılan mikrodenetleyici çok düşük güç tüketimine sahip olduğundan, okuyucu ile etkileşim mesafesi kısa tutularak yüksek miktarda enerji aktarımı yapmak ve etiketi pasif olarak çalıştırmak mümkün olabilir. Bu tez kapsamında böyle bir çalışma yapılmamıştır ancak çalışma altyapısı hazırlanmıştır.

Bu tez kapsamında yapılan çalışma, envanter takibi, otoyol ücretlendirmeleri, akıllı park sistemleri, akıllı geçişler gibi bir çok alanda kullanılabilir durumdadır.

KAYNAKLAR

- A. Infocenter.** *Cortex-M3 instructions*, Technical Publications, adres: <http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0337h/CHDDIGAC.html>. alındığı tarih: 18.02.2013
- Alfred Menezes, Paul C. van Oorschot, and Scott A. Vanstone.** (1996) *Handbook of Applied Cryptography*. Florida: CRC Press.
- B. Glover and H. Bhatt.** (2006). *RFID Essentials*. California: Sebastopol, CA : O'Reilly.
- B.Song.** (2009). *RFID authentication protocols using symmetric cryptography*. (Master's thesis). University of London, Egham, England, Aralık
- D. Henrici and P. Muller** (2004). Hash-Based Enhancement of Location Privacy for Radio-Frequency Identification Devices using Varying Identifiers. *Communications in Computer and Information Science*. 339 (1), sf. 149-153.
- D. J. Bernstein and T. Lange.** (2012). The new SHA-3 software shootout. *Proceeding of the Third SHA-3 Candidate Conference*. 4 (1), sf. 1-13..
- G. Bertoni, J. Daemen, M. Peeters and G. Van Assche.** (2011). The Keccak reference. Submission to NIST, adres: <http://keccak.noekeon.org/Keccak-reference-3.0.pdf> alındığı tarih: 22.04.2013
- G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche.** (2007). Sponge functions, Ecrypt Hash Workshop, adres: http://www.csrc.nist.gov/pki/HashWorkshop/Public_Comments/2007_May.html. alındığı tarih : 22.04.2013.
- H. Stockman.** (1948). *Communication by means of reflected power*. New York: The Institute of Radio Engineers. sf. 1196-1204.
- H. Wu.** (2009). *The hash function jh*. Submission to NIST (güncellendi), adres: http://icsd.i2r.a-star.edu.sg/staff/hongjun/jh/jh_round2.pdf. alındığı tarih : 22.04.2013.
- J. P. Aumasson, L. Henzen, W. Meier, and R. C.-W. Phan.** (2008). Sha-3 proposal Blake. Sub-mission to NIST, adres: <http://131002.net/blake/blake.pdf>. alındığı tarih: 22.04.2013.
- J. W. Bos and D. Stefan.** (2010). Performance analysis of the SHA-3 candidates on exotic multi-core architectures, *Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, 6225(3). sf. 279–293.

- K. Davelaar.** (2011) Analyzing the NIST SHA-3 competition candidates implementing the top cryptographic hash functions in java,. *Proceedings of the 14th Twente Student Conference on IT (TSCONIT)*, 3(1) sf 39-47.
- K. Venkat.** (2006). *Efficient multiplication and division using MSP430*, Application Report (SLAA329), Texas Instruments, Eylül, adres: <http://www.ti.com/lit/an/slaa329/slaa329.pdf>. alındığı tarih: 18.02.2013
- M. Rieback, B. Crispo, ve A. Tanenbaum.** (2006). The evolution of RFID security. *IEEE Pervasive Computing*. 5 (1), sf. 62-69.
- MIT Auto-ID Center.** <<http://www.autoidcenter.org>>. alındığı tarih:28.04.2013.
- N. Ferguson, S. Lucks, B. Schneier, D. Whiting, M. Bellare, T. Kohno, J. Callas, and J. Walker.** (2009). *The skein hash function family*. Submission to NIST (Round 2), adres: <http://www.skein-hash.info/sites/default/files/skein1.2.pdf>. alındığı tarih : 23.04.2013.
- NIST.** (2007). *Announcing request for candidate algorithm nominations for a new cryptographic hash algorithm (SHA-3) family*. Federal Register, 72 (112), adres: http://csrc.nist.gov/groups/ST/hash/documents/FR_Notice_Nov07.pdf.
- P. Gauravaram, L. R. Knudsen, K. Matusiewicz, F. Mendel, C. Rechberger, M. Schlaffer, and S. S. Thomsen.** (2008). *Groestl – a sha-3 candidate*. Submission to NIST, adres: <http://www.groestl.info/Groestl.pdf>. alındığı tarih : 22.04.2013.
- Rankl W.** (2000). *Smart Card Handbook*, 2nd edn. , New York :Wiley, sf. 20-82
- Simson Garfinkel.** (2002). *An RFID Bill of Rights*. Adres: <http://www.technologyreview.com/article/401660/an-rfid-bill-of-rights/>. alındığı tarih: 28.04.2013.
- Weis, S. ve Arkadaşları.** (2003). Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems, *Security in Pervasive Computing*. 2802 (3), sf 201-212.
- Xiaoyun Wang and Hongbo Yu.** (2005). How to Break MD5 and Other HashFunctions. *Advances in Cryptology – EUROCRYPT* . 3494 (1), sf 19-35.
- Xiaoyun Wang, Hongbo Yu, and Yiqun Lisa Yin.** (2005). Efficient Collision search Attacks on SHA-0. *Advances in Cryptology - CRYPTO*. 3621 (1), sf 1-16.
- Xiaoyun Wang, Xuejia Lai, Dengguo Feng, Hui Chen, ve Xiuyuan Yu.** (2005). Cryptanalysis of the Hash Functions MD4 and RIPEMD. *Advances in Cryptology – EUROCRYPT* . 3494 (1), sf 1-18.
- Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu.** (2005). Finding Collisions in the Full SHA-1. *Advances in Cryptology - CRYPTO*. 3621 (1), sf 17-36.

- [1] **B.Song.** (2009). *RFID authentication protocols using symmetric cryptography*. (Master's thesis). University of London, Egham, England, Aralık
- [2] **M. Rieback, B. Crispo, ve A. Tanenbaum.** (2006). The evolution of RFID security. *IEEE Pervasive Computing*. 5 (1), sf. 62-69.
- [3] **H. Stockman.** (1948). *Communication by means of reflected power*. New York: The Institute of Radio Engineers. sf. 1196-1204.
- [4] **Rankl W.** (2000). *Smart Card Handbook*, 2nd edn. , New York :Wiley, sf. 20-82
- [5] **B. Glover and H. Bhatt.** (2006). *RFID Essentials*. California: Sebastopol, CA : O'Reilly.
- [6] **MIT Auto-ID Center.** <<http://www.autoidcenter.org>>. alındığı tarih:28.04.2013.
- [7] **Simson Garfinkel.** (2002). *An RFID Bill of Rights*. Adres: <http://www.technologyreview.com/article/401660/an-rfid-bill-of-rights/>. alındığı tarih: 28.04.2013.
- [8] **Xiaoyun Wang and Hongbo Yu.** (2005). How to Break MD5 and Other HashFunctions. *Advances in Cryptology – EUROCRYPT* . 3494 (1), sf 19-35.
- [9] **Xiaoyun Wang, Xuejia Lai, Dengguo Feng, Hui Chen, ve Xiuyuan Yu.** (2005). Cryptanalysis of the Hash Functions MD4 and RIPEMD. *Advances in Cryptology – EUROCRYPT* . 3494 (1), sf 1-18.
- [10] **Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu.** (2005). Finding Collisions in the Full SHA-1. *Advances in Cryptology - CRYPTO*. 3621 (1), sf 17-36.
- [11] **Xiaoyun Wang, Hongbo Yu, and Yiqun Lisa Yin.** (2005). Efficient Collision search Attacks on SHA-0. *Advances in Cryptology - CRYPTO*. 3621 (1), sf 1-16.
- [12] **Alfred Menezes, Paul C. van Oorschot, and Scott A. Vanstone.** (1996) *Handbook of Applied Cryptography*. Florida: CRC Press.
- [13] **NIST.** (2007). *Announcing request for candidate algorithmnominations for a new cryptographic hash algorithm (SHA-3) family*. Federal Register, 72 (112), adres: http://csrc.nist.gov/groups/ST/hash/documents/FR_Notice_Nov07.pdf.
- [14] **J. P. Aumasson, L. Henzen, W. Meier, and R. C.-W. Phan.** (2008). Sha-3 proposal Blake. Sub-mission to NIST, adres: <http://131002.net/blake/blake.pdf>. alındığı tarih: 22.04.2013.
- [15] **P. Gauravaram, L. R. Knudsen, K. Matusiewicz, F. Mendel, C. Rechberger, M. Schlaffer, and S. S. Thomsen.** (2008). *Grøstl – a sha-3 candidate*. Submission to NIST, adres: <http://www.groestl.info/Groestl.pdf>. alındığı tarih : 22.04.2013.
- [16] **H. Wu.** (2009). *The hash function jh*. Submission to NIST (güncellendi), adres: http://icsd.i2r.a-star.edu.sg/staff/hongjun/jh/jh_round2.pdf. alındığı tarih : 22.04.2013.

- [17] **N. Ferguson, S. Lucks, B. Schneier, D. Whiting, M. Bellare, T. Kohno, J. Callas, and J. Walker.** (2009). *The skein hash function family*. Submission to NIST (Round 2), adres: <http://www.skein-hash.info/sites/default/files/skein1.2.pdf>. alındığı tarih : 23.04.2013.
- [18] **G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche.** (2007). *Sponge functions*, Ecrypt Hash Workshop, adres: http://www.csrc.nist.gov/pki/HashWorkshop/Public_Comments/2007_May.html. alındığı tarih : 22.04.2013.
- [19] **G. Bertoni, J. Daemen, M. Peeters and G. Van Assche.** (2011). *The Keccak reference*. Submission to NIST, adres: <http://keccak.noekeon.org/Keccak-reference-3.0.pdf> alındığı tarih: 22.04.2013
- [20] **D. Henrici and P. Muller** (2004). Hash-Based Enhancement of Location Privacy for Radio-Frequency Identification Devices using Varying Identifiers. *Communications in Computer and Information Science*. 339 (1), sf. 149-153.
- [21] **Weis, S. ve Arkadaşları.** (2003). Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems, *Security in Pervasive Computing*. 2802 (3), sf 201-212.
- [22] **D. J. Bernstein and T. Lange.** (2012). The new SHA-3 software shootout. *Proceeding of the Third SHA-3 Candidate Conference*. 4 (1), sf. 1-13..
- [23] **K. Davelaar.** (2011) Analyzing the NIST SHA-3 competition candidates implementing the top cryptographic hash functions in java,. *Proceedings of the 14th Twente Student Conference on IT (TSCONIT)*, 3(1) sf 39-47.
- [24] **J. W. Bos and D. Stefan.** (2010). Performance analysis of the SHA-3 candidates on exotic multi-core architectures, *Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, 6225(3). sf. 279–293.
- [25] **K. Venkat.** (2006). *Efficient multiplication and division using MSP430*, Application Report (SLAA329), Texas Instruments, Eylül, adres: <http://www.ti.com/lit/an/slaa329/slaa329.pdf>. alındığı tarih: 18.02.2013
- [26] **A. Infocenter.** *Cortex-M3 instructions*, Technical Publications, adres: <http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0337/h/CHDDIGAC.html>. alındığı tarih: 18.02.2013

EKLER

EK A: Bilgisayar Programları ve Yazılımlar CD'si

ÖZGEÇMİŞ

VESİKALIK
FOTO

Ad Soyad: Özen Özkaya

Doğum Yeri ve Tarihi: Kocaeli, 10 Ekim 1988

Adres: İstasyon Mh. 1447 Sok. No:13/3 Gebze, Kocaeli

E-Posta: ozenozkaya@gmail.com

Lisans: İstanbul Teknik Üniversitesi, Elektronik Mühendisliği

Yayın ve Patent Listesi:

- Z. Iscan, Ö. Özkaya, Z. Dokur. (2011). Classification of EEG in A Steady State Visual Evoked Potential Based Brain Computer Interface . *10th International Conference, ICANNGA* . 6594 (2), p81-88.