University of Mosul

College of Electronic Engineering

FPGA Implementation of Viterbi Algorithm for IoT Applications

A Thesis Submitted by

Hiba Saad Mahmood

To

The Council of College of Electronic Engineering

University of Mosul

In Partial Fulfillment of the Requirements

For the Degree of Master of Sciences

In

Computer and Information Engineering

Supervised by

Assistant Prof. Mohammed Hazim Al-Jammas

2019 A.C. 1440 A.H.

بسم الله الرحمن الرحيم

﴿ اقْرَأْ بِاسْمِ رَبِّكَ الَّذِي خَلَقَ * خَلَقَ
الإِنْسَانَ مِنْ عَلَقٍ * اقْرَأْ وَرَبُّكَ الأَكْرَمُ *
الَّذِي عَلَّمَ بِالْقَلَمِ * عَلَّمَ الإِنْسَانَ مَا لَمْ
يَعْلَمْ ﴾

العلق: ١-٥

Supervisor's Certification

I certify that the dissertation entitled (**FPGA Implementation of Viterbi Algorithm for IoT Applications**) was prepared by **Hiba Saad Mahmood** under my supervision at the Department of Computer and Information Engineering, University of Mosul, as a partial requirement for the Master of Science Degree in Computer and Information Engineering.

Signature:

Name: Dr. Mohammed H.Al-Jammas

Department of Computer and Information Engineering

Date: / /2019

Report of Linguistic Reviewer

I certify that the linguistic reviewer of this dissertation was carried out by me and it is accepted linguistically and in expression.

Signature:

Name:

Date: : / /2019

Report of the Head of Department

I certify that this dissertation was carried out in the Department of Computer and Information Engineering. I nominate it to be forwarded to discussion.

Signature:

Name: Dr .Abdulbary Raouf Suleiman

Date: / /2019

Report of the Head of Postgraduate Studies Committee

According to the recommendations presented by the supervisor of this dissertation and the linguistic reviewer, I nominate this dissertation to be forwarded to discussion.

Signature:

Name:

Date: / /2019

Acknowledgments

All praises to Almighty Allah, Who has induced me with intelligence, knowledge, sight to observe and mind to think and given me enough strength and health to perform and accomplish my scientific project successfully.

I would like to take this opportunity to express my deep sense of gratitude and thank to my supervisor **Dr.Mohammed H. Al-Jammas** for offering me help and guidance.

My appreciation is extended to the Head of the Computer and Information Engineering Department.

Finally, I would gratefully like to express my gratitude and love feeling to my family and dearest husband for their encouragement, moral support, and patience and for the many sacrifices during the course of the study.

Table of Contents

Chapter Three

Simulation of Viterbi Decoder

Chapter Four

FPGA Implementation of Viterbi Decoder

List of Table

List of Figures

List of Abbreviations

Abbreviation	Name
ACSU	Add Compare Select Unit
AWGN	Additive White Gaussian Noise
BER	Bit Error Rate
BMU	Branch Metric Unit
FPGA	Field Programmable Gate Array
IoT	Internet of Thing
ISE	Institute Of Software Engineers
TBU	Trace Back Unit
VD	Viterbi Decoder
VHDL	VHSIC Hardware Description Language
VD	Viterbi Decoder

Abstract

In communication systems, Viterbi decoder that is use to decode the convolutional codes is consider as a powerful error-correcting technique to control and correct the errors during transmission over a noisy channel.

When data is transmit in Internet of Thing (IoT) applications over wireless communication channels, it is possible to be subject to noise, distortion and interference. This leads to various amounts of errors and data corruption at the receiver side. The 802.11ah wireless transmission standard is adopt as wireless communication protocol in IoT applications.

The Viterbi decoder implementation has a trade-off between performance from one side, and the design complexity and decoding time from the other side. In this thesis, Viterbi decoder designed and implemented using Matlab simulations, and FPGA design kit to evaluate and compare the performance of this decoder at various specifications and settings. Using Matlab (2013a), Viterbi decoder performance is tested for five encoding models at constraint lengths of 3, 4, 5, 6 and 7, using a code rate of 1/2. This is to perform a comparison between these models and evaluate the performance of each design. This performance evaluation is measure using the decoding time and BER at various SNR levels. Results show that constraint lengths of 6 and 7 have similar BER which is higher than other implemented lengths, with length 6 having less complexity and less decoding time by 4.7ms and 3.5ms in hard and soft types, respectively.

The FPGA implementation used NEXYS 4DDR kit involving Viterbi decoder design at constraint lengths of 6 and 7 in both hard and soft decoding. In addition to the lower complexity and decoding delay time, length 6 shown to have lower resource utilization. Therefore, the

requirements of IoT applications error correction are best achieve with Viterbi decoder with length 6, providing high data receiving performance, with the least possible data loss.

Chapter1
Introduction

1.1 Background

Internet of Thing (IoT) is one of the most important field in the world [1]. It is a new technology for access to the internet, the all objects in IoT can access and use information that collected by other object [2]. The concept of IoT that every devices merging with the human existence, the physical object can intelligently communicate with each other to carry out daily operations. Every device can communicate with other, respond intelligently, transfer data, and retrieve data [3]. The IoT applications need new methods and theories to support their requirements. The important requirements is the communication protocol that have long-range, low data rate, and low-power, thus the characteristic of 802.11ah standard can supports and meet the requirements of the IoT applications [1]. In the wireless transmission for 802.11ah standard, the convolutional encoding is important method providing error-correcting codes.

1.2 Application of IoT

IoT is not only a compatibility theory. It is a useful application technology for human lives. IoT works to turn lifestyle into a better one. From a few years, IoT technology has spread around the world and has entered into a variety of lifestyles to move beyond the applications of smartphones, to make human life simpler and more comfortable in various aspects of life with the spread of modern communication technologies. The importance of IoT in our lives comes from widely applications fields. It seems to be an interconnected set of smart home applications, wearable and smart industry components but in reality, it could have a wider range. When the world connected every things reality, IoT will turn everything to be smart from home to hospital, transportation and smart industry. Implement solutions for these applications will be the main driver of the invention. Currently, there are several successful applications have already been developed in different areas such as: [4].

1.2.1 Smart City

The smart city makes human life easier and more convenient, such as solving traffic congestion, making cities safer and reducing pollution. Many smart city applications such as traffic regulation, intelligent water distribution, efficient use of energy and intelligent transportation.

1.2.2 Industrial Internet

The industrial internet can be seen as an IoTs concept by many market research like Gartner or Cisco and has high-end possibilities. The industrial internet applications include intelligent factories that work with smartly connected equipment. In 2014, General Electric (GE's) revenues from industrial internet products totaled nearly 1 billion $.

1.2.3 Smart Home

IoT is an innovative technology that makes our life smarter [5]. In many field the word "smart" is used and accepts in a wide range to mean intelligence [6]. You cannot imagine the idea of a connected world without IoT. An example is the smart home based IoT [5]. With the emergence of IoTs, and the rapid evolution of internet technology and communications, a smart home with high computing and communications features has emerged. Smart home based on IoT concept are being an important part of the smart cities that are design and develop in over the worlds [5]. Intelligent home service, one of the innovative technologies in the age of IoT, is changing home appliances to become smarter, remotely controllable and communicate to each other, as well as, the level of intelligence or control of the smart house service that the user wants may vary depending on the user requirements [6]. Smart home or connected home is the environment for living with highly automated and controlled systems. The goal of the smart home to improving the standard of living, safety and security in addition to providing resources and energy and it is playing an important role in improvement society [5]. Smart home is suitable place to describe communication in our home with strong automation systems that can monitor and control for example thermostats, smoke detectors, light bulbs, home appliances, door locks, entertainment systems, windows, and many more. famous companies in the design of smart home Nest, Apple, Phillips and Belkin. So the IoT has a distinctive role in building a smart home [5].

1.2.4 Medical and Health Care

With the emergence of the concept of internet things and the evolution of information technology and medical device technologies, the concept of the medical interne has gradually become integrate with the people life's. Medical IoT is a kind of technology that connects wireless sensors of medical devices with the internet and connects them with hospitals, patients and medical device to improve modern medical level. It can make hospitals capable of achieving smart medical treatment and management that includes collect, store, transmit and share medical information about equipment, personnel information, management information for the hospital and information on medicines. The medical IoT includes many applications such as: management and monitor of medical equipment and medicines, telemedicine care, personal medical care, mobile medical care, health care management and many more [7].

1.2.5 Transportation

Smart transportation system offers intelligent control, and make more successfully management for transportation use advanced sensor information technology and network. Smart transportation have various applications in many aspects of daily life such as: [8]

- **Smart Parking**

  Smart parking management provides the best solution for drivers, saving fuel and time, where offers accurate information on available parking spaces, making traffic easy and reducing traffic jams.

- **3D Assisted Driving**

  Vehicles that embedded with sensors such as cars, buses and trains can provide vehicle driver with useful information on navigation and safety. Using driving with three-dimensional help, drivers can set the right track with the prior knowledge about situation of jams and traffic accidents.[4]

1.2.6 Smart Agriculture

As the world's population continues to increase, the demand is very high for food supplies, with the future direction to smart ideas by developed technologies replacing smart applications (automation), the emergence of IoT made intelligent agriculture one of important application in IoT. Farmers use sensors to measure soil moisture and nutrients, to determine the specific fertilizers and to control the use of water for plant growth. These parameters are one of the simplest applications used in the smart agriculture. The data from sensors is using to achieve the better production of crop. One of the application in this field is automatic irrigation based on measure some environment parameters such as temperature, air humidity measurement and soil information in real time and sending to server to be analyzed. According to the results, automatic irrigation is controlled and improving agricultural crop. [9][10]

1.3 Challenges of IoT

IoT are connect "things" between the real world and the virtual world, allowing communication at anytime and anywhere for anything and not just anyone. It is allowing every beings, machines, physical object in the world connected, and interact together with peoples in any place and any time to providing services [11]. These objects embedded with sensors connected to internet by a wired or wireless network [2]. Millions of smart devices and sensors spreading daily in every aspect of our life creating better understanding of how we interact with our environment, at home, in the city, in office, etc. That make life more easy which have clearly affected our life in terms of costs reduction, energy saving and improving customer service [12]. Although the above applications are very interesting and offer smart technologies for everything, some challenges facing applications of IoT, including the IoT applications implementation cost, which is expected to be at low cost for the large possible number of things. In addition to the cost challenge, there are other challenges such as [2]:

- **Scalability:** IoT have a greater scope than conventional internet. Where needs new functionality to increase scalability and enable things that collaborate in an open environment to make functions equally efficiently.
- **Security and Privacy:** The Internet requirements in security and privacy aspects such as the reliability of communication, the integrity of the message and the credibility of communication companies. IoT also need these requirements, devices of IoT must been protected from hackers.
- **Fault Tolerance**: In the IoT, the objects have to be more flexible and dynamic than the conventional Internet and in unexpected ways change quickly. The structured of IoT requires could to automatically adapting.

- **Software Complexity**: There is a need to design powerful and more comprehensive software infrastructure on the network to support and provide services for smart objects.
- **Interoperability:** Every kind of intelligent objects in IoT can have different information, processing and communication capabilities. Different smart objects are subject to many conditions, such as available power and bandwidth needed for communications. There is a need for Common standards to facilitate communication and cooperation between these objects.
- **Data Volumes:** Huge information of data are gather from sensor networks and large-scale networks, a huge amount of information will be on the network servers. New technologies will be required to manage, store and translate this information.
- **Self-Organizing:** Smart objects in IoT need to be able to configure themselves, and able to establish a connection automatically.
- **Working with Multiple Communications Standards**: The huge increase in range of used IoT devices has led to the multiple emergence of new standards and wireless communication technologies and being to use, and each standard has different testing requirements. Therefore, the great challenge for designers is to ensure all components must work together efficiently and support more than one standard at the same time [13].
- **The Need for Long Battery Life**: In order to reduce the cost of maintenance, especially when designs feature in large numbers of sensors, it is necessary to have a long battery life. Designers are looking to increase the operating life of the battery by implementing sleep patterns and idle modes whenever possible. In high-performance machines, different parts require different amounts of

energy. Designers need to know how much use for energy, the amount of current required and the time spent in each operating mode. These considerations can increase battery life [13].

1.4 Literature Survey

Newly, the IoT has become more interesting for researchers around the world; it aims to make home and cities smarter. Things in IoT are connect to each other by embedded with sensor devices that collect information about things and share information with the server computer. Sensors in the Internet of things need to wireless communicate transceiver to be able to exchange information among them, one of the standards used in the wireless connection is the IEEE 802.11 ah which is expected to be used widely for IoT applications. Viterbi decoder (VD) is considered one of the most important devices in the IEEE 802.11 ah receiver, which works to decode the convolutional codes received from the transmitter while at the same time correcting the bit errors, below are some research for IoT and Viterbi Decoder:

- In **2012 Yan Sun and Zhizhong Ding** [14], propose and presented FPGA implementation a (2,1,7) convolution encoder based on IEEE.802.11a WLAN standard in OFDM baseband modulation and design Viterbi decoder with constraint length=7, code rate 1/2 and decode depth 36 symbols . They designed and implementation using Xilinx Virtex-II by ISE 7.1 environments; and obtained simulation results using Verilog HDL. The Viterbi decoder implemented in parallel structure to improve the speed of calculations in ACS unit and adoption of optimal data storage to avoid overflow, the results of this design decreasing 10% of chip logic and used around 2.99 % of total chip logic elements and reducing 5% of the power consumption. The synthesis results showed of FPGA implementation can operating on 80 MHz frequency and this is suitable to current applications.

- In **2014 T. Adame, A. Bel, B. Bellalta, J. Barcelo, M. Oliver** [15], propose using and evaluation IEEE 802.11ah on four scenarios of (M2M) applications (intelligent metering, industry automation at indoor, animal control and agriculture outdoor). With the significant development of (M2M) communication, M2M depending a new standard to meet its requirements addressed by new amendment IEEE 802 .11ah. The most important requirements of (M2M) are a large number of power-limited stations, transmission for long range, small and infrequent data messages, and low data rates. The IEEE 802.11ah standard ensures meet these requirements in terms of increasing the number of operated stations arranged by a hierarchical organization up to 8191 stations sharing one Access Point (AP). Their results showed that more than 99% of the time in the stations was in the sleep mode.
- In **2015 Thi Hong Tran, Hiromasa Kato, Shinya Takamaeda-Yamazaki, Yasuhiko Nakashima** [16], propose and presented study to developed low complex-Viterbi decoder to meet requirements applications of IoT. They work to study the effects of Viterbi Decoder parameters on (Bit Error Rate) BER, and (Packet Error Rate) PER on communication system. These parameters include, trace back length (L), width of input data (D), and log likelihood ratio (LLR) a truncated value (E). The best result obtain when (20 < L< 40), if L continuous to increase (i.e. L >=60) the performance become not important, also BER and PER performance improved when the width of input data (D) be in range (1 to 4), the best result when increase 2 to 3. From this results they were able to

use (20 < L < 40), D=3, LLR=1.75 to obtaining develop Viterbi decoder for IoT applications.

- In **September 2015 Minyoung Park**[17], presents overview of physical layer (PHY) and Medium Access Control (MAC) layer features of 802.11 ah that can suitable and support sensors and applications of IoT. One of the most important features of PHY layer in 802.11 ah that supports the applications of the IoT in outdoor by providing data rate of up to 150 Kbps to 347 Mbps and the transmission distance of 1 km using physical layer parameters. The long-range transmission provides optimal power consumption using very low transmission power provided by MAC protocols and MAC frame formats that have improvement for sensors and IoT applications that supports a large number of stations.
- In **2016 Hiromasa Kato, Thi Hong Tran, Yasuhiko Nakashima** [18], propose and developed the (K-best Viterbi decoder) based on an (M) algorithm that work to reduce the number of paths in the trellis to (K) with the same traditional Viterbi decoder performance to meet the requirements of IoT. The Viterbi decoder that decode convolutional code in the receiver and correct the errors, which works ceaselessly so it consumes a lot of energy. Therefore, the establishment of a low-power wireless and small transceiver is more important for IoT applications. They implemented simulation on (Verilog HDL) and the results showed that the best value for K is 4 to obtain the same Packet Error Rate (PER), they found results of (K-best Viterbi) can reduce 69% of area and 91% power consumption.
- In **2016 Thi Hong Tran, Duc Phuc Nguyen, and Yasuhiko Nakashima**[19], proposed a new algorithm called K-min Viterbi decoder, that less complexity than conventional Viterbi decoder for

wireless sensors that meet requirements IoT applications, in wireless sensors the transceiver is a very important unit and Viterbi decoder is needed in wireless transceiver to enhanced decoding performance. They evaluate the performance of (PER) of the algorithm that depend on some factors such as modulation type, trace-back length, and channel type with some value of K by using 802.11ah simulator have 64-state Viterbi decoder (VD) and showing complexity of K-min VD is less than conventional VD by 64/K times. They recommend K value between (3 and 5), they reached to reduced complexity of VD by (12.8 to 21.3) times than conventional VD.

- In **March 2017 Rami Akeela and Yacoub Elziq** [20], propose introduced the Hardware implementation of IEEE 802.11ah standard, which provides a solution to meet the requirements of IoTs. They have been introduced the new features on amendment for the Wi-Fi standard IEEE 802.11ah on the MAC layer to support the largest number of stations, increase operating range and reduce power consumption. They was performed simulation using (Xilinx Vivado) to verify the design. The results of IEEE 802.11ah structure analysis indicated that is suitable for FPGA and ASIC based applications. The synthesized done on the ZYNQ-7 ZC702 FPGA board and result show decrease the hardware utilization and low power consumption.

1.5 Thesis Layout

This thesis is arranged as follows:

in addition to the current chapter, chapter two present an overview of the IEEE 802.11ah and IoT as well as detailed explanation for Viterbi decoder structure and procedure and the tool used in implementing design.

Chapter three present the simulation of the design of Viterbi decoder using MATLAB (2013a) software program. Includes five different encoding types and make comparison between them to choose the best model.

Chapter four presents FPGA implementation for the designed Viterbi decoder at constraint length 6 an 7 in detailed, including VHDL implementation, flow charts and implementation result.

Finally, conclusions and future works are given in chapter five.

Chapter2

IEEE 802.11ah and IoT Overview

2.1 Introduction

In recent years, the term "Internet of Thing" (IoT) has emerged and has attracted the interest of many researchers around the world. This term refers to a network of different objects and devices that able to communicate with each other. These objects or devices are integrate with sensors to collect information about objects and to communicate with the server computer to exchange the information. Thus, a person can control a particular device without being in a specific place to deal with the device. Sensors in IoT are need to wireless communication transceiver in order things to be able share information. In IoT connections, it will be expect of widespread to using IEEE 802.11ah technology. It is a one of the Wi-Fi standards for wireless communication of IoT sensors, which created by IEEE 802.11 committee members in 2015. [18]

2.2 IoT Concept

Internet of things (IoT) or called Internet of Objects is a network of interconnected devices that can to interaction with each other and with humans and physical objects in the world to execute a diversity of tasks. IoT is a new technology for connecting to the internet. Objects in IoT are communicate with each other and behave smartly by making the right decisions in real time as well as can distinguish themselves and accessing information gathered by other things. The IoT devices embedded with sensors to ensure the devices connection with the physical world. Figure (2.1) show that anything's can connection to the internet from any place to offer any services at any time. The using of sensors enables IoT devices to sense changes in their surroundings, enabling them to improve procedures to continue carry out their task, as well as make effective independent decisions to fit changes and interacts with the physical world. This interaction with the physical world make a IoT of efficient applications have spread widely and easy to use, making IoT devices is available in many applications fields, from smart cities to smart industries, from smart home to smart personal healthcare etc. With the widespread use of IoT devices, become more popular. By 2020, devices connect to internet will be 30 billion, depending for the business insider report. In the next five years, the cost of manufacturing of IoT devices will be more than $ 6 trillion dollars [2][21].

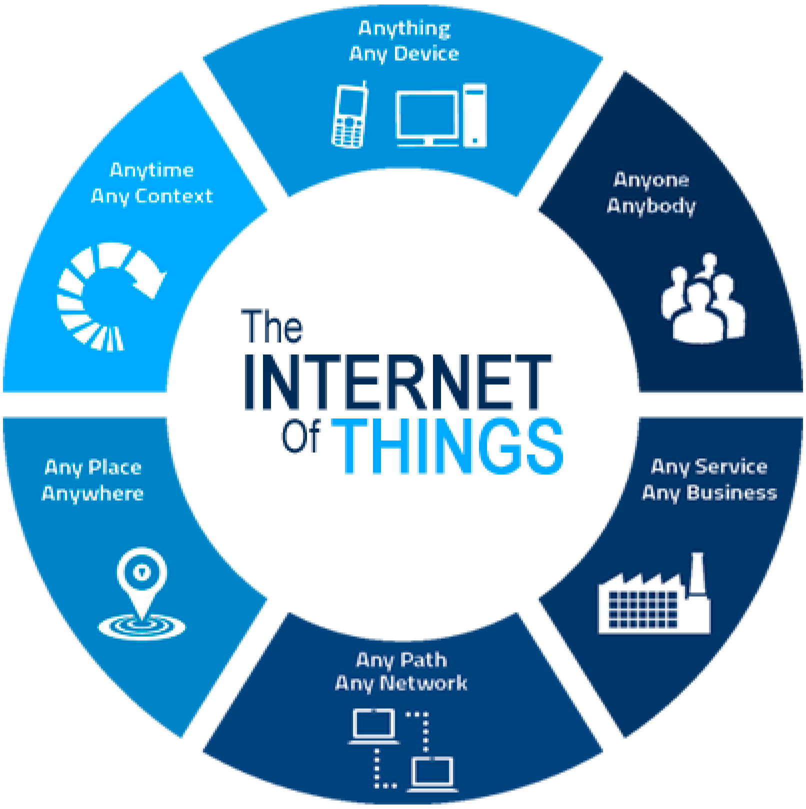


Figure (2.1). IoT Concept

2.3 IoT Application Requirements

With the emergence of IoT concept, several technologies and applications have been create that are aimed to changing aspects of day life for the better [22]. In IoT applications devices, the sensors are include which operate on small data transfers over short distances from remote locations. One of the most important requirements is long range, most applications covering short distances ranging from 10 to 100 meters, many of which require a range of over 100 meters to several kilometers, and data rate can considered less than 1 Mbit / s be a typical . The second important thing is the low power consumption for applications where most applications run on the battery, therefor require a long battery life. IoT applications that need to short range can be using traditional communication technologies such as ZigBee, Bluetooth, and Wi-Fi, which its frequency provide the short-range. Energy consumption is another challenge. It's known in the physics of radio waves that the range is the reverse of frequency. This means, in low frequency signals the transmission range is high in contrast to high-frequency signals. The frequency range of the basic communication techniques is either 2.4 GHz or 5 GHz in the short rang. The range can be extend using low frequencies. The benefits of working in the frequency band below 1 GHz is the range expansion as well as the low frequencies enable penetrate buildings better than the high frequencies [23].

One of the important features of a communication protocol that supports IoT applications are having a long transmission range, low data rate and low power consumption[1]. The basic features of the for IEEE 802.11ah specification that amend from IEEE 802.11 legacy specification can able to support IoT application requirements comparing with current communication protocols. IEEE 802.11ah is a new technology operating at 1GHz, where the low frequency ensure the expansion of transmission band. Thus, IEEE 802.11ah can be adopt as a communication protocol standard covering IoT requirements.

2.4 IEEE 802.11ah and IoT

In the seeing an intelligent world, the spread of (IoT) evolves very quickly. IoT having things in various aspects of our daily lives such as mobile phones, sensors, vehicles, clothing, computers and many other things. The number of devices is increasing in the internet, according to the Cisco Internet Business Solutions Group (IBSG). The number of devices in 2020 will reach 50 billion. According to the large expected number of devices, wireless technology is suitable for connecting this number of devices. Wireless technology was develop to connect a limit number of devices that are over short distances with each other at a high transfer rate. So, the IEEE 802 LAN / MAN Standards Committee (LMSC) was set a new standard IEEE 802.11ah Task Group (TGah) amend from legacy IEEE 802.11 standard to fit the low power requirements of IoT devices connected to internet. The future is predictive of the increasing number of low-power intelligent devices connected to the internet through low-power wireless technologies. Therefore, the task group has expanded the scope of 802.11 networks, which develop energy-efficient protocols to connect thousands of devices that can to operate within the same region. The expanding of the concept of IoT become covering many application in everyday aspects of human life. With the evolution and spread of wireless communication protocols the Wi-Fi network is become used in various daily fields, at home, university, cafe and work places to connect to the internet across a wireless access point. So, with widespread use of Wi-Fi, it has become the best choice of variety IoT applications. A huge number of IoT applications devices are connect to internet by choosing a low-cost wireless technology and power consumption. IEEE 802.11ah is a new communication protocol designed for the purpose to achieving a long-distance communication between a large numbers of low-power devices to meet the requirements of the IoT. It works in the low-frequency range, one gigahertz (900MHz), which has a wide spread capacity compared to conventional Wi-Fi technologies such as 6Lowpan, ZigBee. This band supports transmission up to 1 km in outdoor and with the possibility of connecting a large number of the devices up to 8191 with one access point.[24][25]

2.5 IoT Architecture

IoT concept provides the possibility of connecting devices and sensors to the Internet and therefore available to communicate to anyone at anytime and anywhere. IoT applications are moving to include home appliances, vehicles, the environment, and their susceptibility to sensing other things and interacting with each other without any human intervention. IoT has become a new revolution in the information world. The IoT network has many sensors that collect information about things and make the right decision in real time to be able to manage them [26].

IoT architecture consist three layered called the perception, network, in additions, application layers. Each layer has specific functionality as below:

- The perception layer, It is a physical layer, whose main function is to identify objects and collect information about the environment in which they are located by using sensors as well as identifying other smart objects around them.[27]
- The network layer, the main objective of this layer is to transfer the collected data from the sensors across the perception layer, as well as manage communicating to network devices, smart objects, and servers.[28]
- The application layer, this layer is responsible for implementing smart solutions and offering application specific services to the user, and can

deployed the IoT for different applications such as Smart Cities, Smart Health, Smart Home.[26][28]

2.6 IoT and Viterbi Decoder

Wireless technologies are being important aspect of everyday users' lives, and their impact will be greater in the future. Recently, the communication habits among consumers have been changing according to developments in wireless technologies. A report by the World Wireless Research Forum (WWRF) expected that by 2020 seven trillion wireless devices would be deploy to seven billion people. Business Insider report is more recent in 2014 expect by 2020 the number of IoT devices connecting to the internet will be 24 billion. With this increase in the number of devices connected to the internet, will guaranteed that there is a connection to the internet for everything from anywhere and at any time, therefore will be an introduction to the concept of IoT. Things in internet of thing can contains a wide range of different elements in daily life such as tablets, digital cameras and smart phones that embedded with sensors [23]. Sensors are using to collect data about objects, so it considered being a core technology in Internet of thing applications. They are small, wireless low-power electronic devices designed to perform a specific function or embedded with other devices such as smartphones. Sensors can be consider the main engine to IoT expanding. IoT contains a variety devices and heterogeneous communication protocols for data exchange between devices and data servers. In IoT sensors, wireless communication transceiver are needed for exchange data [10]. Sensors monitor physical changes and measure them to convert into raw data stored digitally to be ready for analysis. In order to fit IoT application requirements, the digital signals emerging from sensors have little power that make are subject to noise during wireless transmissions. Since, in IoT the physical layer components of transmitter and receiver are like to component of Wi-Fi transmitter and receiver. It is necessary to make Research on a low complex Viterbi decoder Suitable for IoT applications [10]. One of the most important and complex components of the wireless transceiver is the Viterbi decoder (VD). The VD function in the receiver is to decode the convolutional codes coming from transmitter and correct the resulting errors during transmission. VD is consuming a lot of energy continuously because it operates nonstop in the wireless transceiver. It is important to development a small Viterbi decoder consumes a small amount of power to suit wireless transceiver for IoT applications [2]. The most essential in 802.11ah transceiver is to making the circuit in low power and small size. One of the most complex block in receiver of IEEE 802.11ah is Viterbi decoder (VD). The search on low-complexity VD is important for the development of a communication transceiver suitable IoT application [2][29].

2.7 Error Detection and Correction (EDC)

In wireless communication systems, the interference and noise accrue and it is very high. It affects the Signal to Noise Ratio (SNR). Therefore, the Error Detection and Correction (EDC) techniques are need to improve SNR [30]. Convolutional code is error correct techniques almost used in most of communication systems. Transmitter used convolution encoder add redundancy bits to the information before transmitted. Therefore, the effective technique to decode the convolutionally encoded data is Viterbi Decoder (VD). The general flow of information over a noisy channel shown in fig. (2.2).

`

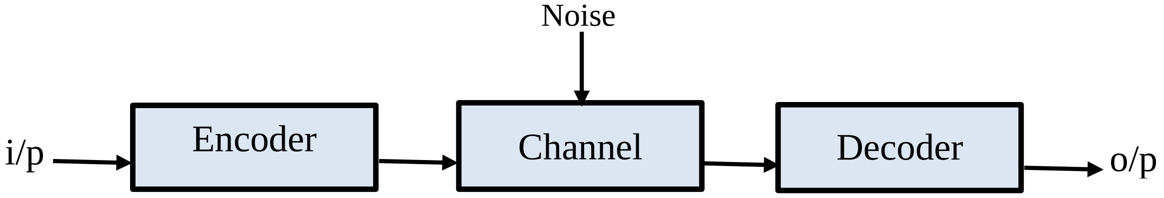


Figure (2.2). The Block Diagram of Flow Information over a Noisy Channel

2.8 Convolutional Encoder

Convolution codes are one of the powerful and widely used in numerous applications because of ability of error correction [31]. The convolution encoding is done by entered a fixed number (m) of input data to encoder to result in an n_ bits symbol [32]. The input bits passed through the shift register and merged using (XOR) gates with many outputs of shift register cells [33]. This process is equivalent to convolution method and called a convolution coding [32]. A convolution encoder protects data by adding redundant bits into the binary data stream through the linear shift registers. Convolution encoder can be described by the three following parameters (n ,m ,L) as summarized:

n : number of output symbols.

m: number of input symbols.

L: number of shift register.

Constraint length (K) = L+1 is the number of stages which the input bit pass through in the encoding shift register [34]. Convolution encoder accepts an (m) input bits that are fed to the shift register and calculates (n) outputs from generator polynomials by (XOR). In generator polynomial, (1) refers to the connections and (0) indicates that there are no connections between shift register and (XOR) gates. Fig. (2.3) shows a simple convolution encoder example with m=1, n=2, K=3, u1=111, u2=101 [35].

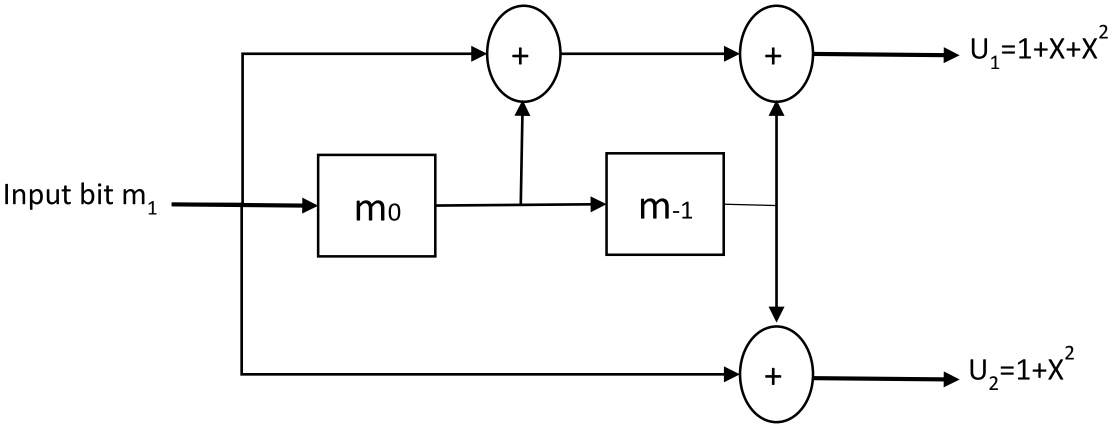


Figure (2.3). Convolutional Coding, Rate 1/2, Generator Polynomials 7, and 5

There are three methods using to represent the Convolutional encoder by state diagram, state table and trellis diagram. The number of state code can be defined by the combinations of bits number in the shift registers that calculated by: number of state =2^L, in the code (2,1,3) above L=2; so the number of states equal to four and defined by : (00) ,(01),(10),(11). When input equal (0) the transition is represent by solid line and when input (1) represent transition by dotted line. Figure (2.4) shown the transition between different four states in state diagram.

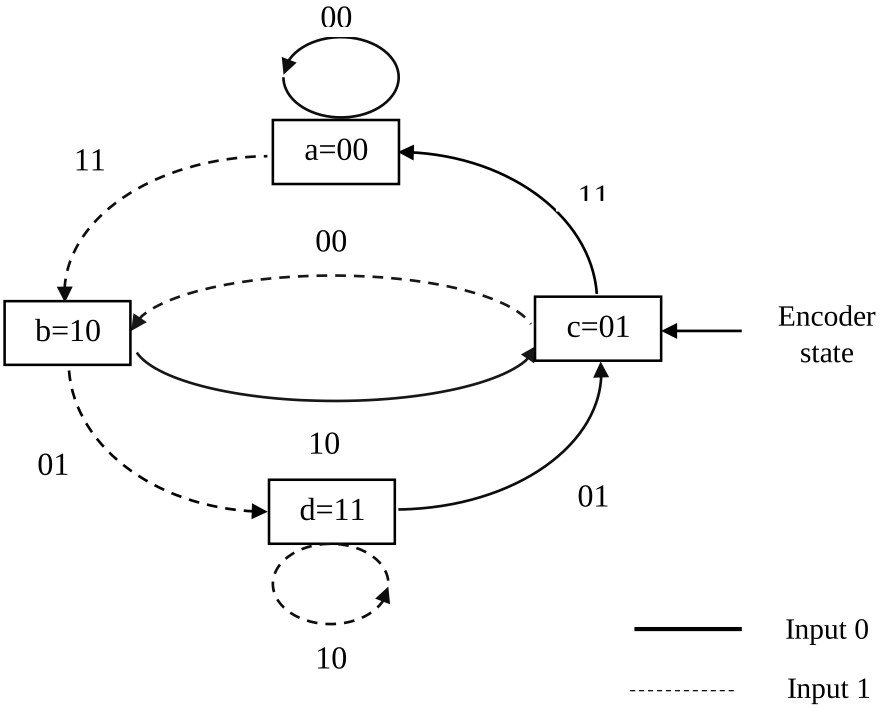


Figure (2.4). State Diagram of Code (2, 1, 3)

The four states table are shown in table (2.1).

The trellis diagram illustrates the states diagram in the encoder with a time line; in every time unit is represent a separate state diagram and the number of state is (2^K-1) . The trellis consist of matrix of nodes, these nodes represented the number of states in the encoder, each column in the matrix represents the encoder states for a specific time (the first column starting from the left represents the possible states at t=0 and the second column represents the possible states for t=1 and so on). Each state in the trellis have 2^m branches leaving from it and 2^m branches arriving to it. Figure (2.5) show the trellis diagram for code (2 ,1 ,3).

In the encoder, each state transition results in one code word that may be corrupted through transmission over a noisy channel. The sequence of data input of the encoder can be reconstructed by Viterbi decoder through calculating the most likely sequence of state transition [33]. The trellis diagram in Viterbi decoder using to calculate accumulated distances (named a path metric) to decode transmission data.

Table 2.1.State Table

Input bit	Input stat		next stat	Output bit
0	00	S0	S0	00
1			S2	11
0	01	S1	S0	11
1			S2	00
0	10	S2	S1	10
1			S3	01
0	11	S3	S1	01
1			S3	10

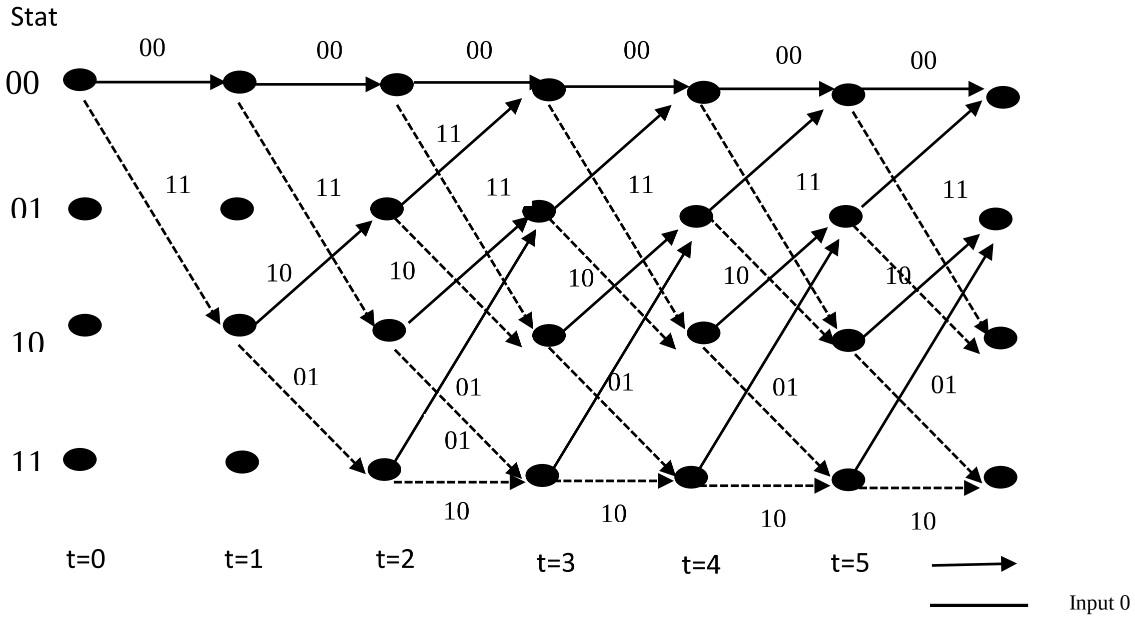


Figure (2.5). Trellis Diagram of Code (2, 1, 3)

2.9 Viterbi Decoder

To decode a noisy signal received, the Viterbi decoder is used [36]. The Viterbi decoder is the most common way to decode convolutional code that may be corrupt through transmit [33]. Viterbi decoder uses Viterbi algorithm to decode convolutionally encoded data. The decoder configuration depends on encoder's generator polynomial, code rate and constraint length. The encoder adds to original information bits redundant bits for the purpose of error recovery. The output from encoder is transmit across the channel. In the receiver, the data contains information with redundancy bits that may be corrupted through transmission, Viterbi decoder tries to extract the original information by start moving along the trellis in the front direction then make some calculations, after this in backward direction move to find the transmitted bits sequence before encoding [37]. Decoding complexity is the main drawback it growing exponentially with the length of code, so it can be used only for relatively short codes [36]. The computational complexity in Viterbi decoder can be reduced by using simpler trellis structure. Decoder can be able to correct information bits corrupts by noisy channel through transition due to the structure of convolutional encoded code. The decoding will start from zero state by the assumption that the encoding will start from the same state. The Viterbi decoding select the low metric path to be winning path. The winning path will have low bit error path compared to other paths. Viterbi decoder implementation have two types, namely hard decision and soft decision Viterbi decoding. In hard decision, a hamming distance metric is use, and in soft decision the Euclidean distance metric used. Hamming distance metric is the number of differ bits between received code and actual code and the path with lowest Hamming distance metric is called survivor path. If hamming distance is zero, this indicate that the symbol is receive without any error. Euclidean distance metric is squared difference between received symbol and actual symbol. The path with minimum Euclidean distance called a survivor path. The received symbol in Soft decision decoding is quantized in to more than two levels, so the soft decision decoder is better than hard decision because soft decision decoding will have more information about the received information [38][31].

Fig. (2.6) shows the general block diagram of Viterbi decoding algorithm. Viterbi decoder has three basic units [39]:

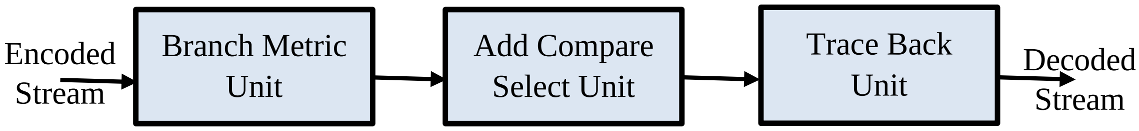


Figure (2.6). Block Diagram of Viterbi Decoder

2.9.1 Branch Metric Unit (BMU)

The first unit in the Viterbi decoder is branch metric unit, this unit calculate the distance between received code from noisy channel and legal codewards for all trellis branch by compares between them and calculate the number of different bits. There are two measure in BMU, hamming distance in the case of hard input decoding or Euclidean distance when soft input decoding [33]. Hamming distance calculated by XOR-ing received code with legal codewards and counts the number of 1's in the result ,this can illustrates in figure(2.7).[38] thus minimum Hamming distance consider as optimal path through the trellis[39]. Euclidean distance is calculate the squared distance between received symbol and ideal symbol to find the distance in soft-decision decoding; the resultant is branch metric (BM) as shown in equation (2.1) [37] [38].

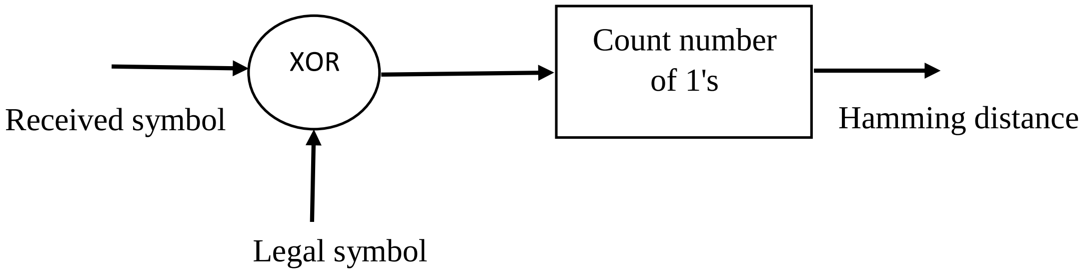


Figure (2.7). Branch Metric Unit (BMU)

$$D= (A-A_0)^2+ (B-B_0)^2 \qquad 2.1$$

2.9.2 Add Compare Select Unit (ACSU)

The second unit is add compare select unit, also known as the path metric unit (PMU) calculate new path metric values. It is sub divided into three parts: add, compare and select units. Consist of two adder and comparator. in trellis each current state can reached from the previous stage (from two states), so by help adder each current state have two path metric through adding current branch metric with path metric to every two branch coming from previous state. After this, the comparator select the least metric, and store it as the new path metric to current state. The ACSU block diagram is showing in figure (2.8) [37].

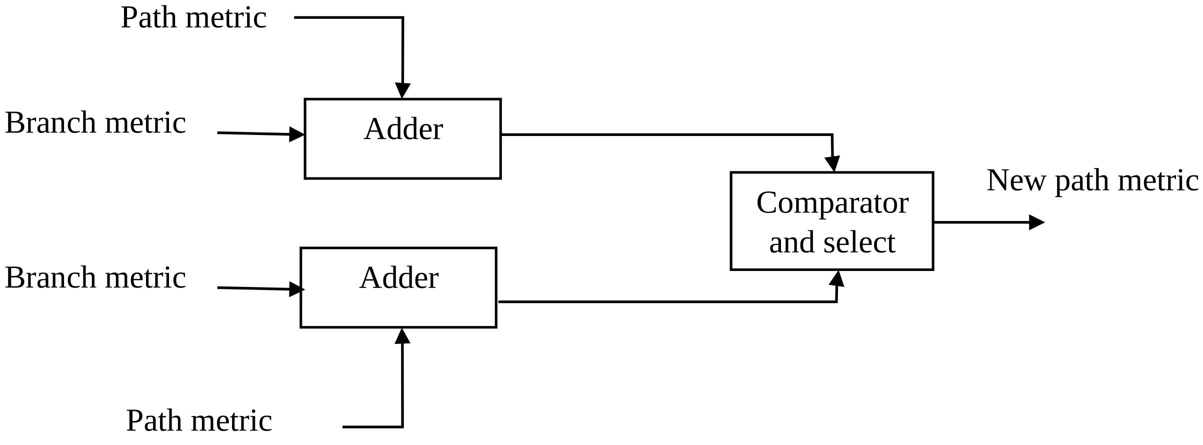


Figure (2.8). Add Compare Select Unit(ACSU)

2.9.3 Trace Back Unit (TBU)

After storing all possible survivor paths by ACSU and assigned one bit for the local survivor branch to every state in the trellis to allocate if the winner branch come from upper or lower position, starting the decoding for a block of data (determined by the trace back length) through TBU. This is done by traces back the trellis when reached the end of the trellis defined by trace-back length (decoding depth) [38]. TBU starting from the path of the last survivor, for node of the minimum path and then trace back paths that return the path of the initial survivor track of state 0. Then the original information data corresponding to the encoded data is determined. In order to get better performance, the traceback length is calculated by equation [40]:

$$(2 \text{ to } 3) *(K-1) / (1-r) \qquad 2.2$$

Where: K=Constraint length and r=n/m.m

To show the method of operation followed in Viterbi decoders, below is a simple trellis diagram example at code rate=1/2, K=3 for convolution encoder and a message length of six bits [1 0 0 1 0 0]. The four states are implemented as four-dot columns. For each column, the branch metric is calculated to find the smallest path for each time line (t). For the assumed input sequence bits, the output result from the encoder will be: [11 10 11 11 10 11]. During the transmission, the bit number ten is assumed to be corrupted, so the decoder input is [11 10 11 11 11 11]. Viterbi decoder works on reconstructing the original data and correcting this error as illustrated in the trellis diagram shown in figure (2.9). This figure gives the calculated metrics for the assumed inputs at all the required time lines, which requires re-plotting a trellis diagram six times to show the calculated metrics values at all the processing stages.

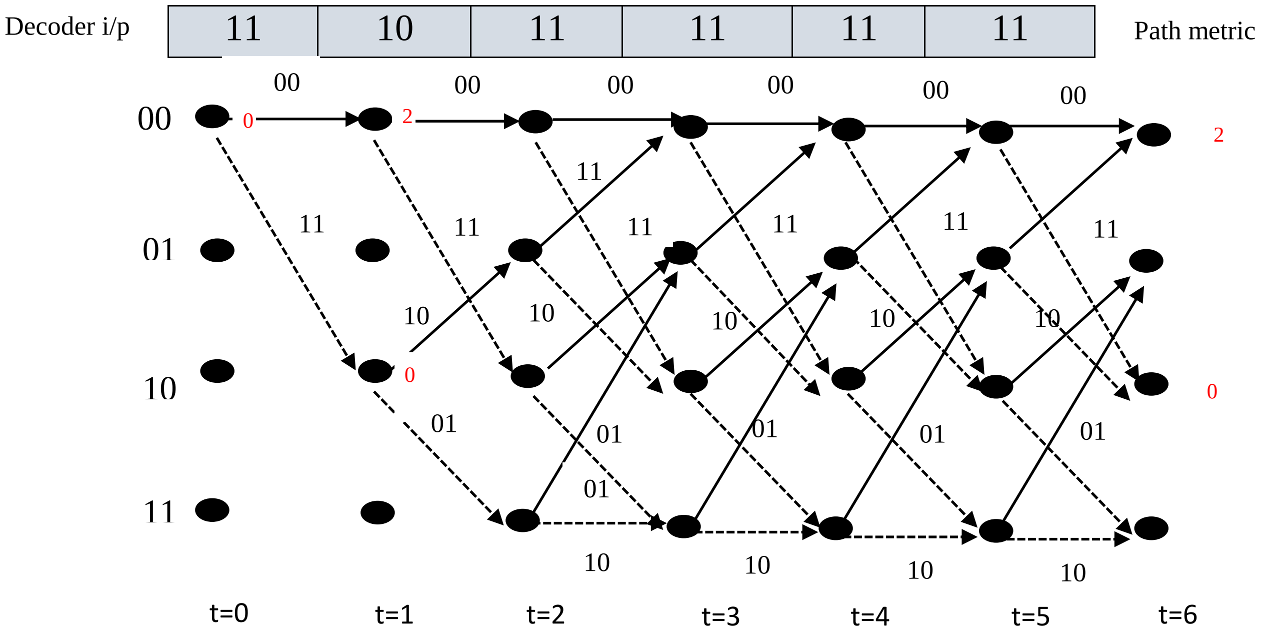


Figure (2.9.a). Metrics for the First Input

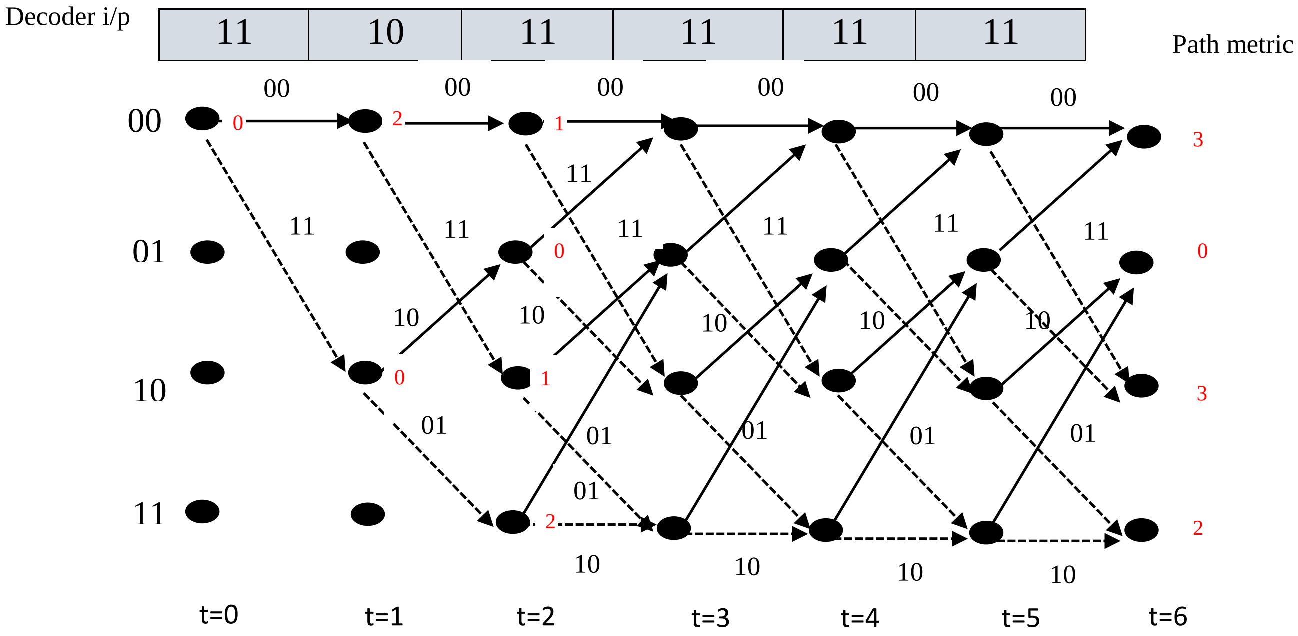


Figure (2.9.b). Metrics for the Second Input

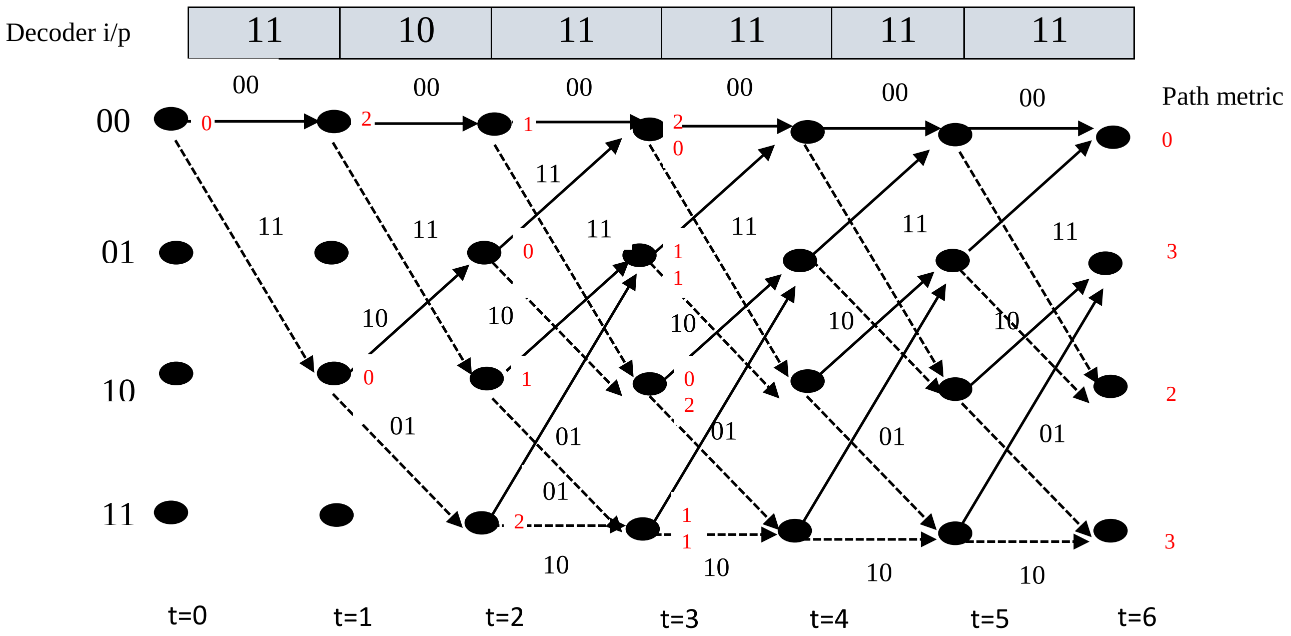


Figure (2.9.c). Metrics for the Third Input

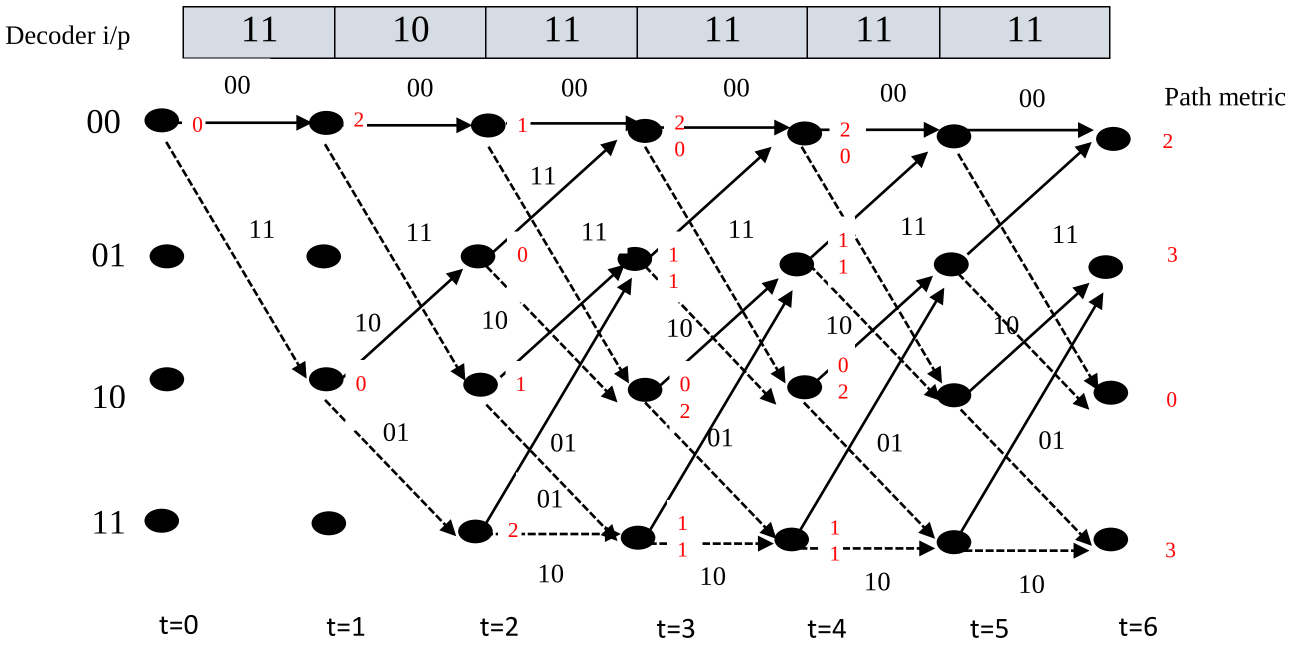


Figure (2.9.d). Metrics for the Fourth Input

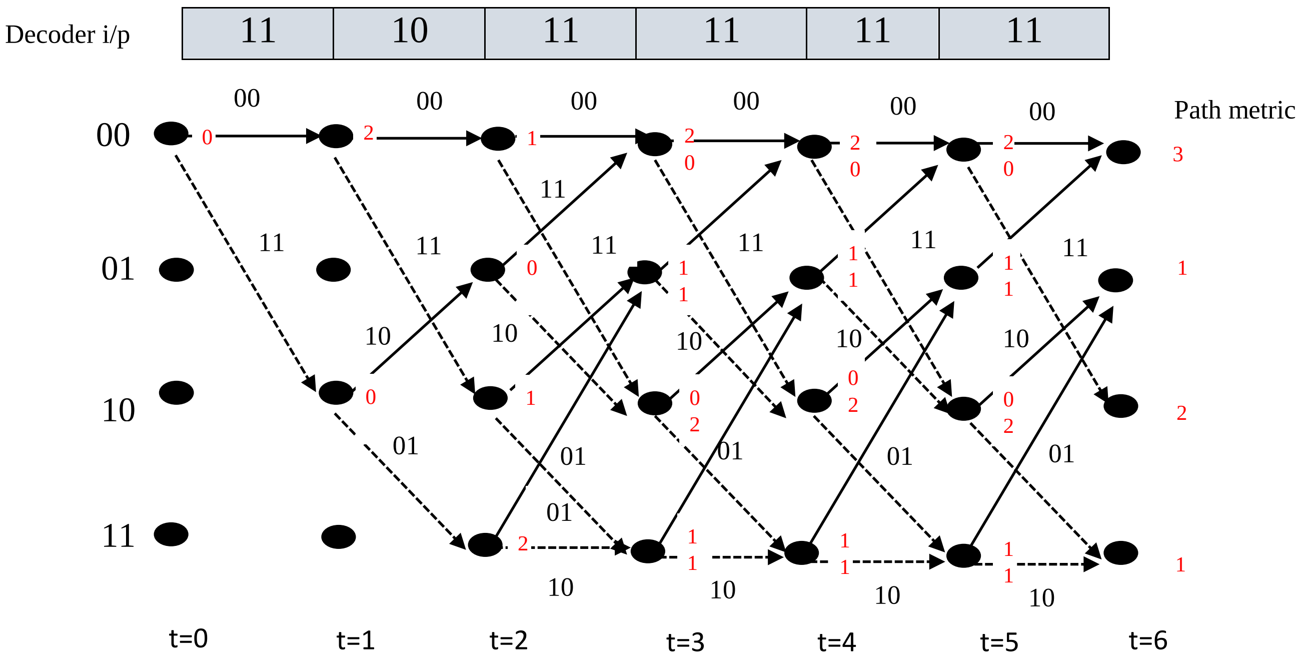


Figure (2.9.e) Metrics for the Fifth Input

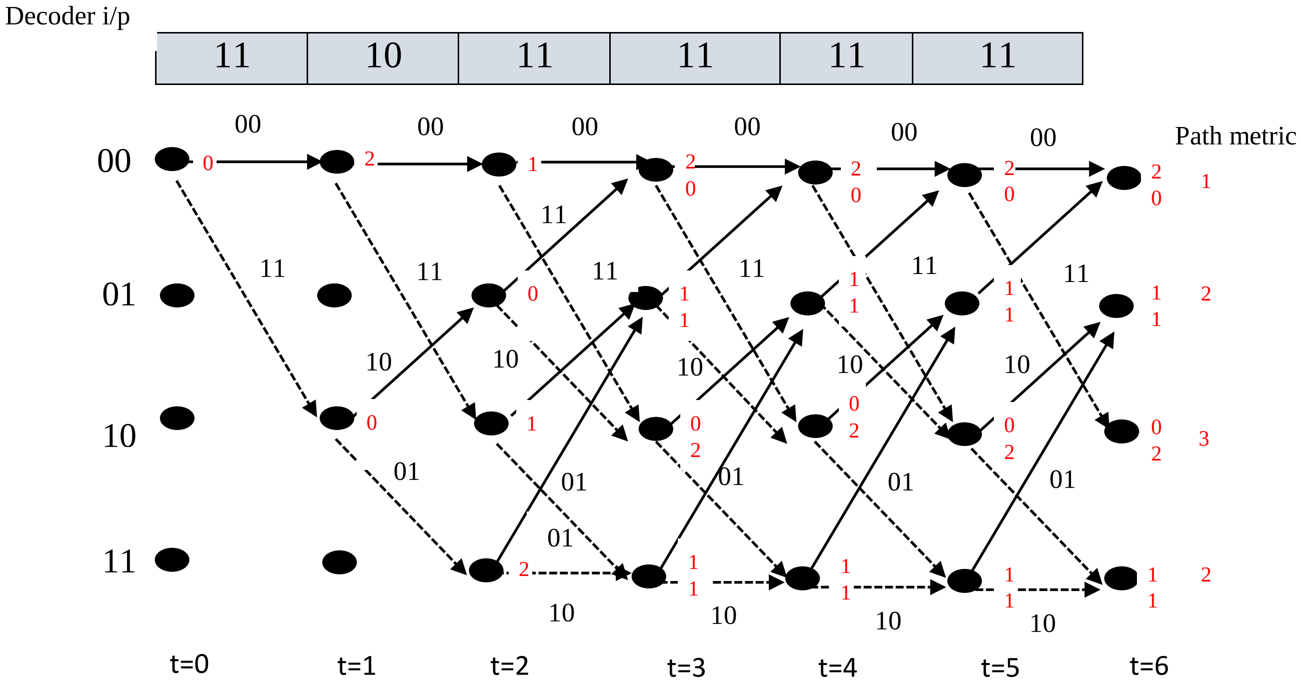


Figure (2.9.f) Metrics for the Sixth Input

After reaching the trace back length at t=6, VD begins from the state that has the smallest path to trace back the path that leads to the initial survivor track of state 0. Then the original information data corresponding to the encoded data is determined.

data i/p	1	0	0	1	0	0
Decoder i/p	11	10	11	11	11	11

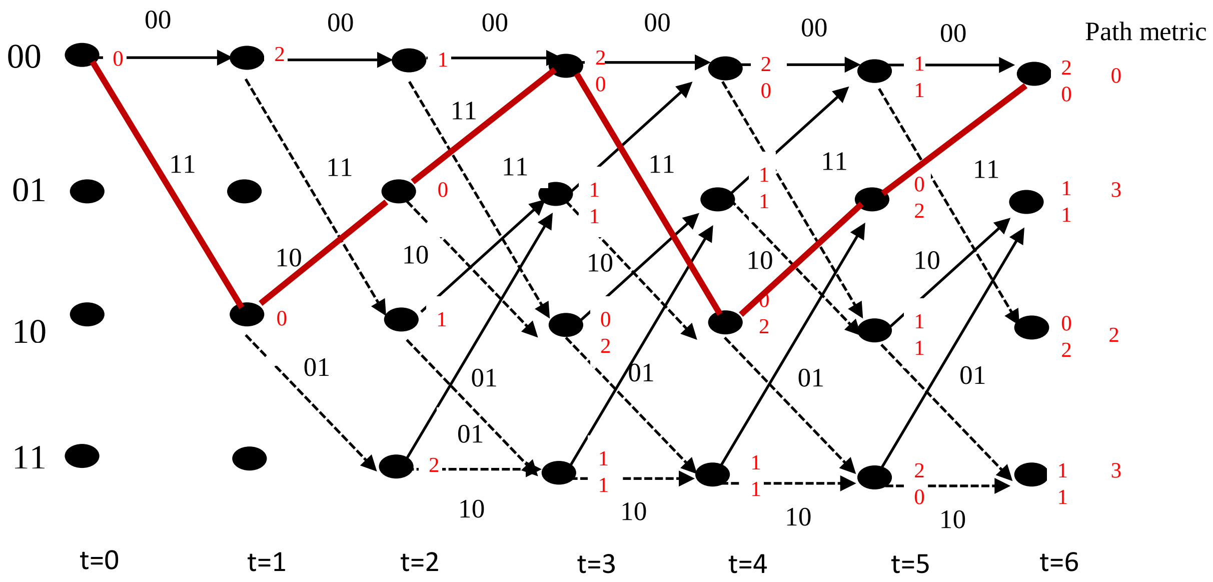


Figure (2.9.g).Trackback the Trellis

From the above example, it is clear that Viterbi decoder reconstructs the original input data as well as corrects the errors that may occur during transmission. A flowchart that shows the Viterbi decoder operation steps shown in figure (2.10).

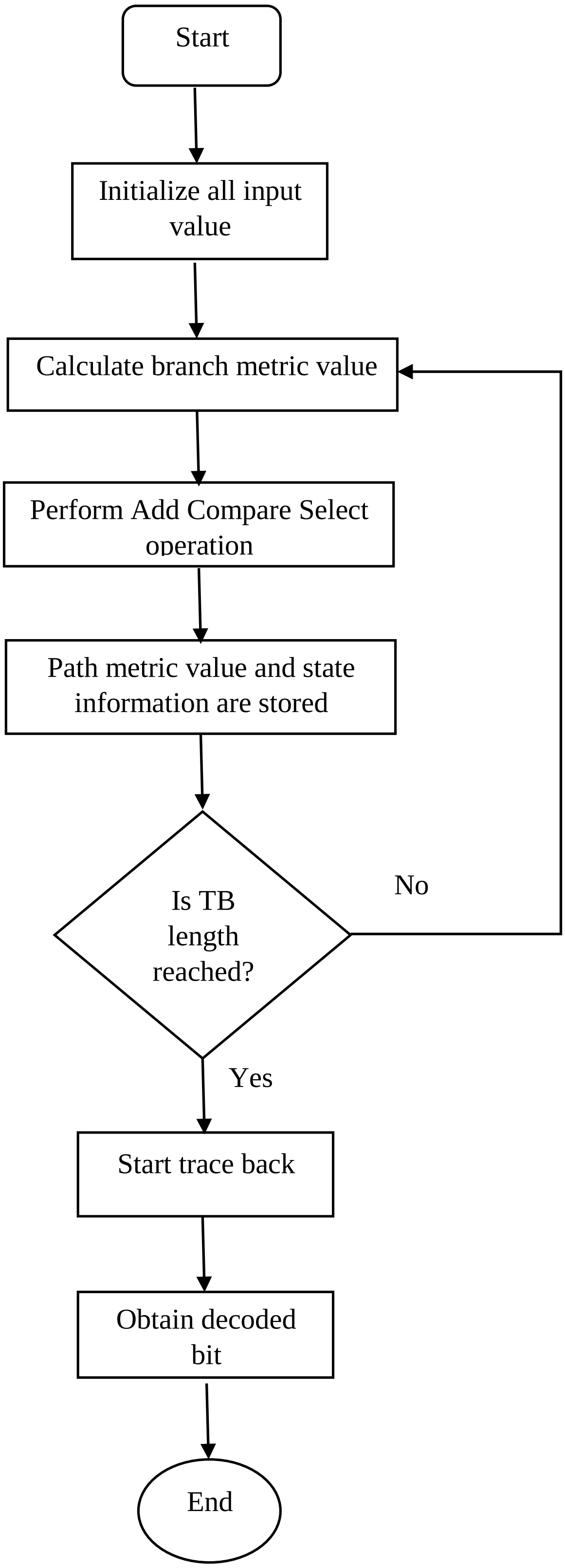

Start
Initialize all input value
Calculate branch metric value
Perform Add Compare Select operation
Path metric value and state information are stored
Is TB length reached?
No
Yes
Start trace back
Obtain decoded bit
End

Figure (2.10). Viterbi Decoder Flowchart

2.10 Viterbi Decoder Hardware Implementation Tool

A Project Navigator interface that enables to access the hardware provided by FPGA devices is the software programming tools produced by Xilinx and Altera [41]. The implementation of Viterbi decoder in this thesis was carried out using Xilinx ISE14.7 as a register transfer level (RTL) design tool. This interface provides the ability to display and access the source files in any designed project, in addition to providing the access to run processes for currently designated sources. It also provides quick access to creating projects and frequently accessing various materials, tutorials and documentation. It provides a display of status messages, errors, and warnings. It enables to view design reports, text files, simulation waveforms and schematics.

The implementation of a specific design requires translating,` mapping, placing, routing, and generating operations for a bit stream file for the design. These design tools are embedded in the Xilinx ISE Design Suite [42].

2.10.1 Field Programmable Gate Array (FPGA)

FPGA are digital integrated circuits (ICs) that contain configurable (programmable) blocks of logic along with configurable interconnects between these blocks. They are created in such a way that is possible for engineers to configure the design in the field to perform a tremendous variety of tasks [43].

The first FPGA, introduced in 1985, consisted of 2000 gates. Recently FPGA, devices consist of up to two million logic cells that can be configured to implement a variety of software algorithms [43]. Nowadays, there are many vendors of FPGA devices. One of the most popular advanced FPGA families in industry is the FPGA series produced by Xilinx [45].

All Xilinx FPGAs contain the following basics resources [46]:

- Configurable logic blocks (CLBs): provide the functional elements for constructing user's logic.
- Input/output blocks (IOBs): provide the interface between the package pins and internal signal lines.
- Programmable interconnections (PLs): provide routing paths to connect the inputs and outputs of the CLBs and IOBs.

The Viterbi decoder architectures are implemented in this thesis using one of the Xilinx FPGA device, The Nexys4 DDR kit board (Xilinx part number XC7A100T-1CSG324C). This device has features include:

- 15,850 logic slices, each with four 6-input LUTs and 8 flip-flops
- 4,860 Kbits of fast block RAM
- Six clock management tiles, each with phase-locked loop (PLL)
- 240 DSP slices
- Internal clock speeds exceeding 450 MHz
- On-chip analog-to-digital converter (XADC)

Figure (2.11) shows an overview of Nexys4 DDR kit board.

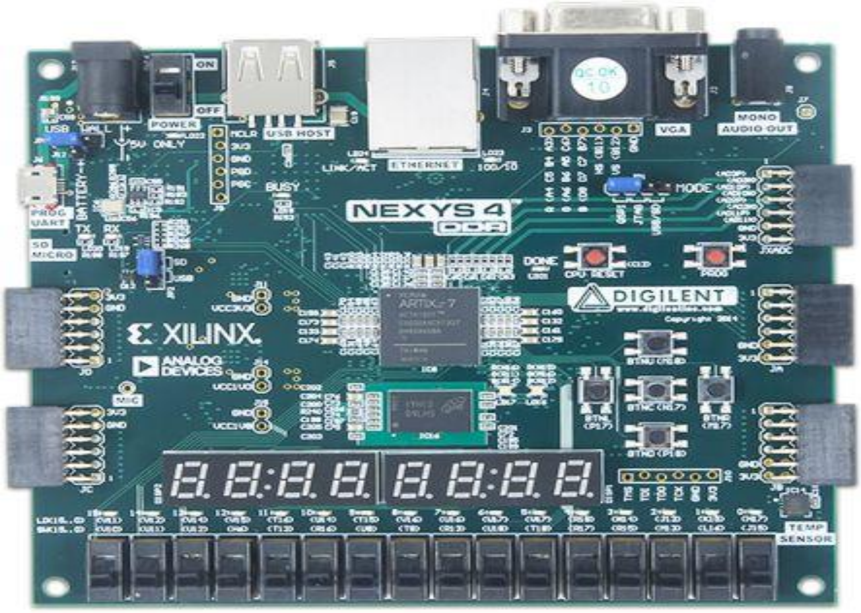


Figure 2.11 NEXYS 4 DDR overview

2.10.2 Hardware Description Languages (HDLs)

The two most popular hardware description languages are VHDL and Verilog. The HDL used in this thesis to implement the designed based on the VHDL, within the use of Xilinx ISE 14.7 as a register transfer level (RTL) design tool.

VHDL is a general –purpose hardware description language that can be used to describe and simulate the operation of wide variety of digital systems, ranging in complexity from few gates to an interconnection of many complex integrated circuits. VHDL was originally developed for the military to allow a uniform method for specifying digital systems. Since then, the VHDL language has become an IEEE standard, and it is widely used in industry [47].

CHAPTER 3

Simulation of Viterbi Decoder

3.1 Introduction

In this chapter, we present a study on decoding the convolutional codes in AWGN channel using the Viterbi algorithm. This study is perform using different constraint lengths at a fixed coding rate of (1/2). Two types of Viterbi decoder, hard and soft decoding are used. The performance of hard and soft Viterbi decoder is evaluate by measuring Bit Error Rate (BER) and decoding delays at various constraint lengths using MATLAB (2013a) program with 32-bit operating system in Intel(R) core(TM) i5-CPU 2.30 GHz and 2 GB RAM. This enables to select the best Viterbi decoder model that suited the requirements of IoT. Increasing constraint length results in further improvements in BER both in soft and hard Viterbi decoders.

3.2 System Overview

The general block diagram of the designed system that involves convolution encoder and Viterbi decoder shown in figure (3.1). This system implements at various constraint lengths. A description of each part of this system given in the following subsections, with including some of the MATLAB code used to implement each part.

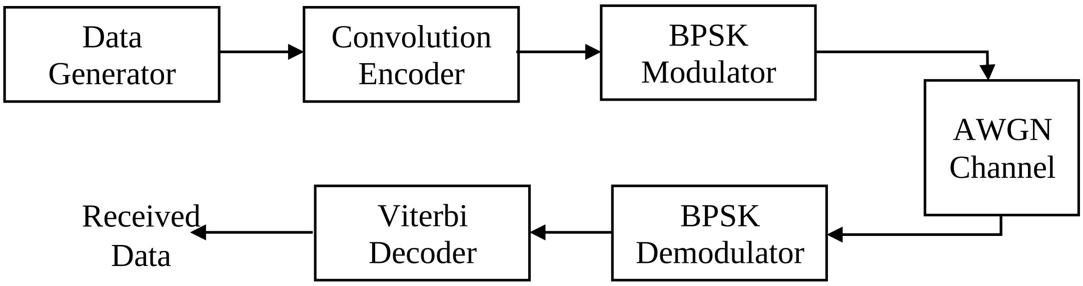


Figure (3.1) Block Diagram for the System

3.2.1 Data Generator

In this block, a random function generator in MATLAB used to generate binary random input that is pass into convolution encoder. In this simulation, 10^5 random bits are generate and coded to be transmit through AWGN channel.

3.2.2 Convolution Encoder

The convolution encoder (in the case of using a code rate of 1/2) is a made up of a number of memory shift registers and two modulo-2 adders. Each output represents one generator polynomial. A shift register holds only one input bit. In the simulation of this encoder, a comparison is performed between five different encoding schemes in order to select the best encoding scheme that suites the designed Viterbi decoder and helps getting the best performance, measured using BER at various SNR values. Each scheme model has different parameters. The number of state for each convolution encoder equals to (2^{K-1}). Figures (3.2) and (3.3) illustrate the encoding schemes for constraint length *K* (3 and 6) and the parameters for five encoding schemes given in table (3.1) where each generator polynomial equation can be represented in a binary or equivalent octal format.

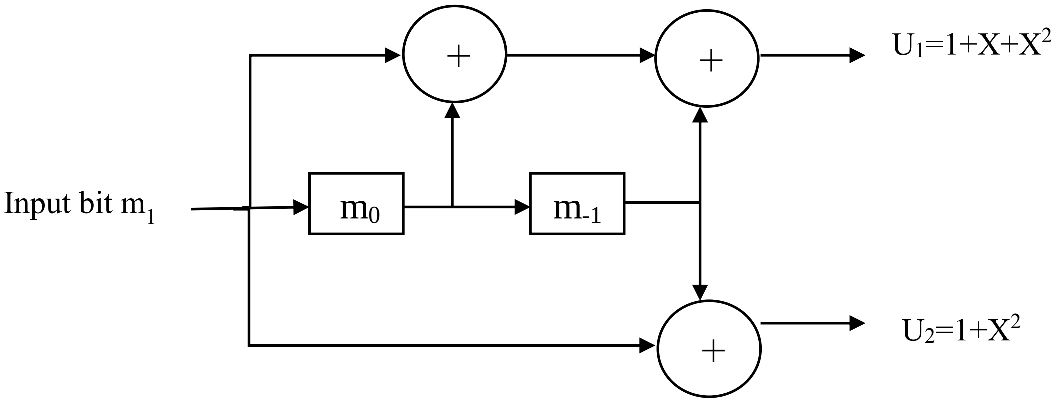


Figure (3.2) Convolution Encoder for K=3

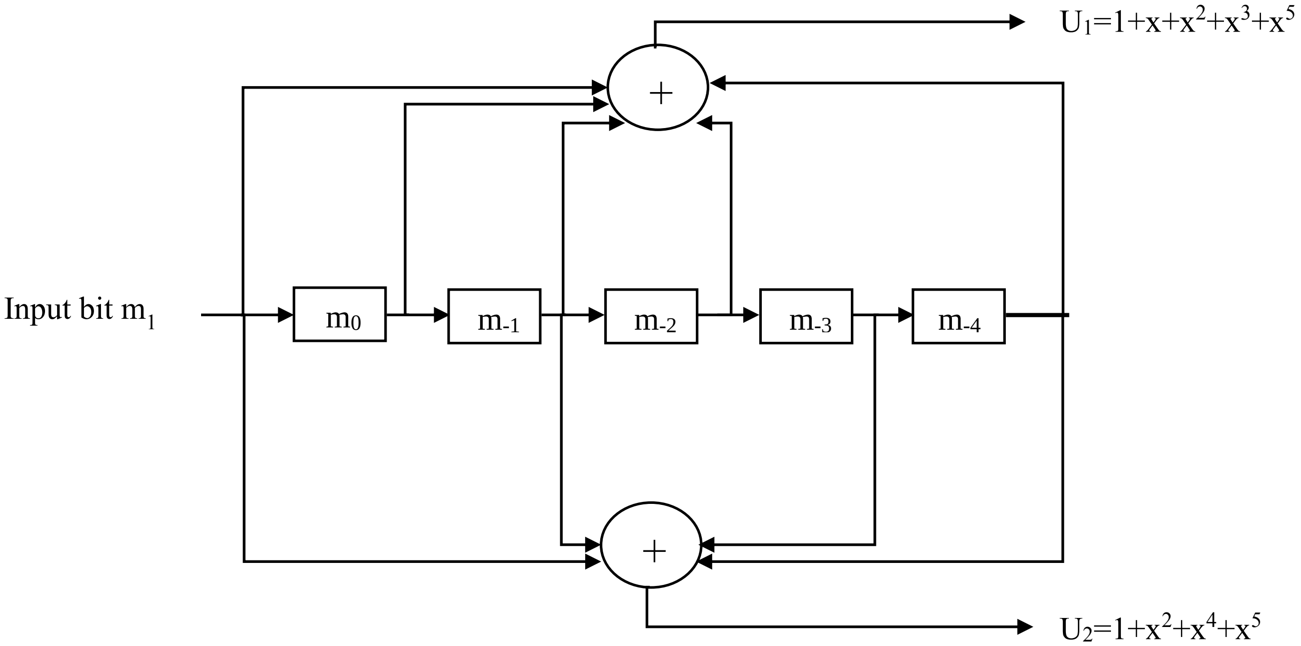


Figure (3.3) Convolution Encoder for K=6

3.2.3 BPSK Modulator

The BPSK modulator block utilize to modulate the received data from convolution encoder. This is done by giving an output of "-1" when the convolution encoder's output is '0' and "+1" when the encoder's output is '1'. This performed in MATLAB by using the following equation:

$$S_M=2*B_{code} -1 \tag{3.1}$$

Where, S_M represents the output signal of BPSK modulator and B_{code} is the convolution encoder output.

Table (3.1) Parameters of Convolution Encoder of Rate 1/2

Constraint length (K)	**Generator polynomial 1 (U1)**	**Generator polynomial 2 (U2)**
3	[111]7_8	[101]5_8
4	[1101]15_8	[1011]13_8
5	[11101]35_8	[10011]23_8
6	[111101]75_8	[101011]53_8
7	[1111001]171_8	[1011011]133_8

3.2.4 BPSK Demodulator

The AWGN channel gives a complex number sequence in the output. The value of this output ranges between "-1" and "+1". Because Viterbi decoder is not able to process this form of outputs, BPSK demodulator used to convert these complex numbers into real numbers, providing a matching stage between Viterbi decoder and the AWGN channel output.

In order to carry out BPSK modulator in MATLAB, the following function used when hard decision is done:

S_D =real (mod_n)>0

While in the case of soft decision decoding, the following MATLAB code is used:

S_D = real (mod_n)

where S_D represents the output data sequence value from BPSK demodulator and 'mod_n' represents the modulated signal which is the output coming from AWGN channel.

3.2.5 Viterbi Decoder

To recover the original transmitted data, Viterbi decoder is used. Viterbi decoder uses three stages to decode transmitted data, Branch Metric Unit (BMU), Add Compare and Select Unit (ACSU), and Trace Back Unit (TBU).

3.3 Viterbi Decoder Implementation in MATLAB

The operation and performance of Viterbi decoder are evaluated using MATLAB program simulations.

3.3.1 Branch Metric Unit (BMU)

Branch metric unit calculates the transition branch metric for the received code. This operation implemented in MATLAB by using XOR operation between the received code and the legal code of the branch. This operation is perform for a block of data which length is limited by the traceback length. The method used to calculate the traceback length was explaine in section (2.9.3).

The following sub code used to calculate the branch metric, where two branches metrics are compute for each of the four states:

```
Brarnch_metric(1,j)=sum([0,0]~=[y0(j+n-1),y1(j+n-1)]);
Brarnch_metric(2,j)=sum([1,1]~=[y0(j+n-1),y1(j+n-1)]);

Brarnch_metric(3,j)=sum([1,0]~=[y0(j+n-1),y1(j+n-1)]);
Brarnch_metric(4,j)=sum([0,1]~=[y0(j+n-1),y1(j+n-1)]);

Brarnch_metric(5,j)=sum([1,1]~=[y0(j+n-1),y1(j+n-1)]);
Brarnch_metric(6,j)=sum([0,0]~=[y0(j+n-1),y1(j+n-1)]);

Brarnch_metric(7,j)=sum([0,1]~=[y0(j+n-1),y1(j+n-1)]);
Brarnch_metric(8,j)=sum([1,0]~=[y0(j+n-1),y1(j+n-1)]);
```

Where [(00), (11), (10), (01), (11), (00), (01), (10)] represent legal codes while (y_0, y1) is the received code.

3.3.2 Add Compare Select Unit (ACSU)

After computing the eight branches metric, the second step is to calculate the path metric by adding each branch metric with the corresponding path metric from previous state and select the least path metric to be consider as a survivor path. The following sub program used to calculate the path metric:

```
Path_dis(1,j+1)=min(path_dis(1,j)+Branch_metric(1,j),path
_dis(2,j)+Branch_metric(2,j));

Path_dis(2,j+1)=min(path_dis(3,j)+Branch_metric(3,j),path
_dis(4,j)+Branch_metric(4,j));

Path_dis(3,j+1)=min(path_dis(1,j)+Branch_metric(5,j),path
_dis(2,j)+Branch_metric(6,j));

Path_dis(4,j+1)=min(path_dis(3,j)+Branch_metric(7,j),path
_dis(4,j)+Branch_metric(8,j));
```

Where "`Path_dis(1,i+1)`" refers to the path metric calculated for state j+1 and "`path_dis(1,i)`" is path metric for the previous state j.

3.3.3 Trace Back Unit (TBU)

The finale stage is tracing back the survivor path calculated from the previous step, starting from the state that has minimum path after reaching the end of the block of data and tracing back the path to initial state. The following sub program used to trace back the trellis and reconstruct the original transmitted data, where in this sub code, the previous state is calculated and the transition branch that leads to this state represents the decoded bit.

```
[pr-state,decoded_bit]=
prev_stage(state,distance,Branch_metric)

if(state==1)
if(distance(1)+Branch_metric(1)<=
distance(2)+branch_metric(2))
pr-state=1;decoded_bit=0;
else
pr-state=2;decoded_bit=0;
end
end

if(state==2)
if(distance(3)+Branch_metric(3)<=
distance(4)+Branch_metric(4))
pr-state=3;decoded_bit=0;
else
pr-state=4;decoded_bit=0;
end
end
```

3.4 Simulation Results

For the designed simulation system, (10^5) random stream of binary bits are generated. The values of Signal to Noise Ratio (SNR) are set from 0 to 10 dB. A BPSK modulation is applied and signals are pass through AWGN channel, then, decoded by Viterbi decoder. Analyses performance of Viterbi decoder is performed by plotting the Bit Error Ratio (BER) versus

(SNR) for AWGN channel. Simulation runs for different generator polynomials, and constraint lengths, also we take a varying trace back length. The constraint length equal (3, 4, 5, 6, and 7) with tracback length (10, 15, 20, 25, and 30), the simulation result for hard decoder is show in table (3.2), and for soft decoder shown in table (3.3). Figure (3.4) shows a comparative between them in Bit Error Rate (BER) for hard decision decoder, while figure (3.5) shows the BER for soft decision decoder. Figure (3.6) compares between soft and hard decoder with different constraint length (6, and 7). Decoding time for soft and hard decoder with different constraint length equal (3, 4, 5, 6, and 7) is shown in figure (3.7).

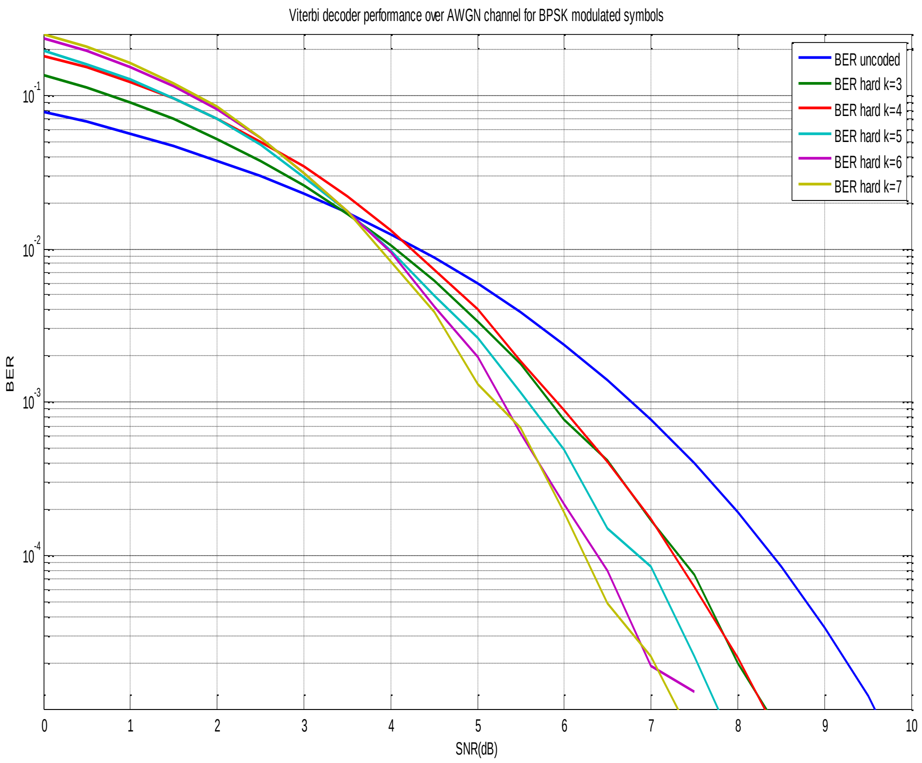


Figure (3.4) Performance of Hard Decoder for Different Constraint Length

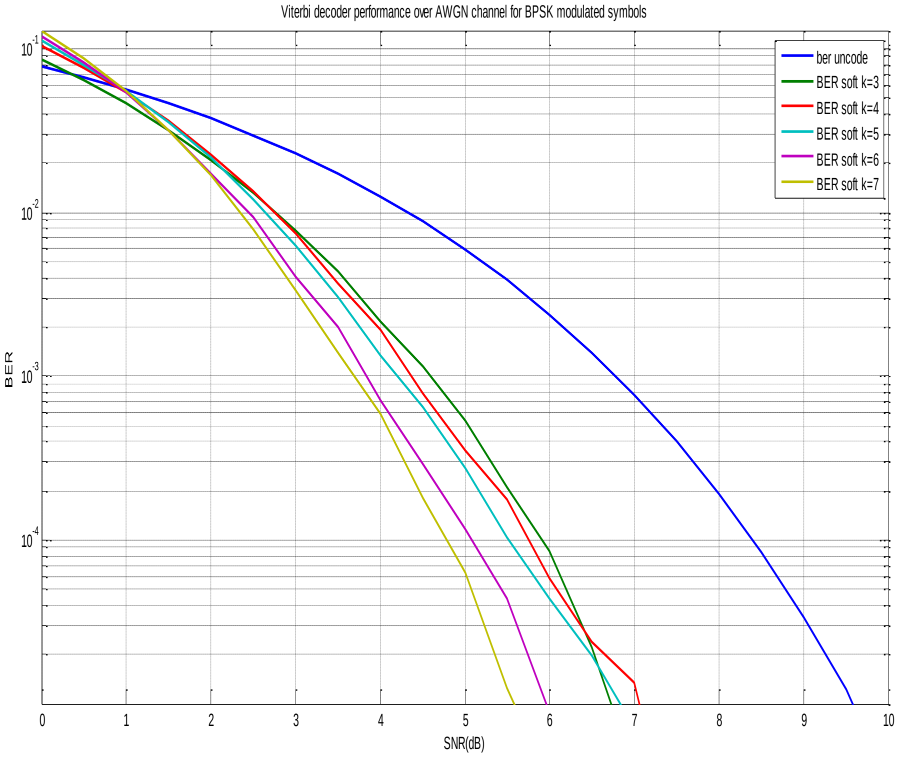


Figure (3.5) Performance of Soft Decoder for Different Constraint Length

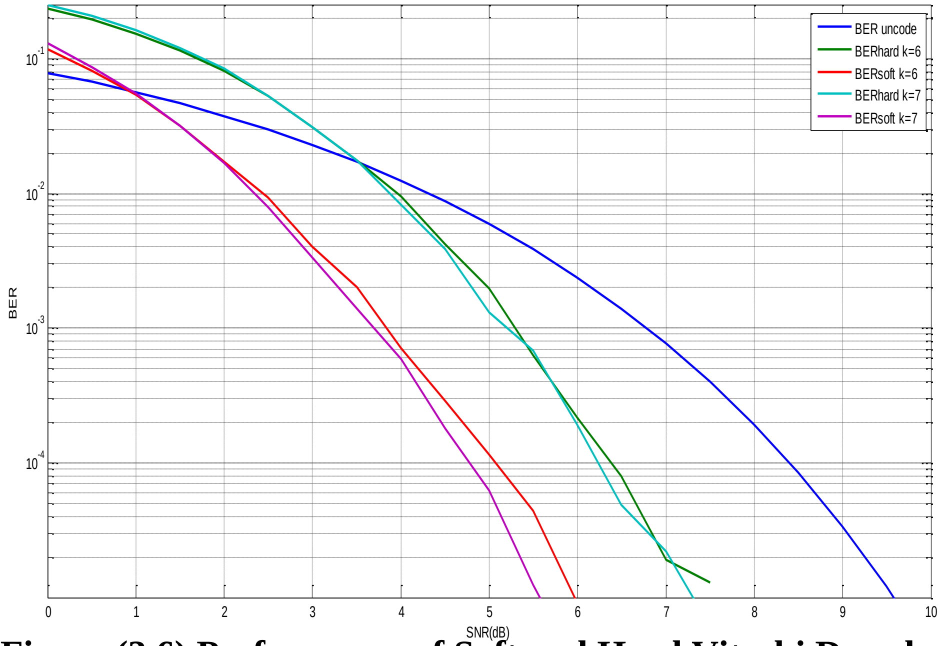
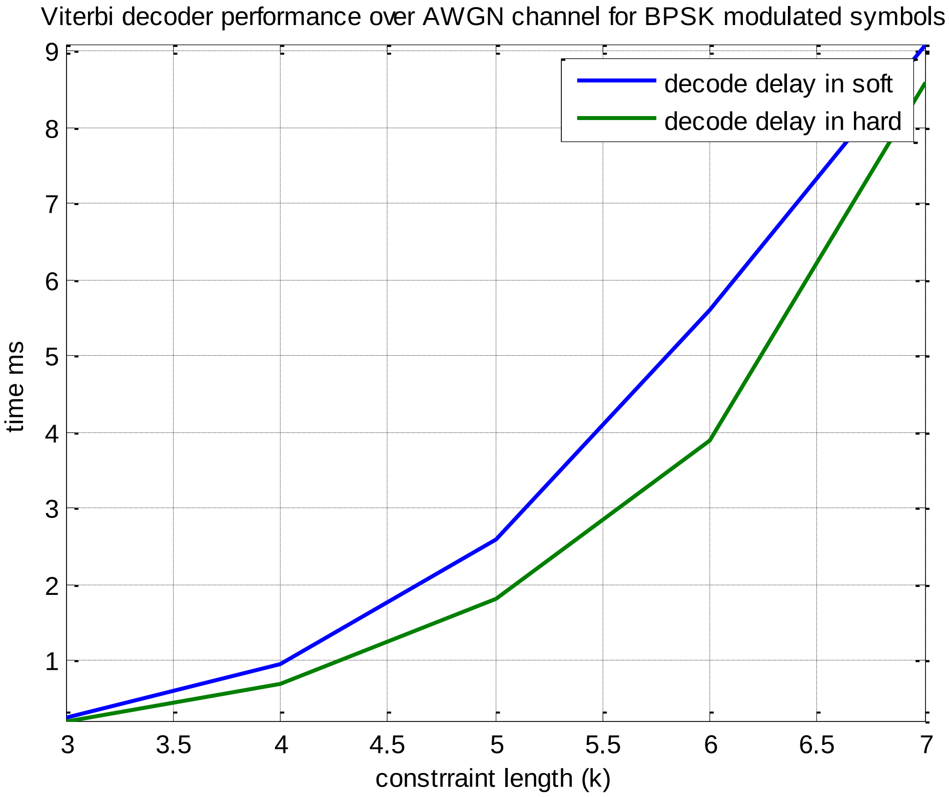


Figure (3.6) Performance of Soft and Hard Viterbi Decoder at Constraint Lengths of (6 and 7)



Figure(3.7) Decoding Delay time in Hard Decoder and Soft Decoder for Constraint Length (3,4,5,6 and 7)

Table (3.2). BER for Hard Viterbi Decoder

BER with code rate(1/2) for hard decoder					
SNR	**BER for K=3&TB=10**	**BER for K=4&TB=15**	**BER for K=5&TB=20**	**BER for K=6&TB=25**	**BER for K=7&TB=30**
0	0.135604	0.182164	0.1965061	0.235999	0.2530552
0.5	0.112859	0.1525342	0.161047	0.194081	0.2079072
1	0.090242	0.123613	0.1282135	0.154045	0.1642739
1.5	0.069822	0.09519	0.0966448	0.115681	0.1208208
2	0.051547	0.0706905	0.0705208	0.081346	0.0848927
2.5	0.037056	0.0503313	0.0474833	0.052852	0.0534071
3	0.026054	0.0341905	0.0291062	0.03139	0.0308927
3.5	0.016884	0.0218333	0.0175062	0.017679	0.0176219
4	0.010535	0.0131553	0.0097844	0.009456	0.0082125
4.5	0.00619	0.0072764	0.004934	0.004241	0.0038563
5	0.003354	0.0040031	0.0026229	0.001982	0.0013104
5.5	0.00177	0.0018623	0.0011615	0.000633	0.0006865
6	0.000769	0.0008881	0.0004917	0.000217	0.0001906
6.5	0.0004190	0.0004079	0.0001489	7.9e-05	4.895e-05
7	0.000171	0.0001729	8.541e-05	1.9e-05	2.188e-05
7.5	7.5e-05	6.211e-05	2.187e-05	1.3e-05	6.25e-06
8	2.e-05	2.173e-05	5.208e-06	0	8.333e-06
8.5	7.e-06	6.212e-06	3.125e-06	0	0
9	3.e-06	4.140e-06	2.083e-06	0	0
9.5	1.e-6	0	0	0	0
10	0	0	0	0	0

Table (3.3). BER for Soft Viterbi Decoder

BER with code rate(1/2) for soft decoder					
SNR	**BER for K=3&TB=10**	**BER for K=4&TB=15**	**BER for K=5&TB=20**	**BER for K=6&TB=25**	**BER for K=7&TB=30**
0	0.085191	0.1041759	0.1113188	0.118038	0.1292531
0.5	0.064603	0.0770683	0.0790052	0.082069	0.0873239
1	0.046605	0.0539182	0.0557542	0.053725	0.0552906
1.5	0.031789	0.0363385	0.0354198	0.032055	0.031826
2	0.020805	0.0224099	0.0215406	0.017093	0.016875
2.5	0.013233	0.0134772	0.0120042	0.009342	0.0079239
3	0.007716	0.0075186	0.0063042	0.004056	0.0033583
3.5	0.004333	0.0036563	0.0030208	0.001999	0.001391
4	0.002175	0.0019089	0.001325	0.000711	0.0005854
4.5	0.001145	0.0007795	0.0006521	0.00029	0.0001812
5	0.000532	0.0003509	0.0002760	0.000116	6.250e-05
5.5	0.000208	0.0001770	0.0001031	4.400e-05	1.250e-05
6	8.60e-05	5.797e-05	4.37e-05	9.e-06	3.125e-06
6.5	2.2e-05	2.381e-05	1.979e-05	5.00e-06	2.083e-06
7	4.0e-06	1.345e-05	7.291e-06	0	0
7.5	3.e-06	1.035e-06	4.166e-06	0	0
8	1.0e-06	0	0	0	0
8.5	0	0	0	0	0
9	0	0	0	0	0
9.5	0	0	0	0	0
10	0	0	0	0	0

3.5 Simulation Analysis

The results shown in the above figures are simulate by Matlab2013a. Figures show the performance of Viterbi decoder in both hard and soft decoder in BER for different constraint lengths and polynomial equations with code rate=1/2. Soft decoder was shown to have higher BER than hard decoder by (1 to 2) dB, which can be noticed from figures (5) and (6). Further improvement in BER can be achieve by increasing the constraint length in both cases. At constraint lengths of 6 and 7, the designed Viterbi decoder had, the highest performance calculated using BER and decoding time. From figure (3.6) and table (3.4) that is show the BER and decoding delay time for both hard and soft decoder at constraint length (6, and 7). It is clear that the performance of BER is almost similar at constraint lengths (6 and 7); therefore, we select the decoder for constraint length 6 in the designed Viterbi decoder. This is because of having lower complexity than length 7 with approximately same performance and less decoding time by 4.7ms and 3.5ms in hard and soft types as illustrate from table (3.5), respectively. The results show that the decoding time was faster in hard decision than that in soft decision.

Table (3.4). BER for Soft and Hard Viterbi Decoder at Constraint Lengths of (6 and 7)

BER and decoding delay with code rate ½				
SNR	**BER for K=6 (hard)**	**BER for K=6(soft)**	**BER for K=7(hard)**	**BER for K=7(soft)**
0	0.235999	0.118038	0.2530552	0.1292531
0.5	0.194081	0.082069	0.2079072	0.0873239
1	0.154045	0.053725	0.1642739	0.0552906
1.5	0.115681	0.032055	0.1208208	0.031826
2	0.081346	0.017093	0.0848927	0.016875
2.5	0.052852	0.009342	0.0534071	0.0079239
3	0.03139	0.004056	0.0308927	0.0033583
3.5	0.017679	0.001999	0.0176219	0.001391
4	0.009456	0.000711	0.0082125	0.0005854
4.5	0.004241	0.00029	0.0038563	0.0001812
5	0.001982	0.000116	0.0013104	6.250e-05
5.5	0.000633	4.400e-05	0.0006865	1.250e-05
6	0.000217	9.e-06	0.0001906	3.125e-06
6.5	7.9e-05	5.00e-06	4.895e-05	2.083e-06
7	1.9e-05	0	2.188e-05	0
7.5	1.3e-05	0	6.25e-06	0
8	0	0	8.333e-06	0
8.5	0	0	0	0
9	0	0	0	0

Table (3.5) Decoding Time for K 6 and 7 in Hard and Soft

Constraint Lengths (K)	**Decoding Delay Time in Hard (ms)**	**Decoding Delay Time in Soft (ms)**
6	3.9	5.6
7	8.6	9.1

Chapter Four

FPGA Implementation of Viterbi Decoder

4.1 Introduction

The Viterbi decoder was implemented using MATLAB program in order check the accuracy and performance efficiency of each stage of the design. The design was present in detail in chapter 3. However, time consumption and hardware utilization are highly important metrics that measure the efficiency of any design and require implementing the design, so that they are possible to be measure.

This chapter presents the FPGA implementation of the Viterbi decoder in detail. The synthesis stage of the proposed Viterbi decoder is carried out using Xilinx Nexys4 DDR board Artix-7 XC7A100T device CSG324 package -1 speed tool based on VHDL hardware description language at a code rate of ½ and constraint lengths K=6 and 7 for two types of Viterbi decoder, hard and soft decoding. The design is carry out with the help of Xilinx ISE14.7 as a register transfer level (RTL) design tool. This software provides a design environment for the engineering who uses hardware description languages to describe his own design, and enables to test the design operation before synthesizing it using one of the FPGA devices.

4.2 Implementation of Viterbi Decoder in VHDL

It is usually possible to describe an architecture through programming using more than one method. However, each method will result in a different resource allocation, which may lead to an inefficient utilization. Therefore, when describing a structure in VHDL, it is highly important to follow the method that leads to an optimal design in order to reduce hardware utilization of the FPGA device to the minimum and have a satisfactory efficient implementation.

A random input sequence bit is created and then has been encoded using convolutional encoder of constraint length K=6 and code rate 1/2 then transmitted over AWGN channel. In real case, the output codes symbol may contain some errors that occur through wireless transmission. This requires the use of AWGN to add an error to the symbols. AWGN is implemente in Matlab in order to get the noisy encoded symbols that are used as the signal given to the input of the Viterbi decoder implemented in VHDL.

The input signal processed in the VHDL code is defined as an array of 25 elements length, each element has a length 2-bit. This is because in the VHDL code, a trace back length of (25) is used, which is calculated according to equation (2.2). Therefore, the encoded symbol entry of the decoder has a length of fifty bits (twice the trace back decoder as a code rate of 1/2 is used).

Viterbi decoder consists of three units: Branch metric Unit, Add Compare Select Unit and Trace Back Unit, as explained in chapter 2. The FPGA implementation of these three units using VHDL language are describe in detail in the following subsections.

4.2.1 Branch Metric Unit

The Branch Metric unit accepts an input of 2-bits length with each clock cycle. This requires 25 clock cycle to process the total input coded symbols. For each input, 64 integer numbers representing the branch metrics of 32 states are generated, with their values ranging between 0 and 2. This is performed in VHDL using if statements. A flow chart that represents the implementation of Branch Metric Unit in VHDL shown in figure (4.1).

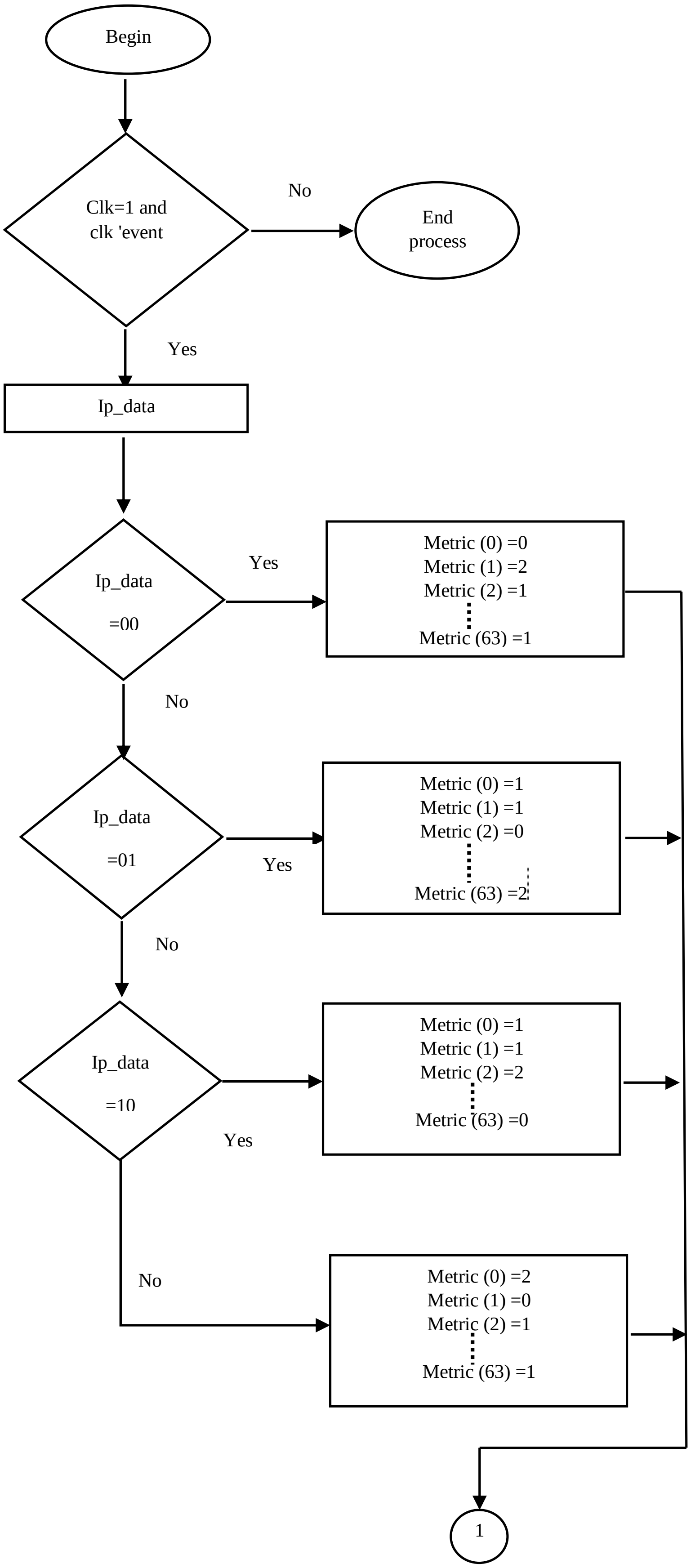


Figure (4.1) Flow Chart of Branch Metric Unit

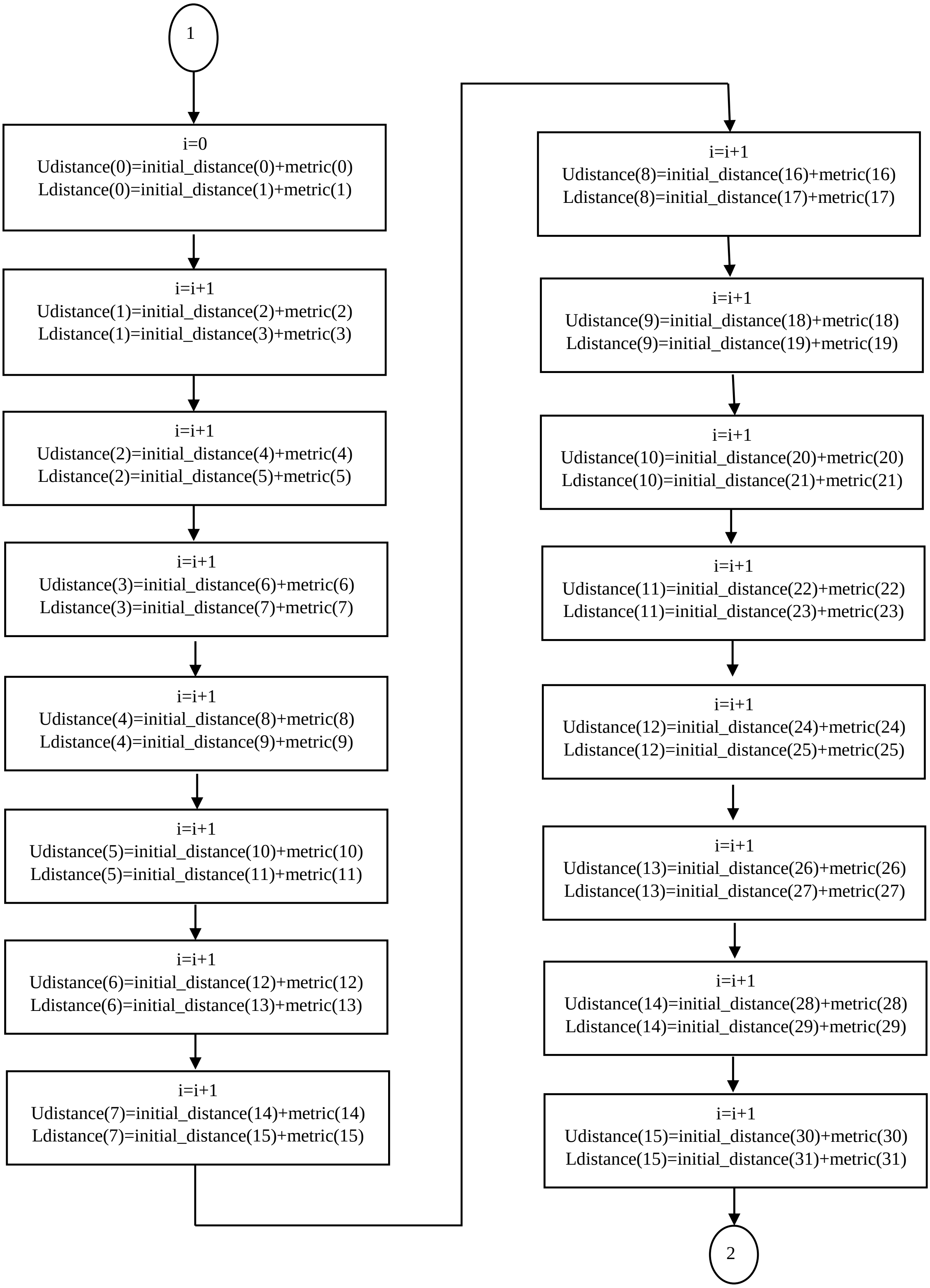
4.2.2 Add Compare Select Unit

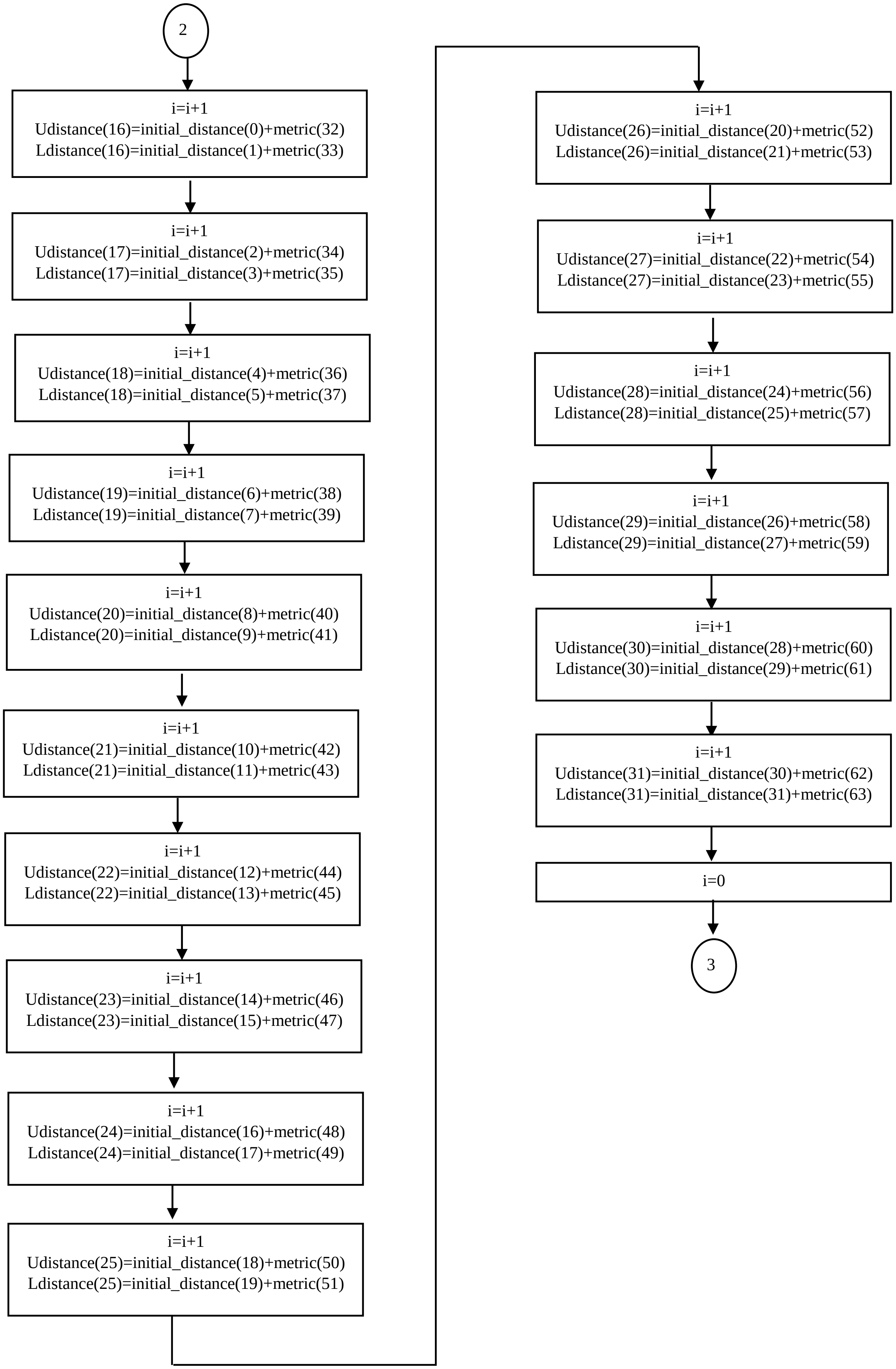
The Add Compare Select Unit receives the 64 integer output of the Branch Metric Unit and performs addition operation for the current state between each initial distance and the branch metric coming from previous state for a specific input. This requires allocating 64 addition circuits, each two consecutive additions belong to one state. Each addition circuit accepts two integer inputs, the first input (initial distance) ranges between 0 and 15 and the second input (branch metric) is between 0 and 2. The output produced from each addition is an integer ranging between 0 and 31.

In order to compare between the outputs of each two-addition operations that belong to one state, a comparison circuit required to be implemented. Therefore, for the 32 states, a 32 comparison circuit are implemented. This process of addition and comparison produces 32 distance with each clock cycle. After completing the calculation of all the distances, the number of the state that has the smallest distance save to be used in the final stage of Viterbi decoder for reconstructing the original data. A flow chart that represents the implementation of Add Compare Select Unit in VHDL shown in figure (4.2).

4.2.3 Trace Back Unit

The Trace Back Unit receives 32-integer distance for each input code from add compare select unit. Afterwards, it is begin to reconstructed original data. The operation of tracing back the trills starts from the state that has the lowest distance (the distance is the value of each of the previous unit outputs). The state value is integer number ranging from (0 to 31). If the state number between 0 and 15, the value of the branch that leads to this state is 0. While the state number between 16 and 31, the value of the branch that leads to this state is 1. By repeating the comparison operation that performed in the previous stage (ACSU) and led to the received distances, the value of the pre-state is predicted for the chosen distance. With each new clock cycle, one output bit is produce. Thus, the total output of this stage of 25-bit length resembles the reconstructed data. Therefore, each coded symbol require two clock cycle for reconstructed. A flow chart that represents the implementation of Trace Back Unit in VHDL shown in figure (4.3).


1
i=0
Udistance(0)=initial_distance(0)+metric(0)
Ldistance(0)=initial_distance(1)+metric(1)
i=i+1
Udistance(1)=initial_distance(2)+metric(2)
Ldistance(1)=initial_distance(3)+metric(3)
i=i+1
Udistance(2)=initial_distance(4)+metric(4)
Ldistance(2)=initial_distance(5)+metric(5)
i=i+1
Udistance(3)=initial_distance(6)+metric(6)
Ldistance(3)=initial_distance(7)+metric(7)
i=i+1
Udistance(4)=initial_distance(8)+metric(8)
Ldistance(4)=initial_distance(9)+metric(9)
i=i+1
Udistance(5)=initial_distance(10)+metric(10)
Ldistance(5)=initial_distance(11)+metric(11)
i=i+1
Udistance(6)=initial_distance(12)+metric(12)
Ldistance(6)=initial_distance(13)+metric(13)
i=i+1
Udistance(7)=initial_distance(14)+metric(14)
Ldistance(7)=initial_distance(15)+metric(15)
i=i+1
Udistance(8)=initial_distance(16)+metric(16)
Ldistance(8)=initial_distance(17)+metric(17)
i=i+1
Udistance(9)=initial_distance(18)+metric(18)
Ldistance(9)=initial_distance(19)+metric(19)
i=i+1
Udistance(10)=initial_distance(20)+metric(20)
Ldistance(10)=initial_distance(21)+metric(21)
i=i+1
Udistance(11)=initial_distance(22)+metric(22)
Ldistance(11)=initial_distance(23)+metric(23)
i=i+1
Udistance(12)=initial_distance(24)+metric(24)
Ldistance(12)=initial_distance(25)+metric(25)
i=i+1
Udistance(13)=initial_distance(26)+metric(26)
Ldistance(13)=initial_distance(27)+metric(27)
i=i+1
Udistance(14)=initial_distance(28)+metric(28)
Ldistance(14)=initial_distance(29)+metric(29)
i=i+1
Udistance(15)=initial_distance(30)+metric(30)
Ldistance(15)=initial_distance(31)+metric(31)
2


2
i=i+1
Udistance(16)=initial_distance(0)+metric(32)
Ldistance(16)=initial_distance(1)+metric(33)
i=i+1
Udistance(17)=initial_distance(2)+metric(34)
Ldistance(17)=initial_distance(3)+metric(35)
i=i+1
Udistance(18)=initial_distance(4)+metric(36)
Ldistance(18)=initial_distance(5)+metric(37)
i=i+1
Udistance(19)=initial_distance(6)+metric(38)
Ldistance(19)=initial_distance(7)+metric(39)
i=i+1
Udistance(20)=initial_distance(8)+metric(40)
Ldistance(20)=initial_distance(9)+metric(41)
i=i+1
Udistance(21)=initial_distance(10)+metric(42)
Ldistance(21)=initial_distance(11)+metric(43)
i=i+1
Udistance(22)=initial_distance(12)+metric(44)
Ldistance(22)=initial_distance(13)+metric(45)
i=i+1
Udistance(23)=initial_distance(14)+metric(46)
Ldistance(23)=initial_distance(15)+metric(47)
i=i+1
Udistance(24)=initial_distance(16)+metric(48)
Ldistance(24)=initial_distance(17)+metric(49)
i=i+1
Udistance(25)=initial_distance(18)+metric(50)
Ldistance(25)=initial_distance(19)+metric(51)
i=i+1
Udistance(26)=initial_distance(20)+metric(52)
Ldistance(26)=initial_distance(21)+metric(53)
i=i+1
Udistance(27)=initial_distance(22)+metric(54)
Ldistance(27)=initial_distance(23)+metric(55)
i=i+1
Udistance(28)=initial_distance(24)+metric(56)
Ldistance(28)=initial_distance(25)+metric(57)
i=i+1
Udistance(29)=initial_distance(26)+metric(58)
Ldistance(29)=initial_distance(27)+metric(59)
i=i+1
Udistance(30)=initial_distance(28)+metric(60)
Ldistance(30)=initial_distance(29)+metric(61)
i=i+1
Udistance(31)=initial_distance(30)+metric(62)
Ldistance(31)=initial_distance(31)+metric(63)
i=0
3

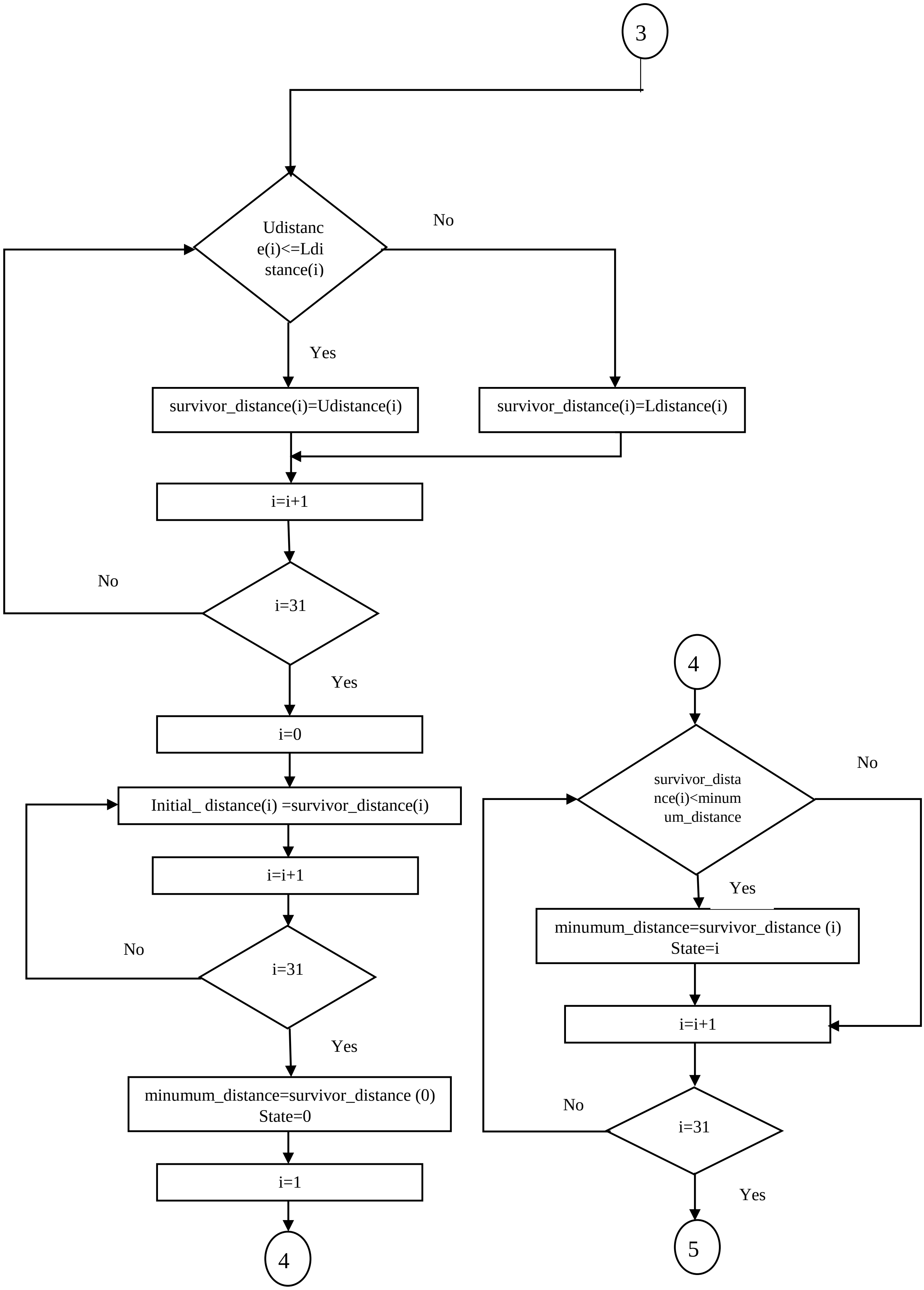

3
Udistanc
e(i)<=Ldi
stance(i)
No
Yes
survivor_distance(i)=Udistance(i)
survivor_distance(i)=Ldistance(i)
i=i+1
No
i=31
Yes
i=0
Initial_ distance(i) =survivor_distance(i)
i=i+1
No
i=31
Yes
minumum_distance=survivor_distance (0)
State=0
i=1
4
4
survivor_dista
nce(i)<minum
um_distance
No
Yes
minumum_distance=survivor_distance (i)
State=i
i=i+1
No
i=31
Yes
5

Figure (4.2) Flow Chart of Add Compare Select Unit

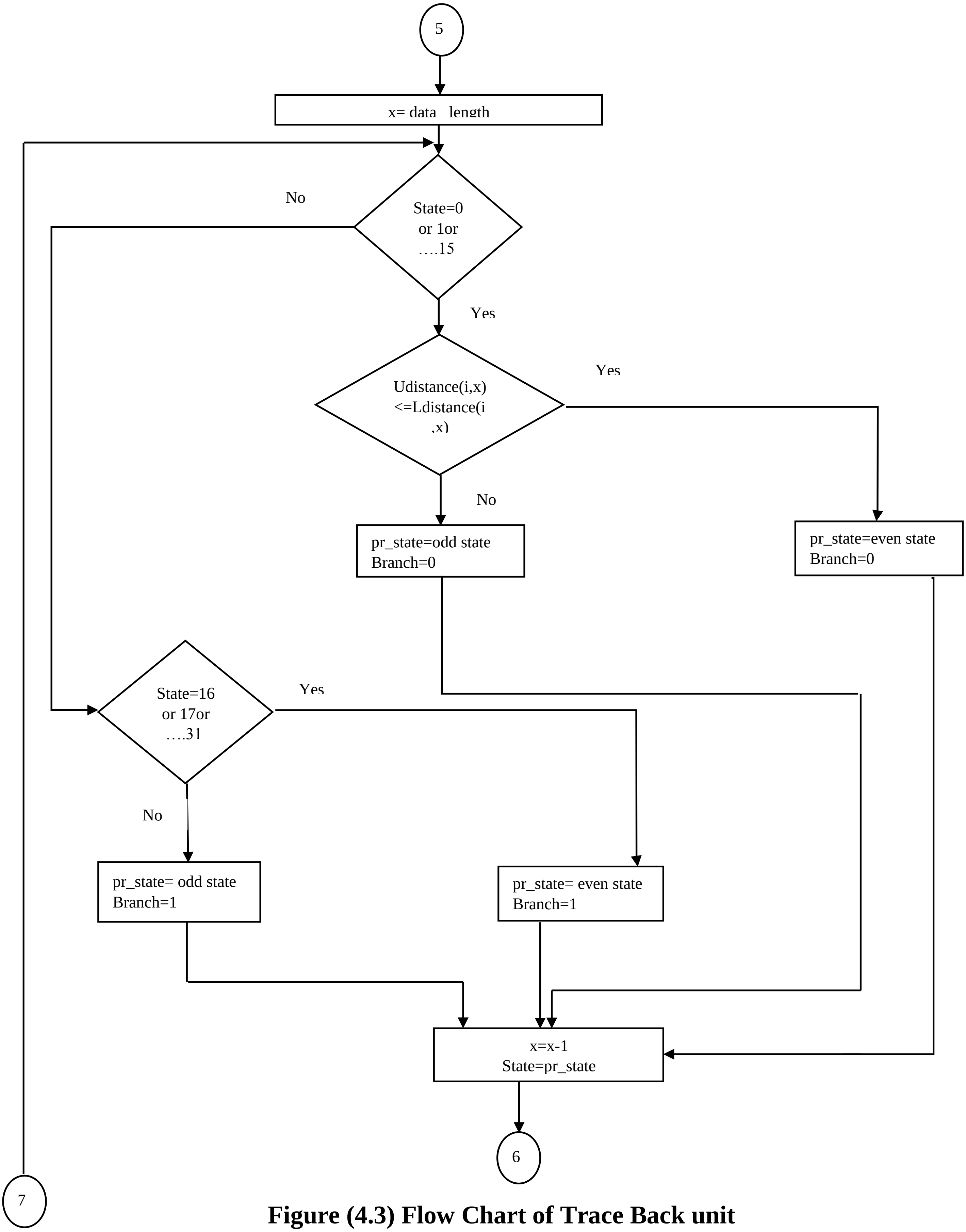


Figure (4.3) Flow Chart of Trace Back unit

4.3 Viterbi Decoder Simulation Result

After completing the design in VHDL, the design is simulate in order to verify the results using Xilinx ISE 14.7 simulator tool. The input values for the Viterbi decoder are insert in the code design; therefore, the simulation processes all input data directly. The simulation runs for two different constraint lengths (6 and 7); the following subsections describe the simulation output waveform of each unit.

4.3.1 Branch Metric Simulation Result

The first calculation in Viterbi decoder is computing the branch metric value. The simulation result of branch metric is presented in figures (4.5) and (4.6), with code rate (1/2), and constraint length 6 and 7 respectively, assuming that the original input sequence and the received code symbols are as shown below:

For Constraint Length = 6.

Input sequence: (25 Bits)

[0001000000110100110100000]

Encoded symbol: (50 Bits)

[00,00,00,11,10,11,10,01,11,00,11,01,01,10, 01,01,10,00,10,10,01,01,01,01,11]

Encoded symbol after noise:

[10,00,10,11,10,11,10,01,11,10,11,01,11,10,01,01,10,10,10,10,01,01,01,01,11]

For Constraint Length = 7.

Input sequence: (30 Bits)

[000010111110101100011010000000]

Encoded symbol: (60 Bits)

[00,00,00,00,11,01,00,01,01,00,01,11,10,11,01,00,01,10,00,01,11,01,11,10,01,10,11,
10,11,00]

Encoded symbol after noise:

[00,00,00,00,10,01,00,01,01,01,01,11,10,11,01,00,01,10,00,01,11,01,11,01,01,01,00,
10,01,11].

The received code symbol given to the input of the branch metric unit and the result shown in the figures. Figure (4.4) shows the computed 64 branch metric intended to 32 state for the encoded bits given above. In addition, figure (4.5) shows the computed 128 branch metric intended to 64 state for the encoded bits given above.

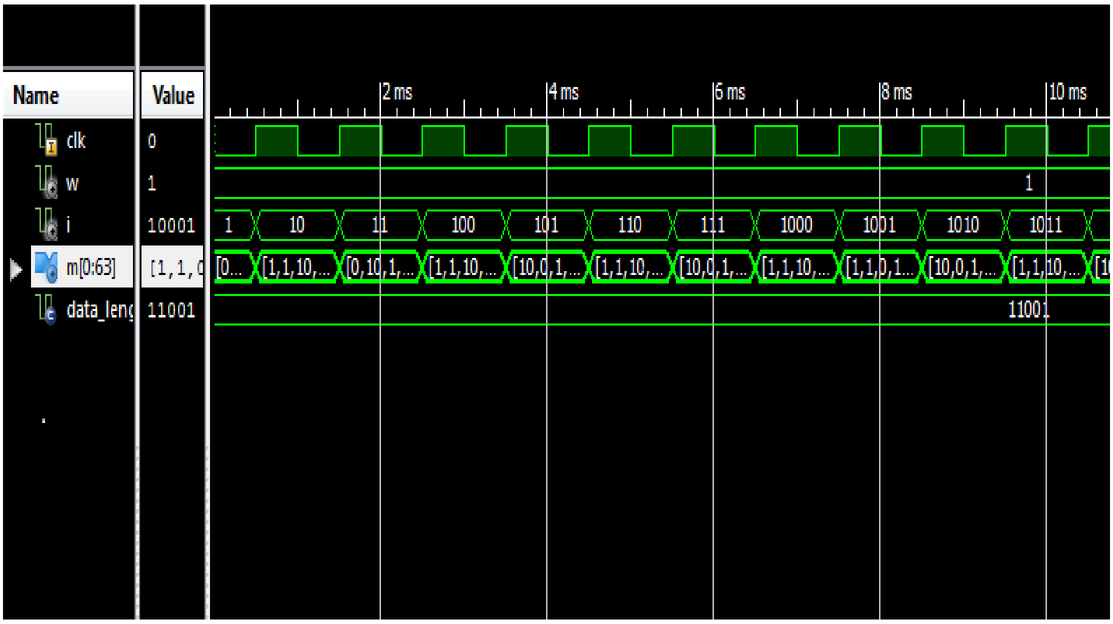


Figure (4.4) Simulation Result for BMU for Constraint Length = 6

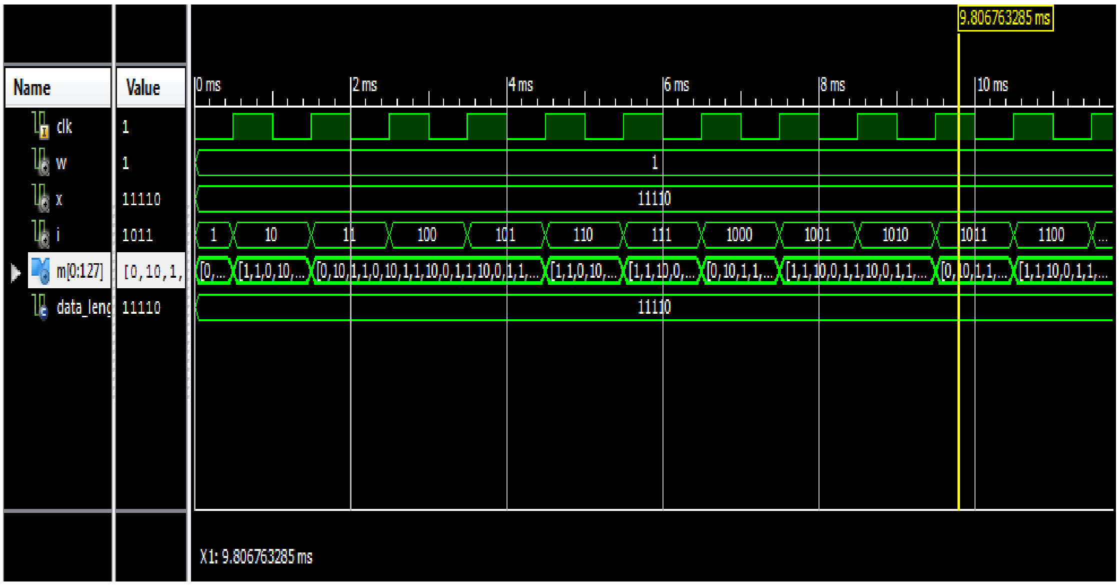


Figure (4.5) Simulation Result for BMU for Constraint Length = 7

4.3.2 Add Compare Select Simulation Result

The simulated of the ACSU is based on the result of BMU, figure (4.6) shows the simulation result for constraint length 6 viewing the 32-integer distance for each encoded input symbole and figure (4.7) shows the simulation result for constraint length 7 viewing the 64 -integer distance for each encoded input symbole

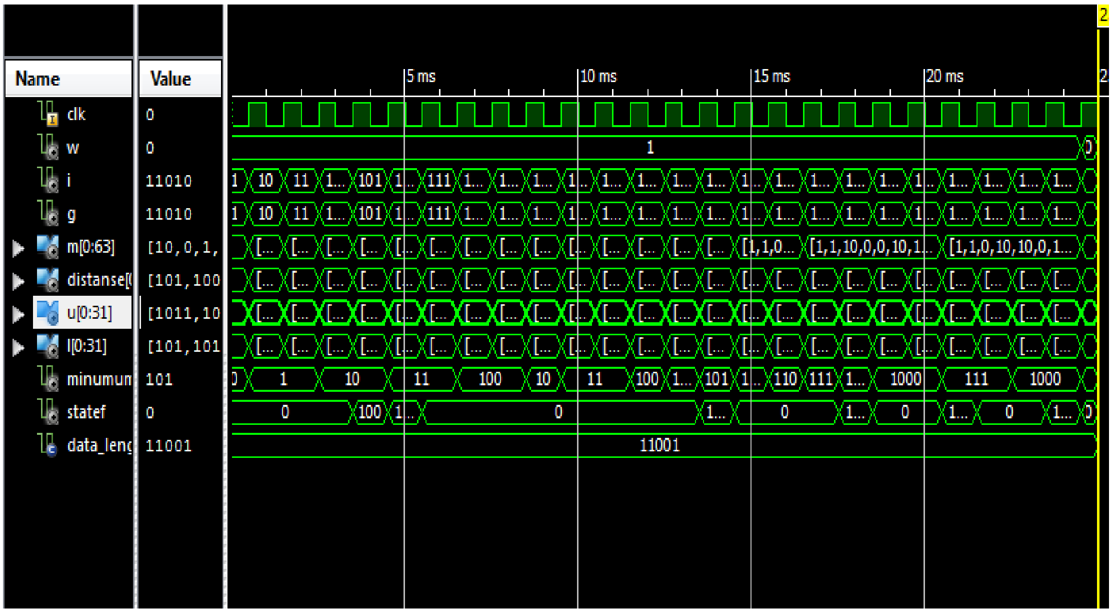


Figure (4.6) Simulation Result for ACSU for Constraint Length = 6

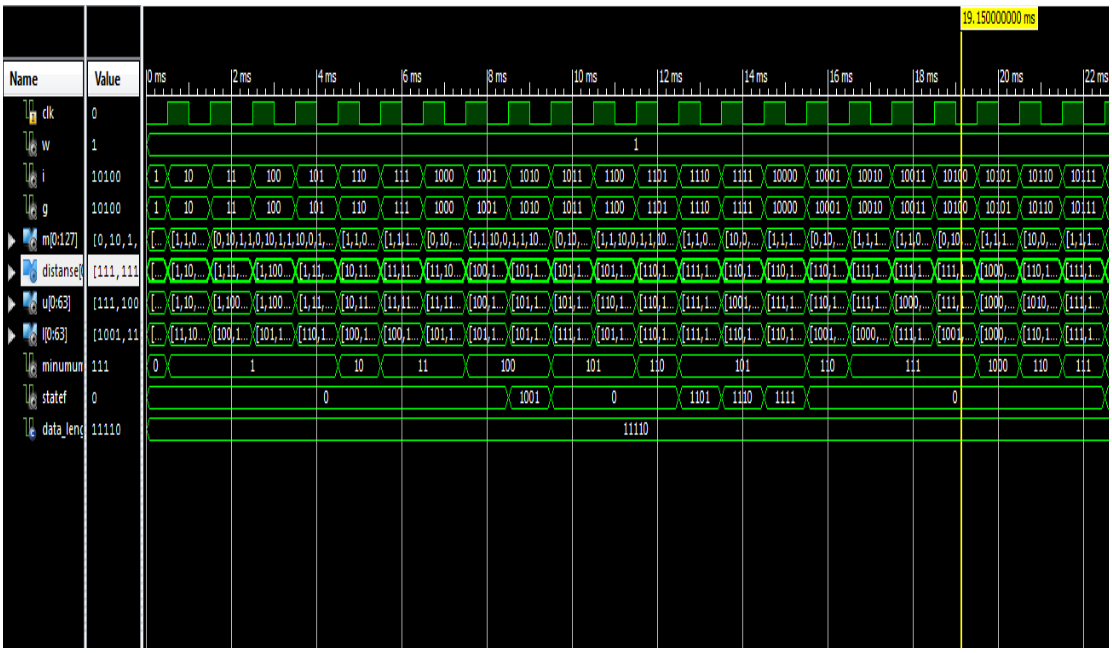


Figure (4.7) Simulation Result for ACSU for Constraint Length = 7

4.3.3 Trace Back Simulation Result

After BMU and ACSU complete their processes and this requires 25 clock cycle to process all the input data in the case of K=6 and 30 clock cycle when K=7, the TBU is now ready to obtain the output from previous unit and start reconstructing data. The output from this unit represents the original 25-bits input sequence estimated by Viterbi decoder when K=6, or 30-bits when K=7, depending on its calculations on the upper and lower branch computed from ACSU to obtain the correct value that represents the previous state. The simulation result of this unit in the case of constraint length 6 is shown in figure (4.8) while figure (4.9) show the simulation result when constraint length 7.

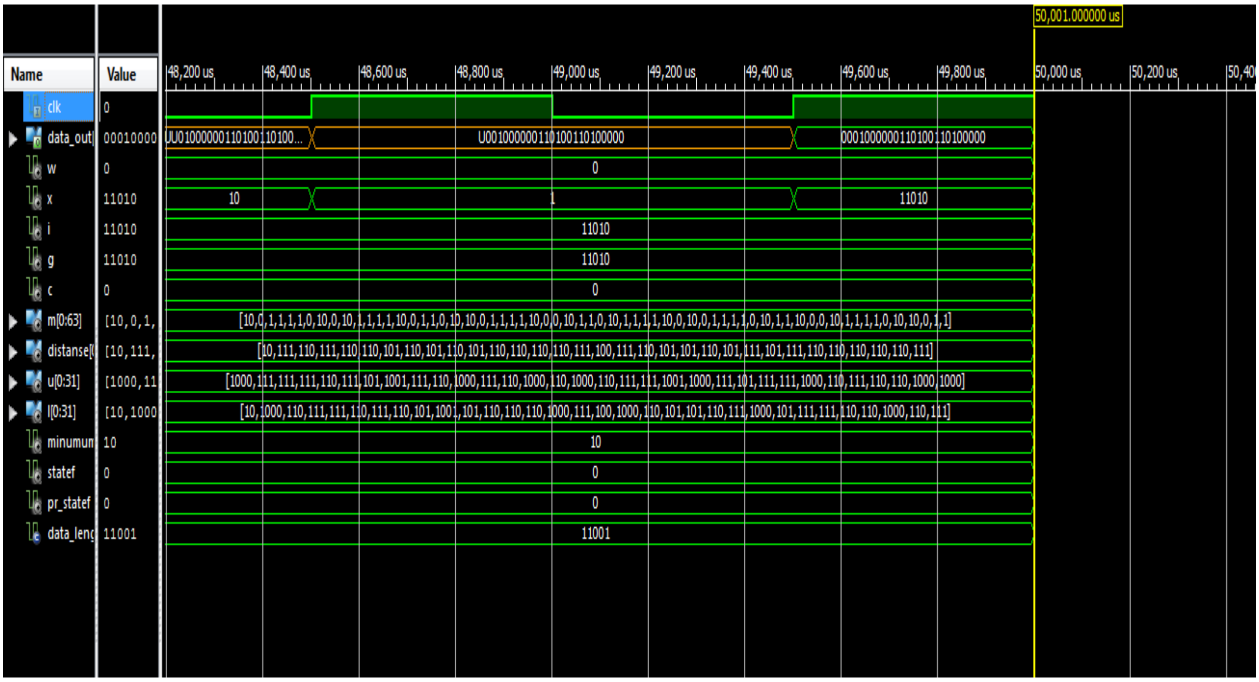


Figure (4.8) Simulation Result for TBU of Constraint Length = 6

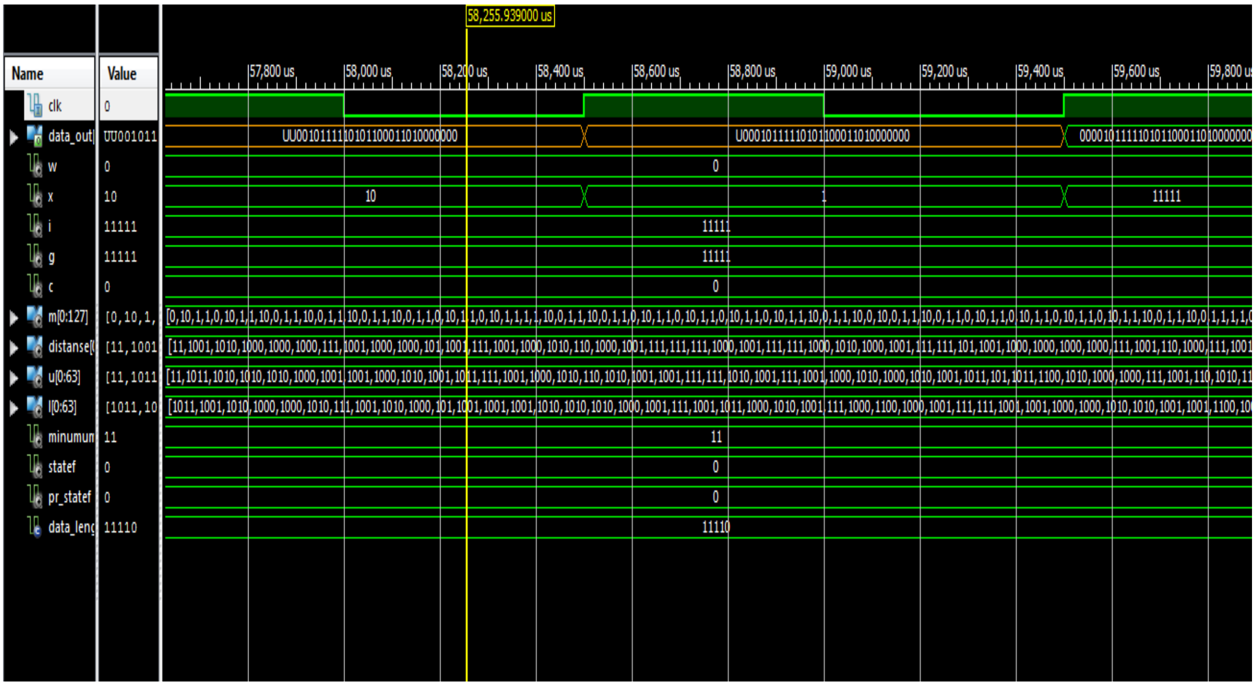


Figure (4.9) Simulation Result for TBU of Constraint Length = 7

4.4 Synthesis and Implementation Viterbi Decoder

After checking the behavior of the design and reviewing results using simulation, this design is ready to be synthesize and implement using Xilinx Artix-7 XC7A100T FPGA device, and to check the amount of hardware resources required to implement the design. Figure (4.10) shows the overall architecture design of Viterbi decoder.

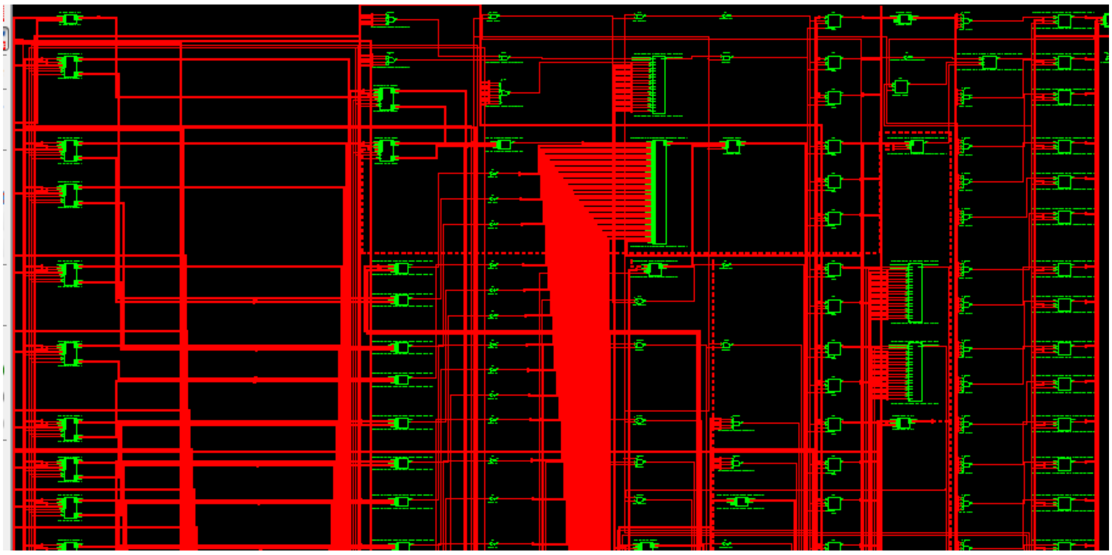


Figure (4.10) Architecture Design of Viterbi decoder

Figure (4.11) show the hardware implementation of Viterbi decoder on Xilinx Artix-7 board with showing of LEDs that are activate indicating the outputs produced from Viterbi decoder.

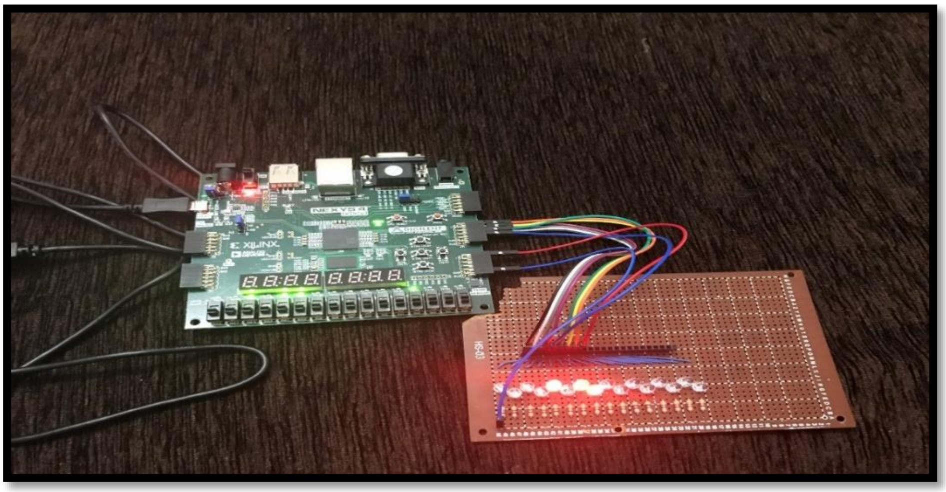
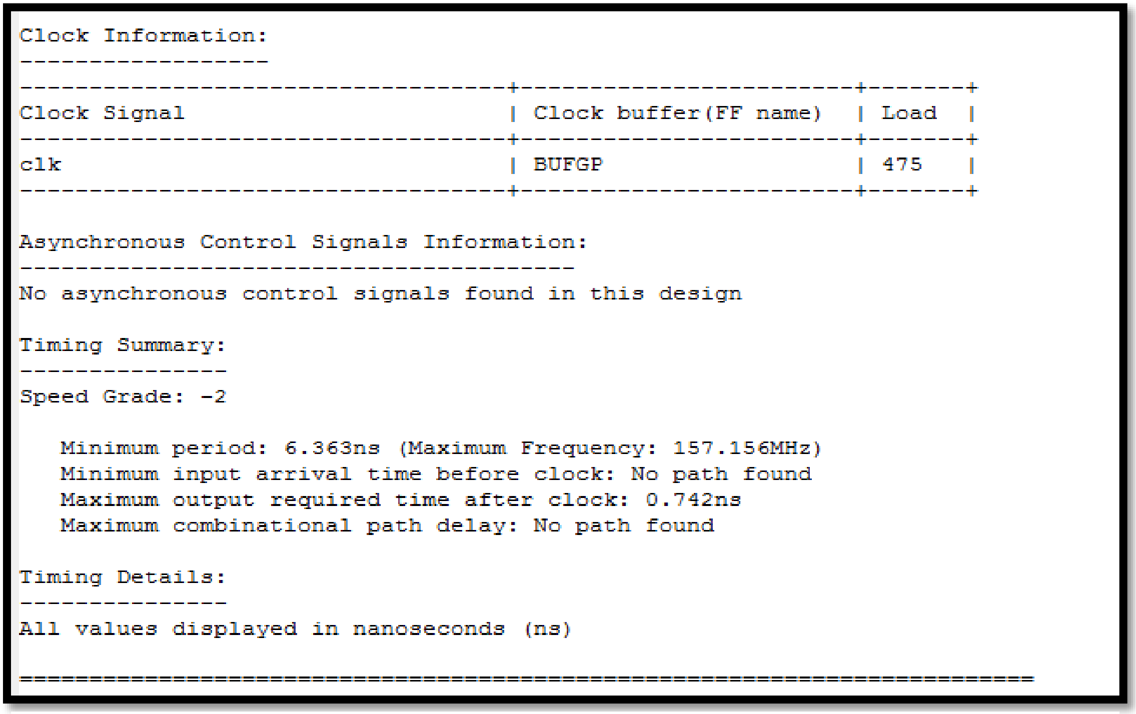


Figure (4.11) Hardware Implementation of the Design

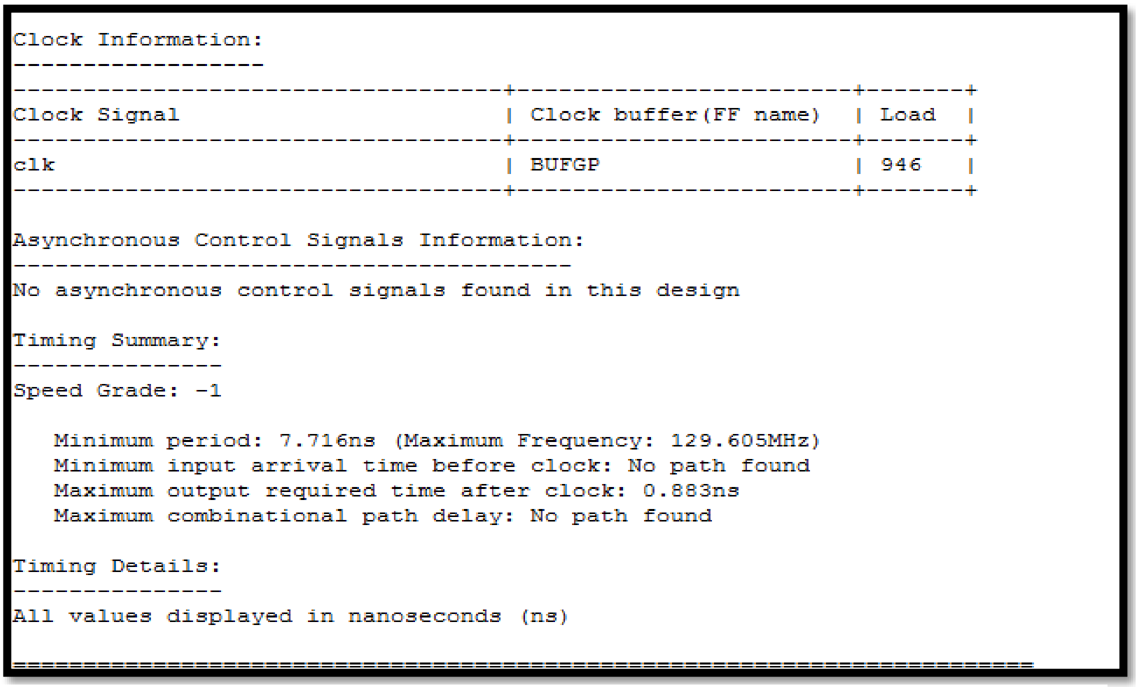
After completing the synthesis of the design, the synthesis report is generated and it is found that the minimum period is equal to (6.363ns) and the maximum operating frequency is equal to (157.156MHz) for constraint length 6 in hard type, as shown in figure (4.12).



```
Clock Information:
------------------
-----------------------------------+------------------------+-------+
Clock Signal                       | Clock buffer(FF name)  | Load  |
-----------------------------------+------------------------+-------+
clk                                | BUFGP                  | 475   |
-----------------------------------+------------------------+-------+

Asynchronous Control Signals Information:
----------------------------------------
No asynchronous control signals found in this design

Timing Summary:
---------------
Speed Grade: -2

   Minimum period: 6.363ns (Maximum Frequency: 157.156MHz)
   Minimum input arrival time before clock: No path found
   Maximum output required time after clock: 0.742ns
   Maximum combinational path delay: No path found

Timing Details:
---------------
All values displayed in nanoseconds (ns)

=========================================================================
```

Figure (4.12) Time Summary of the Design for hard Viterbi decoder at Constraint Length = 6

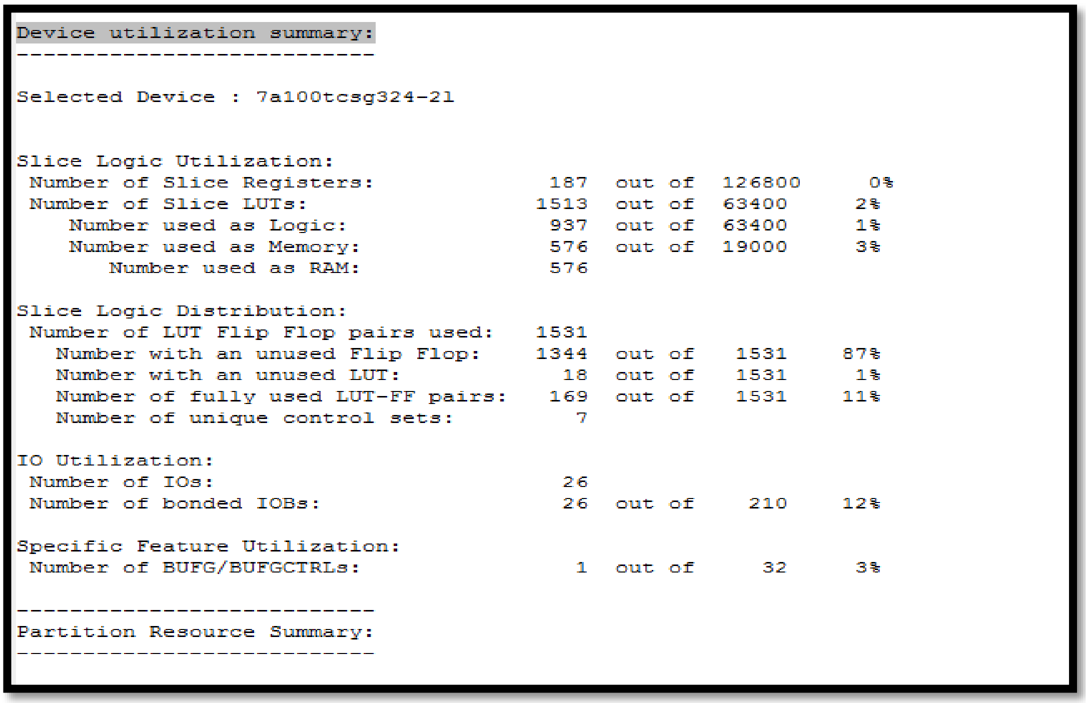
Figure (4.13) illustration that the minimum period is equal to (7.716ns) and the maximum operating frequency is equal to (129.605MHz) for constraint length 7.



```
Clock Information:
------------------
-----------------------------------+------------------------+-------+
Clock Signal                       | Clock buffer(FF name)  | Load  |
-----------------------------------+------------------------+-------+
clk                                | BUFGP                  | 946   |
-----------------------------------+------------------------+-------+

Asynchronous Control Signals Information:
----------------------------------------
No asynchronous control signals found in this design

Timing Summary:
---------------
Speed Grade: -1

   Minimum period: 7.716ns (Maximum Frequency: 129.605MHz)
   Minimum input arrival time before clock: No path found
   Maximum output required time after clock: 0.883ns
   Maximum combinational path delay: No path found

Timing Details:
---------------
All values displayed in nanoseconds (ns)

=========================================================================
```

Figure (4.13) Time Summary of the Design for hard Viterbi decoder at Constraint Length = 7

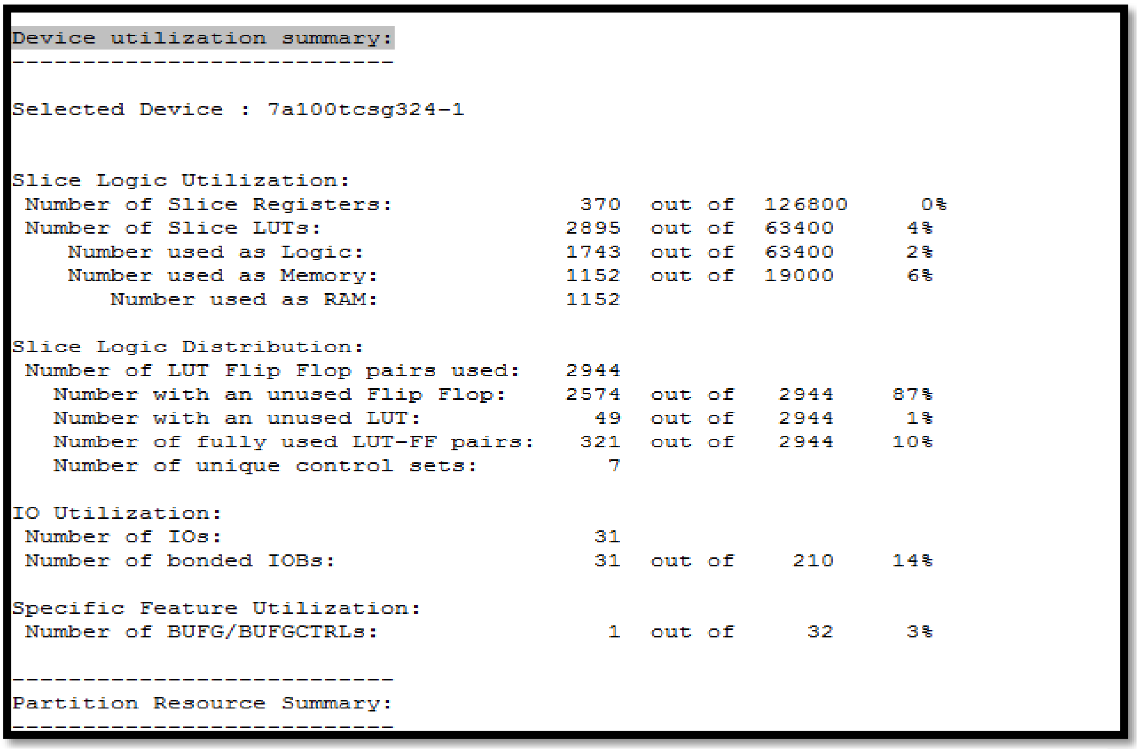
The device utilization of resources available in the ARTIX 7 kit for the design when K=6 in hard type is shown in figure (4.14).



```
Device utilization summary:
---------------------------

Selected Device : 7a100tcsg324-2l


Slice Logic Utilization:
 Number of Slice Registers:              187  out of  126800     0%
 Number of Slice LUTs:                  1513  out of  63400      2%
    Number used as Logic:                937  out of  63400      1%
    Number used as Memory:               576  out of  19000      3%
       Number used as RAM:               576

Slice Logic Distribution:
 Number of LUT Flip Flop pairs used:    1531
   Number with an unused Flip Flop:     1344  out of   1531     87%
   Number with an unused LUT:             18  out of   1531      1%
   Number of fully used LUT-FF pairs:    169  out of   1531     11%
   Number of unique control sets:          7

IO Utilization:
 Number of IOs:                           26
 Number of bonded IOBs:                   26  out of    210     12%

Specific Feature Utilization:
 Number of BUFG/BUFGCTRLs:                 1  out of     32      3%

---------------------------
Partition Resource Summary:
---------------------------
```

Figure (4.14) Device Utilization of the Design

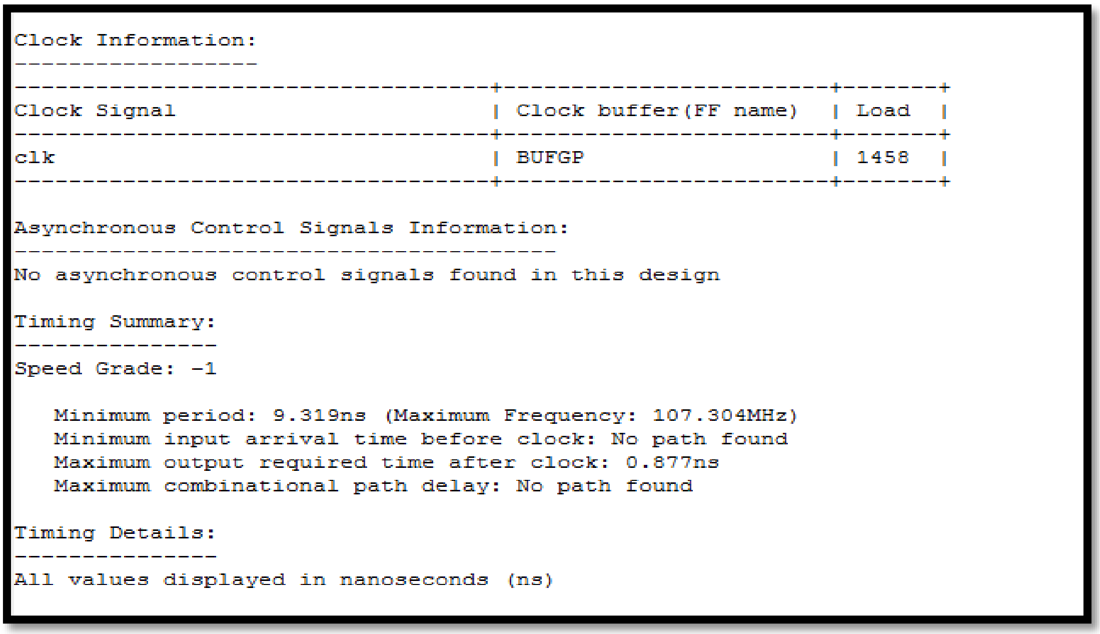
Figure (4.15) show the device utilization of resources available in the ARTIX 7 kit for the design when K=7.



```
Device utilization summary:
---------------------------

Selected Device : 7a100tcsg324-1


Slice Logic Utilization:
 Number of Slice Registers:             370  out of  126800     0%
 Number of Slice LUTs:                 2895  out of  63400      4%
    Number used as Logic:              1743  out of  63400      2%
    Number used as Memory:             1152  out of  19000      6%
       Number used as RAM:             1152

Slice Logic Distribution:
 Number of LUT Flip Flop pairs used:   2944
   Number with an unused Flip Flop:    2574  out of   2944     87%
   Number with an unused LUT:            49  out of   2944      1%
   Number of fully used LUT-FF pairs:   321  out of   2944     10%
   Number of unique control sets:         7

IO Utilization:
 Number of IOs:                          31
 Number of bonded IOBs:                  31  out of    210     14%

Specific Feature Utilization:
 Number of BUFG/BUFGCTRLs:                1  out of     32      3%

---------------------------
Partition Resource Summary:
---------------------------
```

Figure (4.15) Device Utilization of the Design

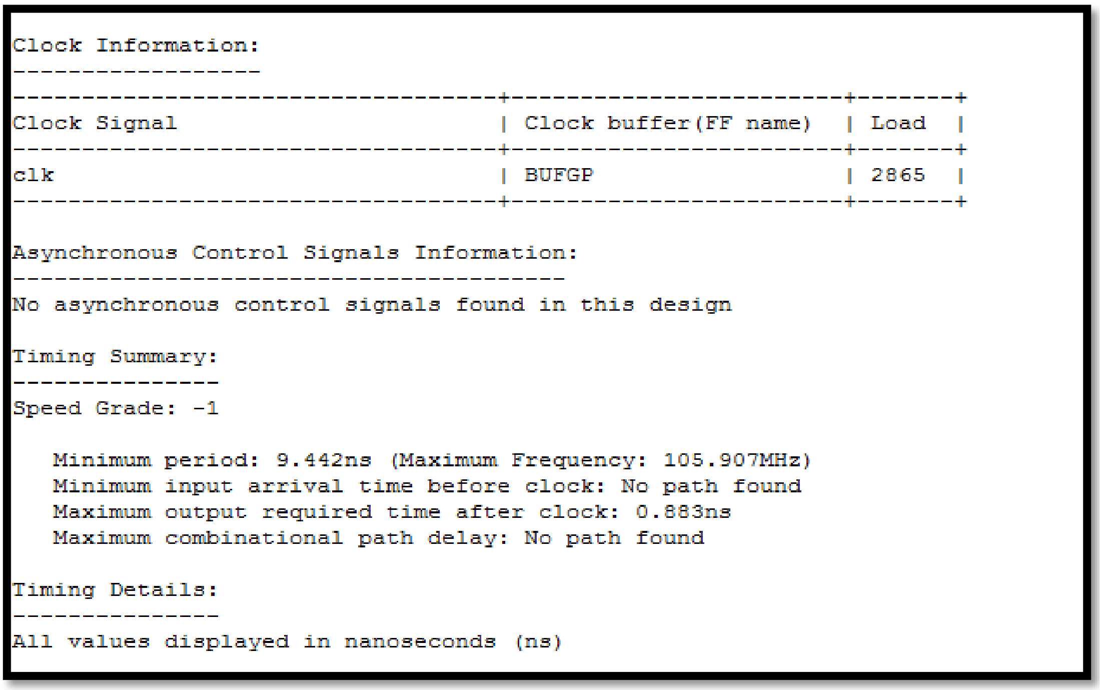
In the case of soft decoding type, for constraint length 6, figure (14.16) show the minimum period is equal to (9.319ns) and the maximum operating frequency is equal to (107.304MHz)



```
Clock Information:
------------------
-----------------------------------+------------------------+-------+
Clock Signal                       | Clock buffer(FF name)  | Load  |
-----------------------------------+------------------------+-------+
clk                                | BUFGP                  | 1458  |
-----------------------------------+------------------------+-------+

Asynchronous Control Signals Information:
----------------------------------------
No asynchronous control signals found in this design

Timing Summary:
---------------
Speed Grade: -1

   Minimum period: 9.319ns (Maximum Frequency: 107.304MHz)
   Minimum input arrival time before clock: No path found
   Maximum output required time after clock: 0.877ns
   Maximum combinational path delay: No path found

Timing Details:
---------------
All values displayed in nanoseconds (ns)
```

Figure (4.16) Time Summary of the Design for Soft Viterbi Decoder at Constraint Length = 6

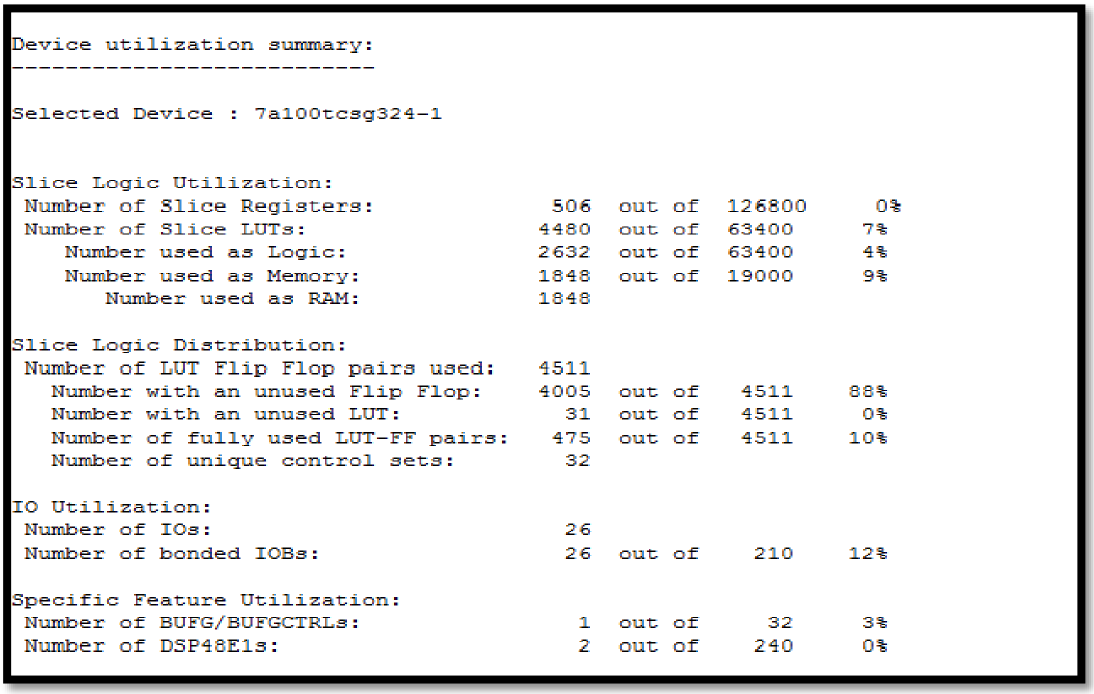
Figure (14.17) show the minimum period is equal to (9.442ns) and the maximum operating frequency is equal to (105.907MHz) when constraint length 7.



```
Clock Information:
------------------
-----------------------------------+------------------------+-------+
Clock Signal                       | Clock buffer(FF name)  | Load  |
-----------------------------------+------------------------+-------+
clk                                | BUFGP                  | 2865  |
-----------------------------------+------------------------+-------+

Asynchronous Control Signals Information:
----------------------------------------
No asynchronous control signals found in this design

Timing Summary:
---------------
Speed Grade: -1

   Minimum period: 9.442ns (Maximum Frequency: 105.907MHz)
   Minimum input arrival time before clock: No path found
   Maximum output required time after clock: 0.883ns
   Maximum combinational path delay: No path found

Timing Details:
---------------
All values displayed in nanoseconds (ns)
```

Figure (4.17) Time Summary of the Design for Soft Viterbi Decoder at Constraint Length = 7

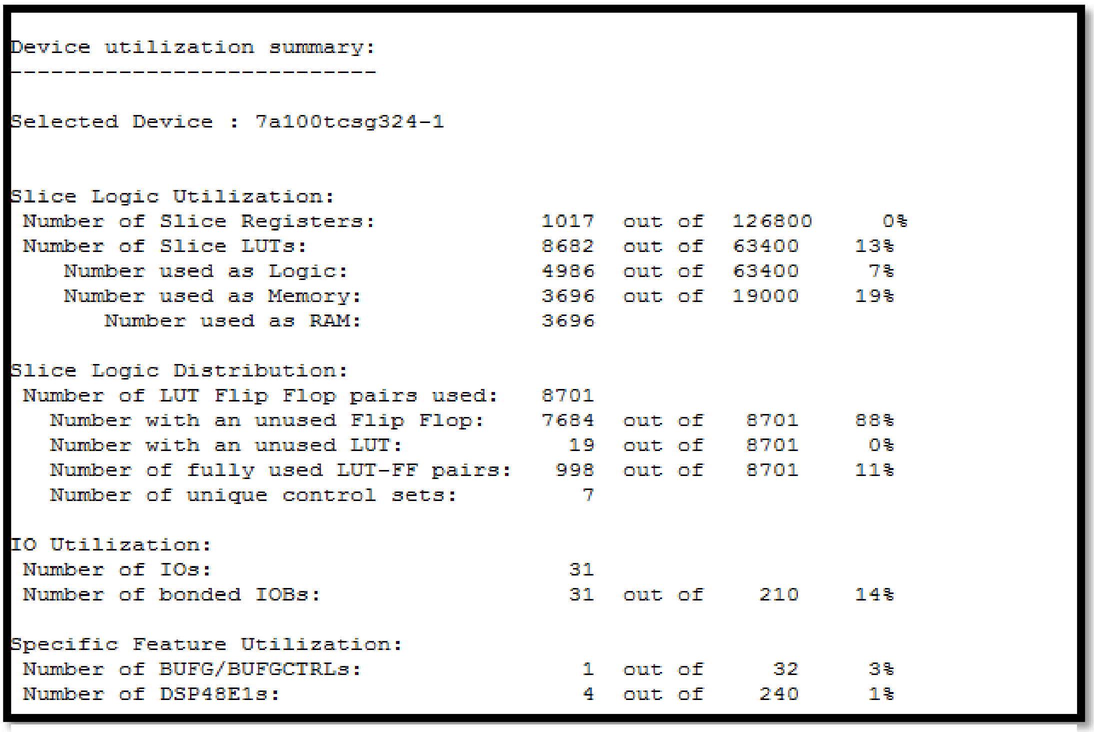
The device utilization of resources available in the ARTIX 7 kit for the design when K=6 in soft type is shown in figure (4.18).



```
Device utilization summary:
---------------------------

Selected Device : 7a100tcsg324-1


Slice Logic Utilization:
 Number of Slice Registers:              506  out of  126800     0%
 Number of Slice LUTs:                  4480  out of  63400      7%
    Number used as Logic:               2632  out of  63400      4%
    Number used as Memory:              1848  out of  19000      9%
       Number used as RAM:              1848

Slice Logic Distribution:
 Number of LUT Flip Flop pairs used:    4511
   Number with an unused Flip Flop:     4005  out of   4511     88%
   Number with an unused LUT:             31  out of   4511      0%
   Number of fully used LUT-FF pairs:    475  out of   4511     10%
   Number of unique control sets:         32

IO Utilization:
 Number of IOs:                           26
 Number of bonded IOBs:                   26  out of    210     12%

Specific Feature Utilization:
 Number of BUFG/BUFGCTRLs:                 1  out of     32      3%
 Number of DSP48E1s:                       2  out of    240      0%
```

Figure (4.18) Device Utilization of the Design

The device utilization of resources available in the ARTIX 7 kit for the design when K=7 in soft type is shown in figure (4.19).



```
Device utilization summary:
---------------------------

Selected Device : 7a100tcsg324-1


Slice Logic Utilization:
 Number of Slice Registers:            1017  out of  126800     0%
 Number of Slice LUTs:                 8682  out of   63400    13%
    Number used as Logic:              4986  out of   63400     7%
    Number used as Memory:             3696  out of   19000    19%
       Number used as RAM:             3696

Slice Logic Distribution:
 Number of LUT Flip Flop pairs used:   8701
   Number with an unused Flip Flop:    7684  out of    8701    88%
   Number with an unused LUT:            19  out of    8701     0%
   Number of fully used LUT-FF pairs:   998  out of    8701    11%
   Number of unique control sets:         7

IO Utilization:
 Number of IOs:                          31
 Number of bonded IOBs:                  31  out of     210    14%

Specific Feature Utilization:
 Number of BUFG/BUFGCTRLs:                1  out of      32     3%
 Number of DSP48E1s:                      4  out of     240     1%
```

Figure 4.19 Device Utilization of the Design

table (4.1) show the decoding time for K 6 and 7 in Hard and Soft viterbi decoder

Table (4.1) Decoding Time For K 6 and 7 in Hard and Soft

Constraint Lengths (K)	Decoding Delay Time in Hard (ns)	Decoding Delay Time in Soft (ns)
6	6.363	9.319
7	7.716	9.442

4.7 Hardware and Software Results Comparison

The performance of simulation result of the designed Viterbi decoder in VHDL is compare with a reference Viterbi decoder that has implemented in MATLAB program.

Table (4.2) explains the match between the results of MATLAB and VHDL code.

Table (4.2) A Comparison Result Between VHDL and MATLAB

SNR	1
Original input sequence	0001000000110100110100000
Encoded symbol	00,00,00,11,10,11,10,01,11,00,11,01,01,10,01,01,10,00,10,10,01,01,01,01,11
Encoded symbol after noise	10,00,10,11,10,11,10,01,11,10,11,01,11,10,01,01,10,10,10,10,01,01,01,01,11
Decoded data in MATLAB	0001000000110100110100000
Decoded data in VHDL	0001000000110100110100000

SNR	3
Original input sequence	0001000000110100110100000
Encoded symbol	00,00,00,11,10,11,10,01,11,00,11,01,01,10,01,01,10,00,10,10,01,01,01,01,11
Encoded symbol after noise	10,00,10,11,10,11,10,01,11,00,11,01,01,10,11,01,10,10,10,10,01,01,01,01,11
Decoded data in MATLAB	0001000000110100110100000
Decoded data in VHDL	0001000000110100110100000

SNR	5
Original input sequence	0001000000110100110100000
Encoded symbol	00,00,00,11,10,11,10,01,11,00,11,01,01,10,01,01,10,00,10,10,01,01,01,01,11
Encoded symbol after noise	00,00,00,11,11,11,10,01,11,00,11,01,01,10,01,01,10,00,10,10,00,01,01,01,11
Decoded data in MATLAB	0001000000110100110100000
Decoded data in VHDL	0001000000110100110100000

Chapter Five

Conclusions and Future Work

5.1 Conclusions

The FPGA implementation of Viterbi decoder given in chapter four has been test for efficiency in terms of frequency, delay and resource utilization. This is to find out the most suitable design of Viterbi decoder that satisfies the requirements of IoT applications. Results of comparison between constraint lengths of 6 and 7 found in Matlab are confirmed through the results of the FPGA implementation.

1. In Matlab, the two constraint lengths of 6 and 7 were found to have approximately similar BER measurements while the decoding delay was notably lower when using length 6 compared to length 7, in addition to having less code complexity because of having less number of computations.
2. In FPGA implementation, when the two constraint lengths are compare, length 6 resulted in lower resource utilization with higher frequency and lower decoding delay by 1.353ns and 0.123ns in hard and soft types as illustrate from table (4.1). The reduced time required to produce the implementation of Viterbi decoder using VHDL when length 6 is used can also be added to the advantages of using this length compared to the use of the constraint length of 7.
3. The FPGA implementation results achieve the requirements of IoT applications, where a frequency of less than 1GHz is achieved for the design that uses constraint lengths of 6 and 7. Decoding time achieved with length 6 is the reason of preferring this length over length 7 when implementing an IoT application, bearing in mind that length 6 also reduces the amount of hardware resource utilization and is easier to be implemented.

5.2 Future Work

The designed Viterbi decoder has more than one parameter that can be changed in order to achieve better performance or improve one of the implementation parameters. The following are the possible developments that can be applied to the implementation carried out in this thesis:

1- During the FPGA implementation of the first two stages of Viterbi decoder (BMU and ACSU), an output was designed to be processed at each clock cycle and is ready to be entered to the last stage (TBU). This helped improving the total decoding time of the decoder. The use of Pipelining technique in FPGA implementation of Viterbi decoder is suggested as a future work so that an output is achieved with each clock cycle which adds more improvement to the decoding delay time. This results from achieving a high throughput of double without affecting the amount of utilized hardware, leading to an efficient decoding time.

2- Another future work can be focused on changing the code rate parameter to other one and observing its effect on the implementation performance of Viterbi decoder at various constraint lengths.

References:

[1] Chunqiu Lu," Exploration of Low-power Viterbi Decoders Design for Low-throughput Application",Master Thesis Eindhoven, July 2016.

[2] Zeinab Kamal Aldein Mohammed," Internet of Things Applications, Challenges And Related Future Technologies", WSN 67(2) (2017) 126-148 EISSN 2392-2192.

[3] Diego Mendez1," Internet of Things: Survey on Security and Privacy", arXiv: 1707.01879v2 [cs.CR] 10 Jul 2017.

[4] Heba Aly," Big Data on Internet of Things: Applications, Architecture, Technologies , Techniques, and Future Directions", International Journal of Computer Science Engineering (IJCSE), ISSN: 2319-7323 Vol. 4 No.06 Nov 2015.

[5] Timothy Malche," Internet of Things (IoT) for building Smart Home System", International conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC 2017), 978-1-5090-3243-3/17/$31.00 ©2017 IEEE.

[6] Heetae Yang," IoT Smart Home Adoption: The Importance of Proper Level Automation", Hindawi Journal of Sensors, Volume 2018, Article ID 6464036, 11 pages, https://doi.org/10.1155/2018/6464036.

[7] Fang Hu," On the Application of the Internet of Things in the Field of Medical and Health Care",2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing, DOI 10.1109/GreenCom-iThings-CPSCom.2013.384.

[8] Rafiullah Khan," Future Internet: The Internet of Things Architecture, Possible Applications and Key Challenges",(pp. 257-260), DOI: 10.1109/FIT.2012.53.

[9] Ravi Gorli1," Future of Smart Farming with Internet of Things", Journal of Information Technology and Its Applications, Volume 2 Issue 1, April 2017, Page 27-38.

[10] Pallavi Sethi," Internet of Things: Architectures, Protocols, and Applications", Hindawi Journal of Electrical and Computer Engineering, Volume 2017, Article ID 9324035, 25 pages ,https://doi.org/10.1155/2017/9324035.

[11] Anass Sedrati," A Survey of Security Challenges in Internet of Things", Advances in Science, Technology and Engineering Systems Journal Vol. 3, No. 1, 274-280 (2018), ISSN: 2415- 6698,

[12] M.Praveen Kumar1," A SURVEY ON IoT PERFORMANCES IN BIG DATA", International Journal of Computer Science and Mobile Computing, IJCSMC, Vol. 6, Issue. 10, October 2017, pg.26 – 34, ISSN 2320–088X.

[13] Andrea Dodini," The Big Five IoT Challenges", POWER MEASUREMENT, Issue 3 2017 Power Electronics Europe.

[14] Yan Sun," FPGA Design and Implementation of a Convolutional Encoder and a Viterbi Decoder Based on 802.11a for OFDM", Wireless Engineering and technology, 2012, 3, 125-131.

[15] T. Adame," IEEE 802.11ah: The Wi-Fi Approach for M2M Communications", arXiv: 1402.4675v2 [cs.NI] 5 Oct 2014.

[16] Thi Hong Tran," Performance Evaluation of 802.11ah Viterbi Decoder for IoT Applications", 2015 International Conference on Advanced Technologies for Communications (ATC), 978-1-4673-8374-5/15/$31.00 ©2015 IEEE.

[17] Minyoung Park," IEEE 802.11ah: Sub-1-GHz License-Exempt Operation for the Internet of Things", IEEE Communications Magazine • September 2015, 0163-6804/15/$25.00 © 2015 IEEE.

[18] Hiromasa Kato," ASIC Design of a Low-Complexity K-best Viterbi Decoder for IoT Applications", APCCAS 2016, 978-1-5090-1570-2/16/$31.00 ©2016 IEEE.

[19] Thi Hong Tran," PER Evaluation of K-min Viterbi Decoder for Wireless Sensors", 2016 Tenth International Conference on Sensing Technology, 978-1-5090-0795-0/15/$31.00 ©2016.

[20] Rami Akeela, "Design and Verification of IEEE 802.11ah for IoTand M2M Applications", Conference Paper March 2017, DOI:10. 109/PERCOMW.2017.7917612.

[21] Amit Kumar Sikder1," A Survey on Sensor-based Threats to Internet-of-Things (IoT) Devices and Applications", arXiv: 1802.02041v1 [cs.CR] 6 Feb 2018.

[22] Victor Baños-Gonzalez 1," IEEE 802.11ah: A technology to face the IoT Challenge", Sensors 2016, 16, 11; doi: 10.3390/s16111960.

[23] Lou frenzel," What is the Difference between IEEE 802.11ah and 802.11af in the IoT? "Electronic Design, Article Jul 17 ,2017.

[24] N. Ahmed," A comparison of 802.11ah and 802.15.4 for IoT", The Korean Institute of Communications and Information Science, ICT Express 2 (2016) 100–102.

[25] Mahmoud Elkhodr," Emerging Wireless Technologies In The Internet Of Things: A Comparative Study", International Journal of Wireless & Mobile Networks (IJWMN) Vol. 8, No. 5, October 2016, DOI: 10.5121/ijwmn.2016.8505.

[26] Aqeel-ur-Rehman1," Communication Technology That Suits IoT – A Critical Review", Conference Paper · April 2013, DOI: 10.1007/978-3-642-41054-3_2, pp. 14–25, 2013.

[27] Avinash P. Ingle," Internet of Things (IoT): Vision, Review, Drivers of IoT, Sensors Nodes, Communication Technologies and Architecture", International Journal of Advances in Computer and Electronics Engineering Volume 2, Issue 8, August 2017, pp. 1–7.

[28] Pallavi Sethi," Internet of Things: Architectures, Protocols, and Applications", Hindawi Journal of Electrical and Computer Engineering, Volume 2017, Article ID 9324035, 25 pages.

[29] Xuying Zhao," A High Performance Multi-standard Viterbi Decoder",978-1-5090-3025-5/17/$31.00 ©2017 IEEE.

[30] Kapil Gupta," A Comparative Study of Viterbi and Fano Decoding Algorithm for Convolution Codes", AIP Conference Proceedings 1324, 34 (2010), doi: 10.1063/1.3526231.

[31] P. Kranthi," Optimization of the Decoding Performance of Rate ⅓ Convolutional Code ",International Journal of Innovative Research & Development, ISSN 2278 – 0211(Online), August, 2014 Vol 3 Issue 8.

[32] Poonam Beniwal1," Convolution Code Encoder Design Using Particle Swarm Optimization", International Journal of Electronics Engineering, 4 (1), 2012, pp. 65– 67, ISSN: 0973-7383,

[33] K.Cholan, a," Design and Implementation of Low Power High Speed Viterbi Decoder", International Conference on Communication Technology and System Design 2011, Procedia Engineering 30 (2012) 61 – 68.

[34] Deepa Kumari," Design and Performance Analysis of Convolutional Encoder and Viterbi Decoder for Various Generator Polynomials", Int. Journal of Engineering Research and Applications, ISSN: 2248-9622, Vol. 6, Issue 5, (Part - 2) May 2016, pp.67-71.

[35] Suneha Gupta,"Design and Implementation of an Optimized Viterbi Decoder", A thesis submitted in partial fulfillment of the requirements for the award of degree of master of technology in VLSI design and CAD, Thapar University July 2012

[36] Mahe Jabeen," Design of Convolution Encoder and Reconfigurable Viterbi Decoder", International Journal of Engineering and Science,ISSN: 2278-4721, Vol. 1, Issue 3 (Sept 2012), PP 15-21.

[37] Amjad Saeed Khan, "Hardware Implementation and performance valuation of Viterbi Decoder using Parallel decoding approach over Xilinx Virtex-6 FPGA", International Journal for Research and Development in Engineering ISSN: 2279-0500.Vol2: Issue3: pp- 01-09. IJRDE 2014. Volume2: Issue3

[38] Ms. Shankari N," Hard Decision Viterbi Decoder: Implementation on FPGA and Comparison of Resource Utilization on Different FPGA Devices", Int. Journal of Engineering Research and Application, ISSN: 2248-9622, Vol. 6, Issue 5, (Part -6) May 2016, pp.43-47.

[39] Ch Sandeep Reddy," FPGA Implementation of Convolution Encoder and Viterbi Decoder", International Journal of Research in Electronics & Communication Technology, Volume 1, Issue 2, October-December, 2013, pp. 166-172, © IASTER 2013 , www.iaster.com, ISSN Online: 2347-6109, print: 2348-0017.

[40] MATLAB Help 2017A.

[41] Ognjen Šcekic," FPGA Comparative Analysis", University of Belgrade ETF – School of Electrical Engineering, Belgrade 2005.

[42] www.xilinx.com," ISE In-Depth Tutorial", UG695 (v14.1) April 24, 2012.

[43] C.Maxfied,"The Design Warrior's Guide to FPGA", Mentor Graphics Corporation And Xilinx, Inc., ISBN:0-7506-7604-3,2004.

[44] Xilinx, Inc. "Introduction to FPGA Design with Vivado High-Level Synthesis (UG998) ", 2013.

[45] Remigiusz WISNIEWSKI," Synthesis of Compositional Microprogram Control Units for Programmable Devices", University of Zielona Góra Press, Poland, 2009, ISBN 978-83-7481-293-1.

[46] Chandra S. Mulpuri," Runtime and Quality Tradeoffs in FPGA Placement and Routing", Master's Thesis July 2001, Northwestern University.

[47] Charles H.Roth, Jr,"Digital Systems Design Using VHDL", Pws Publishing Company ,20 Park Plaza, Boston MA 02116-4324,ISBN:0-534-95099-X,1999.

الخلاصة

يعتبر فاتح الترميز فيتربي (Viterbi decoder)المستخدم لفك تشفير المرمز التلافيفي (convolutional codes) في أنظمة الاتصالات من التقنيات القوية لتصحيح الأخطاء الناتجة عن عمليات الارسال والاستلام.

قد تتعرض البيانات اثناء عمليات الارسال عبر قنوات الاتصالات اللاسلكية في تطبيقات انترنيت الاشياء الى الضوضاء والتشويه والتداخل، مما يؤدي إلى ظهور العديد من الأخطاء والضياع في البيانات في جانب الاستلام. ويعتبر معيار النقل اللاسلكي 802.11ah الذي تم اعتماده كبروتوكول اتصال لاسلكي في تطبيقات انترنيت الأشياء أحد الطرق المهمة لتصحيح هذه الأخطاء.

يتضمن تصميم فاتح الترميز فيتربي موازنة بين الأداء من جانب وتعقيد التصميم ووقت فك التشفير من جانب آخر. تم تصميم وتنفيذ فاتح الترميز فيتربي في هذه الرسالة باستخدام برنامج المحاكاة Matlab (2013a)ومجموعة تصميم مصفوفة البوابات المنطقية المبرمجة حقلياً (FPGA)، وذلك لتقييم عمل فاتح الترميز فيتربي ومقارنة أدائه عند استخدام مواصفات وإعدادات مختلفة. باستخدام برنامج Matlabتم اختبار أداء فاتح الترميز فيتربي لخمسة نماذج تشفير بأطوال قيد تساوي ٣ و٤ و٥ و٦ و٧، وباستخدام معدل كود (٢/١)، حيث تم إجراء مقارنة بين هذه النماذج وتقييم أداء كل منها. وقد تم تقييم الأداء باستخدام وقت فك التشفير ونسبة الخطأ في البتات (BER) عند مستويات SNR مختلفة. تظهر النتائج أن أطوال القيد ٦ و٧ لها BER متشابهة وأعلى من الأطوال الأخرى التي تم تنفيذها، حيث يمتلك طول القيد ٦ تعقيدًا أقل ووقت فك تشفير أقل بمقدار ٤.٧مللي ثانية و٣.٥ مللي ثانية في النوعين الصلب والناعم، على التوالي.

أما التطبيق العملي على جهاز FPGA فقد استخدم مجموعة NEXYS 4DDR التي تتضمن تصميم فاتح الترميز فيتربي عند أطوال القيد ٦ و٧ وللنوعين الصلب والناعم. بالإضافة إلى امتلاكها تعقيداً أقل وسرعة أكبر في فك التشفير، يظهر أن الطول ٦ يستخدم موارداً أقل، ولهذا السبب فإن أفضل طريقة لتنفيذ متطلبات تصحيح أخطاء تطبيقات انترنيت الاشياء بواسطة فاتح الترميز فيتربي تتم باستخدام طول قيد ٦، مما يوفر أداءً عاليًا في استقبال البيانات، مع أقل فقد ممكن.

إقرار المشرف

أشهد بأن هذه الرسالة الموسومة (**تنفيذ لخوارزمية فيتربي لتطبيقات انترنيت الأشياء باستخدام مصفوفة البوابات المبرمجة حقليا**) والمعدة من قبل الطالبة (**هبة سعد محمود شكري**) تحت اشرافي في قسم هندسة الحاسوب والمعلوماتية/ كلية هندسة الالكترونيات / جامعة الموصل، كجزء من متطلبات نيل شهادة الماجستير علوم في اختصاص هندسة الحاسوب والمعلوماتية.

التوقيع:

الاسم: **أ.م. محمد حازم الجماس**

التاريخ: / /٢٠١٩

إقرار المقوم اللغوي

اشهد بأنه قد تمت مراجعة هذه الرسالة من الناحية اللغوية وتصحيح ماورد فيها من أخطاء لغوية وتعبيرية وبذلك أصبحت الرسالة مؤهلة للمناقشة بقدر تعلق الأمر بسلامة الأسلوب أو صحة التعبير.

التوقيع:

الاسم:

التاريخ: / /٢٠١٩

إقرار رئيس قسم هندسة الحاسوب والمعلوماتية

بناءً على التوصيات المقدمة من قبل المشرف والمقوم اللغوي أرشح هذه الرسالة للمناقشة.

التوقيع:

الاسم: **أ.م. عبد الباري رؤوف سليمان**

التاريخ: / /٢٠١٩

إقرار رئيس لجنة الدراسات العليا

بناءً على التوصيات المقدمة من قبل المشرف والمقوم اللغوي ورئيس قسم هندسة الحاسوب والمعلوماتية أرشح هذه الرسالة للمناقشة.

التوقيع:

الاسم:

التاريخ: / /٢٠١٩

جامعة الموصل

كلية هندسة الالكترونيات

تنفيذ لخوارزمية فيتربي لتطبيقات انترنيت الأشياء باستخدام مصفوفة البوابات المبرمجة حقليا

رسالة تقدم بها

هبة سعد محمود

إلى
مجلس كلية هندسة الالكترونيات
جامعة الموصل
كجزء من متطلبات نيل شهادة الماجستير
في
هندسة الحاسوب والمعلوماتية

بإشراف

الأستاذ المساعد الدكتور

محمد حازم الجماس

١٤٤٠ هـ ٢٠١٩م