IMPLEMENTATION OF BLUETOOTH BASEBAND CONTROLLER BASED ON FPGA DESIGN

TAN WENG HOOI

UNIVERSITI SAINS MALAYSIA

2017

IMPLEMENTATION OF BLUETOOTH BASEBAND CONTROLLER BASED ON FPGA DESIGN

BY

TAN WENG HOOI

Thesis submitted in partial fulfilment of the requirements for the degree of

Bachelor of Engineering (Electronic Engineering)

June 2017

ACKNOWLEDGEMENT

First and foremost, I would like to express my deepest appreciation to my supervisor Encik Zulfiqar Ali Bin Abd. Aziz for his continuous supervision, assistance and support from the initial to the final stage of my final project. This dissertation would not be possible without his patient guidance, enthusiastic encouragement and constructive suggestions throughout the whole period. The valuable lessons learnt not only helped me in enhancing my knowledge, but the interpersonal skill and the positive attitude as well.

Besides, I would also like to thank to all the technicians for the given technical support. They are keen in teaching and assisting me when dealing with the machines and various equipment during my laboratory work. Their careful and precious guidance were extremely valuable for my study both theoretically and practically.

Next I would like to express my sincere gratitude to my family who constantly support and encourage me spiritually and mentally. Massive thanks to my fellow course mates for sharing their knowledge and ideas in the construction of this project.

Last but not least, I wish to express my sincere thanks to all the lecturers involved in my final year project for their precious guidance and opinions which are extremely valuable for my study both theoretically and practically. I perceive this research project as a big milestone in my career development. I will strive to use gained skills and knowledge in the best possible way, and I will continue to work on the improvement, in order to attain desired career objectives.

ABSTRAK

Pelaksanaan pengawal jalur asas Bluetooth adalah satu kajian pemprosesan paket dan proses pemulihan data. Sebelum memindahkan data, maklumat mesej perlu diproses untuk memenuhi standard struktur paket Bluetooth untuk tujuan pengalamatan, pemeriksaan ralat dan keselamatan. Objektif projek ini adalah untuk mereka bentuk pengawal jalur asas Bluetooth yang mampu melaksanakan pemprosesan paket dalam laluan penghantar dan proses pemulihan data dalam laluan penerima. Pemprosesan paket dilakukan pada 32 bit maklumat mesej data untuk memproses kepada 270 bit paket data, manakala proses pemulihan data dilakukan ke atas 270 bit paket data untuk mengembalikan asal 32 bit maklumat mesej yang dikehendaki. Proses semakan ralat dijalankan di pertengahan proses pemulihan data dalam laluan penerima. Sejak laluan data bagi pengawal jalur asas adalah dwi-arah, proses pemulihan data ialah proses terbalik bagi pemprosesan paket. Pelaksanaan pengawal jalur asas Bluetooth mengambil berat tentang pemasangan proses laluan data. Oleh itu, struktur keseluruhan laluan data digambarkan, dengan penjelasan bagi setiap yang terperinci. Laluan data terdiri daripada blok yang berbeza, seperti Header Error Check (HEC) blok, Cyclic Redundancy Check (CRC) blok, Pemutihan blok, Forward Error Check (FEC) blok dan Kod Akses penjana dan penyekait blok. Setiap blok mempunyai peranan dan fungsi yang berbeza dalam pemprosesan paket dan proses pemulihan data. Walau bagainmanapun, beberapa blok telah dicadangkan dengan penggunaan Linear-Feedback Shift Register (LFSR), dengan formula generator polinomial tertentu. Kajian ini melaksanakan pengawal jalur asas Bluetooth berdasarkan reka bentuk Field-Programmable Gate Array (FPGA). Verilog Hardware Description Language (VHDL) pula digunakan dalam reka bentuk.

ABSTRACT

Implementation of Bluetooth baseband controller is a study of packet processing and data restoration process. Before the transferring of data, the message information is required to be processed to fulfill the standard Bluetooth packet structure for addressing, error checking and security purposes. The objective of this project is to design a Bluetooth baseband controller that is able to perform packet processing in transmit path and data restoration process in receive path. Packet processing is performed on 32 bits message information data to process it into 270 bits packet data, while data restoration process is performed on that 270 bits packet data to retrieve back the desired original 32 bits message information. The error checking processes are carried out in the middle of data restoration process in receive path. Since the data path of baseband controller is bidirectional, the data restoration process is the reversal process of packet processing. The implementation of Bluetooth baseband controller particularly concerned on the installation of data path process. Therefore, the overall structure of data path is depicted, with every single detail of explanation. Data path is composed of different blocks, such as Header Error Check (HEC) block, Cyclic Redundancy Check (CRC) block, Whitening block, Forward Error Check (FEC) block and Access Code generator and correlator block. Each block possesses different role and function in packet processing and data restoration process. Even some blocks are proposed with the application of Linear-Feedback Shift Register (LFSR), with specific generator polynomial formula. This research implements the Bluetooth baseband controller based on Field-Programmable Gate Array (FPGA) design. Verilog Hardware Description Language (VHDL) is used in the designation.

LIST OF ABBREVIATION

FPGA	Field-Programmable Gate Array
VHDL	Verilog Hardware Description Language
ASIC	Application Specific Integrated Circuit
SCO	Synchronous Connection Oriented
ACL	Asynchronous Connection-Less
ARQ	Automatic Repeat Request
NAK	Negative Acknowledgement
ACK	Acknowledgement
WPAN	Wireless Personal Area Network
ROM	Read-Only Memory
USB	Universal Serial Bus
UART	Universal Asynchronous Receiver/ Transmitter
HEC	Header Error Check
CRC	Cyclic Redundancy Check
FEC	Forward Error Check

TABLE OF CONTENT

LIST OF FIGURE

LIST OF TABLE

CHAPTER 1

INTRODUCTION

1.1 Background

Bluetooth is an application used for wireless data communication. It has limited coverage range in pairing between two devices (master and slave), since it uses a short-range RF transmission method. However, it is widely used nowadays among the latest technologies (mostly in any android smart phone) due to several leading advantages. Bluetooth costs low power consumption. With the success of the development of Bluetooth Low Energy (BLE) or Bluetooth Smart [1], it brought a massive revolution to the technology of Bluetooth as it becomes even lower power consumption. The evaluation of performance in BLE has been carried on in terms of energy consumption, latency, piconet size and throughput as well [2].

Bluetooth allows total of 8 devices to connect and work together at the same times in a group. Within these 8 devices, there composed of 1 master device and 7 slave devices. The said group is called as a piconet. The operation of the Bluetooth follows strictly to the Bluetooth protocol. As overall, it operates in globally unlicensed Industrial, Scientific and Medical (ISM) band at 2.4 GHz frequency [3, 4]. Frequency hopping transmission scheme is used as well.

Recalling back to its history, the first Bluetooth system came from the idea of Ericsson Mobile (SONY) at year 1994 [5, 6]. The reason for the invention of Bluetooth was that it was originally intended to act as a wireless alternative to RS-232 cables. And the times went on, in 1998, Bluetooth Special Interest Group was formed as the result of

the joint effort by many large companies, including Ericsson, Intel, Toshiba, Nokia and IBM [5, 6]. Since then, this technology was shared out.

Until now, the development of Bluetooth technology is still the main topic. It is practically used in different kind of electronic devices. As if we can say, the Bluetooth speakers, keyboards and headsets are the main stream in the market instead of wired type. Some researches has been done about the comparison of wired and wireless network in all aspect of field [7]. It shows that wireless network takes the advantages in the ease of installation, time for installation, mobility and reliability [7].

One of the important specifications in Bluetooth is that it allows the devices from different manufacturer to work together. Due to this reason, it is also defined as a software stack that allows one's Bluetooth application to search for other Bluetooth devices within the coverage area, and then discover for the usable services.

The performance of Bluetooth is decided according to the multiple layers of Bluetooth protocol stack. Each of the protocol layers plays its own roles and responsibility in the preparation and transmission of message information data. Among the protocol layers, there is a layer named baseband controller layer, which handles all sorts of digital signal processing of data. To implement Bluetooth baseband controller, it is important to get use of the detail of this baseband controller layer on how it functions and why is it necessary to be involved.

1.2 Problem Statement

There are two optional methods to implement the operation of Bluetooth baseband controller, by using FPGA or ASIC design. Both have their own advantages and benefits. ASIC can offer a full custom capability [8], providing a dense design with

lower power consumption. However, to implement Bluetooth baseband controller, ASIC is not a good choice since it occupies large scale of area in ASIC design. In this case, FPGA is software based programmable device which has high flexibility to reprogram in the field, which mean it can uploads a new bit stream remotely. It is easier to use and more cost effective compared to ASIC.

The only problem for FPGA is that it is not fully customable, so it is impossible to add the specific analogue blog or integrate RF capability into FPGA itself. However, this is not really a problem for Bluetooth Baseband Controller. Bluetooth baseband controller processes only the incoming digital message signal. It concerns more on digitally processing of signal, before sending out from transmitter and after receiving by receiver. In other words, full custom capability is not really a must in implementing Bluetooth baseband controller.

Regarding to those consideration, FPGA is more suitable to be used in this project.

1.3 Research Objectives

i. To design a Bluetooth baseband controller using FPGA that is able to simulate the actual function of packet processing and data restoration process (digital signal processing).

ii. To design a Bluetooth baseband controller that is able to check the errors for the incoming packet data in receiver.

iii. To design a Bluetooth baseband controller that is able to retrieve back the desired original message information in receive path.

1.4 Scope of Research

The scope of research is limited to the architectures of hardware implementation of Bluetooth baseband controller. It is done with FPGA using Verilog Hardware Description Language (VHDL). It is said that Bluetooth functions following Bluetooth protocol where Bluetooth protocol is defined as a series of layers [9]. Throughout the Bluetooth protocol's layer, there are 4 basic layers that are categorised as core protocol in Bluetooth. They are logical link control, link manager, baseband and Service Discovery Protocol (SDP). The link manager takes care of managing states and packets and controls flow on the link [4]. The logical link control takes care of multiplexing the user protocols [4, 9]. Baseband takes care of the timing and framing of the packet, flow control error detection and error correction [4]. In this project, the focus is only on baseband protocol layer (baseband controller) will be focused on as well. Inside baseband controller, it is composed of many main blocks, such as clock generator, frequency hop selector, access code generator, centralised controller, buffer, data path and RF interface. Frequency hopping and RF interfacing is not the part of concern, which mean all the parts concerned are based on digitally signal processing. Modulation to analogue signal is not required. Data path is the part that put the most effort in implementation.

1.5 Thesis Outline

The thesis is organized into five chapters. In chapter 2, the structure of Bluetooth protocol stack is explained. Among the protocol layers, the baseband controller layer is focused. The standard Bluetooth packet structure is described as well, with the detail of

the structure of access code, header and payload. Other than that, the hardware design of baseband controller and the comparison between Bluetooth and WIFI are reviewed.

Chapter 3 describes the design of baseband controller to be implemented after reviewing related works in chapter 2. The implementation processes are explained in detail for each step and operation. The hardware architecture is also presented in diagram and formulas to support the explanation.

In chapter 4, are the results and discussions of the research that had been done. Simulation of the implemented Bluetooth baseband controller as stated in chapter 3 is shown. The results are verified to confirm the availability in retrieving back the original message information. Some calculations are made to verify the formation of result signals in the simulation.

Chapter 5 concludes the research by summarizing the achieved research objectives. Future work is also suggested to improve the technology of Bluetooth at the end of the project.

CHAPTER 2

LITERATURE REVIEW

2.1 Bluetooth Protocol Stack

Bluetooth protocol stack was developed by the Bluetooth Special Interest Group. These specifications allow for developing interactive services and applications over interoperable radio modules and data communication protocols [10]. These specification set down the protocols that must be followed by different companies when they manufacture and develop both software and hardware [11]. This is to make sure the devices that are manufactured by different companies can work with each other even though they run on different protocol stack. As long as they have one imperative factor, this allows them to be interoperable [11].

The Bluetooth protocol stack is defined as a series of layers [9]. Figure 2.1 shows the Bluetooth protocol stack model. Not all the application makes use of all the protocols shown in Bluetooth protocol stack. An application may run only one or more vertical slices from the protocol stack [10]. Which mean for an example, It may run in a sequence like "Application ➔ WAP ➔ RFCOMM ➔ L2CAP ➔ Link Manager ➔ Baseband Controller ➔ Radio", this sequence considers as a run of vertical slices from protocol stack. Step by step flowing of sequence accordingly to the layers of Bluetooth protocol stack is necessary.

In designing the whole protocol stack, its main principle is to maximise the re-using of existing protocols for different purposes at higher layer instead of re-inventing

it [11]. This helps the existing applications to adapt to the Bluetooth technology. The higher layers of the protocol stack are implemented by the hosts [9] , such as our PCs. To link the higher (host) and lower (Bluetooth module) layers together, Host Controller Interface (HCI) is playing the role as an interface between both layers [9].

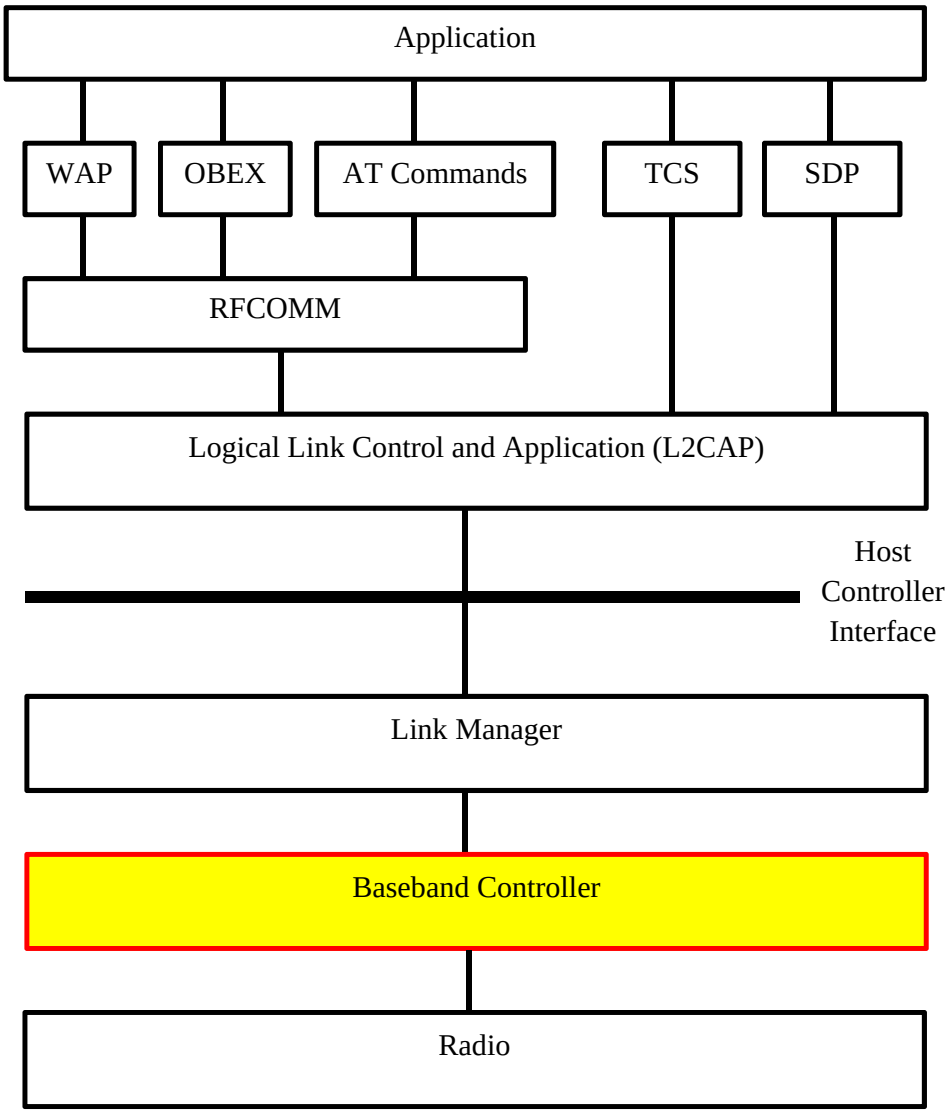


Figure 2.1 Bluetooth Protocol Stack Model

2.1.1 Baseband Controller

Baseband allows the physical connection between devices [3]. It concerned with connection establishment within piconet, addressing, packet format, timing and power control [12]. Baseband Control layer provides two basic types of physical links that can be established between a master device and a slave device [9]. They are Synchronous Connection Oriented (SCO) and Asynchronous Connection-Less (ACL). The comparison between SCO and ACL are shown in Table 2.1.

Table 2.1 Comparison of SCO and ACL [9, 11]

Synchronous Connection Oriented (SCO)	Asynchronous Connection-Less (ACL)
Provides a symmetric link between a master and a slave.	Provides a point-to-multipoint link between a master and all slaves.
Provides a circuit-switched connection.	Provides a packet-switched connection.
Regular periodic exchange of data in the form of reserved slots.	Sporicidal exchange of data in the remaining slots on the channel not used for SCO links.
Used for data only	Can be used for the combination of audio and data.

2.1.1.1 Bluetooth Packet

It is the baseband controller who transforms the incoming message signal into the Bluetooth packet structure as shown in Figure 2.2. The packet structure is composed of 3 sections as well. They are access code, header and payload. Bluetooth packet can

be considered as a complete processed signal in digital form that is ready to be modulated into analogue form afterwards.

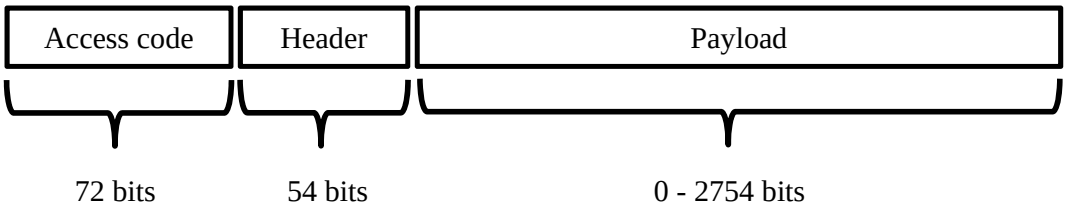


Figure 2.2 Bluetooth Packet Structure

The access code is used to address the packet to specific device [9]. The header contains all the information associated with the packet and the link [9]. The payload contains the actual message information [9]. The message information might be random number of bits, but 32 bits (4 bytes) data is commonly used in the architecture of computer.

2.1.1.2 Access Code

Each Bluetooth device has 48 bits IEEE MAC address that is used for the derivation of the access code [9]. As shown in Figure 2.2, access code has total 72 bits. It has the pseudorandom properties and declares the identity of the piconet master [9]. Within the 72 bits access code in Bluetooth packet, it is composed of 4 bits preamble, 64 bits of sync word and last 4 bits of trailer. Figure 2.3 shows the 72 bits access code structure. Preamble depends on the first bit (most significant bit) of the sync word, either in the form of 0101(0) or 1010(1) [13]. Reversely, trailer depends on the last bit (least significant bit) of the sync word, either in the form of (1)0101 or (0)1010 [13].

Sync word may be different for different devices. It is used to match with the match pattern in the correlator when the packet is received by the receiver. After when the match pattern in correlator is matching with the sync word in the incoming packet, the incoming packet is accepted by the receiver and starts the process to recover back the desired message signal.

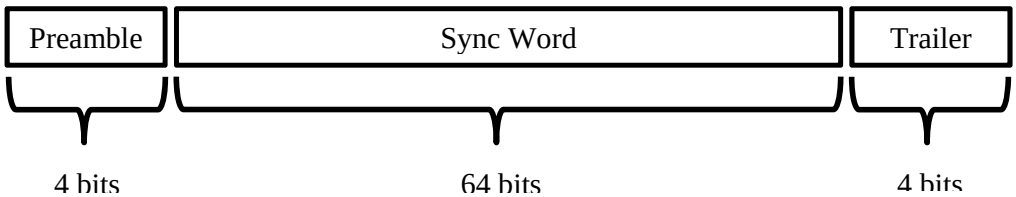
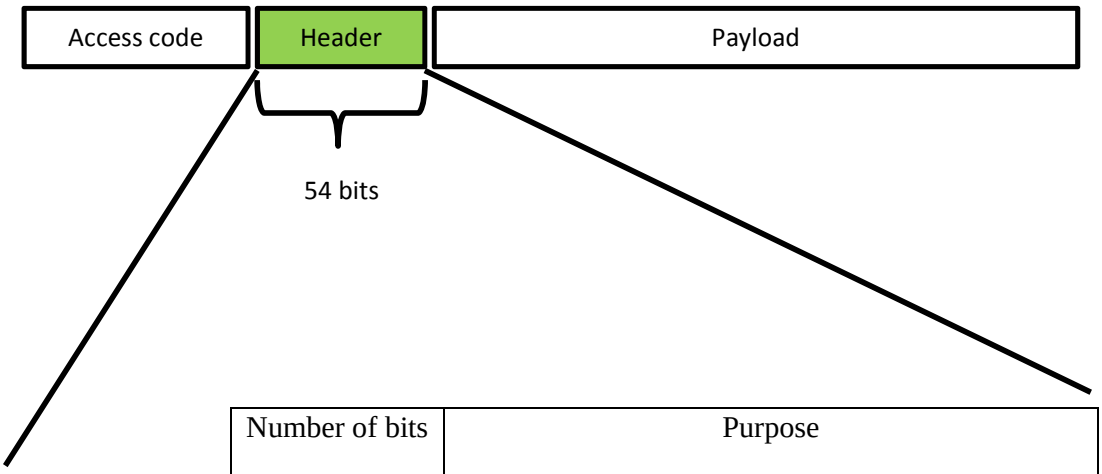


Figure 2.3 Access Code Structure

2.1.1.3 Header

Bluetooth packet with 54 bits header, but actually it consists only 18 bits [13]. It becomes 54 bits header due to the duplication of 18 bits header 3 times through the $\frac{1}{3}$ Forward Error Check (FEC) encoding process. And inside the 18 bits header, 3 bits consist of addressing, 4 bits for determining packet types, 1 bit flow control, 1 bit Automatic Repeat Request (ARQ), 1 bit sequencing and last 8 bits Header Error Check (HEC). Figure 2.4 lists down the compositions of the 18 bits header and describes clearly their own role.



	Number of bits	Purpose
Addressing	3	Active member address, the address of the active slave to which the packet is directed or from which it is sent. (Maximum 7 slaves available)
Packet types	4	16 packet types
Flow control	1	Flow control over the Asynchronous Connection-Less (ACL) link: 0 : stop 1: go
ARQ	1	Acknowledgement indication: 0 : Negative Acknowledgement (NAK) 1 : Acknowledgement (ACK)
Sequencing	1	Sequence number, this bit is triggered for consecutive packets.
HEC	8	Header Error Check, an integrity check value.
	18	

Figure 2.4 Composition of Header in Bluetooth Packet [13]

2.1.1.4 Payload

Payload contains the desired actual message information [9]. Payload does not have a fixed number of bits. As shown in Figure 2.2, its bits number can be any number in the range from 0 to 2754. It depends on what device and what type of message information it is. However, 32 bits (4 bytes) data information is commonly being used in the architecture of computer. A Cyclic Redundancy Check (CRC) is approached behind the message information. So, the payload is actually constructed by the message information with unknown bits number and 16 bits CRC, followed by either $\frac{1}{3}$ Forward Error Check (FEC) encoding or $\frac{2}{3}$ Forward Error Check (FEC) encoding. $\frac{1}{3}$ FEC encoding is the duplication of 1 bit information into 3 same bits, just like what it is done to the header part. In $\frac{2}{3}$ FEC encoding, each sequence of 10 bits is followed by 5 parity bits calculated using a (15, 10) shortened Hamming code. The packet payload must be padded out with trailing zeroes to make it the multiple of 10 bits in length so that it can carry out $\frac{2}{3}$ FEC encoding.

2.1.2 The Bluetooth Network Topology

Communications between Bluetooth devices are normally peer to peer with each device being equal [14]. In the Bluetooth network architecture, the term “piconet” is used to bring up the meaning two or more (maximum 8) devices link into a small ad hoc network [14, 15]. A piconet is a Wireless Personal Area Network (WPAN) [14] which consists of only a single master device and one or more slave devices [6]. It is possible to include more devices in a piconet by placing one or more slaves into what it is referred to as common mode [16]. In command to interchange information with the

allocated slave, master must gross it out of allocated mode and return it to energetic mode [16]. Merely seven slaves can be in lively mode at any given times [16].

A device can either be a master or slave. But not only that, it may at the same time be the slave to different master devices or at the same time be the master to other slaves and slave to other master device. In other words, a device may additionally belong to one or more piconet [6]. This condition is when two piconet are in nearby contiguity, when they are overlapping the coverage areas [16]. One device in each piconet acts as a bridge between two or more piconet forming a “scatternet” [15]. Figure 2.5 illustrates the piconet and scatternet.

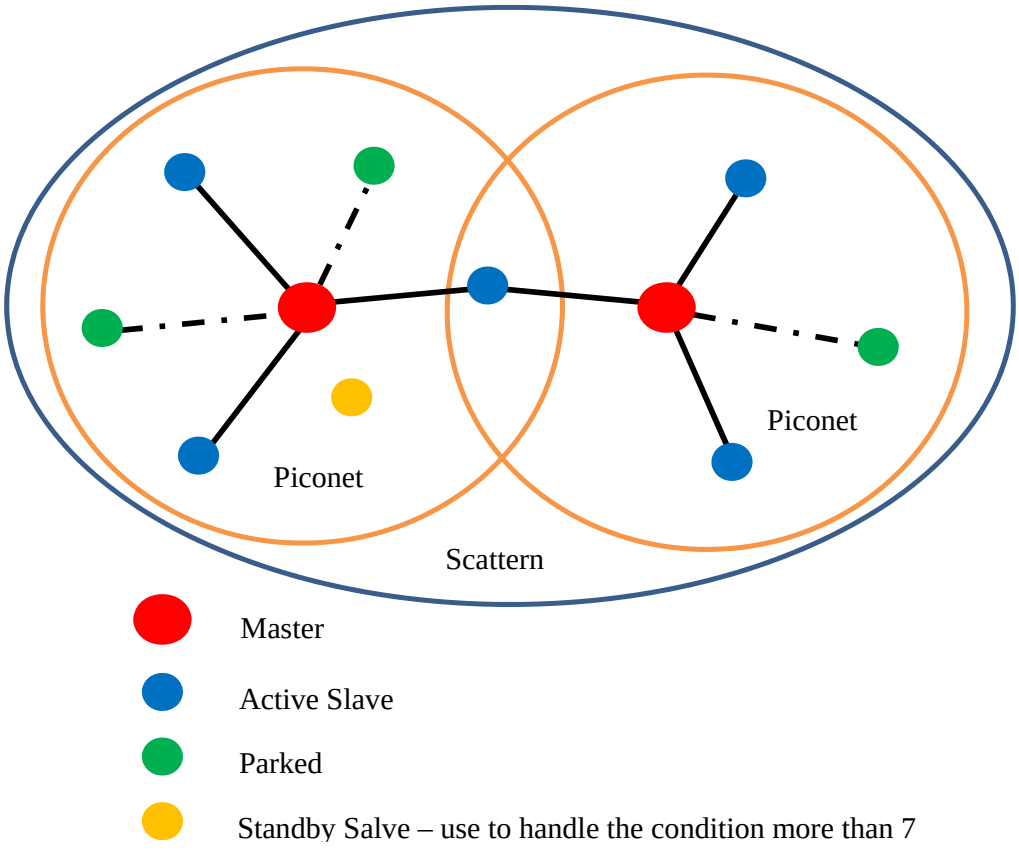


Figure 2.5 Bluetooth Piconet and Scatternet Diagram

2.2 Design of Bluetooth Baseband Controller

Bluetooth baseband controller is commonly designed in Verilog Hardware Description Language (VHDL) and is implemented into a Field Programmable Gate Array (FPGA) [4, 17]. Bluetooth baseband controller handles the baseband link layer that is mentioned in the Bluetooth protocol stack. The baseband link layer performs the functions related to interface interaction between the Bluetooth chipset and an external or integrated radio chip and a host system [17]. In baseband, different modules for data processing are integrated and the development of the modules has been done with VHDL [17].

2.2.1 Hardware Design of Baseband Controller

The designed baseband controller consists of a register file, a controller, a data path, a modem, a hop selector, a clock generator and interface block [4]. Figure 2.6 shows the block diagram of the designed Bluetooth baseband controller, while Table 2.2 describes the role for each part inside the baseband controller.

Table 2.2 Baseband Controller's Parts and Their Own Role [4]

Register File	Exchange information between link manager and baseband
Controller	Takes care of timing and state changing
Data Path	Composes the packet to be transmitted and decomposes the received packet
Modem	Takes care of smoothing the packet in the transmit path and recovering the data and the clock in the receive path
Hop Selector	Selects the hop frequency for packet transmission
Clock Generator	Supplies clock signal to all the other blocks

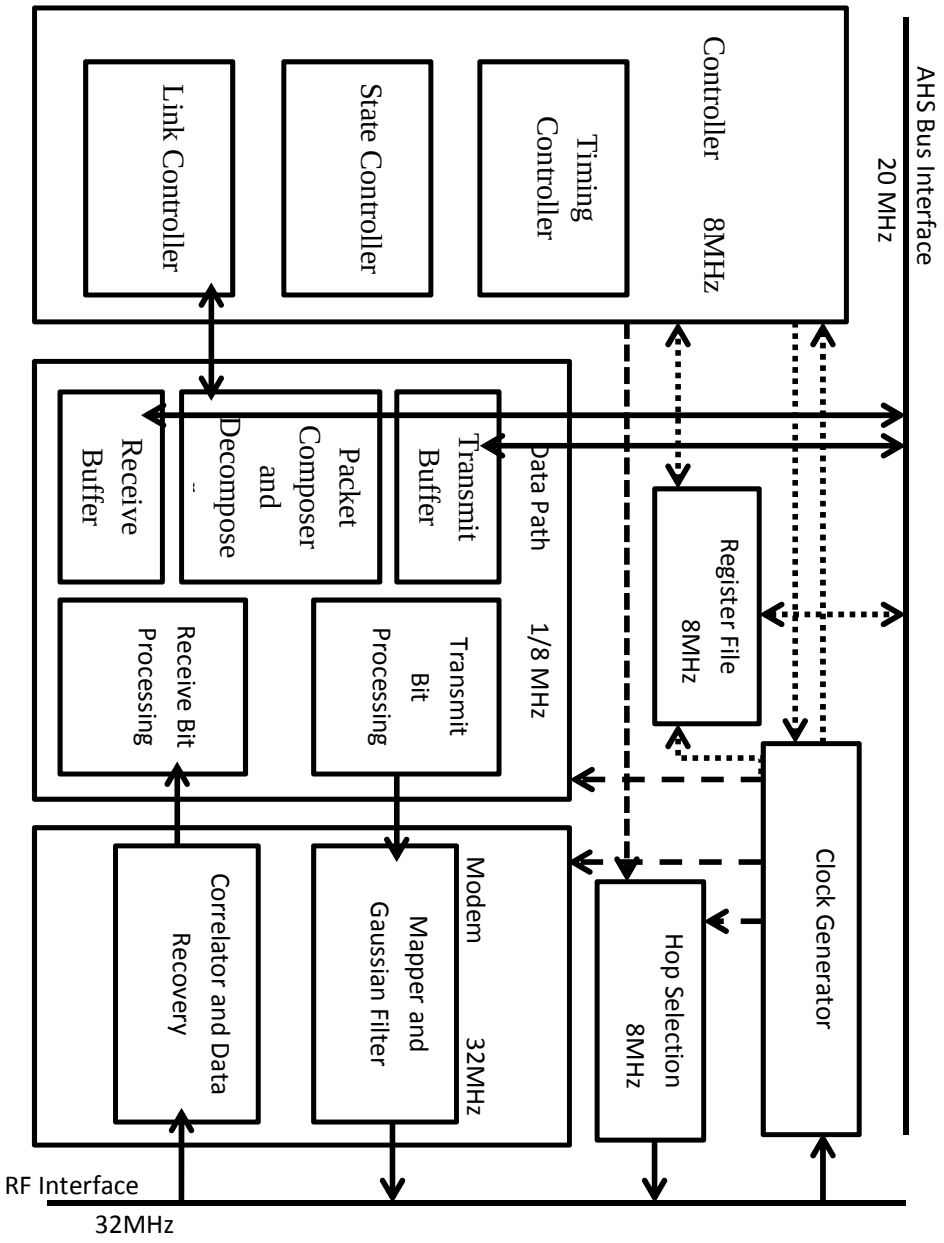


Figure 2.6 Block Diagram of Designed Baseband Controller [4]

2.2.1.1 Data Path

The structure of Bluetooth packet is illustrated in Figure 2.2. As mentioned previously, data path takes part in composing the packet to be transmitted and

decomposing the received packet. Figure 2.7 shows exactly the full digital process to transform the actual message information into packet structure.

In transmit path, the packet composer adds the Header Error Check (HEC) to the header information read from the register file and the link control, and then scrambles the header with whitening word and encodes it at a rate of $\frac{1}{3}$ Forward Error Check (FEC) [4]. The payload information plus the Cyclic Redundancy Check (CRC) is scrambled, encrypted, and encoded at a rate of $\frac{2}{3}$ FEC [4].

In receive path, the packet decomposer extracts the header and the payload information from received packet in reverse order from the packet composer [4].

2.3 Bluetooth versus WIFI

Bluetooth and WIFI have always been compared in the term of costs and power consumption. Bluetooth is intended for portable products, short ranges and limited battery power [18]. That is why Bluetooth is designed to have very low power consumption to cause less damage to the battery life. On the other hand, WIFI is designed for long range connections and can only support devices with a sustainable power supply [18]. This makes Bluetooth to be a better option for home use device. If the greater range is the must to be considered, then WIFI should be the first choice.

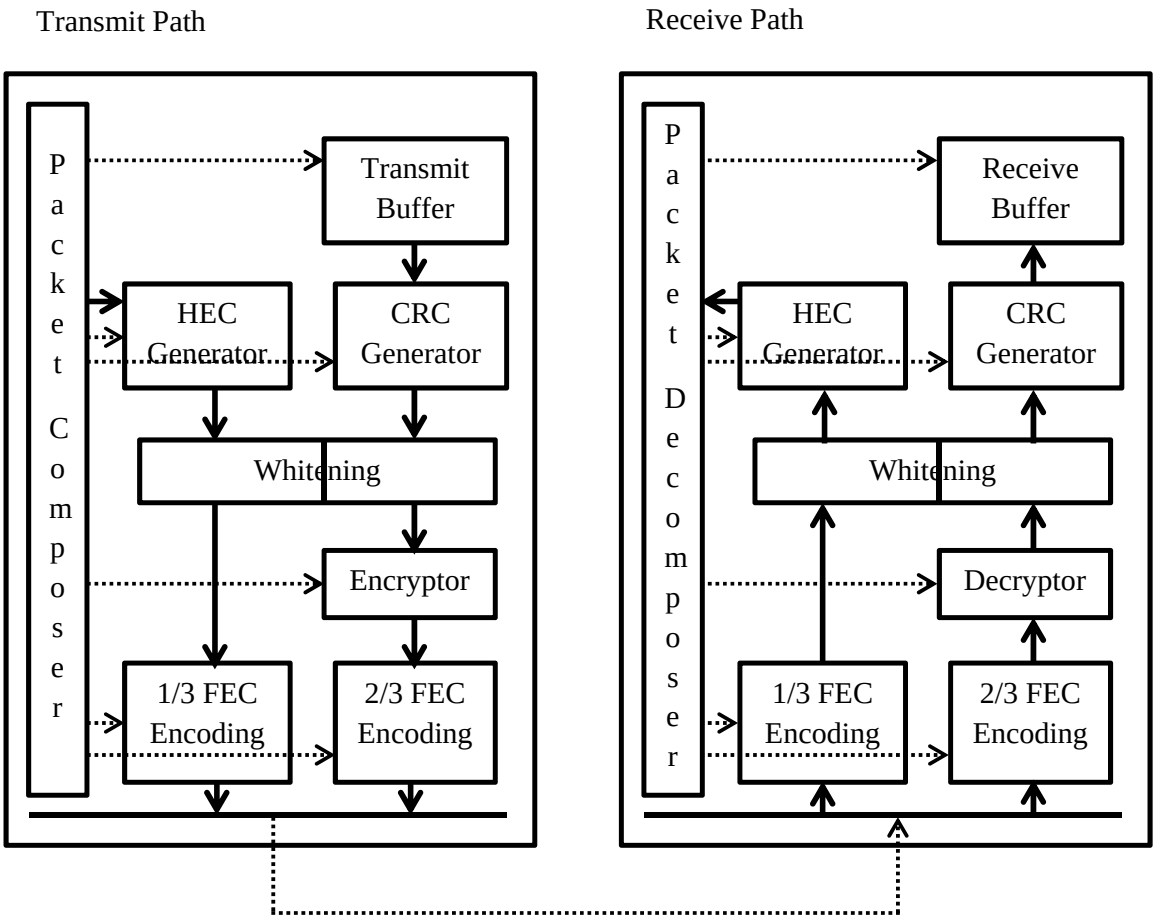


Figure 2.7 Block Diagram of Data Path [4]

2.4 Summary of Chapter

In chapter 2 literature review, the Bluetooth protocol stack is the rules that must be followed strictly. Well, only baseband protocol layer among whole protocol stack is implemented. In the baseband layer, the Bluetooth packet structure is fixed in its standard form, as in Figure 2.2. Baseband controller is built to handle baseband layer and it is designed using FPGA in VHDL. The past article, journal and research done for hardware design of Bluetooth baseband controller has been referred in literature review. In the design of baseband controller, data path is the block being concerned on since it is the main block in performing digital signal processing.

CHAPTER 3

METHODOLOGY

3.1 Bluetooth Controller

Bluetooth hardware consists of microcontroller, flash or Read-Only Memory (ROM), baseband controller and Universal Serial Bus (USB) or Universal Asynchronous Receiver/ Transmitter (UART). USB or UART is used as an interface to connect to the Bluetooth host. Figure 3.1 shows the block diagram of the digital hardware design for Bluetooth controller.

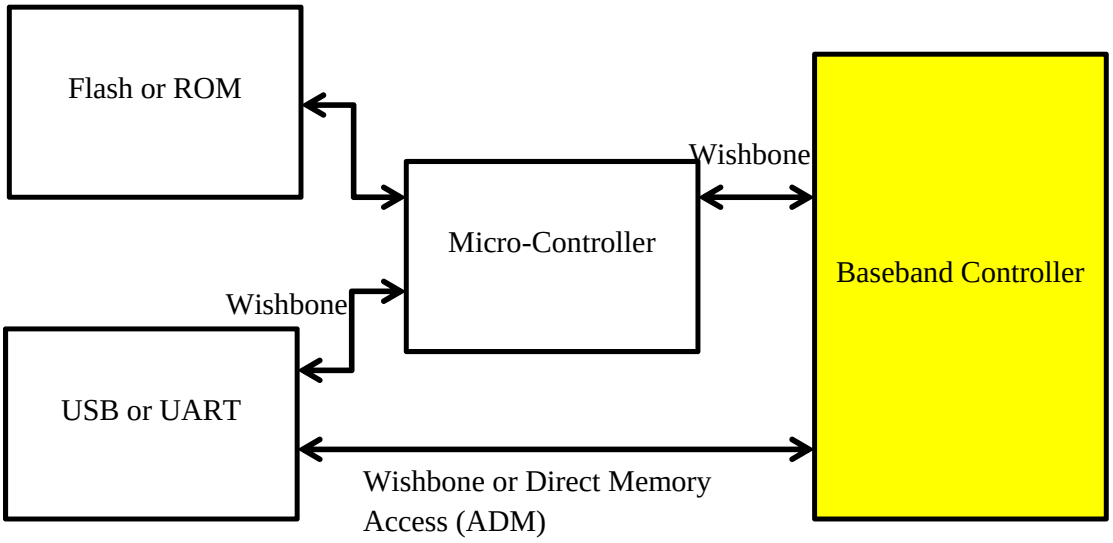


Figure 3.1 Block Diagram of Digital Hardware Design for Bluetooth Controller

***Only baseband controller's part is studied in this project.

3.1.1 Bluetooth Baseband Controller

Baseband controller handles the baseband protocol layer in Bluetooth protocol stack. It takes care of timing and framing of the packet, flow control error detection and error correction [4]. There are 5 main blocks to make up a baseband controller:

i) External interfaces – divided into Host (CPU) interface using Wishbone bus, Direct Memory Access (DMA) engine, Radio Frequency (RF) interface, Pulse-Code Modulation (PCM) interface and software interfaces (registers).

ii) Data buffers for both transmit and receive – buffer with 8 bits width is used.

iii) Data path – carries on bit serial process to both transmitted and received data. The internal blocks perform the operation on Bluetooth bit rate (1 Mbps) using the 1 MHz clock cycle. Besides, these blocks are shared between transmit and receive direction since only one direction is active at a time.

iv) Main centralised controller – controls all internal blocks and perform time scheduling for data path blocks. It performs the link control and Bluetooth states. It uses the Clock (CLK)-I to speed up the link management operation.

v) Clock management, hop frequency calculation, access code generator and miscellaneous blocks.

Figure 3.2 is the structure inside the Bluetooth baseband controller with all its main blocks inserted. It shows how exactly each block is interlinking with each other and the highlighted data path in Figure 3.2 is the block to be concerned on. It handles most of the bit stream process that are going to implement using FPGA.

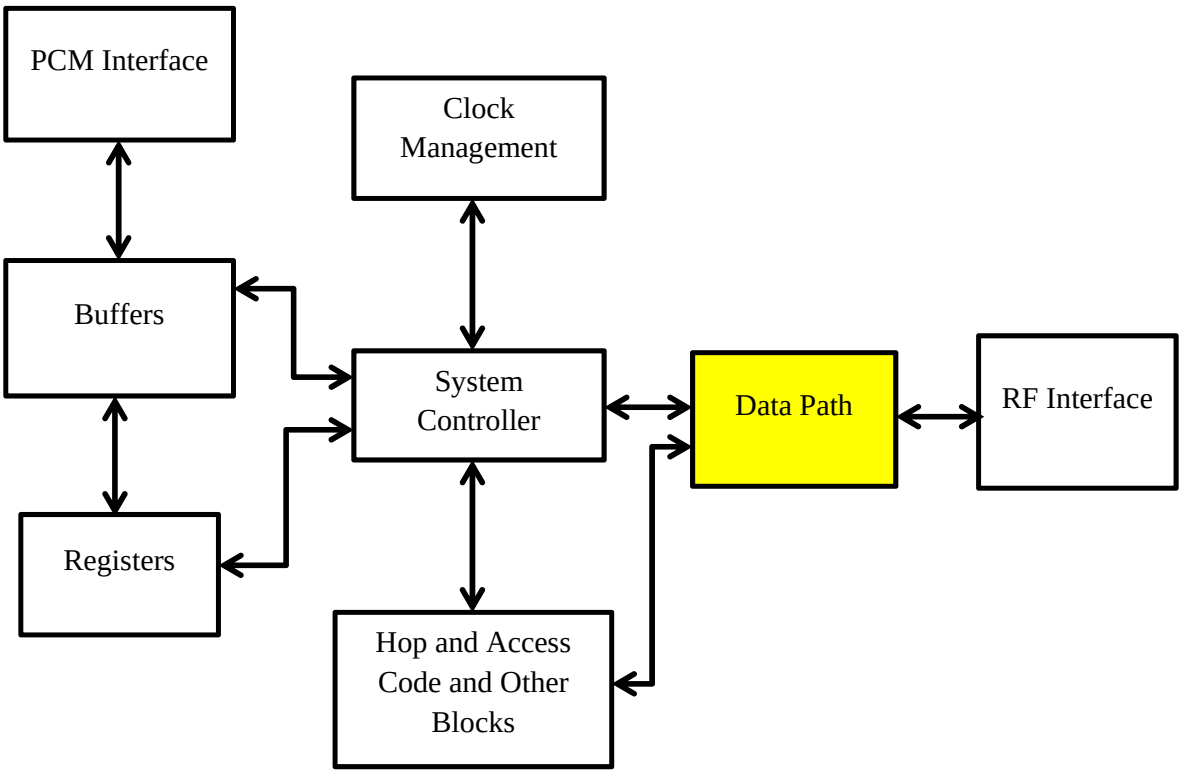


Figure 3.2 Block Diagram of Bluetooth Baseband Controller

3.1.1.1 Data Path

Data path processes data bits serially using 1 MHz clock which is controlled by system clock. 1 MHz clock acts as a switch to enable for data processing. Serial data process will provide a low data rate and reduce the hardware resources. Theoretically, serial process is more suitable to be applied.

Data path is composed of the following blocks:

a) Parallel to serial and serial to parallel converter, which convert the packets only that are passed to or from the system controller for parsing and routing.

b) Cyclic Redundancy Check (CRC) generator and checker of the payload.

c) Header Error Check (HEC) generator and checker.

d) Whitening block for scrambling and descrambling the payload and header.

e) Forward Error Check (FEC) encoder, either $\frac{1}{3}$ FEC or $\frac{2}{3}$ FEC encoding. $\frac{1}{3}$ FEC encoding to header and $\frac{1}{3}$ or $\frac{2}{3}$ FEC encoding to payload. ($\frac{1}{3}$ FEC is chosen in simulation)

f) Access code inserter before transmit.

g) Access code correlator to check the correctness of identity of the received packet. If match, it triggers the start of packet.

h) Flow control block to read the flow control bits and packet types from the header of the received packets, and the result of CRC to decide the flow control bits of the transmitted packets. It also signals the system controller about this result to determine the next packet decides upon the next flow control.

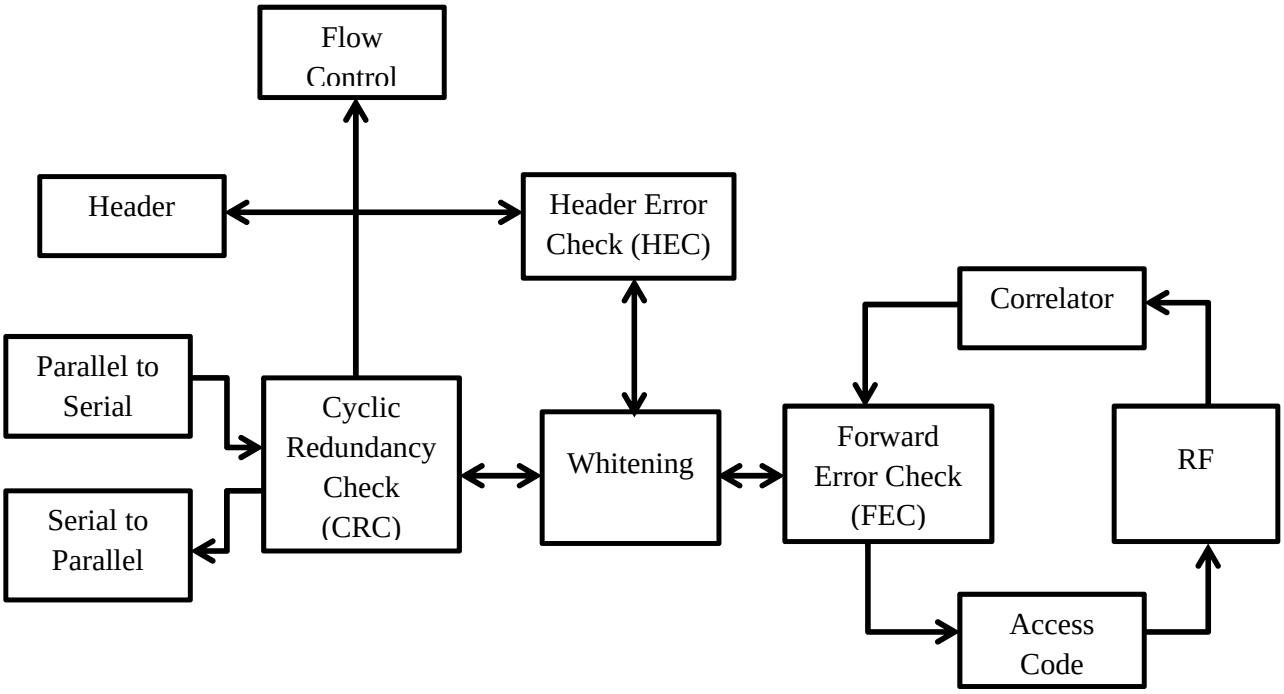


Figure 3.3 Block Diagram of Data Path

3.1.1.2 Header Error Check (HEC)

HEC block is composed of two sub-blocks, one is HEC generator for transmit path and another is HEC checker for receive path. In receiver, it performs error detection and correction on the packet's header. It holds 8 bits HEC of the header in the HEC core, where HEC core is a Linear-Feedback Shift Register (LFSR) which has a generator polynomial of

$$g(D) = D8 + D7 + D5 + D2 + D1 + 1.$$

The structure of LFSR of HEC core is illustrated in Figure 3.4. B0 until B7 in Figure 3.4 are the 8 bits HEC holders. It can be performed on all types of packets. HEC of header is generated after 10 bits header data enters the HEC core, To perform error checking, 18 bits packet's header (10 bits header data + 8 bits generated HEC) enters the HEC core, as shown in Figure 3.4.

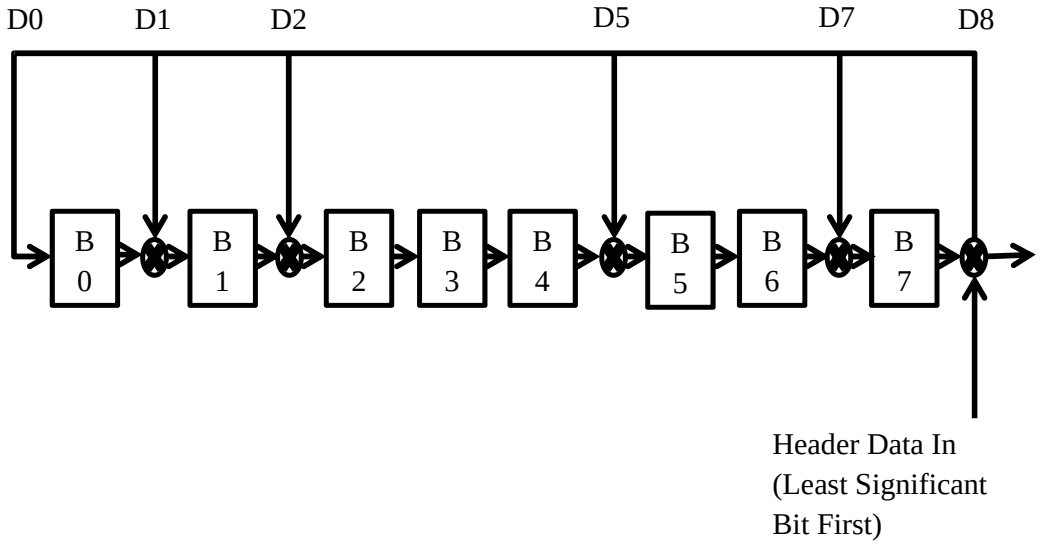


Figure 3.4 HEC core

3.1.1.3 Cyclic Redundancy Check (CRC)

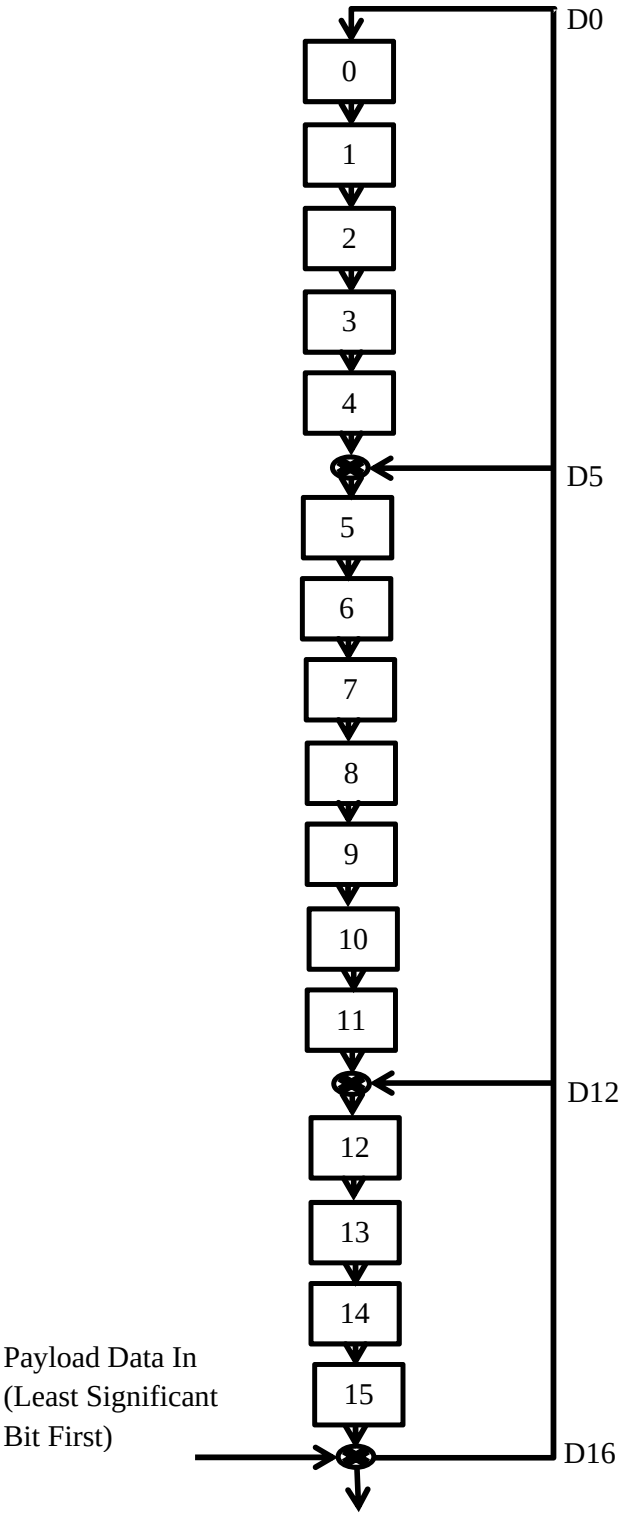


Figure 3.5 CRC Core

CRC block is just similar to HEC block, which also composed by two sub-blocks, CRC generator and CRC checker. It also performs error detection and correction, but on the packet's payload. 16 bits CRC of payload is held in the CRC core. CRC core is a 16-bit LFSR with generator polynomial of

$$g(D) = D16 + D12 + D5 + 1.$$

Referring to Figure 3.5, 0 until 15 small blocks indicate the 16 bits CRC holder. Unlike HEC, CRC is not performed on all types of packets. It only performs on Frequency Hop Synchronization (FHS) packet. CRC of payload is generated after 32 bits message information enters the CRC core, as shown in Figure 3.5 (least significant bit will be entered first). Similarly, error checking is carried out in the same way when 48 bits packet's payload enters the CRC core.

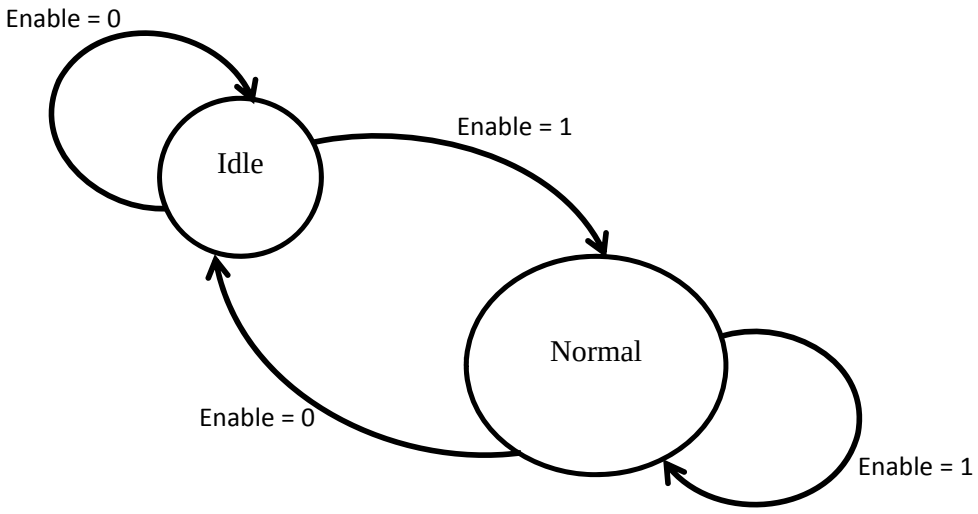


Figure 3.6 State Diagram Operation of CRC

3.1.1.4 Whitening

The purpose to undergo whitening process is to randomise the data signal from high redundant patterns in order to prevent the DC bias when it is transmitted in the