

بسم الله الرحمن الرحيم



## **The Effects of Time Triggered Ethernet on the Network**

دراسة تأثيرات المحاور الاوتوماتيكية المسبب للأترنت على الشبكة

**By**

**Abdallah Nawaf Alkhasawneh**

**Supervisor**

**Prof. Dr. Alaa Al Hamami**

**A Thesis Submitted in Partial Fulfillment of the Requirement**

**for the**

**Degree of Master in Computer Science**

**Department of Computer Science**

**College of Computer Sciences and Informatics**

**Amman Arab University**

**Aug 2010**

**Page Left Blank Intentionally**

## تفويض

أنا عبدالله نواف محمد الخصاونة أفوض جامعة عمان العربية للدراسات العليا بتزويد نسخ من رسالتي الماجستير بعنوان " دراسة تأثيرات المحاور الاوتوماتيكية المسبب للأترنت على الشبكة" للمكتبات أو المؤسسات أو الهيئات أو الاشخاص المخولين عند طلبها.

الاسم : عبدالله نواف محمد الخصاونة

التوقيع: 

التاريخ: ٠٤ ايلول ٢٠١٠

This dissertation titled "Study of effects of time triggered Ethernet on network" has been defended and approved on 07/08/2010 .

| <u>Examining committee</u> | <u>Title</u>          | <u>Signature</u> |
|----------------------------|-----------------------|------------------|
| Dr. Mosbah Jomaa Aqel      | Chair                 | M. J. A. Q.      |
| Prof. Dr. Alaa Al-Hamami   | Member and supervisor | Alahamami        |
| Dr. Mizher shbaan alani    | Member                | M. S. A.         |


## Declaration

I Abdallah Alkhasawneh, hereby declare that this work entitled The Effects of Time Triggered Ethernet on the Network was independently carried out by me under the guidance and supervision of Dr. Ala'a Al Hamami, Amman.

This work is submitted to Amman Arab University for Graduate Studies in partial fulfillment of the requirements for the award of the Master's Degree in computer science during the academic year 2009 - 2010.

Place: Amman  
Date: August 2010

Abdallah Alkhasawneh



## List of abbreviations

| Suffix   | Prefix                                  |
|----------|---|
| ACL      | Access Control List                     |
| AFDX     | Avionic Full Duplex                     |
| ARINC    | Aeronautical Radio INCorporation        |
| A ARCNET | Attached Resource Computer NETwork      |
| ANEG     | AutoNEGotiation                         |
| ARP      | Address Resolution Protocol             |
| ASIC     | Application-Specific Integrated Circuit |
| CAN      | Controller Area Network                 |
| CDMA     | Code Division Multiple Access           |
| CN       | Controlled Node                         |
| C COTS   | Commercial On The Shelf                 |
| CPLD     | Complex Programmable Logic Device       |
| CRC      | Cyclic Redundancy Check                 |
| CSMA     | Carrier Sense Multiple Access           |
| CVS      | Computer System Validation              |
| DLL      | Data Link Layer                         |
| D DTE    | Data Terminal Equipment                 |
| DUT      | Device Under Test                       |
| EDA      | Electronic Design Automation            |
| E EPL    | Ethernet Power Link                     |
| ET       | Event Trigger                           |

|   |      |   |
|---|------|---|
| F | FCC  | Federal Communication Commission                  |
|   | FCS  | Frame Check Sequence                              |
|   | FDDI | Fiber Distributed Data Interface                  |
|   | FPGA | Field Programmable Gate Array                     |
| G | GPS  | Global Positioning System                         |
|   | GPL  | General Public License                            |
| H | HDL  | Hardware Description Language                     |
| I | ICMP | Internet Control Message Protocol                 |
|   | IE   | Industrial Ethernet                               |
|   | IEC  | International Electro Technical Commission        |
|   | IEEE | Institute of Electrical and Electronics Engineers |
|   | IFG  | Inter Frame Gap                                   |
|   | ISO  | International Standard Organization               |
| L | LCD  | Liquid Crystal Display                            |
|   | LLC  | Logical Link Control                              |
|   | LSB  | Least Significant Bit                             |
|   | LUT  | Look Up Table                                     |
| M | MAC  | Media Access Control                              |
|   | MAU  | Medium Attachment Unit                            |
|   | MIG  | Minimum Interface Gap                             |
|   | MII  | Media Independent Interface                       |
|   | MLT  | Multi-Level Transmit                              |
|   | MN   | Managing Node                                     |

|   |      |                                     |
|---|------|-------------------------------------|
|   | MTBF | Mean Time Between Failure           |
| N | NRZ  | Non Return Zero                     |
|   | NRZI | Non Return Zero Inverse             |
|   | NTP  | Network Time Protocol               |
| O | OSI  | Open System Interconnection         |
| P | PAM  | Pulse Amplitude Modulation          |
|   | PDU  | Protocol Data Unit                  |
|   | PHY  | PHYSical layer                      |
|   | PoE  | Power over Ethernet                 |
|   | PTZ  | Pan Tilt and Zoom                   |
|   | PWM  | Pulse Width Modulation              |
| Q | QoS  | Quality of Service                  |
|   | RARP | Reverse Address Resolution Protocol |
|   | RJ   | Registered Jack                     |
|   | RT   | Real Time                           |
|   | RTE  | Real Time Ethernet                  |
|   | RTL  | Register Transfer Level             |
| S | SFD  | Start Frame Delimiter               |
|   | SNMP | Simple Network Management Protocol  |
|   | SVF  | Simple Vector Format                |
| T | TCP  | Transmission Control Protocol       |
|   | TDMA | Time Division Multiple Access       |
|   | TPDU | Transport Protocol Data Unit        |



|   |       |                                    |
|---|-------|------------------------------------|
|   | TTE   | Time Triggered Ethernet            |
|   | UDP   | User Datagram Protocol             |
| U | USB   | Universal Serial Bus               |
|   | UTC   | Coordinate Universal Time          |
|   | VHDL  | Verilog HDL                        |
| V | VHSIC | Very High Speed Integrated Circuit |
|   | VL    | Virtual Link                       |

## Abstract

The transmission speed in twisted pair wiring is generally expressed as being from 0.59 to 0.65 times the propagation speed of light. Therefore, the time delay  $\tau$  will be between 5.13 and 5.65 nanoseconds per meter of distance.

Since the traffic in industrial time triggered Ethernet network is usually time-constrained therefore the Ethernet is not able to provide real-time services. Significant features of the industrial real time Ethernet include the following:

1. Real-time support for cyclic traffic with predefined cycle duration is as short as 2 ms.
2. Worst-case latency down to 2 ms (can be shorter than the cycle duration).
3. Dynamic configuration/set-up of real-time channels with guaranteed performance (Quality of Service (QoS)). Static configuration at system design/start-up is also possible.
4. Utilized clock synchronization used in applications and/or operating systems.

This thesis describes in detail the effects of one of the Real time Ethernet technology, which is the Time Triggered Ethernet (TTE) on the network . This developed protocol can be adopted as a real time solution for critical systems. This Thesis is considered as a perspective for future development.

## **Acknowledgements**

With immense pleasure, I acknowledge the help, guidance and support received from various quarters during the course of my thesis work. Hence, I hereby express my gratitude to all.

My sincere thanks go to my thesis guide and mentor, Dr. Alaa Hamami for his support and guidance. His vision and suggestions throughout the thesis period from choosing, solving and presenting the problem has been fundamental in completing my study.

I have to thank Miss Ghadeer Al Shehan with whom I spent some of the most cherished moments of my study. In addition, my thanks go to those who helped me during my registration periods.

I would like also to thank my beloved parents and my wife for their love and affection. My family members have always been a source of my constant support, encouragement and inspiration. Thanking them would not be sufficient. Therefore, I would like to dedicate this thesis to my parents and my beloved wife.

**Abdallah Alkhasawneh.**

## Table of Contents

|   |             |
|---|-------------|
| <b>Abstract</b> .....                                       | <b>viii</b> |
| <b>Acknowledgements</b> .....                               | <b>ix</b>   |
| <b>Table of Contents</b> .....                              | <b>x</b>    |
| <b>List of figures</b> .....                                | <b>xii</b>  |
| <b>List of tables</b> .....                                 | <b>xiii</b> |
| <b>Chapter one Introduction</b> .....                       | <b>1</b>    |
| 1.1 Thesis Introduction .....                               | 1           |
| 1.2 Objectives and constraints .....                        | 4           |
| 1.3 Methodology .....                                       | 5           |
| 1.4 Literature review .....                                 | 6           |
| 1.5. Statement of the problem .....                         | 11          |
| 1.6 Thesis Contribution.....                                | 11          |
| 1.7 Thesis validation and verification tools.....           | 12          |
| 1.7.1.Ethernet Hub. ....                                    | 12          |
| 1.7.2. Field Programmable Gate Array (FPGA) component. .... | 13          |
| 1.7.3.Verilog Hardware description language VHDL.....       | 14          |
| 1.8.Thesis organization .....                               | 16          |
| <b>Chapter Two Ethernet and Real Time Networks</b> .....    | <b>18</b>   |
| 2.1.Industrial Real time networks .....                     | 18          |
| 2.2. Real time Ethernet network topology overview .....     | 22          |
| 2.3. Ethernet standards .....                               | 23          |
| 2.4 Ethernet Physical layer.....                            | 27          |
| 2.5 Time triggered Ethernet.....                            | 39          |
| <b>Chapter Three Theoretical Design</b> .....               | <b>45</b>   |
| 3.1 Time Triggered Ethernet theoretical performance study   | 45          |
| 3.2 Time Triggered Ethernet practical performance study ... | 49          |
| <b>Chapter Four Simulation and discussion</b> .....         | <b>56</b>   |
| 4.1 Simulation tools and programming language .....         | 56          |
| 4.2 About the design .....                                  | 58          |
| 4.3 Files list and description .....                        | 59          |
| 4.4 Design overview .....                                   | 60          |
| 4.5 Design structure .....                                  | 61          |
| 4.6 Module Descriptions .....                               | 63          |

|  |                     |           |
|--|---------------------|-----------|
| 4.7  | Memory Map .....    | 73        |
| <b>Chapter Five Conclusion and future works.....</b> |                     | <b>75</b> |
| 5.1  | Design issues ..... | 75        |
| 5.2  | Conclusion.....     | 77        |
| 5.3  | Future works.....   | 78        |
| <b>References.....</b>                               |                     | <b>80</b> |

## List of figures

| #   | Figure description                             | Page |
|-----|--|------|
| 1.1 | Proposed configuration                         | 3    |
| 1.2 | WaveForm Pro                                   | 7    |
| 2.1 | Ethernet frame contents                        | 11   |
| 2.2 | Real time Ethernet architecture                | 12   |
| 2.3 | Ethernet frame format                          | 12   |
| 2.4 | AFDX switch                                    | 13   |
| 2.5 | Clock synchronization                          | 14   |
| 2.6 | Multiple Level Transition – 3                  | 15   |
| 2.7 | TTE Controller                                 | 19   |
| 2.8 | TTE Frame                                      | 20   |
| 2.9 | Time triggered Ethernet messages               | 21   |
| 4.1 | The layout of the receive side of the IP stack | 31   |
| 4.2 | The layout of the send side of the IP stack    | 32   |

## List of tables

| #   | Table description                                | Page |
|-----|--|------|
| 2.1 | Industrial Ethernet protocols                    | 10   |
| 2.2 | 4B/5B Encoding                                   | 15   |
| 2.3 | Propagation delay for some mediums               | 16   |
| 3.1 | Theoretical calculation for four frames          | 24   |
| 3.2 | Theoretical calculation for frames latency       | 24   |
| 3.3 | Theoretical delay calculation for 64 byte frame  | 25   |
| 3.4 | Theoretical delay calculation for 256 byte frame | 26   |
| 3.5 | Theoretical delay calculation for 512 byte frame | 27   |
| 4.1 | Memory map                                       | 36   |

# Chapter one

## Introduction

### Thesis Introduction

#### 1.1 Thesis Introduction

The AFDX network (Avionics Full Duplex switched Ethernet), as used on the A380 and A350, is based on switched Ethernet technology. In response to the "Availability" requirement, AFDX network is redundant. In case of major failure in AFDX network (the main and the redundant), there is the backup network ARINC 429 which is the backup network for the AFDX.

In practice, the backup network (ARINC 429) has never been in operation mode, the reason for that is the high reliability of the AFDX network and its redundancy. Therefore, this thesis is a proposal for a replacement technology of ARINC 429 with simpler technology and lower cost, which is in our case, is Time Triggered Ethernet.

The time-triggered Ethernet unifies real-time and non-real-time traffic into single communication architecture. Time-Triggered Ethernet introduces two message classes,

The standard event-triggered Ethernet messages (ET).

The time-triggered Ethernet messages (TT).

All TT messages transmitted periodically and scheduled a priori in a way that there are no conflicts on the network.



The network handles these messages according to the cut-through paradigm. Computer nodes containing TT Ethernet communication controllers establish and maintain global time base [20].

However, nodes containing standard Ethernet controllers are connected to a TT Ethernet system and can send ET messages without affecting the temporal properties of the TT messages. The global time format of the TT Ethernet deploys the Coordinate Universal Time (UTC) time format, which is compatible with the time format of the Institute of Electrical and Electronics Engineers (IEEE 1588) standard

Too many perspectives are considered in the physical layer:

- i. **Autonegotiation** (ANeg): Autonegotiation can be used by devices that are capable of different transmission rates (such as 10 Mbit/s and 100 Mbit/s), different duplex modes (half-duplex and full duplex), and/or different standards at the same speed (though in practice only one standard at each speed is widely supported). Every device declares its *technology abilities*, that is, its possible modes of operation. The two devices then choose the best possible mode of operation that is shared by the two devices, where higher speed (100 Mbit/s) is preferred over lower speed (10 Mbit/s), and full duplex is preferred over half duplex at the same speed [04].

ii. **Power over Ethernet** or **PoE** technology describes a system to transfer safely the electrical power, along with data, to remote devices over standard category 5 cable in an Ethernet network. It does not require modification of existing Ethernet cabling infrastructure.

iii. In computer networks, **propagation delay** is the amount of time it takes for the head of the signal to travel from the sender to the receiver over a medium. **Propagation delay** computed as the ratio between the link length and the propagation speed over the specific medium [19].

Propagation delay =  $d/s$  where  $d$  is the distance and  $s$  is the wave propagation speed. In wireless communication,  $s=c$ , i.e. the speed of light. In copper wires, the speed is typically about 67% of the speed of light.

iv. **Line coding:** Line coding consists of representing the digital signal and transporting it by an amplitude and time discrete signal that is optimally tuned for the specific properties of the physical channel (and of the receiving equipment). The waveform pattern of voltage or current used to represent the 1s and 0s of a digital signal on a transmission link is called line encoding. The common types of line encoding is unipolar, polar, bipolar and Manchester encoding [18].

v. **Quality:** the quality approach places an emphasis on Ethernets such as controls management, defined and well-managed processes, performance and integrity criteria, and identification of records.

## 1.2 Objectives and constraints

The objective of this thesis is to have a real time network, which has the main criteria of real time systems such as a bounded network, predictable and deterministic. This real time network will be a proposal to an alternative network solution.

This solution is for a real time network for critical systems such as aircrafts and vehicles. In addition, there is a proposal for a replacement of some real time networks through analytical study of TTE. This alternative solution based on **Ethernet** interconnection elements has the following properties:

- 1- It is **simpler** than the **existing** technologies.
- 2- It is still providing **deterministic** guarantees.
- 3- In addition to its **Simplicity**, there is the **low cost of material** and **lightweight**.

This alternative solution has efficient data distribution system that results in a reduction in the amount of aircraft or vehicle wiring and equipment interfaces.

### 1.3 Methodology

The effort was always to use open source tools and hardware to avoid proprietary copyrights and cost issues. A great deal of effort was made to search the already accomplished work in concerned fields in order to avoid re-inventing the wheel. No ready-made solution already existed for the objectives so many original works were done to accomplish these objectives. In order to save time and efforts, the approach used the existing work and modified it to our needs as much as possible.

It is proposed to use an existing protocol and existing mechanism using a simple Ethernet hub, and in order to meet the essential conditions for the real time network we had to make an analytical study of the proposed protocol.

It is proposed to use an existing protocol and existing mechanism based on International Electro Technical Commission (IEC 61784-4) which defines the real time protocols (such as EtherCAT, Ethernet power link, Modbus/TCP, PROFINET ... etc.) using a simple Ethernet hub as shown in Figure 1.1, and in order to meet the essential conditions for simple network.

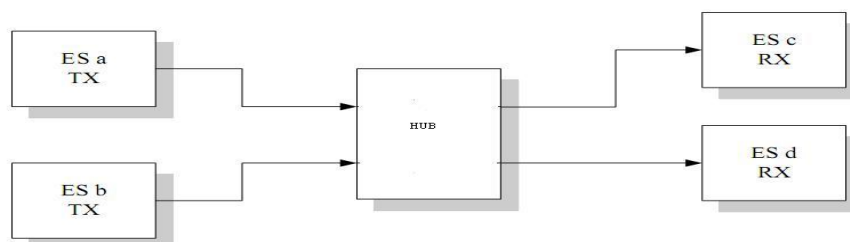


Figure 1.1: Proposed configuration

## 1.4 Literature review

A great deal of effort was made to search the already accomplished work in concerned fields in order to avoid re-inventing the wheel. No ready-made solution already existed for the objectives so many original works were done to accomplish these objectives. In order to save time and efforts, the approach used the existing work and modified it to our needs as much as possible. Some of the related works that were interesting and handled in details the time triggered Ethernet were as the following:

Astrit Ademaj [01] integrates real-time and non-real-time traffic into single communication architecture. TT Ethernet supports applications of different levels of criticality, from simple data acquisition systems, to multimedia systems up to the most demanding fault-tolerant, real-time control systems. The event-triggered traffic in TT Ethernet was handled in conformance with the existing Ethernet standards of the IEEE. The architecture deploys a TT Ethernet switch, which distinguishes between event-triggered (ET) and time-triggered (TT) Ethernet traffic. Time-triggered traffic is transmitted with a predictable transmission delay, whereas event-triggered traffic is transmitted on a best-effort basis. The paper elaborates on the usage of TT Ethernet for in-vehicle communication in order to integrate different in-vehicle communication subsystems into a single one.

Astrit Ademaj [02] IEEE 1588 Precision Clock Synchronization Protocol for Networked Measurement and Control Systems" (PTP), is used to quickly synchronize networked clocks with differing precision, resolution and stability to better than a microsecond-level accuracy. This protocol is designed for low-cost implementation in Ethernet networks with simple installation and maintenance.

Abdallah Alkhasawneh [03] proposed an AFDX backup network in case of major failure in the main network (ARINC 429), this alternative network works in case of major failure in communication. Because this network is redundant and rarely used, it should be simpler than the existing network with low-cost but should meet the basic elements of real time network such as Ethernet based, bounded and predictable. Many protocols studied as a proposed solution such as Ethernet power link (EPL), PROFINET, ETHERCAT and time triggered Ethernet, and finally a network based on the user's requirements which meet the real time Ethernet elements.

Hadriel Kaplan [04] discussed the Ethernet troubleshooting and the design of Ethernet. He discussed the common Ethernet problems and Ethernet hardware issues in order to design a real and complete network. In addition, he introduced a new and upcoming technologies related to the Ethernets.

IEC 61784-2 [05] specifies performance indicators supporting classification schemes for Real-Time Ethernet (RTE) requirements profiles and related network components based on ISO/IEC 8802-3, IEC 61158 series, and IEC 61784-1; RTE solutions that are able to run in parallel with ISO/IEC 8802-3-based applications. These communication profiles are called Real-Time Ethernet communication profiles.

IEEE 802.3 [06] is a collection of IEEE standards defining the Physical Layer and Data Link Layer's media access control (MAC) sub layer of wired Ethernet. This is generally a LAN technology with some WAN applications. Physical connections made between nodes and/or infrastructure devices (hubs, switches, routers) by various types of copper or fiber cable.

IEEE 802.5 [07] is a token ring access methodology, physical layer specification. Stations on a token ring LAN are logically organized in a ring topology with data transmitted sequentially from one ring station to the next with a control token circulating around the ring controlling access. This token passing mechanism is shared by ARCNET, token bus, and FDDI, and has theoretical advantages over the stochastic CSMA/CD of Ethernet.

Klaus Steinhammer [08] intended to support different types of applications, from simple data acquisition systems, via multimedia systems up to the most demanding safety-critical real-time control systems that require a fault-tolerant communication service that must be certified. TT Ethernet distinguishes between two traffic categories: the standard event-triggered Ethernet traffic and the time-triggered traffic with temporal guarantees. The event-triggered traffic in TT Ethernet is handled in conformance with the existing Ethernet standards of the IEEE. The design of TT Ethernet is driven by the requirement for certification of safety-critical systems and by an uncompromising stand with respect to the integration of legacy applications and legacy Ethernet hardware.

Klaus Steinhammer [09], the protocol distinguishes between event-triggered (ET) and time-triggered (TT) Ethernet traffic. Time-triggered traffic is scheduled and transmitted with a predictable transmission delay, whereas event-triggered traffic is transmitted on a best-effort basis. The event-triggered traffic in TT Ethernet is handled in conformance with the existing Ethernet standards of the IEEE.

Petr grillinger [10] presented the design of a Time-Triggered Ethernet (TTE) Switch, which is one of the core units of the Time-Triggered Ethernet system. Time-triggered Ethernet is a communication architecture intended to support event-triggered and time-triggered traffic in a single communication system.



Petr grillinger's [11] paper presents the rationale for and an outline of the design of a time-triggered (TT) Ethernet that unifies real-time and non-real-time traffic into a single coherent communication architecture. TT Ethernet is intended to support all types of applications, from simple data acquisition systems, to multimedia systems up to the most demanding safety-critical real-time control systems that require a certified fault-tolerant communication service. TT Ethernet distinguishes between two traffic categories: the standard event-triggered Ethernet traffic and the time-triggered traffic that is temporally guaranteed. The event-triggered traffic in TT Ethernet is handled in conformance with the existing Ethernet standards of the IEEE. The design of TT Ethernet has been driven by the requirement for certification of safety-critical configurations and an uncompromising stand with respect to the integration of legacy applications and legacy Ethernet hardware.

Vorgelegt von [12], defines two scenarios for RT Ethernet with multiprotocol gateway implementation. A verification environment developed for black box testing of arbitrary gateways. It includes checks of the gateway mappings and therefore tests the gateways data consistency and timing behavior. The test environment is implemented to test and compare different gateway implementations including the developed hardware gateway in realistic environments.

## 1.5. Statement of the problem

Several organizations are developing protocols for using Ethernet in process-related automation technology. No one currently knows which version of Real-Time Ethernet will ultimately prevail in the market; it is likely that several different solutions need to be established.

Providers in the automation industry will therefore have to support several Ethernet-based real-time communication systems in the future. This is why this thesis provides a complete, scalable range of solution, which enables uniform integration in various Ethernet solutions.

This solution is based on IEC 61784-2 which specifies performance indicators supporting classification schemes for Real-Time Ethernet (RTE) requirements profiles and related network components based on ISO/IEC 8802-3, IEC 61158 series, and IEC 61784-1; RTE solutions that are able to run in parallel with ISO/IEC 8802-3-based applications

## 1.6 Thesis Contribution

Based on the results, of the theoretical part and with respect to the (End-to-End delay) of 2 ms, a technology that is compatible with our case and our needs, can be recommended using a special microcontroller (FPGA). Xilinx FPGAs offers the most cost-effective, flexible, scalable platforms for implementation. Xilinx FPGAs also allows system designers to integrate processors and hubs.

Greater Performance and Easier Implementation: an on-board customizable 32-bit MicroBlaze soft processor allows implementation of process control and Real Time Ethernet FPGAs, and can also be implemented easily.

## **1.7 Thesis validation and verification tools**

In order to achieve our objectives, a study of the tools and equipment that are required to build and validate the practical part and the theoretical part is to be conducted. These tools are as follows:

### **1.7.1.Ethernet Hub.**

Although hubs and switches both glue the personal computers (PCs) in a network together, a switch is more expensive and a network built with switches is generally considered faster than one built with hubs. When a hub receives a packet of data (a frame in Ethernet) at one of its ports from a PC on the network, it transmits (repeats) the packet to all of its ports and, thus, to all of the other PCs on the network. If two or more PCs on the network try to send packets at the same time, a collision will occur.

When that happens all of the PCs have to go through a routine to resolve the conflict. The process is prescribed in the Ethernet Carrier Sense Multiple Access with Collision Detection (CSMA/CD) protocol. Each Ethernet Adapter has both a receiver and a transmitter. If the adapters and their receivers had no collisions, they would be able to send data at the same time they are receiving it

(full duplex). Because they have to operate at half duplex (data flows one way at a time) and a hub retransmits data from one PC to all of the PCs, the maximum bandwidth is 100 Mhz and that bandwidth is shared by all of the PC's connected to the hub.

A hub shares the total bandwidth among all users, while a switch provides a dedicated line at full bandwidth between every two devices transmitting to each other. The switch provides dedicated channels between each pair of users, whereas the hub shares its bandwidth with all users [15].

### **1.7.2. Field Programmable Gate Array (FPGA) component.**

A Field-Programmable Gate Array (FPGA) is an integrated circuit and it is designed to configure by the user after manufacturing—hence "field-programmable". The FPGA configuration is generally specified using a Hardware Description Language (HDL), similar to that used for an Application-Specific Integrated Circuit (ASIC). FPGA is used to implement any logical function that an ASIC could perform. The ability to update the functionality after shipping, and the low non-recurring engineering costs relative to an ASIC design, offers advantages for many applications.

FPGAs contain programmable logic components called "logic blocks", and a hierarchy of reconfigurable interconnects that allow the blocks to be "wired together" like a one-chip programmable

breadboard. Logic blocks configured to perform complex combinational functions, or merely simple logic gates like AND, XOR. In most FPGAs, the logic blocks also include memory elements, which may be simple flip-flops or more complete blocks of memory [17].

### 1.7.3. Verilog Hardware description language VHDL

VHDL (*VHSIC hardware description language*; VHSIC: *very-high-speed integrated circuit*) is a hardware description language used in electronic design automation to describe digital and mixed-signal systems such as field-programmable gate arrays and integrated circuits.

VHDL is a general-purpose language, and it does not require a simulator on which to run the code. There are many VHDL compilers, which build executable binaries. It can read and write files on the host computer, so a VHDL program written to generate another VHDL program incorporated in the design is developed. Because of this general-purpose nature, it is possible to use VHDL to write a *test bench* that verifies the functionality of the design using files on the host computer to define stimuli, interacts with the user, and compares results with those expected.

VHDL is not a case sensitive language. One can design hardware in a VHDL (such as Xilinx or Quartus) to produce the Register Transfer Level (RTL) schematic of the desired circuit. After that, the generated schematic can be verified using simulation

software (such as WaveFrom Pro) which shows the waveforms of inputs and outputs of the circuit after generating the appropriate test bench. To generate an appropriate test bench for a particular circuit or VHDL code, the inputs must be defined correctly.

The key advantages of VHDL when used for systems design are :

1. It allows the behavior of the required system described (modeled) and verified (simulated) before synthesis tools translate the design into real hardware (gates and wires).

2. It allows the description of a concurrent system (many parts, each with its own sub-behavior, working together at the same time). VHDL is a Dataflow language, unlike procedural computing languages such as BASIC, C, and assembly code, which all run sequentially, one instruction at a time [24].

### **1.7.1 WaveFormer Pro.**

WaveFormer Pro (figure 1.2) is a revolutionary new rapid-prototyping electronic design automation (EDA) tool that helps in faster design. WaveFormer Pro automatically determines critical paths, verifies timing margins and performs "what if" analysis to determine optimum clock speed.

WaveFormer Pro also specifies and analyzes system timing and perform Boolean level simulation without the need for

schematics or simulation models. WaveFormer Pro also has the ability to import and annotate simulation and logic analyzer data, for publication quality design documentation [25].

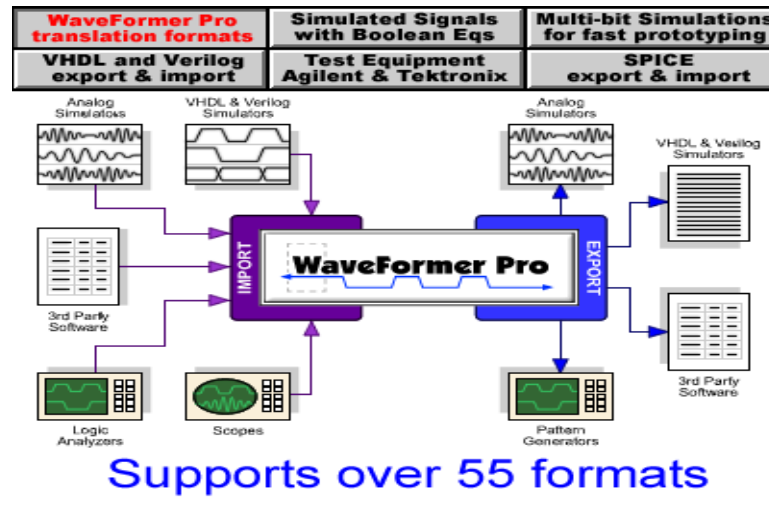


Figure 1.2: WaveFormer Pro

## 1.8.Thesis organization

This thesis describes in detail one of the Real time Ethernet technology, which is Time Triggered Ethernet (TTE) development of software and hardware adopted as a real time solution for critical systems. This thesis is considered as a perspective for future development.

This work consists of five main parts. The **first** part is an introduction for this thesis and it describes the general purpose (objectives) and constraints, the methodology by defining the existing technologies (parameters, specifications, and the reason for proposing a different type of technology (ies), thesis management and the validation and verification tools of this thesis.

The **second** part is a general description of Ethernet technology and real time networks. This goes through the industrial Ethernet technologies and real time Ethernet and defines the standards, which are compatible with the adopted solution such as IEC 61784-2, IEEE 802.3 and IEEE 1588. It also examines in more details the physical layer of the Ethernet and finally the time triggered Ethernet architecture and mechanism.

The **third** part is a study of the theoretical design through analyzing the theoretical and practical performance of time triggered Ethernet in different frames size (64, 128, 256 bytes).

The **fourth** part is the practical part and the coding section through VHDL language and simulation through WaveFromer Pro and the results obtained from the code for the proposed solution. Finally, the **last** part is the core of this thesis; it includes the conclusion based on the theoretical and practical and simulation results and then suggestions for future work.



## Chapter Two Ethernet and Real Time Networks

### 2.1.Industrial Real time networks

Industrial Ethernet (IE) is the name given to the use of the Ethernet Communications Interface in an industrial environment, for automation and production machine control. It is a plant, production, process and control focused technology unlike the standard information technology (IT) that is user focused. Industrial Ethernet focused on the production of items that make a company profitable through some manufacturing process.

IE components usually deployed on plant areas also designed to work and survive in these very harsh environments, where normal IT equipment would easily fail and without the stringent backup, survivability and mean time between failure (MTBF) requirements. Industrial Ethernet environments often involve unknown, hazardous environments and factors that can strongly influence the overall operation of standard Ethernet devices. In fact IE supports many factors that when loss of control occurs could cause serious disasters or the loss of life. See Table (2.1) [14].

Table 2.1: Industrial Ethernet protocols

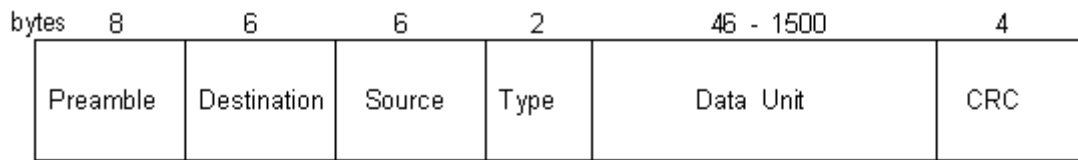
**Popular Industrial Fieldbus and Industrial Ethernet Protocols**

| Serial Protocols       | Ethernet-Based Protocols |
|------------------------|--------------------------|
| CANopen                | EtherCAT                 |
| CANopen                | Ethernet Power link      |
| DeviceNet/ControlNet   | Ethernet/IP              |
| LonWorks               | Lon over Ethernet        |
| Modbus/RTU             | Modbus/TCP               |
| PROFIBUS               | PROFINET                 |
| SERCOS I/II            | SERCOS III               |
| -                      | VARAN                    |
| CC-Link                | CC-Link IE               |
| Fieldbus Foundation H1 | Fieldbus Foundation HSE  |

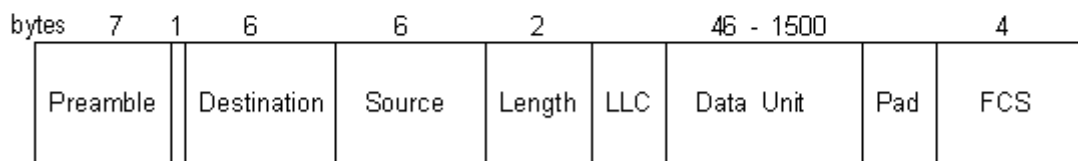
Ethernet is a family of frame-based computer networking technologies for local area networks (LANs). The name comes from the physical concept of the ether. It defines a number of wiring and signaling standards for the Physical Layer of the OSI networking model, through means of network access at the Media Access Control (MAC) /Data Link Layer (DLL), and a common addressing format.

Ethernet standardized as IEEE 802.3 (Figure 2.1). The combination of the twisted pair versions of Ethernet for connecting end systems to the network, along with the fiber optic versions for

site backbones, is the most widespread-wired LAN technology. It has been in use from around 1980 to the present, largely replacing competing LAN standards such as token ring, FDDI, and ARCNET.



DIX Ethernet Packet



IEEE 802.3 Frame

Figure 2.1: Ethernet frame contents

The **preamble** indicates the beginning of a transmission. A series of alternating 1's and 0's allows the receiving stations to synchronize signal detection to the transmission.

The **start of frame** is a special eight-bit signal pattern, "10101011", that signals the start of the packet information.

The **destination address** is the Ethernet address of the station that is to receive the message. All Ethernet addresses are 6 bytes long, and every Ethernet station is manufactured with a unique Ethernet address. A destination address of all 1's is the "broadcast" address; all stations on the network will receive a message sent to that address.

The **source address** identifies the station initiating the transmission.

The **frame length** indicates how many bytes are in the data portion of the packet. The data is the information transmitted.

The **cyclic redundancy check (CRC)** found from the bit pattern of the entire previous contents of the packet, using a mathematical formula. If the CRC is incorrect, there was a transmission error somewhere in the packet, and the entire packet is retransmitted.

All bits in an Ethernet frame are encoded in **Manchester code**. This code represents a 0 as a high-to-low voltage transition, and a 1 as a low-to-high voltage transition. This coding scheme is much more reliable than non-return zero (NRZ) coding, but requires twice the bandwidth, since each bit is encoded as two bit NRZ sequence (0 becomes 1-0, and 1 becomes 0-1) [16].

The frequency of collisions will limit the efficiency of an Ethernet network if too many data transmissions are attempted, the rate of collisions will become probably high. An Ethernet is optimal for networks where each station transmits only intermittently, and the probability of many stations transmitting simultaneously is low. In typical Ethernet implementations, collisions will limit the efficiency to

about 40% of the maximum possible packet transmission rate. If only one station is transmitting, the efficiency can increase to about 90% (since it must provide inter-frame spaces to monitor the network for other traffic).

## 2.2. Real time Ethernet network topology overview

The real time Ethernet (RTE) architecture (Figure 2.2) has the following characteristics.

- RTE is rather a network than a bus.
- Connections are full-duplex 10Mbits/s or 100MBit/s.
- RTE uses special extensions to provide deterministic timing and redundancy
  - End Systems (E/S) exchange Frames through Virtual Links (VL).

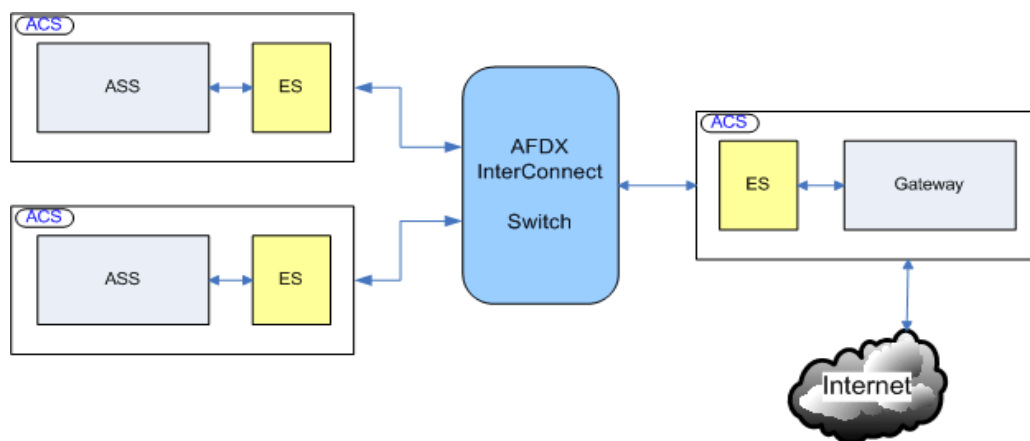


Figure 2.2: RTE architecture (ACS: Avionic Computer System, ES: End System, ASS: Avionic Sub-System)

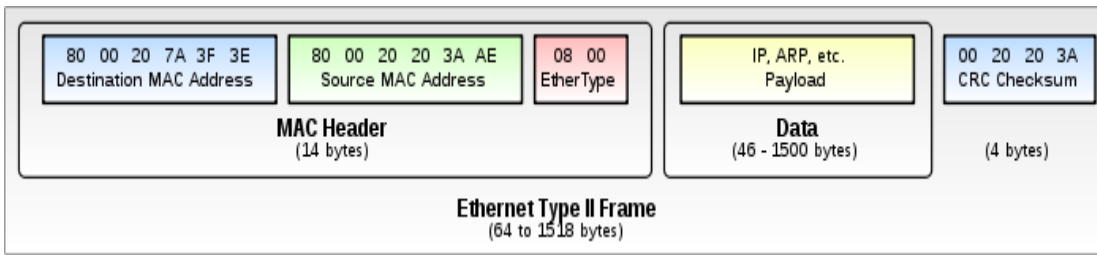


Figure 2.3: Ethernet frame format

## 2.3. Ethernet standards

Ethernet is originally based on the idea of computers communicating over a shared coaxial cable acting as a broadcast transmission medium. Ethernet evolved into the complex networking technology that today underlies most LANs. The coaxial cable is replaced with point-to-point links connected by Ethernet hubs and/or switches to reduce installation costs, increase reliability, and enable point-to-point management and troubleshooting [04].

Above the physical layer, Ethernet stations communicate by sending each other data packets, blocks of data that individually sent and delivered. With other IEEE 802, each Ethernet station has 48-bit MAC address to specify both the destination and the source of each data packet. Network interface cards (NICs) or chips normally do not accept packets addressed to other Ethernet stations. Adapters generally come programmed with a globally unique address, but this overridden, either to avoid an address change when an adapter replaced, or to use locally administered addresses.

Despite the significant changes in Ethernet from a thick coaxial cable bus running at 10 Mbit/s to point-to-point links running at 1 Gbit/s and beyond, all generations of Ethernet (excluding early experimental versions) share the same frame formats (and hence the same interface for higher layers), and can be readily interconnected.

### **2.3.1. International Electro technical Commission (IEC61784-2)**

RTE performance indicators, whose values are provided with RTE devices based on communication profiles specified in International electro technical commission (IEC 61784-2), enable the user to match network devices with application dependent performance requirements of an RTE network. Communication profiles within IEC 61784-2 refer to IEC 61158 series. The concept of IEC 61158 series, how the protocol and services are specified, is interpreted. Using the same concept to specify the different communication networks supports building bridges between the different communication networks [05].

The six primary aspects of RTE include (Full duplex, Redundancy, Deterministic, High-speed performance) [21].

Real time Ethernet is based on IEEE 802.3 standard (commercial Ethernet using wire) with either 10Mbits/s or 100Mbits/s, real time Ethernet uses a special protocol to provide deterministic timing and redundancy management, the main elements of an RTE network are End systems, switches and links (figure 2.4).

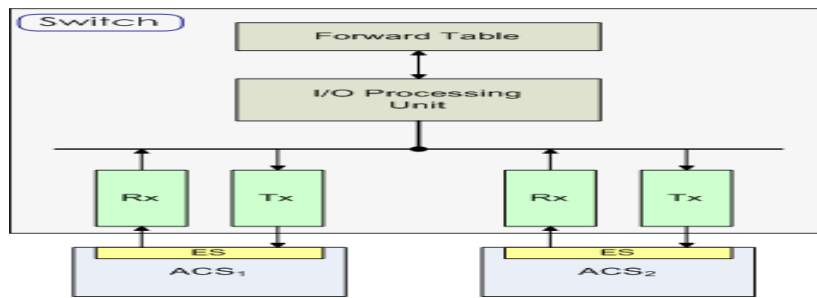


Figure 2.4: AFDX switch

### **2.3.2. Institute of Electrical and Electronics engineers (IEEE 802.3)**

Institute of electrical and electronics engineers (**IEEE 802.3**) made a collection of IEEE standards defining the Physical Layer and Data Link Layer's media access control (MAC) sublayer of wired Ethernet. This is generally a LAN technology with some WAN applications. Physical connections made between nodes and/or infrastructure devices (hubs, switches, routers) by various types of copper or fiber cable. 802.3 is a technology that supports the IEEE 802.1 network architecture [22].

The maximum packet size is 1518 bytes, but to allow the Q-tag for Virtual LAN and priority data in 802.3ac it is extended to 1522 bytes. If the upper layer protocol submits a protocol data unit (PDU) less than 64 bytes, 802.3 will pad the data field to achieve the minimum 64 bytes. The minimum Frame size will then always be 64 bytes.



Although it is not technically correct, the terms *packet* and *frame* are often used interchangeably. The ISO/IEC 8802-3 and ANSI/IEEE 802.3 standards refer to MAC sub-layer frames consisting of the destination address, the source address; length/type, data payload, and frame check sequence (FCS) fields. The preamble and Start Frame Delimiter (SFD) (usually) are together considered a header to the MAC frame. This header and the MAC frame constitute a *packet*.

### **3.2.3. Institute of Electrical and Electronics Engineers (IEEE 1588)**

The IEEE 1588 is a protocol designed to synchronize the real time clock in the nodes of distributed systems that communicate using a network (Figure 2.5). The target is groups of relatively stable components, locally networked (a few subnet), cooperating on a set of well-defined tasks.

Measurement and control systems have a number of requirements that met by a clock synchronization technology. In particular:

- Timing accuracies are often in the sub-microsecond range,
- A minimum of administration is highly desirable,
- The technology must be capable of implementation on low cost.
- The required network and computing resources should be minimal.

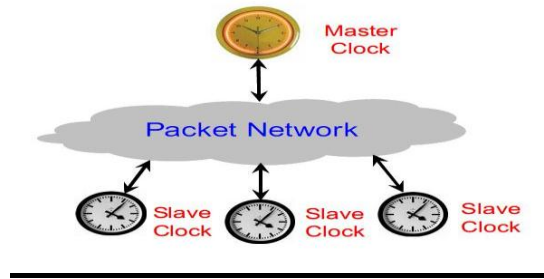


Figure 2.5: clock synchronization

This standard defines a protocol enabling precise synchronization of clocks in measurement and control systems implemented with technologies such as network communication, local computing and distributed objects. The protocol will enable heterogeneous systems that include clocks of various inherent precision, resolution and stability to synchronize. The protocols will support system-wide synchronization accuracy in the sub-microsecond range with minimal network and local clock computing resources [02].

## 2.4 Ethernet Physical layer

### 2.4.1 100BASE-T

100BASE-T Is the IEEE standard that defines the requirement for sending information at 100 Mbps on unshielded twisted-pair cabling, and defines various aspects of running baseband Ethernet on this cabling.

To increase the speed of Ethernet transmission to 100 Mbps, different encoding strategies are required to reduce the bandwidth of the transmission. Encoding data at 100 Mbps using Manchester

encoding would create a bit stream of 200 Mbps, which would require at least 100 MHz of analog bandwidth. Category 5 unshielded twisted pair (UTP) cable rated up to 100 MHz, but transmitting a signal of 100 MHz bandwidth would be unreliable. To reduce the bandwidth of the signal, a different encoding scheme is required [16].

Because negative voltages are as easy to transmit as positive voltages, 100BASE-TX Ethernet uses a multiple level-encoding scheme called MLT-3 (Multiple Level Transition - 3) as shown in Figure 2.6:

0: No transition

1: Signal transition (low to zero, high to zero, zero to low or high).

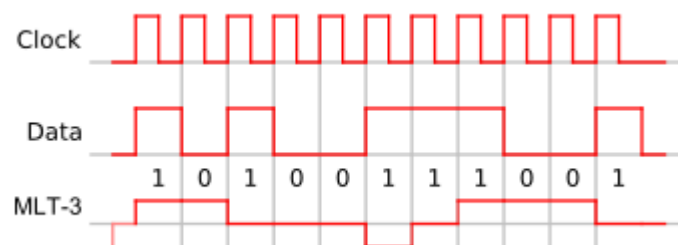


Figure 2.6: Multiple Levels Transition – 3

The bandwidth required to transmit this signal is one-half the bandwidth required for a two level scheme. However, a long stream of zeros (common in most data streams) will cause the line to hold a constant voltage. During this time, the receiver may lose the clocking synchronization necessary to read the signal. For this reason,

the data is encoded at the physical layer (in the hardware) to encode each 4-bit sequence as a 5-bit pattern. This scheme is called 4B5B encoding (see Table 2.2).

It increases the bit rate to 125 Mbps, but it allows the receiver to read a steady stream of zeros without losing synchronization. No more than 2 consecutive zeros will be transmitted in any encoded stream [18].

Table 2.2: 4B/5B Encoding.

| Hex Data | 4B Binary Data | 5B Encoded Data |
|----------|----------------|-----------------|
| 0        | 0000           | 11110           |
| 1        | 0001           | 01001           |
| 2        | 0010           | 10100           |
| 3        | 0011           | 10101           |
| 4        | 0100           | 01010           |
| 5        | 0101           | 01011           |
| 6        | 0110           | 01110           |
| 7        | 0111           | 01111           |
| 8        | 1000           | 10010           |
| 9        | 1001           | 10011           |
| A        | 1010           | 10110           |
| B        | 1011           | 10111           |

|   |      |       |
|---|------|-------|
| C | 1100 | 11010 |
| D | 1101 | 11011 |
| E | 1110 | 11100 |
| F | 1111 | 11101 |

## 2.4.2 Propagation Delay

The propagation speed of a medium refers to the speed that the data travels through that medium. Propagation delays differ between mediums, which affect the maximum possible length of the Ethernet topology running on that medium. In the table 2.3, C refers to the speed of light in a vacuum, or 300,000 kilometers per second.

Table 2.3: Propagation delay for some mediums

| Medium       | Propagation speed     |
|--------------|-----------------------|
| Thick coax   | .77c (231,000 km/sec) |
| Thin coax    | .65c (195,000 km/sec) |
| Twisted pair | .59c (177,000 km/sec) |
| Fiber        | .66c (198,000 km/sec) |
| AUI cable    | .65c (195,000 km/sec) |

From these values, the size of a bit on 10BaseT is calculated. 10BaseT is twisted pair, which has a propagation delay of 177,000

km/sec. 177,000 km/sec divided by 10 million bits per second is 17.7 meters, or the size of a single bit on a 10BaseT network.

The maximum propagation delay through a network is calculated through dividing the maximum length over the speed.

The time delay (T) of a twisted pair cable depends on the amount of inductance and capacitance of the wire pair [26]:

$$T = C * Z_0 * L \quad \text{where}$$

- **C**=speed of light,
- **Z<sub>0</sub>**: line impedance which is the ratio of the amplitudes of a *single* pair of voltage and current waves propagating along the line in the absence of reflections,
- **L**: length of the cable

The transmission speed in twisted pair wiring is generally expressed as being from 0.59 to 0.65 times the propagation speed of light. Therefore, the time delay  $\tau$  will be between 5.13 and 5.65 nanoseconds per meter of distance.

### **2.4.3 Auto negotiation (ANeg)**

Many different modes of operations (10BASE-T half duplex, 10BASE-T full duplex, 100BASE-TX half duplex ...) exist for Ethernet over twisted pair cable using 8P8C modular connectors, and most devices are capable of different modes of operations.

The autonegotiation standard contained a mechanism for detecting the speed but not the duplex setting of an Ethernet peer that did not use autonegotiation. An autonegotiating device defaults

to half duplex, when the remote does not negotiate, as the remote peer assumed a hub (which always has autonegotiation disabled and supports only half-duplex mode). If the remote is operating in half-duplex mode this works. However, if the remote is in full duplex mode, this generates a duplex mismatch. When two interfaces are connected and set to different "duplex" modes, the effect of the duplex mismatch is a network that works, but is much slower than its nominal speed, and generates more collisions. The primary rule for avoiding this is never set one end of a connection to a forced full duplex setting and the other end to autonegotiation.

When two interfaces are connected to each other with one set to autonegotiation and one set to full duplex mode, a duplex mismatch results because the autonegotiation process fails and half duplex is assumed. The interface in full duplex mode then transmits at the same time as receiving, and the interface in half-duplex mode then gives up on transmitting a frame. The interface in half-duplex mode is not ready to receive a frame, so it signals a collision, and transmissions halted, for amounts of time based on back off (random wait times) algorithms. When both packets start trying to transmit again, they interfere again and the back off strategy may result in a longer and longer wait time before attempting to transmit again; eventually a transmission succeeds but this then causes the flood and collisions to resume [07]



Autonegotiation (ANeg) can be used by devices that are capable of different transmission rates (such as 10 Mbit/s and 100 Mbit/s), different duplex modes (half-duplex and full duplex), and/or different standards at the same speed (though in practice only one standard at each speed is widely supported). Every device declares its *technology abilities*, that is, its possible modes of operation. The two devices then choose the best possible mode of operation that is shared by the two devices, where higher speed (100 Mbit/s) is preferred over lower speed (10 Mbit/s), and full duplex is preferred over half duplex at the same speed.

Parallel detection is used when a device that is capable of autonegotiation is connected to one that is not. This happens if the other device does not support autonegotiation or autonegotiation disabled via software. In this condition, the device that is capable of autonegotiation can determine the speed of the other device, and then choose the same speed for itself. This procedure cannot determine the presence of full duplex, so half-duplex is always assumed. A duplex mismatch will result if the other device is in full duplex mode, that is, one device is using full duplex while the other one is using half duplex. The typical effect of duplex mismatch is that the connection is working but at a very low speed.

Autonegotiation on the link is exchanged when:

- i. Link is initially connected
- ii. Device at either end of the link is powered up
- iii. Device is reset or initialized
- iv. Renegotiation request is made

#### **2.4.4 Determinism**

Determinism, usually means that access to the control network by a node maybe delayed by at most sometime  $t$ , where  $t$  is known. Every node thus has fair and equal access to the network since no one node is delayed from gaining access for a longer than time  $t$ .

Proponents of determinism claim that knowing the value of  $t$  makes the design of control networks much easier, and that networked industrial control systems should only use deterministic protocols.

There are four basic families of media access protocols: time division multiplexing (TDM), token bus, token ring and CSMA. Of these four, all but CSMA are commonly considered deterministic.

**Synchronous TDM** protocols assign a unique time for each node relative to a clock pulse (commonly called “start of frame”) on the medium, each node counts down to its time and transmits typically a single byte per frame. As long as all nodes have the same time, the delay to access the network, is bounded by the number of time slots assigned.

The principle weakness of TDM protocols are that they waste the bandwidth if all the nodes do not have always something to send, and a loss of synchronization to the start of frame signal results in the loss of all communications [13].

**Token passing protocols** (token bus, token ring) have a special message called “token” which upon receipt, grants the right to transmit on the medium. Each node may hold the token for a limited time before passing the token to the next node in the network. In this way, access to the network is bounded by maximum latency of the token as it passes to each of the nodes. In token ring network, only the next station on the wire receives the token. In token bus network, all stations receive the token and must then decide if it is intended for them.

**CSMA** is a listen-before-transmit scheme in which a node with a message to transmit first listens to the network. If no message traffic is detected, evidenced by the absence of a carrier signal, then the node will transmit. Unlike the other media access protocols, the CSMA protocol family has many variants. The best-known form of the CSMA protocol is Ethernet, also known IEEE 802.3. Ethernet was not designed for control systems and exhibits very poor characteristics near network overload; for this reason it is not often used for control systems.

The limitations of Ethernet have created the impression that CSMA protocols are not suitable for computer control applications, though at least one variation of CSMA is ideally suited for such an application [23].

CSMA is fully deterministic when used in master-slave operation, however, it cannot be deterministic in peer-to-peer operation because nodes are not provided with equal access to the network. Nodes transmit on the basis of their ability to resolve packet collisions, and sometimes on the basis of priority messaging as well, and it is precisely these features which make a properly implemented, non-deterministic CSMA protocol so well suited for control applications [06].

#### **2.4.5 Power over Ethernet**

**Power over Ethernet** or **PoE** technology describes a system for passing electrical power safely, along with data, on Ethernet cabling. Standard versions of PoE specify category 5 cable or higher. Power can come from a power supply within a PoE-enabled networking device such as an Ethernet switch or from a device built for "injecting" power onto the Ethernet cabling.

The IEEE 802.3af PoE standard provides up to 15.4 W of DC power (minimum 44 V DC and 350 mA) to each device. [ $P = I^2 * R \Rightarrow I^2 * V/I \Rightarrow I * V = 0.35 * 44 = 15.4 \text{ W}$ ]

Only 12.95 W is assured to be available at the powered device as some power dissipated in the cable.

The IEEE 802.3at PoE standard, sometimes called "POE+", provides up to 25 W of power. Some vendors have announced products that claim to comply with the new 802.3at standard and offer up to 51 W of power over a single cable by utilizing all 4 pairs in the Cat.5 cable.

PoE is presently deployed in applications where universal serial bus (USB) is unsuitable and where alternative current (AC) power would be inconvenient, expensive (mains wiring must often be done by qualified and/or licensed electricians for legal or insurance reasons) or infeasible to supply. However, even where USB or AC power is used, PoE has several advantages over either, including the following:

- Cheaper cabling even category 5 cables is cheaper than USB repeaters, and the task of meeting building code requirements to run AC power cable is eliminated.
- Direct injection from standard 48 V Direct current (DC) battery power arrays; this enables critical infrastructure to run more easily in outages, and make power rationing decisions centrally for all the PoE devices [19].

## 2.5 Time triggered Ethernet

The reason of choosing time triggered Ethernet is that it has advantages over other real time protocols such [09]:

1- There is no need for a master for communication but only for synchronization.

2- It has a best usage for the bandwidth in case of a heavy traffic.

3- Predictable real-time response times within the network.

Even though they have some drawbacks such as

1- It is difficult to fit messages with differing periods into a static schedule.

2- Traffic requires tight synchronization.

3- There is a bandwidth waste in case of light traffic, because of having empty slots.

### 2.5.1 Time triggered Ethernet architecture

A TTE system consists of a set of computer nodes that communicate via the TTE protocol. A TTE node consists of a host computer and a TTE communication controller that is connected to a switch via a bidirectional point-to-point link. User applications are executed on the host computer, whereas the communication controller is in charge of executing the TTE communication protocol [08].

The TTE controller is implemented in either software or in hardware (e.g., FPGA). As long as both implementations follow the TTE protocol specification, they may mix in a single network without compatibility problems (figure 2.7) [12].

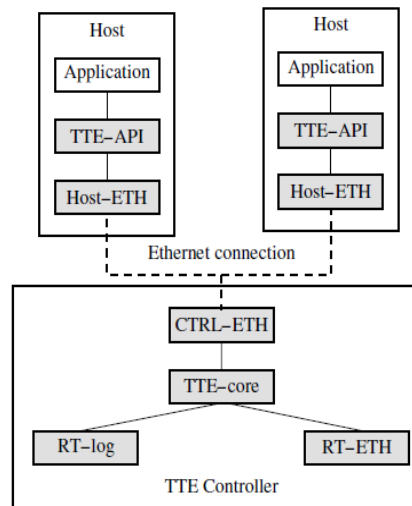


Figure 2.7: TTE Controller

The TTE protocol distinguishes between two different classes of traffic:

- The event-triggered (ET) traffic is not required to meet strict temporal requirements regarding the transmission delay.
- The time-triggered (TT) traffic whose transmission is scheduled based on a conflict free scheme, and which has to meet strict temporal requirements regarding the transmission delay [11].

The TTE message will be based on the standard Ethernet message format. The standard Ethernet frame header contains a two-byte Type Field that specifies the type of an Ethernet message, which provides a context for interpretation of the data field of the frame (for protocol identification). The Ethernet standard authority of the IEEE administers the contents of the Type Field in order to assure a consistent development of Ethernet. This standard authority has assigned the value 0x88D7 as the content of the Type Field to uniquely identify a TTE message and the associated message protocol (see figure 2.8).

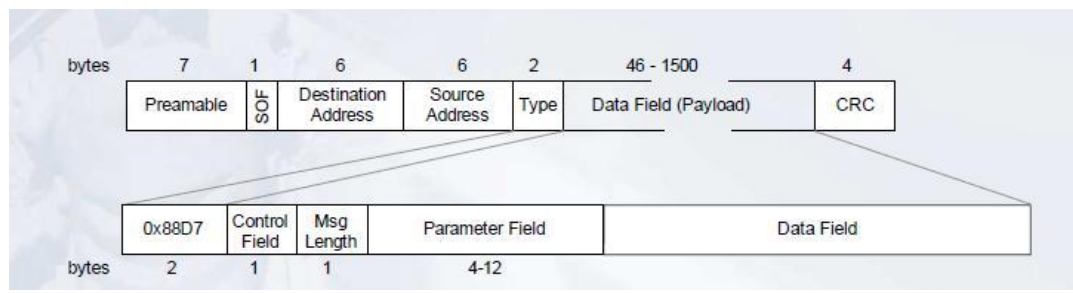


Figure 2.8: TTE Frame

Since TT messages are scheduled and ET messages are not, a run time conflict situation between a TT message and ET traffic might occur. To solve this problem, TTE introduces a specific switch, which is denoted as TTE switch. The TTE switch guarantees predictable transmission delays for TT traffic, even in the case when ET and TT messages are transmitted using the same Ethernet network. Whenever there is, a run-time conflict between ET traffic and a TT message the TTE switch preempt the ET traffic and transmit the TT message with a constant transmission delay.



The TTE switch autonomously retransmits any preempted ET message immediately after the transmission of the TT message has finished (using a best-effort method) [10].

The following types of different messages are provided:

- **Time-triggered messages** sent over the network at predefined times and take precedence over all other message types. The occurrence, temporal delay and precision of time-triggered messages are predefined and guaranteed. The messages have as little delay on the network as possible and their temporal precision is as accurate as necessary.
- **Rate-constrained messages** are used for applications with less stringent determinism and real-time requirements. These messages guarantee that the bandwidth is predefined for each application and delays and temporal deviations have defined limits.
- **Best-effort messages** follow a method, which is well known in classical Ethernet networks. There is no guarantee whether, when these messages are transmitted, what delays occur, and if messages arrive at the recipient. Best-effort messages use the remaining bandwidth of the network and have less priority than the other two types of messages [20].

Figure 2.9 gives a better understanding of the three types of messages:

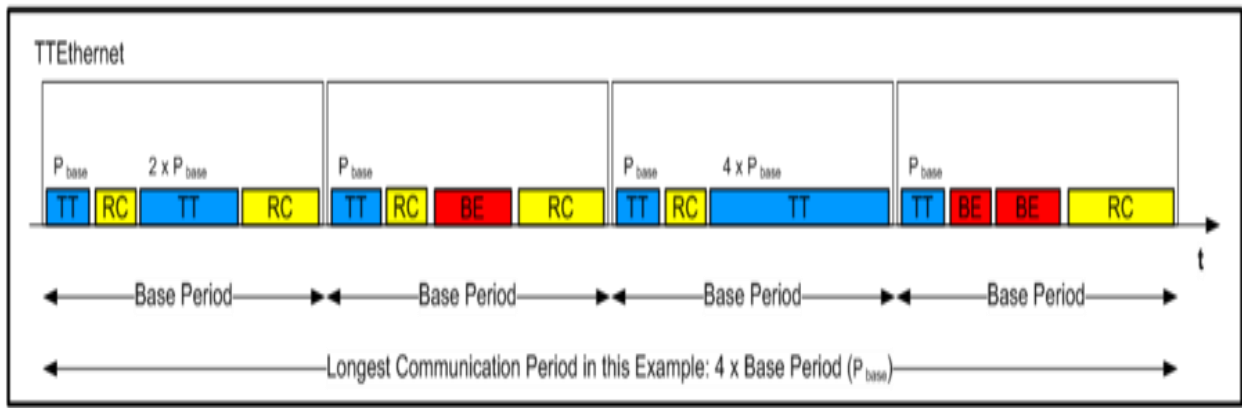


Figure 2.9: Time triggered Ethernet messages

### 2.5.2 Time triggered Ethernet mechanism

Time-Triggered Ethernet (TTE) allows competing senders to coexist with cooperative senders on the same network and preserves the temporal predictability of the traffic among the cooperative senders. To integrate competing traffic seamlessly in the same network without interfering with the cooperative traffic, Time-Triggered Ethernet introduces two message classes using the standardized Ethernet frame format [01]:

- The standard Ethernet messages used for event- triggered traffic by competing transmitters,
- The Time-Triggered Ethernet messages transferred among cooperative senders.

Standard Ethernet messages are transmitted in those time intervals where the communication medium is not needed for transmission of time-triggered messages. To avoid that standard Ethernet traffic affects the temporal properties of the time-triggered traffic, time-triggered messages preempt all standard Ethernet messages that are in their transmission path. This preemption is implemented in a dedicated TTE switch.

Introducing the two message classes guarantees the compatibility with standard Ethernet “commercial of-the-shelf” (COTS) components and existing networking protocols on the one hand, and the possibility to realize a scheduled message transfer between all cooperative senders by establishing and maintaining a global time base on the other hand.

A TTE communication controller manages the timely reception and transmission of TT messages, builds and maintains the global view of time, maintains current cluster status information, and exchanges data with the application.

## Chapter Three Theoretical Design

### 3.1 Time Triggered Ethernet theoretical performance study

The Time-Triggered Ethernet communication is not via a master. There is no request to send before the emission but a data slot is allocated to each terminal. In our context, the terminals do not produce an Event-Triggered Message. There is no pre-emption to do. In addition, the use of one (or more) hub is better than using a switch for reasons of speed of transmission. The hubs of the potential collisions are avoided through TDMA.

With regard to this thesis, the terminals have no real need to send frames in largelength. A frame size of 256 bytes or 512 bytes is sufficient. The issuance of a frame of 256 bytes on a 10 Mbps link takes 211.2  $\mu$ s. To this the interframe Gap (IFG) of 9.60  $\mu$ s must be added. Therefore, we can take a slot of 220.8  $\mu$ s. The jitter is in the duration of the slot to avoid overlapping slots and collisions of frames. The hub introduces jitter of more than 200  $\mu$ s for 256 frames of bytes, while the TT-switch jitters are typically 0.27  $\mu$ s.

Since a "byte" is a unit of data comprised of eight separate bits.

Number of bits = frame size \* 8 bits so

064 bytes frame size = 064 \* 8 bits = 0512 bits,

128 bytes frame size = 128 \* 8 bits = 1024 bits,

256 bytes frame size = 256 \* 8 bits = 2048 bits,

512 bytes frame size = 512 \* 8 bits = 4096 bits.

The propagation delay for using 10Mb/s line speed = number of bits / line speed

Therefore, the propagation for each frame size is

0512 bits / (10 \* 10<sup>6</sup>) = 051.2 \* 10<sup>-6</sup> = 051.2 μs

, 1024 bits / (10 \* 10<sup>6</sup>) = 102.4 \* 10<sup>-6</sup> = 102.4 μs

, 2048 bits / (10 \* 10<sup>6</sup>) = 204.8 \* 10<sup>-6</sup> = 204.8 μs

, 4096 bits / (10 \* 10<sup>6</sup>) = 409.6 \* 10<sup>-6</sup> = 409.6 μs.

Preamble = 8 bytes \* 8 = 64 bits

The propagation delay for using 10Mb/s line speed = number of bits / line speed

So

64 / (10 \* 10<sup>6</sup>) = 6.4 \* 10<sup>-6</sup> = 6400 ns

IFG = 12 byte \* 8 = 96 bits

The propagation delay for using 10Mb/s line speed = number of bits / line speed

So

$$96 / (10 * 10^6) = 9.6 * 10^{-6} = 9600 \text{ ns}$$

$$\text{Jitter} = 12 \text{ byte} * 8 = 96 \text{ bits}$$

The propagation delay for using 10Mb/s line speed = number of bits / line speed

So

$$96 / (10 * 10^6) = 9.6 * 10^{-6} = 9600 \text{ ns}$$

The theoretical results presented consider 4 slot sizes (as shown in table 3.1). The first value corresponds to the propagation delay for each frame, the second corresponds to the emission duration of the frame with a header, preamble and IFG, the delay is introduced by the hub (s), the maximum jitter is introduced by the physical layer transmitter and receiver and by the hub (s).

Table 3.1: Theoretical calculation for four frames

|                        | <b>64</b> | <b>128</b> | <b>256</b> | <b>512</b> |
|------------------------|-----------|------------|------------|------------|
| Propagation delay (µs) | 51,2      | 102,4      | 204,8      | 409,6      |
| Preamble (ns)          | 6400      | 6400       | 6400       | 6400       |
| Issuance (µs)          | 52,8      | 105,6      | 211,2      | 420,4      |
| IFG (ns)               | 9600      | 9600       | 9600       | 9600       |
| Jitter (µs)            | 200       | 200        | 200        | 200        |
| Total (µs)             | 320       | 424        | 447.68     | 1046       |

The theoretical calculations for the latency of these frames will be

$$= \text{Frame size} + \text{preamble} + \text{IFG}$$

$$064 + 8 + 12 = 084 \text{ bytes,}$$

$$128 + 8 + 12 = 148 \text{ bytes,}$$

$$256 + 8 + 12 = 276 \text{ bytes,}$$

$$512 + 8 + 12 = 532 \text{ bytes.}$$

The number of bits will be = frame size \* 8 bits

$$084 \text{ bytes} * 8 \text{ bits} = 0672 \text{ bits,}$$

$$148 \text{ bytes} * 8 \text{ bits} = 1184 \text{ bits,}$$

$$276 \text{ bytes} * 8 \text{ bits} = 2208 \text{ bits,}$$

$$532 \text{ bytes} * 8 \text{ bits} = 4256 \text{ bits.}$$

The latency for using 10Mb/s line speed = number of bits / line speed

$$0672 \text{ bits} / (10 * 10^6) = 067.2 \mu\text{s,}$$

$$1184 \text{ bits} / (10 * 10^6) = 118.4 \mu\text{s,}$$

$$2208 \text{ bits} / (10 * 10^6) = 220.8 \mu\text{s,}$$

$$4256 \text{ bits} / (10 * 10^6) = 425.6 \mu\text{s.}$$

The theoretical calculations for the latency of these frames are shown in the following table (Table 3.2):

Table 3.2: Theoretical calculation for frames latency

| Frame size | Preamble (Bytes) | IFG (Bytes) | Total bytes | Total bits | Latency ( $\mu\text{s}$ ) |
|------------|------------------|-------------|-------------|------------|---------------------------|
| 64         | 8                | 12          | 084         | 672        | 067.2                     |
| 128        | 8                | 12          | 148         | 1184       | 118.4                     |
| 256        | 8                | 12          | 276         | 2208       | 220.8                     |
| 512        | 8                | 12          | 532         | 4256       | 425.6                     |

## 3.2 Time Triggered Ethernet practical performance study

Based on the theoretical study, a practical experiment took place in 2009 ISAE university, Toulouse-France through a collision analysis for a hub. The TDMA mechanism was used in order to avoid collisions through a traffic generator device called IXIA 400T [03].

### 3.2.1 TTE practical performance study for 64 bytes frame size

Taking advantage of TDMA mechanism two ports for transmission were defined, the **first** port had two types of sequential frames, the first frame was an (64-byte) frame size. It was called shift frame with preamble of 8 bytes, an inter-frame gap (2012 bytes long) and it was transmitted as 1 packet per burst and 1 burst per stream. The second frame was an (256 bytes) frame with 8 bytes preamble and only one packet per burst. This was sent after the first frame in continuous mode [03].

The **second** port had the same parameters except the IFG of the shift frame was 2096 bytes. In addition, it is important to know that this IFG depended on the number of the devices connected to the network in order to know the minimum time slot for each device. This difference ( $2096 - 2012 = 84$ ) was 64 bytes the frame size and 8 bytes preamble and 12 bytes gap. All ports interleaved in their transmission [03].



Total frame size = preamble + shift frame + IFG

8+64+2012 = 2084 bytes \* 8 bits = 16672 bits,

8+64+2096 = 2168 bytes \* 8 bits = 17344 bits,

8+64+2180 = 2252 bytes \* 8 bits = 18016 bits,

8+64+2264 = 2336 bytes \* 8 bits = 18688 bits.

The transmission delay using 10Mb/s line speed = number of bits / line speed

16672 bits/ (10 \* 10<sup>6</sup>) = 1667.2 μs,

17344 bits/ (10 \* 10<sup>6</sup>) = 1734.4 μs,

18016 bits/ (10 \* 10<sup>6</sup>) = 1801.6 μs,

18688 bits/ (10 \* 10<sup>6</sup>) = 1868.8 μs.

Table 3.3: Theoretical calculation for four devices transmitting of 64-frame size

|          | Bytes | Bits  | Time in μs |
|----------|-------|-------|------------|
| 8+64+201 | 2084  | 16672 | 1667.2     |
| 2        |       |       |            |
| 8+64+209 | 2168  | 17344 | 1734.4     |
| 6        |       |       |            |

|          |      |       |        |
|----------|------|-------|--------|
| 8+64+218 | 2252 | 18016 | 1801.6 |
| 0        |      |       |        |
| 8+64+226 | 2336 | 18688 | 1868.8 |
| 4        |      |       |        |

### **3.2.2 TTE practical performance study for 256 bytes frame size**

For the 256 bytes frame size, we had the same configuration, as the previous experiment with the difference that the second port had the same parameters except the IFG was 2288 bytes). In addition, it is important to know that this IFG depended on the number of the devices that they connected to the network in order to know the minimum time slot for each device. This difference was (2288 – 2012 = 276) 256 bytes the frame size and 8 bytes preamble and 12 bytes gap.

Total frame size = preamble + shift frame + IFG

$$8+64+2012 = 2084 \text{ bytes} * 8 \text{ bits} = 16672 \text{ bits,}$$

$$8+64+2288 = 2352 \text{ bytes} * 8 \text{ bits} = 18880 \text{ bits,}$$

$$8+64+2564 = 2636 \text{ bytes} * 8 \text{ bits} = 21088 \text{ bits,}$$

$$8+64+2840 = 2912 \text{ bytes} * 8 \text{ bits} = 23296 \text{ bits.}$$

The transmission delay using 10Mb/s line speed = number of bits / line speed

$$16672 \text{ bits} / (10 * 10^6) = 1667.2 \mu\text{s,}$$

$$18880 \text{ bits} / (10 * 10^6) = 1888.0 \mu\text{s,}$$

$$21088 \text{ bits} / (10 * 10^6) = 2108.8 \mu\text{s,}$$

$$23296 \text{ bits} / (10 * 10^6) = 2329.6 \mu\text{s}.$$

Table 3.4: Theoretical calculation for four devices transmitting of 256-frame size

|           | Bytes | Bits  | Time<br>in $\mu\text{s}$ |
|-----------|-------|-------|--------------------------|
| 8+64+2012 | 2084  | 16672 | 1667.2                   |
| 8+64+2288 | 2352  | 18880 | 1888.0                   |
| 8+64+2564 | 2636  | 21088 | 2108.8                   |
| 8+64+2840 | 2912  | 23296 | 2329.6                   |

Therefore, with multiple ports we could have calculated the pause frame for each port and managed to transmit a frame through a hub without having any collision. Of course, since the shift is frame for all ports transmitted at the same time, for the first transmission, we will have a collision, but since each port will wait for a random time to retransmit, and thanks to the inter-frame gap we will manage to have a mechanism similar to the TDMA.

### **3.2.3 TTE practical performance study for 512 bytes frame size**

For the 512 bytes frame size, we had the same configuration, as the previous experiment with the difference that the second port had the same parameters except the IFG was 2288 bytes). In addition, it is important to know that this IFG depended on the number of the devices connected to the network in order to know the

minimum time slot for each device. This difference (2544 – 2012 = 532) was 512 bytes the frame size and 8 bytes preamble and 12 bytes gap.

Total frame size = preamble + shift frame + IFG

$$8+64+2012 = 2084 \text{ bytes} * 8 \text{ bits} = 16672 \text{ bits,}$$

$$8+64+2544 = 2616 \text{ bytes} * 8 \text{ bits} = 20928 \text{ bits,}$$

$$8+64+3076 = 3148 \text{ bytes} * 8 \text{ bits} = 25184 \text{ bits,}$$

$$8+64+3608 = 3680 \text{ bytes} * 8 \text{ bits} = 29440 \text{ bits.}$$

The transmission delay using 10Mb/s line speed = number of bits / line speed

$$16672 \text{ bits} / (10 * 10^6) = 1667.2 \mu\text{s,}$$

$$20928 \text{ bits} / (10 * 10^6) = 2092.8 \mu\text{s,}$$

$$25184 \text{ bits} / (10 * 10^6) = 2518.4 \mu\text{s,}$$

$$29440 \text{ bits} / (10 * 10^6) = 2944.0 \mu\text{s.}$$

Table 3.5: Theoretical calculation for four devices transmitting of 512-frame size

|           | Bytes | Bits  | Time<br>in $\mu$ s |
|-----------|-------|-------|--------------------|
| 8+64+2012 | 2084  | 16672 | 1667.2             |
| 8+64+2544 | 2616  | 20928 | 2092.8             |
| 8+64+3076 | 3148  | 25184 | 2518.4             |
| 8+64+3608 | 3680  | 29440 | 2944.0             |

## Chapter Four

### Simulation and discussion

#### 4.1 Simulation tools and programming language

##### 4.1.1 Verilog Hardware description language VHDL

VHDL (*VHSIC hardware description language*; VHSIC: *very-high-speed integrated circuit*) is a hardware description language used in electronic design automation to describe digital and mixed-signal systems such as field-programmable gate arrays and integrated circuits.

VHDL is a general-purpose language, and it does not require a simulator on which to run the code. There are many VHDL compilers, which build executable binaries. It can read and write files on the host computer, so a VHDL program written to generate another VHDL program incorporated in the design is developed. Because of this general-purpose nature, it is possible to use VHDL to write a *test bench* that verifies the functionality of the design using files on the host computer to define stimuli, interacts with the user, and compares results with those expected.

VHDL is not a case sensitive language. One can design hardware in a VHDL (such as Xilinx or Quartus) to produce the Register Transfer Level (RTL) schematic of the desired circuit. After that, the generated schematic can be verified using simulation software (such as WaveFrom Pro) which shows the waveforms of inputs and outputs of the circuit after generating the appropriate test

bench. To generate an appropriate test bench for a particular circuit or VHDL code, the inputs must be defined correctly. For example, for clock input, a loop process or an iterative statement is required [24].

The key advantage of VHDL when used for systems design is

3. It allows the behavior of the required system described (modeled) and verified (simulated) before synthesis tools translate the design into real hardware (gates and wires).

4. It allows the description of a concurrent system (many parts, each with its own sub-behavior, working together at the same time). VHDL is a Dataflow language, unlike procedural computing languages such as BASIC, C, and assembly code, which all run sequentially, one instruction at a time.

The following is a list of requirements for Ethernet switch. All requirements must function both in the simulator and on the NetFPGA hardware. Code must be compiled using the Xilinx tools.

- Learn media access control (MAC) source addresses
- Lookup MAC destination addresses via forwarding table
- Central processing unit (CPU) Queues
- Run at 125MHz in order to receive full credit.

#### **4.1.2 WaveFormer Pro.**

WaveFormer Pro is revolutionary new rapid-prototyping electronic design automation (EDA) tool that helps in faster design.



WaveFormer Pro automatically determine critical paths, verify timing margins and perform "what if" analysis to determine optimum clock speed [25].

WaveFormer Pro also specifies and analyzes system timing and performs Boolean level simulation without the need for schematics or simulation models. WaveFormer Pro also has the ability to import and annotate simulation and logic analyzer data, for publication quality design documentation.

#### **4.2 About the design**

This IP stack for an FPGA is a complex design because of the number of layers and the complexity of each that is required. It is limited to 10 Mb/s operations and designed for a full duplex switched network. It implements the lower layers of a standard TCP/IP stack.

Further implementation is needed to make it work specifically for a certain purpose. There is support to read and write to RAM from the PC via the parallel port as well, for debugging and tests purposes.

### 4.3 Files list and description

|                            |  |
|----------------------------|--|
| arp3.vhd                   | Handles ARP requests and replies that received, and manages the ARP table. |
| arpsnd.vhd                 | Sends ARP requests, replies and handles lookups.                           |
| crcgenerator.vhd           | Generate CRC checksums.  |
| ethernet.vhd               | Receives Ethernet frames off the network.                                  |
| ethernetsnd.vhd            | Puts Ethernet frames onto the network.                                     |
| globalconstants.vhd        | Contains the IP and MAC addresses of the device.                           |
| icmp.vhd                   | Responds to ICMP echo requests with an echo reply.                         |
| internet.vhd               | Handles IP datagrams and reassembly.                                       |
| internetsnd.vhd            | Sends IP datagrams and handles fragmentation.                              |
| memorymultiplexor-sv01.vhd | Allows either the PC or the IP stack to access RAM.                        |
| pctosraminterface-sv06.vhd | Handles the protocol for the communication between the PC and the board.   |

|  |   |
|--|---|
| sram512kleft16bit50mhz<br>readreq-sv05.vhd | Main interface to the left bank RAM,<br>controls reading and writing.   |
| sraminterfacewithpport-<br>sv01.vhd        | Top level for the RAM files.  |
| stack3.vhd                                 | Top level for the design, also contains<br>a RAM arbitrator to share RAM usage<br>between the different layers. |
| udp.vhd                                    | UDP Implementation file.  |
| stackpins.ucf                              | Constraints file for the design.  |
| cpldnet.vhd                                | CPLD reconfiguration to set the<br>outputs of the CPLDs connected to the<br>PHY. (Compile to an SVF file)       |
| cpldnetpins.ucf                            | Constraints file for the CPLD design.   |
| XSVRAMUtility.exe                          | Reading and writing to the onboard<br>RAM from the PC.  |

#### 4.4 Design overview

This VHDL design is a simple implementation of IP stack, but it is lacking a few transport level protocols (such as TCP). It does, however, provide a rich set of lower layers that should allow higher-level protocols (such as TCP) to be plugged easily on top of what already exists.

The protocol stack could be applied to XSV-300 board (version 1.0) from the XESS Corp. Using Foundation 3.1 and associated tools to make the design. The board makes use of an LXT970A Ethernet Transceiver physical layer (PHY) from Level One (an Intel company) to encode and decode the signals onto a 10Mb/s full duplex twisted pair switched network. (The chip is forced to 10 Mb/s operations and is changed by reprogramming the CPLD)

#### **4.5 Design structure**

The design is based upon the layered model of a TCP/IP stack, there is an Ethernet (Host to Network) layer that interacts with the PHY device, and communicates with the Address Resolution Protocol (ARP) and Internet layers. ARP (which holds the ARP lookup table), on the receive side, is considered to act as another protocol on the Internet layer parallel to IP, and on the transmit side it lies transparently between the Internet layer and the Ethernet layer, with a connection to ARP on the send side.

The main Internet layer protocol (IP) communicates with the transport layer and its associated protocols internet control message protocol (ICMP) and user datagram protocol (UDP), (see Figure 4.1). Each individual layer is split up into separate VHDL files and processes to simplify the design.

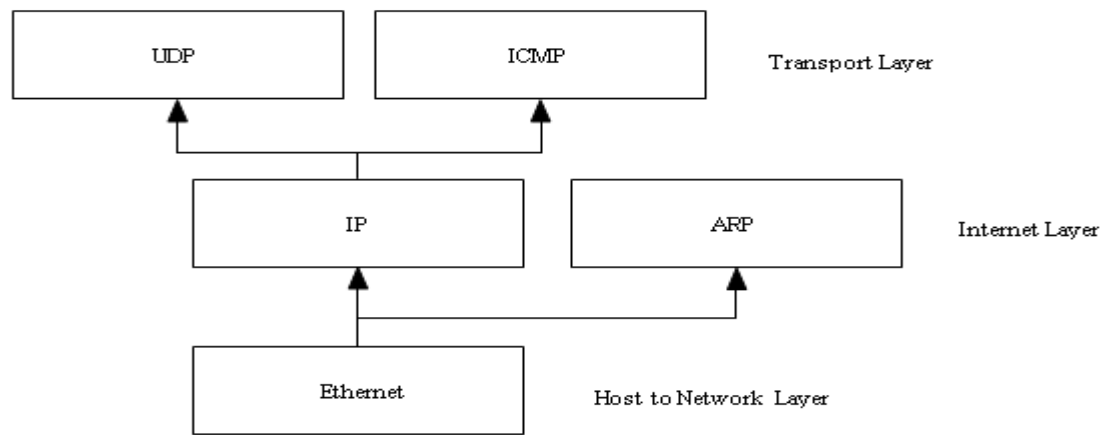


Figure 4.1: The layout of the receive side of the IP stack.

It is important to make a distinction between the send and receive sides of the stack. Coming back down the stack, only transport layer protocol that currently sends data (ICMP) lies at the top and is connected to the Internet Send layer, which lies directly below it. The Internet Send layer only has the IP available at this layer, as ARP handled differently on the send side. Directly below the Internet Send layer lays the ARP Send “pseudo” layer, which is transparent to the layers above and below it. The ARP Send layer also is connected to the ARP layer (see Figure 4.2). Below the ARP Send layer lays the Host to Network Send (Ethernet) layer, which is connected to the PHY and thus the physical network.

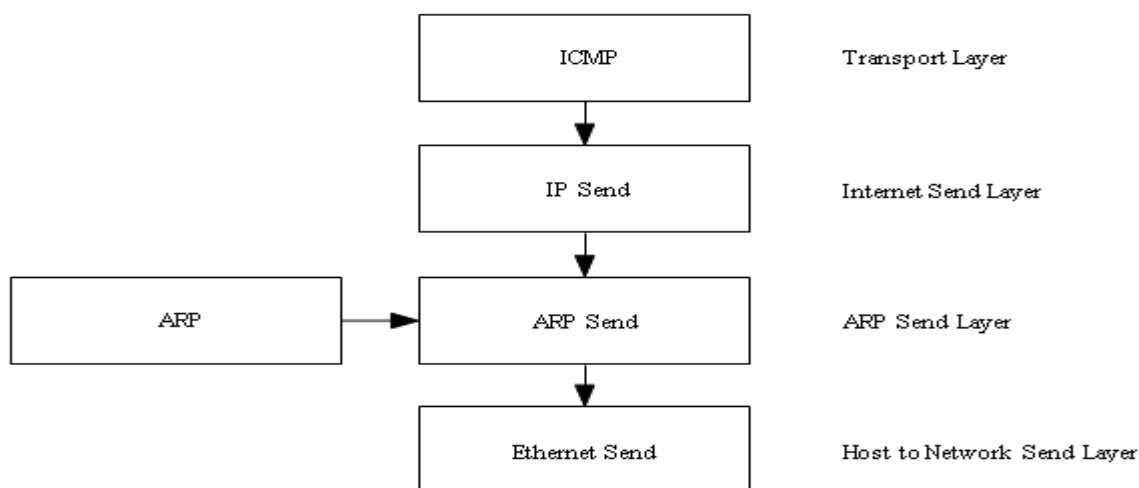


Figure 4.2: The layout of the send side of the IP stack.

Any more transport layer protocols should be able to plug in parallel to the ICMP layer, with little modification to the stack required. Each layer looks similar to a normal TCP/IP stack in concept and in behavior, with the slight exception of ARP and ARP Send.

## 4.6 Module Descriptions

### 4.6.1 Ethernet Receiver

The Ethernet receiver is connected to the PHY via a media independent interface (MII), with a 4 bit (nibble) wide data bus, a receive data valid signal, a receive error signal and a receive clock. When the PHY receives a frame, rx\_dv is asserted for the entirety of the frame. When in 10Mb operation, the PHY strips the first 60 bits off the preamble, and leaves the SFD as the first nibble received. On each rising edge of rx\_clk, a valid nibble of data will be valid to latch.

The Ethernet receiver ignores the start frame delimiter (SFD), and then processes the frame header. If the target MAC field in the

header of the frame received does not equal the one set in the global constants package or is not a broadcast, then the frame is ignored by jumping to a state and doing nothing until rx\_dv goes back low. The source MAC field of the header is ignored. If the Ethernet type field is not either 0800h or 0806h then the frame also is ignored, as the only frames accepted are IP and ARP messages.

After the Ethernet type is received, two signals are asserted to tell the above layers that a new frame is processed, and the type of message that is contained in the frame (either ARP or IP). After each byte (every second nibble) is received, another signal is asserted to tell the above layers that a new byte is received and the byte is passed up. The four CRC bytes are also passed to the next layer, and are ignored on the next layer. As each byte is received (including the frame header and data but not including the SFD) it is also passed to the CRC generator so the CRC is checked later.

Once rx\_dv goes low and the frame has finished receive, the CRC is checked, and the end frame signal is asserted. If the CRC is not zero (i.e. a correct CRC) then the valid signal gets zero, and the next layer should ignore the frame due to the invalid CRC. Otherwise, both the end frame and frame valid signals are asserted at the same time to tell the next layer that a valid frame has been fully received.

The Ethernet receiver passes the received frame as a byte stream. It saves slow RAM accesses and saves potential frames from being lost as when a very small frame follows a large one, the above layer could miss it, as the time to read and then store a large frame to somewhere else in RAM is longer than it takes to receive a very small frame. However, it does not check for frames of an invalid length (below 46 and above 1500 bytes for Ethernet), and will only operate at 10Mb/s.

#### **4.6.2 Ethernet Sender**

The Ethernet sender is connected to the PHY via a similar interface as the Ethernet receiver. From the PHY to the Ethernet sender there is tx\_clk, which is used as a clock to synchronize the data between the two devices. The other connections in the MII are a four-bit txdata.

The Ethernet sender can transmit either ARP or IP frames, as specified by the frame type input. Once it is instructed to send a frame, it will generate the preamble for the frame (7 bytes of 55h followed by 5Dh) and construct the header (based on the destination MAC input; the device MAC from the global constants package; and the frame type input to set the Ethernet type field). Once the header has been constructed, the Ethernet sender will read the data from RAM (the location depends on the type of frame) and send each byte read. Any frames smaller than 46 bytes are automatically padded to 46 bytes with whatever (pseudo random) data is currently contained in RAM.



All the data sent to the PHY a nibble at a time using the txdata bus, least significant nibble first. Each byte sent is also given to the CRC generator. Once all data has been sent, 4 consecutive bytes of zeros are sent to the CRC generator to correctly calculate the CRC for the frame. The 4 bytes of CRC are then sent at the end of the frame, least significant bit (LSB) first.

Once the frame is sent, a signal is asserted to the layer who requested it to inform them that the frame has been sent. The Ethernet sender is then not available for the time it would take to send 12 bytes over the network. The collision and carrier sense inputs ignored from the PHY for both the Ethernet and Ethernet Sender (the PHY does not make proper use of these in full duplex mode).

### **4.6.3     ARP**

The ARP layer handles ARP replies and ARP requests that are received, and manages the ARP table and receives ARP table lookups. It also tells the ARP Sender to generate ARP replies when needed.

When an Ethernet frame is received with an Ethernet type field of 0806h, then the ARP receives an ARP frame from the Ethernet Receiver via a byte stream. As ARP messages are always a standard length, the padding on the Ethernet frame and the CRC included in the byte stream can easily be ignored. If the ARP message is not meant for the IP specified in the global constants

package, if it is not a valid ARP message for Ethernet and IPv4, or if it is anything other than an ARP request or reply, then the message is ignored. If it is an ARP request, then the sender is added to the ARP table and the ARP sender is informed to send out a reply and the ARP table waits until notification is received that the reply is sent. If it is an ARP reply, then the sender is added to the ARP table.

Because of the two-way handshake between the ARP and the ARP Sender, a possible lockup could occur. Consider the following situation:

- The ARP Sender sends out an ARP request (request A)
- An ARP request arrives from someone else (request B)
- ARP needs the ARP Sender to say that it has sent the reply to request B before it can receive more requests and replies
- The ARP Sender waits for ARP to receive the reply to request A and adds it to the ARP table before it can send out more frames to the Ethernet Sender

Both ARP and the ARP Sender will now be waiting for each other in a lockup situation, as each requires a signal from the other to continue. These results in all incoming ARP messages are missed and no frames are sent until the ARP Sender times out and resets. To avoid this, the signals to tell the ARP Sender to send an ARP reply are set and reset in a separate process, leaving the main ARP

process free to receive further replies and set the ARP table until the ARP Sender then becomes free to send the reply. Should a second request arrive before the required ARP reply comes in, then the first request (request B) will now be ignored and the signals will be set to tell the ARP Sender to send a reply to the second request when it becomes free.

The ARP table is a two-entry first in first out table. Lookups are performed on the table by setting the lookup IP input. If the IP is in the ARP table, the entry valid output is asserted and the corresponding MAC address outputted. When an entry is added to the table, it is first checked to see if that entry already exists. If it does exist, then the entry is moved to the newest entry position in the table and updated. If the entry is not previously in the table, then it is added into the newest entry position, the previous newest entry is moved into the older entry position and the older entry is discarded.

#### **4.6.4 ARP Sender**

The ARP sender lies transparently between the Ethernet Send and the Internet Send layers and constructs ARP requests and replies; handles ARP table lookups; and passes frames from the Internet Send layer to the Ethernet Send layer. The ARP Sender will always be unavailable while the Ethernet Sender is sending a frame.

The ARP Sender will construct ARP replies when requested to do so by the ARP layer. The target IP is obtained from the input from the ARP layer to the ARP Sender, and the target MAC address is obtained by performing a lookup in the ARP table by using the input IP. Once the ARP reply has been constructed and stored to location 0 in RAM then the ARP Sender will inform the Ethernet Send layer to send a frame. Once it is sent, the ARP Sender will inform the ARP layer that the reply has been sent.

If the ARP Sender receives a frame from the Internet Sender, then it will perform a lookup on the destination IP (received from the Internet Send layer) in the ARP table. If it does not have a valid entry for the destination IP, then it will generate an ARP request for that IP into location 0 in RAM, and inform the Ethernet Sender to send the request. Then a timeout counter is started. It has 21.5 seconds in which to receive the ARP reply and put a valid entry for that IP into the ARP table or the counter will overflow, and it will inform the above Internet send layer that the frame is sent and return to an idle state, with the frame not sent. If the entry was originally valid or became valid after the ARP request is sent, then it will inform the Ethernet Sender to send the frame to the looked up MAC address. Once the frame is sent, it informs the Internet Send layer that the frame has been sent.

### **4.6.5 Internet**

The Internet layer receives datagrams from the Ethernet Receiver, strips the IP header, works out what done with it, and if no more fragments are coming, then passes the datagram to the required protocol on the above transport layer.

Each datagram is received from the Ethernet Receiver via a byte stream. The amount of bytes expected are pulled from the IP datagram length field in the IP header, so the CRC that is sent at the end of the byte stream can easily be ignored. Different signals are asserted to the Internet layer when a new frame arrives, when each consecutive byte arrives, when the frame is finished and if the frame is valid (i.e. it has the correct CRC). ARP receives the byte stream from the Ethernet Receiver in the same way.

The length of the header of the datagram is taken from the first byte received from the IP header, and any options that may be included in the IP header are then ignored. The protocol, length, identification and source IP fields of the header are latched. If the destination IP field is not the same as the IP in the global constants package or is not a broadcast IP; the checksum on the header is not correct; or the datagram is not for IPv4 then the datagram is dropped.

When a new datagram is received, it is stored into the first available buffer (the two IP buffers at locations 10000h and 20000h in memory and are numbered 0 and 1 respectively). Reassembly of incoming packets is restricted to packets that arrive in network order. Out of order packets are dropped as will duplicate packets for data that arrived previously. The IP layer always tries to write to buffer zero first, and will only write to buffer 1 if buffer 0 contains a partially received datagram. When a buffer is utilized, the source IP, identification number and protocol from the IP header are stored and all three must match the incoming frame to continue using the buffer. Every time a new valid fragment is received, a timeout counter is restarted, and when this reaches its maximum value, the buffer is freed again for a new incoming packet. Any valid fragment size is received and reassembled properly.

#### **4.6.6 Internet Sender**

The Internet Sender handles the creation of IP datagrams from Transport Protocol Data Units (TPDUs) and the TDPUs are fragmented if they are too large. Once it forms the correct header and creates the datagram, it passes the nearly formed IP datagram down to the ARP Sender if it is available.

When the Internet Sender receives a new TPDU to send, it first creates the header, getting the destination IP, and protocol and length fields from its inputs. The identification field is obtained from a counter that increments approximately every 20us, and the source

IP is obtained from the global constants package. The options in the IP header are never used. When the header is created, the header checksum is created and added to it. The data is then pulled from RAM (the location depends on the address offset input - this facilitates the use of different buffers for different protocols) and stored in location 800h. The ARP Sender then is informed that a new frame is ready for sending, and the Internet Sender will then do nothing until it receives from the ARP Sender, that the frame has been sent.

Fragmentation occurs when the datagram that is required to be sent is above 1480 bytes. This is because 1500 bytes is the maximum frame size allowable over Ethernet, and that has to include a 20-byte IP header. When a packet needs to be fragmented, 1044-byte fragments (1024 bytes of data, 20 bytes of IP header) are sent until a final fragment with less than 1480 bytes of data is sent. Fragments are sent in order as soon as they pass out of the Ethernet layer. The Internet Sender will not be able to send more data until it has finished transmitting the entire TPDU. As the Internet Sender only sends 1024 bytes of data for every fragment except the last, it will send more packets than is usually required, but this greatly simplifies the hardware required to implement fragmentation.

#### **4.6.7 ICMP**

The ICMP protocol is implemented only to respond to an echo request. Any other ICMP message is ignored. ICMP can respond to a valid ping containing between 0 and  $2^{16} - 8$  bytes of data.

If an IP datagram is received with the protocol field set to ICMP and the ICMP code and type fields in the ICMP header are for an echo request, then ICMP will strip the ICMP headers and recreate them into an echo reply format. It will get the ICMP data out of one of the Internet buffers depending on the value of the buffer select input. It will then store the created ICMP echo reply and optional data to location 40000h in RAM and inform the Internet Sender to create an IP datagram to send the echo reply.

#### **4.6.8 UDP**

This simple implementation of UDP catches UDP messages on a specified port and stores them to location 30000h in RAM. Once an IP datagram arrives with the protocol field set to UDP, then the UDP process will remove the UDP headers and check the port, and then move the data to the new location in RAM. The UDP checksum is ignored (the pseudo header is not recreated).

### **4.7 Memory Map**

The IP stack only uses the left bank of memory to hold the data used by the stack. Below in table 4.1 is a memory map of what portions of the RAM are used and what parts are free. The free parts of the memory are available to hold more buffers. Each address is considered 8 bits (one byte) - the other 8 bits are ignored.



Table 4.1: Memory Map.

| Memory Range  | Memory Usage                              |
|---------------|---|
| 00000 - 007FF | ARP Send buffer – holds ARP replies /     |
| 00800 – 01000 | IP Sender buffer – holds IP frames to be  |
| 01001 - 0FFFF | Free                                      |
| 10000 - 1FFFF | IP Receive buffer 0 – holds reconstructed |
| 20000 - 2FFFF | IP Receive buffer 1 – holds reconstructed |
| 30000 - 3FFFF | UDP Receive buffer – holds received UDP   |
| 40000 - 4FFFF | ICMP Reply buffer – holds created echo    |
| 50000 - 5FFFF | Free                                      |
| 60000 - 6FFFF | Free                                      |
| 70000 - 7FFFF | Free                                      |

## Chapter Five

### Conclusion and future works

#### 5.1 Design issues

##### 5.1.1 Design limitations

There are several key limitations to the design of the IP stack, most of which are due to the limited amount of hardware, RAM and buffer space available on an FPGA (assuming the IP stack shouldn't take up over half the FPGA in size). Following are some of the main limitations:

- Only a small number of connections are handled at once – if more than two other addresses try talking to the board simultaneously with all of them sending data to the board, a lot of the time is spent sending off and processing ARP requests and replies while the ARP table has only two entries. The ARP table can easily grow in size if needed, but it needs an extra 80 flip-flops and more logic for table lookups for every entry added.
- The lack of buffer space also creates problems if multiple datagrams are received and reassembled at once, or if the transport layer protocols are busy then datagrams will have to be dropped, as no IP buffers will be free unless more IP buffers or transport layer buffers are allocated. Increasing the number of buffers results in a large increase of memory usage and logic needed for controlling them. Due to the large timeouts on the IP

- buffers and the ARP Sender, any packet loss can result in several modules of the system tied up for a long period. However, if these timeouts are shortened, then late arriving packets can cause other lockups or cause packets not to be sent at all.

### **5.1.2 Interfacing and enhancing the design**

The IP stack is designed intentionally not just as a stand-alone design. As such it would be useful for algorithms that require a fast transfer of data, whose parallel port is not provided. However, the IP stack is a lot more demanding of hardware and RAM and is more complex than the parallel port interface. Preferably, it could simply receive and store data in RAM, and inform the application when new data has arrived. If the board is configured to support partial reconfiguration of the Virtex FPGA, then perhaps reconfiguration over the network is a (non-trivial) application.

To enhance the design, many other protocols such as TCP, RARP and support for 802.2 frames are added. However, the IP stack is written with the intention that a multiplexer (much like the RAM arbitrator) would be used between the transport and the Internet Send layers. This would keep the send signals of the transport layer protocol asserted until it is informed from the Internet Send layer that the datagram has been sent.

This allows transport layer protocols to be added without the Internet Send layer caring, facilitating the addition of higher-level protocols.

Several other enhancements are possible – for example, when receiving and storing data, only 8 bits of the 16 bits of data of each RAM address are used, and although more logic would be required, a lot of RAM is saved if the full 16 bits of data were used. Due to a lack of time, the designs have not been optimized (in many of the files, logic can be reduced by the addition of a few states, and the two huge case statements in arpsnd.vhd can be optimized into one by playing with the nextState signals).

## **5.2 Conclusion**

There are too many proposed solutions that could be an AFDX backup network, but in order to have Ethernet interconnection elements, bounded, predictable, deterministic, simple, light in weight and low cost of material because of using simple hub instead of using special TTEthernet switch. The time triggered Ethernet (TTE) is much better than the other protocols.

TTE can be implemented with simple Ethernet hub either using TDMA mechanism with bounded cycle, or with microcontroller which makes the traffic policy as a watching dog to have a predictable cycle with no collisions or lost frames.

The reason of studying the time triggered Ethernet is the fact that it has advantages over other real time protocols such as:

1- There is no need for a master for communication but only for synchronization.

2- It is the best usage of the bandwidth in case of a heavy traffic.

4- Predictable real-time response times within the network.

5- Can be implemented with simple Ethernet hub, which is much cheaper than TTE switch.

### **5.3 Future works**

The IP stack for an FPGA is a complex design because of the number of layers and the complexity of each that is required, it is limited to 10Mb/s and for full duplex switched network.

Further implementation of the following modifications is possible:

1. Support the 100Mb/s. For higher speed and best usage of the bandwidth taking into account the number of pairs for transmission and the signal frequency and encoding type.

2. Support other protocols such as reverse address resolution protocol (RARP). To maintain a database of mappings of Link Layer addresses to their respective protocol addresses.

3. Support logical link control (LLC) for multiplexing protocols transmitted over the MAC layer (when transmitting) and decoding them (when receiving) and providing flow and error control.

4. Support TCP for controlling segment size, flow control, the rate of data exchange, and network traffic congestion

## References

- [01] Astrit Ademaj, “**Integration of predictable and flexible In-vehicle communication using time triggered Ethernet**”, Vienna University of Technology April 2006.
- [02] Ademaj, A.Kopetz, “**IEEE 1588, IEEE standard for information technology for clock synchronization**”. Vienna Univ. of Technol, Vienna, Publication Date: 1-3 Oct. 2007 On page(s): 41-43.
- [03] Abdallah Alkhasawneh, “**AFDX backup network**” master thesis, AIRBUS Toulouse France 2009.
- [04] Hadriel Kaplan & Bob Noseworthy, “**The Ethernets evolution from 10 to 10,000 Mbps**”, Atlanta 2000.
- [05] [Http://iec.ch](http://iec.ch) (IEC 61784-2) , Industrial communication networks - Profiles - Part 3-2: Functional safety fieldbuses - Additional specifications for CPF 2 Publication Date: 2007-12-14.
- [06] [Http://ieeexplore.ieee.org](http://ieeexplore.ieee.org) ( IEEE 802.3 2008) , IEEE standard for information technology part 3 local and metropolitan networks specific requirements carrier sense multiple access with collision detection CSMA/CD access method and physical layer specifications.

- [07] [Http://ieeexplore.ieee.org](http://ieeexplore.ieee.org) ( **IEEE 802.5 1998** ) , token ring access methodology and physical layer specifications.
- [08] Klaus Steinhammer, Astrit Ademaj, “**Hardware implementation for Time triggered Ethernet**”, 2006.
- [09] Klaus Steinhammer, Hermann Kopetz, “**Time triggered Ethernet**”, Junior Scientist Conference - JSC 2006.
- [10] Petr grillinger, “**A time triggered Ethernet switch**”, Vienna 2006.
- [11] Petr grillinger, “**The time triggered Ethernet design**”, Vienna 2005.
- [12] Petr grillinger, “**Software implementation of time triggered Ethernet controller**”, Vienna 2007.
- [13] Vorgelegt von, “**Advanced gateways in automotive applications**”. Berlin 2008.
- [14] <http://altera.com/end-markets/industrial/automation/ethernet/protocols/ind-fieldbuses.html>. Accessed 22/01/2010.



**[15]** [http://en.wikipedia.org/wiki/Ethernet\\_hub](http://en.wikipedia.org/wiki/Ethernet_hub). Accessed 10/02/2010.

**[16]** [http://en.wikipedia.org/wiki/Fast\\_Ethernet](http://en.wikipedia.org/wiki/Fast_Ethernet). Accessed 10/02/2010.

**[17]** [http://en.wikipedia.org/wiki/Field-programmable\\_gate\\_array](http://en.wikipedia.org/wiki/Field-programmable_gate_array). Accessed 02/02/2010.

**[18]** [http://en.wikipedia.org/wiki/Line\\_code](http://en.wikipedia.org/wiki/Line_code). Accessed 18/01/2010.

**[19]** [http://en.wikipedia.org/wiki/Power\\_over\\_Ethernet](http://en.wikipedia.org/wiki/Power_over_Ethernet). Accessed 01/11/2009.

**[20]** <http://en.wikipedia.org/wiki/TTEthernet>. Accessed 10/02/2010.

**[21]** <http://iec.ch> ( IEC 61784-2 ) , Industrial communication networks - Profiles - Part 3-2. Accessed 01/11/2009.

**[22]** <http://ieeexplore.ieee.org>. Accessed 01/11/2009.

**[23]**

<http://informationphilosopher.com/freedom/determinism.html>.

Accessed 22/01/2010.

**[24]** [http://seas.upenn.edu/~ese201/vhdl/vhdl\\_primer.html](http://seas.upenn.edu/~ese201/vhdl/vhdl_primer.html).

Accessed 18/01/2010.

**[25]** [http://syncad.com/waveformer\\_waveform\\_editor.htm](http://syncad.com/waveformer_waveform_editor.htm).

Accessed 09/09/2009.

**[26]** [http://stason.org/TULARC/networking/lans-ethernet/3-11-](http://stason.org/TULARC/networking/lans-ethernet/3-11-What-is-propagation-delay-Ethernet-Physical-Layer.html)

[What-is-propagation-delay-Ethernet-Physical-Layer.html](http://stason.org/TULARC/networking/lans-ethernet/3-11-What-is-propagation-delay-Ethernet-Physical-Layer.html).

Accessed 09/09/2009.

