

University of the West of England

BEng Individual Project

**The ECSS SpaceWire Communications Protocol implemented into**

**IEEE 802.3 Power over Ethernet**

A report submitted in fulfilment of the requirements for the degree

of Electronic Engineering BEng (Hons)

in the

Engineering Design and Mathematics Department

Faculty of Environment and Technology

Author: Jack Martin

Supervisor: Yaseen Zaidi

# Abstract

Engineering Design and Mathematics Department

Faculty of Environment and Technology

Electronic Engineering BEng (Hons)

*The ECSS SpaceWire Communications Protocol implemented into*

*IEEE 802.3 Power over Ethernet*

By Jack Martin

The operational capability and sustainability of Spacecraft is constrained by the physical system architecture, highly customised monolithic design and the limited or no availability of servicing and maintenance. The research of cost effective, high performing, scalable and flexible solutions is essential to the future of the European Space Agency to spearhead the commercial space industry strategy. Project MOSAR is an EU funded initiative that will aim to raise the degree of modularity of space systems by an order of magnitude with respect to current space industry standards.

The modularisation of space mission systems can be supported by using existing and widely adopted industry standards for key functionalities of mission equipment. This report details the feasibility investigation of the implementation of the ECSS SpaceWire Protocol into IEEE 802.3 Power over Ethernet, to enable the transmission of data and power between spacecraft sub-systems. A Literature Review was conducted of the SpaceWire and IEEE standards to ascertain the technical requirements and

**Contents**

Abstract ii

1. Introduction 1

1.1 Background 1

1.2 Aims and Objectives 2

1.3 Deliverables 2

1.4 Project Management & Artefacts 2

1.4.1 Project Initiation Forms 2

1.4.2 Meeting Minutes and Record of Decisions 3

1.4.3 Project Dependencies 3

1.4.4 Configuration Management 3

1.4.5 Resources 3

2. Documentation Review 4

2.1 OSI Concept Model 4

2.2 ECSS SpaceWire 5

2.2.1 ECSS SpaceWire Overview 5

2.2.2 ECSS SpaceWire Protocol Stack 6

2.2.3 Physical Layer Overview 7

2.2.4 Data Link Layer Overview 8

2.2.5 Network Layer Overview 9

2.2.6 SpaceWire Packets 10

2.2.7 SpaceWire Packet Addressing 10

2.2.8 Broadcast and Time-Codes 10

2.2.9 Management Information Base 11

2.3 IEEE 802.3 Ethernet 12

2.3.1 IEEE 802.3 Overview 12

2.3.2 Physical Layer Overview 12

2.3.3 Data-Link Layer Overview 13

2.3.4 IEEE 802.3 Power over Ethernet Overview 16

2.4 PoE and SpaceWire Protocol Stack Implementation 17

3. Embedded Systems Development 19

3.1 Hardware 19

3.1.1 System Architecture 19

3.2 Software 19

3.2.1 UML Diagram 19

4. Functional Testing and Results 20

4.1 Testing Plans 20

4.2 Results and Analysis 20

5. Conclusion 21

5.1 Results and Analysis 21

5.2 Recommendations 21

5.3 Summary 21

References 22

Annexes 23

**Figures**

[**Figure 1** –Project MOSAR Concept of Modular Spacecraft Assembly 1](#_Toc56190918)

[**Figure 2** – OSI Concept Model 4](#_Toc56190919)

[**Figure 3** - Example SpaceWire Application 5](#_Toc56190920)

[**Figure 4** - SpaceWire Protocol Stack compared to OSI Concept Model Layers 6](#_Toc56190921)

[**Figure 5** - SpaceWire Protocol Stack 7](#_Toc56190922)

[**Figure 6** – SpaceWire Type A Connector 7](#_Toc56190923)

[**Figure 7** - Data Strobing Coding Scheme 8](#_Toc56190924)

[**Figure 8** - SpaceWire Data and Control Characters and Control Codes 9](#_Toc56190925)

[**Figure 9** - SpaceWire Packet Format 10](#_Toc56190926)

[**Figure 10** - SpaceWire Time-Code 11](#_Toc56190927)

[**Figure 11** - 802.3 Ethernet OSI Concept Model Comparison 12](#_Toc56190928)

[**Figure 12** - IEEE 802.3 RJ45 Twisted Pair Connector and Pinout 13](#_Toc56190929)

[**Figure 13** - 802.3 Ethernet MAC Frame and Packet Specification 13](#_Toc56190930)

[**Figure 14 -** Example Manchester Encoding Waveforms 15](#_Toc56190931)

[**Figure 15** - Non-Return Zero Inversion Example 15](#_Toc56190932)

[**Figure 16** - IEEE 802.3u 4B/5B Encoding Scheme 16](#_Toc56190933)

[**Figure 17** - IEEE 802.3 PoE Standards Summary 16](#_Toc56190934)

[**Figure 18** - Example PoE System 17](#_Toc56190935)

**Tables**

[**Table 1** - PoE and SpaceWire Data-Link Layer Comparison 17](#_Toc56190910)

[**Table 2** - ECSS SpaceWire and IEEE 802.3 PoE Full Protocol Stack Implementation 18](#_Toc56190911)

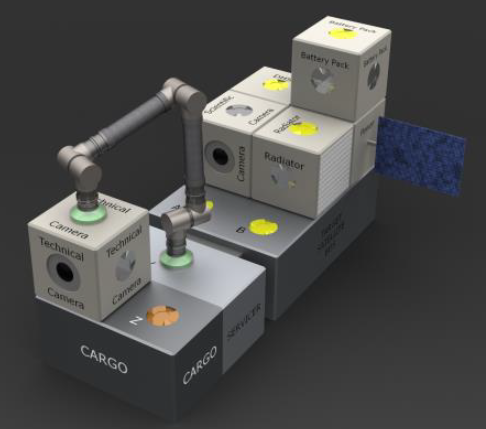
# Introduction

## Background

#### The operational capability and sustainability of current spacecraft is constrained by their inherent physical system architecture, highly customised monolithic design and the limited or no availability of servicing and maintenance. The research of cost effective, high performing, scalable and flexible solutions is essential to the future of the European Space Agency to spearhead the commercial space industry strategy.

#### Project MOSAR is an EU funded initiative made of international consortium of 9 partners that will aim to raise the degree of modularity of space systems by an order of magnitude with respect to current space industry standards. Thales Alenia Space UK is a world leader in space systems designs, integration and manufacturing with advanced space engineering facilities located locally to the University of West of England (UWE), acting as the leader on the design of satellite propulsions system, satellite sub-systems and system design studies.

#### It is desirable that an existing standardised full protocol stack could provide the underpinning inter-communications between future spacecraft sub-systems as well as provide a common architecture to also distribute power; a solution to support the project MOSAR objective of modularising spacecraft sub-systems.



**Figure 1** –Project MOSAR Concept of Modular Spacecraft Assembly

## Aims and Objectives

#### The aim of the project was to conduct a feasibility study of the given topic; “The ECSS SpaceWire Communications Protocol implemented into IEEE 802.3 Power over Ethernet (PoE)”, theoretically, analytically and through functional testing. From the described aim, objectives for this project were identified as below;

#### Demonstrate the concept of utilising existing engineering standards and topologies whilst respecting existing space industry standards for the purpose of providing key functional requirements of spacecraft sub-systems, in support of Project MOSAR.

#### Demonstrate the feasibility study and project aim, through Literature and Documentation Review, Hardware and Software realisation and Functional Testing and Analysis.

#### Identify the readiness level of the proposed feasibility study and provide the basis of the extended recommended work.

#### Demonstrate evidence against all Project Module Specification (UFMFX8-30-3) Learning Outcomes.

#### Activities associated with the project are to be identified within a Risk Assessment and mitigated to Tolerable and As Low as Reasonably Practicable (ALARP).

## Deliverables

#### The project was to submit the following deliverables;

#### Interim Research Proposal

#### BEng Individual Project Dissertation Report (this document)

#### BEng Individual Project Poster/Presentation

## Project Management & Artefacts

### Project Initiation Forms

#### It was of the responsibility of the student (document author) to ensure that the project is managed appropriately, ensuring that the following information listed below is captured and regularly updated; if required the issues are to be to notified to the Project Supervisor.

#### Baselined Project Schedule/Gantt Chart

#### Project Resources Form

#### Health and Safety Risk Assessment and ALARP Justification

#### Ethics Checklist

#### Non-Disclosure Agreements (where necessary)

#### Security Awareness Form

#### A Baselined Project Schedule [1] was generated at Project Initiation to ensure the key activities, milestones and critical path could be identified, to plan the project accordingly with the given time limitations and resources. A Health and Safety Risk Assessment was conducted at the initiation of the project and regularly updated as appropriate, shown in Annex A.

#### It was agreed between the Student and Project Supervisor that no ethical issues could arise from the area of research and activities associated with this project.

### Meeting Minutes and Record of Decisions

#### Project Meeting Minutes [2] were generated by the student throughout the project, capturing a Record of Decisions, Discussion, Attendees and Actions. These were captured within the on-line logbook on Microsoft OneNote with a supporting Action Tracker.

### Project Dependencies

Only one dependency was identified for the project, which was for the resources (hardware) to enable functional testing aspects of the project to be supplied by UWE.

### Configuration Management

Configuration Management for the project was essential for the control of source code that was to be written for the hardware devices, to ensure that the latest versions of software were appropriately stored and could not be mistakenly altered which would inhibit progress. The chosen tool for source-code control throughout the project was GitHub; a Github Repository can be created within a Project File Folder that tracks the changes and provides version control.

### Resources

Where identified the resources required to achieve the project which incur additional processes or procurement were identified within the Project Resources Form (Appendix X). The following resources were utilised throughout the project which were already readily available;

#### Microsoft Visual Studio Code

#### PlatformIO IDE

#### Arduino IDE

#### Microsoft Office Software

#### Github

#### UWE Library (Standards)

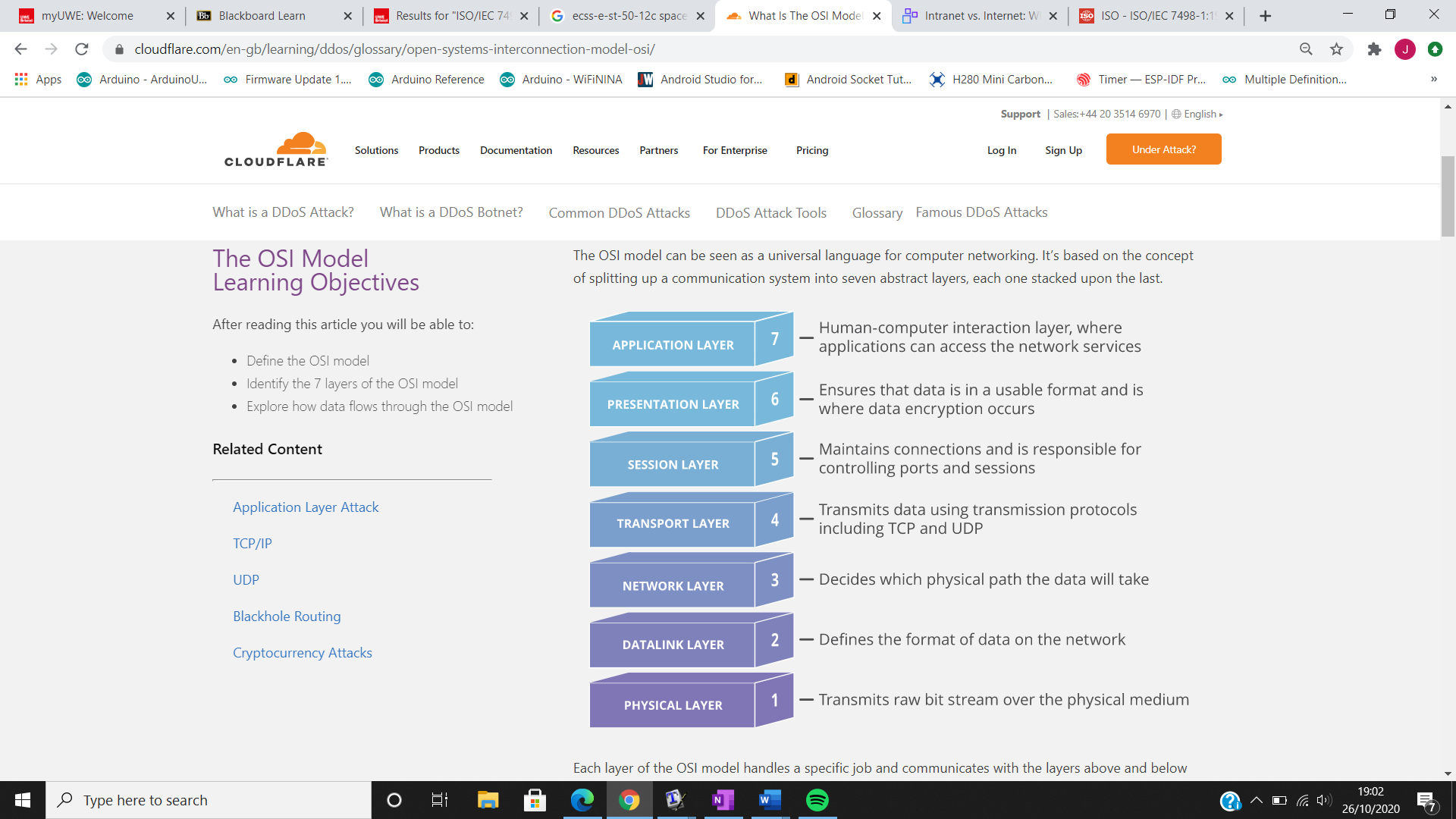
#### PuTTY

#### ESP-IDF

# Documentation Review

## OSI Concept Model

The Open Systems Interconnection (OSI) Protocol is described within Reference [3] as a theoretical model that enables diverse communications systems to communicate using standard protocols. It is based on the concept of splitting up a communications system into seven abstract layers, each one stacked up upon the last, compartmentalising the data to show how it is handled at each stage of a communications protocol. The Basic Model is defined within “ISO/IEC 7498-1 The Basic Model”. The seven layers of a communications protocol as defined by International Organisation of Standardisation is shown in Figure2.



#### **Figure 2** – OSI Concept Model

To achieve the aim of this project, it was necessary to recognise the requirements of what is regarded as a full communications protocol stack (OSI Model), to allow the transmission and receiving of information between communications systems.

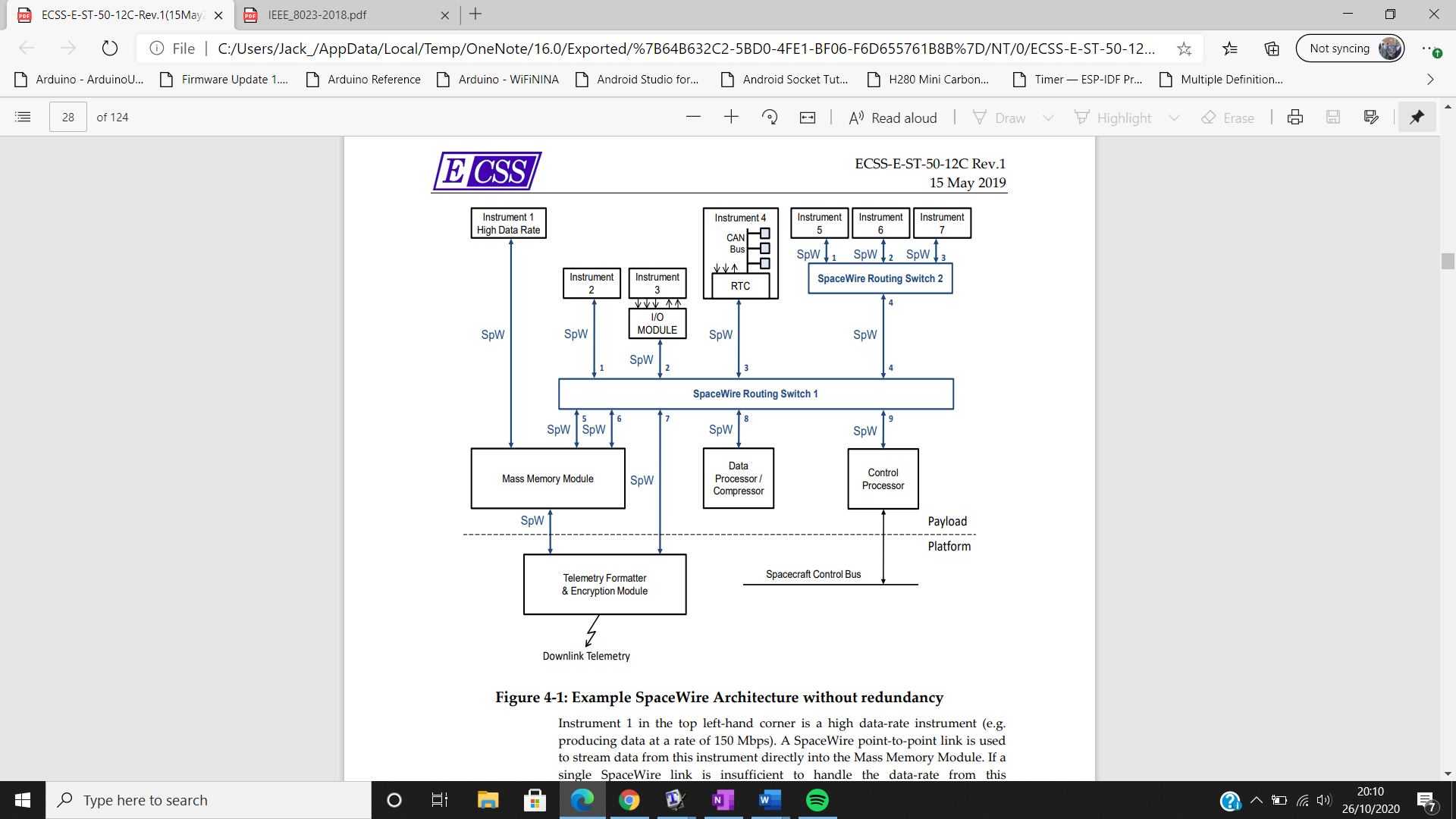
## ECSS SpaceWire

### ECSS SpaceWire Overview

#### The European Cooperation for Space Standardization (ECSS) SpaceWire is defined within the ECSS-E-ST-50-12C Rev.1 standard [4]. The rationale behind SpaceWire was to standardise the proprietary and ad-hoc approach adopted by space equipment manufacturers for interunit communications. This led to several different types of communication links being used on spacecraft, which subsequently increased the cost and time required for spacecraft integration and testing.

#### The use of the SpaceWire standard ensures that all equipment used on spacecraft is compatible at both the component and sub-system levels, allowing greater re-usability and reliability whilst also reducing development costs. The typical application of a SpaceWire based inter-communications network is shown below in Figure 3.

#### SpaceWire provides a 2 – 200 Mbps, bi-directional, full-duplex data link that connects together SpaceWire enabled equipment. The SpaceWire implementation can either be a point to point link or the use of a network routing switch. The typical SpaceWire application is shown in Figure 3, the inter-connection of several SpaceWire enabled equipment through the use of SpaceWire compliant routing switches.



**Figure 3** - Example SpaceWire Application

### ECSS SpaceWire Protocol Stack

#### The SpaceWire Protocol Stack is based upon the OSI Concept Model (Figure 2) to establish the methodology for the network communications between sub-systems and components within a spacecraft mission system. SpaceWire mandates the use of four layers of the OSI concept model, notably being the Physical, Data Link, Network and Application Layer. SpaceWire also specifies the use of a Management Information Base that configures and determines the status of the SpaceWire network. The SpaceWire Protocol Stack is shown diagrammatically in reference to the OSI Concept Model in Figure 4**.**

#### The data flow through the SpaceWire Protocol Stack is shown in Figure 5. The model displays how transmitted data from the application layer is parsed and formatted down to the individual bit-stream to be transmitted over the physical layer. For receiving data this process is reversed, showing the parsing and formatting from the received bit-stream useable characters/data for the end-user (person or hardware).

#### Machine generated alternative text: SpaceWire user Application Time-codes Network Layer Distributed Interrupts OSI Network Layer Packets Broadcast Codes Data Link N-Chars OSI Data Link Layer Encoding Layer Physical OSI Physical Layer Layer Bit Stream Figure A-2: Comparison of SpaceWire layers to OSI layers

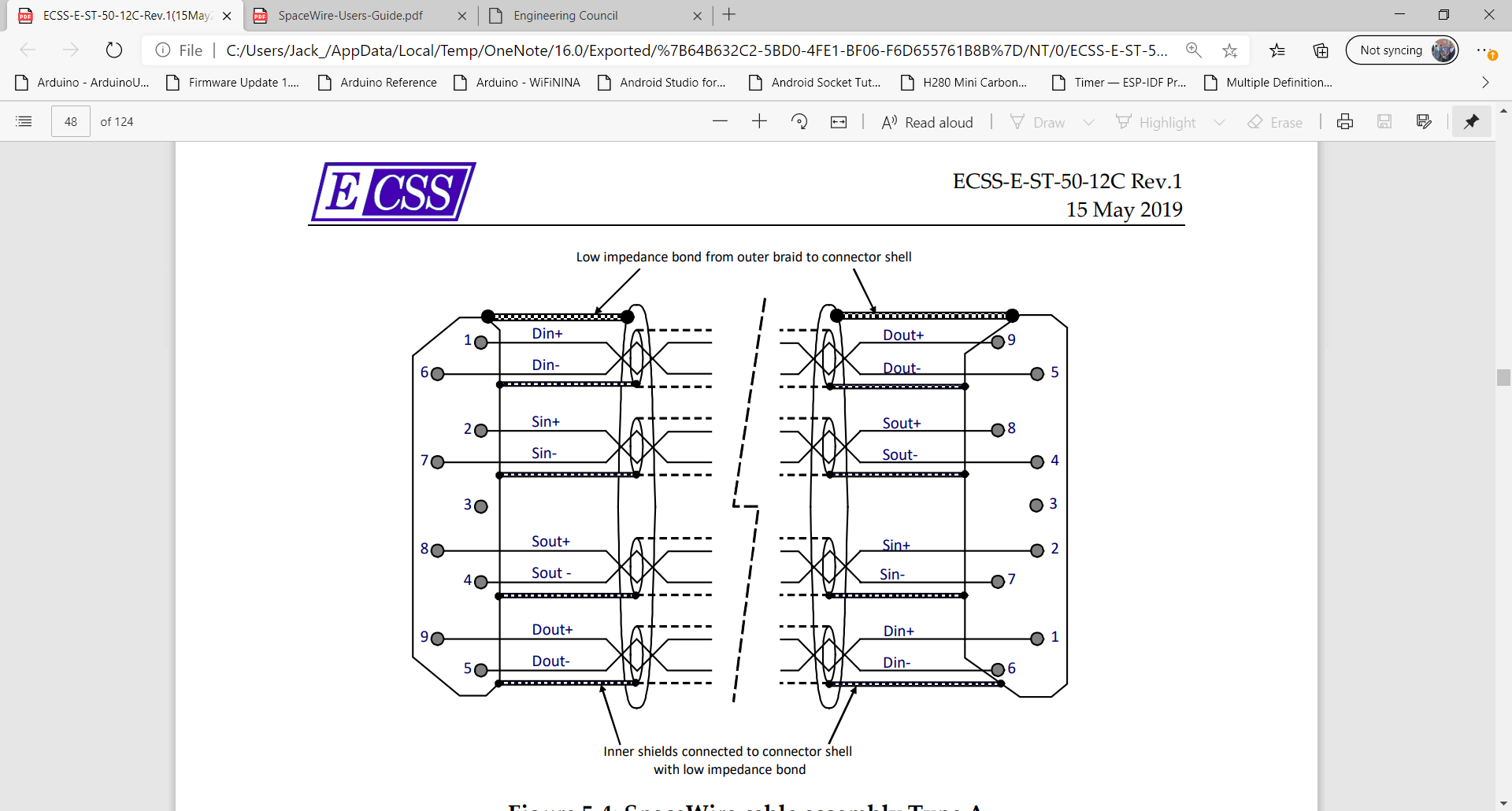
**Figure 4** - SpaceWire Protocol Stack compared to OSI Concept Model Layers

Machine generated alternative text:
nformation 
Application 
Time-codes Interrupts 
Network Layer 
Broadcast 
Data Link Layer 
Receive 
i 
TX Chars 
Control 
Enable 
RX chars 
& Control gotNuII 
Encoding Layer 
Data. Strobe Tx 
Physical Layer 
Data-strobe Rx 
switches 
Time-codes 
Link State machine 
Link error 
Encoding of characters 
Data •Strobe Encoding 
Cables 
Figure 5-1: SpaceWire protocol stack 

**Figure 5** - SpaceWire Protocol Stack

### Physical Layer Overview

The SpaceWire Physical Layer states the cables, connectors and line drivers for transmitting and receiving SpaceWire data and strobe signals over the physical medium. The physical connector consists of a nine-pin micro-miniature D Type Plug which encapsulates 4 twisted pairs of differential signalling wires. The SpaceWire Type A connector is shown in Figure 6.



**Figure 6** – SpaceWire Type A Connector

#### Figure 6 shows the use of 4 differential signal pairs that allows bi-directional data transmission. The signalling wires provide the physical medium for Low Voltage Differential Signalling (LVDS) for SpaceWire Data and Data Strobing Signals. The strobing signal is used to encode and recover the transmission clock signal which is simply recovered by XORing the Data and Strobe Lines together. The coding scheme is shown in Figure 7, illustrating how the transmission clock signal is encoded into the data and strobe signals.

#### 

#### **Figure 7** - Data Strobing Coding Scheme

### Data Link Layer Overview

From Figure 4 the Data-Link and Encoding Layer within the SpaceWire protocol stack can be considered just the Data-Link Layer in reference to the OSI Concept Model (Figure 2). The Data-Link Layer within the SpaceWire standard specifies how communication is established across a link, how the information flow is controlled, how N-chars and broadcast codes are sent and received, and how communications is re-established across the link after an error has occurred. The Encoding Layer within the SpaceWire standard specifies the encoding/decoding of characters into symbols, the serialisation/de-serialisation of the encoded symbols into a bit stream, and the data-strobe encoding/decoding of the serial bit stream.

SpaceWire has two types of characters which are the data and control characters. Data characters hold 8-bit values (char/byte) and also a parity bit and data-control flag, therefore containing 10 bits per character. Control characters hold a 2-bit control code in addition to a parity bit and data-control flag. In addition to this control codes are also used for Link Control and Time-Code distribution over a SpaceWire network. SpaceWire data characters, control characters and control codes are shown in Figure 8. The only further encoding that occurs is the Data Strobing that is driven over the physical layer as described in 2.2.3.

p 
Parity ad 
FCT 
EEP 
ESC 
Normal End Of padet 
padet 
nnnnnonnnannnn 
0123 4567 
Figure 32 Data and Control Characters and Control Codes 

**Figure 8** - SpaceWire Data and Control Characters and Control Codes

### Network Layer Overview

The Network Layer specifies how SpaceWire Packets, time-codes and distributed interrupts are transferred over a SpaceWire network. SpaceWire Networks are built from links, nodes and routers which are connected together so the can exchange information and work together to perform a desired function. The Network Layer is defined to provide the three principal services within [4] listed below.

1. A packet service which sends and receives packets over a SpaceWire network.
2. A time-code service which sends and receives time-codes over a SpaceWire network.
3. A distributed interrupt service which sends and receives distributed interrupts over a SpaceWire Network.

Within a SpaceWire Network the nodes are required to act as the following three requirements;

1. A source of packets and broadcast codes sent over a SpaceWire network.
2. A destination for packets and broadcast codes received over a SpaceWire network.
3. Both a source and destination of packets and broadcast codes.

### SpaceWire Packets

#### Information is distributed across the network through the use of SpaceWire Packets defined in Reference [4] clause 5.6.2.1; a SpaceWire Packet consists of the format shown in Figure 9. A SpaceWire Packet is made up of three elements;

#### **Destination Address**. The first part of the packet to be sent which is a list of data characters that represent the path a packet should take through the SpaceWire network to reach the destination node. If the link is a point to point link between two nodes, the Destination Address is not required.

#### **Cargo**. The data that is sent following the Destination Address. The length of N-Chars/Bytes that can be transferred in the cargo section is not limited to any number, although it is advised that data is split into appropriate frames to reduce likelihood of errors and need for re-transmissions causing delay.

#### **End of Packet (EoP).** The last section of the packet that indicates the current packet of data has ended. The data character following an EoP is the start of the next packet.

#### Machine generated alternative text: 5.6.2.1 SpaceWire packet A SpaceWire packet shall comprise one or more data characters followed by an end of packet marker (EOP) or error end of packet marker (EEP). start Of a packet shall be either: b. d. e. 2. The first data character sent after a link is initialised or re- initialised following a disconnect. The data character that immediately follows an EOP or EEP. N CYrE The first d at a character after the end Of the previous packet is the start of the next packet. The end of a packet shall be indicated by an EOP or EEP. packet shall contain zero or more data characters. N OTE If a packet conta ins zero data characters, the packet is discarded by the first routing switch that it encounters. Zero or more data characters at the front of a packet shall form a destination address. remaining data characters, following the destination address and up to the EOp or EEP, shall form the cargo. NOrE The format of a SpaceWi re packet is illustrated Destination Address In Figure 5-21. EOP/ Cargo Figure 5-21: SpaceWire packet format

#### **Figure 9** - SpaceWire Packet Format

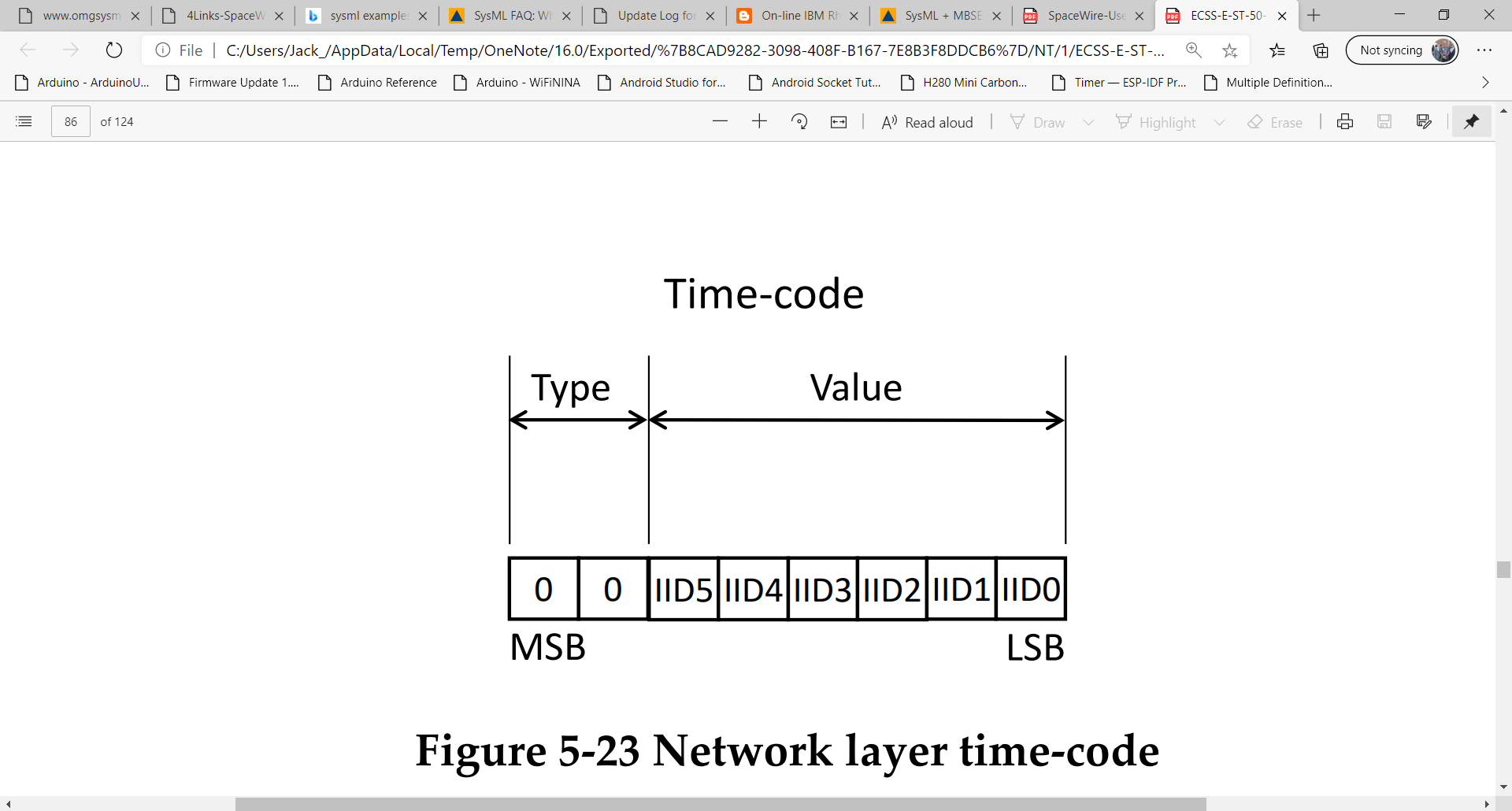
### SpaceWire Packet Addressing

#### The Destination Address at the front of a SpaceWire Packet is used to direct the packet through the SpaceWire Network. The two forms of packet addressing are defined as Path Addressing and Logical Addressing. The main difference lies in how the packets are forwarded to the correct ports from each routing switch to reach the desired destination node; Path Addressing has each path (output port) listed at the front of the packet which is processed by the routing switch, Logical Addressing just has the desired destination node at the front of the packet and the routing switch has pre-defined “directions” (lookup table) on the correct path to take to the desired destination node.

### Broadcast and Time-Codes

SpaceWire supports the use of optional network layer broadcast codes which are used to for the distribution of time-codes and interrupt-codes within a SpaceWire Network. The time-codes allow a network to have a consistent view of time and synchronising sub-systems. The interrupt-codes allow network level interrupts to be distributed across a SpaceWire network for uses such as power-down/up controls, with up to 32 use definable interrupt signals.

The SpaceWire time-code comprises of the SpaceWire ESC character followed by a single 8-bit character, the data character contains two control-flags and a six-bit time count. The SpaceWire time-code format is shown in Figure 10. The time-codes values are stored in a dedicated time-code register by the nodes.



**Figure 10** - SpaceWire Time-Code

The SpaceWire interrupt-code utilises the same 8-bit format as the time-code in Figure 10, except Bit 5 is reserved for interrupt acknowledgment.

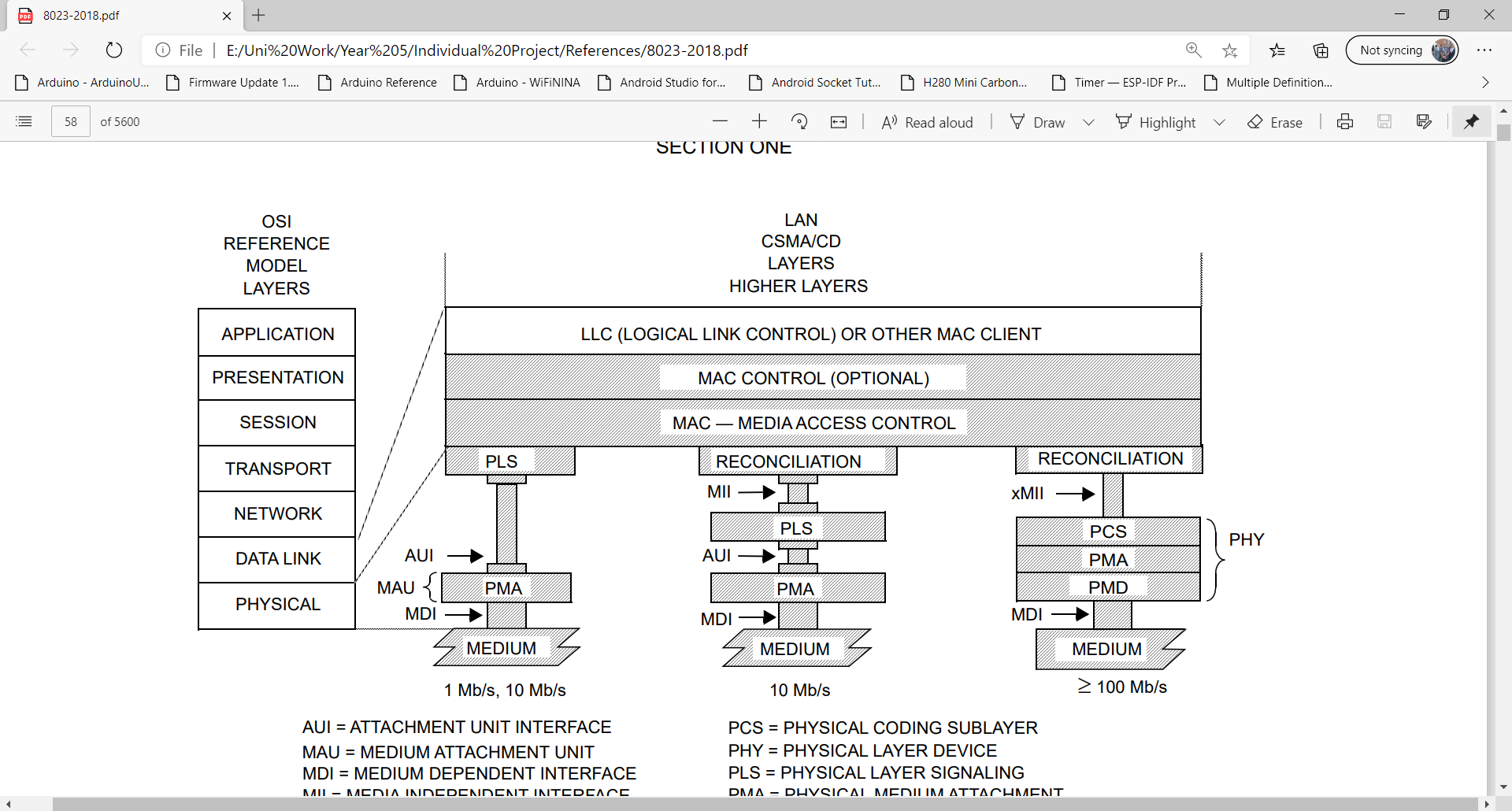
### Management Information Base

SpaceWire regards the Management Base as the way in operations of the other SpaceWire Layers are controlled, configured and monitored. It can be interpreted as the embedded device interfaced with the User Application that communicates and controls the difference layers of the SpaceWire protocol. The specifications of what is required of the Management Interface base for each layer of the protocol is listed within the SpaceWire standard [4].

## IEEE 802.3 Ethernet

### IEEE 802.3 Overview

#### The Institute of Electrical and Electronic Engineers (IEEE) 802.3 Standard [5] is an international standard for Local and Metropolitan Area Networks, employing Carrier Sense Multiple Access/Collision Detection (CSMA/CD) and the Ethernet Protocol (IEEE 802.3) for data communication. In reference to the OSI Conceptual model (Figure 2) the Ethernet Protocol implements the Physical and Data-Link Layer for a data communications. The sub-layers Media Access Control (MAC) and Logical Link Controller (LLC) make up the Data-Link Layer of the OSI Model. The representation of IEEE 802.3 Ethernet Protocol in reference to the OSI Model is shown in Figure 10, with the variances in the physical layer depending on the transmission speeds available within the protocol.



**Figure 11** - 802.3 Ethernet OSI Concept Model Comparison

### Physical Layer Overview

The Physical Layer states the connectors denoted as the Medium Dependant Interfaces (MDI) for the IEEE 802.3 Ethernet Protocol. The implementation of the twisted pair topology are realised by the RJ45 connector as defined in IEC 60603-7:1990. The 8-pins on the connector and physical design are shown in Figure 11, consisting of 4 twisted-pairs. The physical layer provides the interface to allow the signals defined within 802.3 Protocol to be transmitted and received by devices through the use of differential signalling wires.

Machine generated alternative text:
Contact 
MDI signal 
Not used by I OBASE-T 
Not used by IOBASE.T 
Not used by IOBASE-T 
Not used by I ('BASE-T 
Figure MDI connect 
Figure 14-22—Twisted-pair link 
segment connector 

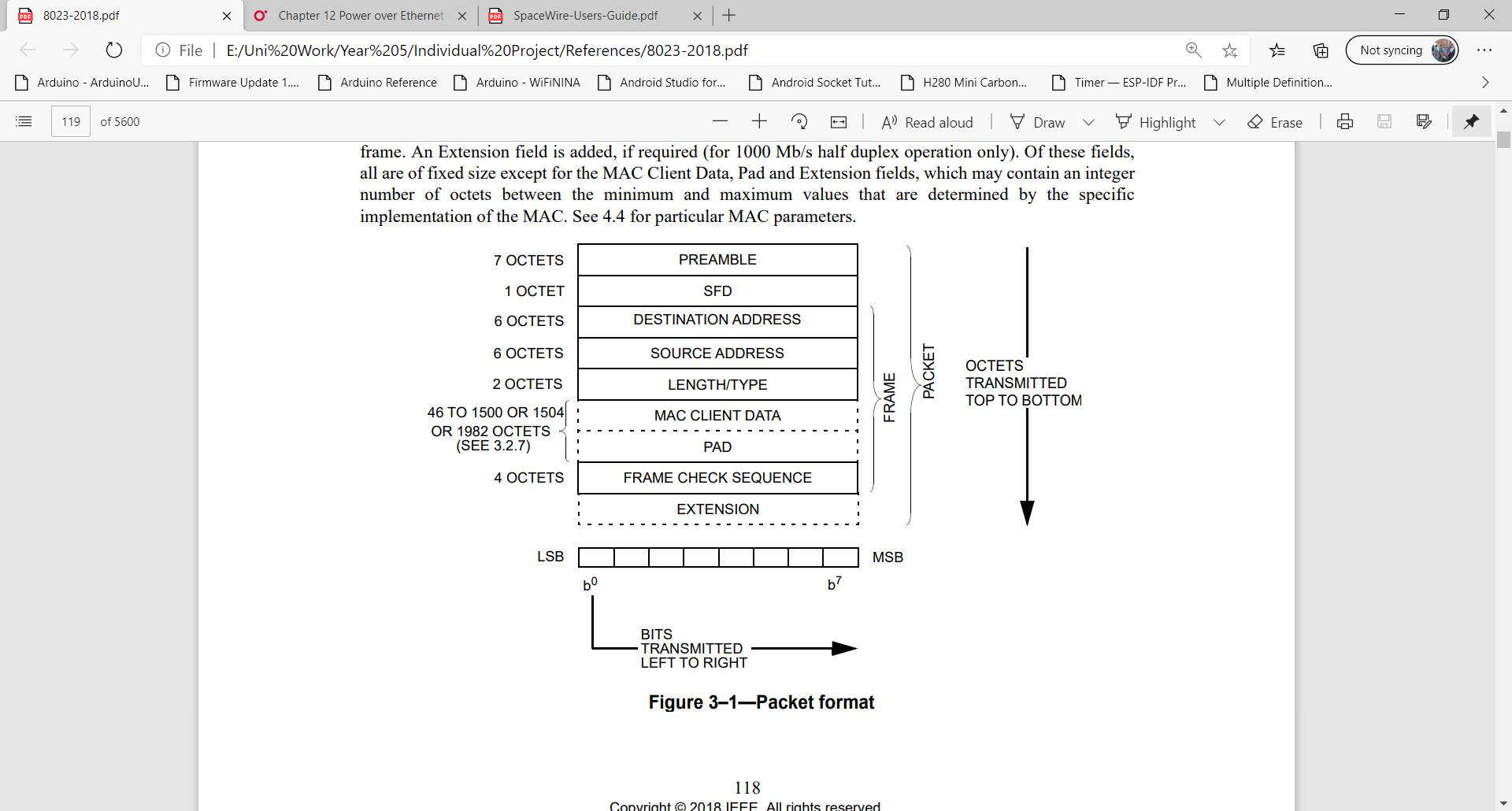
**Figure 12** - IEEE 802.3 RJ45 Twisted Pair Connector and Pinout

### Data-Link Layer Overview

#### The IEEE 802.3 specifies the use of the two sub-layers MAC and LLC for the partitioning of data into frames and ethernet packets that are to be interfaced with the physical layer for transmission. These two sub-layers in unison achieve the below, stated within [5] (Reference [5]).

#### 

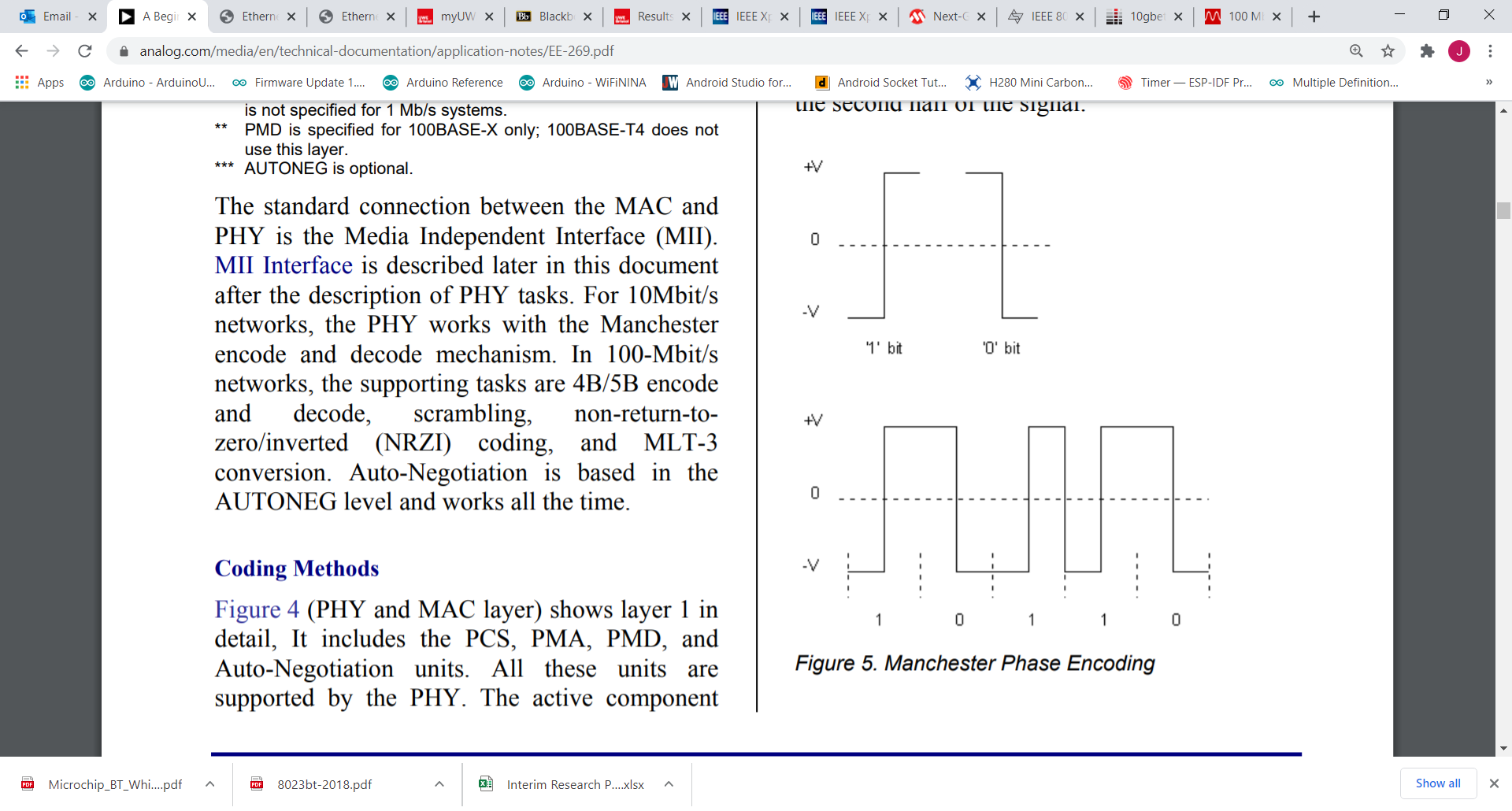
#### The MAC packet and frame format specifications are shown in Figure 12. The MAC packet and frame allows the synchronisation, addressing, client data and error checking of the data to be transmitted and received across the network.



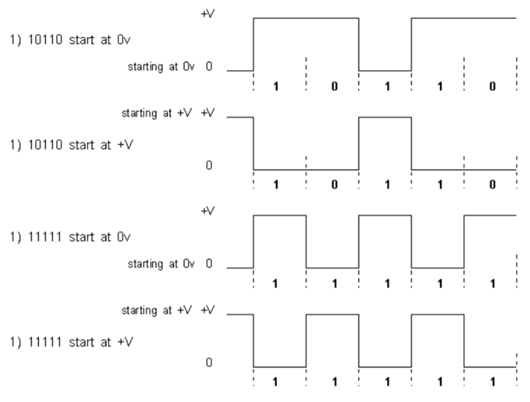
**Figure 13** - 802.3 Ethernet MAC Frame and Packet Specification

The method of encoding for Frames and Ethernet Packets depends on the bit rate of the carrier signal. IEEE 802.3 [5] for 10Mbits/s data rate uses the Manchester Data Encoding Scheme when transmitted at the physical layer. Manchester Encoding uses the transition from Hi and Lo states to signal a ‘1’ or ‘0’ bit-symbol per clock cycle, therefore this allows a 10MHz baud rate. An Example of the Manchester Encoding Scheme is shown below in Figure 13 (Reference [6])for a bit sequence of ‘10110’.

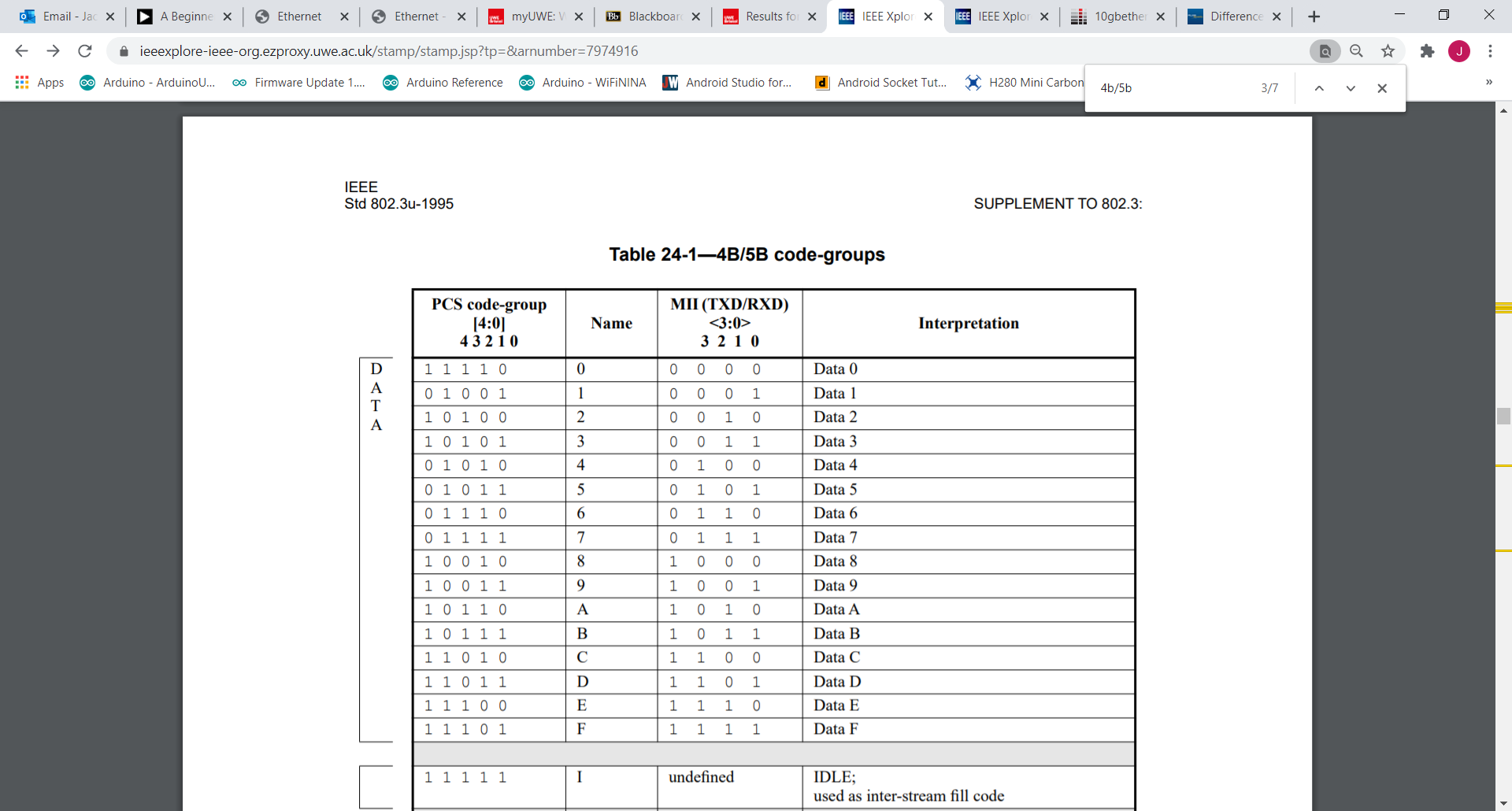
Fast Ethernet or 100BASE-TX is defined by IEEE 802.3u-1995 [7] for 100Mbits/s data rate which utilises the 4B/5B block encoding scheme and transmitted via Non-Return Zero Inversion (NRZ-I). The 4B/5B encoding scheme takes data nibbles (4-bits) to/from five-bit code groups, although only 16 of the 32 codewords are used to allow for atleast 2 signal transitions (two 1-bit values) due to the use of NRZ-I to improve receiver synchronisation. An example of the NRZ-I encoding scheme is shown in Figure 14 and five 4B/5B code-words in Figure 15 .



**Figure 14 -** Example Manchester Encoding Waveforms



**Figure 15** - Non-Return Zero Inversion Example

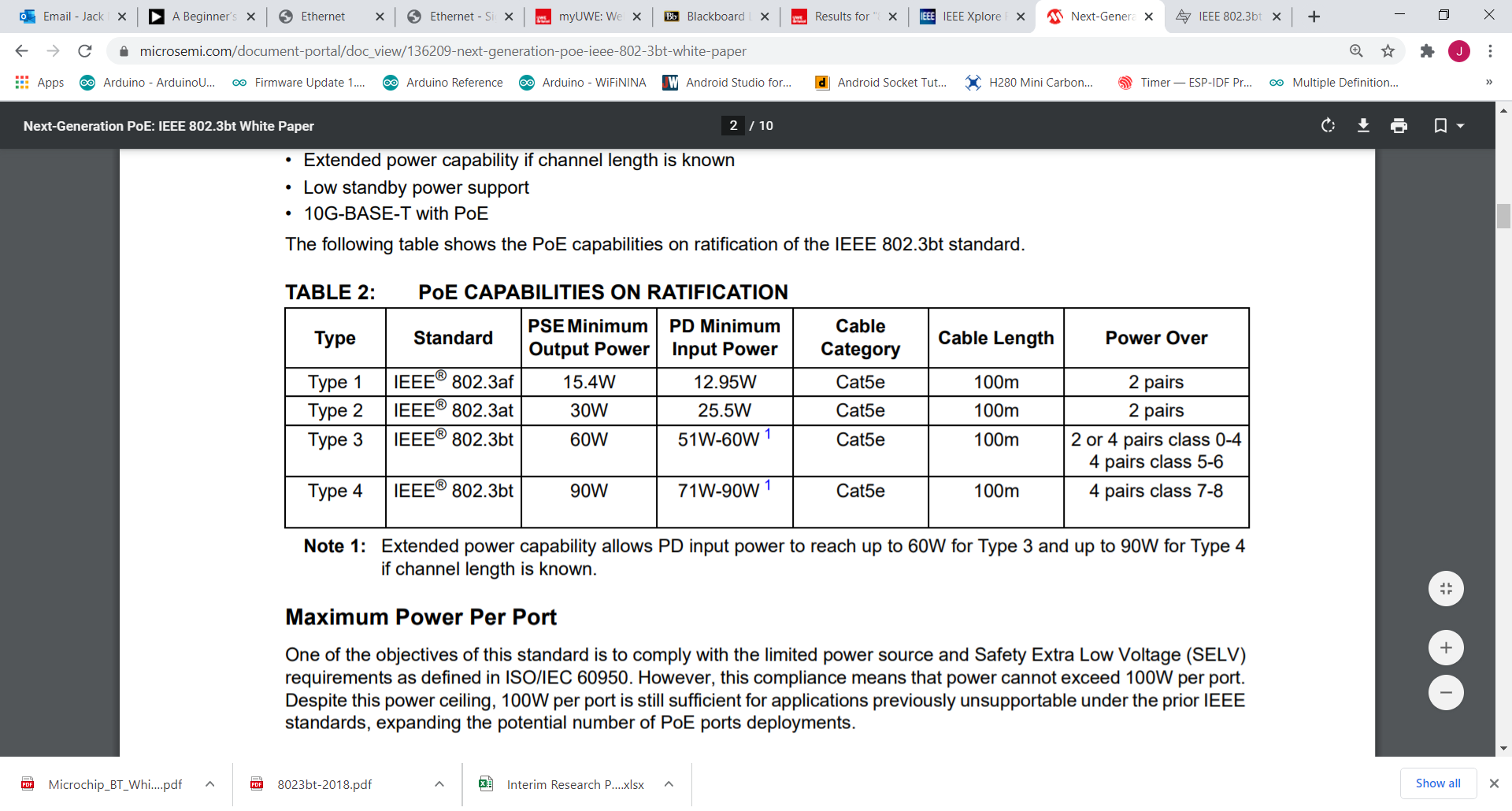


**Figure 16** - IEEE 802.3u 4B/5B Encoding Scheme

### IEEE 802.3 Power over Ethernet Overview

The IEEE 802.3 Ethernet standard defines the specifications to include the distribution of power over an Ethernet based network, in addition to data communications within a network. The latest Power over Ethernet (PoE) capability is stipulated through the IEEE 802.3bt [8] standard utilising the 4-pair twisted pair cabling that has been qualified to be used within the standard. A PoE system comprises of three separated specified major elements, these are the; power supply defined as the Power Sourcing Equipment (PSE), the powered load defined as the Powered Device (PD) and twisted pair cabling connecting the two devices.

There have been many iterations of the IEEE 802.3 PoE Standard which has been enhanced since its introduction in 2003 with the 802.3af standard. Reference [9] provides a summary of the current PoE capabilities that are available, the key parameter being the total amount of output power that can be provided to a powered-device.



**Figure 17** - IEEE 802.3 PoE Standards Summary

An simple example of a PoE system is shown below in Figure 17 (Reference [6]). In this example the diagram shows how the PSE utilises two unused twisted pairs to transmit power to the PD. Various topologies can be implemented which are categorised as Types within the IEEE 802.3bt standard which alters the physical architecture depending on the power source, number of components and the number of cables used. The data signal pairs can be utilised for power transmission also as differential signalling is utilised; therefore, the data signal is resultantly unaffected by a large DC offset.

Machine generated alternative text:
POWERED DEVICE 
POWER SOURCING 
EOLJIPtv1ENT (PSE) 
Figure 15. POE Connection 

**Figure 18** - Example PoE System

## PoE and SpaceWire Protocol Stack Implementation

A communications system should contain a full protocol stack to allow the exchange of information between sub-systems/devices. A full protocol stack can be described as shown in the OSI Concept Model (Figure 2). For the implementation of the ECSS SpaceWire protocol into IEEE 802.3 Ethernet, each layer of the stack needs to be identified and implemented from each protocol in order to provide a complete communications system.

The data-link layer of the SpaceWire and 802.3 Ethernet Protocol have variances in their metrics and formatting which are detailed in Table 1, summarised in Reference [10]. Overall SpaceWire is alot lighter in its packets and more flexibility which is desirable for power consumption and modularisation of sub-systems.

|  |  |  |  |
| --- | --- | --- | --- |
| **Protocol** | **Data-rate Speed** | **Packet Size** | **Error Check** |
| 802.3 Ethernet | 10Mbps/100Mbps/1Gbps/10Gbps | Min 64 bytes  Max 1518 bytes | 32-bit CRC |
| ECSS SpaceWire | 2-200+ Mbps (hardware dependent) | No limit on length | 1-bit Parity |

**Table 1** - PoE and SpaceWire Data-Link Layer Comparison

From the documentation review of the ECSS and IEEE standards, Table 2 demonstrates how the SpaceWire Protocol can be implemented using the existing layers of the Ethernet standard to fulfil a complete protocol stack. The Physical and Data-Link Layers of the OSI Model are implemented using existing hardware that are IEEE 802.3 PoE compliant. The Network Layer is implemented by software to receive and transmit SpaceWire Packets. The Application and Presentation Layers are not required for the functioning of the SpaceWire Protocol, although shall be used for functional testing, ad-hoc software development and debugging.

|  |  |  |  |
| --- | --- | --- | --- |
| **OSI layer** | **Layer Implementation** | **Project Implementation** | **Description/Function** |
| Application | User GUI/Embedded System | Data Request/Control Signals | Shall act as the User Interface for manual requests of data and control codes to be distributed to SpaceWire Nodes |
| Presentation | Embedded System | N-Chars, ASCII | Shall take the data from the network layer (packets) and encode them into the appropriate format depending on the purpose. ASCII Characters to be used for human-readable. |
| Session | Not used. | N/A |  |
| Transport | Not used. | N/A |  |
| Network | SpaceWire | Software Protocol | Shall format the data and control codes received from the User Interface or other SpaceWire Nodes through the data-link layer into the SpaceWire Packet format. |
| Data-Link | IEEE 802.3 Ethernet | IEEE 802.3 Ethernet PoE compliant hardware | Shall establish and maintain a link with other Ethernet Nodes on the Network.  Shall encode the SpaceWire packets received from the Network Layer into Frames to using Manchester Encoding (10Mbps) or 4B/5B NRZ-I Encoding (100Mbps).  Shall decode the bitstream received from the Physical Layer into Frames to be passed up to the Network Layer to receive the SpaceWire Packet.  Shall direct the Frames to the required Destination Address using the MAC protocol. |
| Physical | IEEE 802.3 Ethernet | IEEE 802.3 Ethernet PoE compliant hardware | Shall drive the bitstream data signals through the twisted pair cabling using differential signalling. |

**Table 2** - ECSS SpaceWire and IEEE 802.3 PoE Full Protocol Stack Implementation

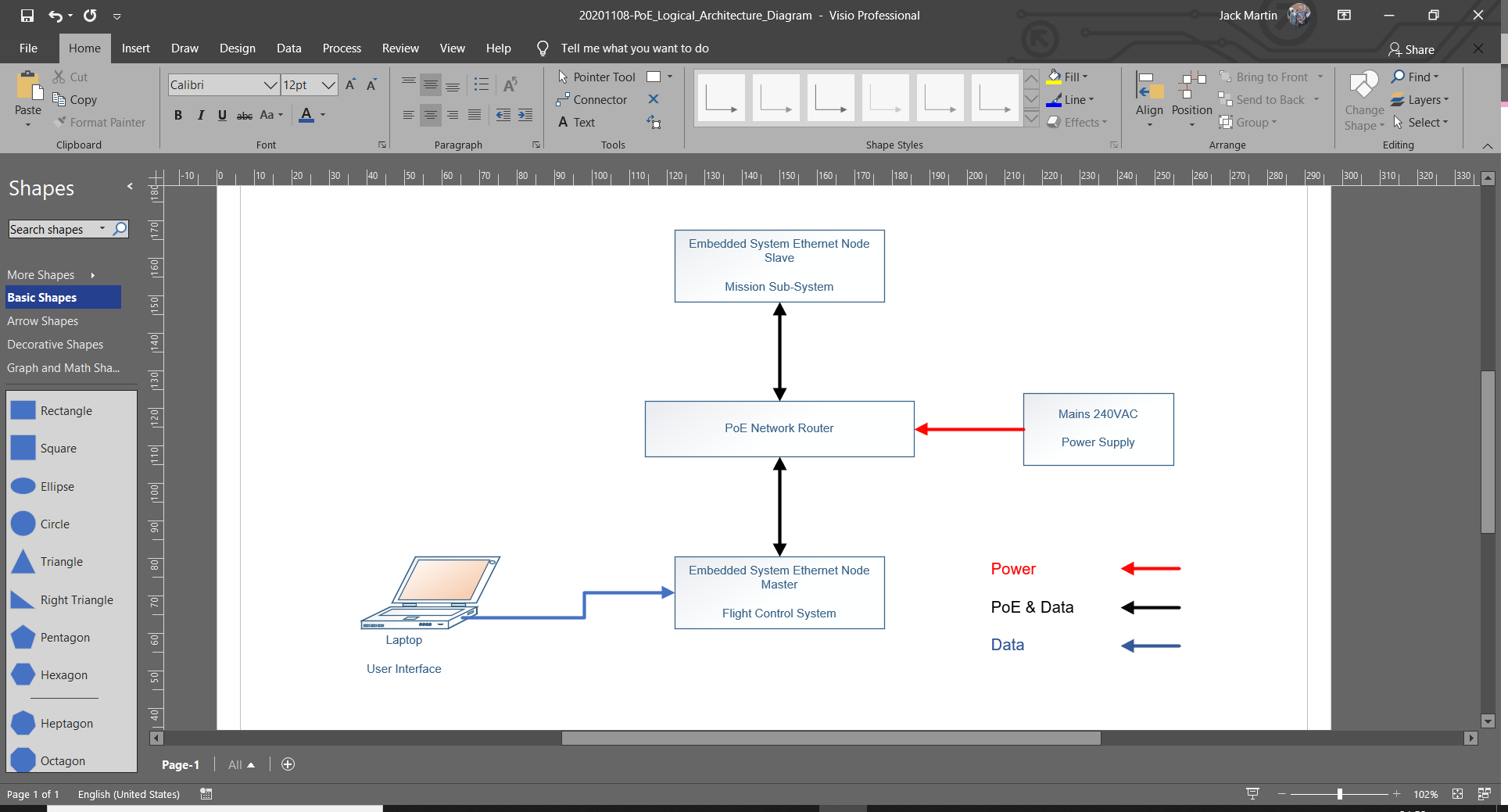
# Embedded Systems Development

## Hardware

### System Architecture

To demonstrate the feasibility of the concept of the project to its greatest extent and understand fully the system requirements for the design and functional testing, it was desirable to replicate a simplified spacecraft system architecture use-case. The Logical Architecture Diagram shown in represents the typical topology of a spacecraft mission system and how a SpaceWire network would be utilised as the method of communications between mission sub-systems. The key components of the Logical Architecture and their relationship to a spacecraft mission system use-case are listed below;

1. Laptop



## Software

### UML Diagram

# Functional Testing and Results

## Testing Plans

## Results and Analysis

# Conclusion

## Results and Analysis

## Recommendations

## Summary

# References

|  |  |
| --- | --- |
| [1] | J. Martin, *20201010-Individual\_Project\_PoE,* Oct 2020. |
| [2] | J. Martin, *20201026-PoE\_Project\_Meeting\_Minutes,* Oct 2020. |
| [3] | ISO, *ISO/IEC 7498-1:1994 The Basic Model,* 1994. |
| [4] | European Cooperation for Space Standardisation, *SpaceWire - Links, Nodes, Routers, and Networks, ECSS-E-ST-50-12C Rev.1,* ESA Requirements and Standards Division, May 2019. |
| [5] | IEEE, *IEEE 802.3-2018 Standard for Ethernet,* Jun 2018. |
| [6] | Analog Devices, *A Beginner' Guide to Ethernet 802.3,* Jun 2005. |
| [7] | IEEE, *IEEE 802.3u-1995 - Media Access Control (MAC) Parameters, Physical Layer, Medium Attachment Units, and Repeater for 100 Mb/s Operation, Type 100BASE-T (Clauses 21–30),* Jun 1995. |
| [8] | IEEE, *IEEE 802.3bt-2018 - IEEE Standard for Ethernet—Amendment 2: Physical Layer and Management Parameters,* Sep 2018. |
| [9] | Microchip Technology Inc., *Next-Generation PoE: IEEE 802.3bt White Paper,* Mar 2019. |
| [10] | STAR-Dundee, Steve Parkes, *SpaceWire User's Guide,* STAR-Dundee Limited, 2012. |

# Annexes