ETHERNET ENCAPSULATION AND EMULATION FOR DVB/MPEG MULTIMEDIA WIRELESS SYSTEMS: BASE STATION AND SUBSCRIBER INTERFACE VHDL DESIGNS

Wai Yuen, Mun

A thesis submitted in partial fulfilment of the requirements of
Nottingham Trent University for the degree of
Doctor of Philosophy

School of Computing & Informatics

SEPTEMBER 2005

ABSTRACT

This research focuses on novel network interfaces for a 42GHz DVB-based Multimedia Wireless System (MWS): The Nottingham Trent University Campus MWS trial. 'Very high speed integrated circuits Hardware Description Language' (VHDL) designs of Base station and subscriber MWS interfaces developed in this research are tailored for the provision of high bandwidth Ethernet-based services which are in demand for last-mile data access. This research proposes efficient Ethernet encapsulation and emulation techniques for implementation in dedicated Field Programmable Gate Arrays (FPGAs) to simplify the internetworking architecture while enabling the low latency and high throughput performance required for last-mile Ethernet-based services.

A variety of encapsulation techniques are analysed to find an algorithm that can be entirely and cheaply implemented in dedicated FPGAs suitable for the MWS application. A byte-stuffing type encapsulation is proposed instead of frame-length marking (FLM) type encapsulation used in previous MWS interfaces. To support this, a unique byte-level acquisition and subsequent analysis of 'live' Ethernet frames was undertaken to address several prevailing criticisms against the use of byte stuffing algorithms. A new variant of byte stuffing algorithm, Optimised Byte Stuffing (OBS) tailored for this specific application, was developed using the results drawn from Ethernet traffic analysis. While VHDL design and synthesis demonstrated low cost implementation of OBS in FPGA, the Ethernet traffic analysis reveals that OBS can reduce latency compared to FLM type encapsulation and reduce jitter compared to the commonly used Point-to-Point Protocol byte stuffing encapsulation.

A further investigation was undertaken to make the underlying MWS emulate an Ethernet hub, thus minimizing internetworking complexity and facilitating the provision of Ethernet-based services. This lead to the proposition of an Ethernet Hub Emulation (EHE) MWS architecture, which yields a more simplistic and lower cost solution to known IP routing problems due the topology inherent to MWS. The EHE architecture is primarily implemented in the MWS base station interface which can mimic the broadcast-like nature of Ethernet networks while maintaining low latencies and high throughputs.

Novel MWS base station and subscriber interface designs incorporating the OBS encapsulation and EHE architecture, were successfully hardware prototyped using an integrated suit of modern Electronic Design Automation (EDA) design flow. The designs were described in VHDL, simulated, synthesized and programmed in low-cost FPGAs. Custom developed electronic hardware and equipment from the Nottingham Trent University MWS campus trial provided the opportunity to deploy and evaluate the designs under real conditions. The MWS test platforms, in point-to-point and point-to-multipoint configurations, facilitated live demonstrations of IP/web and Ethernet-based services over MWS. Using standard benchmarking software, the test-platforms enabled cost, flexibility, and service-level performance of the novel MWS base station and subscriber interfaces to be compared with previous systems. Results from the work done provide conclusive evidence of the advantages of the proposed MWS interfaces for the provision of Ethernet-based services. The low cost hardware prototypes developed form the building blocks for further work, where deployment of the novel MWS base station and subscriber interfaces in a widespread trial is envisaged.

ACKNOWLEDGEMENTS

I offer my utmost thanks and appreciation to my supervisors Dr. Richard Germon and Dr. Steve Clark for supporting me through my studies as a doctorial student. Limitless thanks to Richard for the impeccable guidance and patience throughout the years and especially for reading and commenting on this thesis, though any remaining faults are mine.

I would like to thank those closest to me; LiLi, Adrian, David, KumWah, Sharon, EeBeng, Charlye and Joyce. They were my main source of encouragement and whose company was always a welcome distraction that made the tougher times of research seem easy.

To my parents KamSeng and Jenny, my brother Ean, and sister Emy; no words are enough to describe their love, understanding and occasional monetary boost that helped me through my studies. I give my heartfelt gratitude.

Last but not least, to all those that I have not mentioned, thank you.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

AAL	ATM Adaptation Layer
ACTS	Advanced Communications Technologies and Services
ARP	Address Resolution Protocol
ASCII	American Standard Code for Information Interchange
ASIC	Application Specific Integrated Circuit
ATM	Asynchronous Transfer Mode
ATM-PVC	Asynchronous Transfer Mode - Permanent Virtual Channel
BFWA	Broadband Fixed Wireless Access
BGP	Border Gateway Protocol
BS	Byte Stuffing
CABSINET	Cellular Access to Broadband Services and Interactive Television
CLIP	Classical IP over ATM
COBS	Consistent Overhead Byte Stuffing
CRABS	Cellular Radio Access for Broadband Services
CRC	Cyclic Redundancy Check
CS	Convergence Sublayer
CSMA/CD	Carrier Sense Multiple Access / Collision Detection
DA	Destination Address
DAVIC	Digital Audio Video Council
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name Server
DOCSIS	Data-Over-Cable Service Interface Specification
DVB	Digital Video Broadcasting
DVB-C	Digital Video Broadcasting - Cable
DVB-LMDS	Digital Video Broadcasting – Local Multipoint Distribution System
DVB-S	Digital Video Broadcasting - Satellite
DVB-SPI	Digital Video Broadcasting - Synchronous Parallel Interface
EDA	Electronic Design Automation
EHE	Ethernet Hub Emulation
EMBRACE	Efficient Millimeter Broadband Radio Access for Convergence and Evolution
EN	Enable
EOF	End of Frame
ERC	European Radio Communication Committee
ETSI	European Telecommunications Standards Institute
FDMA	Frequency Division Multiple Access
FIFO	First In First Out

FLM	Frame Length Marking
FLP	Fast Link Pulse
FPGA	Field Programmable Gate Array
GFP	Generic Framing Procedure
HDLC	High Level Data Link Control
HT	Header Type
IEEE	Institute of Electronic and Electrical Engineering
IETF	Internet Engineering Task Force
INA	Interactive Network Adaptor
IP	Internet Protocol
LAN	Local Area Network
LE	Logic Element
LED	Light Emitting Display
LLC	Logical Link Control
LMDS	Local multipoint distribution system
LVDS	Low Voltage Differential Signaling
MAC	Media Access Control
MAN	Metropolitan Area Network
Mbps	Mega bits per second
MDC	Management Data Clock
MDIO	Management Data Input/Output
MII	Medium Independent Interface
MPE	Multi Protocol Encapsulation
MPEG	Motion Pictures Expert Group
MPEG-TS	Motion Pictures Expert Group - Transport Stream
MPLS	Multiprotocol Label Switching
MWS	Multimedia Wireless System
NTU	Nottingham Trent University
OBS	Optimised Byte Stuffing
OSI	Open Systems Interconnection
PDU	Protocol Data Unit
PES	Packetized Elementary Stream
PID	Program IDentity
PLL	Phase Locked Loop
PLS	Physical Layer Signaling
PPP	Point-to-Point Protocol
PTI	Payload Type Indication
PUSI	Payload Start Indicator
RS	Reed Solomon
RTL	Register Transfer Logic
Rx	Receiver

SA	Source Address
SAP	Service Access Points
SDL	Simple Data Link
SFD	Start Frame Delimiter
SID	Station IDentity
SOF	Start of Frame
SPI	Synchronous Parallel Interface
SSL	Secure Sockets Layer
TCP	Transmission Control Protocol
TDMA	Time Division Multiple Access
TTL	Transistor-Transistor Logic
Tx	Transmitter
UDP	User Datagram Protocol
ULW	Ultra Lightweight Encapsulation
VC	Virtual Circuit
VHDL	Very High speed integrated circuit Hardware Description Language
VPN	Virtual Private Network

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

Multimedia Wireless Systems (MWS) are cost-effective high-capacity access networks particularly suitable for the 'last-mile' provision of triple-play services - data, voice (IP-based) and television broadcast services [1,2]. An MWS consists of all the network elements necessary for it to provide these broadband services using millimetre radio waves as its fundamental transport medium. Throughout Europe, a harmonized spectrum band of 40.5-43.5 GHz has been allocated for MWS by regulatory organisations following the European Radio communications Committee (ERC) decision in June 1999 [3]. The use of 42 GHz spectrum sets MWS apart from other types of Broadband Fixed Wireless Access (BFWA) networks as signal propagation at this frequency has higher attenuation due to rain compared to lower frequencies. Even with the use of sophisticated forward-error-correction techniques, typical MWS links are restricted to line-of-sight and distances up to 5 km [4]. However, MWS is able to provide greater capacity per square kilometre due to larger spectrum bandwidth and higher density of base stations. It is therefore vital for an MWS to take advantage of its capacity to support the said triple-play services for it to compete with BFWA at lower frequencies [5].

Widespread deployment of MWS is yet to be realised, leaving the allocated spectrum largely unused to this date. Explanations from a work [5] in this field, specifically referring to MWS, suggests that; "for the time being, equipment is generally too expensive. Low volume and lack of focus on entertainment services ... has made both the base station and user radio terminal rather expensive... This in turn hinders broad market acceptance". This view is also shared by another work [4] which suggested that "the main technological challenge is production of a real low-cost two-way user terminal for the private market". Whilst, the Institute of Electronic and Electrical Engineering (IEEE) had initially published a standard (IEEE 802.16-2003) for BFWA systems between the 11 GHz to 66GHz bands, the current focus of WiMAX forum is on the IEEE 802.16e-2004 amendment which encourages mass production of equipment for BFWA systems below 11 GHz [6, 7].

One way of achieving a low-cost system is to adapt existing mass-produced equipment from established markets. The Nottingham Trent University (NTU) MWS trial [8] adapted equipment from Digital Video Broadcasting (DVB) networks originally meant

for cable and satellite networks. The ongoing research on MWS is based on a trial network formed through a collaboration between MMRadiolink Ltd (Philips, UK), Hughes Network Systems Ltd, and The Nottingham Trent University Communications Research Group [9,10].

Adapting existing DVB based equipment for MWS has another significant advantage; a DVB-based MWS will inherit the infrastructure to readily deliver robust 'Motion Picture Expert Group Transport Stream' (MPEG-TS) based television broadcast service – one of the above mentioned triple-play services. The emergence of MPEG-TS [11,12] as the *de facto* standard for digital audio-video transmission with support for rich features, such as content protection and interactive data, has found wide applicability in satellite, cable, and terrestrial broadcasting [13,14,15]. In related research Lam [16] focused on a long-term 42GHz measurement campaign, the NTU MWS trial has already demonstrated integration with commercial DVB satellite and terrestrial networks which enabled real-time re-broadcasting and reception of MPEG-TS broadcast services.

1.1. Ethernet-based Services in Metropolitan Area Networks

Whilst MWS subscribers from residential markets can be realised with low-cost subscriber nodes, a significant portion are likely to remain in the business arena that can sustain high value subscriptions for large bandwidth data and voice services [17]. This puts MWS in competition with fibre optic networks that currently serve the business market [5,17]. Due to the dominance of Ethernet in the Local Area Network (LAN) environment, Ethernet-based services are viewed as the key technology for providing data and voice access services especially to businesses. In fibre optic Metropolitan Area Networks (MAN), the support for Ethernet-based services has already been shown to provide a more flexible, lower-cost and less complex networking architecture than legacy networking technologies which benefit both the subscribers and service provider [18,19]. The following paragraphs are two examples out of many works in the field that stress the importance of Ethernet-based data services in the future of metropolitan area networking.

"Service providers are on the lookout for supporting technologies that enable newer Ethernet-based services such as transparent LAN service (TLS) connecting various customer sites across a metropolitan domain (metro Ethernet)"[18].

"The enormous diffusion of the Ethernet technology in the LAN [Local Area Network] environments and the availability of Ethernet interfaces with a very low cost has almost forced telecom operators to consider it as the only possible MAN [Metropolitan Area Network] access technology to sell Internet access services to the large public and business"[19].

MWS can also be a suitable access network to provide Ethernet-based services as it offers several infrastructure advantages over competing MAN access networks [4]. For example; ease and speed of system deployment with minimal disruption to the community, and lower capital costs as the consumer infrastructure is not preinstalled. However, current MWS interfaces are not optimised (if capable) for the transport of Ethernet-based services in a transparent and hassle-free manner. This forms the motivation for this research to investigate and develop cost-efficient base station and subscriber MWS interfaces for the NTU DVB-based MWS and be able to efficiently support Ethernet-based services.

1.2. Research Objectives

The main thrust of this research is to integrate Ethernet technology into a DVB-based MWS so that it can also be competitive with existing access networks in providing data and voice (IP-based) services. This is done by investigating and developing novel subscriber and base station network interfaces for the NTU MWS. In the context of providing data services, current experience from existing BFWA system trials including the NTU MWS trial (thoroughly surveyed in Chapter 2) had revealed limitations in the network interfaces of existing equipment based on DVB and other technologies. Hence, the research aims of the novel subscriber and base station interfaces was not only to enable seamless Ethernet-based data services, but also address some of the cost and performance limitations found in previous systems. The primary objectives that were set out are as follows:

i) Investigate and develop a tailored encapsulation algorithm to efficiently encapsulate Ethernet frames into DVB/MPEG-TS that enables the NTU MWS to provide transparent Ethernet-based data services, while minimising latency and jitter, suitable for an access network.

ii) Using the Ethernet technology, investigate and develop an MWS architecture that addresses internetworking complexity issues found in previous systems.

iii) Design of the MWS subscriber and base station interfaces incorporating the above features into low-cost FPGA to lower overall equipment costs.

iv) Develop prototypes of the subscriber and base station interfaces with at least 56Mbps encoding rates to match maximum operating speed of DVB-S equipment.

v) Deploy and investigate the capabilities of prototypes in a live Ethernet environment to measure 'real' performance and provide proof of theoretical concepts.

1.3. Research Contributions

The work undertaken in this research has resulted in the following main contributions:

- Acquisition of a unique byte-level trace of typical-composition from a 'live' Ethernet network, and subsequent frame-size and byte-value distributions analysis.

  Existing network traces were acquired at frame-level which did not retain the necessary byte-level information for analysis of Byte Stuffing algorithms. The results from statistical analysis of the trace facilitated a fair comparison of Byte Stuffing and Frame-length marking encapsulation algorithms. Also a statistically 'best' byte-value of the traces was determined that can be used by BS algorithms to achieve higher efficiency.

- Creation of a Byte Stuffing algorithm, Optimised Byte Stuffing (OBS), its design in VHDL (Very High speed integrated circuit Hardware Description Language) and its FPGA synthesis.

  Previous BFWA interfaces had adopted Frame-Length-Marking type encapsulation algorithms with excessive headers or were inefficient for Ethernet transport. Besides using statistically optimal byte-values, the OBS algorithm uses unique mechanisms for frame encapsulation, synchronisation and stuffing, while minimising headers for efficient transport of Ethernet frames.

- VHDL design, simulation and synthesis into low-cost FPGAs of the 'Ethernet to DVB/MPEG-TS Encoder and Decoder' interface incorporating the OBS algorithm. This formed the necessary blocks for subscriber interfaces for the NTU DVB-Based MWS. Programming of the designs into an FPGA development board and integration into the NTU MWS equipments allowed the construction of a point-to-point MWS demonstrator.

  Whilst the encoding rates of the interfaces implemented in hardware or software had limited the potential of previous BFWA systems such as; encoding bottle neck and source of latencies up to 100ms, the 'Ethernet to DVB/MPEG-TS Encoder and Decoder' interface was implemented entirely in FPGA hardware and was shown to surpass the 56 Mbps required encoding rates to saturate the DVB equipments, and also guaranteeing less than 5 ms of latency. Over a point-to-point MWS test platform, ICMP, TCP/IP and UDP/IP based experiments revealed that 'Ethernet to DVB/MPEG-TS Encoder and Decoder' interface hardware were robust with consistent results when cross verified with simulations. Also, pre-calibrated experiments and comparison with wired Ethernet produced correlated empirical results which prove that the theoretical benefits of OBS and other modules can be realized in practice.

- VHDL design, simulation and synthesis of two versions of MWS base station interface incorporated a proposed Ethernet-Hub-Emulation (EHE) architecture. The second version of the MWS base station was programmed into FPGA

development boards which produced a working prototype. Connecting the base station and two subscriber interface prototypes formed a complete point-to-multipoint MWS demonstrator.

Using the MWS demonstrator, the university Ethernet network was extended to an isolated LAN Ethernet with a 802.11g Wi-Fi access point. Experiments conducted showed the MWS was able to provide Ethernet-based and voice-over-IP services with hassle-free connectivity. In addition, the overall simplification of the MWS internetworking architecture and implementation into dedicated FPGAs enabled the use of the low-cost MWS interfaces instead of more sophisticated and expensive internetworking equipments in the NTU MWS. Service-level latency and throughput performance results were also compared to previous BFWA systems which showed the significance of latency and operational speed improvements.

1.4. Thesis Structure

The organisation and brief description of the thesis on a chapter by chapter basis are as follows:

Chapter 2 is the literature review chapter that provides the theoretical background as well as a survey of related prior work. Rather than focusing on just existing MWS interfaces, the review takes a broader scope covering related literature on BFWA including standard documents and published work on previous trials. This is done so that the experience and innovations from these systems, where applicable, can be learnt and built upon.

Chapter 3 presents the reasons for using a Byte Stuffing type algorithm instead of Frame-Length marking type exclusively used in all the surveyed systems. Using empirical results from analysis of a byte-level trace from a 'live' Ethernet network, the primary criticisms of the using Byte Stuffing algorithms are shown to be unsubstantiated in this application. Also a proposed Optimised Byte Stuffing (OBS) algorithm and its VHDL design that is tailored for the subscriber and base station MWS interfaces investigated are detailed.

Chapter 4 details the VHDL design, simulation, synthesis and hardware prototyping of the 'Ethernet to DVB/MPEG Encoder and Decoder' which forms an MWS subscriber interface. Explanation of key design innovations and functions at the individual module level, that make up the 'Ethernet to DVB/MPEG Encoder and Decoder' are also described. Simulation and experimental results are cross verified to check the integrity of methodology based on Electronic Design Automation (EDA) for FPGA prototyping from the VHDL designs. Merging the simulation and experimental results, the performance profile graph of the 'Ethernet to DVB/MPEG Encoder and Decoder' is established which shows that the worst case scenario will still be within the design specifications. The performance profile graph can also be used to predict the encoding/decoding latency of any frame given the frame-size and current utilisation level.

Chapter 5 describes the Ethernet-Hub-Emulation (EHE) MWS architecture adopted to simplify the internetworking issues present in previous systems. Here, the VHDL design, simulation and synthesis of two versions of the MWS base station interface incorporating the mechanisms to support the EHE architecture are detailed together with the hardware prototyping of the second base station design. The latter enabled the construction of a complete point-to-multipoint MWS using the base station and two subscriber interface prototypes. Whilst the prototyping of the MWS provides evidence for the feasibility of low-cost MWS interfaces, the experimentation conducted demonstrates the claimed flexibility and performance benefits of the EHE MWS architecture.

Finally, Chapter 6 reprises the contributions of this research as well as discusses some open issues in light of the experience gained. Possible further works are also discussed for continuing research while suggesting possible solutions to some immediate issues.

1.5. Terms and Definitions

Some terms used in this thesis might have ambiguous meanings, depending on the readers own frame of reference. A list of several terms that are known to have alternative meanings is provided below together with thier definition used in the context of this thesis:

Congestion control

A Network layer flow control mechanism.

Flow control

A Data Link Layer flow control mechanism.

Frame

A Data link Layer data unit.

Maximum Transmission Unit (MTU)

The value of maximum size of a packet that can be transmitted over a subnet.

Packet

A Network Layer data unit.

Subnetwork

A physical network segment that is part of an internetwork

Subnet

A logical set of addresses in the IP protocol.

Transparent network

When a host does not know of the existence of a subnetwork along the path to another host, that subnetwork can be said to be transparent.

2 BFWA NETWORK INTERFACE REVIEW

A literature review of material related to the research undertaken is covered with the aim of providing background information of key theories and technologies, as well as a comprehensive survey of related prior work in the field. The literature review establishes that existing Broadband Fixed Wireless Access (BFWA) network interfaces have several drawbacks that can limit performance, cost and scalability when adapted for the Multimedia Wireless System (MWS) investigated. The survey covered in this chapter identifies some areas for investigation that need to be addressed to achieve the aims of this research. For example, it was identified that previous BFWA interfaces; (i) employed encapsulation methods that have inefficient framing and synchronisation mechanisms, and use excessive headers for encapsulating Ethernet frames (ii) have low operational speeds and high latencies which limited application performance (iii) have complicated methods to address internetworking issues over the MWS topology that impact equipment costs and scalability of a system.

The sections of this chapter are organized as follows. Section 2.1 reviews the fundamental data interface functions at the Datalink layer – also known as layer-2 of the OSI model (see Appendix A) [20]. Section 2.2 covers a review of Datalink layer encapsulation technologies in BFWA and broadcasting standards. Section 2.3 surveys existing BFWA trials in research literature with focus on the data interfaces employed and their internetworking mechanisms with core networks. Section 2.4, reviews the dominant local area networking technology, Ethernet, with which the novel MWS interfaces proposed in this research are designed to be more closely integrated and be part of existing data networks.

2.1. BFWA Datalink layer mechanisms

This section reviews the theories of underlying the mechanisms that are typically implemented at the Datalink layer of a BFWA interface. These Datalink layer mechanisms are also compared to those employed by the Transmission Control Protocol (TCP) and Internet Protocol (IP) as they are widely used by network hosts for reliable communication. Special attention was paid to the basic mechanisms implemented in the investigated MWS interface which are; segmentation and reassembly, frame encapsulation and synchronisation, header suppression, media access control, and flow control. Other advanced Datalink layer mechanisms such as source coding and encryption were also reviewed. As these advanced mechanisms are considered optional they are not implemented in the current hardware developed in this research. However, reservation of flags in the header space are provisioned for these mechanisms in the proposed protocols framing structure for future use.

2.1.1. Segmentation and reassembly

A Network Layer interconnection device that implements the IP protocol achieves transparency by IP's *'fragmentation and reassembly'* mechanism which can break larger packets into smaller ones that will not violate the MTU of the subnet [21]. The similar function that is implemented at the Datalink Layer is called *'segmentation and reassembly'*[22]. Subnetworks can have unique quirks such as proprietary framing or frame size limitation. Segmentation and reassembly allows a subnetwork to forward packets although it has a maximum Datalink layer frame size that is smaller than the Network layer's packet MTU. A transparent Datalink layer subnetwork is desirable because the hosts do not need additional protocols or facilities to use it.

To illustrate that fact, consider the network architecture in Figure 2-1 that consists of two hosts connected via three internetworking devices; one operating at the Network Layer and two at the Datalink layer. When Host A creates a packet destined for Host B, it will forward the packet to its default IP router, which is the Network Layer internetworking device in the figure. The IP protocol implemented in Host A will detect the MTU of IP subnet 1 and will create a packet that does not violate this value. The IP router will

forward the packet to Host B by fragmenting it into smaller packets that does not violate the MTU of IP subnet 2. Although the maximum frame size of subnetwork 3 is smaller than the MTU of subnet 2, the Datalink Layer interconnection devices can still relay the packet by transparently segmenting the fragments. This process is transparent to the router and Host B i.e. both can transmit packets up to the MTU size of Subnetwork 2 and 4 without regard to the frame size limitation of subnetwork 3.

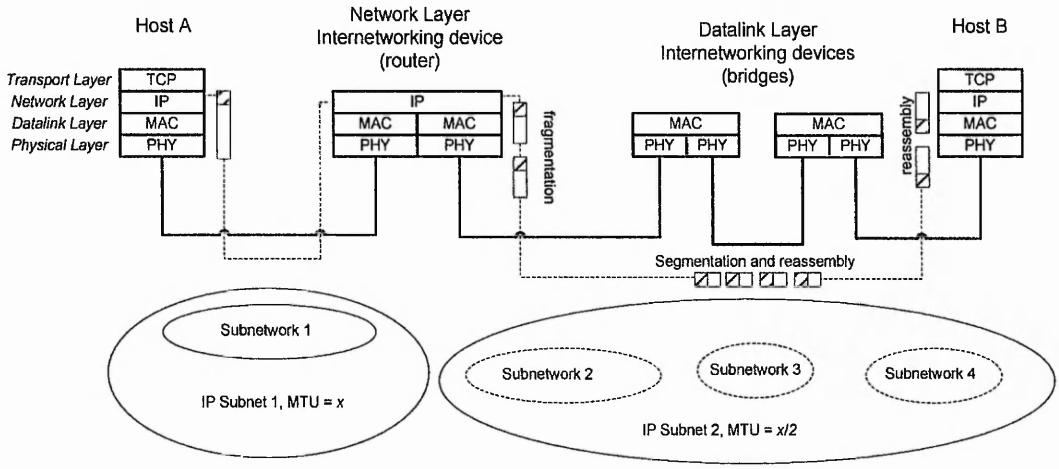


Figure 2-1: Segmentation and assembly mechanisms

The Segmentation and reassembly function establishes a transparent bridge across an intermediate network. This can be visualised in the figure above where a transparent bridge is formed over subnetwork 3 which requires subnetworks 2 and 4 to be of the same technology. In the segmentation and reassembly process done in a MWS interface, two sub-processes have to take place before a frame can be arbitrarily segmented and framed into the framing structure of an intermediate subnetwork. These sub-processes are *Frame encapsulation* and *synchronization* will be discussed in the next section.

2.1.2. Frame Encapsulation and Synchronization

The *frame (re)synchronisation*[1] [23] sub-process is needed in the case where a new receiver is joining in a transmission or one that has just recovered from a link-breakdown event to (re)start the reassembly of frames [24]. Before synchronisation, the receiver will only know of the transparent subnetwork's frame structure but not necessarily the location of the encapsulation markers. Once the encapsulation scheme's structure is established by locating the first encapsulated frame (and its encapsulation markers) that is embedded in the payload, the resynchronization process is completed successfully and the subsequent frames can be located using the encapsulation scheme's own mechanisms.

The *frame encapsulation* sub-process that is used to mark the start and end of a frame so that a receiver can reassemble the original frame from a transparent subnetwork's framing structure. As the markings are performed in an in-band manner, the encapsulation procedure employs an algorithm to differentiate the encapsulation markings from the data of the frame itself. The author categorises byte-wise[2] [25] frame encapsulation algorithms into two categories as shown in the taxonomy in Figure 2-2.

---

[1] The ability of an encapsulation algorithm to resynchronize is essential for a MWS. The typical availability of a 42 Ghz link is deployed with a link budget for 99.9% availability (see reference in main text). Thus it should be expected that a link will fail during operation. In order for the link to be robust, the segmentation and reassembly mechanism should be able to detect the error and resynchronize to the base station broadcast.

[2] Bit-wise or non-byte sized algorithms are beyond the scope of this review. The reason for this is based on the assumption that broadband (high-speed) networks normally use octets as atomic data units in their framing structures.

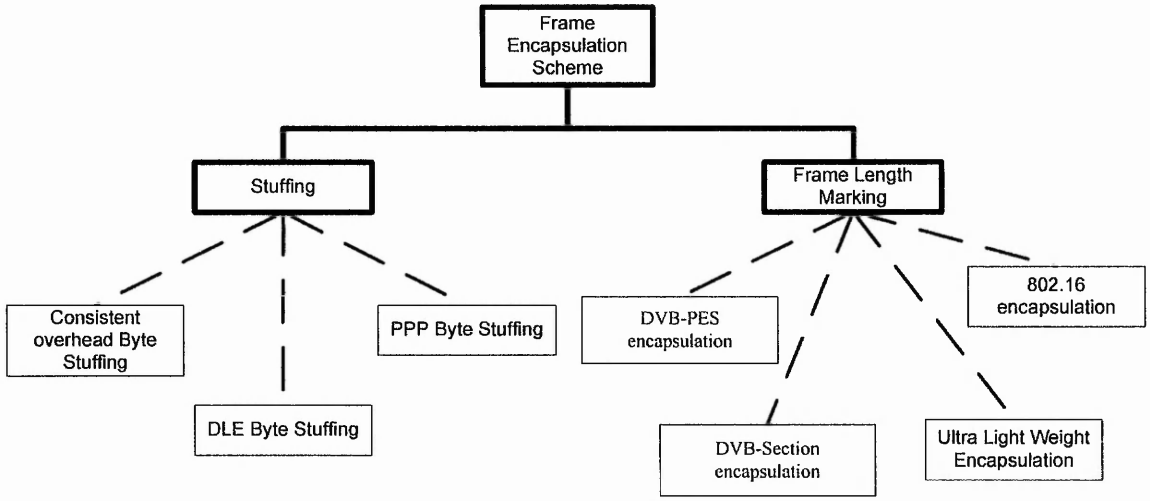


Figure 2-2 Taxonomy of Frame Encapsulation Schemes

The first category Frame Length Marking (FLM) algorithms refer to encapsulation procedures that mark the frame boundary using a frame-length field that is usually located in the header of a frame. The end of a frame can be determined since the frame-length field holds the number of bytes of the frame being encapsulated, in the form of a binary number. The second category, Byte Stuffing (BS) algorithms, refer to encapsulation algorithms that mark the boundary of a frame using reserved byte value(s). In BS, frame boundaries are marked by placing the reserved byte value at the start and end of the frame being encapsulated. To achieve transparency where any byte value can be used in the payload of a frame, Byte stuffing algorithms must also perform a lossless transformation to differentiate the boundary markers from reserved byte values occurring in the payload data. Details of the frame encapsulation and resynchronisation mechanisms of FLM algorithms can be found in sections 2.2 and 2.3 while details of BS mechanism can be found in section 3.1.

2.1.3. Media Access Control and Flow control

The Media Access Control (MAC) and Flow control are essentially traffic regulation mechanisms that should work together to ensure optimal and fair usage of the available bandwidth of the medium. The MAC and flow control mechanism employed by the MWS interface and interfacing networks can skew the performance of MWS interface in

a real networking environment which should be understood to avoid significant impact to efficiency.

Although MACs have other roles in a network interface, achieving fairness is a major goal which is done by deciding which node gains access to the shared medium. In the MWS topology investigated, aggregation of traffic occurs only at the base station where the MAC should maintain the quality of service and fairness [26,27]. The MAC can employ a scheduling algorithm e.g. Round Robin or First Come First served. For the prototype MWS base station interface developed in this research, a simple algorithm will suffice for a small number of subscriber interfaces. However, more complex algorithms might be needed to take into account the priority of the traffic [28] if quality of service is to be maintained as the number of nodes are increased.

Flow control mechanisms generally regulate the amount of traffic a on a network path. The scope of a path depends on which layer of the protocol stack the flow control mechanism is operating at. While flow control of data traffic usually involves the TCP mechanism at the Transport layer, real-time streaming traffic such as voice or video usually employs its own flow control schemes at the Application layer that bypasses the TCP mechanism. However, flow control mechanisms may also be found at the Datalink layer which influences traffic congestion in a network. A brief description of the Internet Model's TCP and Datalink control mechanisms are discussed in the following paragraphs.

The term 'congestion control' in data communication normally refers to a Transport Layer flow-control mechanism [29]. The most commonly deployed congestion control mechanism is TCP/IP's window based mechanism which regulates the flow of traffic between two hosts of a network/internetwork[30]. While such Transport Layer flow-control operate on an end-to-end (or host-to-host) basis, the equivalent may be implemented at the Datalink layer on a hop-by-hop basis i.e. between two network ports connected to the same medium [31]. Different literature uses the terms flow control, back pressure and congestion control interchangeably. In this thesis, *'congestion control'* will always refer to TCP's mechanism while *'flow control'* will refer to Ethernet's mechanism to avoid ambiguity as illustrated in Figure 2-3 below:

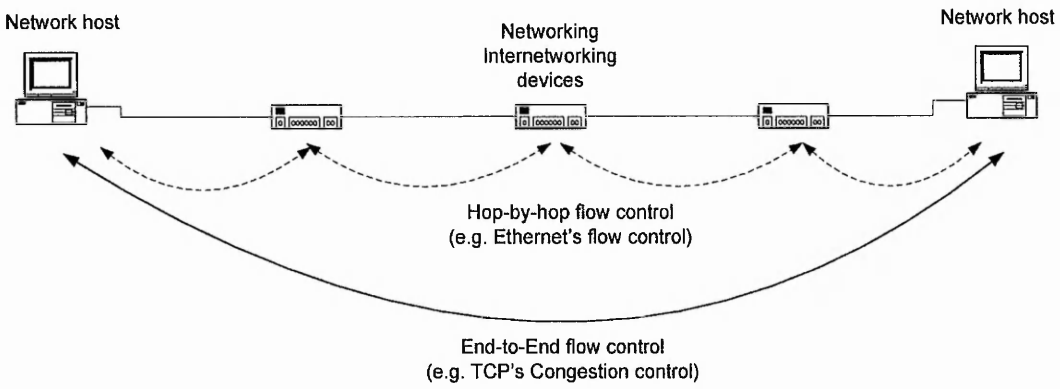


Figure 2-3: Ethernet flow control and TCP congestion control mechanisms

Some work has suggested that backpressure complements congestion control while others believe it causes the congestion to spread to other parts of the network due to head-of-line blocking effects [32]. Nevertheless, the fact that backpressure can prevent packet loss (due to congestion) is not disputed. As packet loss wastes both valuable processing power and buffer space in interconnection devices as well as network bandwidth, avoiding it is particularly benefiting to MWS systems where such resources are more costly than in Local Area Networks (LANs).

2.1.4. Source Coding and Encryption

A major benefit of digital communications is that it is easier to change the form of the digital information source with electronics prior to transmission [33]. *Source coding* can be used to reduce the average number of bytes required to transmit the source digital information. Elsewhere, *Encryption* can be used to code messages using a cipher to prevent unauthorized hosts making use of the information. Source coding and encryption at the Datalink layer are possible at the expense of additional delays and processing power [34,35].

Two categories of source coding that are usually implemented at the Network/Datalink layer include *header suppression* and *block compression* [36]. Block compression algorithms constructs a dictionary of common sequences within the data of a frame and matches each sequence to a shorter (compressed) representation. However, applying block compression algorithms at the Datalink layer was shown to give insignificant

benefits, and even negatively impacted efficiency, when higher layer compression (e.g. ZIP, IPCOMP, or multimedia CODECs) had already been performed on the data [36].

Header Suppression works by exploiting the redundancy which is often present in successive frames in the same flow [36,37]. Tye, Fairhurst, Perkins and Mutka [36,37] had shown that while header suppression can reduce the original size of frames, efficiency gains are more significant on smaller frames where the headers are large compared with the payload. An issue of decreased robustness that is associated with header compression was claimed to increase the packet loss of a data communication link [38]. In order for a wireless link with high losses to take advantage of header compression, it would have to trade off communication range of the wire link to ensure that bit error rates do not increase beyond a threshold. In that work, a bit error rate of about $1x10^{-6}$ was calculated to be the maximum error rate that can be tolerated for header compression to be beneficial. In the MWS investigated, DVB based physical layers are used that can guarantee quasi-error free wireless links (bit error rates of $1x10^{-11}$). This is more than sufficient for header compression to be beneficial. Further, as the concatenation of forward error correction techniques employed by the DVB physical layers induces very steep bit error rate degration behaviour, impact to communication range is negligible. For this reason, header compression is viewed to be very suitable for the MWS investigated. Further exploration of the use of header compression in the MWS interfaces investigated can be found in subsection 4.3.3.

Using Encryption at the Datalink layer brings up similar dilemmas as block compression. Encryptions done at a higher layer (at a host) are generally much more powerful than those that can be implemented in internetworking devices. Taking the 802.11 wireless LAN technology as a case study; it was shown by workers in the field [39,40] that encryption implemented at the Datalink layer was weak. Also, since security class information is usually not available at the Datalink Layer, encryption will be performed on all frames regardless whether they contain sensitive information or not. Hence, using encryption at the Datalink layer of a MWS may amount to wasting resources when data is not sensitive, or giving a false sense of security when the encryption was weak.

Summary of Section 2.1

This section reviewed the mechanisms that can be theoretically implemented in a MWS interface at the Datalink layer. While certain mechanisms are necessary, some are non-essential or undesireable when used in conjunction with existing networks that are likely to be integrated with MWS or existing protocols such as TCP/IP.

In conclusion, where design of the MWS interface is concerned, the use of the correct encapsulation scheme, media access and flow control, source coding, encryption must be carefully considered to avoid negatively impacting the efficiency, reliability or latency of the MWS. The effectiveness of each mechanism also has to be considered as implementing complex mechanisms can unnecessarily increase the cost of the MWS interfaces, especially where processing power and memory are scarce.

2.2. BFWA Interfaces Specified in Standards

This section surveys the 'segmentation and reassembly' and 'frame encapsulation and synchronisation' mechanisms recommended in standardisation literature which are specifically for BFWA, as well as mechanisms from the broader scope of networking and broadcast standards commonly adopted in first generation BFWA trials. The goal of standardization is to enable equipment from different manufacturers to interoperate. If a standard becomes widely accepted, economies of scale can be realised from mass production of equipment. Specific standardization literature for BFWA systems was only published recently and hence equipment manufactured based on them are scarce. Research and commercial trials therefore adapted equipment from other networking and broadcast systems such as core ATM, cable or satellite networks which were commercially available. The timeline of the standards related to BFWA are shown in Figure 2-4 based on information from official websites [41, 42, 43, 44, 45, 46, 47].

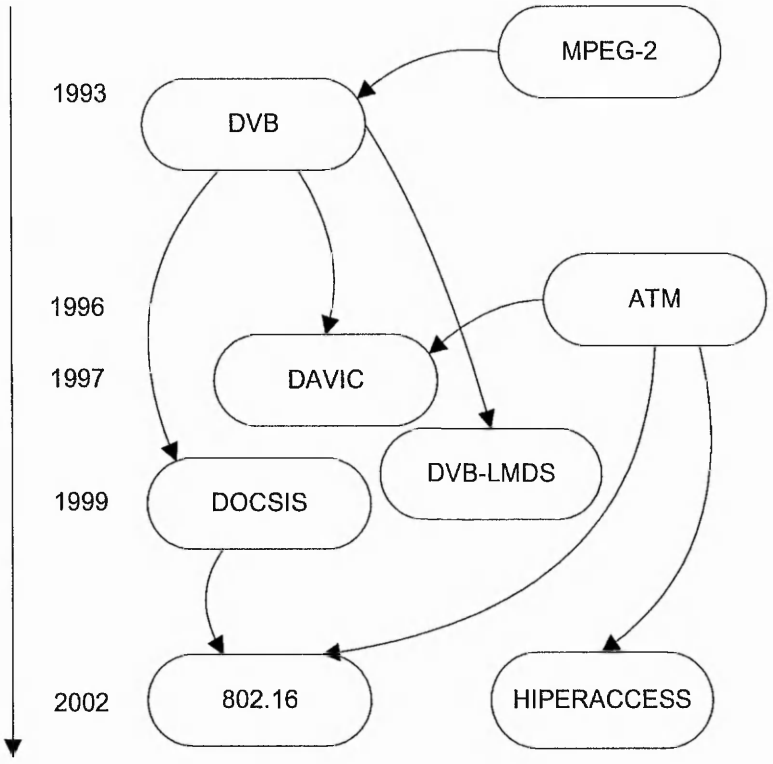


Figure 2-4: Timeline of Standards related to BFWA

Recent specific standards for BFWA mainly scoped around the Datalink layer (MAC layer), which were developed by three major international technical groups; DVB, ETSI and IEEE [48]. DVB published the DVB-LMDS standard which was in fact an adapted combination of DVB-S on the downlink and DVB-C on the uplink [49]. ETSI's standard known as HIPERACCESS was relatively slow to be published which was probably due to its limited commercial interest. Before the HIPERACCESS standard was completed, the IEEE began and completed the bulk of the IEEE 802.16 standard between the years 2000 and 2003 that is sufficient for standardized equipment to be manufactured.

As mass scale production of equipment based on specific standards for BFWA systems were unavailable, equipment based on legacy technologies particularly from satellite and cable networks were used frequently in previous BFWA trials. These equipments found their way into previous BFWA trials because of similar topology, and suitable features such as those provided by MPEG and ATM technologies. Therefore it is unsurprising that the newer specific standards for BFWA were drafted based on these legacy technologies.

2.2.1. DVB: MPEG Transport Stream and SPI interface

Although the DVB Transport SAP is defined as a service interface in DVB, it can be more easily understood as a Datalink/Physical layer interface in a DVB system - analogous to ATM cells in an ATM network. DVB Transport SAP structure has been adopted from the MPEG standards, which is why the DVB Transport is also commonly referred to as MPEG- Transport Stream (MPEG-TS). The DVB-SPI (Digital Video Broadcasting – Synchronous Parallel Interface) specification is an electrical interfacing specification over which MPEG-TS cells are carried. These two DVB standards were commonly implemented in previous systems, as well as in the novel MWS interfaces designed in this research, together with DVB equipment such as DVB-S Modulators and Decoders.

Unlike other SAPs that are described in the DVB standard, the Transport SAP does not have segmentation and reassembly mechanisms. In fact, various segmentation and reassembly from other standards (including the other DVB SAPs) are defined to map onto the Transport SAP to access the DVB Physical Layer. Since the MPEG-TS frames have a fixed size of 188 bytes including headers, some literature [50] refer to them as cells instead of frames. This convention is followed in this thesis.

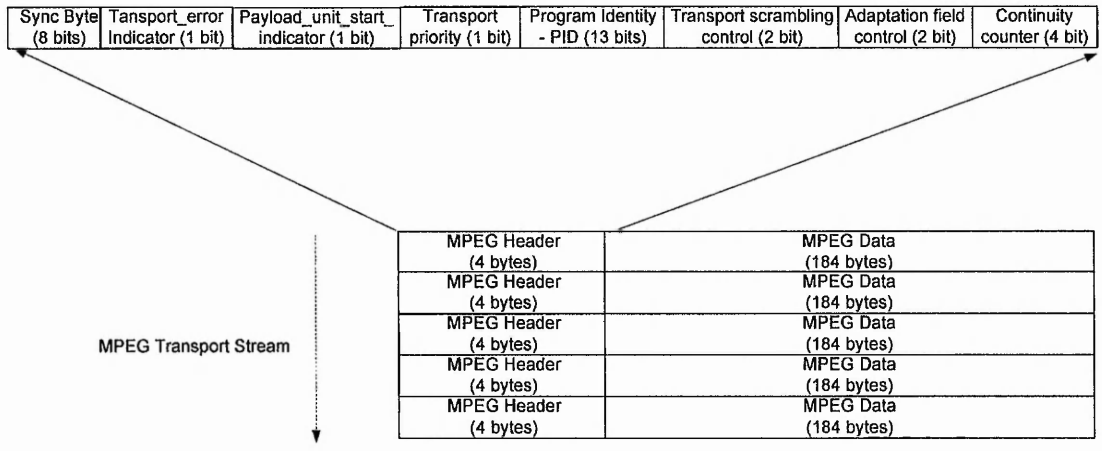


Figure 2-5: Motion Pictures Expert Group (MPEG) - Transport Stream (TS) Structure

The MPEG-TS was primarily designed to carry Digital video, however specifications for data transport, referred to as 'private data' in the ISO standard, are provided [51]. Data transport may use all the fields of the MPEG header in a service private way except for Sync_byte and program identity (PID). The PID field indicates the type of data stored in the packet payload. This is used by a segmentation and reassembly decoder to differentiate the cells that are destined for it. A unique PID can be registered with the standardisation body according to the PID table, Table 2-1, so that it will not be used by any other service.

PID value	Description
0x0000	Program Association Table
0x0001	Conditional Access Table
0x0002 – 0x000F	Reserved
0x00010 – 0x1FFE	May be assigned as network_PID, Program_map_PID, elementary_PID, or for other purposes
0X1FFF	Null packet

Table 2-1: Standardised PID value allocations

The sync_byte has a fixed value of '0100 0111' (0x47), which is used by the demodulator to synchronize with the MPEG-TS. Also according to the standard, macro synchronization of the cells may be performed after encoding by inverting every eighth sync byte.

Figure 2-6 below illustrates the signal and waveform of the DVB-SPI standard interface to transport MPEG-TS cells in 188 and 204 formats. The DVB-SPI is specified to use standard 25-pin sub-d physical connectors, as found in standard parallel ports used on PCs. For TTL (Transistor-transistor logic) DVB-SPI interfaces, one pin represents one signal line; eight pins are used for SPI Data bus, one pin for SPI P_Sync, one pin for SPI_Dvalid, one for SPI_Clock and one pin for Ground. In LVDS (Low Voltage Differential Signal) DVB-SPI interfaces each signal line requires two pins. Therefore twice the number of pins is used in LVDS DVB-SPI compared to TTL DVB-SPI. SPI P_Sync signal signifies a synchronization pulse which is only asserted as logic '1' during the first byte of MPEG-TS header, the Sync byte. Whilst SPI D_valid (Data valid) is

always asserted logic '1' in 188 byte mode, it is asserted as logic '0' for 16 bytes of the Reed-Solomon (RS) field in 204 byte mode.

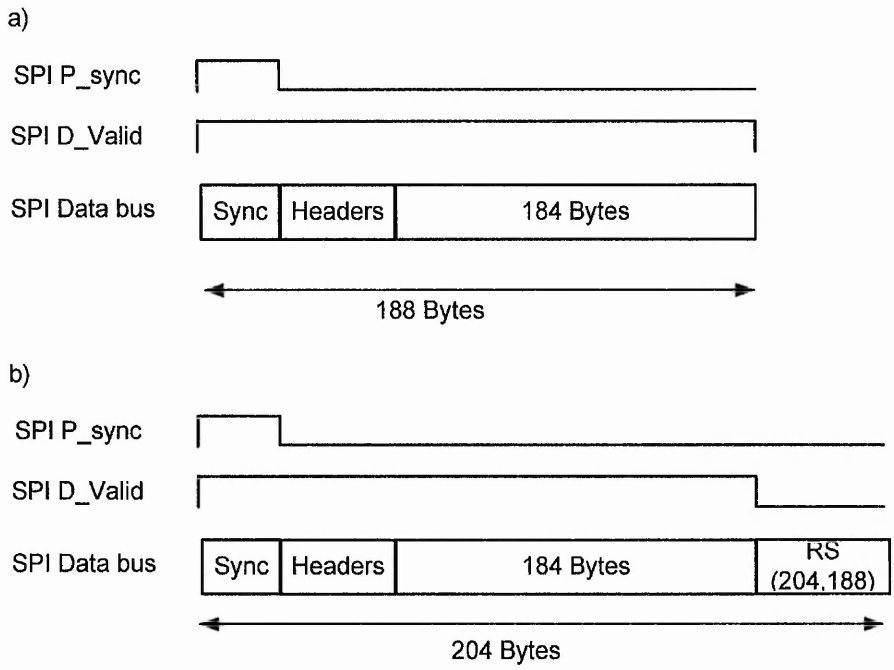


Figure 2-6: DVB-SPI signals for transporting MPEG-TS frames in; (a) 188 format, and (b) 204 format

2.2.2. DVB: Packetized Elementary Stream (PES) encapsulation

The mapping of the PES frame into MPEG-TS cells is defined in the MPEG standard [51] and adopted by DVB [52]. Some DVB based satellite systems view the Packet Elementary Stream (PES) SAP as the ideal SAP to insert data because it supports maximum sized IP packets (i.e. up to 65536 bytes). This means that IP packets can be directly mapped without fragmentation onto a DVB system. However, this is false economy when taking the mapping between the PES frames and the MPEG-TS into consideration i.e. the segmentation and reassembly of PES frames to/from MPEG-TS cells. Figure 2-7 illustrates the mapping of PES and MPEG-TS according to the standards [51,52].

PES encapsulation is a FLM based mechanism using the PES_packet_length field (Label A). The encapsulated IP packet or bridged frame is treated as the payload of the PES frame (Label B). Synchronisation is achieved by aligning the start of each PES Stream Segment to always appear after the MPEG header. Therefore, the cell that carries the last byte of the PES frame is padded with padding bytes (0xFF) to complete the MPEG/DVB-TS cell. The next PES packet starts at the beginning of a new MPEG/DVB-TS cell. The payload_unit_start_indicator (PUSI, Label C) will be set to '1' to indicate the first PES segment and '0' for middle and last PES frame segments.

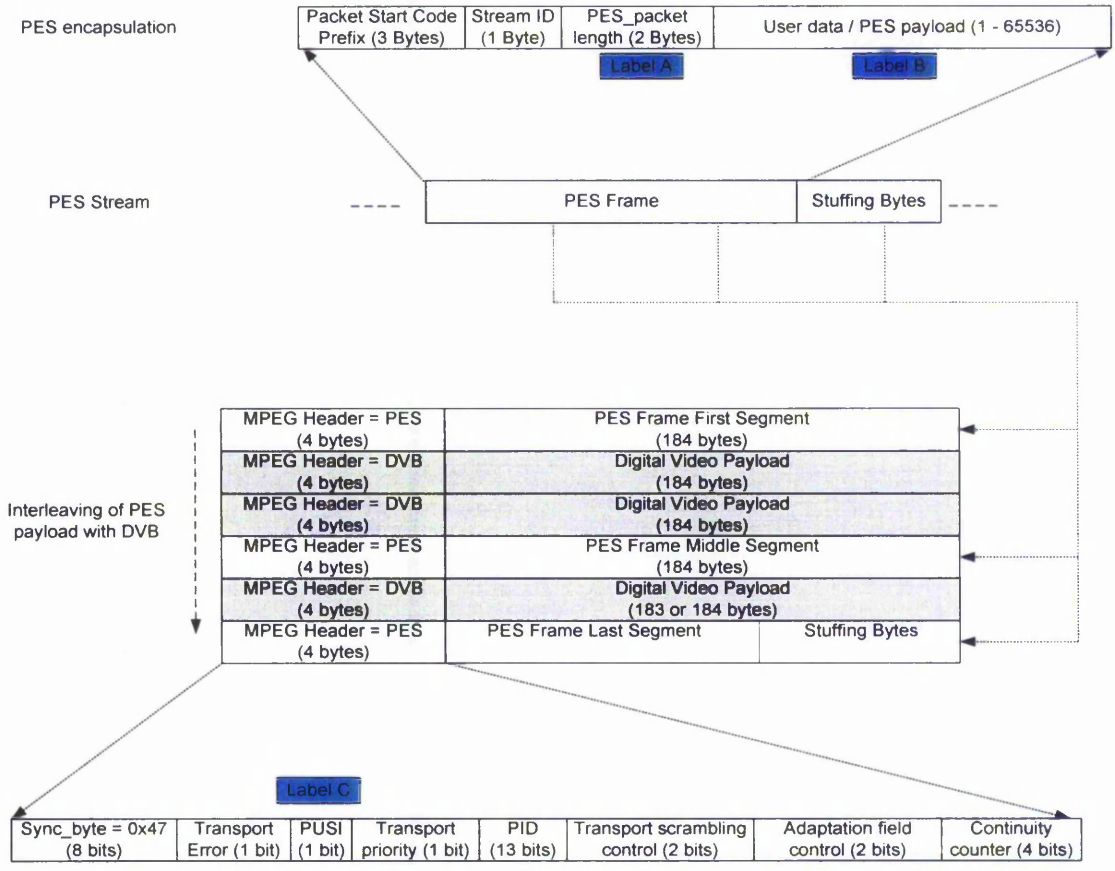


Figure 2-7: 'Digital Video Broadcast (DVB) -Packet Elementary Stream (PES)' encapsulation scheme

The inefficiencies of PES framing can be exemplified in the scenario of two back-to-back 64-byte IP packets arriving at the interface. After each frame is encapsulated in PES frames, it will be passed to the segmentation process. Because each of the PES frame has to begin in a new MPEG-TS frame, they are carried in two MPEG-TS cells.

120 stuffing bytes will be added to the PES frame in each MPEG-TS cell to achieve 184 bytes of payload. In total, this amounts to 246 bytes of overhead (including the PES headers) for transporting the two IP packets.

2.2.3. DVB: Section encapsulation

A segmentation and reassembly mechanism called Multi Protocol Encapsulation (MPE) was introduced in the DVB standard as an alternative method to encapsulate IP packets over the Section SAP. The Section SAP was introduced in the MPEG standard primarily to broadcast control information in a form of System internal Information (SI) tables [53]. The broadcasting station will transmit the SI tables at regular intervals to maintain an updated list of the channels. The subscribers will then use the SI tables to extract information needed to understand and organize the channels.

The DVB specification of MPE is actually the extension of *private* SI tables defined in the MPEG standard - specifically the 'datagram section' compatible with the 'DSMCC section' for private data [54]. The protocol structure in Figure 2-8 summarises the MPE encapsulation mechanism described in the DVB standard [54] and the mapping of the section into MPEG-2 Transport Stream packets is defined in MPEG standard [51].

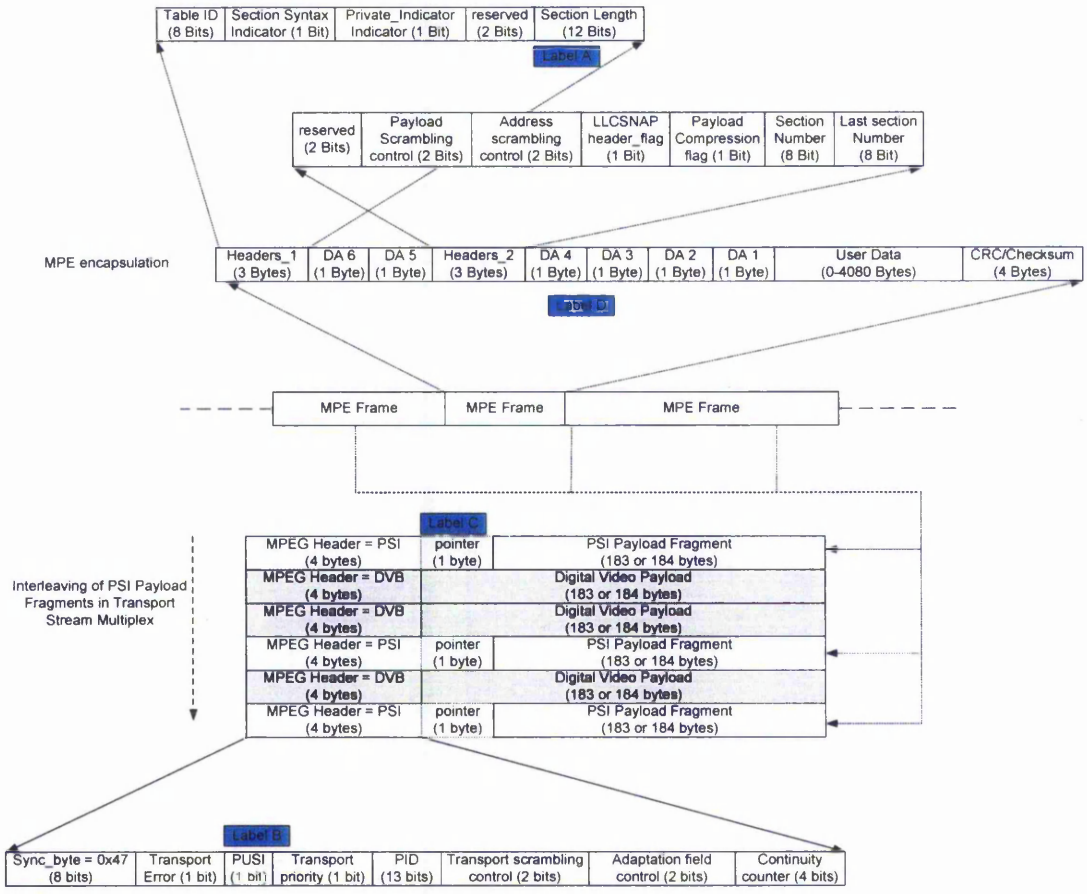


Figure 2-8: 'Digital Video Broadcast – Section' encapsulation scheme

The MPE encapsulation uses a frame length marking mechanism with the section_length field (Label A). Since the section_length field is 12 bits, IP packets up to the size of 4080 bytes can be carried while larger packets has to be fragmented by IP. The MPE frames are packed back to back (without padding bytes) and segmented into MPEG-TS cells.

Synchronization is achieved using the PUSI field (Label B) to indicate (set to '1') the presence of a pointer immediately after the header. The pointer will point to the first MPE frame that starts in the MPEG-TS cell. The payload size will be adjusted to be 183 bytes in the presence of the pointer field (Label C) and 184 bytes without.

When no data is available at the encapsulator buffers to start a new segment in the same MPEG-TS cell, pad_bytes will be used as padding to complete the cell. Since pad_bytes always takes the value of 0xFF, the Table_id of that value is forbidden [51].

The superior segmentation and reassembly mechanism of the Section SAP makes it the most efficient mechanism specified by DVB [55]. Therefore, equipment manufacturers offering data integration devices for DVB systems tend to implement MPE [56]. Although MPE was carefully designed, there are still some reservations by workers in the field about the use of MPE for encapsulating variable sized frames or packets [56]. A work on encapsulation of IP into DVB for satellite network criticized MPE as being "neither elegant nor efficient" [57]. They were probably referring to the unnecessary fragmentation and mirroring of the MAC destination address (DA; Label D) and excessive headers of MPE. Furthermore, MPE had explicitly required the filtering (removal) of MAC source address which had negative impact on certain functions at the higher layers [58]. A work related to the MWS project EMBRACE [59,], had defined a version of MPE that reintroduced the source address. This work will be discussed in more detail in Section 2.3.6.

2.2.4. Cable Labs: DOCSIS encapsulation

Although the data-over-cable service interface specification (DOCSIS) is designed for interoperability among cable modems and related products, with a few modifications, it can also be used in MWS environments [60]. DOCSIS defines its own segmentation and reassembly which maps onto the DVB physical layer. The specifications can be found in separate publications: the Downstream Transmission Convergence Sublayer section [61] and the DOCSIS MAC section [62] which are combined and summarized in Figure 2-9.

User data from the data interface are encapsulated in DOCSIS frames (Label A). A completed DOCSIS frame will be appended to other DOCSIS frames in a buffer to be passed to the Convergence Sublayer. The Convergence Sublayer will segment the DOCSIS frames into the MPEG-TS cell format. The frame-length-marking encapsulation scheme uses the 2-byte LEN_field (Label B), which provides the information to the receiver where a DOCSIS MAC Frame ends and where the next begins. Although the 2-byte frame-length-marking can theoretically encapsulate IP packets or bridged frames size of up to 65536 bytes, the DOCSIS limits this value to 1518 bytes. This specification was probably made so that cable modem devices that are

based on this standard would not require the implementation of large buffers to process frames larger than 1518 bytes.

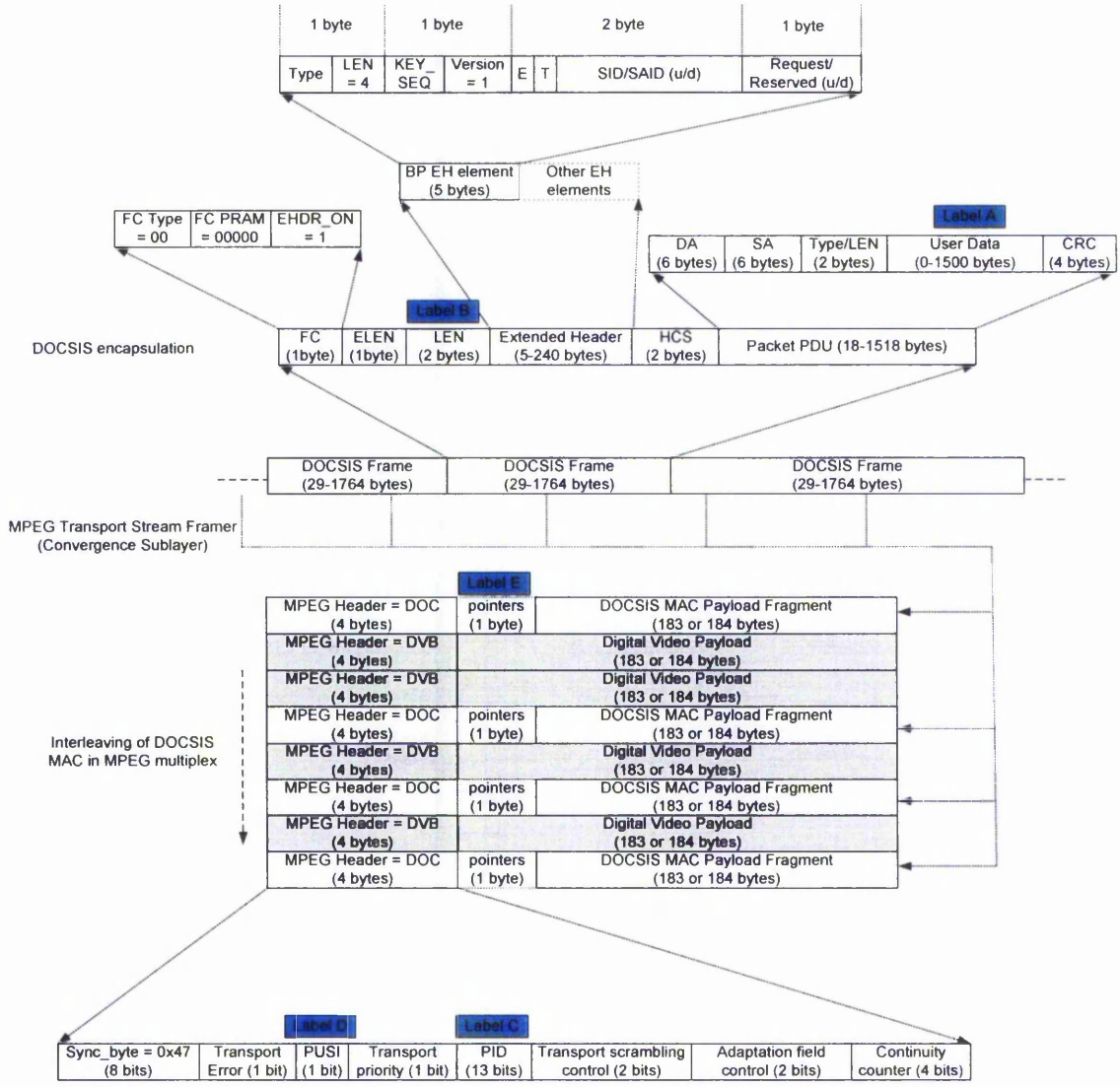


Figure 2-9: 'Data Over Cable Service Interface Specification (DOCSIS)' Encapsulation

The PID used in the MPEG-TS cells that contain encapsulated DOCSIS frames has the well-known (registered) value of 0x1FFE (Label C, also see Table 2-1). Once DOCSIS encapsulates the user data into DOCSIS MAC frames, they can be packed back-to-back

without padding. This also allows the DOCSIS MPEG Transport Stream framer to arbitrarily segment the packed DOCSIS MAC Frames without regard to the boundaries.

The synchronisation mechanism of DOCSIS is similar as in DVB Section MPE. The PUSI (Label D) bit of value '1' indicates the presence of the Pointer field (Label E) after the MPEG Header. The Pointer points to the location of the first byte of a DOCSIS MAC Frame where the Length Field can be found with a 2 byte offset. When the PUSI value is set to '0', this indicates that there is no pointer field in the payload of the current MPEG Transport Stream cell, which implies that all 184 bytes are a middle segment of a DOCSIS frame. The receiver has to look at the next MPEG Transport cell to find the beginning of a DOCSIS frame.

2.2.5. IETF draft: Ultra Lightweight Encapsulation

In 2002, a working group part of the Internet Engineering Task Force (IETF) set out to define an optimised encapsulation mechanism specifically for MPEG-TS in parallel with this research. ULE is still in progress with refinements being regularly drawn out by the 'IETF working group: ULE' which are constantly updated on the IETF website [63]. This thesis reviews the most current version at the point of writing on Ultra Lightweight Encapsulation (ULE) [56].

ULE improves the mechanisms specified in the DVB, DAVIC or DOCSIS standards by removing unnecessary headers while explicitly defining mappings for a number of both packet and frame types. The encapsulation and synchronisation processes are the same as the DVB Section MPE and DOCSIS mechanisms. However, ULE's frame length marking uses a 15 bit length field (Label A) instead, which allows payload of up to 32766 bytes. Like the DVB Section MPE mechanism, padding_bytes (referred to as stuffing bytes in the specification) are used to complete a MPEG-TS cell should there be no frames waiting in the buffer.

At the receiver, the existence of padding can only be determined by successfully scanning two consecutive padding_bytes after the end of the previous encapsulated frame segment. Two consecutive padding_bytes i.e. 0xFF 0xFF denotes an End Indicator

which means that there are no further encapsulated Ethernet frames within the current MPEG-TS cell. The combined D and length fields of the ULE structure are forbidden to take the value of 0xFFFF to avoid the emulation of those padding_bytes. Although there are a few other special cases to consider which are explained in the internet draft, the basic framing structure is illustrated in Figure 2-10.

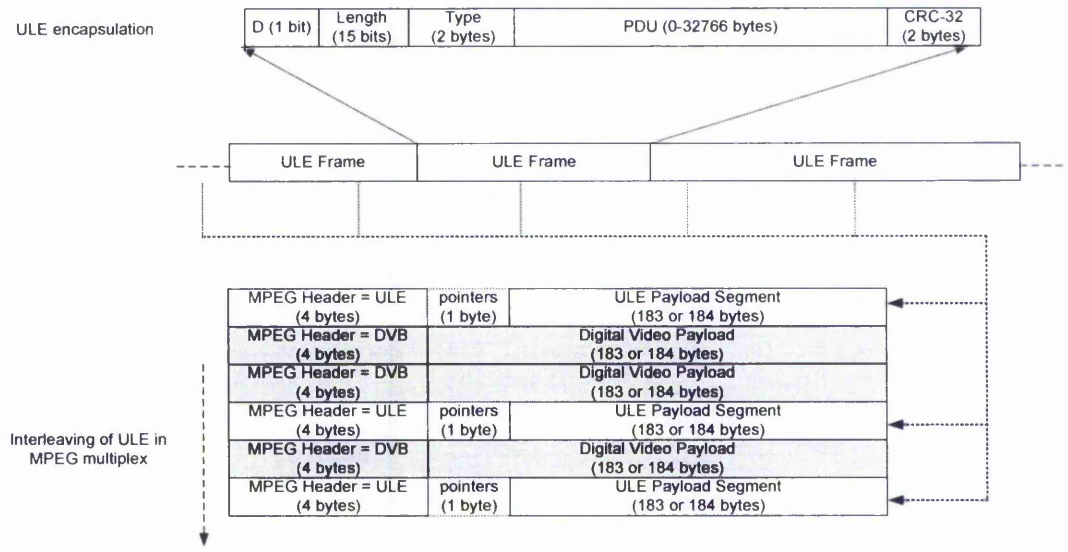


Figure 2-10: Ultra Lightweight Encapsulation (ULE)

ULE maintained a minimal set of headers and so it can be considered as a 'no frills' version of the DOCSIS mechanism. Yet the Type field can be used to indicate the presence of additional headers, effectively extending its functionality as and when needed. This makes ULE the most efficient mechanism among the MPEG-TS based mechanisms reviewed. The following subsections will discuss some BFWA standards that are non-MPEG based.

2.2.6. DAVIC and ETSI: HIPERACCESS

Digital Audio Video Council (DAVIC) is a non-profit association registered in Geneva, Switzerland. The DAVIC standard is a result of the collaboration of two hundred and twenty two international companies aiming towards end-to-end interoperability of broadcast and interactive digital audio-visual information, and of multimedia communication including specifications for BFWA systems [44]. The associated DAVIC

standard itself [64] does not specify any new Datalink layer protocol for its segmentation and reassembly mechanisms. For this, they have specified two options that are borrowed from other standards for downlink. These are either an in-band DVB-Section(MPE) mechanism or an out-of-band ATM downlink transmission. Readers can refer to the earlier Section 2.2.3 for the description of DVB-Section (MPE). The uplink mechanism also uses the ATM mechanism. The ATM encapsulation mechanisms used in DAVIC will be discussed in the following paragraphs, together with HIPERACCESS standard due to their similarity.

HIPERACCESS is ETSI's flavour of a BFWA standard [65]. The HIPERACCESS standard has departed from MPEG based standards and adopts the ATM standard for both its uplink and downlink Datalink Layer mechanisms. HIPERACCESS' Physical layer is specifically designed to support ATM where optimization is achieved by a one-to-one mapping of ATM cells to HIPERACCESS cells [66]. It is noted that the difference between the DAVIC and HIPERACCESS standards is that, the former standard's uplink uses full 53 byte ATM cells, where else the latter only uses 51 bytes with the HEC and VPI fields omitted.

As HIPERACCESS and DAVIC systems are designed for ATM core networks, they are very efficient in an ATM context. However, the transport of variable length IP packets or bridged Ethernet frames introduces overheads as ATM's segmentation and reassembly mechanism called ATM Adaptation Layer – 5 (AAL5) is required. ATM has other mechanisms which are AAL1, AAL2 and AAL3/4 but AAL5 is regarded the most efficient among them [67]. In Figure 2-11 the AAL5 encapsulation being mapped to HIPERACCESS cells mechanisms as described in the standards are illustrated [68].

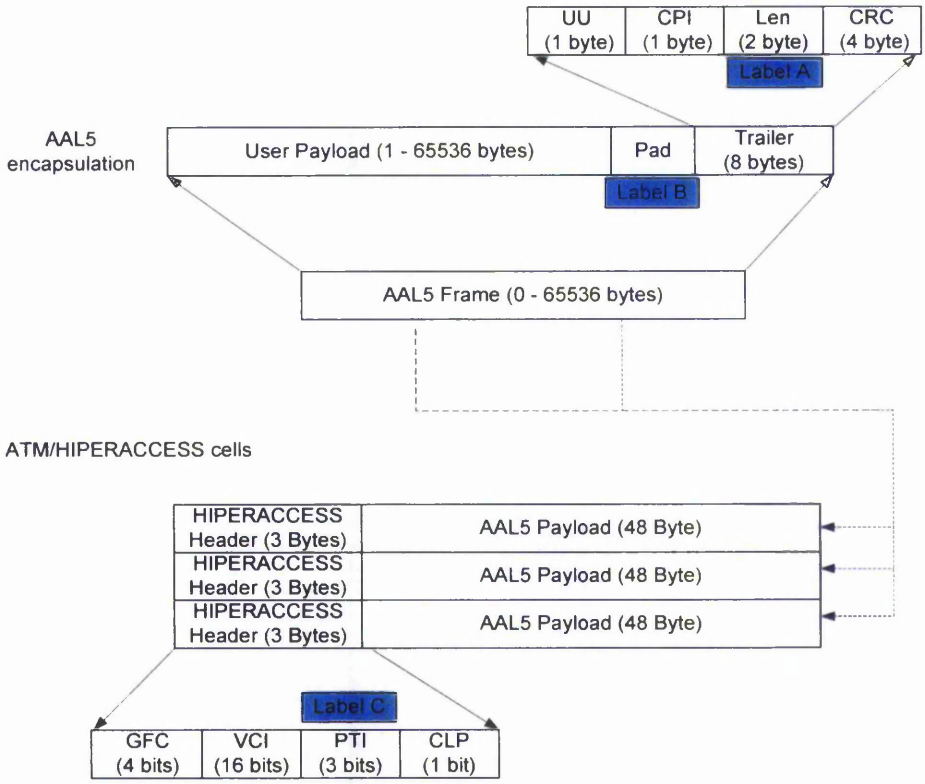


Figure 2-11: Asynchronous Transfer Mode (ATM) - Adaptation Layer 5 (AAL5) encapsulation scheme

A BFWA research project, CABSINET (see subsection 2.3.5), that uses both MPEG-TS and ATM cells in their system compared the framing overhead of the two structures in detail [69]. This work showed that MPEG-TS cells introduce 20% less framing overheads and argued that the "ATM cell is quite suboptimal for LMDS [as]... the incurred header overhead and transmission delay is a too high a price to pay". The fundamental cause of ATMs inefficiency in this argument is that the ATM cells are small and so require more cells, and thus more headers, to transport the same amount of data.

ATM AAL5 uses a frame length marking encapsulation method with a 2 byte Len field (Label A). Unlike the other mechanisms previously reviewed that normally place the Len field in the header, here, the Len field is placed as the 8 byte trailer. This was probably done to reduce store-and-forward latency as the entire frame can be transmitted while the frame length is being counted and then only concatenated at the end. The synchronisation mechanism uses padding (Label B) to align the trailer to always appear

within the last 8 bytes of a cell to ensure that receivers will be able to find it. Finally, the Payload Type Indication (PTI) field (Label C) in the cell header will then be used to indicate the cell that contains the trailer.

The consequence of placing a Len field placed at the trailer meant that AAL5 could not pack more than one encapsulated frame in an ATM cell. Remaining unused payload of ATM cells has to be padded to complete a cell, which impacts efficiency of the segmentation and reassembly mechanism. The table below illustrates the overhead of typical data traffic from padding alone. In the table, the packet distribution probabilities are assumed to be TCP/IP traffic similar to that in empirical model [70,71] or real traffic traces [72] that are normally accepted to represent typical traffic.

Packet length	64	128	256	576	1024	1518
Probability	0.50	0.01	0.01	0.05	0.01	0.2
Cells after AAL5 encapsulation.	2	3	6	11	22	32
Padding overhead (%)	33%	11%	11%	3%	3%	1%

Table 2-2 : Padding overhead for IP over ATM AAL5 Encapsulation

2.2.7. IEEE: 802.16 / WiMax

The IEEE 802.16 standard [73] is the most recent and promising BFWA standard. Although not yet in large scale deployment, industry leaders show strong support of IEEE 802.16 through the WiMAX Forum [6] that aims for equipment manufacturing interoperability. Although WiMAX currently focuses of BFWA systems below 11Ghz frequencies, the 802.16 standard was built from ground up for systems operating in all frequencies up to 66 GHz [74].

802.16 specified two Convergence Sublayers (CS); namely the ATM CS and the Packet CS, as shown in Figure 2-12. The ATM CS is specifically designed to transport ATM cells with service classification that maintains the ATM semantics. On the other hand, the Packet CS is used for the transport of all variable length protocols including IP

packets and bridged Ethernet frames. Instead of MPEG-TS cells used in DVB based systems, the ATM or Packet CS frame are mapped onto Codewords which are typically 255 bytes in length (the last Codeword in a burst transmission can be between 6-255 bytes).

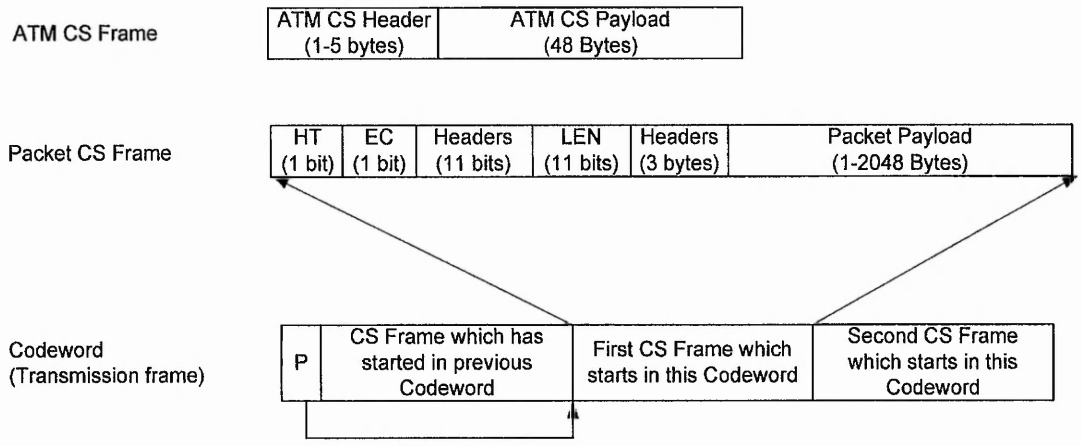


Figure 2-12: 802.16 encapsulation

The Packet CS frame is another example of a FLM encapsulation technique. The 11-bit LEN field can encapsulate user data carried as the Packet_payload by indicating its size, which can be up to 2048 bytes. The codeword synchronization mechanism adopted is identical to most of the MPEG-TS based encapsulation teechniques reviewed, such as ULE and DVB Section/MPE, which uses a pointer field, P, placed at the first byte of a Codeword. The pointer field will tell the receiving station the location of the first CS frame which starts in the current Codeword.

The advanced 802.16 Datalink/Physical Layers are envisaged to support; control messages, encryption, header-suppression, source coding and error correction; although some of which are optional [73]. The Packet CS Frame was designed to accommodate a variety of equipment types by allowing the indication of each supported mechanism using flag bits in the header fields. For example, an Encryption (EC) flag set as '1' indicates the use of encryption on the payload, while EC set as '0' indicates encryption was not performed. Likewise, a Header Type (HT) flag set as '0' indicates a Packet CS carrying user payload, while HT set as '1' indicates a control message for bandwidth

request. Header flags are also used to future proof the Packet CS Frame structure by reserving header flags for new mechanisms.

IEEE's view for BFWA applications as a data access network, rather than broadcast service distribution, can be evidenced by the lack of a convergence sublayer (CS) to transport the widely accepted MPEG-TS cells. The departure from an MPEG-TS based Datalink framing structure mean that MPEG based broadcast services are not inherently supported by 802.16. Nevertheless, if MPEG broadcast services are to be supported in 802.16 systems, MPEG-TS cells could be mapped onto either the ATM CS or packet CS. Techniques described by the ATM Forum [75, 76] or workers in the field [77, 78] could be used for this purpose, but, will inevitably induce further encapsulation overheads and impact efficiency.

Summary of Section 2.2

The encapsulation mechanisms of standards related to BFWA system are summarized and compared in Table 2-3.

Frame Length Marking Encapsulation: MPEG-TS based Interfaces	*Overhead per frame from encapsulation headers (bytes)*	*Overhead from Padding and pointer (bytes)*	*Max. Frame/ Packet size that can be encapsulated (bytes)*	*Sync. mechanism*
IETF-ULE	6	1	32766	Pointer
DVB-PES	6	0-183	65536	Padding
DVB-Section (MPE)	10	1	4080	Pointer
DAVIC-downlink	10	1- 44	4080	Pointer
Cable Labs-DOCSIS	17	5-240	1500	Pointer
Non-MPEGTS based Interfaces				
802.16	6	1	2048	Pointer
HIPERACCESS	8	1- 44	65536	Padding
DAVIC-uplink	10	1-144	65536	Padding

Table 2-3: Summary of standardised encapsulation schemes related to BFWA interfaces

The first observation is that all the standards reviewed employ Frame Length Marking and none use a Byte Stuffing scheme for their encapsulation method. Among the MPEG-

TS standards, ULE is the most efficient as it has only 6 bytes of header, and uses 'pointers' for synchronization. Although DVB-PES encapsulation has as little header bytes as ULE for its encapsulation scheme, its synchronization method uses padding which is very inefficient. The use of padding for synchronization would deem DVB-PES as the least efficient among the MPEG-TS based encapsulation. Therefore, in descending order of efficiency for MPEG-TS based schemes are IETF-ULE, DVB-Section/MPE, DAVIC-downlink, CableLabs-DOCSIS and finally DVB-PES. Among non-MPEGTS based schemes, 802.16 is the most efficient followed by HIPERACCESS, and finally DAVIC. However among all schemes only DVB-PES, HIPERACCESS and DAVIC-uplink can support maximum sized IP packets and hence can avoid IP fragmentation at end hosts.

2.3. Interfaces and Internetworking Methods of previous BFWA Trials

This section reviews five recent BFWA trials representing worldwide academic research in this field with publications detailing the inner workings of their systems. These trials give unique insights and experiences to network interfacing and internetworking given the wide range of existing networking technologies and software. Furthermore, as these trials also tend to adapt equipment from cable and satellite environments for their associated cost benefits and availability, experience gathered in long term operation and maintenance of their systems can be learnt. While the experiences, where relevant, are invaluable to this research, low frequency and high frequency BFWA systems should not be directly compared as the bandwidth, architecture, and capacity are different [5]. In this thesis, MWS exclusively refers to a 42GHz BFWA system (see diagram in Appendix A).

2.3.1. CRABS MWS Trial

The Cellular Radio Access for Broadband Services (CRABS) project was a collaborative research and technology development supported by the European Advanced Communications Technologies and Services (ACTS) programme. The project ran from 1996 till the end of 1999 which produced detailed deliverable reports and working demonstrations of Multimedia Wireless Systems [79]. The demonstrations included

provision of broadband interactive digital television and multi-media services through several user and service trials in five areas in Europe. A CRABS trial network set-up by Telenor R&D demonstrated an in-band MWS return channel as shown in a simplified diagram below:

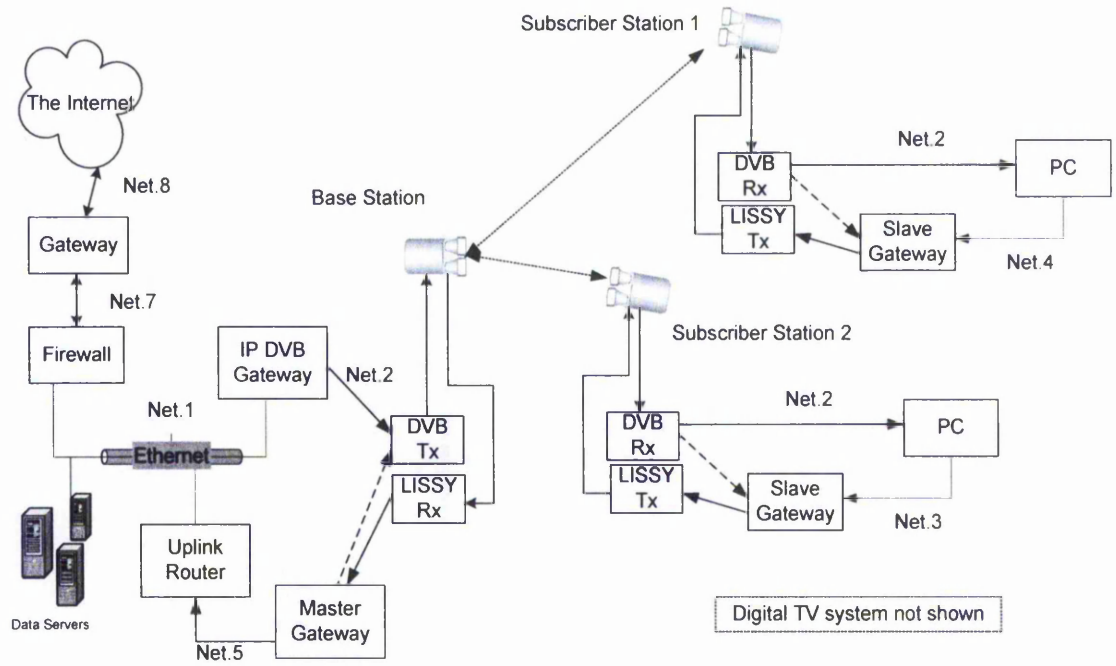


Figure 2-13: CRABS trial set up

The CRABS set up can be understood by considering it as two subsystems; the Uplink interaction system and the Downlink system. The uplink system shown is called LISSY (Local Network Interconnection via Satellite System) which consists of Master Gateway and two Slave Gateways. The access scheme for the Slave gateways is Time Division Multiple Access (TDMA) with dynamic on demand assignment of bandwidth. The LISSY system supports up to 64 Slave Gateways per Master Gateway with maximum bandwidth of 2 Mbps. All internal switching and routing of LISSY are based on ATM and spans between the Master and Slave gateways. Master or Slave gateways are layer-3 devices, hence one IP subnet was needed for each station which are represented as Net.3, Net.4 and Net.5 in the figure.

The Downlink system is a unidirectional broadband broadcast channel including video, audio and data. These services are delivered as an MPEG-TS multiplex according to the

ETSI standard [80]. IP packets in the base station subnet, Net.1, destined for a subscriber station are processed by the IP DVB gateway. The IP-DVB gateway is a Windows NT PC that acts as a router between the bi-directional Ethernet network and the unidirectional Downlink network. The interface to the Ethernet network is via a standard 10Mbps card while the interface to the DVB modulator (DVB Tx) is via a Philips proprietary MPEG-TS streamer card with a DVB-SPI interface. IP packets are encapsulated into MPEG-TS cells using the ETSI standard MPE technique as described in subsection 2.2.3. The Downlink system forms an IP subnet, Net.2. Therefore, to connect to both uplink and downlink systems, each subscriber PC shown in the figure has a Net.2 IP address and an uplink network address, either Net.3 or Net.4.

Experiments conducted on the CRABS MWS demonstrated the possibilities of a MWS and revealed some problem areas. Application latency was measured using the Ping program where a 110 ms round-trip-time was determined. A known limitation of the MPEG-TS streamer card (lack of flushing mechanism) required a constant 0.6 Mbps UDP stream running in parallel to flush out packets being stuck in its buffers to achieve the stated round trip time result. Without background UDP flushing, the packets will be indefinitely stuck and result in time-outs. The high round trip time was reported to impact throughput which required modification of TCP Window size configurations in the PCs. Whilst the change from a 8 Kbyte to 64 Kbyte TCP window size increased download throughput from 245 Kbps to 2.1 Mbps, the performance was still only a third of what was achievable using the UDP protocol. In the experiments conducted, the downlink throughput performance using UDP was limited to 6 Mbps, attributed to the IP DVB gateway performing the MPE encapsulation process. Real Uplink performance was also limited to 350 Kbps using TCP although the LISSY system was capable of 2 Mbps. The reason for the low uplink throughput over LISSY was not found. However, the project objective to implement a DVB broadband downlink and in-band return path over LISSY with TDMA demand on assignment was successfully demonstrated in the experiments.

2.3.2. Prior Nottingham Trent University MWS Trial

The Nottingham Trent University 42GHz MWS campus trial [8,9,10,81] was set up by the University's Communication Research Group in collaboration with Philips UK and Hughes Network Systems which provided the radio and networking equipment respectively. The trial aim was to establish a live multi-user radio network on which to demonstrate broadband applications, which includes deployment and implementation of the radio hardware, and analysis of signal propagation at 42 GHz [82]. The NTU trial base station is located on the same building used by the Communication Research Group which links subscriber nodes located at other university buildings, hall of residence and individual houses.

The trial was set up set-up as illustrated in Figure 2-14 which adapted DVB-C equipment intended for cable networks. The DVB-C equipment was configured to use a IP over ATM over DVB for broadcast downlink, and IP over ATM over TDMA for uplink as specified in the ETSI standard [83]. The Interactive Network Adapter (INA) is a highly integrated equipment which manages the Time Division Multiple Access (TDMA) uplink as well as frame forwarding between subscriber stations. The INA effectively creates an IP subnet for subscriber nodes and acts as an IP router between the subscriber subnet and the base station subnet. The base station IP subnet is shown as Net.1 and user subnet as Net.2 in the figure.

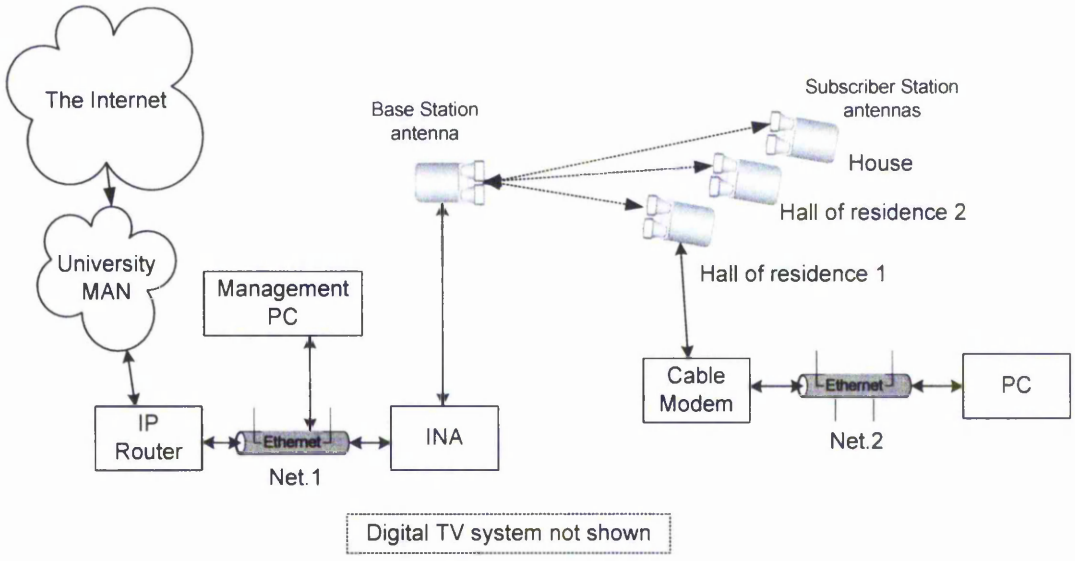


Figure 2-14: Prior Nottingham Trent University MWS Trial

An important experience was gained about the adaptation of commercial cable modem equipment in MWS. It was found that the cable modem was periodically loosing lock on the signal which would happen in increasing frequency within the period of several months of the trial. This problem was eventually pinpointed to the frequency drift (due to age) of the 50 Mhz crystal used to generate the 42Ghz carrier signal in the radio antenna. As the cable modem used was designed with a relatively narrow band, the drift had caused the band to move to the edge of the cable modem's bandwidth scope which caused the temporary loss of lock. This problem hindered the deployment of the cable modem equipment in the trial for a long term measurement campaign.

2.3.3. AT&T Laboratory New Jersey MMDS trial

The AT&T trial claims to be one of the first two-way broadband fixed wireless access systems anywhere [84]. Since it started in 1997 the trial has expanded to 12 user sites in 1999. Similar to the NTU trial, the AT&T trial adapted cable modem technology. However, the cable modem used was based on the DAVIC instead of DVB-C used in NTU trial. The AT&T trial also operated at a comparatively much lower frequency of 2.6 GHz. The set up of the AT&T trial is shown in Figure 2-15.

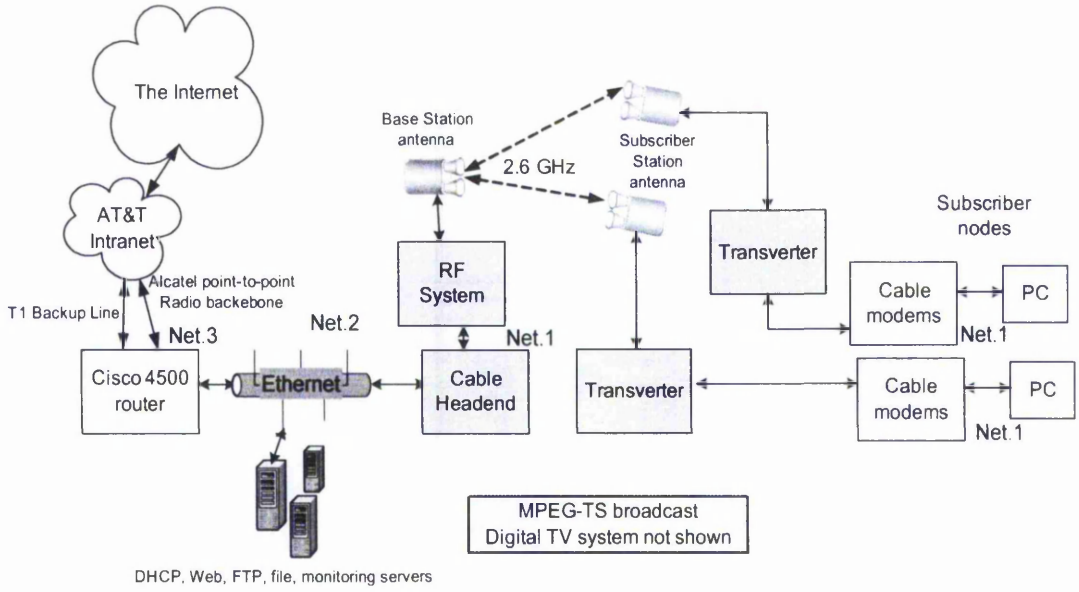


Figure 2-15: AT&T - New Jersey trial set-up

The Cable Headend manages the TDMA uplink and performs internal forwarding between subscriber nodes. The Cable Headend effectively creates an IP subnet for subscriber nodes and acts as an IP router between the subscriber subnet and the base station subnet. The internetwork design for the trial consisted of three IP networks, the user subnet, the base station subnet and the AT&T Labs network shown as Net.1, Net.2 and Net.3 in the figure respectively. In the user subnet, MAC source addresses are omitted when subscriber packets are processed by the Cable Headend as the MPE/DVB-Section encapsulation specified in the DAVIC standard was used. Hence packets in the user subnet would not have valid MAC source addresses. The lack of valid MAC source address has been reported in other systems to impact the function of certain applications (see EMBRACE trial in subsection 2.3.6).

Nevertheless, the "lengthy period of operation has allowed [them] to make observations about (DAVIC) equipment performance" when adapted and deployed in a MMDS environment [85]. Similar to the experience of the NTU trial, "the stability of uplink carriers has been the most common cause of service disruption" which was essentially caused because the tolerance of 10% carrier recovery for a cable system translates to a carrier stability oscillators of 2 PPM at 2.6 GHz. It was suggested that the use of more

stable and expensive oscillators that meets the 2 PPM accuracy to achieve more satisfactory performance.

2.3.4. The Cambridge MMDS Trail

The Cambridge trial [86] encompasses the Cambridge City area and its immediate surroundings. It was formed with the collaboration among University of Cambridge, AT&T Laboratory Cambridge and Adaptive Broadband Limited. While the networking equipment for the trial was contributed to the group by Adaptive Broadband Ltd, a research team at the University of Cambridge deployed the MMDS system at 5.2 GHz and 5.7 GHz.

The equipment provided by Adaptive Broadband Limited was based on the ATM technology and hence, networking infrastructure between the system and end nodes was mostly ATM. Even at Phase I, the trial size was extensive with up to three MMDS cells connecting twenty nodes. Preliminary applications showed the successful distribution of digitally encoded TV channels and broadband data networking. The system also provided a flexible choice of data interface with the use of ATM to Ethernet adaptors. The end nodes had the option to use either 10Mbps Ethernet or an ATM-25 interface. Figure 2-16 below illustrates the Cambridge trial set up.

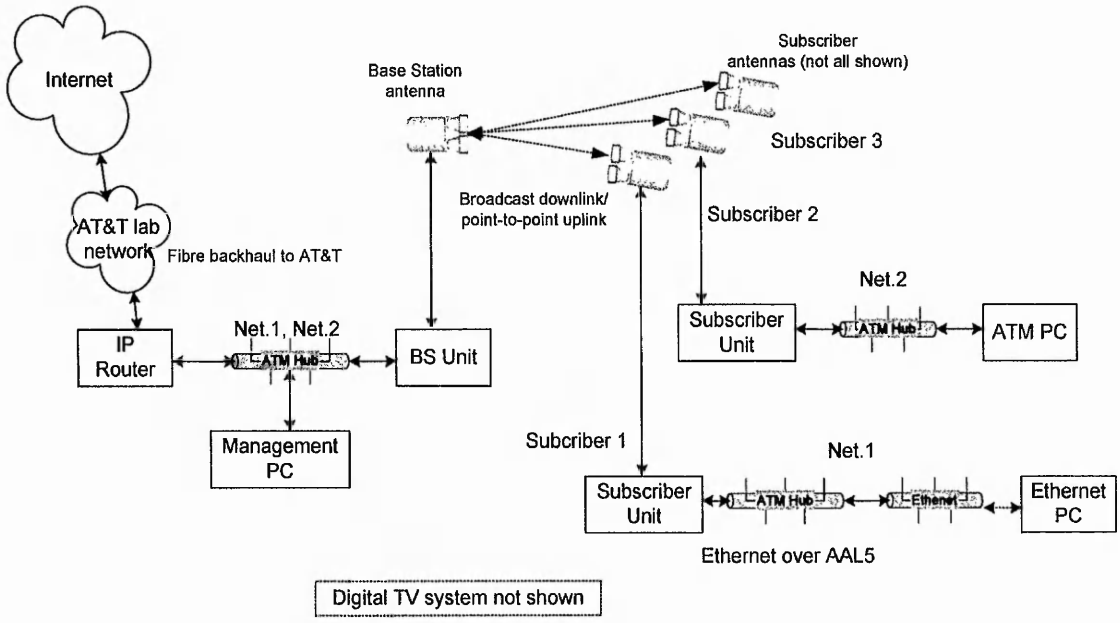


Figure 2-16: Cambridge MMDS trial set-up

During the deployment phase, the trial had to overcome internetworking issues that were in essence caused by the point to multipoint topology of the radios. To overcome this, they had suggested the use of a "complicated suite of ATM routing protocols" which are listed in their publication [86] to solve the routing problems. However, they did not implement this solution as the ATM equipment provided did not support all the required protocols. They also suggested the use of a mesh of bi-directional ATM Virtual Circuits (VC) from each node to every other node in the MMDS cell to achieve full layer-2 connectivity but realised that a mesh of ATM VCs was not going to be scalable. In the end, they opted to create VCs only from each subscriber PC to a common IP router which is located at the base station site. Each ATM VC forms a virtual IP subnet from a PC at the subscriber node to the IP router at the base station. The IP router knows that it was responsible to route frames received from one subscriber PC to another because it recognises that they belong to different IP subnets.

In the system, IP support was provided using Classical IP over ATM (CLIP) encapsulation which is based on the IETF standard [87]. However, they mentioned a few limitations of CLIP when it is applied to their system as it does not support full ATM signalling. ATM ARP and other facilities were used to solve the problems at the expense of end-to-end performance, though the degree of performance impact was not quantified in their publications. The Latency results based on the observation of the Cambridge system showed very good Round Trip Time (RTT); between 2.5ms-4ms for Ethernet, and 1.3ms – 3.3 ms for ATM interfaces, where the variation were caused by varying background network load. Significant unfair bandwidth hogging (up to 40% more) between subscriber nodes was also observed.

The direct support of MPEG based broadcast services was not shown to work in this system although they had successfully piped TV channels over ATM using proprietary equipment. However, based on their findings, ATM did not give an advantage over MPEG-TS based systems when it comes to internetworking. This was probably due to the immaturity of ATM equipment used compared with DVB equipment that were available to them at that time.

2.3.5. The CABSINET MWS/MMDS trial

The Cellular Access to Broadband Services and Interactive Television (CABSINET) trial [88] was done between 1996 and 2000, with the full field trial deployed in Berlin in the year 1999. The trial uses a unique two layer architecture that is claimed to be more flexible than the traditional one layer architectures [89]. The first-layer 40GHz macro-cells communicate with smaller second-layer micro-cells operating at 5.8GHz. Subscriber units can be connected to either level of the hierarchy using unique radio equipment appropriate for the specific frequency, but share uniform networking and data interface equipment. The basic architecture of the CABSINET trial is illustrated in Figure 2-17.

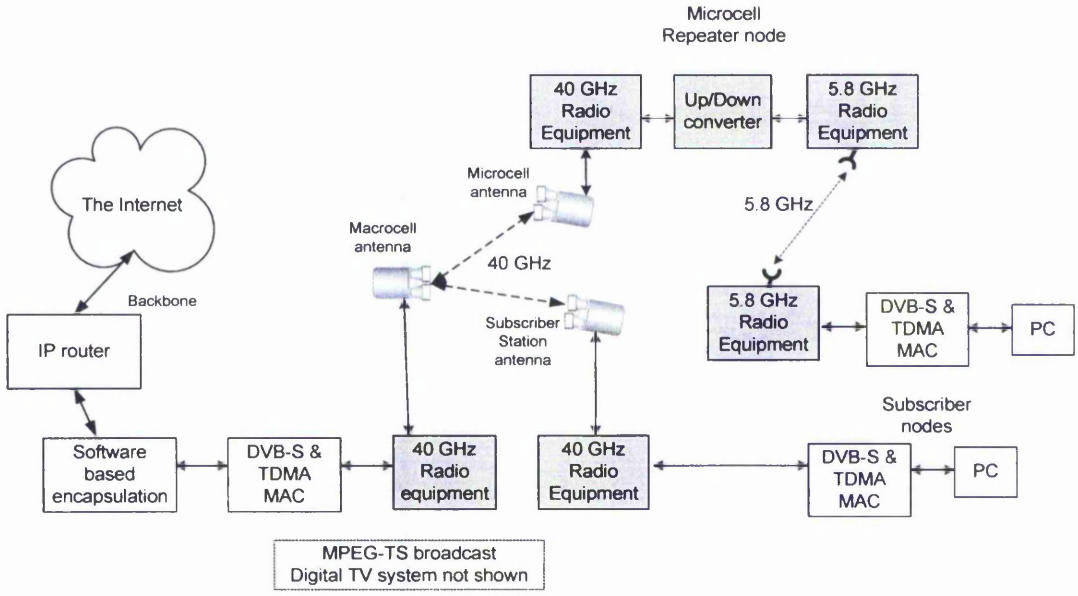


Figure 2-17: Cellular Access to Broadband Services and Interactive Television (CABSINET) trial set-up

Research within the CABSINET project [90,91,92] was focused on studying the effects ATM and MPEG framing in MWS/MMDS. They compared the impact of the two framing methods in terms of incurred overheads and efficiency performance using a simulation study where it was concluded that MPEG framing was better than ATM, performing 20% more efficiently. The use of ATM as the underlying technology is usually justified by its ability to cooperate with core ATM networks. However, the

CABSINET trial did not maintain end-to-end ATM semantics over their MWS/MMDS due to limitations in the ATM equipment and opted for just "cell-level ATM interoperability". Although this limited CABSINET to operate ATM on a best-effort basis the network was still "very reliable, and predictive transmission [was] achieved" [91].

In one of their publications [91], they reported that "occasionally in some traffic traces, one discovered large delay peaks [on the order of 500 ms compared to ordinarily constant 100 ms], which were generated because of lost packets and the time required to recover the situation". On another observation acknowledged in [89], they had reported a "major problem that affects TCP/IP performance in the implementation of MWS/MMDS system is downlink delay". "This delay is caused by head-end video server software performing the encapsulation of IP packets into MPEG frames". The typical 100 ms latency that was observed, due to the software encapsulation, can be considered very large compared to some systems reviewed here. In that work, they also concluded that the increased network latency had negative impact to the throughput performance to TCP and higher layers.

2.3.6. The EMBRACE MWS trial

The Efficient Millimetre Broadband Radio Access for Convergence and Evolution (EMBRACE) MWS trial [93] is an extension of work carried out in CRABS which had identified a need for improvement of the return link and coverage issues in MWS. A demonstration of their work included a system that used the DVB standard for forward link and ATM for return link. The EMBRACE trial demonstrated a variety of services including: broadcast television over IP, voice over IP, IP videoconferencing and multicast push and proxy services. Diversity routing to counteract heavy fading was also an essential part of the project. The EMBRACE trial architecture is illustrated in Figure 2-18 below [94].

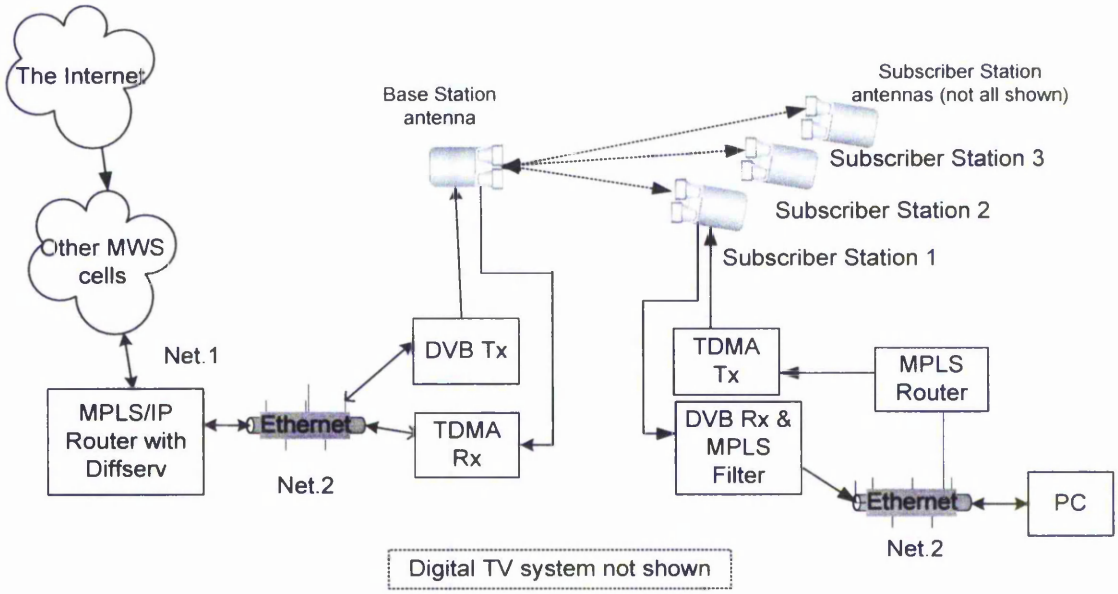


Figure 2-18: EMBRACE Trial set-up

The EMBRACE approached the IP routing problem over the BFWA topology using Multi Protocol Label Switching (MPLS) technologies. The EMBRACE MPLS routers, implemented as Linux workstations, were manually configured with unique MPLS labels. The MPLS Routers at the base station and subscriber nodes ensure that all packets entering the EMBRACE MWS are labelled with MPLS addresses. An IP packet sent by PC at a subscriber station will be forwarded by the subscriber MPLS router to the base station and looped-back by the Ethernet at the base station. However, the MPLS Filter implemented in the DVB Rx equipment will prevent the packet from reaching the source PC. Since all loop-backs are prevented in this manner, the MPLS routers effectively create layer-2 connectivity between subscriber nodes which allowed all subscriber hosts to be internetworked with Ethernet bridges and addressed using a single IP subnet address space. Using this technique the IP routing issues associated with the topology of the MWS are solved. However, base station premises equipment had to be placed in a separate IP subnet before the MPLS routers so that traffic to and from them will pass through the MPLS/IP router before entering MWS, as labelled as net.1 and net.2 in Figure 2-18.

An issue with the MPE/DVB-Section encapsulation employed by the DVB equipment initially prevented Ethernet frames to be transparently forwarded. As Ethernet MAC

source address was omitted by MPE/DVB-Section encapsulation, a novel encapsulation had to be defined that can provide full Ethernet MAC header information [95]. The encapsulation was called 'Ethernet over Section' which modifies standard DVB-Section/MPE as shown in Figure 2-19 below.

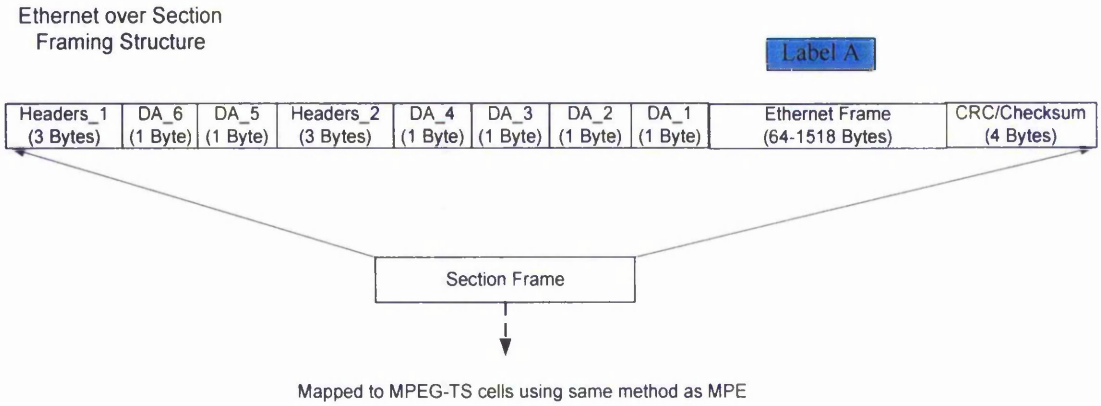


Figure 2-19: Ethernet over Section Encapsulation

An entire Ethernet frame except its Destination address (DA) is placed in the payload of a Section Frame (Label A). Since the Destination address are already present in the headers of a Section Frame (fields DA_1 through DA_6) and Source address are included in the Ethernet frame, information required to reconstruct the original Ethernet frame was encapsulated to achieve full transparency. While the 'Ethernet over Section' encapsulation was used to transport all user data traffic, DVB Section MPE encapsulation was still used for synchronization of the TDMA uplink of the system. MPE data, Ethernet over Section, MPEG-2 digital TV channels and other services are multiplexed at the MPEG-TS level and broadcast in the downlink.

Summary of Section 2.3

This Section reviewed previous trials from which valuable lessons can be learnt from their experiences particularly on internetworking issues caused by the BFWA topology, equipment deployment and encapsulation implementation.

From the experiences of the CABSINET trial and Cambridge trial the use of ATM for internetworking had a large impact on protocol overhead since heavy translations were required to interface with base station and subscriber premises Ethernet local area networks. Also, the use of software to perform encapsulation should be avoided as it adds a significant amount of latency to the network. From the experience of the AT&T Labs trial and NTU Trial, two conclusions can be made about the use of DAVIC based cable modem technology. Firstly, full Ethernet-based services cannot be realised using existing DVB equipment without a custom encapsulation mechanism, such as that implemented by the EMBRACE trial. Secondly, the use of mass produced cable modems can at first seem to reduce the cost of the deployment, however, this cost savings might be offset by an increased in requirement for carrier oscillator stability for BFWA frequencies in the GHz range. The table below summarizes the set-up and implementation of the trials reviewed.

MPEG-TS based Trials	Encapsulation Method	Internetworking methods	Equipment	Frequency (BFWA Class)
EMBRACE Trial	Ethernet over Section	Ethernet switching / MPLS	DVB/ MPLS	42GHz (MWS)
AT&T Labs	DAVIC/ MPE	ATM switching/ IP routing	DAVIC	2.6 GHz (MMDS)
CRABS Trial	DVB/ MPE	ATM switching/ IP routing	DVB/ ATM/ LISSY	42 GHz (MWS)
The NTU trial	DVB/ ATM	ATM switching/ IP routing	DVB-C	42GHz (MWS)
Non-MPEGTS based Trials				
Cambridge Trial	ATM	Layer-2 ATM / PVC	ATM	5.2 / 5.7GHz (MMDS)
The CABSINET	ATM	Layer-2 ATM / PVC	DVB-S	42 GHz / 5.7 GHz (MWS/MMDS)

Table 2-4: Summary of previous BFWA trials

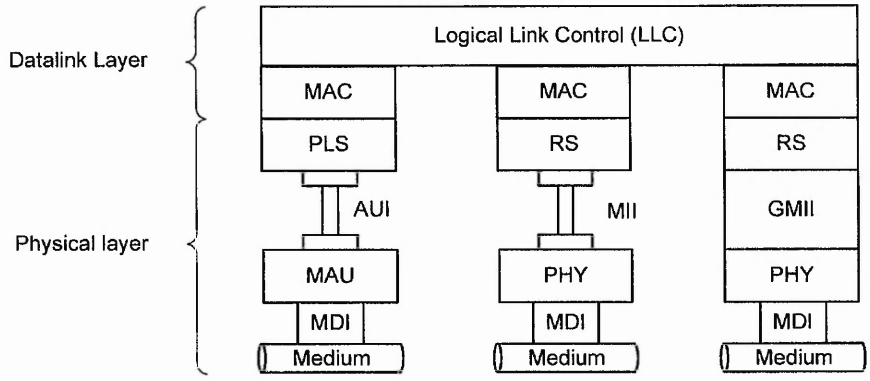
2.4. The Ethernet technology

In 1973, at Xerox Corporation's Palo Alto Research Centre, researcher Robert Metcalfe designed and tested the first Ethernet network and published a landmark paper on Ethernet [96]. Later, the IEEE standardised Ethernet with the 802.3 specification; however, the original name remain a legacy and is still commonly used. 802.3 was since maintained by IEEE, and evolved to support a variety of media that extends its reach outside its traditional LAN applications. Ethernet grew to become the most popular and widely deployed LAN technology in businesses, even threatening ATM deployment in MAN access networks. Due to the dominance of Ethernet, most data traffic begins and ends in Ethernet networks. Hence it makes sense to also provide Ethernet over MAN networks.

A key direction of this research is to conceive a novel MWS interface that is optimized for the provision of Ethernet-based services, and investigate the adaptation of Ethernet technology to resolve known MWS architectural issues. This section reviews the Ethernet technology based on the 802.3 standard [97] and texts on the subject [98,99,100] while focusing on the mechanisms that are important in the implementation of the novel MWS interfaces developed in this research.

2.4.1. IEEE 802.3 Layers and interfaces

Through the evolution of Ethernet, the data rates were incremented by 10 times each generation to match the exponential communication demands of computers. Legacy Ethernet operated at a data rate of 10 Mbps; Fast Ethernet operated at 100 Mbps; Gigabit Ethernet at 1Gbps; and 10Gig Ethernet at 10 Gbps.



AUI: Attachment Unit Interface
MAC: Medium Access Control
MAU: Medium Attachment Unit
MDI: Medium Dependent Interface
MII: Medium Independent Interface
GMII: Gigabit Medium Independent Interface
PHY: Physical Layer Entity
PLS: Physical Layer Signaling
RS: Reconciliation Signaling

Figure 2-20: IEEE layered model of Ethernet Technologies

In Ethernet, the Datalink Layer corresponds to the Logical Link Control (LLC) and Medium Access Control (MAC) sublayers. The LLC sublayer was originally designed to be the same for all types of LAN technology in order to support interoperability. However, the LLC is currently not used often for that purpose. Instead, interoperability is usually provided by IP at the Network Layer.

The Medium Independent Interface (MII) is defined for Fast Ethernet to replace the AUI interface that is used by Legacy Ethernet. Initial specifications of the MII interface included a physical cable and interlocking connectors. However, as the Physical layers (PHYs) got integrated into the Network Interface Card (NIC) hardware, the MII cable was reduced to copper traces on a PCB or into silicon in integrated MAC/PHY chips. The MII interface include a data transmit port, a data receive port, a bidirectional management port, carrier sense signal and collision signal (Figure 2-21 below).

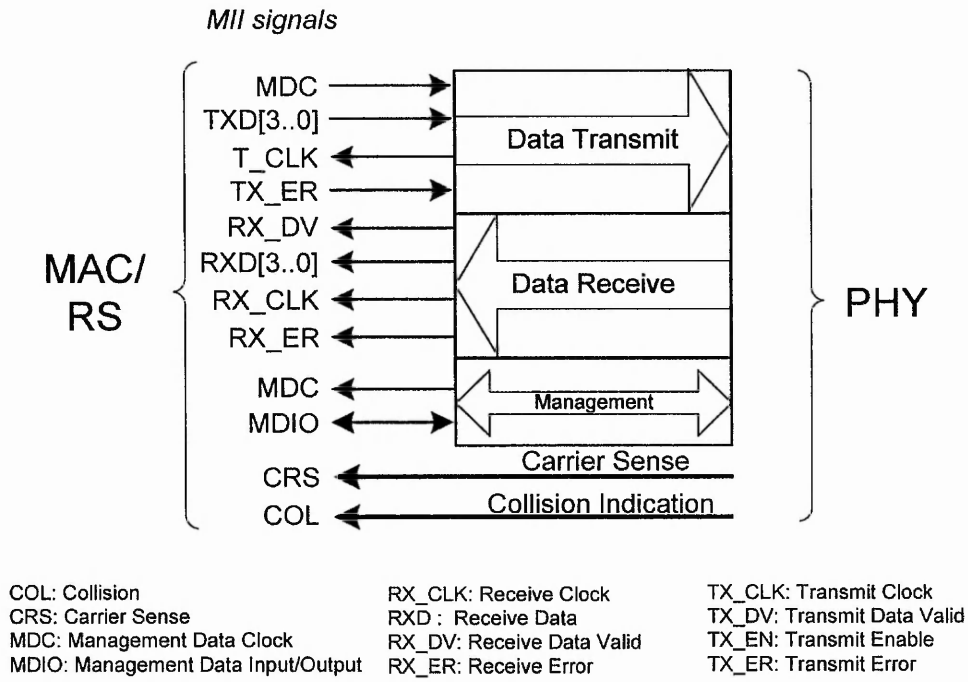


Figure 2-21: Signals of a MII port

The reconciliation sublayer (RS) in Fast Ethernet replaces the Physical Layer Signalling (PLS) sublayer in Legacy Ethernet. Line transmission encoding and decoding, which were performed by the PLS, are moved to the PHY sublayer. The RS sublayer is left with the function to convert 8-bit wide Ethernet MAC frames to 4-bit format to be passed over the MII interface. The diminishing importance of the RS sublayer was probably why it is often left out in models used in literature [101].

The combination of the Management Data Clock (MDC) and the Management Data Input/Output (MDIO) forms the bidirectional management port. Through which, PHY tables that store status and configuration information, can be accessed. Auto negotiation (detailed later in subsection 2.4.4) is an example of a function that extensively relies on these tables.

2.4.2. Ethernet MAC Frame

The Ethernet frame structure used in Legacy Ethernet remains the same in all Ethernet generations Figure 2-22.

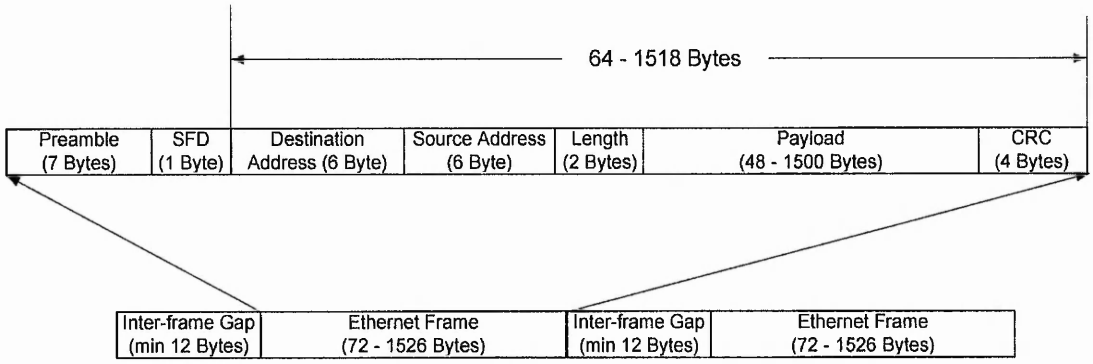


Figure 2-22: The Ethernet Frame Structure

Preamble. This field consists of alternating 0s and 1s that alert the receiving hardware of the incoming frame. The timing information carried in the 56-bit pattern gives sufficient time for the hardware to synchronize its own clock to the frame.

Start Frame Delimiter (SFD). The SFD has a binary value of '10101011' which signals the end of the preamble and the beginning of the Ethernet frame.

Destination Address (DA) and Source Address (SA). The DA and SA contain the 48-bit MAC addresses of the receiver and sender respectively. Each Ethernet MAC is assigned an address that is unique throughout the world. However, certain addresses are reserved to signify a special function or to address more than one host; for example broadcast packets, multicast packets and flow control packets.

Length/Type. This field holds the binary value representing the number of bytes in the payload of the Ethernet packet and can be in the range of 46 to 1536. When the binary value is greater than 1536, it represents the 'type' of the encapsulated packet in the payload instead.

Payload. This field carries the encapsulated packet from the network layer which is almost always IP packets. However, some internetworking devices encapsulate their own proprietary frames in Ethernet frames to exchange information with peer internetworking devices e.g. routing updates and network status.

The minimum length of 46 for the payload field was a restriction inherited from Legacy Ethernet. Basically, the CSMA/CD (Carrier Sense Multiple Access / Collision Detection) protocol used in Legacy Ethernet required the minimum frame size to guarantee that the first byte of a frame will reach all hosts connected to the medium before the last byte leaves the sender. This way all collisions can be detected by the sender who can retransmit the frame. However, in full-duplex operation supported by Fast Ethernet and Gigabit Ethernet, the CSMA/CD protocol is not needed. Nevertheless, the 46 byte minimum payload length restriction still applies for backward compatibility.

2.4.3. Ethernet physical and logical topology

The physical topology refers to the way in which a network is laid out physically. The commonly used Ethernet technology today is 100BaseT also known as Fast Ethernet. Fast Ethernet interconnection devices (Ethernet hubs) and UTP-CAT5 (Unshielded Twisted Pair - Category 5) cabling has a physically laid topology of star or tree [101]. In a star or tree topology only one path exist between any two stations, which avoid loops or duplication of data packets from alternate paths.

The logical topology describes the network from the viewpoint of the frames travelling the network. Ethernet's logical topology is a bus where each transmission is broadcast throughout the network. In this topology, all packets sent by a host (layer-3 entity) can be assumed to reach all other hosts in the same network. The broadcast nature of Ethernet networks is also referred as 'full layer-2 connectivity'.

The example in Figure 2-23 shows a Fast Ethernet network connected by several internetworking devices in a tree physical topology. In simplified terms, full layer-2 connectivity is achieved because all Ethernet interconnection devices (Ethernet hubs) apply a basic rule to every frame: "forward the frame to all ports except the port in which it came from". This is illustrated in the diagram where Host A creates a unicast packet to

Host B. The frame forwarded by the Ethernet network and will eventually reach Host B as can be seen by the paths marked by the green arrows in the diagram. All other hosts will also receive a copy of the packet as shown by the red arrows. However, the hosts will safely ignore frames not destined for it by checking the destination Ethernet MAC address in the frame header.

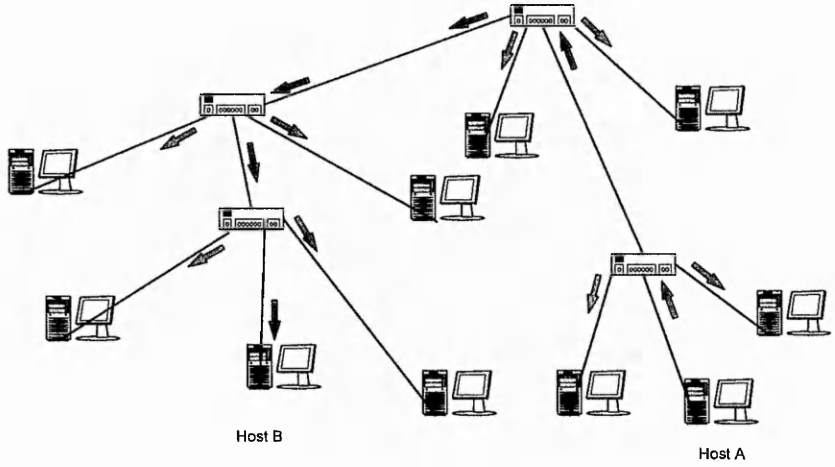


Figure 2-23: Ethernet network in a tree topology with full layer-2 connectivity

The example illustrated was a unicast transmission as the frame was meant for only one host. There are two other modes of a frame transmission which are broadcast and multicast. In a broadcast transmission, the well known broadcast destination MAC address of FF-FF-FF-FF-FF-FF is used which will be accepted by all hosts in the network. A frame that is sent as a multicast transmission means that the frame is meant for a specific subset of hosts on the network. A multicast frame can be differentiated from a unicast frame by looking at the LSB of the most significant byte in of the destination address as shown in Figure 2-24. Hosts that have previously registered to be part of a multicast, using a higher layer mechanism [101], would know which multicast frames to accept or ignore.

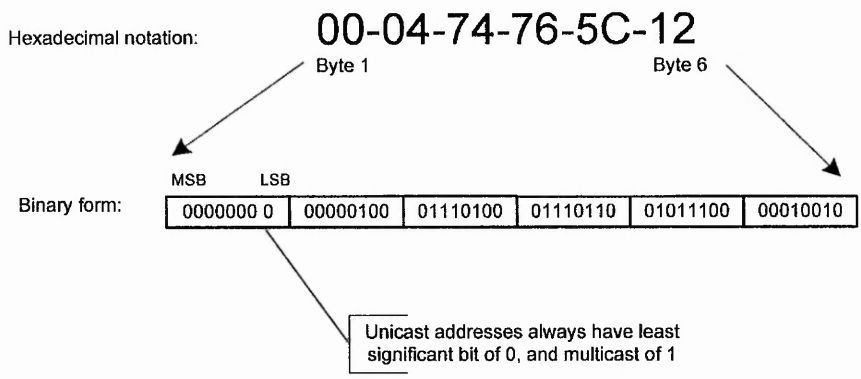


Figure 2-24: Ethernet Frame type identification

Layer-2 Ethernet interconnection devices (Ethernet hubs) are categorised into two kinds, repeaters or bridges as defined in IEEE 802.3 standard. Repeaters are used to build a shared-medium network which forms a 'collision domain'. Hosts on a shared-medium network implement the CSMA/CD protocol to resolve contention and communicate efficiently. However, the CSMA/CD protocol imposes strict timings requirements to network deployment that restrict the physical size of the network. For example, in order to satisfy maximum 'round-trip collision delay' requirement of 5.12 μs (512 bit time) the maximum cable segment is limited to 100 m and up to two Class II repeaters can exist between any two hosts.

Ethernet Bridges can also be used to extend the size of a network, however, it does not extend the CSMA/CD collision domain. 10/100BaseT Ethernet bridges can connect heterogeneous networks (mixed CSMA/CD and full-duplex networks). So, when interfaced with a CSMA/CD network, an Ethernet bridge appears as a MAC layer entity to the collision domain rather than a repeater. Unlike CSMA/CD networks, the physical size of full-duplex networks is not limited by the round-trip collision propagation delay. Instead, the maximum cable length between two hosts is limited only by the signal transmission characteristics of the specific cable.

Legacy Ethernet bridge implementations store entire frames and perform layer-2 frame conditioning to ensure the integrity of each frame before they are forwarded. As a consequence of advancements in Ethernet media, the probability of errored frames is remote. Hence, some modern Ethernet bridges also known as Ethernet switches

disregard layer-2 frame conditioning and implement cut-through-forwarding by immediately forwarding a frame before it is fully received to reduce forwarding latency.

Ethernet switches, are also semi-intelligent internetworking devices that can optimize data carriage over a network [98]. They can exploit the difference between unicast, multicast and broadcast transmissions and only forward frames on ports where the destination node can be found. Furthermore, modern switches also commonly implement the Spanning tree protocol that can detect and resolve physical loops that were accidentally (or deliberately) formed in the creation of the physical network. The spanning tree protocol will 'switch' off certain links in the network to maintain the tree topology of the physical network. Besides correcting human cabling errors, spanning tree allows redundant links to be made to increase the reliability of a network. With such features and advancement of electronics that reduced the costs of internetworking devices, Ethernet switches are quickly phasing out the need for traditional Ethernet repeaters.

2.4.4. Auto Negotiation and Full-Duplex

One of the strongest strengths of Ethernet is its ability to be backward compatible with previous generations. The auto-negotiation process takes place between Ethernet PHYs to set up the physical connection and agree on a set of MAC protocols that will be used. Auto-negotiation process uses the Ethernet medium to communicate with its link partner, however, Ethernet MAC frame structure cannot be used yet as the common protocols have not been established. Instead they use the Fast Link Pulse (FLP) mechanism [102], to transmit the capabilities table that are normally implemented as a register in the PHY hardware. Each link partner periodically transmits its capabilities and they would settle for the best common technology. The preferential order of Ethernet technology used by auto-negotiation over UTP-CAT5 is as follows.

Technology	Duplex	Global Throughput
1000BaseT	Full Duplex	2000 Mbps
1000BaseT	Half Duplex	1000 Mbps
100BaseT2	Full Duplex	200 Mbps
100BaseTX	Full Duplex	200 Mbps
100BaseT2	Half Duplex	100 Mbps
100BaseT4	Half Duplex	100 Mbps
100BaseTX	Half Duplex	100 Mbps
10BaseT	Full Duplex	20 Mbps
10BaseT	Half Duplex	10 Mbps

Table 2-5: Preferential order of technologies in Ethernet Auto-negotiation

Full-duplex operation was added as a supplement to the 802.3 standard that exploits the medium and physical topology of some Ethernet technologies. To remain compatible with all Ethernet technologies and older Ethernet devices, full-duplex relies on the auto negotiation process to handle its use dynamically. Full-duplex operation was accepted well by the industry as it brought significant improvements to Ethernet without much modification to existing infrastructure.

The twisted pair cabling used in 1000BaseT, 100BaseT2, 100BaseT and 10BaseT take a tree topology where each host is connected on a point-to-point cable to the internetworking device. Each of these cables intrinsically supports full-duplex communication as they consist of individual pairs of conductors that can provide dedicated channels for each direction of communication [99].

Full-duplex operation eliminates the need for the CSMA/CD protocol altogether as no collision can occur on the Ethernet medium. Not only did the network capacity increased by a factor of 2 when using full-duplex, known scalability limitations associated with the CSMA/CD protocol [101] was also solved. In fact, most Gigabit Ethernet implementations only supports full-duplex operation as the CSMA/CD protocol for half-duplex proved to scale poorly at that speed [99].

2.4.5. Flow Control in Ethernet

In CSMA/CD Ethernet, vendors implemented an unofficial flow-control mechanism with the use of jamming frames. Jamming frames may not concur to the 1518 byte frame size limit and may span as long as congestion exist in the host sending the jamming frames. While a jamming frame occupies the medium, the CSMA/CD protocol implemented in the other hosts will prevent them from transmitting new frames into the network to avoid collisions. Therefore, this mechanism effectively causes a backpressure type flow-control to ease congestion in a network.

When a link pair is in full-duplex mode, jamming frames cannot be used as CSMA/CD protocols are disabled. The 802.3 specification specified a 'full-duplex flow control' mechanism whose support is conveyed by auto-negotiation process. This is a simple mechanism in which a host can send a pause frame (see Figure 2-25) to its link partner or internetworking device to temporarily stop it from sending new frames on the link.

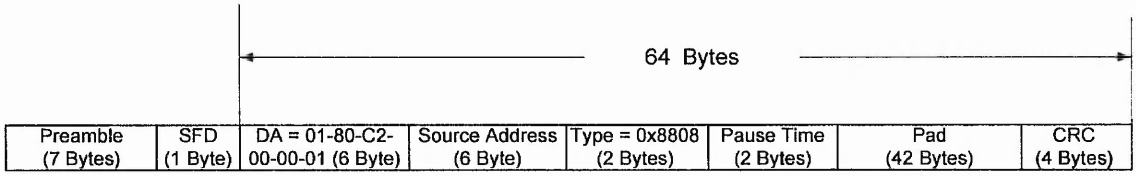


Figure 2-25: Ethernet Pause frame Structure

Hosts and internetworking devices can recognize pause frames from its unique DA MAC address of 01-80-C2-00-00-01 and type field of 0x8808. Asymmetric flow control might be advertised during the auto negotiation process. This might be useful in cases such as an Ethernet switch having the right to impose silence to hosts but not vice versa. The rationale behind the use of such asymmetric flow control is that the global capacity of the buffers in hosts is considered largely superior to that of the switch.

Summary of Section 2.4

The Ethernet technology is the most widely deployed LAN technology. With the vast innovations from academics and the industry, Ethernet networks are arguably far more inexpensive and yet comparable in terms of functionality to ATM. This is reflected in the

acceptance of Ethernet as a carrier class network by international standard bodies. In fact, because Ethernet is "able to offer considerably improved flexibility to customers through a much simpler and lower cost interface" it is becoming the preferred choice compared to ATM, forcing backbone operators to offer Ethernet Emulation over ATM to capitalise on previous ATM investments [103]. In previous BFWA systems, Ethernet-based services have been a value-added feature that is limited within an MWS cell. However, as Ethernet is the most widely accepted LAN technology and also rapidly being adopted in carrier class networks, there is little reason not to deploy Ethernet as the primary interface and inter-cell MWS networking technology. Optimising an MWS primarily for Ethernet will bring performance benefits while enabling it to inherit the continuous innovations of Ethernet.

Chapter 2 Conclusions

From this review, a common cause of inefficiency for carrying data in DVB-based BFWA systems lies within framing structure and encapsulation mechanisms inherited from broadcast centric networks. There is often a large portion of the protocol headers which are redundant for carrying data as well as inefficient encapsulation mechanisms to cope with the variable length units of packet based data networks.

This is fortunately realised by an IETF working group which responded with a Frame Length Marking (FLM) type encapsulation named 'Ultra Lightweight Encapsulation' (ULE) that drastically minimises protocol headers to bear minimum while maintaining functional flexibility by providing headers flags for future extensions. As yet, ULE remains in the protocol drafting stage and have not been implemented in BFWA interfaces. However, BFWA interfaces using very similar FLM based algorithms that were employed by previous trials have shown very high latencies (up to 100ms) and poor encapsulation rates. The performance limitations may be attributed to software implementation and/or complex high-layer internetworking functions. Based on the experiences of the trials reviewed, the complex internetworking functions were performed in the internetworking equipment because previous BFWA systems adopted IP routing or ATM semantics that were inadequate to understand the MWS topology and hence required costly workarounds.

EMBRACE's innovative solution of using 'Ethernet over Section' encapsulation and MPLS routers provides full layer-2 connectivity between subscribers over which Ethernet switches can be used instead of IP routers in customer premises. However, the cost savings for providing a simpler topology are likely to be offset by the cost and configuration complexity of the MPLS routers required at the base station and customer premises. Further, as 'Ethernet over Section' encapsulation used by EMBRACE were based on the DVB-section encapsulation, it lacks the elegance and efficiency of ULE.

The novel MWS interfaces that are explored in this research can benefit from the experience of two prevailing mechanisms which are ULE for its simplicity and EMBRACE for its Ethernet encapsulation. However, the solution for adequate MWS interfaces for the provision of Ethernet-based services remains to be found which incorporates the following; low implementation cost, simple internetworking configurations, low latency, and high encapsulation rates. In all of the reviewed encapsulation algorithms from standards and trials FLM encapsulation techniques are mutually used. This leaves an unexplored avenue to use a Byte Stuffing type encapsulation in a MWS interface. Byte Stuffing can be a strong alternative that complements Ethernet forwarding as it allows cut-through forwarding in Ethernet that are disallowed by FLM. While full hardware implementation of the MWS interfaces using VHDL design and FPGA prototyping was explored to address processing bottle neck and software related latencies, several prevailing criticisms on the use of Byte Stuffing encapsulation has to be first addressed. This is the focus of the next chapter.

3 BYTE STUFFING ENCAPSULATION FOR MWS

This chapter details the analytical study and VHDL designs that establish the benefits of using a Byte Stuffing (BS) algorithm, instead of Frame Length Marking (FLM) algorithms, for the encapsulation of Ethernet frames in Multimedia Wireless Systems (MWS). BS algorithms are generally perceived to have numerous drawbacks of high implementation complexity and cost, and unpredictability of output due to its worst case overhead characteristics. However, reassessment of BS algorithms done in this research using real networking traffic reveals that the actual frame expansion and unpredictability are minimal. Further, a modification to the commonly used Point-to-Point Protocol (PPP) BS was developed and resulted in an Optimised Byte Stuffing (OBS) algorithm that is more suited to the MWS investigated while retaining the general advantageous characteristics of BS over FLM. Complexity and cost concerns are also addressed in this chapter through implementation innovations and pipelining techniques adopted in hardware. The implementation results, explained using detailed schematic diagrams generated from VHDL source code, show that the OBS algorithm can be realised in a low-cost FPGA at a fraction of the cost of the FLM implementations surveyed. The OBS encapsulation cores form the major modules for the investigated MWS interfaces which address the limitations of FLM-based interfaces adopted previously namely high latency, low efficiency or slow operational speeds.

This chapter is organized into three sections. The first section, 3.1, describes the PPP BS and OBS encapsulation, and compares them with FLM. In section 3.2, a plausible MWS traffic distribution is acquired from a live network based on previous work in the field. The results of that analysis were used for a fair comparison study of the performance of a variety of BS and FLM encapsulation techniques. Section 3.3 details the design, implementation and synthesis results of the proposed BS algorithm.

3.1. Byte Stuffing Algorithm for MWS Interfaces

In the context of the investigated MWS system, this section presents the case for using Byte Stuffing (BS) algorithms. In the first subsection, 3.1.1, advantages of BS for use in MWS interfaces are critically discussed and compared to Frame Length Marking (FLM) from a theoretical standpoint. In 3.1.2, primary criticisms against the use of BS established by workers in the field are identified which outlines the probable reasons why an FLM type algorithm was adopted in all BFWA systems reviewed. Subsection 3.1.3 describes the encapsulation procedure and characteristics of the canonical Point-to-Point Point (PPP) Byte Stuffing (BS) algorithm. In light of the findings from the critical study, a new OBS algorithm was developed that is better suited for the MWS investigated (Subsection 3.1.4). Lastly, the primary disadvantages of BS algorithms are summarised, which form the key questions that are addressed the remaining sections of this chapter.

3.1.1. Theoretical Advantages of Byte Stuffing over FLM Algorithms

In the Byte Stuffing encapsulation considered, for each byte of input data the output generation is immediate. BS encoders do not have to wait until it has received a fixed amount of input before it is able to decide what the output will be. In contrast, FLM encoders must buffer an entire Ethernet frame before it can generate a single byte of output. In the words of a published work [104] from research in this field:

> *A FLM encoder with "length encapsulation requires knowledge of each packet's length before the first bytes of the encapsulated packet ... can be transmitted. Consequently, ...[FLM techniques] requires store-and-forward processing...".*

Since the 13th and 14th bytes of an Ethernet frame might define the length of a frame, it can theoretically be used by FLM mechanisms to avoid the need to buffer entire Ethernet frames. To address this possible argument, it is noted that the 13th and 14th bytes of an Ethernet frame are defined as a Length/Type field, which means that it can alternatively signify a Type code instead of a length. Type codes are commonly used by protocols such as NETBios and IPX which follow the Ethernet II standard formed by Digital, Intel

and Xerox prior to IEEE 802.3. Some Ethernet II types implicitly define a constant length while others have a length field at another specified location in the header. Since it is difficult and costly in practice (if possible) to maintain an up to date list of all 64036 type codes, it is assumed that such method is beyond the scope of implementation in an MWS interface.

Latency is defined here as the time from reception of the first byte of data presented, to the output of the first byte from the encapsulation algorithm as per IETF definition for bit forwarding internetworking devices [105]. The theoretical limit of encapsulation latency is determined by the amount of data the encapsulation algorithm has to 'see' before producing output. As the rate at which the data is received depends on the speed of the physical layer, it makes sense to measure latency in units of Tb, where Tb is the time it takes to receive one byte of data. For example, if the physical layer is 10 Mbps Ethernet, the Tb will be equal to 800 ns $\left(8\times\frac{1}{10\times10^{6}}\right)$. The graph below compares the FLM and BS encoding latency for the range of possible Ethernet frame sizes.

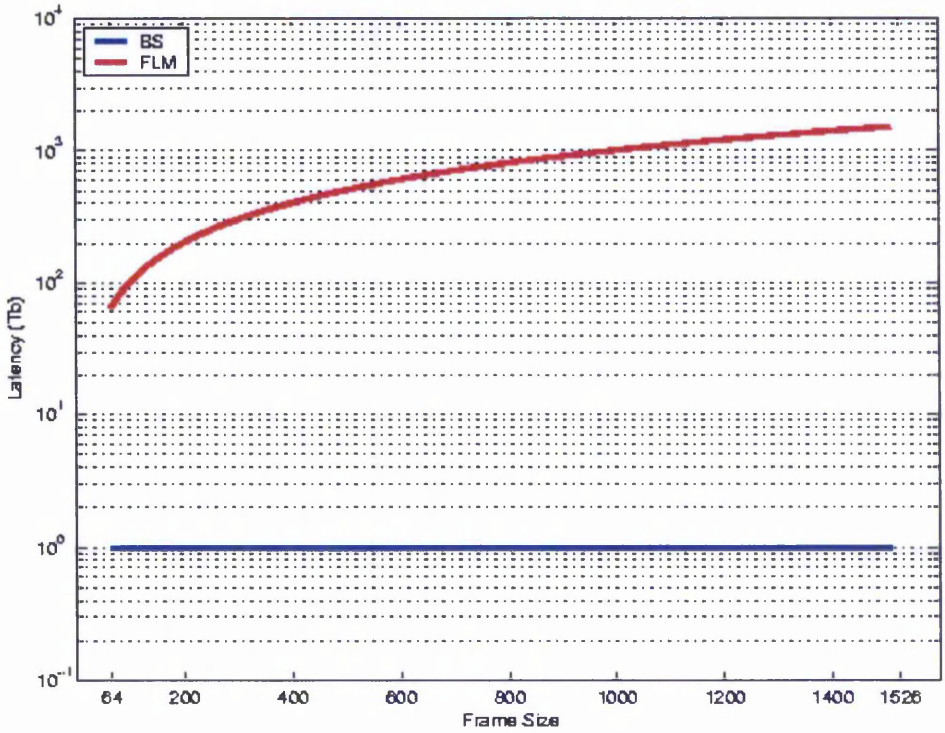


Figure 3-1: Comparison of encoding latency theoretical limits of BS and FLM encapsulation

From the graph, it can be seen that the BS encoding algorithm has not only a much lower theoretical encoding latency (about two or three orders) compared to FLM, but it also maintains a constant encoding latency for all frame sizes that can reduce inter-frame jitter. When quantified for one hop over an internetworking device, a BS encoder can theoretically output the first byte in Tb = 1 = 800 ns for any input frame size. For FLM, the theoretical encoding latency ranges from 51.2 μs to 1.21 ms (Tb = 64 to Tb = 1500) per 'hop' i.e. between two Ethernet MACs. The immediate generation of output bytes for each byte of input would not be beneficial to previous BFWA systems that adopted other mechanisms requiring store-and-forward processing. However, the immediate generation of output can be exploited by the Ethernet forwarding MWS interfaces investigated as the transmitter side can operate at cut-through forwarding mode.

In practice, it is shown that when the proposed Optimised Byte Stuffing encapsulation is implemented in VHDL (see subsection 3.3.4) an encoding latency of Tb = 4 can be achieved. This is close to the theoretical limit of Tb = 1 as explained above. While maintaining encoding latencies near the theoretical limit, very high operational speed can also be achieved, shown through FPGA synthesis results to be around 200 Mbps. When incorporating OBS cores into the MWS interfaces investigated in Chapter 4 and 5, the low latency and high-speed operation can still be maintained.

In all BFWA interfaces reviewed, the IP protocol was implemented as the internetworking layer. Since IP protocol forwarding requires error-free Network layer information, full decoding and error checking of Ethernet frames or ATM cells at the Datalink layer were required before making a routing decision. This implies the requirement for store-and-forward mode of operation at both transmitting and receiving MWS interfaces using IP based routing. It may seem at first that FLM encapsulation can theoretically allow cut-through forwarding at the receiver interface for an Ethernet forwarding since the length of a frame can be know as soon as the header is received. Based on knowledge of frame length, the receiving FLM decoder may predict when the last byte of the Ethernet frame will be received and begin to release a partially decoded Ethernet frame when a certain amount of buffering is achieved. However, in the MWS system considered, MPEG-TS multiplexing can invalidate FLM encapsulation predictions. When an Ethernet frame is encapsulated in more than one MPEG-TS cell,

the tail section of the Ethernet frame can be delayed by an amount equivalent to the number of MPEG-TS cells that are inserted in the middle of the encapsulated Ethernet frame. For example, in a DVB based MWS, MPEG-TS cells of encapsulated Ethernet services may be multiplexed with MPEG-TS cells from other broadcast services, (see 'transport MUX' in Figure 5-3). Also, null MPEG-TS cells can be inserted when the DVB modulator performs rate adaptation (see 'DVB modulator Tx' in Figure 5-3). To avoid mid-frame buffer underflow, an entire Ethernet frame must be reconstructed from the multiplexed MPEG-TS stream at the receiver interface before it can be released to an Ethernet network. Due to these reasons, store-and-forward processing at the receiver MWS interfaces has to be done regardless of whether a BS or FLM encapsulation scheme was employed.

Some of the performance gains (detailed in Chapter 4) of the proposed 'Ethernet to DVB/MPEG encoder' are due to of the ability of the OBS core to perform a fine granularity of flushing. It is important when encapsulating packet traffic in an MPEG-TS to flush out short frames or tail segments so that they do not have to be held in a buffer until another frame arrives. In an explanation from other research [106]:

> *"MPEG-2 multiplexers do not usually flush their buffers, but store TS packets [cells] until the buffer fills, assuming that the data comes in a more or less continuous stream. In the case of data traffic, this assumption no longer holds, leading to the problem that the last IP packet [or Ethernet frame] will only be partly transmitted..."*

BS encapsulation can address this problem by inserting flush bytes one at a time which saves encoding latency and buffer requirements at the transmitter. A BS encapsulator can add pad bytes when there is no data in the output buffer leaving the option open to the possibility of a new Ethernet frame arriving at the interface to fill the remaining MPEG-TS payload. In contrast, the synchronisation mechanism of FLM uses a 'pointer' which is transmitted at the first byte of a MPEG-TS cell. Therefore, since the pointer points to the location of the first encapsulated frame, it has to have prior knowledge of where the first encapsulated frame begins. Similarly, FLM indicates post-frame padding using an End indicator which signifies that the remaining unused payload is considered as

padding. These mechanisms effectively impose a course flushing granularity in FLM. Hence FLM would require more memory to pre-construct the MPEG-TS and a further theoretical delay of up to 182 Tb (146 µs) above the store-and-forward latencies. In practice, work on a FLM type encapsulation, ULE [56], had suggested a 20 ms to 40 ms pre-construct time before flushing is performed to maximize utilization. In subsection 4.3.5, it is shown that maximum utilization was achieved in hardware despite the zero pre-construct used by the OBS Encoder.

3.1.2. Reasons Against Byte Stuffing Algorithms

This subsection discusses three primary reasons against the use of BS algorithms for high-speed packet networks from recent research in the field of encapsulation protocol design for network interfaces. These arguments are equally valid for BFWA networks, and even more so in MWS where the available capacity is generally greater than other forms of BFWA networks. The list of reasons identified from those research are summarised in point form below and elaborated in following paragraphs.

- BS induces large variability, unpredictability and jitter.
- BS requires twice the amount of buffer required at the receiver interface compared to FLM.
- BS has high complexity of implementation which limits its use in high-speed packet networks.

The first reason against the implementation of BS stems from its factor-of-two worst-case overhead. In theory, Byte stuffing can expand the original input data up to twice its original size which induces large variability, unpredictability and jitter [107,108]. This concern is addressed in detail in Section 3.2 where it is shown and evidenced by network traffic from a real network, that the variability caused by real LAN or WAN traffic is minimal. Nevertheless, a new stuffing algorithm, Optimized Byte Stuffing (OBS), was developed in this research to further reduce the variability. The OBS algorithm that is described in detail in subsection 3.1.4 is able to 'stuff' one character value instead of the two required for PPP BS. Moreover, since OBS frame delimiters use statistically optimal values for MWS traffic, residual variability and jitter can be reduced further still. In

subsection 3.2.4, simulation results using real Ethernet traces are shown to reduce the variability and jitter of PPP BS by around two thirds.

The second reason against using BS algorithms is that its implementation requires twice the buffer space compared to FLM at the receiver interface to prevent overflow from a possible factor-of-two worst case overhead. Although it is usually accepted that frames that actually double in size rarely occur naturally (if ever), the possibility for factor of two overhead cannot be ignored [109]. According to workers in the field, the receiver buffer has to be twice as large as the largest frame it expects to receive. "Without a factor of two safety margin, the network device would be open to malicious attack through artificially constructed pathological packets" [110]. Other work in the field [104] also mirrors this belief by assuming that:

> Byte Stuffing type encapsulation *"Creates random variable-length transmission overhead for different PDUs [frames] of the same size, which increases storage requirement, [up to twice the size of the largest expected frame], on the data link receiver for PDU extraction".*

This concern is addressed in subsection 3.3.4, where it is explained how an innovative Byte Stuffing decoder design can allow decoding before storage so that implementation of BS based receiver interface will require the same amount of buffer space as an FLM one. Essentially, a BS decoder implemented in a pipelined manner enables received frames to be decoded on-the-fly, which allows the Byte Stuffing decoding to be placed before the receiver buffer. Even if artificially constructed pathological packets appear in the network to incur a factor-of-two expansion, the enlarged frames will be decoded back to the original size before it is stored in the receiver buffer.

A third reason for not implementing BS type encapsulation is due to the preconception that it is complex to implement which limits its use to low-speed lines, such as telephone modems. In fact, the complexity of BS is seen as a primary motivation for recent work for alternative encapsulation protocols. For example, the recent research work [108] proposing a novel Simple Data Link (SDL) Protocol for packet encapsulation claimed:

Referring to *" Flag-based packet delineation [Byte Stuffing] ... used in the frame relay and PPP/HDLC data link protocols"; "The stuffing of the escape patterns by the transmitter and their removal by the receiver requires complex real-time pattern match and processing at bit/byte level, which compromises scalability for optical channel applications".*

In a similar view, a work [111] that focused on a new FLM method of encapsulating data packets onto MPEG-TS streams claimed:

"Since the data stream must be scanned for the special flag, this solution [Byte Stuffing] consumes considerable processing power which limits its use in high-speed systems. The HDLC [or Byte Stuffing] based framing schemes used in PPP over low-speed line is an example for this approach".

Section 3.3 of this thesis addresses this concern in detail where the implementation of BS proposed is shown to be extremely cheap in hardware. Both BS encoder and decoder can be implemented in a fraction of the FPGA Logic Elements of that required by the SDL FLM method proposed in the previous work. Although, the BS encoder and decoder intellectual property core designed in this research was implemented in low-cost FPGA, post synthesis results showed operational speeds of almost one giga-bits per second. Hence, BS type encapsulations are indeed not limited to low-speed lines as previously assumed.

3.1.3. The PPP Byte Stuffing Algorithm Description

The PPP Byte stuffing algorithm is described as an internet standard and defined in RFC 1662 [112] which is commonly used as an application layer encapsulation in the internet. While there are other mechanisms built into the Point to Point Protocol (PPP), this research is only interested in the mechanism that allows arbitrary segmentation and reassembly in PPP. This mechanism is a Byte Stuffing (BS) type algorithm and is referred here as the PPP BS algorithm. A possible implementation of the PPP BS

algorithm for encapsulating Ethernet frames into MPEG-TS cells is described in Figure 3-2.

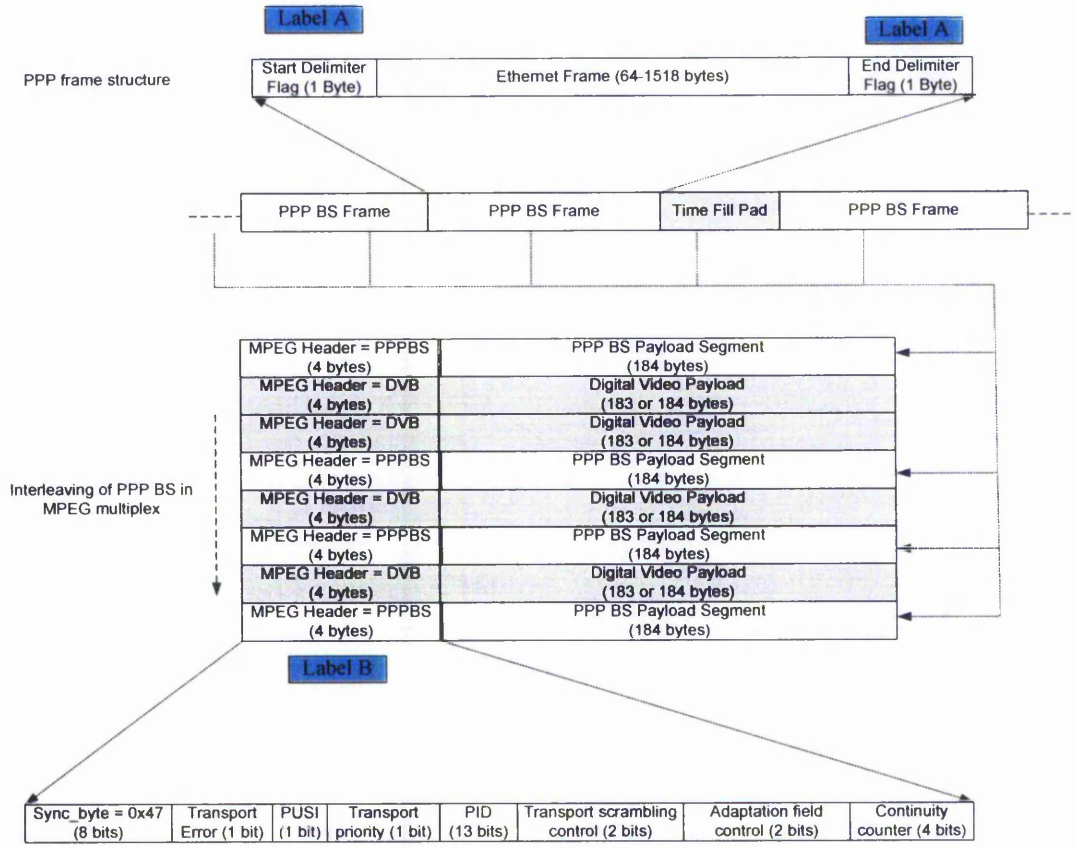


Figure 3-2: PPP Byte Stuffing for Ethernet over MPEG-TS

Frame encapsulation is achieved by delimiting the start and end of a frame with a Flag sequence that corresponds with the 0x7e byte value (Label A) and then eliminating this special flag sequence from the data payload of the frame. The data payload of each Ethernet frame is process as follows:

- Each occurrence of 0x7e is replaced by a 0x7d 0x5e sequence.
- Each occurrence of 0x7d is replaced by a 0x7d 0x5d sequence.

To achieve frame synchronisation, the data stream must be appended with the time fill sequence during inter-frame gaps which is also the same as the flag sequence 0x7e byte value. A receiver just joining a broadcast or one that has experienced an error event will

be able to (re)synchronise with the received byte stream by looking for the first non-0x7e byte in the stream. The first non-0x7e byte signifies the start of a new encoded Ethernet frame. A receiver can recover the original Ethernet frame payload by performing a reverse transformation. That is, every occurrence of 0x7d 0x5e is replaced by 0x7e, and 0x7d 0x5d is replaced by 0x7d. End of the encoded frame is reached when the receiver encounters another 0x7e delimiter in the byte stream.

An advantage of PPP BS is that it is able to apply end-of-frame/start-of-frame sharing to save one bytes worth of encapsulation overhead per frame when presented with back-to-back frames. In such a case, as the start and end frame delimiters uses the same flag, one of them may be omitted as shown in Figure 3-3. When comparing encapsulation algorithms down to the last byte of efficiency, this advantage may become significant.

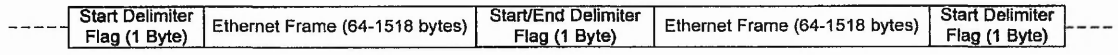


Figure 3-3: PPP Byte Stuffing encapsulation of back-to-back Ethernet frames

Note that the 4 byte MPEG Header portion (Label B) is assumed to be not explicitly used by the PPP BS algorithm but is retained for compatibility with MPEG-TS multiplexing.

3.1.4. Optimised Byte Stuffing (OBS) Algorithm Description

In this research, a modification is done to the PPP Byte Stuffing algorithm to achieve a different overhead characteristic more optimised towards MWS network traffic. The proposed Byte Stuffing algorithm is able to achieve synchronisation and encapsulation like PPP BS, but only one byte value is stuffed instead of two. This can potentially reduce the variability and unpredictability effects to a fraction of that present in PPP BS. Moreover, the stuffed character chosen for OBS is also statistically optimal for MWS traffic based on an analysis of real Ethernet data presented in Section 3.2 which could further improve the results.

An interesting Byte Stuffing algorithm, Consistent Overhead Byte Stuffing (COBS), invented by Stuart Cheshire [110] is able to perform frame encapsulation with negligible variability and unpredictability, however, it has to sacrifice the ability of being able to immediately generate output per byte of input. OBS on the other hand retains the encapsulation latency advantage of PPP BS over FLM techniques. The framing structure of OBS is shown in Figure 3-4 .

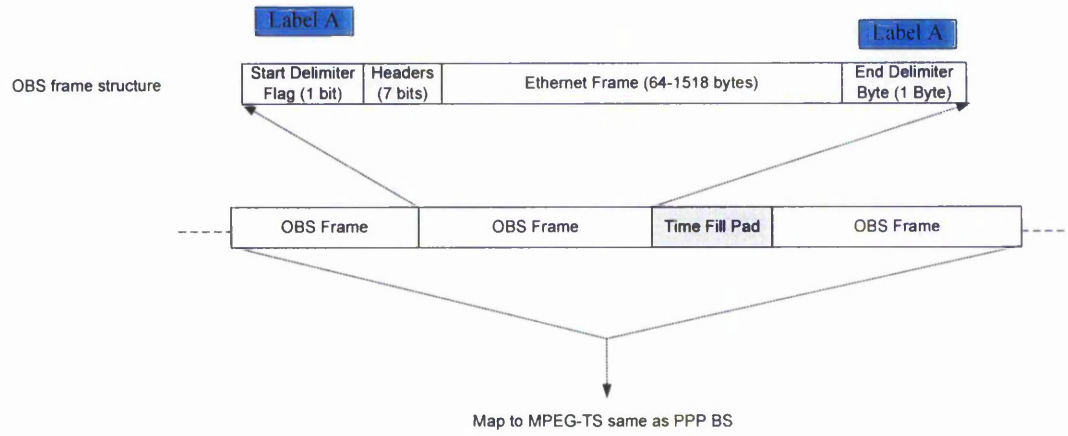


Figure 3-4: OBS framing structure

OBS achieves frame encapsulation and synchronisation differently from PPP BS. Instead of using Flag sequence of 0x7e byte value for both start and end-of-frame, OBS uses a one-bit Start delimiter and 0xEF byte End-delimiters (Labels A). The payload of each Ethernet frame is then processed in the following way:

- For each occurrence of 0xEF replace by a 0xEF 0xEF sequence.

Unlike PPP, OBS time fill byte values for inter frame gaps can be any value that satisfies the bitwise Boolean expression $TimeFill = '10xxxxxx'$. For simplicity, the value of $TimeFill = '10000000' = 0x80$ can be used. Frame synchronisation is achieved at the receiver by scanning each byte in the stream and searching for a byte containing the start delimiter bit, i.e. a byte that has the values of $SFDbyte = '0xxxxxxx'$. After the start of frame is found the receiver can begin recovering the payload of the Ethernet frame. The receiver will know that if two consecutive 0xEF characters are found, i.e. 0xEF 0xEF, then it must be a stuffed data payload and replaces it with a single 0xEF character to

recover the original data. If a single 0xEF character is found and is followed by any other character, then it must be the end-of-frame delimiter.

When comparing Ethernet payload encapsulation efficiency of PPP BS and OBS, the difference is that PPP BS is able to save a further 1 byte per frame at maximum throughput, while OBS can save a further $\left(\frac{1}{256}\right) = 0.39\%$ on average of overall throughput. This implies that PPP BS is more optimal for smaller frames while OBS for larger frames. A comparison between overhead for PPP BS, OBS and COBS based on real MWS traffic is done in Section 3.2.3. Although the extra 7bit of header space provided by OBS cannot be used to carry Ethernet payload, it can be very useful as flag indicators to signify the presence of other layer-2 mechanisms. While the Datalink layer mechanisms investigated in this research does not use (reserves) the extra header space provided, a possible header assignment is suggested as future work in Chapter 6.

Summary of Section 3.1

In this section, a key proposition of this research is presented; to use BS type encapsulation instead of FLM types typically employed in previous BFWA network interfaces. In the layer-2 based forwarding MWS investigated, it was shown theoretically that BS can improve encapsulation latency by 2 to 3 orders depending on the Ethernet frame size being processed. However, existing works in the field of protocol encapsulation design had assumed several concerns for the use of BS type encapsulation in high-speed packet based interfaces. An Optimised Byte Stuffing (OBS) algorithm derived from Point-to-Point (PPP) Byte Stuffing (BS) algorithm was created to address some of those concerns. While the solutions are briefly introduced in this section, they are dealt with in detail throughout in sections 3.2 and 3.3. To aid clarity, the table below shows specifically where the solutions for the identified problems can be found.

Problems	Solution further addressed in
BS induces large variability, unpredictability and jitter.	3.2 - 3.2.5.
BS requires twice the amount of buffer required at the receiver interface compared to FLM	3.3 - 3.3.3.
BS has high complexity of implementation which limits its use in high-speed packet networks	3.3 - 3.3.4.

3.2. MWS traffic characteristics and Algorithm comparison

Acquisition and analysis of real network traffic characteristics of a plausible scenario for the MWS are investigated. Unlike traces acquired from previous work, the network traces acquired in this research preserve frame information that influences the efficiency of Byte Stuffing type encapsulation algorithms. Network traffic traces from real networks provided by other research are usually 'sanitised' due to privacy reasons. As such, information in the headers are scrambled or replaced by a different subset. More significantly, such traces usually only retain packet headers since the research investigates higher layer characteristics of the internet such as internet caching, web request inter-arrival and TCP responses. Payload information is typically omitted for both storage and privacy reasons. In this research, the sanitisation done on the traces collected are random rearranging of bytes within a frame so that necessary header and payload information are retained. This enables BS and FLM encapsulation algorithm performance comparisons, and statistically optimal delimiter byte values for BS encapsulation to be obtained.

3.2.1. Protocol Distribution of MWS traffic

It is difficult to quantify and compare the efficiency of encapsulation algorithms as their behaviour is heavily dependent on the nature of the data traffic. Worst case or best case analysis gives little insight to the typical performance of an encapsulation algorithm as frames that do induce such behaviour rarely happen naturally, if at all. To investigate realistic performance, network traffic from a typical scenario is needed. Since the MWS considered in this research is envisaged to carry a mix of Local Area Network (LAN) and Wide Area Network (WAN) traffic, data from these environments were acquired and analysed. In this investigation, frame size distributions and byte value distributions are of interest, which are in turn dependent on the aggregate contribution of content types.

Characterising aggregate network traffic to obtain a dataset that represents typical WAN/LAN traffic is non trivial for many reasons. For instance, the internets

heterogeneous nature, network application diversity, and changing user behaviour can influence the composition of content [113]. It was therefore understood that "despite the value of Internet traffic measurements as a research methodology, any data set collected from an operational network represents but a snapshot at one point in time in the internets evolution" [102]. Rather than obtaining the characteristics of aggregate content from a single network, this research approximates observations of several real packet network content characterisations [114] – [124].

Based on results of those researches, a 'composition of content' pie chart, was formed to categorise a plausible aggregated traffic that can be found in a typical MWS environment. The pie chart is shown in Figure 3-5 below while details of how the chart was generated are described in the following paragraphs. This chart will provide the basis of a 'challenge' network trace for the comparison of encapsulation algorithms.

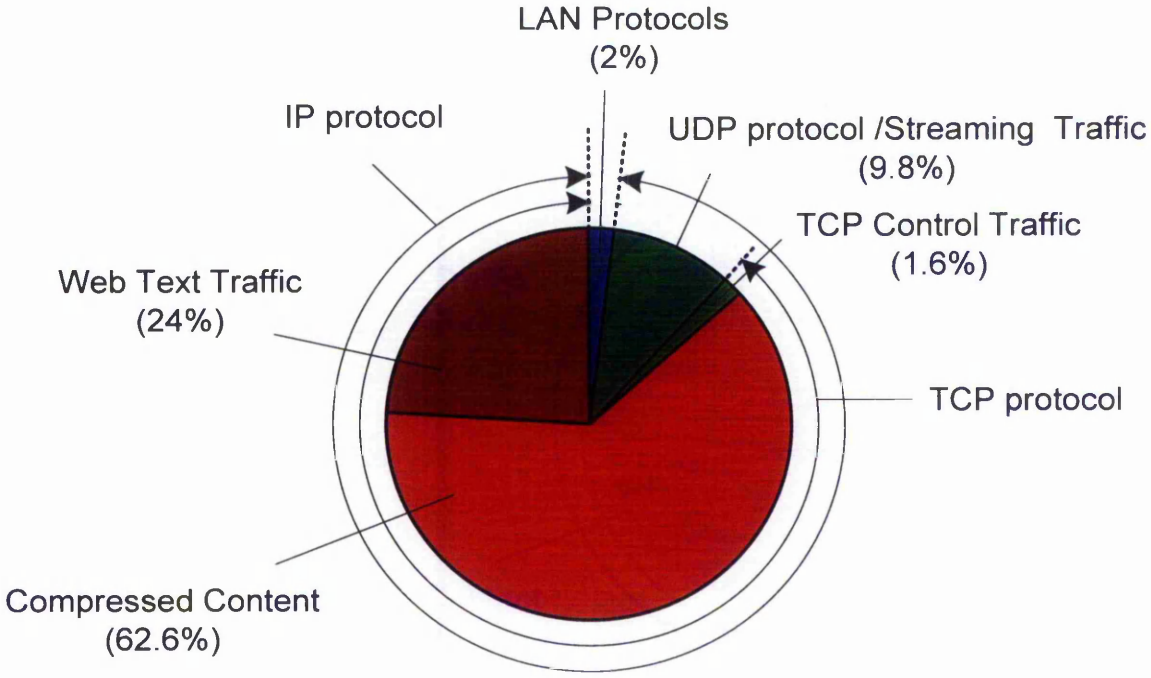


Figure 3-5: Estimated Composition of Content in MWS environment

The ratio between LAN traffic and IP traffic are based on the traffic distribution of large organisations investigated in [115,116,124]. In those networks it is normally observed that 98% of network load are IP traffic and 2% are LAN traffic. LAN traffic consists of conversations among hosts and internetworking devices over protocols such as Address

Resolution Protocol (ARP), Dynamic Host Configuration Protocol (DHCP), Border Gateway Protocol (BGP), Domain Name Server (DNS) and others. IP traffic dominates the vast majority of bandwidth and can be subdivided into TCP and UDP traffic. For IP traffic a 9:1 ratio between TCP and UDP was chosen as based on observations in [117,118,119]. In those works TCP traffic had always contributed to the majority of IP packets, from 85% to 95%. UDP typically serves streaming-media content which are highly compressed web objects that exhibit real-time characteristics. The content served by TCP traffic can be further partitioned into many content classes [123,124]. To simplify the 'composition of content' pie chart the content classes are grouped into compressed traffic, web text traffic and TCP control traffic. Compressed traffic which amounts to 62.6% of overall content comprises of highly compressed files or web objects such as JPG, GIF, ZIP, PDF normally transported over reliable TCP protocol [121,122]. Web text traffic takes up 24% of bandwidth representing uncompressed and unencrypted web text objects usually encoded in the hyper text mark-up language (html) code set. Finally TCP control traffic are non user payload traffic that are generated by TCP for its own mechanisms such as congestion control, repeat request and hand shaking.

Results of those researches also reveal that their aggregated traffic exhibits surprisingly consistent frame size distribution which suggests the predominance of minimum sized frames (64 bytes) and maximum sized frames (1518 bytes). This bimodal distribution of frame size is a generally accepted characteristic typical of network traffic and common to academic, core, dial-up, transit and leased-line networks [103]. As an example of such a distribution, a traffic trace from an academic campus network, studied in Scholten [104] and available from their internet repository, was processed using a Matlab program developed to obtain the graphs as shown in Figure 3-6 below.

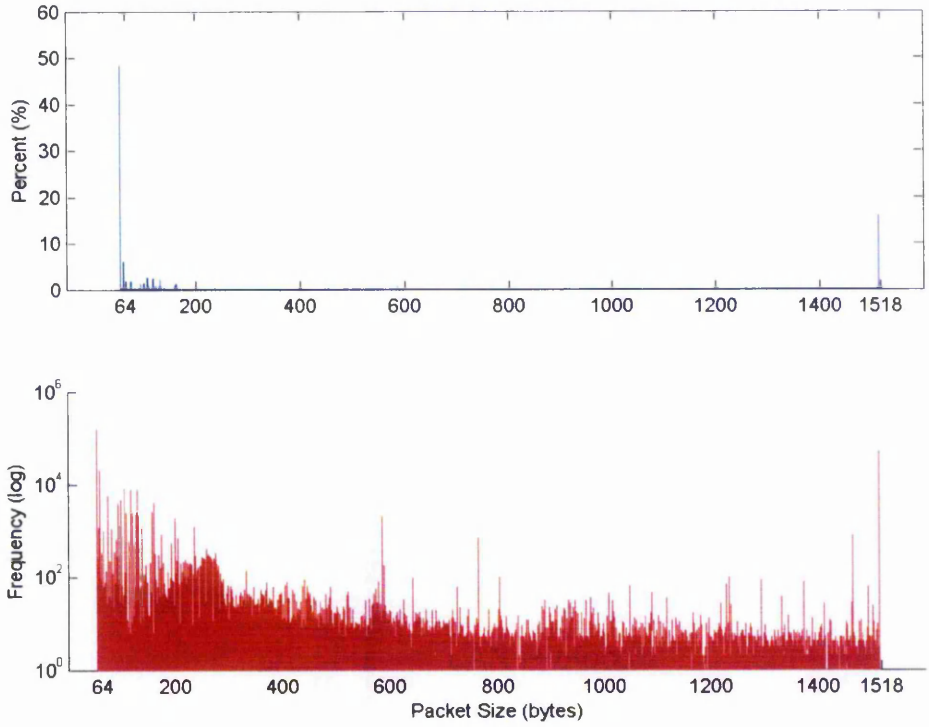


Figure 3-6: Frame Size Distribution from Lucas et.al. [115]

The bimodal distribution is usually attributed to the nature of the TCP/IP protocol over Ethernet networks. Fragmentation of large file transfers into Maximum Transmission Unit (MTU) of Ethernet can explain the high occurrence of maximum sized frames. On the other hand, TCP Control packets and LAN traffic explains high occurrence of small frames. Secondary spikes can also occur due to older TCP/IP implementations that do not perform MTU discovery, or streaming media over UDP/IP protocols.

3.2.2. Frame Size Distribution Profiles of Content Types

Traces were acquired from the Nottingham Trent University campus network to sample traffic from the five content types that make up the composition of content of a plausible MWS environment. The trace acquisition were neither conducted in very strict environment, nor was the trace counter checked for network and storage errors as done in the previous works reviewed. As this research is not primarily focused on

characterisation of networking traffic, it was felt that a basic acquisition methodology was sufficient.

The network traces presented in this research was acquired using the Ethereal tool on a Linux platform. Using specific filters for each content type, Ethereal logged real Ethernet frames in promiscuous mode received on the network interface card. The traces were then stored in the 'tcpdump' format, which is a common format used by other research to store and exchange such data. The common format allowed verification of the traces acquired in this research and the analysis MatLab program developed. The total acquired traffic consisted of 90 Mbytes of data with content ratios equivalent to the 'composition of content' pie chart determined in subsection 3.2.1. Comparing results of the acquired traffic shown in Figure 3-7 below and traffic from a previous research (Figure 3-6 above) shows that the traces are similar. The acquired traffic shows the typical bimodal distribution with secondary spikes due to old TCP implementations and streaming media.

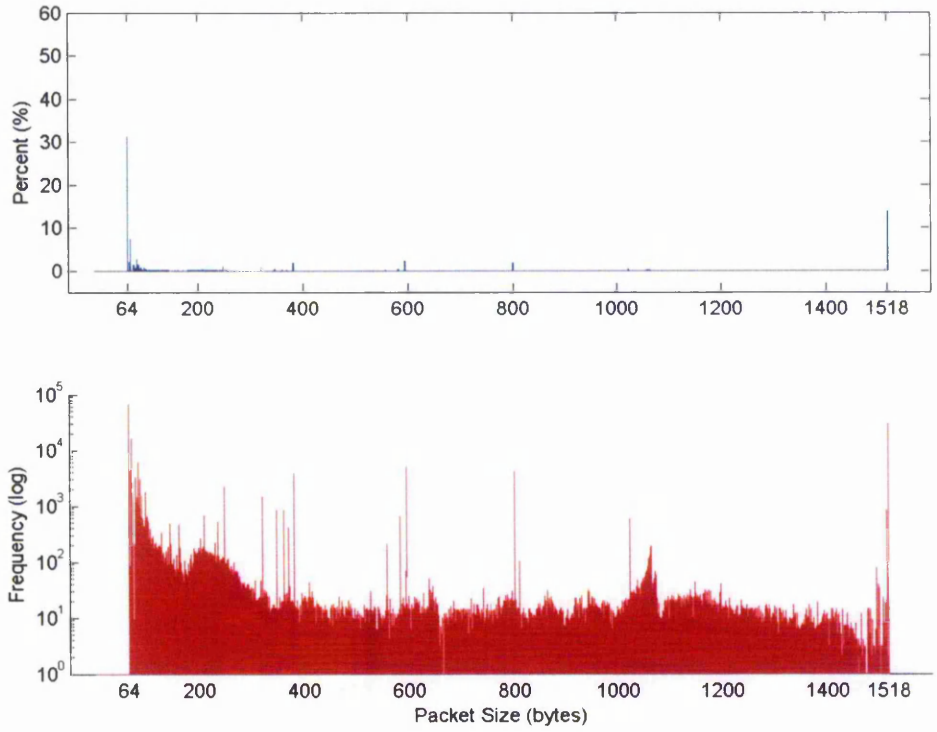


Figure 3-7: Aggregated MWS traffic frame size distribution

Four experiments in total were conducted; each collecting one of the identified content types which are: compressed content, web text, streaming media content, and internetworking LAN traffic. TCP Control traffic were naturally present in compressed and text content traces, hence there was no need for a specific experiment to acquire that content type. For all traces except for internetworking LAN trace, all traffic other than that to and from the workstation pc were filtered. The distribution of frame sizes of the captured aggregated MWS traffic is shown in Figure 3-7. In an attempt to understand the nature of this frame size distribution, each content type was examined individually and several observations were made.

A web page consists of many embedded objects each representing a paragraph of text, an icon, an image etc [122]. According to that work, a HTTP browser client establishes a TCP connection for each object to acquire it from the server. They had also found that about 85% of embedded objects are less than 10,000 bytes, and 50% are less than 1000 bytes. The frame distribution of compressed content and text content could be explained by examining a transaction of an object over TCP (Figure 3-8). Compressed traffic and text traffic were acquired from surfing websites high in the respective content (see Figure 3-9, Figure 3-10 respectively).

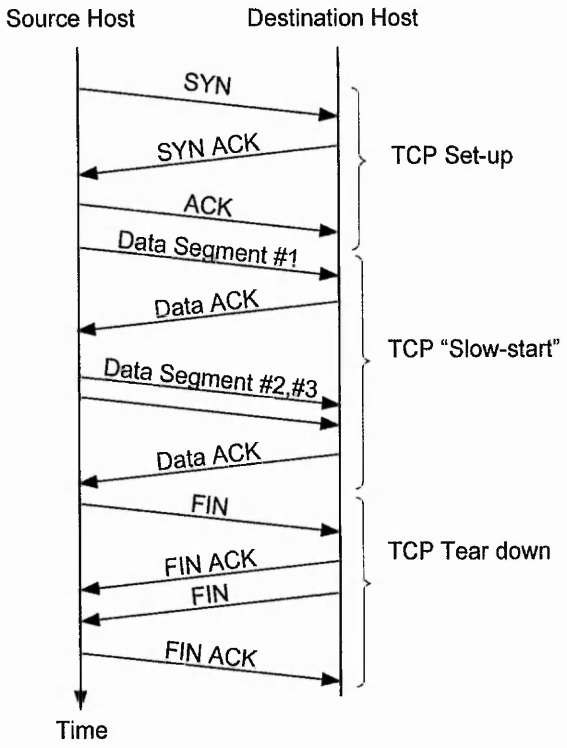


Figure 3-8: TCP Transaction Sequence

Both compressed and text content exhibits high occurrence of small packets due to TCP Control packets. As shown in figure above reliable connection set-up and tear-down, as well as 'TCP slow start' phase [125,126] induced nine TCP Control frames to transfer three data frames that can contain web objects up to 4500 bytes. Compressed content have a higher occurrence of maximum sized frames with a more or less even distribution in between. Its distribution can be attributed to compressed content objects being relatively larger; producing one or more maximum sized frame, and a tail frame that can be any size within the range. In contrast, text objects are generally small, skewing the distribution towards a heavier occurrence of smaller frames.

Frame distribution of Streaming media was observed to have a very different frame distribution from other traffic types; going against the bimodal distribution usually accepted (see Figure 3-11). This could be explained by the UDP/IP protocols used by Streaming media instead of TCP/IP. Hence, the frame distribution does not exhibit similarity to other types of content that use TCP/IP protocols. One sharp peak at frame size of 800 bytes was observed with a heavier distribution towards larger frame sizes. The single peak could be explained by constant bit rate audio or video streams. Although

a streaming content object is likely to be greater than 1518 bytes, the real-time nature probably forced fragmentation into sizes between 64 and 1518 bytes skewing slightly towards frame sizes between 1000 to 1400 bytes.

LAN traffic (Figure 3-12) has an unsurprisingly predominance (98%) of frames less than 600 bytes with minimum sized frames amounting to 48%. It was observed that the vast majority of traffic from this trace consists of conversational traffic between hosts that maintain the topology of the network using protocols such as ARP, DHCP and Microsoft's BROWSER protocols. Such conversational traffic are typically broadcast type that probably serve to refresh the existence of hosts and to detect broken or congested links within the network. It was also observed that there were at least 20 unique types of Ethernet II frames that uses the 13^{th} and 14^{th} bit as a type field as opposed to a length field in the IEEE 802.3 standard, suggesting that both standards are still currently used in real networking environments. This implies that if an FLM technique relies on the length field of an Ethernet frame, it would have to face the problem of determining the length of frames adhering to the Ethernet II standard.

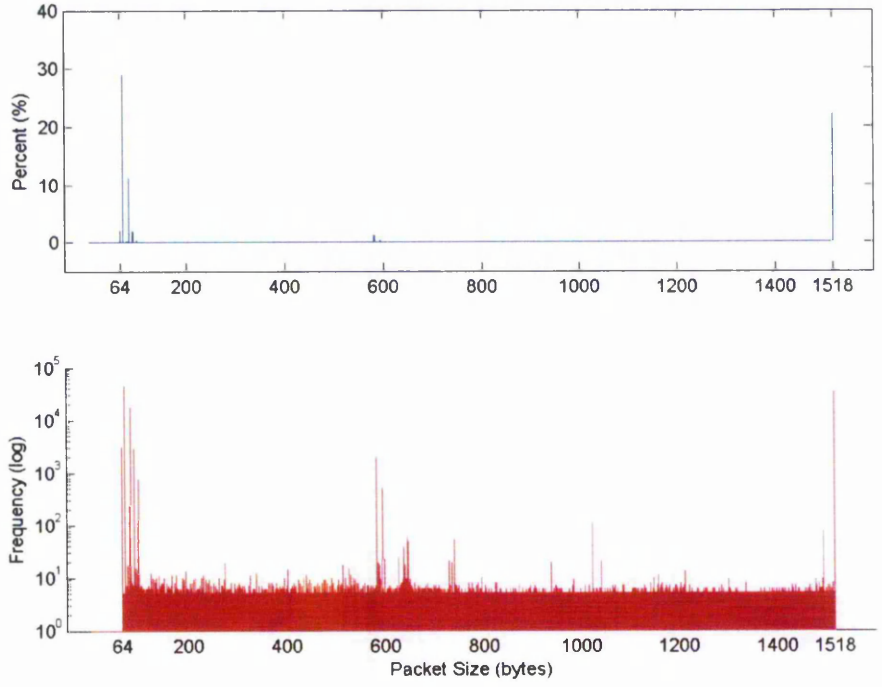


Figure 3-9: Compressed Content frame size distribution

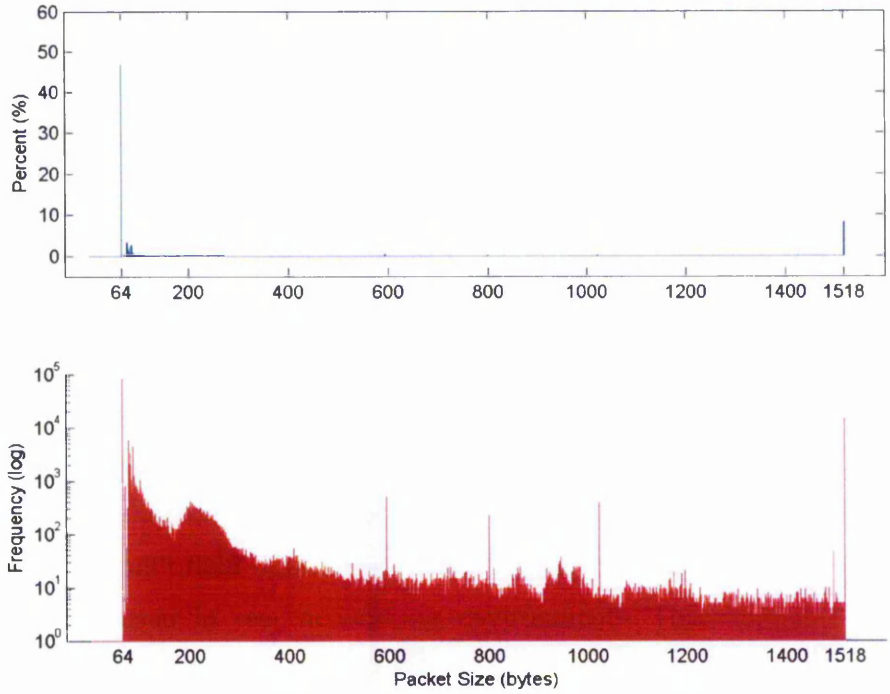


Figure 3-10: Web Text frame size distribution

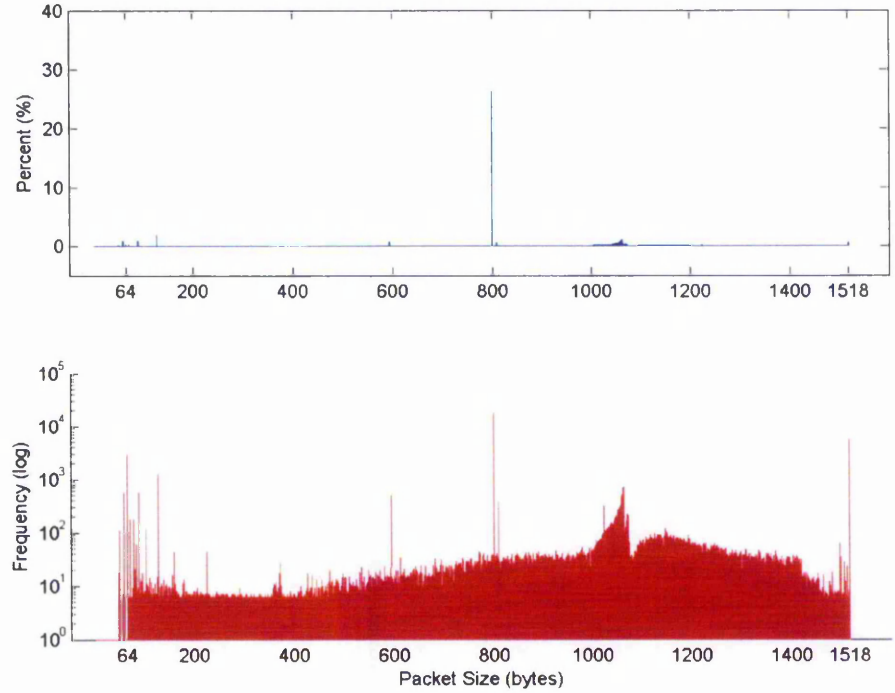


Figure 3-11: Streaming media frame size distribution

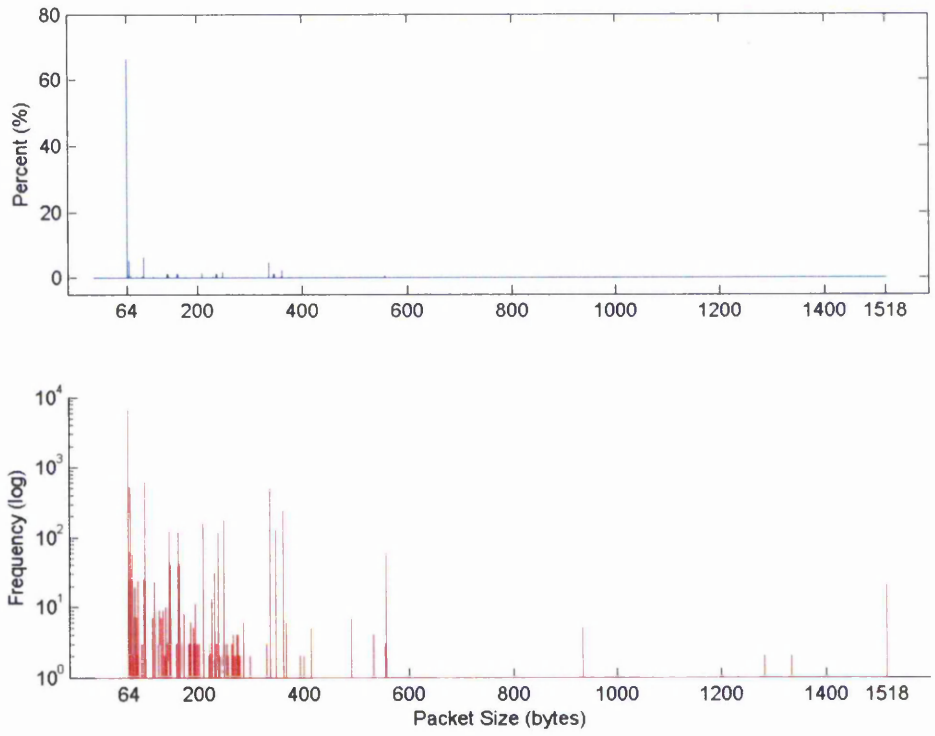


Figure 3-12: LAN Traffic frame size distribution

3.2.3. Byte Value Distribution Analysis

In FLM type encapsulation, "the total overhead for a transmission [only] depends on the packet length distribution and the encapsulation method selected" [127], but to determine overheads in Byte Stuffing algorithms the byte distribution must also be known. The traces acquired in this research retained byte value information from both payload and headers and were analysed to obtain the byte value distribution. The results of the analysis produced a distribution of byte values of the acquired aggregated MWS traffic, Figure 3-13.

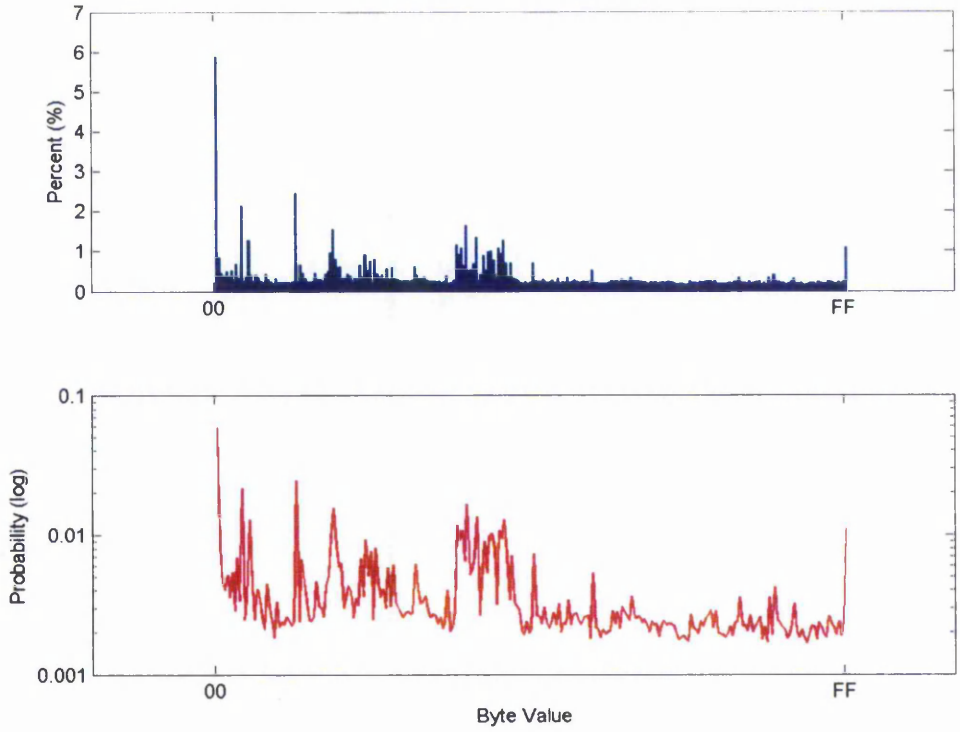


Figure 3-13: Composition of MWS byte value distribution

Each content type of the composition was examined individually from which some observations can be made. When compared earlier, compressed objects and streaming media content have dissimilar frame size distribution. However, when examining byte distribution both content types produced almost identical results with quite uniformly distributed byte values; see Figure 3-14 and Figure 3-15. The uniform byte value distribution can be explained by the nature of the content; as generally, compressed media exploits the entire range of available byte values to achieve a high level of compression. However, it is perhaps unsurprising that compressed content and streaming media are similar as both represent application layer compressed data. Their dissimilar frame size distributions are attributed to different mechanism of delivery.

When compressed objects and text content were compared earlier in subsection 3.2.2, they exhibited similarities in frame distributions. In contrast, their byte distributions are very different see Figure 3-14 and Figure 3-16. The byte distribution of text content can be explained by examining the Hyper Text Markup Language (HTML) code set. The

HTML language is a standard used in websites and supported by web browsers for representing textual documents and the associated formatting information needed to display them [128]. The byte values are skewed towards those representing the twenty six letters of the alphabet and common html characters such as '/', 'space', 'line-feed' and 'carriage-return' as these characters occur more frequently in documents formatted in HTML.

In LAN traffic traces Figure 3-17, about 30% of traffic was contributed by the '00' byte value. This is probably due to LAN messages needing less than the minimum sized Ethernet frame to convey information. Hence, the '00' characters occur more frequently as it is often used as the filler character. Since LAN traffic are usually small frames of a broadcast nature, frames containing the layer-2 broadcast address, 'FF FF FF FF FF FF', or similar layer-3 one of 255.255.255.255 i.e. 'FF.FF.FF.FF' could explain the secondary spike at the 'FF' byte value.

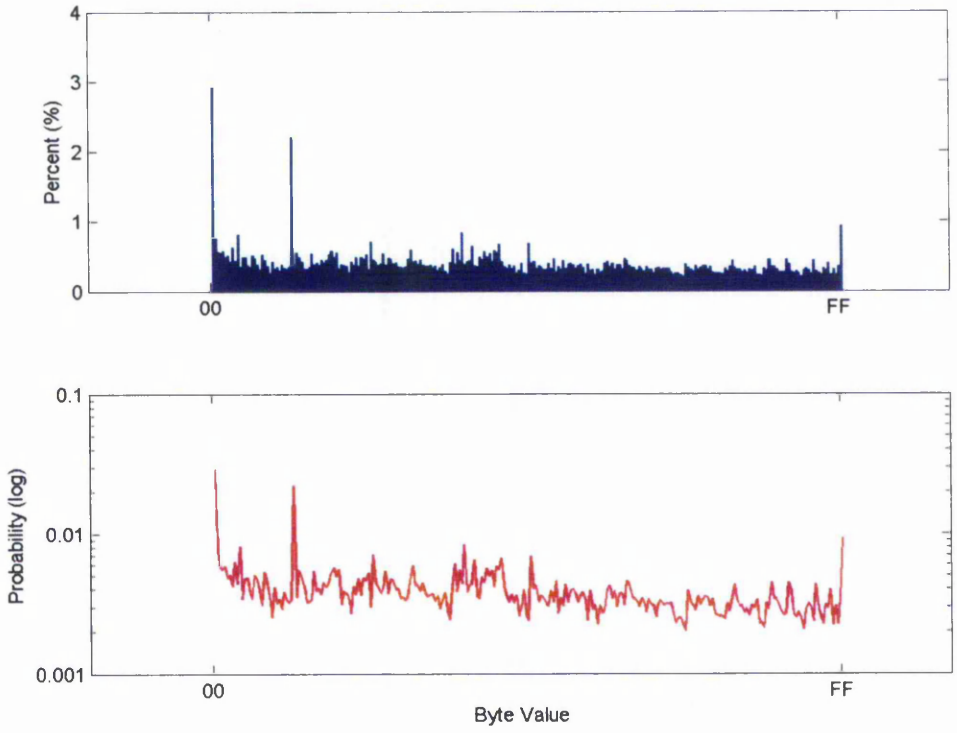


Figure 3-14: Web compressed content byte value distribution

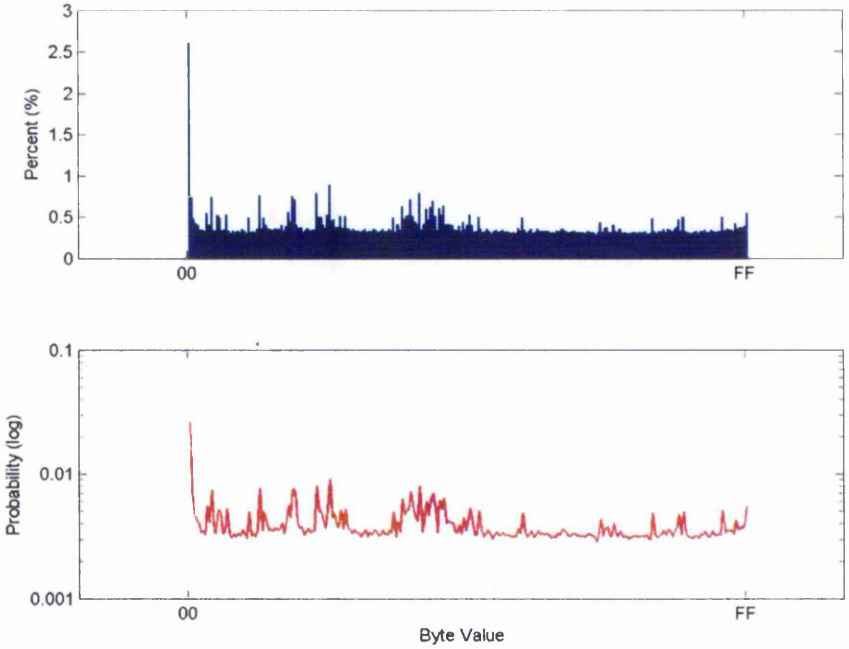


Figure 3-15: Streaming media content

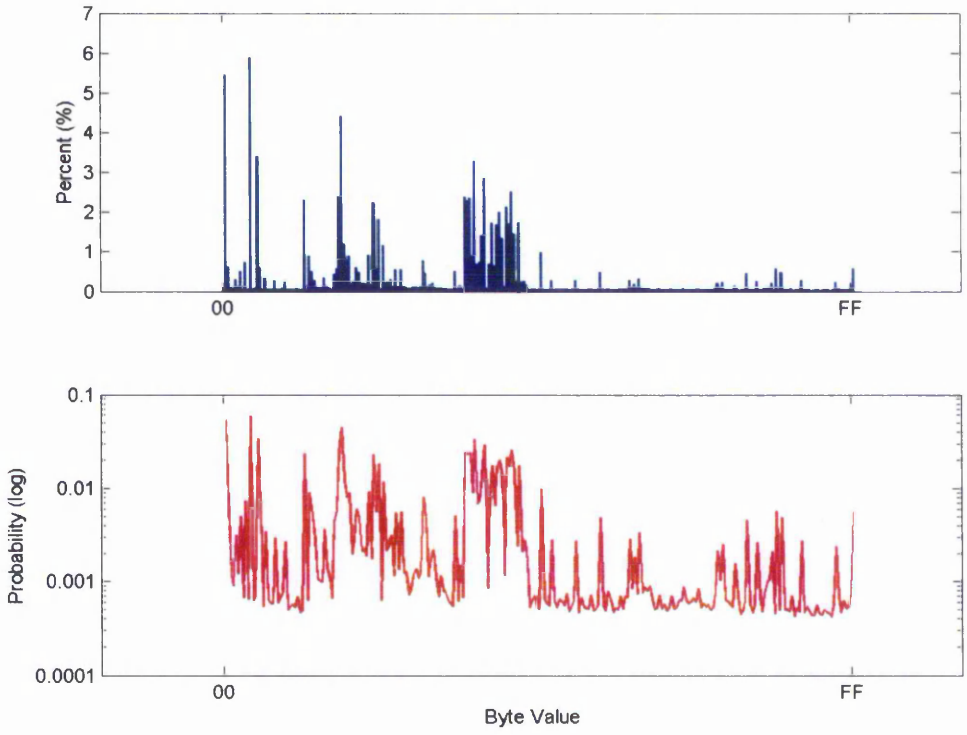


Figure 3-16: Web text content byte distribution

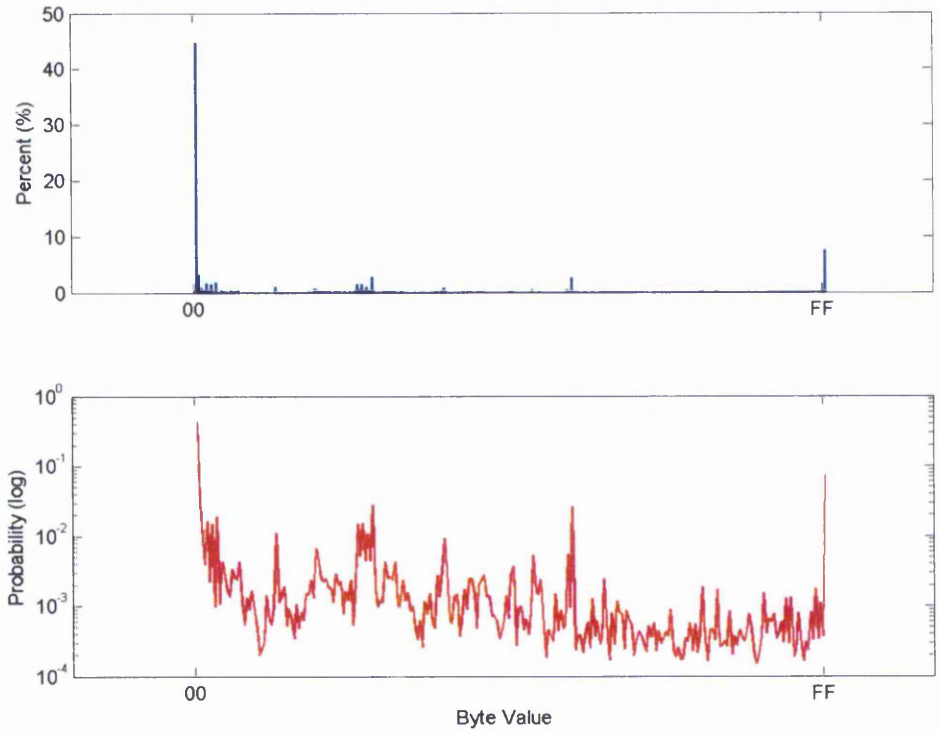


Figure 3-17: LAN traffic traces

As the internet is not static, it is understood that changes in the user behaviour and network application might alter the results of the byte value distribution experiments conducted here. User behaviour that surf websites that do not use the standard HTML code set to encode text objects (e.g. non-English websites), can cause high occurrence of characters popular in another language which would be reflected as peaks in byte value distribution. Also, the encryption of text objects can change the byte distribution results towards that of a uniform distribution. In Appendix B, SSL encrypted traffic high in text content are analysed which showed this property. The uniform distribution is a characteristic of encrypted data since encryption attempts to increase the entropy of the data to make the encryption more powerful. Due to these possible scenarios, the byte distribution based results presented here assumes that the aggregated MWS working traffic are of the commonly used standard type (i.e. uses unencrypted HTML) similar to that analysed in the work surveyed [114]-[123].

3.2.4. Overhead Comparison and Analysis of Encapsulation Algorithms

With both frame size and byte value distributions for a typical layer-2 MWS acquired, a fair comparison of a variety of Frame Length Marking (FLM) and Byte Stuffing (BS) encapsulation algorithms could be done. The primary focus of the investigation presented here is to determine the performance of BS encapsulation algorithms that have the ability to allow cut-through forwarding, and compare it to those adopted in previous systems. However, some other interesting encapsulation algorithms, GFP, ULE, and COBS which is beyond this context are also compared in this analysis to show that the DVB standardised encapsulation, adopted in previous BFWA systems, are quite suboptimal. In total, seven algorithms are compared, three BS algorithms; OBS, PPPBS, and COBS and four FLM algorithms DVB Section; PESSAP, GFP, and ULE. To aid clarity, Table 3-1 provides a summary of properties of the encapsulation algorithms compared, while the benchmark results of the algorithms are shown in Figure 3-18.

Encapsulation Algorithm	Cut-through forwarding	Byte Stuffing (BS)	Frame Length Marking (FLM)	Previously adopted in BFWA systems
Digital Video Broadcasting, Packet Elementary Stream, Service Access Point (PESSAP)			√	√
Digital Video Broadcasting, Section, Service Access Point (Section SAP)			√	√
Generic Framing Procedure (GFP)			√	
Ultra Lightweight Encapsulation (ULE)			√	
Consistent Overhead Byte Stuffing (COBS)		√		
Point-to-Point Protocol Byte Stuffing (PPP BS)	√	√		
Optimized Byte Stuffing (OBS)	√	√		

Table 3-1: Summary of Encapsulation algorithm properties

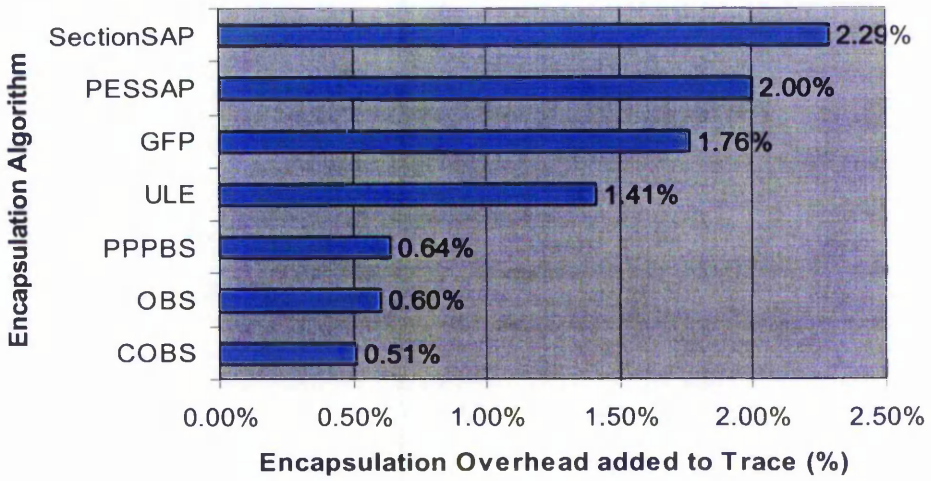


Figure 3-18: Benchmark results for encapsulation algorithms

Among the encapsulation algorithms analysed, the Byte Stuffing type encapsulations managed to introduce less than half the overhead of FLM algorithms. The best algorithm in terms of efficiency is COBS which introduces only 0.51% overhead, beating PPP BS and OBS by a marginal 0.1%. While it was found that COBS was the best option in store-and-forward interfaces because of its high efficiency, it was not chosen for further development since the implementation of a cut-through MWS interface was sought. COBS is not able to immediately generate an output for each byte of input because it processes input as a chunks of code blocks, but as a consequence allow it to perform more efficiently. In the words of the author of COBS: "It is perhaps unsurprising that giving a byte stuffing algorithm access to more data allows it to perform better" [129].

Examination of OBS and PPP BS algorithms reveal that PPP BS is able to save an extra 1 byte per frame at maximum throughput over OBS, while OBS is able to potentially save $\left(\frac{1}{256}\right) = 0.39\%$ of overall throughput over PPP BS. However, from the trace results the efficiency score can be considered equivalent for practical purposes. The BS algorithms only incurred overheads of 0.64% for PPPBS, and 0.60% for OBS. To further analyse, the advantage of the optimised character chosen in OBS gave a mere 0.05 % $\left(\frac{0.2392\% + 0.1973\%}{2} - 0.1681\%\right)$ advantage over PPP BS since the characters chosen by PPPBS algorithm had already a low probability of occurrence. However, due to the closeness of the efficiency scores, the choice of the optimal value in OBS produced the

winning margin of 0.04% (0.64% - 0.60%) less overhead over PPP BS. If it is assumed that OBS had used a delimiter byte value with the same probability of occurrence as those chosen in PPP BS, OBS would have incurred 0.01% ((0.64% – 0.05%) – 0.60%) more overhead than PPP BS.

3.2.5. Unpredictability and Jitter in BS algorithms

The ten statistically most and least occurring byte values based on trace acquired are summarised in Table 3-2. PPP BS's choice of characters which are 0x7d and 0x7e statistically occurred 0.2392% and 0.1973% of the time respectively. Albeit not within the top 10 most optimal values, the fact that neither of the two values incurred a particularly high overhead is a fortunate coincidence; the byte values chosen was based on the evolution from a serial transmission protocol that used zeros to represent the start and stop bits. 0x7e byte value was inherited in PPP BS since the serial sequence of "01111110" will be generated when written in bit form.

Placing	Worst Values (Hex)	Percent (%)
1	0x00	5.8895
2	0x20	2.4614
3	0x0A	2.1440
4	0x65	1.6451
5	0x2F	1.5611
6	0x69	1.3411
7	0x0D	1.2857
8	0x74	1.2716
9	0x61	1.1696
10	0xFF	1.1094

Placing	Best Values (Hex)	Percent (%)
1	0xEF	0.1681
2	0xDF	0.1690
3	0xBF	0.1710
4	0xBB	0.1761
5	0xBE	0.1763
6	0xF3	0.1775
7	0xDD	0.1783
8	0xE7	0.1793
9	0xEE	0.1808
10	0x97	0.1810

Table 3-2: Ten worst and best values for Byte Stuffing delimiter characters

Based on the analysis of the acquired trace, it was found that OBS and PPPBS algorithms induce very small amounts of unpredictability and jitter, which opposes the general preconception that the expansion by up to a factor-of-two in BS algorithms would induce large unpredictability and jitter. In the acquired trace, the amount of frames that actually incurred more than 20 bytes expansion is merely 0.08% and 0.30% for OBS and PPPBS respectively. Although one maximum sized frame in the trace

incurred 160 and 34 bytes of expansion for PPPBS and OBS respectively, this is far from the 1500 bytes worst case overhead which the concern expressed in previous works.

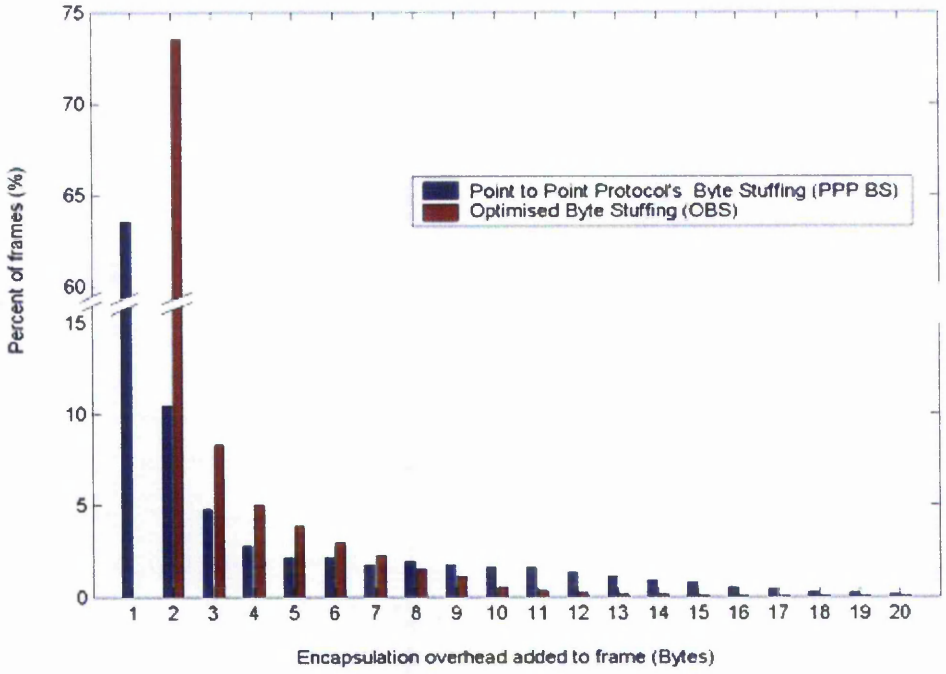


Figure 3-19: Expansion spread of PPPBS and OBS

Analysis of the expansion spread of OBS and PPPBS reveal that OBS was able to further reduce the expansion caused by PPP BS. Figure 3-19 compares the expansion spread of PPPBS and OBS by grouping the frames from the acquired trace according to the number of encapsulation overhead bytes incurred (encapsulation overhead above 20 bytes are not shown). In OBS, 73% of frames did not incur any expansion due to stuffing. This is a 9% margin compared to the 64% of non-expanded frames in PPP BS. In the graph, the frames incurring no overhead due to stuffing for OBS are grouped as 2 byte encapsulation overhead. Since 2 bytes is the minimum required overhead for start-of-frame and end-of-frame delimiters for OBS, there is no result for OBS with 1 byte overhead in the figure. For PPP BS, frames incurring no overhead due to stuffing are grouped as 1 byte encapsulation overhead, as this is the minimum for this algorithm assuming the end-of-frame/start-of-frame sharing perk was applied. The crossing point between OBS and PPP BS expansion curve is 8 bytes of overhead, above which OBS

performs significantly better than PPP BS. Quantitatively, OBS had only 4.21% of frames incurring 8 bytes or more expansion while PPP BS had 12.6%.

Summary of Section 3.2

The investigation presented in this section acquired and studied real Ethernet traffic that retained byte value information, usually omitted in other studies, but were required here to compare performance of the Byte Stuffing algorithms. Whilst the primary reason to use Byte Stuffing encapsulation was to reduce latency and allow cut-through forwarding in the MWS interface investigated, it was found that the Byte Stuffing encapsulation can also increase efficiency between 0.8% and 1.65% compared to FLM techniques surveyed. The major drawback of Byte Stuffing algorithms is its potential for up to factor-of-two overhead, which can induce service-level unpredictability and jitter. However, byte level analysis of the real Ethernet traffic acquired showed that frames causing unpredictability and jitter were far from the factor-of-two or 100% expansion. Also, the proposed OBS algorithm managed to significantly reduce the variability of PPPBS. The maximum frame expansions encountered in the trace are 10.5% $\left(\frac{160}{1518}\times 100\%\right)$ and 2.2% $\left(\frac{34}{1500}\times 100\%\right)$ for PPPBS and OBS respectively. Most frames induced little variability and jitter in both PPPBS and OBS as the results show that 87.4% and 95.8% of frames incurred only 7 bytes or less overheads respectively.

3.3. Hardware Implementation of Optimised Byte Stuffing Encoder and Decoder

The first subsection illustrates the top-down VHDL design flow methodology used to design hardware intellectual property cores in this research. Subsections 3.3.2 and 3.3.3 explain the functional operation of the OBS Encoder and Decoder cores from abstracted top-level blocks of down to individual sub-modules blocks. Possible Register Transfer Logic (RTL) level implementations of the cores are also discussed to show how the OBS Encoder and Decoder can be implemented in just 34 and 51 Altera Logic Elements respectively. Whilst an ideal OBS encoder/decoder can immediately generate an output without any processing latency, the hardware implementation developed introduces some pipelining latency. In the last subsection 3.3.4, the OBS encoder and decoder hardware are evaluated using simulation to find out how close it is from the ideal implementation.

3.3.1. EDA Design Flow Methodology

The Very High speed integrated circuit Hardware Description Language (VHDL) was chosen as the design language for this work. Intellectual properties described in VHDL using standardized IEEE libraries are suitable for research work as the resulting source code is vendor independent, making it portable across other platforms with little modifications. Furthermore, existing source codes available from the Internet and other sources can be used or modified to speed up the design cycle of this research. The benefit of VHDL was demonstrated when the designs were successfully ported from a FLEX device implementation to a more updated and low-cost Cyclone device family that became recently available.

Quality of the VHDL intellectual property developed was maintained by revision documentation and well commented source codes. The intellectual property source codes developed in this research adopted a design flow that made use of the Electronic Design Automation (EDA) facilities available in the university. The EDA facilities cover the basic tasks of a standard design cycle [130]:- design entry, design simulation and verification, design debugging, design synthesis, and design place & route. Development

of the source code usually takes up 70% of a hardware design process [130], therefore emphasis on refining these stages to optimise usage of available design hours. A 1 GHz Pentium III workstation with 512 Mbytes of memory running Windows XP operating system was provided for this research along with the EDA tools. The diagram below illustrates the software tools, process flows, and output file formats used.

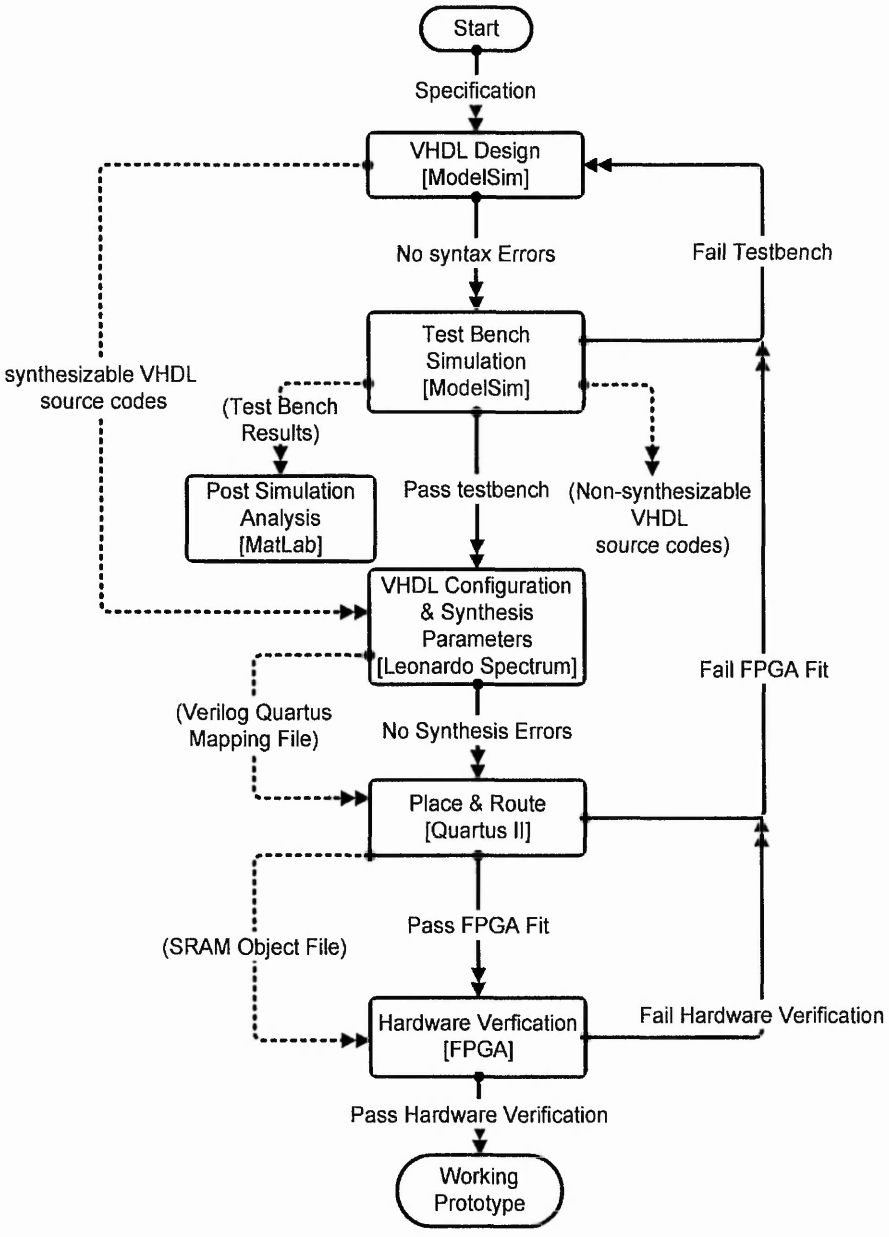


Figure 3-20: Electronic Design Automation (EDA) Design Flow used in Research

VHDL source code design entry is done in the ModelSim tool which are described at Register Transfer Logic (RTL) level. This results in synthesizable VHDL which can be

interpreted by the synthesis tool. For simulation process, VHDL test benches are described at the behavioural level since behavioural designs are virtual components which are generally quicker to code. However, as the behavioural components are considered non-synthesizable VHDL, it cannot be interpreted by the synthesis tool.

Design simulation and verification process are automated by the developed test-benches that take advantage of special features of VHDL. In the VHDL test bench environment, signal vectors can be verified during simulation while assertion statements can be used to halt on a fault or display a warning at any point in the source code. An alternative simulation method initially evaluated - Altera Quartus II v.2.2 graphical waveform simulation environment - does not support VHDL test bench features [131]. As such, only limited post processing of the simulation waveforms output was able to be done automatically. In such an environment, designers often have to resort to manually studying the waveforms.

Once the test-bench completes simulation and functionally verifies the designs, the synthesis tool synthesizes the design into a structural-level specific for the target device, e.g. Cyclone FPGA logic elements. The structural-level design is passed to Quartus II for Place and Route. A detailed example of the synthesis process can be found in subsection 4.2.5. The output is an SRAM object file which is used by the built in programmer provided in Quartus II to program an FPGA. An output from the developed test-bench is raw test results written to a generic ASCII file. These test results are passed to a Matlab analysis program developed for post simulation analysis such as that described in subsection 4.3.5.

3.3.2. Optimised Byte Stuffing (OBS) Encoder Design

Adhering to the EDA design flow methodology, the OBS Encoder core was designed as a module so that it can be easily integrated with other modules to form the MWS interfaces investigated in this research. In the MWS interfaces, the primary task of the OBS encoder is to encapsulate Ethernet frames so that it can be arbitrarily segmented in to MPEG-TS cells.

Module 1 OBS Encoder

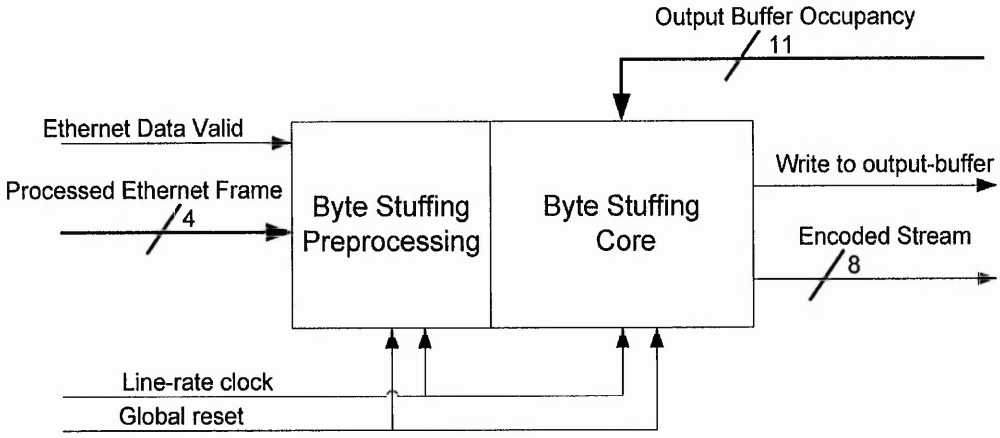


Figure 3-21: Byte Stuffing Encoder Block Diagram

The top-level block diagram is shown in Figure 3-21 showing its input and output interfaces to other modules and global signals. In the MWS interfaces, the OBS Encoder is placed after framing and field related processing modules, but before the output buffer which stores encoded Ethernet frames. Modules for framing and field related processing on the original Ethernet frames such as preamble suppression and subnetwork addressing, were designed to follow and synchronise with the ingress MII interface format: one Ethernet_data_valid signal and a 4-bit bus carrying the data body of the Ethernet frame. Since the OBS encoder processes frames in units of bytes, it requires that all frames are pre-processed into a convenient format before being input to it. In the Byte Stuffing pre-processing module, the bus widths of the frames are reformatted from 4-bits (nibbles) to 8-bits (bytes) and pad_bytes are inserted following each byte of the Ethernet frame data. By having the pad_bytes after each Ethernet data byte, the Encoder will gain an extra clock cycle allowing it to replace the pad_byte with a stuffing byte should it be needed. The format after the Byte Stuffing pre-processing will hereon be referred to as '*alternate-padded Ethernet Frame*' format and is shown in the waveform diagram (Figure 3-22).

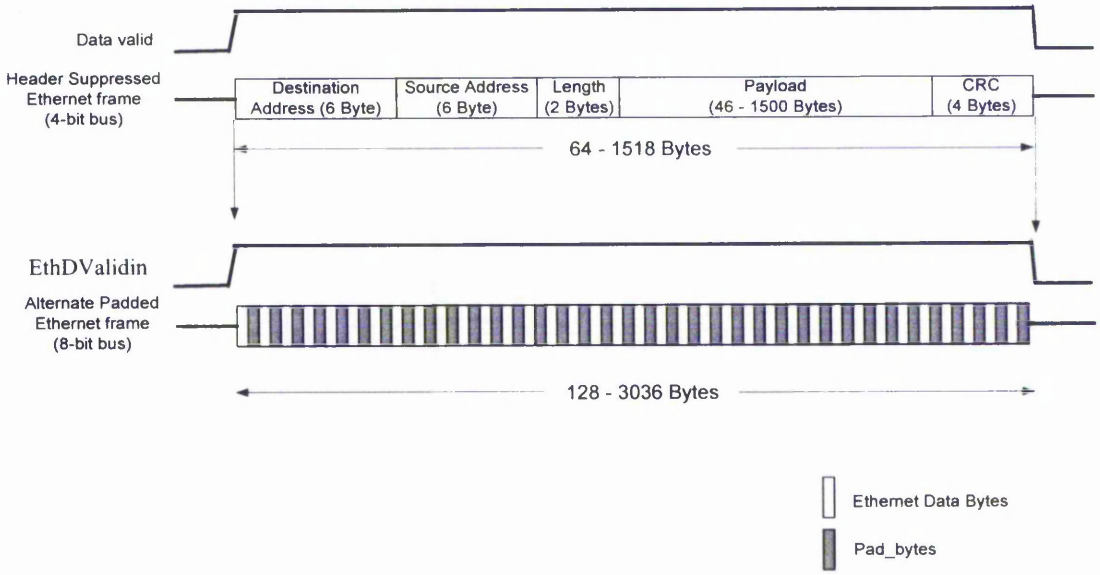
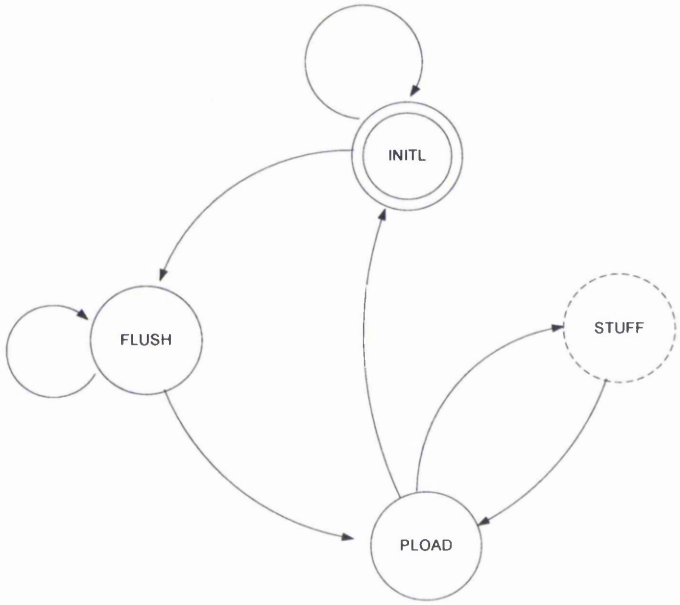


Figure 3-22: Pre-processing of Ethernet frame to 'alternate-padded Ethernet Frame' format

The BS Encoder operates in two modes: 'frame encoding' and 'flushing' modes. Frame encoding mode is activated when there is an Ethernet frame present on the interface to be processed. Otherwise, during idle periods, flushing mode is activated. The operation of the two modes can be understood by examining the OBS state machine illustrated in Figure 3-23:



Transition #	From State	To State	Condition
1	pload	stuff	EthDValidin
2	stuff	pload	
3	flush	pload	EthDValidin
4	flush	flush	!EthDValidin
5	initl	flush	!EthDValidin
6	pload	initl	!EthDValidin
7	initl	initl	EthDValidin

Figure 3-23: State machine of OBS Encoder

Consisting of four states, the state machine scans for the presence of a valid Ethernet frame on the interface and cycles between the states on each positive edge clock pulse. The presence of an Ethernet Frame on the interface is known by scanning the EthDValidin signal as shown in Figure 3-22. In the state machine transition table (Figure 3-23), a frame-present event on the interface is denoted by EthDValidin and absence as !EthDValidin. Each state produces a unique signal that controls a multiplexer to produce the output as required by the algorithm. A pipeline register latches a byte of alternate-padded Ethernet Frame from the input port, effectively giving one clock cycle time for the combinatorial logic of the state machine and multiplexer to decide and produce a byte of output.

In the Frame Encoding mode, the 'pload' and 'stuff' states are utilised which scans every payload byte on the pipe for the 0xEF delimiter character and replaces the following pad_byte with a stuff_byte whenever it is found. Of course, only data_bytes and stuff_bytes are written into the output buffer while the pad_bytes are discarded. The initial Start_of_frame delimiter (0x80) is inserted during transitions from the Flush mode to Frame encoding mode while the end_of_frame delimiter (0xEF) is inserted during transitions from Frame encoding to Flush mode.

Flush mode utilising the 'flush' state is activated when there is no frame on the interface i.e. during inter-frame gaps. As a constant stream of data is not guaranteed in Ethernet, flushing is necessary to prevent the MPEG-TS framer from reading an empty output buffer which would cause buffer underflow errors. However, as flushing means writing pad_bytes into the output buffer, excessive flushing will cause an increase in initial

latency and wastage of bandwidth. 'Flushing mode' is activated by the *Byte Stuffing Encoder* to insert pad bytes when the *Tx Buffer* is empty and no Ethernet frame is entering the interface. Therefore the BS Encoder implements dynamic flushing, which only adds pad_bytes with a granularity of one byte at a time when needed. Dynamic flushing is achieved by constantly monitoring the occupancy level of the output buffer via the Output_buffer_occupancy port provided by the output buffer core. Simplistically, whenever the Output_buffer_occupancy signal indicates only one byte left in the buffer, the BS encoder will flush a single pad_byte to make sure that it always has at least one byte of data buffered.

While the design of the OBS encoder was described in a higher level VHDL code, a possible implementation in Register Transfer Logic (RTL) is shown below. Here the abstracted state machine block can trivially be converted to RTL level registers using one of the methods described in electronic design textbooks such as [132] from the state diagram and table provided above.

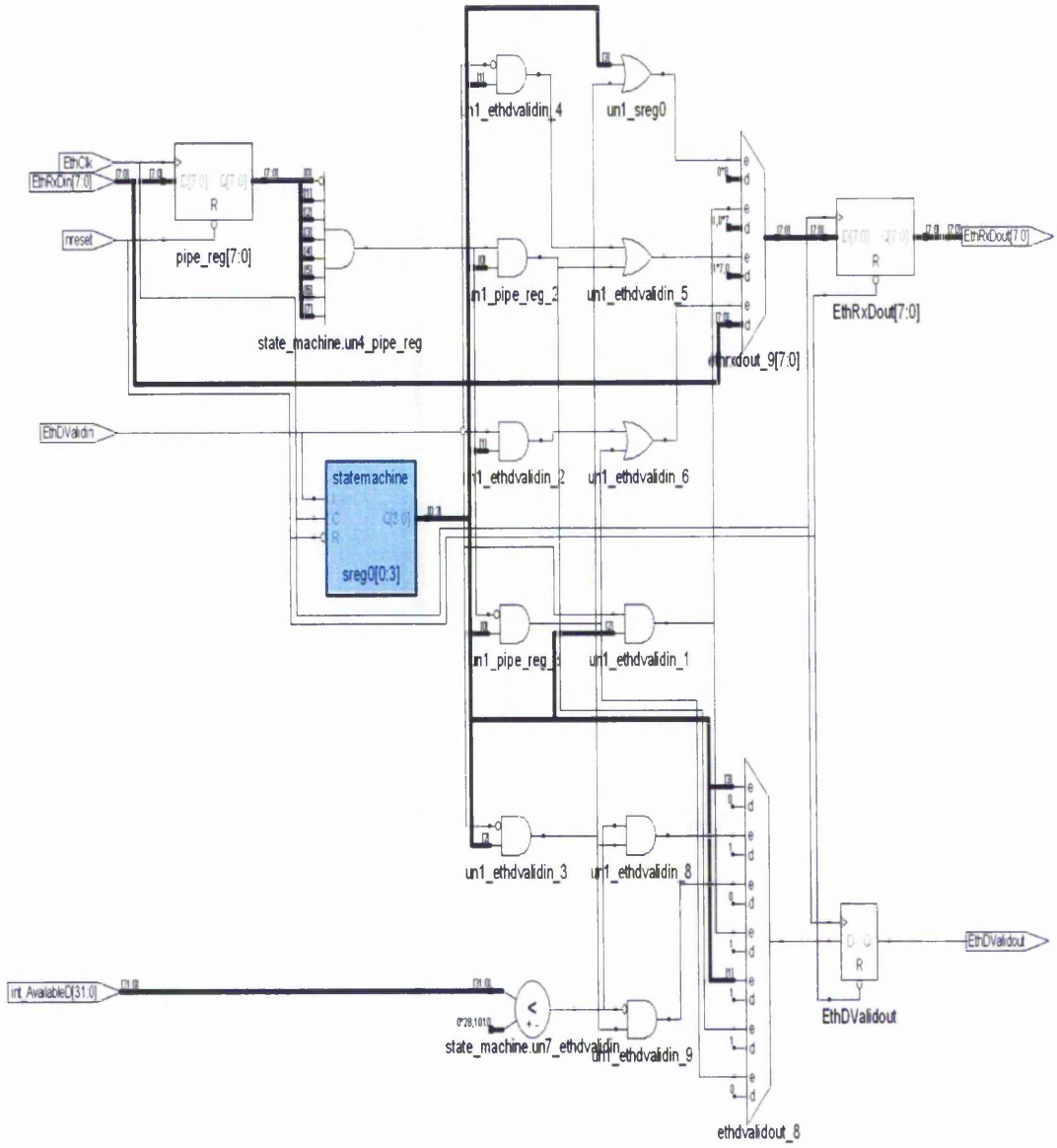


Figure 3-24: Possible Register Transfer Logic (RTL) implementation of OBS encoder

The architecture of the Byte Stuffing Encoder consists of output multiplexer, a state machine, pipeline registers and combinatorial logic. Also included in the design are output registers. While the output registers, EthRxDout[7..0] and EthDValidout, are not essential to the functional operation, they are included to terminate the combinatorial logics to provide better synchronisation to the output interface enabling faster operation.

In the MWS system investigated, the Ethernet interface has a higher capacity than the MWS wireless interface and hence partially encoded Ethernet frames can be segmented into MPEG-TS cells without the possibility of mid packet buffer underflow. Hence, the design of the OBS Encoder does not consider packet underflow conditions. This is a reasonable design assumption; since wired networks generally have higher capacity than wireless ones, and is necessary for cut-through forwarding modes.

3.3.3. Optimised Byte Stuffing (OBS) Decoder Design

A design challenge of the OBS decoder is to perform byte stuffing decoding without an input buffer. In the absence of an input buffer the OBS Decoder has to operate at DVB-SPI line speed, and is not allowed to throttle the input data stream for processing or else data loss might occur. This restriction is imposed to reduce cost of the OBS decoder on memory blocks, and provide a solution to address a general concern of previous work whereby it was believed that receivers implementing BS type encapsulation would require twice the amount of memory compared to FLM based receivers (see subsection 3.1.2). Previous work assumed that although frames causing the worst case overhead in BS rarely occur naturally, receivers implementing BS decoders are forced to provide input buffers twice as large as the maximum expected frame size to protect against buffer overflow due malicious users exploiting the weakness of BS type encapsulation. However, the architecture of the MWS receiver proposed in this research eliminates the need for input buffers. Without the input buffer, frames are not stored in memory prior to decoding. Hence even worst case expanded Ethernet frames are decoded to its original size before they are stored in the output buffer. The top-level block diagram of the OBS decoder is illustrated in Figure 3-25.

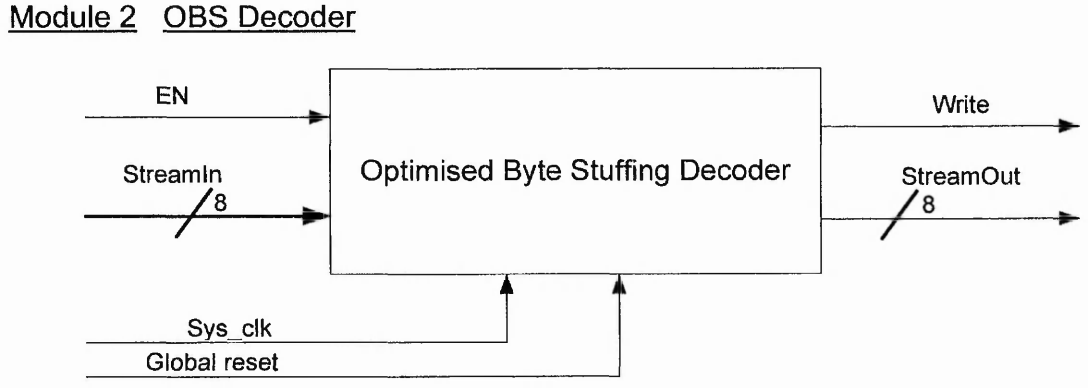
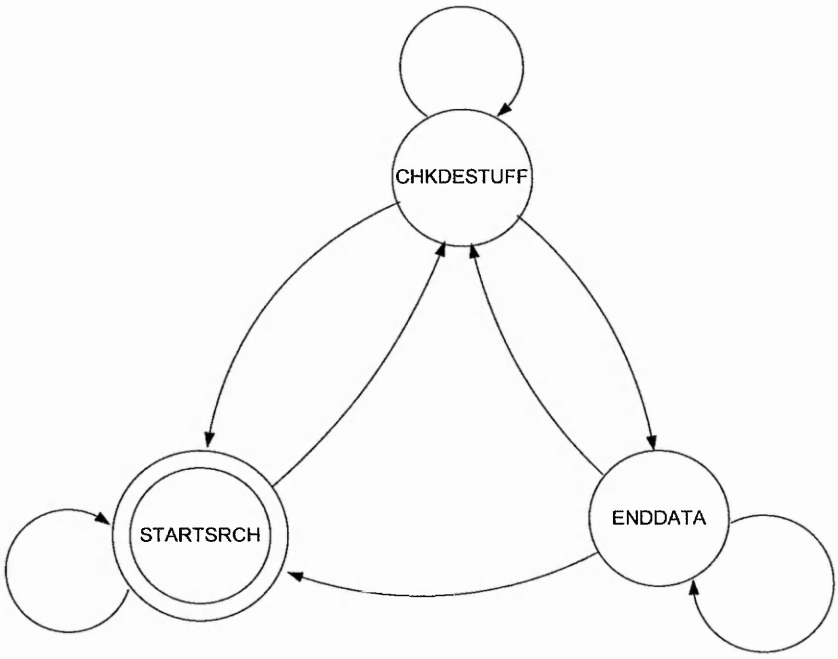


Figure 3-25: Optimised Byte Stuffing (OBS) Decoder top-level block diagram

In the MWS interfaces developed in this research, the OBS decoder is located at the ingress DVB-SPI interface, after the encoded payload is extracted from MPEG-TS frames. The OBS decoder will be presented with chunks of 184 bytes encoded data at a rate equivalent to the symbol rate of the demodulator. The EN input signal signifies to the OBS decoder that valid data is present on the 8-bit StreamIn bus. The OBS decoder decapsulates the MPEG-TS payload, performs de-stuffing, and sends the extracted Ethernet frame payload to an output buffer. The valid Ethernet frame is written into the output buffer by asserting the Write signal when valid Ethernet payload is output to the 8-bit StreamOut bus. At the heart of the OBS decoder is a state machine that actually performs the processing. The state machine diagram is shown the figure below:



Transition #	From State	To State	Condition
1	enddata	enddata	!EN
2	chkdestuff	enddata	EN&!max_reached&!streamIN[0]&streamIN[1]& streamIN[2]&streamIN[3]&streamIN[4]&streamIN[5]& streamIN[6]&streamIN[7]
3	enddata	chkdestuff	EN&streamIN[7]
4	chkdestuff	chkdestuff	!EN !max_reached&streamIN[0] !max_reached&!streamIN[1] !max_reached&!streamIN[2] !max_reached&!streamIN[3] !max_reached&!streamIN[4] !max_reached&!streamIN[5] !max_reached&!streamIN[6] !max_reached&!streamIN[7]
5	startsrch	chkdestuff	EN&streamIN[7]
6	enddata	startsrch	EN&!streamIN[7]
7	chkdestuff	startsrch	EN&max_reached
8	startsrch	startsrch	!streamIN[7] !EN

Figure 3-26: State machine Diagram and transition table of OBS Decoder core

The efficient OBS Decoder core was developed in this research which performs the decoding process using only 3 states of a synchronous state machine – allowing it to process the input stream at line rate. The OBS Decoder state machine initializes at the 'startsrch' state which begins scanning the interface for the occurrence of the Start_of_frame delimiter character. When the Start_of_frame delimiter is found, the state machine will switch to the 'chkdestuff' state and assume that all following bytes until the End_of_frame are payload of the Ethernet frame, in accordance with the algorithm (see 3.1.4). All bytes received hereon are copied onto the output bus, StreamOut, for further processing. However, if two consecutive End_of_frame delimiters are found within the data, the chkdestuff state will only copy one of them to the output bus, which effectively removes the stuff_bytes. The state machine will return to the 'strtsrch' state if only one End_of_frame Delimiter is detected which signifies a valid end of frame. During mid frame decoding of a Ethernet frame, the ingress stream might temporarily stop receiving valid MPEG-TS payload, for instance, during reception of Reed-Solomon headers or null MPEG-TS cells. During such periods the EN signal will be de-asserted and data on StreamIn bus will not be valid. Hence, the state machine constantly scans the EN signal and holds the current state whenever it is de-asserted. The hold function allows the

state_machine to remember its current state when it is disabled so that it can successfully resume where it had left off when EN is reasserted.

Besides performing the Byte Stuffing decoding function, the BS Decoder also performs a frame-size check so that the core will be resilient to errors. The max_reached signal is actually not part of the OBS algorithm but it is monitored by the state machine for the frame-size check function. This extra function is important because there is reasonable probability for End-of-frame delimiters to be in error or lost during transmission in the MWS investigated. Without an End-of-frame delimiter which usually tells the OBS decoder that the frame is completed, the OBS decoder will accept all bytes in the stream as part of the current frame. If the Frame-size check function is not implemented, the situation will likely lead to a drastic buffer-overflow event. The Frame-size check function is implemented as a counter which keeps count on the size of the frame being decoded. Should the frame size exceed Ethernets legal limit of 1518 bytes, the byte Stuffing Decoder automatically ceases decoding of the current frame and searches for the next one. See subsection 5.3.6 for experiments done to tests the workings of this core.

The architecture of the OBS decoder is described in VHDL which consists of multiplexers, a state machine, a resilient counter, output registers, and combinatorial logic. These structures can be seen in the RTL level implementation shown in Figure 3-27 below:

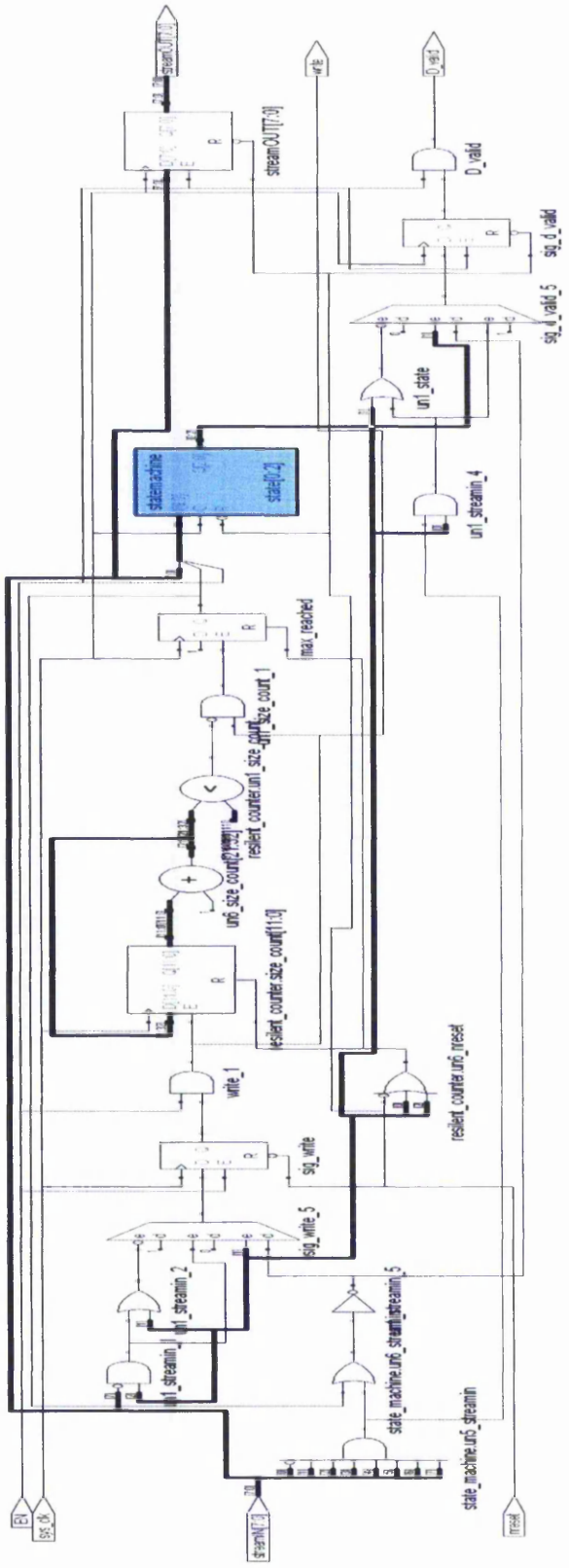


Figure 3-27: Possible RTL implementation of OBS Decoder core

3.3.4. OBS Encoder/Decoder Synthesis Results and Performance

The OBS encoder and decoder described in the previous two sections were implemented in FPGA hardware using a pipelined architecture that resulted in a solution that is low in complexity. In this subsection, synthesis and simulation results are detailed to determine FPGA resource usage, maximum operational performance and latency caused by the pipeline architecture.

Altera FPGA resource usages are usually measured in Logic Elements (LE) which are the smallest unit of logic blocks in Altera devices. Equivalent Application Specific Integrated Circuit (ASIC) gate count for one LE is roughly between 5.5 and 7.4 gates [133]. A LE consists of generic logic and a register that are designed to balance flexibility, efficiency and performance to implement digital circuits. The structure of LEs for Altera Cyclone or Stratix devices are shown below [134].

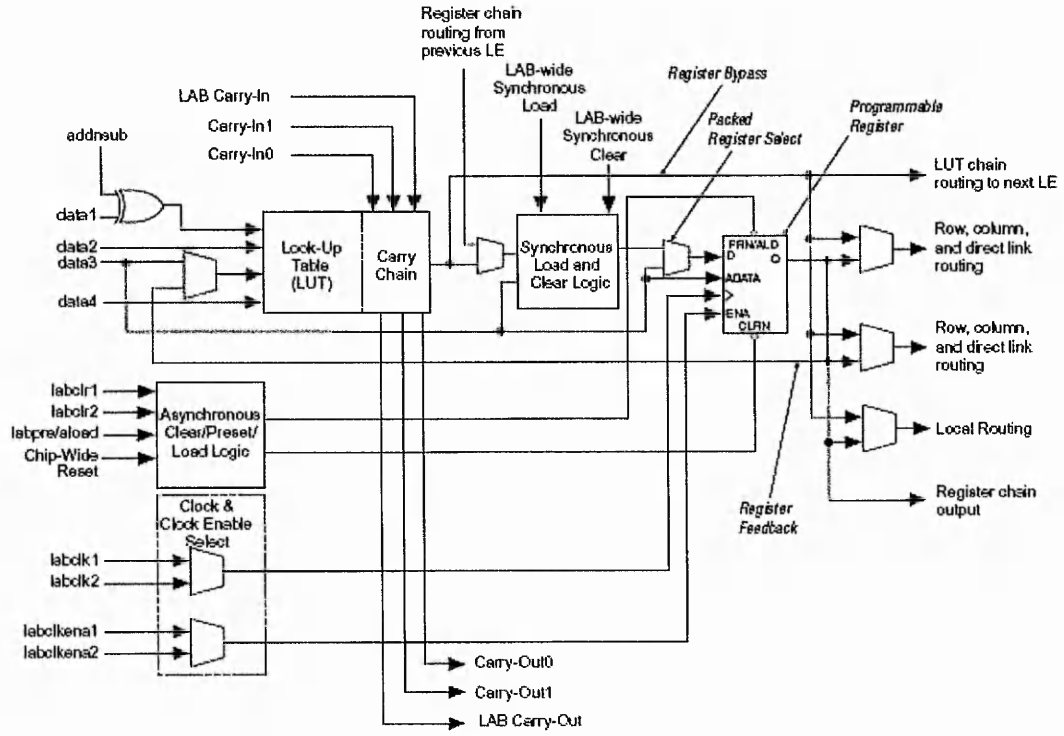


Figure 3-28: Structure of an Altera Cyclone Logic Element

When OBS cores perform satisfactorily in the test bench, they are targeted to a specific device in the FPGA synthesis and place & route design stages. During the synthesis process, the VHDL source codes are compiled down into register transfer logic (RTL) and then synthesized into FPGA LEs. The number of LEs required to implement the cores are reported by the synthesis tool at this stage. In the Place & Route stage, the LEs representing the cores are fitted into the target device, and the maximum operational frequencies are estimated. The design was targeted for several different Altera FPGA devices for comparison, and results are summarized in Table 3-3 below.

Core	Logic Elements	FPGA Device	Speed Grade	Maximum Operational Frequency	Maximum Ethernet Datarate
OBS Encoder	34	Cyclone EP1C3T100	-6	304 MHz	1.216 Gbps
			-8	281 MHz	1.124 Gbps
		Cyclone EP1C6Q240	-6	304 MHz	1.216 Gbps
			-8	261 MHz	1.044 Gbps
		Stratix EP1S10FC672	-6	210 MHz	840 Mbps
OBS Decoder	51	Cyclone EP1C3T100	-6	192 MHz	768 Mbps
			-8	140 MHz	560 Mbps
		Cyclone EP1C6Q240	-6	198 MHz	792 Mbps
			-8	166 MHz	664 Mbps
		Stratix EP1S10FC672	-6	203 MHz	812 Mbps

Table 3-3: Maximum operating frequencies for OBS cores

Whilst the general trade-off of using a pipeline architecture is usually increased latency, clock-cycle level examination of the simulation waveforms showed that the actual amount of pipeline latency introduced is very small, especially when the amount of encapsulation latency saved by the OBS algorithm is considered. To quantify pipeline latency, the results of the simulation process was used to compare the synchronous digital implementation of OBS, developed in this research, with theoretically ideal implementations that assume 1 Tb processing latency. Clock cycle comparison shows exactly 4 Tb (or 4 clock cycles) of latency is added in encoding including OBS preprocessing, and 2 Tb for decoding. Whilst an objective of this research was to achieve the theoretical limit for both encoding and decoding latency, the additional pipeline latencies are accepted as a generous trade off for low complexity and high speed implementation.

In addition, the thorough testing using VHDL assertions that ensured high quality intellectual property outcomes. The main purpose for testing the Byte Stuffing Encoder and Decoder cores was to eliminate bugs and verify its functionality. As these are particularly important cores in this research, it was run in a test bench for long periods using test vectors designed for a thorough stress test. To specifically stress test boundary cases of the cores, syntactic conditions were simulated. Each simulation run consisted of at least 5000 packets with minimum inter-frame gaps and varying sizes within the legal limit 64-1518 defined by the Ethernet standard. The simulation runs included the following boundary conditions:

- All payload bytes consisting of the Start_of_Frame Delimiter.
- All payload bytes consisting of the End_of_Frame Delimiter.
- All payload bytes consisting of alternating Start_of_Frame Delimiter and End_of_Frame Delimiter.

The OBS Encoder and Decoder cores were only qualified for hardware implementation after they have passed all simulation runs without firing any assertions.

Section 3.3 summary

Overall, the synthesis results of OBS as reported in the Quartus II tool showed the significant benefits of this solution, gauged in terms of low FPGA resource usage and high operational speeds when targeted to a variety of Altera devices. Primary factors that contributed to the benefits was the pipelined design of the cores as well as sufficient combinatorial logic termination using registers. In direct comparison to commercially available GFP FLM cores, OBS only consumes a fraction of the FPGA resources while still able to support slightly higher Ethernet data rates. Although, pipelining a Byte Stuffing algorithm doubles the clock cycles required to process a byte of input, the reduction in complexity had allowed it to operate roughly twice as fast. The comparison details of OBS with a FLM based algorithm are summarised in Table 3-4. The FLM cores compared are implementations of the GFP algorithm from Intec (GFP #1) [135] and Innocor (GFP #2) [136].

Core	Logic Elements	Device	Speed Grade	Maximum Operational Frequency	Ethernet Datarate
OBS	85	Stratix	-6	203 MHz	812 Mbps
GFP #1	3300	Stratix	-6	90.9 MHz	727 Mbps
GFP #2	7845	Stratix	-6	85.6 MHz	685 Mbps

Table 3-4: Comparison of OBS and GFP intellectual property cores

Chapter 3 Conclusions

This chapter presented the benefits of using Byte Stuffing (BS) algorithms for encapsulation of Ethernet frames to reduce latency and enable cut-through forwarding, in the Multimedia Wireless System (MWS) interfaces investigated. BS algorithms are generally perceived to have drawbacks of: a factor-of-two worst case overhead which can cause high service-level unpredictability and high implementation complexity and memory requirements which limits its use to low speed interfaces. However, the use of a new Optimised Byte Stuffing (OBS) algorithm and its innovative hardware implementation was shown to address these drawbacks while maintaining the advantageous characteristics of BS algorithms.

Acquisition of Ethernet traces from a live network, which retained byte level information of typical MWS traffic characteristics, enabled a comparison of several encapsulation algorithms. Based on the analysis of the acquired trace, actual unpredictability and jitter due to byte stuffing expansion was found to be very small. Nevertheless, the Optimised Byte Stuffing (OBS) algorithm, created in this research, was shown to reduce the number of frames incurring more than 8 bytes of expansion to 4.21% compared to 12.6% of Point to Point Protocol Byte Stuffing (PPP BS). Innovation in the structure (removal of input buffer) of the OBS implementation was shown to eliminate the need of high memory requisites in OBS receiver, while the pipeline hardware implementation allowed high operational speeds and low resource usage. Comparison of the OBS core with two commercial implementations of a Frame Length Marking cores reveal that OBS can be implemented at a fraction of the FPGA resource while maintaining support for high speed Ethernet data rates.

In conclusion, the significant improvement to encapsulation latency of BS, compared to the previously adopted methods in Broadband Fixed Wireless Access (BFWA) implementation, can indeed be realised and its drawbacks significantly reduced by applying the techniques detailed. But since the proposed OBS encapsulation Encoder and Decoder are just two of the many modules used to construct the Ethernet to DVB/MPEG-TS cores investigated in the next chapter, the high speed operation and pipelined structure has to be consistent in all subsequent modules of the MWS interfaces in order to maintain performance and operate at MWS bandwidths. From experience of previous BFWA systems, this is not a trivial task as the full potential of the surveyed systems was hindered by the internetworking interfaces performing encapsulation which caused throughput bottlenecks and large latencies despite lower bandwidth requirements. A fundamental architectural change is viewed to be necessary to scale interfaces for higher speed MWS interfaces. The benefits of using cut-through layer-2 switching devices to replace IP routers is already experienced in other LAN/WAN environments [137,138]. Hence, instead of complex store-and-forward IP routing implemented in previous systems, the MWS interfaces investigated in the next chapters can exploit layer-2 cut-through forwarding - allowed by the OBS cores - to achieve the target specifications.

4 ETHERNET TO DVB/MPEG-TS ENCODER AND DECODER FOR MWS

This chapter details the complete process towards a working demonstration of a MWS Subscriber Interface that consists of an 'Ethernet to DVB/MPEG-TS Encoder' and an 'Ethernet to DVB/MPEG-TS Decoder'. Beginning from the specification stage, the design, simulation, synthesis and finally hardware implementation of the MWS Subscriber Interface into FPGA development boards were done. By integration of the FPGA chip into DVB-S satellite equipment, two MWS Subscriber Nodes were formed for a working point-to-point microwave link demonstration. Through experiments on the test platform developed, realistic performance measurements were attained. As the MWS Subscriber Interfaces implemented the Optimised Byte Stuffing cores that were previously untried in existing DVB based systems, the experiments conducted revealed unique latency and throughput characteristics which were found to be superior to previous BFWA interfaces.

This Chapter is organised into three sections. In section 4.1, the latency budget and encoding rate are specified as design and synthesis targets. In section 4.2, a top-level of integration, the MWS Subscriber Interface that converges Ethernet and DVB frameworks is illustrated. This is followed by modular description of the MWS Subscriber Interface design. Section 4.3 describes the simulation models developed, which led to the characterisation of the MWS Subscriber Interface, and parametric optimisations for a specific implementation in view of an FPGA chip prototype. Finally, Section 4.3 details the construction of the test platform itself, and the experiments conducted using popular benchmarking software to obtain realistic performance results. The prototypes and tools developed here also aids further investigation into a novel point-to-multipoint MWS architecture detailed in Chapter 5.

4.1. Target Latency Budget and Operational Speed

For the target operational speed, the value of 56 Mbps encoding rate is chosen to match the maximum operational speed of the DVB-S demodulator equipment. The target latency budget of below 5 ms is chosen based on recommendations of Ethernet in the First Mile (EFM) networks since the MWS investigated offer very similar first mile Ethernet access services [139]. It is noted that quality of service between the two (or more) end hosts depends on the end-to-end latency - equal to the sum of delays contributed by each network in the path [140]. A tight latency budget is applicable since an MWS typically services the first/last few kilometres (up to 5 km) of the journey a packet takes across the Internet that geographically spans tens of thousands of kilometres to reach its destination. To maintain the end-to-end quality of service, a work by Angelopoulos et.al [141] on EFM technologies used a very stringent figure of "3 ms round-trip (or 1.5 ms one-way) delay budget — mandated by FSAN for demanding real-time services", referring to the recommendations of the FSAN/ITU-T (Full Services Access Network / International Telecommunication Union - Telecommunication Standardization Sector) standard [142]. Another work [143], focused on quality of service management at the base station for EFM, defined access network latency budgets of 5ms for constant bit rate (CBR), 15 ms for real time variable bit rate (VBR), and 100 ms for non-real time VBR services. To summarize, the target latency budget and operational speed is illustrated in Figure 4-1 below. Note that the MWS Subscriber node consists of a 'Ethernet to DVB/MPEG-TS Encoder/Decoder' pair as illustrated later in Figure 4-2.

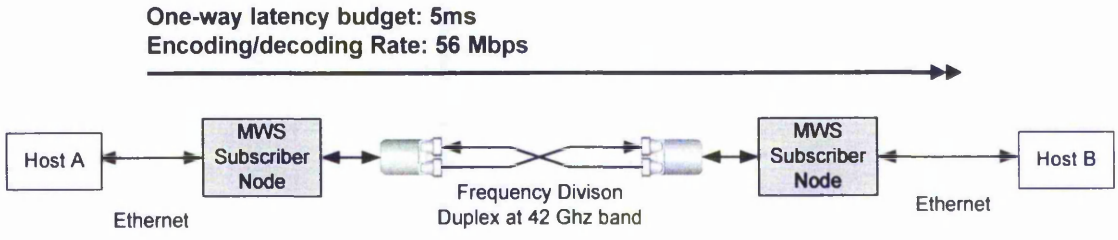


Figure 4-1: Latency and Operational speed targets for 'Ethernet to DVB/MPEG-TS Encoder/Decoder'

To achieve a high operational speed and low latency for 'Ethernet to DVB/MPEG-TS Encoder/Decoder' is non-trivial and often limits the full potential of a BFWA system as exemplified in the previous works surveyed (see section 2.2). In the CABSINET trial, "One major problem that affects TCP/IP performance in the implementation of the LMDS system is downlink delay. This delay [~50ms latency] is caused by head-end video server software performing the encapsulation of IP packets into MPEG frames" [90]. In the CRABS trial, an IP-DVB gateway Linux workstation was used to encode IP packets into DVB frames. Although the workstation had a 10Mbps Ethernet and 40Mbps MPEG-TS cards installed, the experiments conducted revealed: "The maximum throughput [encoding rate] is 6 Mbit/s which is the current limit of the IP-DVB gateway" [59]. The experiments conducted in CRABS also revealed one-way latencies between 45ms and 130 ms. Whilst the latencies reported in the previous BFWA systems were using 64 byte minimum sized frames, the 5 ms latency budget set for the 'Ethernet to DVB/MPEG-TS Encoder/Decoder' in this research has to be met for all frame sizes up to the 1500 byte maximum sized frames.

4.2. Design of MWS Subscriber Interface

This section focuses on the design and functional operation of the MWS Subscriber Interface. The first subsection (4.2.1) will provide a top-level overview of the MWS Subscriber Interface and its integration into the DVB and Ethernet frameworks. After this, the three major blocks of the MWS Subscriber Interfaces, which are the DVB/MPEG-TS Encoder, DVB/MPEG-TS Decoder, and Debug module, will be modularly described in detail in subsections 4.2.2 through 4.2.4. The last subsection, 4.2.5, concludes the section with resource usage results reported after synthesis, along with a brief description of experiences acquired from this stage of the design.

4.2.1. Integration into Ethernet and DVB framework

The MWS Subscriber interface converges with DVB and Ethernet at the Data Link layer through electrical interfaces as defined in the respective standards. Using standard interfaces allow integration to existing equipment and commercially available chipsets. For integration to the DVB framework, the DVB Synchronous-Parallel-Interface (DVB-SPI) was used. DVB-SPI was supported by the DVB modulator equipment and also

accessible on the printed circuit board inside the DVB-Satellite set top box which were available to this research. For integration to the Ethernet framework, the Media Independent Interface (MII) was used. All together, the MWS subscriber Interface hardware incorporates one DVB-SPI input, one DVB-SPI output and one bidirectional MII ports which totals 44 input/output signals. The block diagram below illustrates the integration of a MWS Subscriber Interface into the DVB and Ethernet frameworks to form an MWS Subscriber Node:

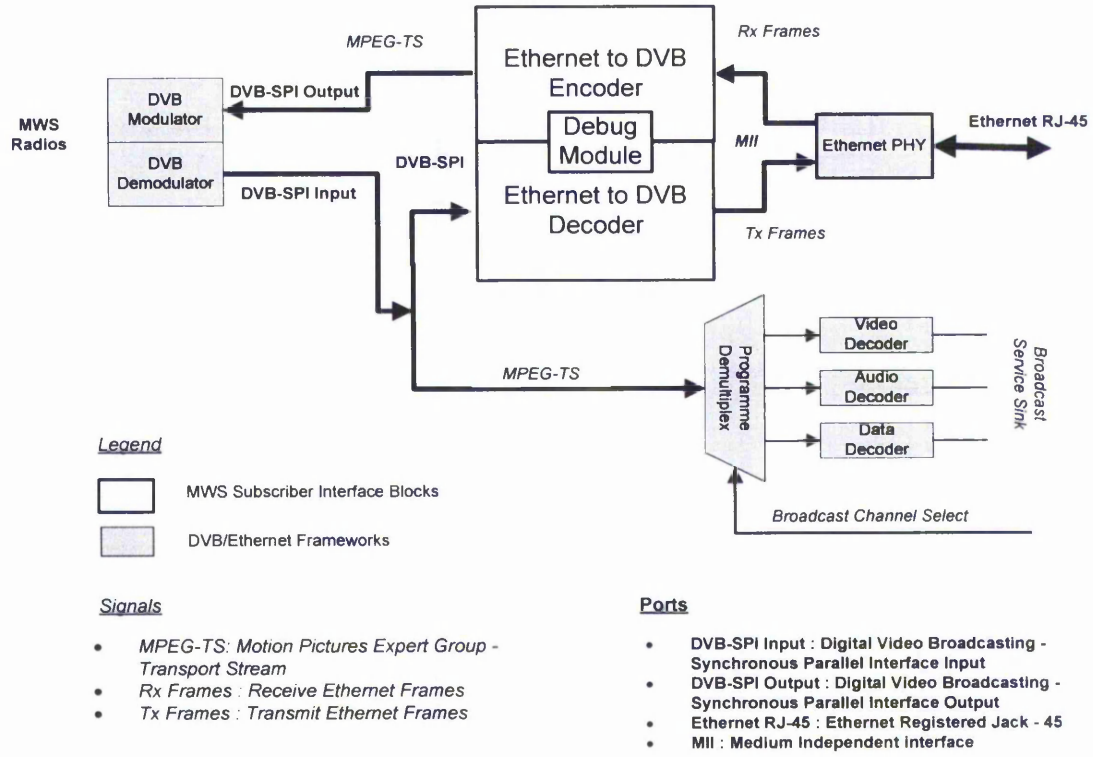


Figure 4-2: Block diagram of a MWS Subscriber Node

An advantage of this method of integration is that the DVB framework is left intact. This allows existing low cost DVB chipsets that are already integrated in DVB-S satellite boxes to be used to process the broadcast MPEG-TS services independently and simultaneously with the Ethernet service. "This is a potentially valuable feature, because in households the probability of concurrent TV and computer use is high" which "can justify a higher equipment price" [69].

The MPEG broadcast and Ethernet are received from the DVB-S set top box over the DVB-SPI input port in the form of a multiplexed MPEG-TS stream. The Ethernet frames that are received from the MII Interface (Rx Frames) are encoded into an MPEG-TS

stream which is designed to only occupy one PID channel. Use of one PID channel would not waste PID address space and hence is an appealing solution. For experimental purposes, the PID value of 0x1FFF was chosen which corresponds with test-stream PID assignment that is ignored by the DVB-S set top box. Likewise, the Receive Pipeline reads the MPEG-TS multiplex and only processes packets with the 0x1FFF PID value ignoring the MPEG-TS stream channels meant for MPEG broadcast service.

If a pair of MWS Subscriber Nodes is connected in a cross-over manner, a point-to-point microwave link can be formed without a MWS Base Station. A cross-over connection can be deployed by ensuring that the DVB Modulator equipment and 42GHz radios are tuned to the frequency of the receiver. This set up is shown in Figure 4-3:

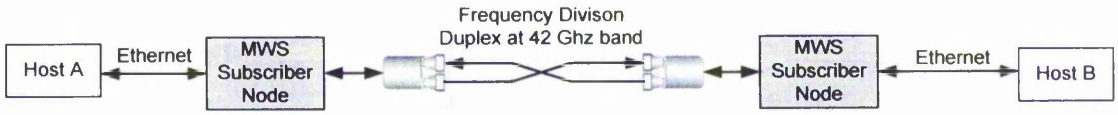


Figure 4-3: Point to point microwave link deployment

When viewed from a networking perspective, the Protocol Stack model can be used to put it into that context. An end-to-end path taken by user data from host A to B over the Ethernet networks and the point-to-point link are shown using this model (Figure 4-4).

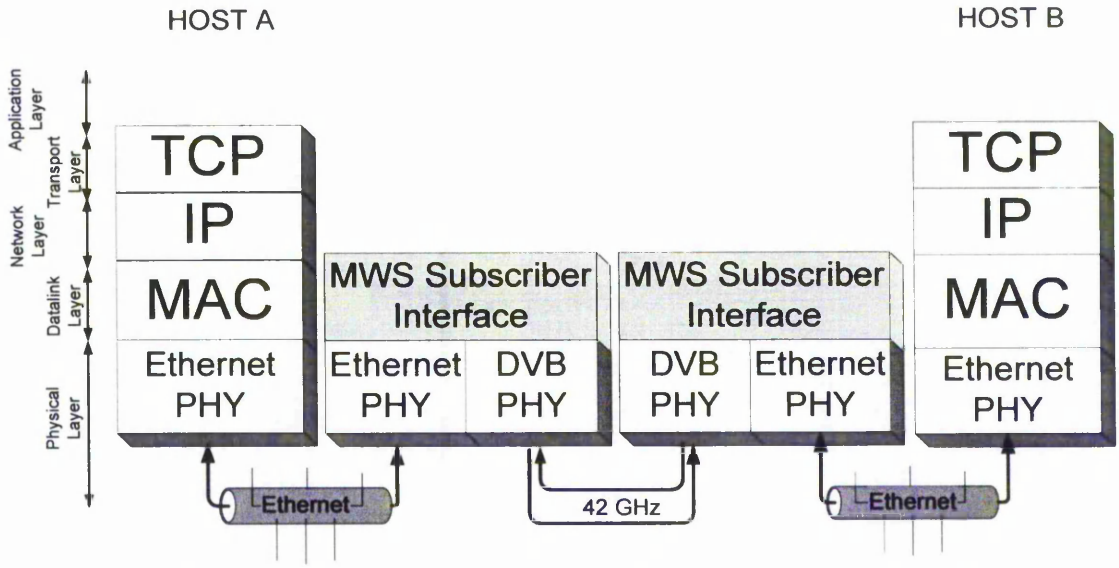


Figure 4-4: Network protocol stack model of MWS Subscriber Node

In the figure it is shown that the MWS Subscriber Nodes operate at the Datalink Layer where an Ethernet MAC is expected at after the Ethernet PHY. The Ethernet MAC of the MWS Subscriber Interfaces is somewhat simplified as it does not implement the CSMA/CD protocol usually present. This is because the MWS Subscriber Node operates only in full-duplex mode, which makes the CSMA/CD protocol required for half-duplex mode redundant. Using the Ethernet auto-negotiation process to advertise full-duplex only capabilities, the MWS subscriber nodes will prevent half-duplex connection on Ethernet from being established. With this configuration, the MWS Subscriber Nodes can be abstracted and treated as simple full-duplex transparent Ethernet bridges in an Ethernet networking perspective.

4.2.2. Ethernet to DVB/MPEG-TS Encoder Design

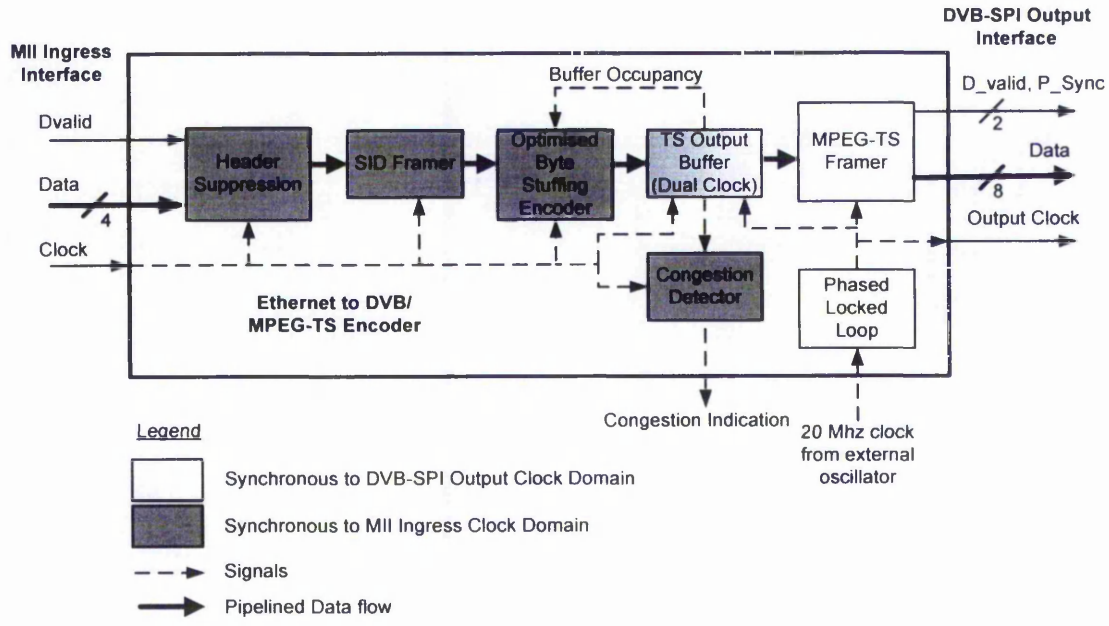


Figure 4-5: Ethernet to DVB/MPEG-TS Encoder block diagram

The 'Ethernet to DVB/MPEG-TS Encoder' receives Ethernet frames from an Ethernet PHY hardware via the MII interface, performs several stages of encoding, and outputs a constant stream of MPEG-TS frames to a DVB PHY hardware via the DVB-SPI Output interface (Figure 4-5). Each Ethernet frame received is processed in the sequence of: Header Suppression, SID framer, BS Encoder, SPI Output buffer and lastly MPEG-TS

framer. The stages of the encoding process are illustrated in the waveforms below (Figure 4-6).

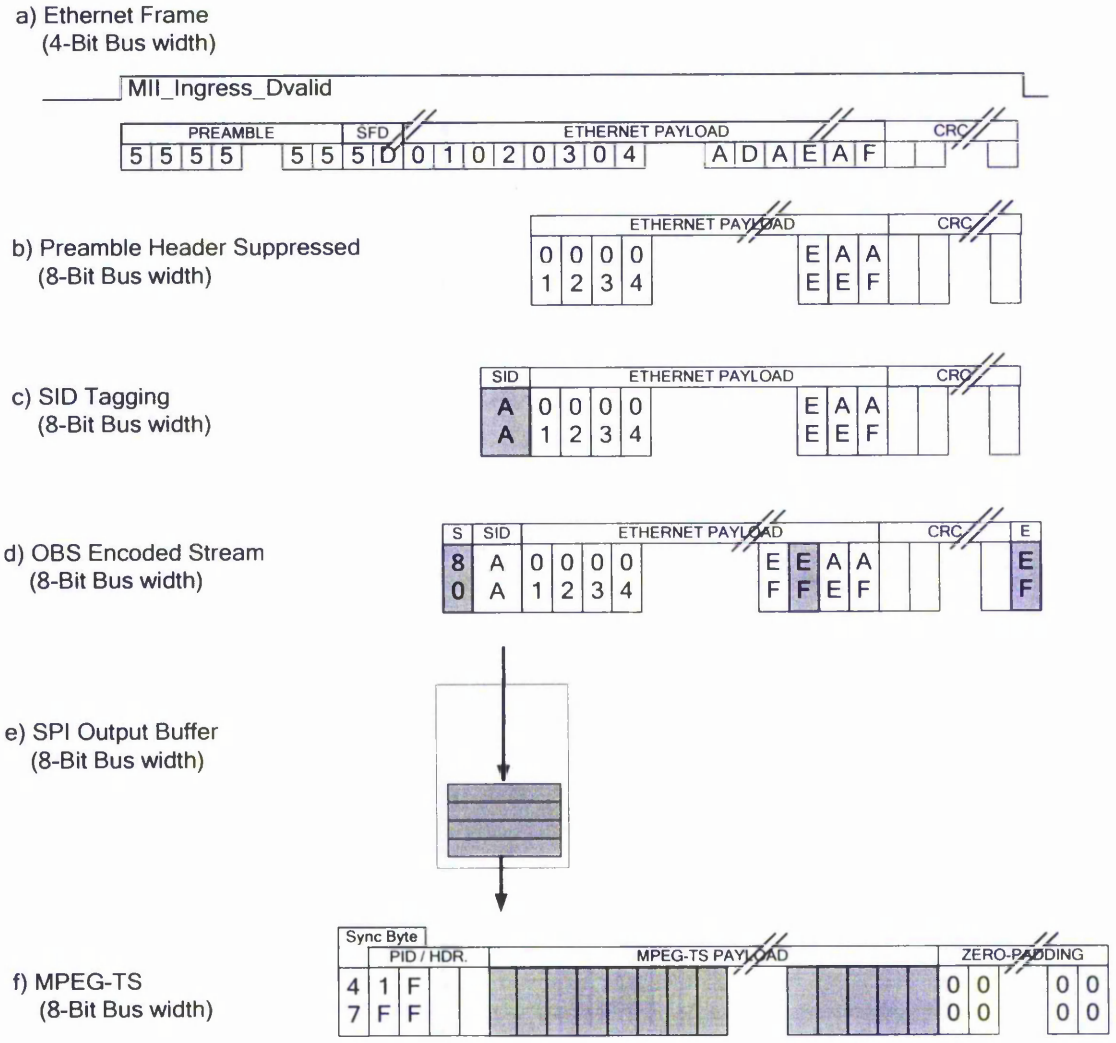


Figure 4-6: Encoding process waveforms

Module 3 Header Suppression:

The first stage of the encoding process is performed by the Header suppressor core which removes the Ethernet preamble and SFD fields. A straight forward implementation would use a counter to count to 14 so that the first 14 bytes (the size of a preamble) can be skipped. However, in real Ethernet networks, preamble shortening can occur in some internetworking devices [97] and hence, the preamble and start of frame delimiter (SFD) cannot be assumed to always span the first 14 bytes of each Ethernet frame. The header suppression core implemented here uses a state-machine to scan and verify each byte of the preamble sequence ('1010 1010') until an SFD sequence ('1010

1011') is detected. Note that the order of bits is always inverted in the Ethernet frame on the MII interface. For example in the waveforms in Figure 4-6, the SFD is shown as 0x5D ('0101 1101') which represents actual bit sequence of '1010 1011'. The output of the state-machine is AND-ed with the MII_Ingress_Dvalid signal to produce a Data valid signal that is asserted '1' only on the payload of the Ethernet frame hence telling the next module to ignore the preamble and SFD headers.

Module 4 Station Identity (SID) Tag:

After the header suppression stage, the SID Tag framer appends each Ethernet frame with a Station IDentity (SID) that is unique to each MWS Base Station or subscriber node. After the SID tag module, Ethernet frames are processed by the BS Encoder. The BS Encoder performs the encapsulation process as explained in Section 3.3.2.

The design of the SID Tag module uses a default generic parameter value which can be easily overwritten using an explicit SID assignment during component instantiation in simulation or synthesis. The SID value of a MWS Base Station or subscriber node has to be only assigned once at the top-level entity as the value is propagated to the SID Tag module and SID Filter Module (Module 9). The following paragraphs explain the method for default assignment, simulation assignment and synthesis assignment for the SID generic parameter.

The VHDL code below shows the design of the SID Tag module's entity declaration with an SID generic parameter. In the declaration, a default value of 0x00 is assigned in the case where the generic parameter is not explicitly assigned in simulation or synthesis. In VHDL, if a generic parameter is explicitly assigned during component instantiation, the default generic value is safely ignored.

```
...
entity SIDTag IS
     Generic (SID: std_logic_vector (7 downto 0):= "00000000");
     Port (...);
End entity
...
```

For simulation, SID parameter is assigned using Generic map in VHDL test bench source code. The source code below is part of the test-bench source which shows how

the SID is assigned with 0x01 value during top-level component instantiation of the subscriber interface.

```
...
UnitUnderTest : SubscriberInterface
          Generic Map ( SID => "00000001")
          Port map (...);
...
```

For Synthesis, Generic parameters declared at the top-level entity can be automatically extracted by the Leonardo Spectrum synthesis tool. The example below shows a SID is assigned with 0xAA by manually filling in the generic parameter value.

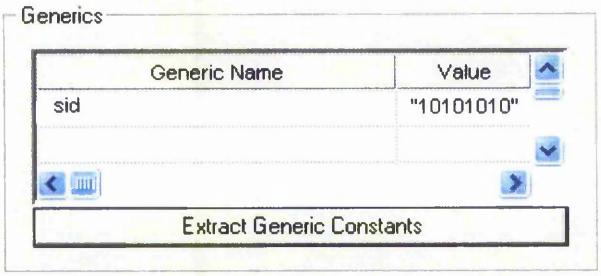


Figure 4-7: Assignment of generic parameter in Synthesis tool

Module 5 SPI Output Buffer and Congestion detector

In the 'Ethernet to DVB/MPEG Encoder' pipeline, the encoded stream output of the Optimised Byte Stuffing (OBS) encoder (see Module 1) is stored in the SPI Output Buffer. The SPI Output Buffer has another function besides providing elastic buffer storage; which is to allow data from one clock domain to be passed to another clock domain. In the 'Ethernet to DVB/MPEG Encoder' data from the Ethernet Ingress MII clock domain are passed to the DVB-SPI Output clock domain via the SPI Output Buffer. The SPI Output Buffer is implemented using the Altera DC_FIFO (Dual clock First in First Out) intellectual Property core [144] that can be synthesized free of charge on Altera devices. Altera DC_FIFO core is instantiated with a bus width setting of 8 bytes, matching that of the OBS encoder output. The depth setting, however, is determined after the optimal threshold value is obtained (see subsection 4.3.4). The depth setting should be 3000 bytes above the optimal threshold value to store a worst-

case expanded maximum-sized Ethernet frame. Status signals that indicate buffer occupancy as the number words stored in the buffer are also instantiated in the Altera DC_FIFO. Two sets of buffer occupancy status signals are instantiated; one synchronous to MII Ingress clock, and one synchronous to DVB-SPI output clock. MII Ingress clock is obtained from the external Ethernet MII Ingress port and is generated by the Ethernet PHY. The status signals synchronous to the MII Ingress clock are used by the OBS encoder (for flushing mechanisms), and also the Congestion Detector to assert a congestion event to the 'Ethernet to DVB/MPEG-TS Decoder' core via the one pin congestion indication signal. The Congestion detector is implemented as combinatorial logic which assert logic '1' whenever the buffer occupancy is greater than the optimal threshold value. Status signals synchronous to DVB-SPI output clock are used by the MPEG-TS Framer as described in the following paragraph.

Module 6 MPEG-TS Framer

The MPEG-TS framer reads the encoded stream from the SPI Output Buffer and appends a Sync_byte and 3 bytes of headers including the MPEG-TS channel PID of 1FFF, for every 184 bytes of data according to the DVB standard (See subsection 2.2.1). The Sync Byte is used for synchronisation purposes within the DVB framework (see Figure 2-6). At the core of the MPEG-TS Framer is a state-machine. The state-machine will check the occupancy of the SPI output buffer and will begin to immediately construct an MPEG-TS cell if there are six or more bytes buffered for true cut-through forwarding. The value of six bytes (instead of one) was used because the dual-clock nature of the SPI Output buffer require complicated synchronisation circuitry which causes "the rdusedw [read buffer occupancy] signal to indicate that the FIFO is emptier than it actually is" and "the wrusedw [write buffer occupancy] signal to indicate that the FIFO is fuller than it actually is" [144]. The impact to performance is six bytes of latency and up to six bytes of additional flushing per burst of Ethernet frames. P_sync, and D_valid signals are also generated by the state-machine to form the complete MPEG-TS stream which is transmitted via DVB-SPI output port to be modulated.

The MPEG-TS Framer core developed implements two configurable features: i) Zero-padding can be turned on or off, ii) MPEG-TS symbol rates can be derived from the internal PLL or from an external clock (e.g. feedback clock from a DVB-S modulator). When the core is configured with Zero-Padding option turned 'on', sixteen h00 value

padding-bytes are padded to the MPEG-TS stream according to the DVB standard producing an output known as the 204 byte MPEG-TS stream [80]. This format is required for interfacing with some Reed-Solomon forward error correction cores [145] which may be employed by some DVB modulators [146]. The external clock configuration lets the DVB-S modulator determine the symbol rate of the MPEG-TS framer.

Module 7 Phased Lock Loop

Configurability of MPEG-TS symbol rates are implemented using a combination of an analogue Phase Locked Loop (PLL) and a digital Clock divider. Analogue PLL devices are embedded in the Altera Cyclone FPGA chips and are used by component declaration and configuration in VHDL code. Configuration of the analogue PLL device include the clock multiplier variable 'x' and clock divisor variable 'y' which can produce several frequency options from a reference crystal. As the values of x and y are somewhat restricted [134], a digital clock divider was developed to provide another stage of division 'z' for even more frequency options. Hence the symbol rate of the MPEG-TS output stream, 'r', is determined by the function:

$$r = 20MHz \times \frac{x}{y.z}$$

A 20MHz crystal is available on the FPGA development Cyclone board is connected on a dedicated clock pin to the on-chip analogue PLL. For the 56Mbps 'Ethernet to DVB/MPEG-TS Encoder', MPEG-TS Stream symbol rate of 7Mhz ($\frac{56Mbps}{8bits}$) is needed from the 20MHz clock. This is achieved by setting the PLL variables to $x = 7, y = 10$ and digital clock divider to $z = 2$. Similarly, for a 14Mbps 'Ethernet to DVB/MPEG-TS Encoder' used in an investigation presented in section 5.3, the settings of $x = 7, y = 10$ *and* $z = 8$ produces the 1.75 MHz clock required.

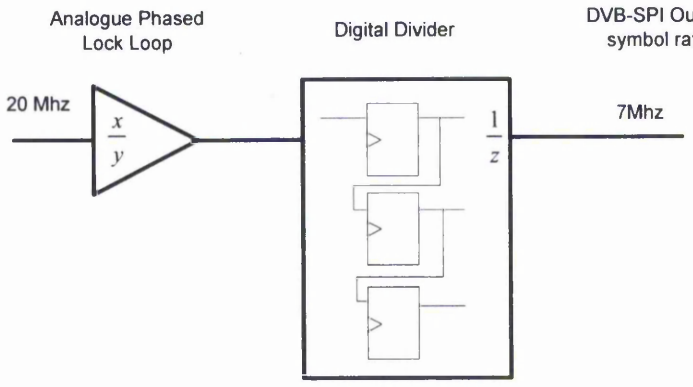


Figure 4-8: MPEG-TS symbol rate generator Schematic Diagram

4.2.3. Ethernet to DVB/MPEG-TS Decoder Design

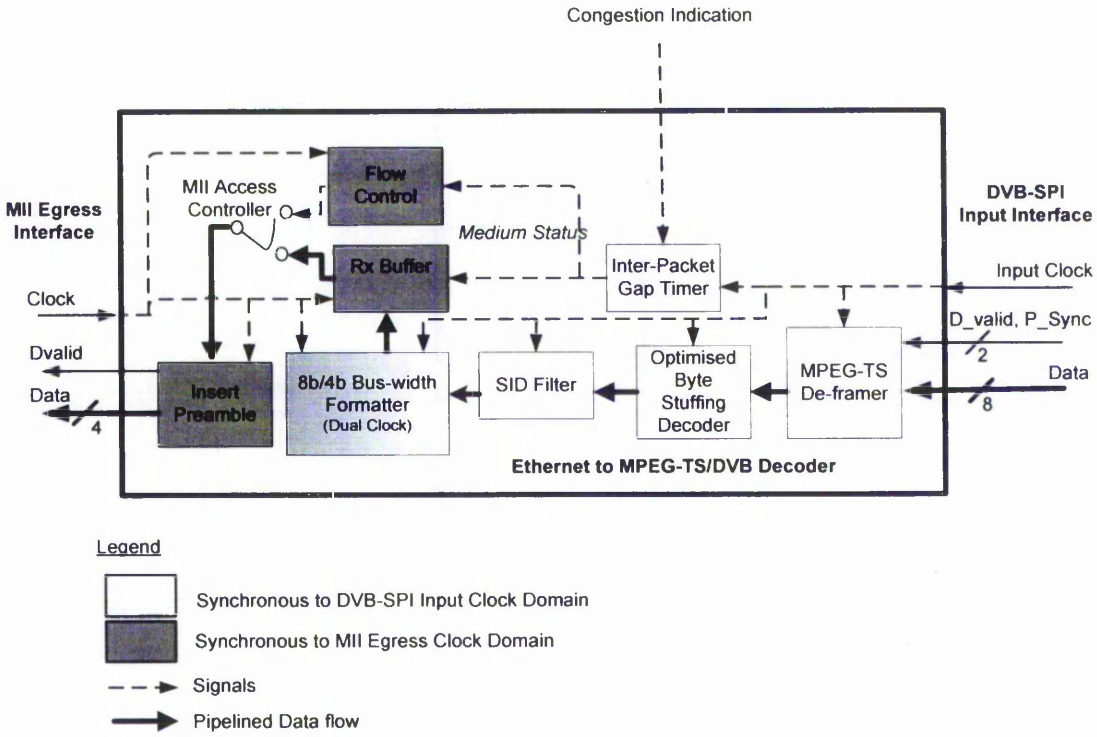


Figure 4-9: 'Ethernet to DVB/MPEG-TS Decoder' Block Diagram

The 'Ethernet to DVB/MPEG-TS Decoder' is responsible for two major functions. The first function is to read MPEG-TS cells from the DVB-SPI input interface, perform several stages of decoding to recover the original Ethernet frames and then output them to the egress MII interface. The second function is to monitor the SPI Output Buffer

occupancy of the paired 'Ethernet to DVB/MPEG-TS Encoder' and perform local Ethernet flow-control to prevent loss of packet due to buffer overflow.

For the first function, the modules that form the Receive Pipeline in the order of decoding stages are: MPEG-TS De-framer, BS Decoder, SID Filter, 8b/4b Bus-width Formatter, Ethernet Egress Buffer and finally Ethernet Framer. The stages of the decoding procedure are sequentially shown in the waveforms below:

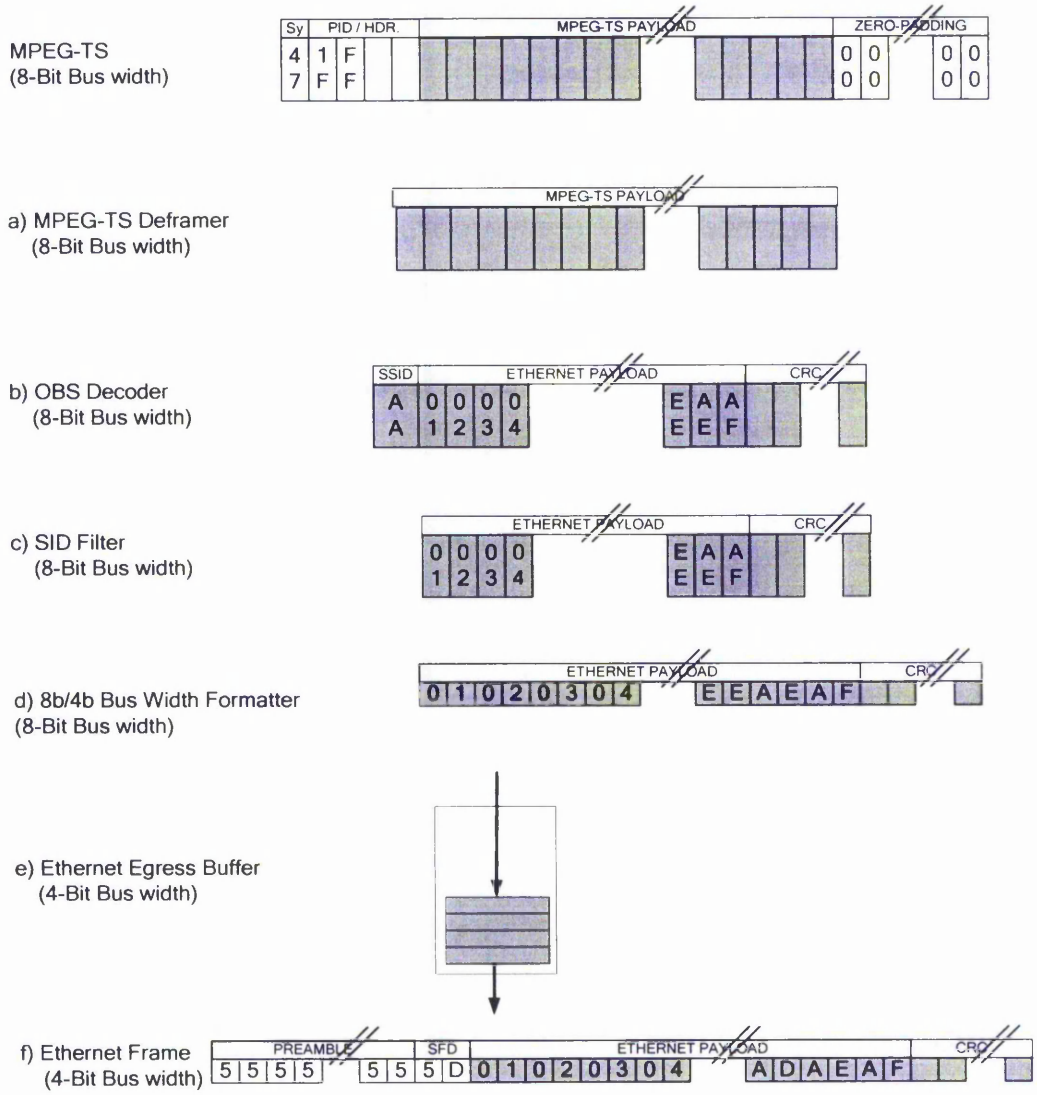


Figure 4-10: Decoding Process Waveform

Module 8 MPEG-TS Deframer

The MPEG-TS defamer synchronizes with the MPEG-TS stream by scanning for a valid P_sync pulse on the DVB-SPI interface. When the state-machine detects the P_sync pulse, it applies a PID filtering rule where only MPEG-TS cells with the PID 0x1FFF are passed. The PID can be configured to be any value but 0x1FFF was chosen for Ethernet services in a MPEG-TS multiplex as explained in section 4.2.1. The MPEG-TS De-framer uses a counter to count from 1 to 188 to generate a signal to obtain the first 188 bytes after a P_sync pulse. Hence, the SPI_Dvalid signal on the DVB-SPI port is not used, but the core still supports automatic detection of 188 or 204 byte MPEG-TS format, which means that either format will be parsed without explicit reconfiguration of the core. The output payload bytes from the MPEG-TS De-framer is passed to the Byte Stuffing Decoder core, as explained in Section 3.3.3, which reconstructs Ethernet frames (payload portion). In the 'Ethernet to DVB/MPEG-TS Decoder', the Ethernet frames from the Byte Stuffing Decoder core are passed to the SID Filter for further processing.

Module 9 SID Filter

The SID Filter module processes each supplied Ethernet frame input to it that is tagged with an SID address. A filtering rule is applied on the SID address so Ethernet frames matching a the 'Ethernet to DVB/MPEG-TS Decoder' SID are discarded. The SID value used by the filter is configured via a top-level VHDL Generic as explained in Module 4. The purpose of the SID Filter is so that Ethernet frames originating from the local Ethernet network, and looped back by the remote MWS base station (see subsection 5.1.2), are not allowed to pass through. In other words, only frames originating from remote Ethernet networks are forwarded to the 8b/4b Bus-width Formatter. Note that SID addressing have no significance in a point-to-point link MWS (see subsection 4.4.1), as loop-back does not occur in that deployment.

Module 10 8b/4b Bus-width Formatter

The 8b/4b Bus-width formatter converts Ethernet frames output from the SID filter in 8-bit bus format to 4-bit bus format. Since the clock rate has to be increased during 8-bit to 4-bit conversion, a dual-clock FIFO with 8-bit width is instantiated to handle processing of data from one clock domain to the other. The FIFO output side is synchronous to the DVB-SPI input clock whilst the input side is synchronous to Ethernet MII Egress port. Since the effective bandwidth of the output bus (100 Mbps) is much greater than the

input bus (up to 56 Mbps) elastic buffering was not required for the purpose of bus width conversion. Hence, the FIFO depth was reduced to 256 words to save memory resource usage. A state machine and a latch is implemented to control the FIFO and condition the FIFO output signals so that the Ethernet data-bus and Ethernet data-valid appears in 4-bit format. In the 'Ethernet to DVB/MPEG-TS Decoder', the output of the '8b/4b Bus-width converter' module is written into the 'Ethernet Egress buffer' module for further processing.

Module 11 Ethernet Egress Buffer

Since the Ethernet bit rate (100 Mbps) is greater than the DVB-SPI input bit rate (up to 56 Mbps) the Ethernet Egress Buffer is needed to ensure that an entire Ethernet frame is stored before it is forwarded to the local network. The Altera SingleClock_FIFO intellectual property core was used to instantiate a 4-bit buffer. In order to facilitate store-and-forward operation, additional logic was required to condition the word-wise buffer occupancy signals to frame-wise capacity indicator. Hence, edge detectors and a frame counter were designed around the Altera core to generate the appropriate signals. The frames are held in the Ethernet Output Buffer until 'MII Access Controller' module (see Module 12) grants access to release Ethernet Frames on to the local Ethernet network via the MII egress interface. Since the 'Ethernet Egress Buffer' bus-width matches the MII Egress bus-width, frames stored in it can be directly output to the local network once access is granted.

Module 12 Flow Control Module

The second function of the 'Ethernet to DVB/MPEG Decoder' is to perform local Ethernet flow control for the paired 'Ethernet to DVB/MPEG Encoder' in a MWS subscribe interface. The Flow Control module implements a state machine to request access from the MII Access controller whenever a congestion condition signal from the 'Ethernet to DVB/MPEG Encoder' is asserted. Once the MII Access controller grants access, the state-machine will copy an image of a flow-control frame, which is stored in a ROM, to the MII Egress port. When the remote Ethernet MAC receives the flow control frame it will temporarily pause Ethernet traffic towards the 'Ethernet to DVB/MPEG Encoder'. VHDL source code was developed to generate a valid Ethernet Pause_frame for the ROM image (see Figure 2-25), including pause quanta and Cyclic Redundancy Check (CRC) fields. The pause quanta can be configured to have any pause

value according to the Ethernet standard. The flow control function can also be configured to be disabled (used for experimental purposes).

Module 13 MII Access Controller

MII Access Controller grants access to either the Output Buffer or the Flow Control Module based on a simple decision algorithm. Basically, the decision algorithm treats flow control frames as higher priority and hence it will gain access to the MII egress interface unless it is already being used by the output buffer. It is considered that since flow control frames are only 64-bytes long, the delay caused to the Ethernet Frames waiting in the Ethernet Egress Buffer is minimal. The MII Access Controller also implements an inter frame gap counter which counts to 24 after each Ethernet frame is transmitted to the MII Egress port. During the inter-frame-gap, neither the Ethernet Output Buffer nor the Flow Control module will have access to the MII interface. This process maintains the minimum frame-gap between two Ethernet Frames according to the Ethernet standard.

Module 14 Preamble Generator

The Preamble Generator module generates the preamble headers on-the-fly for each Ethernet frame received from the MII access controller and outputs to the MII Egress interface. A state-machine is implemented to iterate between Preamble_state and SFD_state for each Ethernet frame received. During the preamble generation state, a counter is used to count to thirteen to generate thirteen bytes of Preamble sequence. When the counter reaches thirteen, the state-machine will transition to the SFD_state and the one-byte SFD sequence is generated. After the SFD_state, the state-machine and preamble counter is reset, and is ready to accept the next Ethernet frame from the MII access controller.

4.2.4. Debug Port and Status Indicators

The Status/Debug module is implemented to visually indicate the state of the Subscriber Interface via an array of 10 LEDs implemented on the prototype hardware. The debugging signal assignments to the LED array are shown in the schematic diagram below (Figure 4-11):

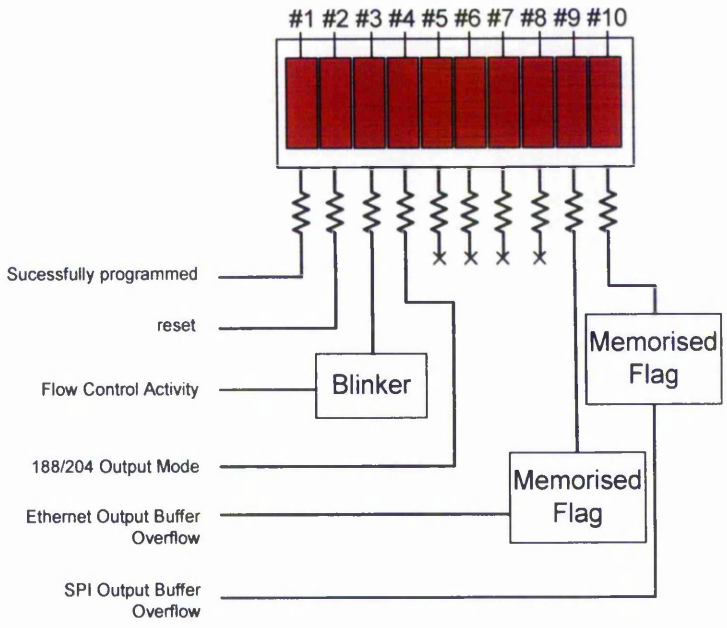


Figure 4-11: Schematic Diagram of Debug Module

LED 1 lights up to indicate that the FPGA device is successfully programmed via the JTAG port. LED 2 is connected to the reset button which lights up in normal mode and switches off when the button is depressed.

LED 3 blinks to indicate that the subscriber interface is performing flow control on the MII interface. As the actual duration of the flow control Ethernet frames are invisible to the human eye, the debug module processes the signal to a more course granularity to drive the LED. The process involves an edge detection of Ethernet Pause Frame and a large counter to ensure the minimum "on"/"off" or blink cycle to have a minimum T_v of : $\frac{1}{25Hz} = 40ms$. The waveform of the LED Blinker driver shown in Figure 4-12 below:

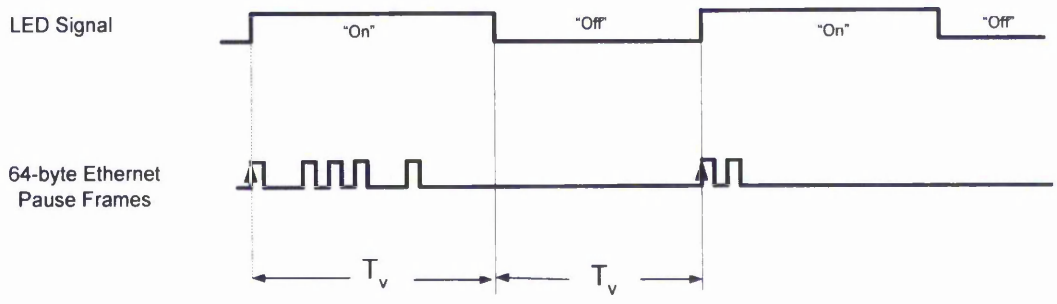


Figure 4-12: Blinking LED Waveform

LED 4 indicates a the current configuration of MPEG-TS stream as either 188 mode by turning "on" the LED; or 204 mode with LED turned "off". LED 5 to 8 are not used in the design and hence are reserved for future use.

LED 9 and 10 are turned on and stay on once a buffer overflow is detected. The LED can only be switched "off" by performing a hardware wide reset by depressing the reset button. The buffer overflow events are considered critical and hence generate memorized flags so that it would not be missed.

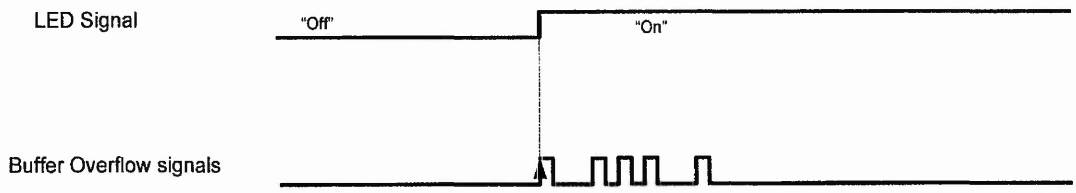


Figure 4-13: Memorised Flag LED waveform

4.2.5. Synthesis process and Resource Usage

This subsection details the synthesis process of the MWS Subscriber interface and the experiences of the author using the Leonardo Spectrum synthesis tool, leading up to a summary of the resource usage reported after place-&-route onto a Cyclone FPGA device. The synthesis process used is largely automated by the EDA tools where a designer is only involved in choosing the synthesis options. However, when the synthesis produces unsatisfactory results, the designer can attempt to change the synthesis options on a trial-and-error manner in hope that the synthesis tool would produce better results.

Limitations experienced in using Altera-Quartus II ver.2.2 tool are; i) a design source code written in VHDL is allowed to have only one architecture defined per entity, and ii) VHDL configuration files are not supported. In order to take advantage of multiple architectures per entity, a third-party synthesis tool was required. A trial version of Leonardo Spectrum synthesis tool was used in this research and was found to work well

with Quartus II. One advantage of multiple architectures is that new designs can be tried out while keeping previous designs intact, making revision tracking simpler. Another advantage is that an entity can be more flexible and serve different functions in the same project by selecting the appropriate architecture for each function, for example, see Module 17. A 'VHDL configuration' file (special purpose file defined in the VHDL standard) was coded for each FPGA. Using features of the VHDL configuration file, the entire hierarchy of the MWS Subscriber Interface is described by defining architecture/entity pair bindings. A configuration file is interpreted by the synthesis tool to synthesize a specific project implementation and targeted to a specific FPGA device. It is noted that Altera intellectual property cores were treated as 'black boxes' in Leonardo spectrum and are only synthesized in Quartus II. This was probably necessary to protect the Altera intellectual properties.

Besides providing support for 'multiple architectures' and 'VHDL configuration' features, Leonardo Spectrum also allowed various synthesis parameters to be defined so that a designer can influence the synthesis process. The customizable parameters used in the synthesis of the MWS Subscriber interface are: required clock rates, area/speed trade-off setting, FPGA device assignment and pin assignments. Once all parameters were set in Leonardo Spectrum, the 'place & route' process can be automated from Leonardo Spectrum by selecting 'compile in Quartus' option. In essence, Leonardo Spectrum generates a Verilog Quartus Mapping (.vqm) file from the VHDL design source codes that can be interpreted by Quartus II. Leonardo Spectrum communicates with Quartus II via Native Link and a scripting called tcl (tool command language), forming an integrated EDA environment. Quartus II will perform the 'place & route' process and generate a '.sof' file which is used to programme the target FPGA device.

Sometimes, Quartus II would fail to find a 'fit' for the design in the target FPGA device, due to clock skew. The first approach when encountered with a failed 'fit' was to change some parameters in Leonardo Spectrum and rerun the synthesis and place & route process. If this did not work, the design flow would iterate back to the VHDL source code design process (see subsection 3.3.1). It is possible to use the 'floor-plan editor', in Quartus II, to manually move the logic cells about in attempt to eliminate clock skew. However, this was found to be very time consuming and not effective. The 'required clock' settings (synthesized speed) set in the synthesis tool that achieved a successful fit,

and their respective post-fit slack times are shown in Table 4-1. FPGA device details and resource usage for the MWS subscriber interface are summarised in Table 4-2. Finally a detailed breakdown of the logic element usage for each of the design modules are illustrated in Table 4-3.

Clocks	Required Speed	Subscriber Interface		Maximum Encoding Rate‡ or Maximum Ethernet Rate†
		Synthesized Speed	Slack time	
Eth. MII Ingress	25 MHz	25 MHz	28.711 ns	100 Mbps†
Eth. MII Egress	25 MHz	25 MHz	14.189 ns	100 Mbps†
Reference Clock	20 MHz	20 MHz	45.645 ns	
Internal DVB-SPI Out	7 MHz	25 MHz	65.998 ns	200 Mbps‡
External DVB-SPI Out*	7 MHz	25 MHz	15.626 ns	200 Mbps‡
DVB-SPI In	7 MHz	10 MHz	33.101 ns	80 Mbps‡

*MPEG-TS Framer configured to use external clock

Table 4-1: Synthesized speed and encoding/decoding rates of MWS Subscriber interface

MWS Subscriber Interface FPGA	Resource Usage / Resource Capacity
Device	EP1C6Q240C6
Speed Grade	-6
Logic Element Utilization	829 / 5,980 (13%)
Memory Bits	57,088 / 92,160 (61%)
PLL (analogue)	1 / 2 (50%)
Pins	80 / 185 (43%)

Table 4-2: FPGA details and resource usage of MWS subscriber interface

Module Name	Logic Elements	RAM bits	Mod.No.
Ethernet to DVB/MPEG-TS Encoder	**328**	**32,768**	
DVB-SPI output Buffer	222	32,768	5
MPEG-TS framer	44	0	6
Byte Stuffing Encoder	28	0	1
Header Suppression	15	0	3
SID Tag	11	0	4
Phase Locked Loop	4	0	7
Congestion Detector	4	0	6
Ethernet to DVB/MPEG-TS Decoder	**447**	**24,320**	
8bit-4bit converter	188	2,560	10
Ethernet Egress Buffer	90	20,480	11
OBS Decoder	46	0	2
Flow Control	45	1,280	12
Preamble Generator	20	0	14
MPEG-TS Deframer	20	0	8
SID Filter	18	0	9
MII Access Controller	15	0	13
Glue Logic	5	0	
Debug Module	**54**	**0**	
MWS Subscriber Interface	**829**	**57,088**	

Table 4-3: FPGA Logic Element and RAM usage of MWS Subscriber interface

Summary of Section 4.2

This section described the VHDL design and synthesis of the 'Ethernet to DVB/MPEG-TS Encoder' and 'Ethernet to DVB/MPEG-TS Decoder' which forms the MWS subscriber interface. The major design objective to achieve an encoding rate of at least 56 Mbps was exceeded with post-synthesis encoding rate of 200Mbps when targeted to an EP1C6240 Altera FPGA device. The design included an in-build hardware debug module which would aid in cross verification and hardware prototype debugging (see later sections 4.3 and 4.4). An integrated Electronic Design Automation (EDA) environment was developed using ModelSim, Leonardo Spectrum and Quartus II tools. Proper VHDL designs coding style and optimisations done in the EDA environment allowed the MWS Subscriber interface to be synthesized in low-cost Cyclone FPGA that exceeded initial specifications, and within the time limits of the project. The use of a low-cost FPGA were in line with the requirements of a cost effective subscriber interface for further investigation into a wide spread trial. The resulting source codes were also of high-quality that integrated configurable parameters for flexible instantiations of MWS subscriber interfaces. The configurable parameters of the MWS subscriber interface

design are listed in Table 4-4, specifying the module and VHDL generic or variable used.

Module	Description	VHDL Generic/Variable	Values/Architecture
OBS Encoder	OBS End of Frame delimiter value	***endVal***	0x00 to 0xFF
OBS Decoder	OBS End of Frame delimiter value	***endVal***	0x00 to 0xFF
Ethernet to DVB Encoder/Decoder	Station Identity (SID) Address	***SID***	0x00 to 0xFF
Flow Control	Pause Quanta	***PauseTime***	0 to 65535
MPEG-TS Framer	MPEG-TS output encoding clock source	***Clock_Source***	"Internal" or "External"
MPEG-TS Framer	MPEG-TS output format	***Stream_Format***	"188" or "204"
MPEG-TS Deframer	MPEG-TS input format	-	auto selected
Congestion Detector	Congestion threshold	***Congestion_threshold***	0 to 4096
PLL	Clock rate for internal MPEG-TS output clock	***x, y, z***	up to 640 MHz*

*synthesized rate up to 25MHz

Table 4-4: Configurable parameters for Ethernet to DVB/MPEG-TS Encoder/Decoder

4.3. Simulation and Optimisation of MWS Subscriber Interface

This section details the work carried out in a simulation environment for verification, analysis and optimisation of the MWS Subscriber interface cores developed. The first subsection 4.3.1 details the VHDL test-bench behavioural models developed for the simulation platform. Subsection 4.3.2 describes how the MWS Subscriber Interface core was thoroughly tested to achieve a high quality intellectual property source code. The remaining subsections details the adaptive characteristics and optimisations for buffer threshold settings of the 'MWS Subscriber interface'. The optimisations are done for two specific operational speeds. The first implementation is optimised and simulated at 56Mbps 'Ethernet to DVB/MPEG Encoder' in view of constructing a broadband point-to-point microwave link as described in section 4.4. The second implementation was optimised and simulated at 14 Mbps in view of constructing subscriber nodes for the Ethernet Hub Emulation MWS prototype as described in section 5.3.

4.3.1. The Test-bench for Subscriber Interface Simulation

Several behavioural VHDL test-bench cores were developed in this research to construct the simulation platform. The first objective was to achieve a system that could facilitate automated simulation and thorough verification of the MWS Subscriber Interface core. The second objective was to enable statistics collection for further analysis using Matlab.

The VHDL behavioural cores used in the virtual test rig set-up are: Ethernet frame generator, DVB Physical layer Emulator, a behavioural PLL and an Ethernet Frame verifier. These cores are interfaced with the Subscriber interface to create a loop-back interface for verification as shown in Figure 4-14.

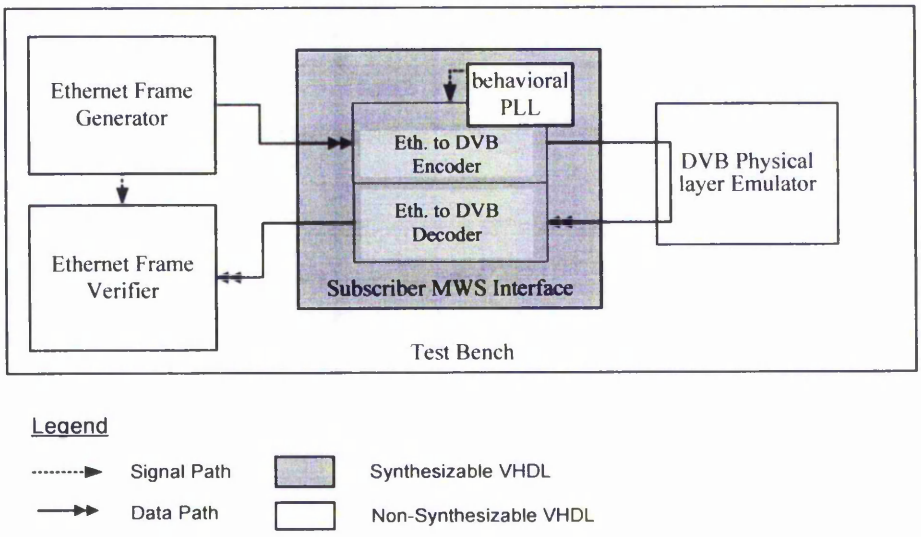


Figure 4-14: Subscriber Interface Test-bench Block Diagram

The testing strategy chosen was to use black box testing [147]; as it suited the research for several reasons which include:

- The generic nature can produce test cases that may not be thought of using white box testing [147].
- Able to use the same test vectors and assertions for individual cores in the design.
- Easier to code test vectors than white box which results in less lines of source code and faster simulation run time.

- Verification can be done automatically using assertions which are usually very tedious in white box testing.

The Ethernet Frame Generator behavioural module was developed to generate user defined test vectors. Interfaced via the MWS Subscriber interface's MII port, the generated Ethernet frame vectors are injected. This core also facilitates a disable function which will halt the generation process as long as the disable signal is triggered. The disable function is primarily used by the Ethernet Frame Monitor module for Ethernet MAC flow-control emulation.

The DVB Physical layer emulator behavioural module was used to emulate the functional operation of Physical layers of the MWS where error-rates are assumed to be zero. This core implements several configurable features to emulate different DVB physical layer equipment set-up. A rate adaptor is implemented that inserts null MPEG-TS Cells as would be done by a DVB Modulator if the DVB SPI Output port symbol rate is lower than the DVB-SPI Input Port symbol rate. A 188 MPEG-TS to 204 MPEG-TS format conversion process is also implemented to emulate the external conversion process that can be performed by the DVB Physical layer Emulator. If this function is enabled, the 188 MPEG-TS on the DVB-SPI output port will be converted to 204 MPEG-TS format. Lastly, the DVB Physical layer emulator module can also be configured as one of the following modes; i) 'loop-back bypass mode' – where DVB-SPI Output port is directly connected to DVB-SPI Input port; ii) 'loop-back with DVB physical layer emulation mode' – where the processes in the DVB Physical layer equipment are emulated; or iii) 'no loop back mode' – where the loop-back path is broken. Since the function of the SID Tag and SID Filter modules are designed to prevent loop-back Ethernet frames, it is noted that the SID generic parameters for the SID Tag and SID filter modules have to be configured to use different SID addresses to disable loop-back prevention.

The simulation model that describes the analogue *Phased Locked Loop (PLL)* device is provided by Altera and was used initially in simulations and can be instantiated as a component in VHDL. The actual device is a hardwired analogue Phased Locked Loop (PLL) available on the Cyclone FPGA. This device is used by the MPEG-TS framer of the 'Ethernet to DVB/MPEG-TS Encoder' core and hence an abstract simulation model

is required for simulation. Although the simulation model provided by Altera was found to work well, the required time for each simulation run was very long on the workstation. This is understandable because the Altera PLL model required a fine granularity of 1 ps for simulation timing due to the analogue nature of the device. To solve this problem, a further abstraction was done with the development of a behavioural PLL model. In simulation, the behavioural PLL model was used in place of the Altera PLL model which allowed the simulation to run at a courser granularity of 1 ns for simulation timing, and at a significantly increased simulation speed. However the behavioural model has to be replaced by the PLL component instantiation for it to by synthesizable. The VHDL codes for PLL in synthesis and simulation models are shown below.

Code used to for synthesis,

```
...
core_pll : cycn_pll
port map
(
      inclk0      => reference_clk,
      areset      => reset,
      c0          => pll_clk -- 14MHz
);
...
```

VHDL Code used for simulation,

```
...
behavior_pllclk : process
begin
      pll_clk <= '0','1' after 35 ns; -- 14MHz
      wait for 71 ns;
end process;
...
```

The Ethernet Frame Monitor behavioural module reads all Frames from the MII interface and verifies it against the same test vector set that was used by the Ethernet frame generator Module. Once an anomaly in the received frames is detected, an assertion is fired which halts the simulation run. A message that accompanies the assertion will tell the designer when the anomaly is detected and the symptoms that caused the anomaly. It is then up to the designer to locate and fix the anomaly or 'bug' in the design. If the simulation run is completed without firing any assertions, a message indicating the statistics of the simulation is printed on screen to indicate to the designer

various performance parameters so that the design can be adjusted for further optimisation.

Another function of the Ethernet Frame Monitor module is to detect Ethernet Pause frames that are sent by the device under test via the interfacing MII interface. The Ethernet frame monitor must detect Ethernet Pause frames and react to it in a way that emulates a remote Ethernet MAC. This function is implemented using a counter and combinatorial logic that asserts the disable signal to the Ethernet Frame Generator module for the duration indicated by the Pause Frame received.

4.3.2. Test Vectors and Assertions

To test the MWS subscriber interface core, the Ethernet frame generator was programmed to generate vectors in two phases. Roughly equal halves of the simulation period will be used to run Phase One and Phase Two. The two test phases are designed to cover a broad area of the MWS Subscriber Interface functional operations where code coverage of 100% is reported by the ModelSim simulation tool, for simulation run durations above 1.3 s or about 30,000 Ethernet frames. Duration for a simulation run can be set by specifying Ts generic variable in the VHDL test-bench source code. The simulation phases can be visualised in the diagram below:

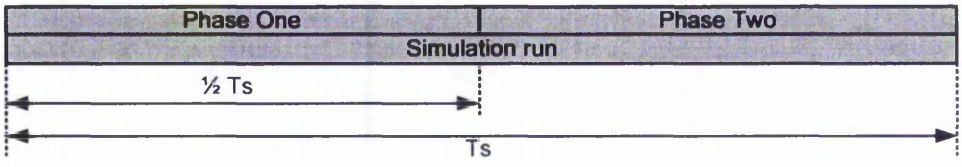
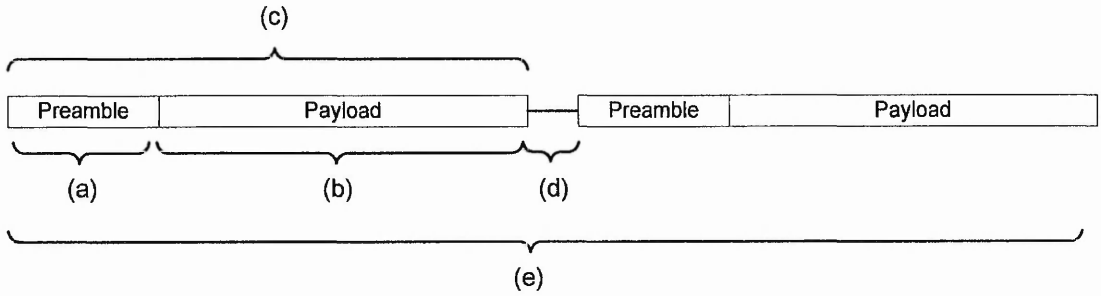


Figure 4-15: Simulation Test Phases

Phase One test vectors consist of a series of Ethernet frames incrementing in size by one byte from the minimum to maximum legal size. The series iterates to the minimum size frame once the maximum size is reached. The payload inside each frame is filled with byte values incrementing through the repeating sequence of the hexadecimal range 00 to FF. Phase one was designed to ensure that the Subscriber interface cores will be able to support frames of all sizes and of all byte values.

In Phase Two, test vectors are generated from a constant stream of Ethernet Frames with a preconfigured size of 'i' bytes and inter-frame gap of 'j' bytes. The payload are also filled with a preconfigured constant byte value 'k'. For example, a test run with the configuration settings of i = 1518, j = 24 and k = SOF|EOF, will produce a stream of maximum sized frames, with minimum inter-frame-gap and filled with SOF|EOF byte values which can be used to measure the syntactic worst case condition of the MWS subscriber interface.

The Ethernet Frame Verifier is responsible for the automatic verification process while simulation is being run. The Ethernet Frame Monitor expects to receive the exact copy of frames sent out by the Ethernet Frame generator because the frames are looped back by the DVB Physical Layer Emulator. Each byte of every frame is checked using VHDL assertion statements and if any of the five check parameters are not met (see figure below), the corresponding assertion will be fired. When the assertion is fired, the simulation process is halted and a message indicating the location of the error is displayed on to the console interface of the ModelSim simulation tool for manual debugging.



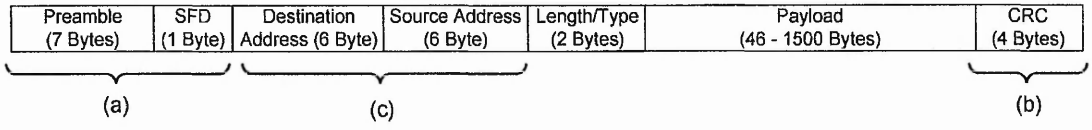
Assertions
(a) Fired if the number of preamble and SFD are not exactly 14 nibbles.
(b) Fired if the payload byte value does not exactly match the test vector.
(c) Fired if the frame size is smaller or larger than the expected.
(d) Fired if the minimum inter-frame-gap of at least 24 nibbles is not reached.
(e) Fired if frame loss is detected by comparing total frames sent and received.

Figure 4-16: Assertions for Ethernet Frame Verifier module

4.3.3. Header Suppression Effects

Header Suppression was implemented in the MWS interfaces that achieve a gain in throughput without loss of information or robustness of MWS transmission. This section studies the effects of the Header Suppression core implemented in the MWS Subscriber Interface as well as other more sophisticated techniques.

As described in 4.1.2, the Header Suppression core implemented scans and strips Preamble and SFD fields from each Ethernet frame at the transmitter and reconstructs it at the receiver thus shortening the packet that is sent through the MWS. The compulsory preamble sequence that consists of alternating '1's and '0's and a start_of_frame byte are used to synchronize the electronics in receiving Ethernet ports. However, the DVB interfaces use a separate synchronization mechanism. Therefore, regardless of the Ethernet frame size, the Preamble and SFD fields that consists of 56 bits and 8 bits respectively can be removed without any negative consequences to the robustness of the MWS DVB links.

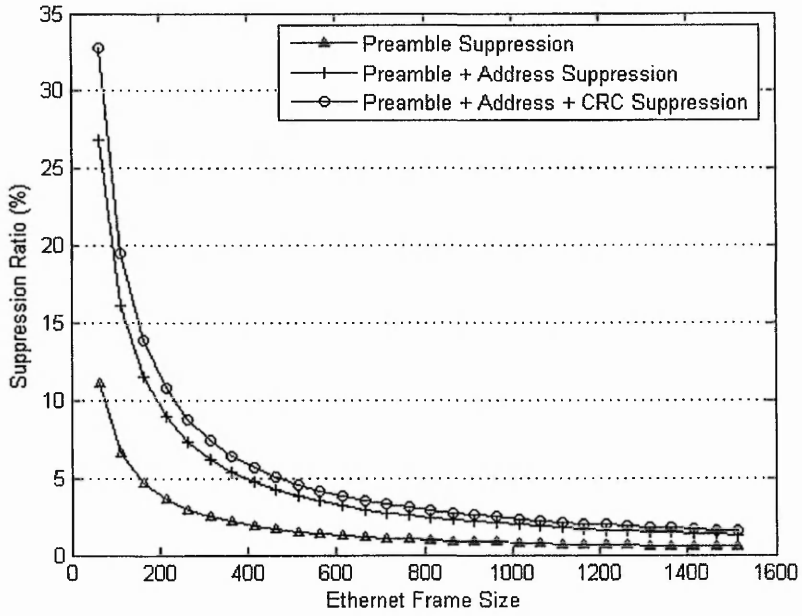


Scheme
(a) Preamble/SFD supression
(b) Address compression
(c) CRC trailer suppression

Figure 4-17: Ethernet Frame Header Compression Schemes

Theoretically, the CRC trailer removal and other more sophisticated techniques such as 'address compression' can be applied to achieve further gain in throughput. The Ethernet frame CRC trailer can be removed completely at the transmitter to shorten the amount of data needed to be sent. At the receiver, the CRC trailers can be regenerated from the payload data of the Ethernet frames. Address compression works by exploiting the fact that address fields are often repeated in consecutive frames in a data stream. Hence, the addresses can be abbreviated and regenerated if this fact is known by the receiver as explained in [36].

The effects of header suppression are illustrated in the figure below. Preamble, SFD and CRC fields have a fixed size where the compression ratio can be easily calculated. However, an ideal address compression is assumed here where the Ethernet Address fields are completely removed achieving the theoretical limit of layer-2 address compression.



As can be seen in the graph, Header Suppression works much better with 64 Byte frames with compression ratio of up to 33.3% compared to 1.6% for 1518 byte frames assuming CRC, Address and Preamble suppression are applied. However, the usual statistics (see 3.1) suggest that total bandwidth is normally dominated by large 1500 byte frames. Therefore, header suppression may not be very beneficial, especially if schemes such as address suppression and CRC removal are used that can induce negative impact on the robustness of the data traffic - since errors may not be detected.

4.3.4. Optimal Threshold value for Maximum Utilisation

This subsection describes a simulation study that uses the MWS Subscriber Test-bench. The purpose of this study are; i) to determine an optimal 'congestion threshold' setting in the Congestion Detector module (Module 5) of the 'Ethernet to DVB/MPEG-TS Encoder'; and ii) to measure the maximum raw Ethernet throughput using an ideal transport and network protocol stack (instead of TCP/IP or UDP/IP). The ideal transport

and network protocol stack is implemented by the 'Ethernet Frame Generator' simulation model which is capable of aggressively utilising available bandwidth with zero processing delays. Four sets of simulation runs that are discussed in this subsection are given simulation indices 1 to 4 in to aid clarity, see Table 4-5 below. For each simulation set, the encoding rate or Ethernet frame size are varied.

	Minimum sized Ethernet Frames	Maximum sized Ethernet Frames
14 Mbps MWS Subscriber Interface	Simulation Set 1	Simulation Set 2
56 Mbps MWS Subscriber Interface	Simulation Set 3	Simulation Set 4

Table 4-5: Indices for Simulation runs

'Congestion threshold' is defined here as the number of bytes held in the buffer i.e. the buffer occupancy level of the SPI Output Buffer before it is considered to be in congested state. Due to the bursty nature of packet based Ethernet networks coupled with an in-band flow control mechanism, a certain amount of buffering is needed to ensure that the DVB-SPI Output port always has data to transmit. In view of this being a hardware prototype, the aim of this simulation experiment is to find an ideal buffer threshold value for the DVB-SPI Output to achieve minimum latency at 100% sustained utilisation for two specific instantiation of the MWS Subscriber Interface; one for operation at 56 Mbps encoding rate and another at 14 Mbps. 56 Mbps MWS Subscriber Interfaces are used for the point-to-point microwave link demonstration as described in section 4.4, while the 14 Mbps MWS Subscriber Interfaces are used in the Ethernet Hub Emulation (EHE) MWS demonstration as described in section 5.3. The congestion threshold of the SPI Output buffer can be visualised in the diagram below:

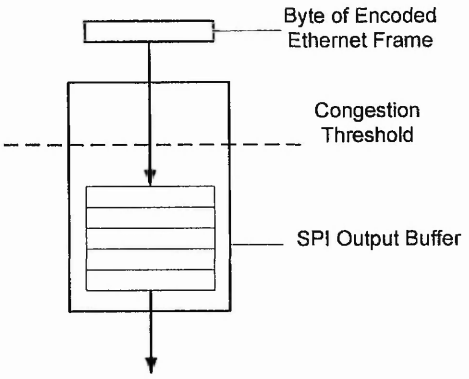


Figure 4-18: Buffer Threshold of SPI output Buffer

The test-bench set up for this investigation instantiates an Ethernet Frame generator that behaves as an infinite data source. For each buffer threshold configuration in Simulation sets 1 and 3, 2000 minimum sized frames were queued. Elsewhere for simulation sets 2 and 4, 220 maximum sized frames were queued. As the Ethernet Frame Generator will attempt to transmit all frames as quickly as possible, the MWS Subscriber interface will immediately fall into congestion state and will begin to perform flow control.

In each simulation set, the test-bench will complete ten queues for a range of congestion_threshold configurations. Each frame entering or exiting the MWS Subscriber Interface are time-stamped and recorded in a file for post-simulation analysis. The time-stamp files are used by a Matlab analysis program developed to calculate the 'maximum sustained throughput' and 'average latency' performance matrix. The equations and calculations performed by the Matlab analysis program are detailed in the following paragraphs whilst the output of the analysis program for the four simulation sets are summarized in Figure 4-19 to Figure 4-22.

The 'sustained maximum throughput' performance for each queue is calculated using the following equations:

The time to complete sending all frames in the queue, $T_{sim,}$ is given by:

$$Tsim = T_N - T_1 \qquad \textbf{Eq. 4-1}$$

Where,

T_1 is the time-stamp of first frame of the queue and

T_N is the time-stamp of the last frame of the queue

$$SustainedMaxThroughput = \frac{(N-1).S_f}{T_{sim}} \qquad \textbf{Eq. 4-2}$$

Where,

N is the number of frames in the queue

S_f is the size of the frames

The 'average latency' is calculated by examining the ingress and egress time-stamps for each Ethernet frame. T_i, is the time-stamp when the first nibble of an Ethernet frame arrives at the ingress MII port and T_e, is the time-stamp when the first nibble exits the egress MII port. The frame latency, L_f, is defined as the period between those two instants and can be expressed as:

$$Lf = Te - Ti \qquad \textbf{Eq. 4-3}$$

The average latency, $L_{ave,}$ of N number of frames can be expressed as:

$$L_{ave} = \frac{\sum Lf}{N} \qquad \textbf{Eq.4-4}$$

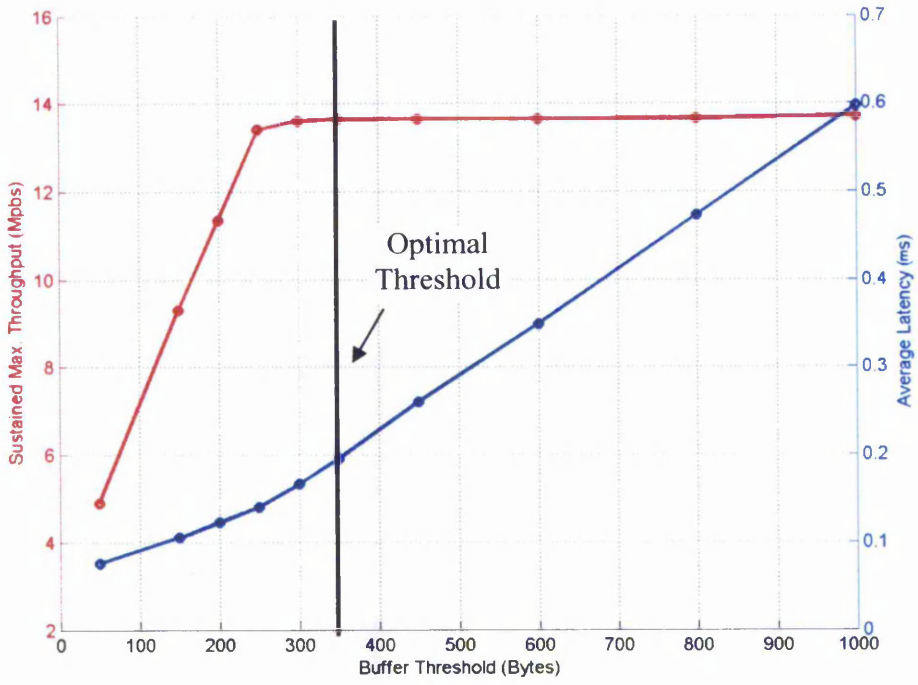


Figure 4-19: Simulation Set 1 - 14Mbps 'Ethernet to MPEG-TS/DVB Encoder' Optimal Congestion threshold for minimum Ethernet frame size

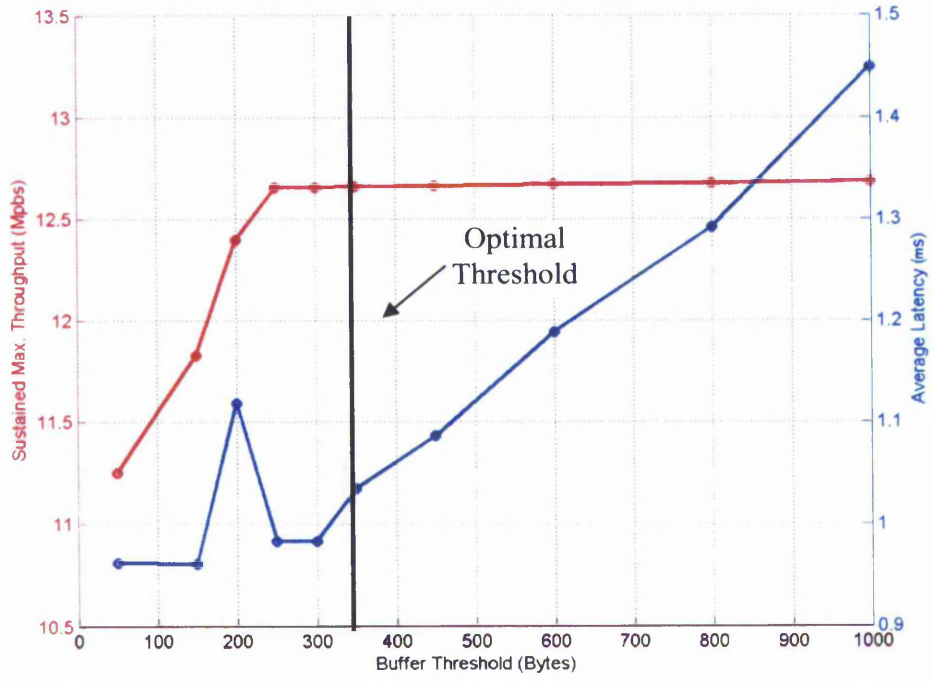


Figure 4-20: Simulation Set 2 – 14Mbps 'Ethernet to MPEG-TS/DVB Encoder' Optimal Congestion threshold for maximum Ethernet frame size

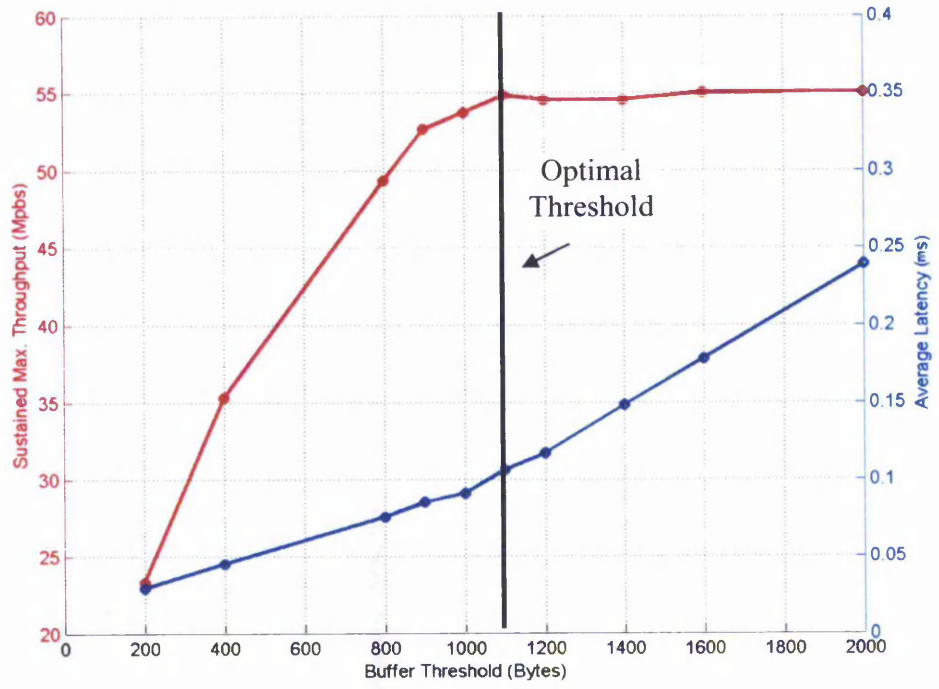


Figure 4-21: Simulation Set 3 – 56 Mbps 'Ethernet to MPEG-TS/DVB Encoder' Optimal Congestion threshold for minimum Ethernet frame size

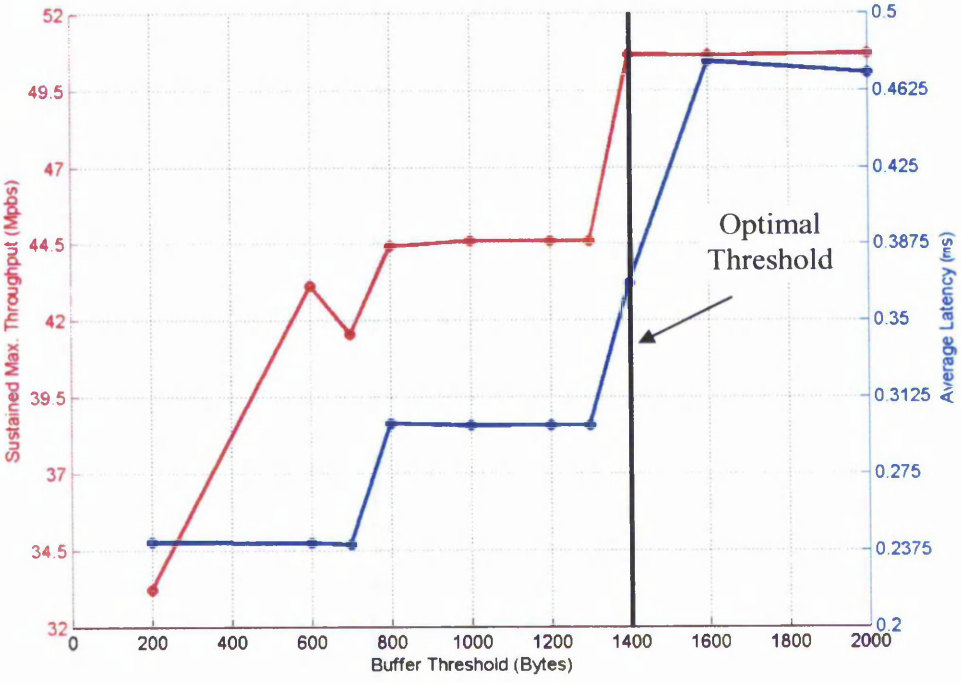


Figure 4-22: Simulation Set 4 – 56 Mbps 'Ethernet to MPEG-TS/DVB Encoder' Optimal Congestion threshold for maximum Ethernet frame size

The optimal threshold value was determined by observing the 'maximum sustained throughput' curve. Below the optimal value, an increase in congestion threshold significantly increased the 'maximum sustained throughput'. However, increasing the congestion threshold above the optimal value will not have significant benefit (if any) to 'maximum throughput'. Besides requiring more memory resources, increasing the congestion threshold will generally improve of the 'average latency' performance. In Simulation sets 2 and 4 it was observed that the curves are less gradual than Simulation sets 1 and 3. This is attributed to the maximum sized frames used in the simulation as larger frames would tend to have a coarser response to changes in buffer size. For the 14Mbps MWS Subscriber interface, the higher value between Simulation 1 and 2 was taken as the optimal congestion threshold; and the higher value of Simulation sets 3 and 4 was taken for 56Mbps MWS subscriber. This resulted in optimal congestion threshold values of 350 Bytes and 1400 Bytes for 14Mbps and 56Mbps MWS subscriber interface implementations respectively.

The 'maximum raw Ethernet throughput' achieved by the 14 Mbps MWS subscriber interface using the optimal threshold value of 350 Bytes was; 13.63 Mbps for minimum sized Ethernet frames (Simulation 1), and 12.66 Mbps for maximum sized Ethernet frames (Simulation 2). For 56 Mbps MWS subscriber interface 'maximum raw Ethernet throughput' using the optimal threshold value of 1400 Bytes was; 54.54 Mbps for minimum sized Ethernet frames (Simulation 3), and 50.67 Mbps for maximum sized Ethernet frames (Simulation 4).

The 'maximum raw Ethernet throughput' is cross verified with a theoretical estimated value using mathematical calculations. For these calculations the Ethernet payload bytes in each frame are assumed to consist of uniform random byte values. Comparison of the theoretical estimates and the simulated throughput show very close correlation, suggesting good integrity of FPGA implementation (see Table 4-6 in section summary). It is observed that 'maximum raw Ethernet throughput' of simulation is always slightly higher compared to the theoretical estimates. This is attributed to the initial burst of frames allowed by the buffers in simulation which skews the average throughput. Details of the calculations are described in the following paragraphs using the function for 'maximum raw Ethernet throughput' shown below:

$$Max.Eth.Throughput = QPSKcoeffnt \times PuncturingCoeffnt \times SymbolRate \times MPEGHeaders \times MPEGmodeCoeffnt \times HeaderCompresionGain \times EncapOverheadCoeffnt \quad \textbf{Eq. 4-5}$$

Where,

QPSKcoeffn is the coefficient for QPSK modulation = *2/1*

PuncturingCoeffnt is the coefficient for Puncturing rate = 7/8

SymbolRate is the QPSK Symbol rate

MPEGHeaders is the coefficient for MPEG Header overheads = 184/188

MPEGmodeCoeffnt is the overhead coefficient for MPEG format = 188/204

HeaderCompressionGain is the gain coefficient for preamble suppression

EncapOverheadCoeffnt is the overhead coefficient for OBS encapsulation

For Simulation set 1, *SymbolRate* = 8 MSym/s, *HeaderCompressionGain* = $\frac{72}{64}$ and *EncapOverheadCoeffnt*= $\frac{64}{67}.\frac{255}{256}$. Hence, the theoretical estimated 'raw Ethernet throughput' for Simulation set 1 is:

$$Sim1.Eth.Througput = \left(\frac{2}{1}\right)\times\left(\frac{7}{8}\right)\times\left(8\times10^{6}\right)\times\left(\frac{184}{188}\right)\times\left(\frac{188}{204}\right)\times\left(\frac{72}{64}\right)\times\left(\frac{64}{67}.\frac{255}{256}\right) \quad \textbf{Eq. 4-6}$$

$$= 13.52Mbps$$

For Simulation set 2, *SymbolRate* = 8 MSym/s, *HeaderCompressionGain* = $\frac{1526}{1518}$ and *EncapOverheadCoeffnt*= $\frac{1518}{1521}.\frac{255}{256}$. Hence, the theoretical estimated 'raw Ethernet throughput' for Simulation set 2 is:

$$Sim2.Eth.Througput = \left(\frac{2}{1}\right)\times\left(\frac{7}{8}\right)\times\left(8\times10^{6}\right)\times\left(\frac{184}{188}\right)\times\left(\frac{188}{204}\right)\times\left(\frac{1526}{1518}\right)\times\left(\frac{1518}{1521}.\frac{255}{256}\right) \quad \textbf{Eq. 4-7}$$

$$= 12.62Mbps$$

For Simulation set 3, *SymbolRate* = 32 MSym/s, *HeaderCompressionGain* = $\frac{72}{64}$ and *EncapOverheadCoeffnt*= $\frac{64}{67}.\frac{255}{256}$. Hence, the theoretical estimated 'raw Ethernet throughput' for Simulation set 3 is:

$$Sim3.Eth.Througput = \left(\frac{2}{1}\right)\times\left(\frac{7}{8}\right)\times\left(32\times10^{6}\right)\times\left(\frac{184}{188}\right)\times\left(\frac{188}{204}\right)\times\left(\frac{72}{64}\right)\times\left(\frac{64}{67}.\frac{255}{256}\right)$$

Eq. 4-8

$$= 54.06Mbps$$

For Simulation set 4, *SymbolRate* = 32 MSym/s, *HeaderCompressionGain* = $\frac{1526}{1518}$ and *EncapOverheadCoeffnt*= $\frac{1518}{1521}.\frac{255}{256}$. Hence, the theoretical estimated 'raw Ethernet throughput' for Simulation set 4 is:

$$Sim4.Eth.Througput = \left(\frac{2}{1}\right)\times\left(\frac{7}{8}\right)\times\left(32\times10^{6}\right)\times\left(\frac{184}{188}\right)\times\left(\frac{188}{204}\right)\times\left(\frac{1526}{1518}\right)\times\left(\frac{1518}{1521}.\frac{255}{256}\right)$$

Eq. 4-9

$$= 50.48Mbps$$

4.3.5. Characterisation of Latency/Throughput Adaptive Behaviour

The simulation study presented in this subsection analyses the adaptive behaviour characteristic of the MWS Subscriber Interface whose mechanism is implemented by the OBS Encoder (see Module 1). The study was done for the 56 Mbps version of MWS Subscriber Interface's by determining its 'saturation level'. The 'saturation level' is defined here as the level of capacity utilisation at which the MWS Subscriber Interface begins the latency/throughput adaptation. Basically, the MWS Subscriber Interface will encode Ethernet Frames at very low latency when the DVB-SPI output is not under heavy utilisation (below saturation level). However, it will begin to adapt to increase in utilisation by trading-off its low latency performance for increased sustained throughput; and eventually reaching its maximum sustained throughput.

For this simulation study, the Ethernet Frame Generator was programmed to generate constant streams of a specific throughput rate Rt, by ensuring that the Inter-Frame-Gap between each Ethernet frame is equivalent to the product G.Tb. Where G is the number of clock cycles and T_b is the period for a clock cycle of the MII. The equation for the throughput rate Rt can be written as follows:

$$R_t = \frac{(S_f)(4)}{\left(\frac{G}{2} + S_f\right)(T_b)} \qquad \textbf{Eq. 4-10}$$

Here, S_f is measured in bytes; Rt in bits per second; and T_b in seconds.

The equation can be rearranged to find G from Rt as follows:

$$G = 2\left(\frac{(S_f)(4)}{(R_t)(T_b)} - S_f\right) \qquad \textbf{Eq. 4-11}$$

For example, if the target throughput rate to be generated is Rt = 50Mbps using 72-Byte Ethernet frames, $S_f = 72\,bytes$. The value for G can be calculated as follows:

$$\begin{aligned} Tb &= \frac{S_{mii}}{\mathrm{Re}} \\ &= \frac{4}{100\times10^6} \\ &= 40\times10^{-9}\,s \\ &= 40ns \end{aligned} \qquad \textbf{Eq. 4-12}$$

Here, $\mathrm{Re} = 100\times10^6$ as 100 Mbps Ethernet is assumed; and symbol size, S_{mii} = 4 since each clock cycle carries 4 bits of data over the MII Bus.

Therefore,

$$\begin{aligned} G &= 2\left(\frac{(S_f)(8)}{(R_t)2Tb}\right) - S_f \\ &= 2\left(\frac{(72)(8)}{(50\times10^6)(2)(40\times10^{-9})}\right) - 72 \\ &= 144 \end{aligned}$$

In other words, for a 50Mbps throughput using 72 byte frames, the inter-frame-gap between consecutive frames should be: $ifg = G.Tb = 144\times40ns = 5760ns$. Thus, using

equation Eq.4-11 above, the Ethernet Frame generator core is able to automatically calculate the value for G given the target throughput Rt.

For a specific hardware 56 Mbps MWS Subscriber Interface prototype, the saturation level may be determined in this simulation experiment. The setting for the Congestion_threshold value is 1400 (see subsection 4.3.4). Ten simulation runs were carried out varying the offered load values between 1Mbps through 55 Mbps. For each throughput value given to the Ethernet Frame generator, five thousand (N = 5000) 72-byte Ethernet frames are generated and sent to the Subscriber MWS interface. Then, the average latencies, $L_{ave,}$ for each of the simulation runs were calculated in the Matlab tool using the procedure described in section 4.3.4, Eq.4-4. Finally, the results can be plotted in a graph to illustrate the adaptive behaviour of the MWS Subscriber interface as shown in Figure 4-23.

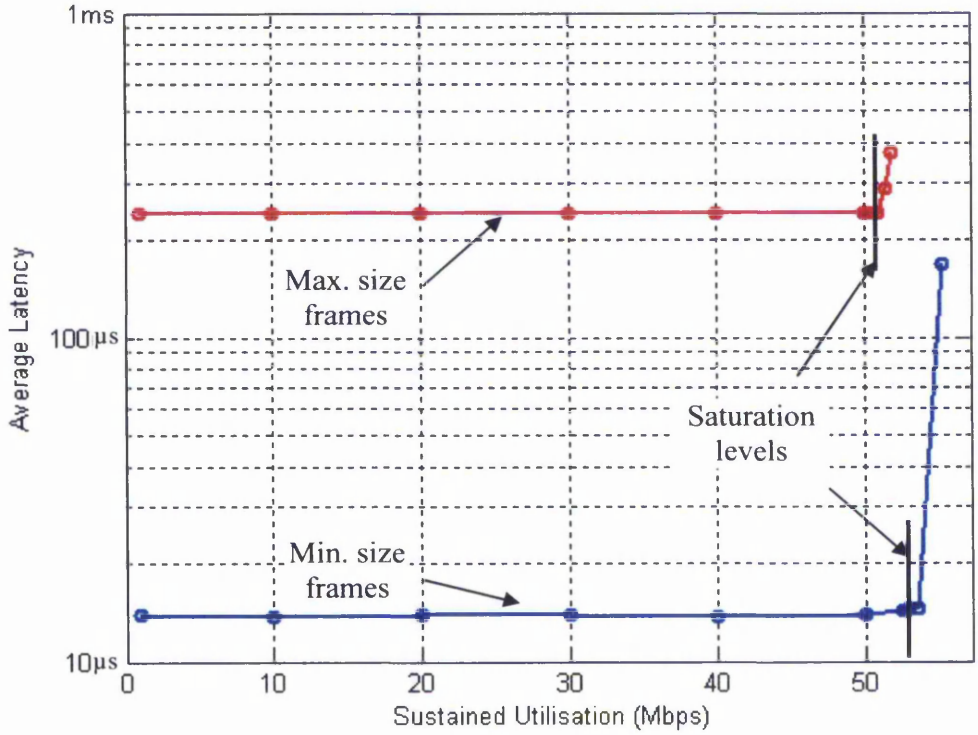


Figure 4-23: MWS interface adaptive behaviour

From the graph, the 'saturation level' was determined to be 53.3 Mbps for minimum frame size and 50.2 Mbps for maximum frame size which is about 98% of the DVB-SPI Output port capacity. In conclusion, the 'flushing' performed by the Byte Stuffing

Encoder module successfully minimised the latency for utilisation levels below the link saturation level. Above the saturation level the latency increases dramatically as buffering was needed in order to make use of the last 2% of capacity. It can also be concluded here that for small frames, when the link utilisation is above the saturation level, the latency induced by buffering will be much greater than the latencies introduced by the encoding and store-and-forward latencies of the decoding processes of the hardware.

Summary of Section 4.3

The objectives of the simulation platform were met because it facilitated the thorough verification of the MWS Subscriber core that greatly helped to rid the design of bugs. Further, analysis, using the Matlab tool, of the data collected from the simulation platform, revealed the optimal Congestion Threshold parameters for two specific implementations. Table 4-6 summarises the key simulation results from this section including; the identified optimal Congestion_Threshold, achieved performance, and latency performance at maximum utilisation. The simulation and analysis provides evidence that the 56Mbps MWS Subscriber Interface has several advantageous characteristics when compared to previous systems such as:

- Dynamic adaptation to current utilisation levels, which maintain near-zero buffering latencies up to 98% utilisation level.
- Able to achieve sub 1 millisecond total encoding/decoding latency for all frame sizes even at maximum utilisation.
- Able to constantly sustain the maximum utilisation of the available MWS capacity with high efficiency.

Module Type	Optimal Congestion Threshold configuration	Theoretical Max. Throughput (*min. frame size/ max. frame size*)	Simulated Max. Throughput (*min. frame size/ max. frame size*)	Simulated Max. Latency (*min. frame size/ max. frame size*)
14 Mbps	350	13.52 / 12.62 Mbps	13.63 / 12.66 Mbps	1.0 / 1.9 ms
56 Mbps	1400	54.06 / 50.48 Mbps	54.54 / 50.67 Mbps	0.11 / 0.38 ms

Table 4-6: Summary of Simulation results

4.4. Experiments of Subscriber interfaces

In this section, development of a hardware test-platform and investigation of results from experiments are presented. The intention of developing a hardware prototype of the system is to gauge realistic and typical performance of the MWS Subscriber interfaces, where the complexity of the higher layer mechanisms as well as the relatively large time scales required for analysis renders VHDL simulation impractical. The focus is on the 56 Mbps version of the interface where experiments conducted reveal that the mechanisms incorporated complement each other and also interact well with higher layer mechanisms of networked hosts. It is shown that in addition to achieving high efficiency, the OBS encoder/decoder and its close integration with Ethernet successfully enable a sub 1.1 ms host-to-host latency. Furthermore, the Ethernet flow control mechanisms negotiated data rates with an unoptimised TCP/IP and UDP/IP stacks and achieved sustained utilisation of 94%; realistically transferring 45.95 Mbps of user data without packet loss.

4.4.1. The Point-to-Point MWS Experimentation Platform

An experimentation platform was built to evaluate realistic performance of a point-to-point link deployment between two MWS subscriber Nodes. The experiments were conducted using the 56 Mbps version of the Subscriber interfaces. Among the set of measurement capabilities of the test platform are; round-trip-time (latency) and jitter, and raw user maximum throughput for TCP/IP and UDP/IP protocols. The test platform is constructed from PC Hosts installed with benchmarking tools, MWS Subscriber Interface prototypes, DVB-S modulators, DVB-S set-top-boxes and available test instrument facilities. Set-up of the test platform is illustrated in the Figure 4-24, readers can refer to Appendix C for photographs.

Host A and Host B are installed with two operating systems (dual boot) which are Microsoft Windows XP SP2 and RedHat Linux kernel 2.14-18. For the Microsoft Windows operating system, the PCATTCP (PCA Test TCP) [148] benchmarking tool was installed. PCATTCP is a common tool used in the research community [149], which is used in the test platform for throughput and frame loss rate benchmarking. For latency and jitter measurement, the Ping tool was used within the Linux environment.

It was found that the internal operating system timer in Linux was more accurate than in Windows XP where the Ping tool reported round-trip-time measurements to 1 microsecond resolution in Linux, compared to 1 millisecond in Windows XP. A further investigation was done to calibrate the Ping tool, as explained in subsection 4.4.2, which shows that the Ping tool's accuracy is actually within 10 microseconds of the electrical signals measured on the wire. Linux would have been the operating system of choice since TTCP was also available in Linux, however the generic drivers in Linux operating system for the Ethernet network interface cards did not support Ethernet flow control which limits measurements to those below the saturation level in Linux. As the Ping measurements are performed below the saturation level, valid latency and jitter measurements were made in Linux using its high resolution timers.

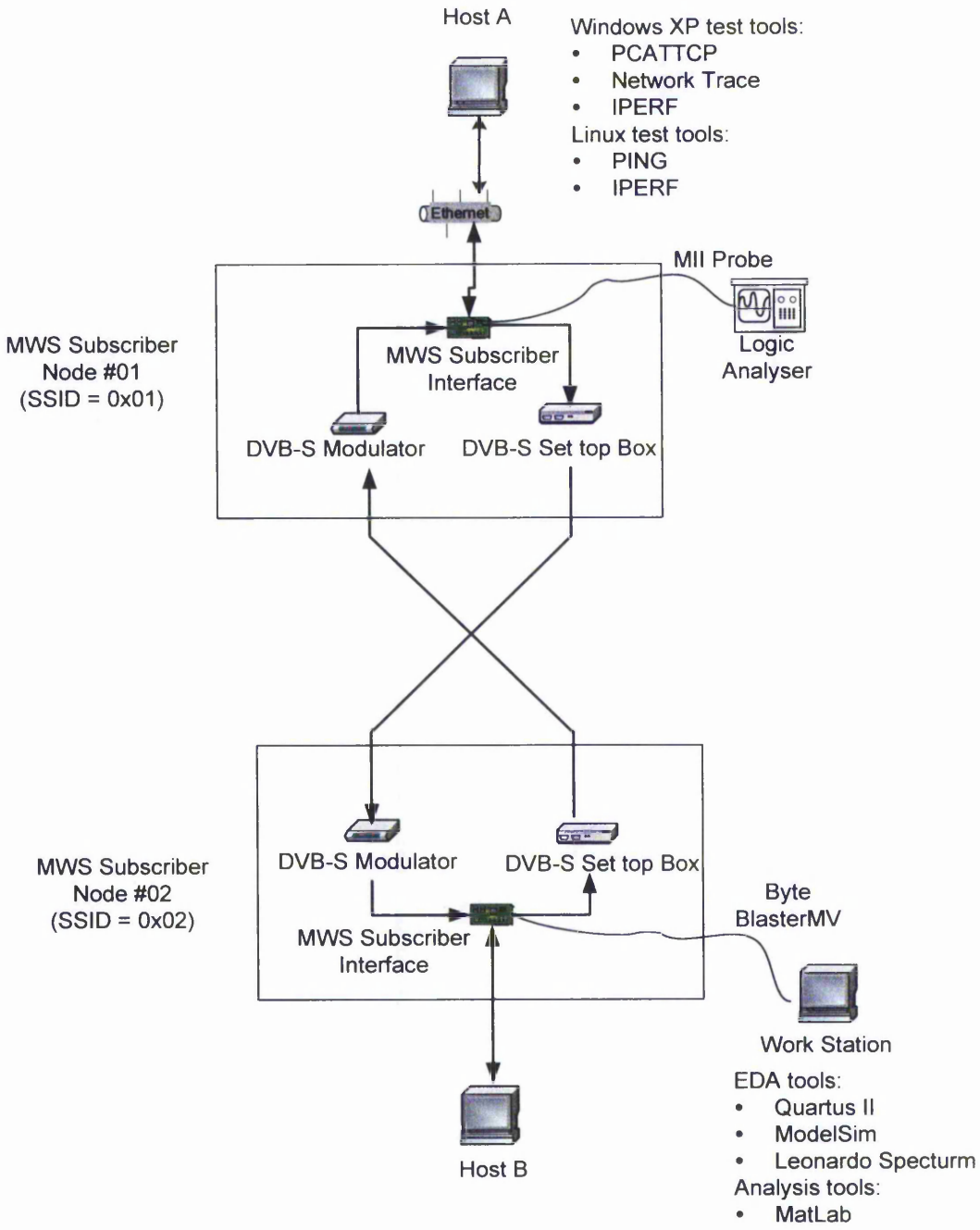


Figure 4-24: Point to Point Experimental Platform

Two prototypes of the MWS Subscriber Interface design were programmed onto Altera Cyclone EP1C6240-6 devices using the Quartus II Tool. Each of the MWS Subscriber Node is configured with a unique SID, i.e SID 0x01 and SID 0x02. The prototyping process involves Quartus II tool's in-built programmer which is used to 'download' the SRAM object file (.sof) via the Byte Blaster MV cable. The Byte Blaster MV cable

connects the workstation to the FPGA device via the parallel port on the workstation, and a JTAG port on the Cyclone development board. MWS Subscriber interfaces and DVB-S Modulator and Set-top-box were configured according to the specifications below, to form the prototype MWS Subscriber Nodes.

- Byte Stuffing Encoding : *Cut-through forwarding*
- Byte Stuffing Decoding : *Store-and-forward*
- Modulation: *QPSK*
- MPEG-TS mode: *204*
- Puncturing: *7/8*
- Symbol Rate: *32000 Ksym/s*
- MWS Subscriber Node 0x01 modulator carrier frequency : *950 MHz*
- MWS Subscriber Node 0x02 modulator carrier frequency : *800 MHz*

4.4.2. Calibration of Ping Measurements

In order for the test platform to produce reliable latency results, the experimentation platform must be calibrated. Calibration of the Ping tool was achieved by performing parallel tests with a more accurate measurement instrument and, through comparison, error introduced by the test platform can be known. An analysis of the experiment results were carried out to obtain a formula to accurately predict the error offset. Thus, using the formula, measurements taken from the Ping tool can be used to estimate the actual round trip time measurement. Via observation of the calibration work results explained in the following paragraphs, it was found that the Ping tool enabled latency measurements up to an accuracy of 10 microseconds. This was sufficient for latency measurements to be conducted without the use of a Logic analyser. Hence, the test-platform was able to run more elaborate and automated experiments for latency and jitter measurements as manual reading of latency measurements from logic analyser would not be required.

To sample the accuracy of the Ping tool, Ping request/reply runs from Host A to Host B were recorded by hand from the measurement displayed on the Ping tool console and Logic Analyser screen. The 1 GHz Logic Analyser instrument was initialised with the maximum resolution of 1 giga-sample per second to achieve measurements to 1 nanosecond accuracy. The measurement taken on the Logic analyser probe is assumed to

be the 'actual' round trip time, since the Logic analyser probe is connected to the electrical PCB wire trace of the Ethernet MII port which picks up the TTL (Transistor to Transistor Logic) voltage level changes.

Round-trip-time measurements from the Ping tool are denoted here as Tp, and measurements from the Logic Analyser as Td. The following figures and tables for Ping calibration measurements refer to diagrams included in Appendix D; examples of Tp Ping measurement as displayed on the Linux console can be found in Figure E-1, example of a screenshot Td measurement between a Ping request (PgReqs) and Ping reply (PgRply) captured on the Logic Analyser instrument is shown in Figure E-2, and thirty Tp/Td sample pairs for frame sizes 64, 500, 100, and 1500 were recorded by hand in a spread sheet shown in Table E-1. The results of those measurements are plotted in the Figure 4-25 below:

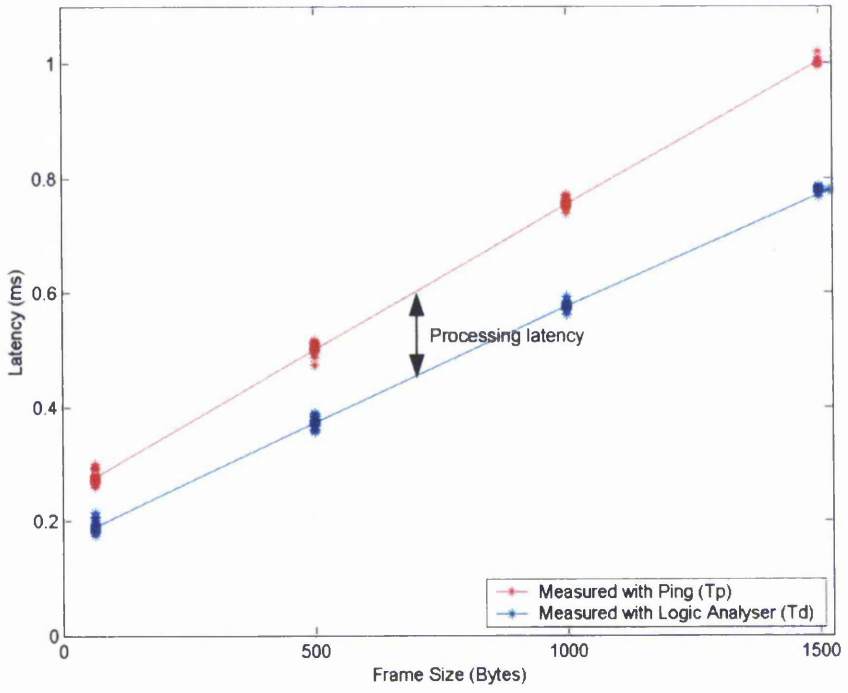


Figure 4-25: Comparison of Ping and Logic Analyser measurements

As the Ping tool operates at the Application layer, a small amount of processing latency is added to the 'actual' round trip time measured on the Logic analyser. This explains why the measurement from the Ping tool is observed to be always greater. However, the

plot on the graph reveals that, this processing latency is not constant for all frame sizes, but increases in correlation to the frame size. This processing latency appears as an error offset in the measurement result.

To determine the error offset for the Ping tool, an analysis on the sample difference of the thirty Tp/Td sample pairs for each of frame sizes 64, 500, 100, and 1500 was carried out as shown in the equations below. It is noted that the raw data were examined for irregularities and outliers that might skew the results.

The point estimate, or mean of the sample difference, $\overline{x_{Tp-Td}}$, for a 30-sample set is determined using the formula below:

$$\overline{x_{Tp-Td}} = \frac{\sum x_{Tp-Td}}{n}$$

The standard deviation, σ_{Tp-Td}, of the sample difference is given by:

$$\sigma_{Tp-Td} = \sqrt{\frac{\sum (x_{Tp-Td} - \overline{x_{Tp-Td}})^2}{(n-1)}}$$

The margin of error associated with the point estimate is calculated as follows, assuming normal distribution. Here, the z value of 2.33 is used for 99% confidence interval, which is read from the standard normal table.

$$\begin{aligned} Margin\ Of\ Error &= \pm z.\sigma_{Tp-Td} \\ &= \pm(2.33).\sigma_{Tp-Td} \end{aligned}$$

The Mean difference and associated margin of error results are plotted on the graph below.

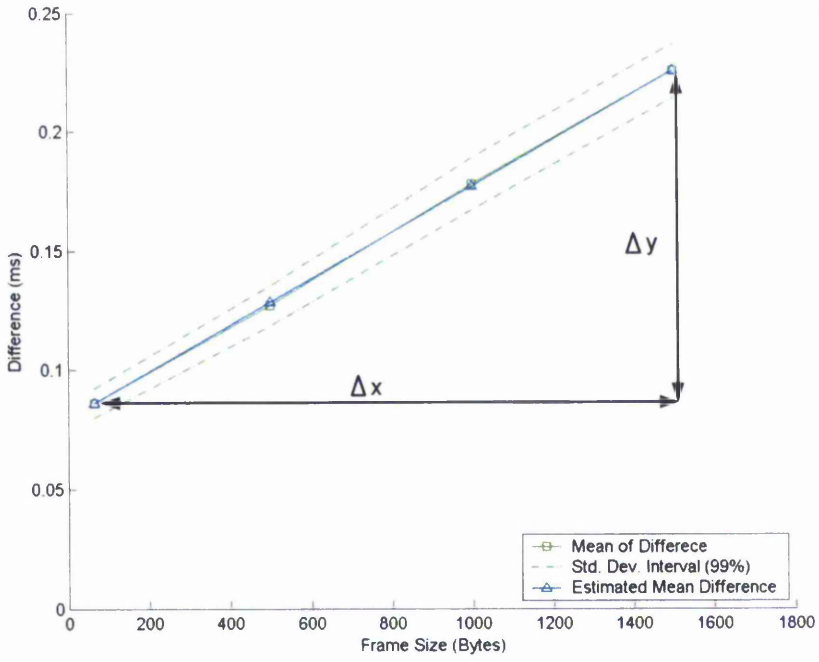


Figure 4-26: Mean Difference and Margin of Error associated with TP and Td Measurements

The analysis reveals that the margin of error is less than ± 0.0114 ms for all frame sizes, or roughly 10 microseconds. From the graph, it can also be seen that the mean difference between Tp and Td is has a linear increase with frame size. In order to estimate the mean difference, the following calculations were done:

From the general straight line function,

$$y = mx + c$$

The function of estimated mean difference, $\overline{x_{Tp-Td}}'$, is denoted as,

$$\begin{aligned}\overline{x_{Tp-Td}}' &= m(FrameSize\ ) + c \\ &= \frac{\Delta y}{\Delta x}(FrameSize\ ) + c \\ &= (9.69 \times 10^{-8})(FrameSize\ ) + 0.08 \times 10^{-3}\end{aligned}$$

Hence, the error offset can be determined by the equation below:

$$ErrorOffset = \overline{x_{Tp-Td}}' \pm Margin\ Of\ Error$$
$$= (9.69 \times 10^{-8})(FrameSize) + 0.08 \times 10^{-3} \pm 0.0114 \times 10^{-3}$$

With the Error Offset value obtained, the actual latency, Td, can be estimated to an accuracy of about 10 microseconds from Ping tool measurements Tp. Hence, a general formula to calculate Td from Tp is shown below:

$$Td = Tp - ErrorOffset$$
$$= Tp - ((9.69 \times 10^{-8})(FrameSize) + 0.08 \times 10^{-3} \pm 0.0114 \times 10^{-3})$$

In other words, it can be said that further individual measurements, using the Ping tool on the developed test platform, will fall within 10 microsecond of the actual round trip time with 99% certainty. Since Td can be estimated from Tp, the use of Logic Analyser instrument was no longer required by the test platform for accurate round-trip-time measurements.

4.4.3. Latency and Jitter Experiments

This subsection details experimentation done to gauge the latencies and jitter introduced by each subsystem of the point-to-point link due to varying Ethernet frame sizes. The experiments would investigate the other latency characteristic dimension (variation of Ethernet frame sizes) of the MWS Subscriber Interface; adding to the simulation analysis in subsection 4.3.5 that studied the latency characteristics due to increase in link utilisation. Finally, the experimental results are cross verified with simulation analysis to show a high degree of consistency. Note that the different dimensions from simulation and experimentation results as well as the cross verification points are graphically shown in the chapter conclusions.

The point-to-point MWS Experimentation platform is divided into three subsystems which are: PC Hosts & Ethernet, MWS Subscriber Interface, and DVB-S modulators & Set-top-boxes. In total, three experiments were conducted with different set-ups, each time adding a subsystem in the chain that make up the point-to-point link. The experiment set-ups are illustrated in Figure 4-27. In order to negate latency caused by link saturation, the experiments are conducted at near zero utilisation.

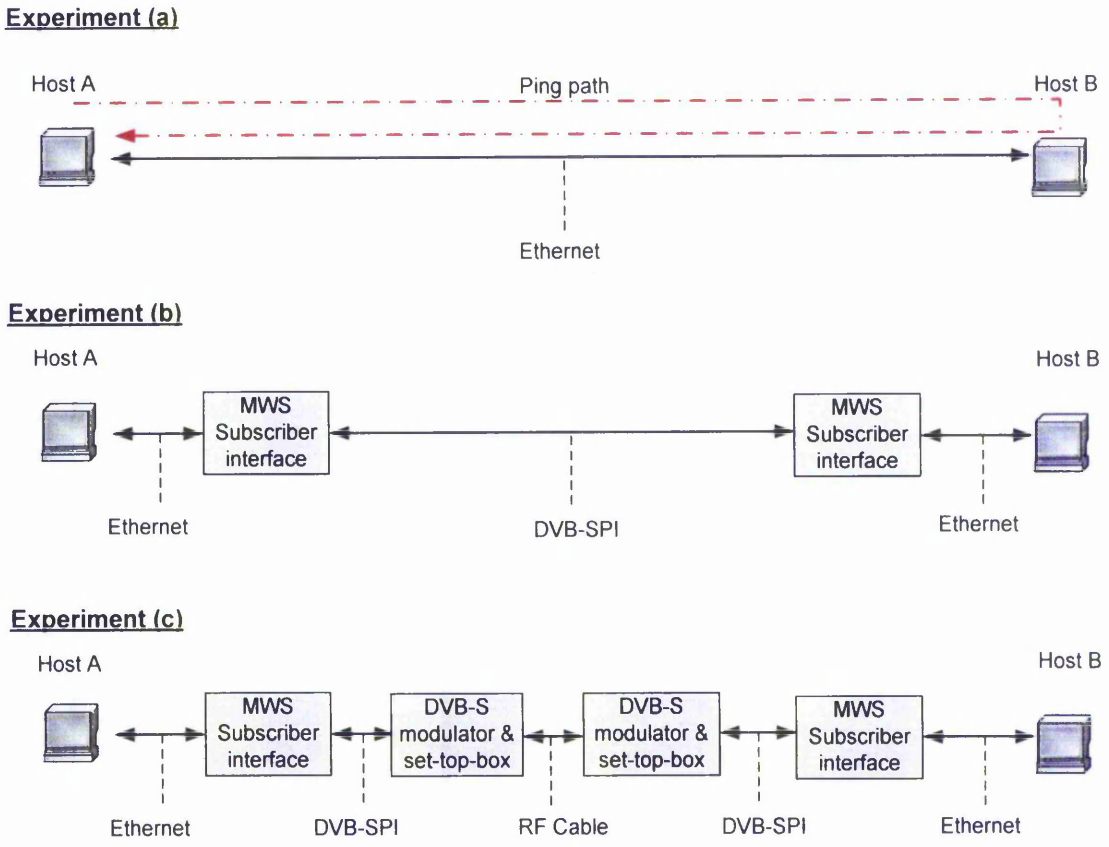


Figure 4-27: Latency and Jitter experiments set-up a) Ethernet cable and Hosts ; b) Ethernet cable, Hosts and MWS Subscriber Interfaces; c) Ethernet cable, Hosts, MWS Subscriber Interfaces and Modulator/set-top-boxes

Since the latency effects studied here vary with the frame size, the legal range of Ethernet frames was evaluated. For each experiment set-up, a Linux script was used to perform Ping round trip time measurements. A total of 3400 Ping request/reply trace results are saved in a file for post processing in Matlab. That is, 200 Ping requests for each frame size from the values of 72-1526 with 100 byte increments. In Matlab, the mean latency for each frame size is calculated and the error offset is applied based on calibration calculations in subsection 4.4.2. The results are plotted in the graph shown in Figure 4-28. In the plot, the variation in latency is shown as jitter bounds.

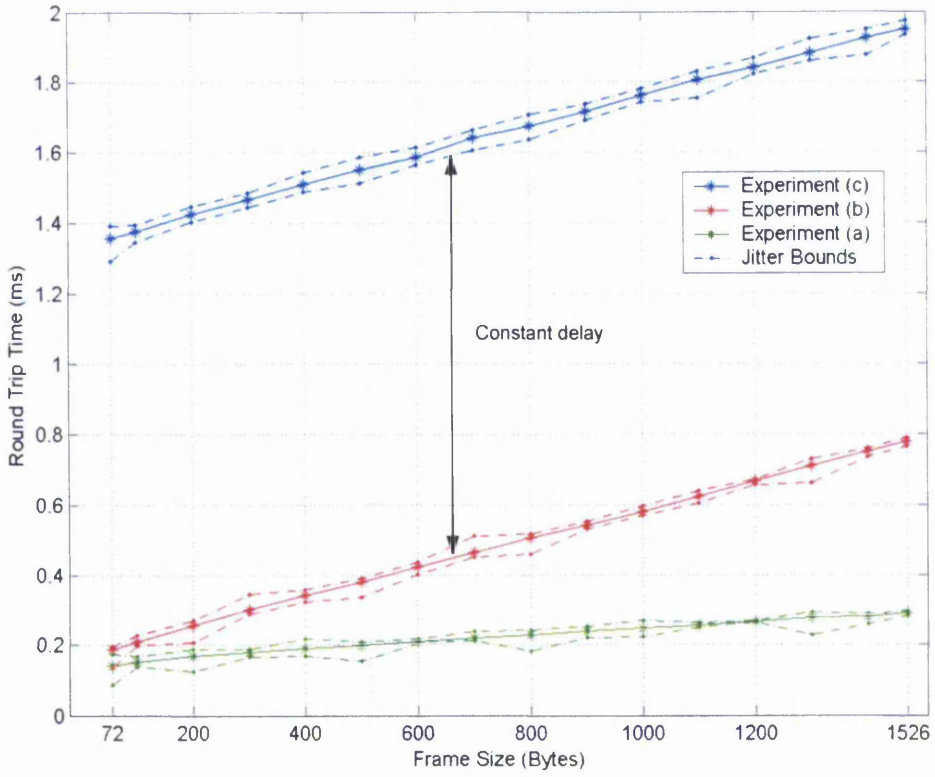


Figure 4-28: Latency and Jitter Measurements of experiments (a) , (b) and (c)

The results on the graph can be analysed by comparing the difference between the experiments, since each experiment, represents the cumulative latency of additional subsystem in the chain. Therefore, each subsystem contributing to the total latency can be isolated by subtracting the results of the experiments. From the graph, it is shown that the latencies between experiments (c) and (b) are constant throughout the range of the frame sizes. This means that the DVB-S Modulator and Set-top-boxes introduce a constant round-trip-time (1.172 ms), or constant delay of 0.586 ms one way. Results from experiment (b) and (a) can be used to isolate latency caused by the MWS Subscriber Interface.

Cross verification was done at two points (i.e at frame sizes 72 and 1526) between experimental latency and simulated latency results. The cross verification revealed that experimental latency shown here is indeed consistent with the simulated latency results presented in subsection 4.3.5. Cross verification points are shown graphically in the

chapter conclusions, Figure 4-34. The detailed calculations for cross verification are shown below, whilst further analysis on jitter results are discussed in the following paragraphs.

From experimental measurements of (a) and (b),

$Td_{a72} = 0.12ms \pm 0.01$
$Td_{b72} = 0.18ms \pm 0.01$
$Td_{a1526} = 0.29ms \pm 0.01$
$Td_{b1526} = 0.78ms \pm 0.01$

Cross verification for Frame size of 72 Bytes:

$$ExperimentalOneWayLatency_{72} = LoopbackLatency_{72} = \frac{\Delta RoudTripTime}{2} = \frac{Td_{a72} - Td_{b72}}{2}$$

$= 0.03 \pm 0.02ms$
$= 0.01\ ms\ to\ 0.05ms$

$Simulated\ Loopback\ Latency_{72} = 0.015ms$

Cross verification for Frame size of 1526 Bytes:

$$ExperimentalOneWayLatency_{1526} = LoopbackLatency_{1526} = \frac{\Delta RoudTripTime}{2} = \frac{Td_{a72} - Td_{b72}}{2}$$

$= 0.245 \pm 0.02ms$
$= 0.225\ ms\ to\ 0.265ms$

$Simulated\ Loopback\ Latency_{1526} = 0.250\ ms$

The cross verification shows a positive result because simulated latency for 72 bytes frame size of 0.015 ms is within the range 0.01 ms to 0.05 ms of experimental latency. Simulated latency for a 1526 byte frame size, 0.250 ms, is also within the range 0.225 ms to 0.265 ms of experimental latency.

Jitter measurements were also analysed which shows that the Byte Stuffing encoding and decoding used by the MWS Subscriber Interface introduces very little jitter. Analysing the jitter bounds results from experiments reveals that a significant portion of jitter was attributed to the test set up in experiment (a). The jitter was most likely caused by the

hosts since the jitter introduced by Ethernet cabling is expected to be near zero. The jitter results isolated to the MWS Subscriber interfaces was found to be the least at only 3.8 microseconds. It should however be noted that jitter introduced by buffering in the MWS Subscriber Interface is not present since the measurements were conducted below the saturation level. Quantified values of jitter introduced by each subsystem are summarised in the table below.

	Host	MWS Subscriber Interface	DVB-S modulator & set-top-box
Jitter (μs)	9.2	3.8	16.9

Table 4-7: Jitter results from experiments

The results from the experiment emphasize the advantageous characteristic of the MWS Subscriber Interface design with cumulative maximum jitter less than 30 μs and latency less than 1.0ms. These characteristics should be maintained as long as the MWS Interface operates below the saturation level (as defined subsection 4.3.5).

4.4.4. Throughput Experiments Methodology

Within the point-to-point MWS Experimentation platform, throughput stress testing was performed to obtain user throughput for the TCP/IP and UDP/IP protocol stacks. The results would give an indication of efficiency of the Byte Stuffing encoding and decoding employed by the MWS Subscriber Interface design, as well as compatibility with mechanisms of commonly used Transport layer protocols.

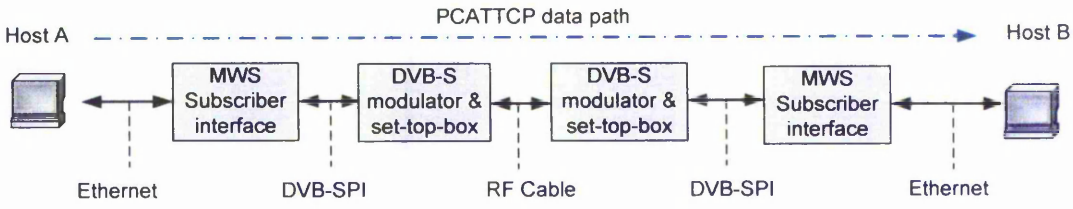


Figure 4-29: Experimental set-up for PCATTCP throughput measurements

The point-to-point MWS experimental platform was set up according to the diagram in Figure 4-29. Host A and Host B were booted up in Microsoft Windows XP operating system environment and PCATTCP was initialised on the command prompt. A simple command prompt based test application is ideal for such an experiment where insufficient processing power of Host processors is notorious to skew measurement results. Using available facilities, Host A was a 600MHz Intel Celeron desktop while Host B, a 600MHz Intel Pentium III laptop. Since the host's processing power is just sufficient for this experiment the processor utilisation was constantly monitored to ensure that experiments were not conducted with processor utilisation above 70%.

Host A is configured to be the data source while Host B is the sink. Throughput measurements are collected in Host B. In total four experiments were conducted by varying the transport protocol (TCP or UDP); and enabling or disabling Ethernet flow control. To aid clarity in the following discussions, the experiments are assigned indices 1 to 4 as shown in the table below:

	Ethernet Flow control Enabled	Ethernet Flow control Disabled
TCP/IP	experiment 1	experiment 2
UDP/IP	experiment 3	experiment 4

Table 4-8: Throughput Experiment Indices

Whilst it is possible to tune parameters of TCP and UDP, or use parallel streaming techniques, to improve throughput and link utilization in most circumstances [150], no such optimization was attempted here because of the intention to investigate a typical scenario where the protocol settings are left at operating system defaults.

Through comparative analysis presented in the two following subsections, it will be shown that the MWS Subscriber Interface is able to achieve 94% utilisation of theoretical available capacity without frame loss. This is attributed to sufficient buffering

and an Ethernet flow control mechanism which was extremely effective in negotiating data rate to a fine granularity.

To provide a comparison, the same experiments were run between Host PCs but the MWS link set-up shown in Figure 4-29 was replaced by either a 100 Mbps straight Ethernet or a 100 Mbps Ethernet switch. For straight Ethernet, a measured throughput of 88.46 Mbps over TCP was observed. Through the Ethernet switch, TCP throughput of 87.98 Mbps was observed. Hence the utilisation of straight Ethernet and Ethernet switch are about 93% of theoretical available capacity and without frame loss. The detailed calculations are shown below:

$$Utilisation = \frac{MeasuredTCPthroughput}{TheoraticalThroughput} \times 100\% \qquad \text{Eq. 4-13}$$

For 100 Mbps straight Ethernet or Ethernet Hub,

$$TheoreticalThroughput = \frac{1526}{1526+24}. \times TcpCoefficient \times 100Mbps$$
$$= 95Mbps$$

Note $TcpCoefficient = 0.965$, as shown in Eq. 4-14.

Therefore, utilisation using 100 Mbps straight Ethernet is:

$$Utilisation = \frac{88.46}{95} \times 100\% = 93.1\%$$

And utilisation using 100 Mbps Ethernet switch is:

$$Utilisation = \frac{87.98}{95} \times 100\% = 92.6\%$$

4.4.5. TCP/IP Throughput Experimentation and Results

In experiment 1, the PCATTCP tool was used to send 500Mb of data using the TCP/IP protocol from Host A to Host B. A measurement trace showing raw Ethernet throughput (i.e. including Ethernet headers) from the Network Trace tool for this experiment is shown in Figure 4-30:

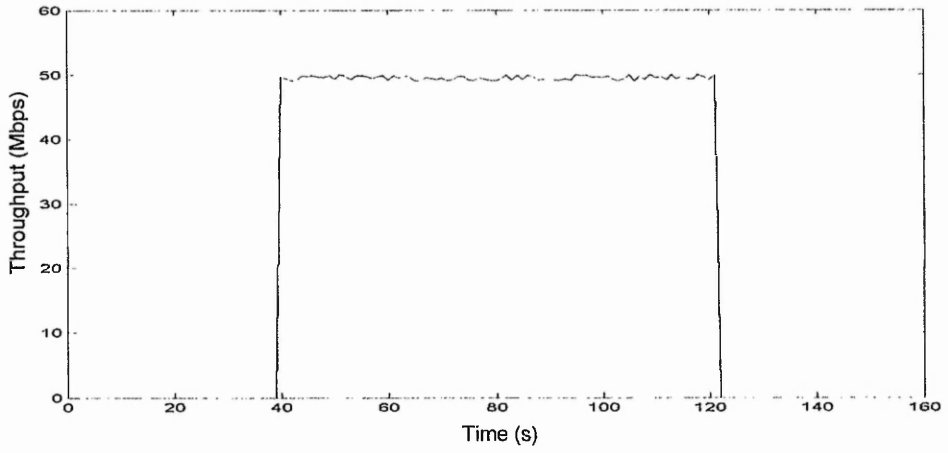


Figure 4-30: Experiment 1 Trace - TCP with Ethernet flow control

The average raw user throughput measured for TCP/IP protocol is 45.95 Mbps using maximum sized Ethernet frames. Raw user throughput for TCP/IP means overhead introduced by TCP and IPv4 headers were not included in the throughput calculation. The theoretical raw user throughput for TCP/IP for the MWS Subscriber Interface prototype can be calculated as follows for a comparison:

The Protocol overhead coefficient of TCP/IP for maximum sized frames are [151]:

IP version 4 headers = 20 *Bytes per frame*,

TCP headers with time stamps = 32 *Bytes per frame*,

$$TcpIpcoeffnt = \frac{1500 - IPv4Headers - TCPheaders}{1500} = 0.965 \qquad \text{Eq. 4-14}$$

The capacity available to Ethernet for maximum sized frames is 50.48 Mbps. Therefore, the theoretical raw user throughput for the 56 Mbps MWS Subscriber Interface prototype using TCP/IP is:

$$TheoreticalThroughput = MaxEthCapacity \times TcpIpcoeffnt$$
$$= 50.48 \times 0.965$$
$$= 48.71 Mbps$$

$$Measured\ Throughput = 45.95\ Mbps$$

Therefore, it is shown that the TCP/IP protocol achieved a utilization of 94% (i.e. $\frac{45.95}{48.71} \times 100\%$) of the available theoretical capacity of the point-to-point MWS link.

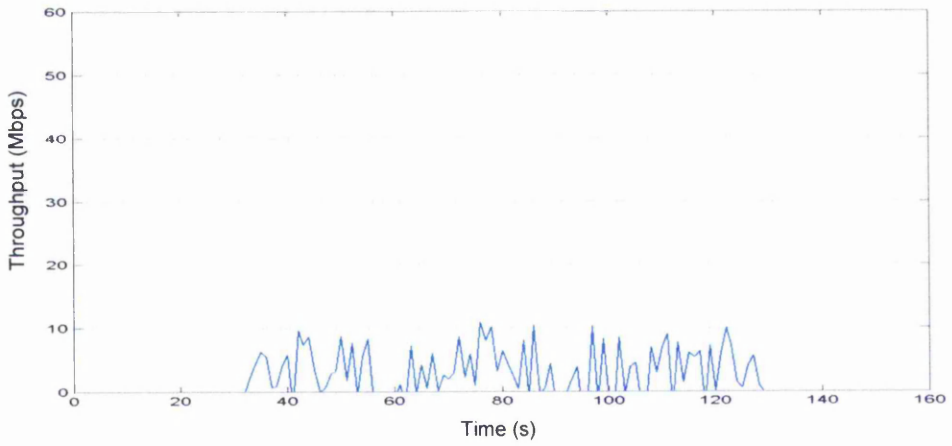


Figure 4-31: Experiment 2 Trace - TCP with Disabled Ethernet flow control

When Ethernet flow control is disabled, the TCP/IP throughput suffers drastically as can be seen in Figure 4-31. Here, the average raw user throughput measured is only 1.59 Mbps, or 3.3% utilization of the available capacity. Poor throughput performance is caused by frame losses due to buffer overflow at the 'Ethernet to DVB/MPEG-TS

Encoder'. The buffer overflow event was confirmed by the Debug Module (see 4.2.4) which indicated overflow of the SPI Output buffer by switching 'on' L.E.D 10. Of course, since TCP/IP is a reliable transport mechanism, no packet losses were detected and the 20 Mb of data sent from Host A was successfully received by Host B in 100.6 seconds.

In a possible explanation, the poor throughput was probably further worsened by the 'exponential cut back mechanism' of TCP [152]. According to workers in the field [153,154], the TCP will detect the loss and go back into slow start phase while reducing its window size by half. But because the total capacity of the MPEG-TS output buffer on the 'Ethernet to DVB/MPEG-TS Encoder' is only 4192 bytes deep, the TCP protocol window size would not expand beyond 3 packets. Theoretically, in order for TCP/IP to achieve an improved throughput, more buffer space would be needed. Workers in the field suggest that the buffer size should to be at least equal to the bandwidth-delay product ($50Mbps \times 1.1ms$) or about 5 packets deep in this case. However, the increase in buffer size would also result in an increase in latency. Hence, the solution to use Ethernet flow control is indeed better as it can provide low latency as well as high utilization.

4.4.6. UDP/IP Throughput Experimentation Results

Experiment 3 was carried out using PCATTCP with UDP option and enabling Ethernet flow control. 500 Mb of data was sent from Host A to Host B. Statistics for throughput received by Host B is recorded. The trace of this experiment as measured by the Network Trace tool is shown in Figure 4-32. Note that the Network Trace tool measures Ethernet throughput including headers.

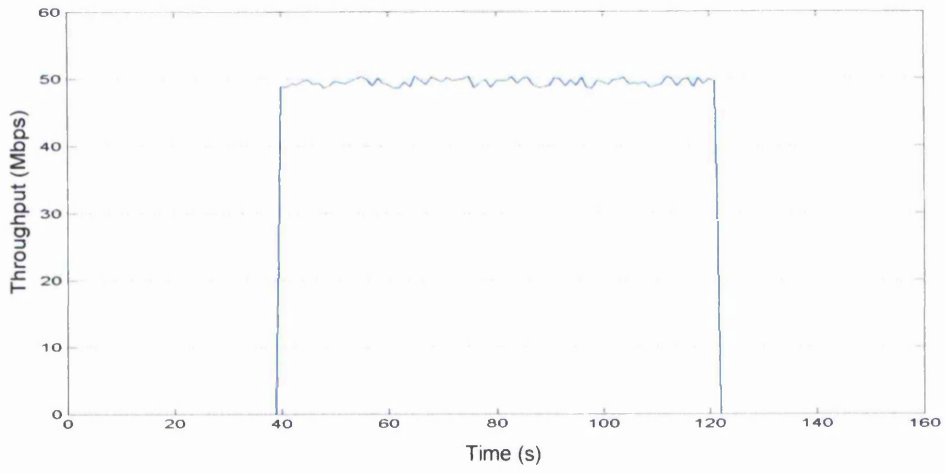


Figure 4-32: Experiment 3 Trace - UDP with Ethernet flow control

In PCATTCP tool, an average user throughput of 46.66 Mbps was measured for Experiment 3. This result is compared with the *theoretical* maximum throughput to obtain the utilisation efficiency. Taking into consideration overhead introduced by UDP and IPv4 headers, the theoretical maximum user throughput using UDP/IP over Ethernet over point-to-point MWS can be calculated as follows:

Given,

IP version 4 headers = 20 *Bytes per frame*,

$UDP\ headers = 8$ *Bytes per frame*,

$$UdpIpcoeffnt = \frac{1500 - IPv4Headers - UDPheaders}{1500}$$

$$= 0.981$$

Therefore the theoretical raw user throughput for the MWS Subscriber Interface prototype using UDP/IP is,

From Eq. 4-9 on page 145,

$MaxEthCapacity = Sim4EthThroughput = 50.48 Mbps$

Therefore,

$$TheoreticalThroughput = MaxEthCapacity \times UdpIpcoeffnt$$
$$= 50.48 \times 0.981$$
$$= 49.52 Mbps$$

$$Experimental\ Throughput = 46.66\ Mbps$$

It is shown that the PCATTCP test using UDP/IP protocol and Ethernet flow control mechanisms achieved a utilization of 94% (i.e. $\frac{46.66}{49.52} \times 100\%$) of the available capacity of the point-to-point MWS link. Surprisingly, the TCP/IP protocol and UDP/IP protocol stacks achieved the same utilization result of 94%. However, since UDP has less headers, it achieved an additional 0.71Mbps (i.e. 46.66Mbps - 45.95Mbps) of raw user throughput compared to TCP.

In experiment 4, PCATTCP over UDP throughput tests were run. However, besides an initial burst of packets, the test tool failed to produce a sustained flow. It is assumed that the proprietary UDP congestion control mechanism implemented by the PCATTCP tool had failed. The failure can be attributed to high packet losses which were caused by buffer overflow of the SPI Output buffer as indicated by the Debug Module. Nevertheless, to obtain results for experiment 4, another tool called IPERF [155] was used to generate a constant UDP load of 50Mbps. Throughput trace of the experiment using IPREF to send 500Mbytes of user data is shown in Figure 4-33 below.

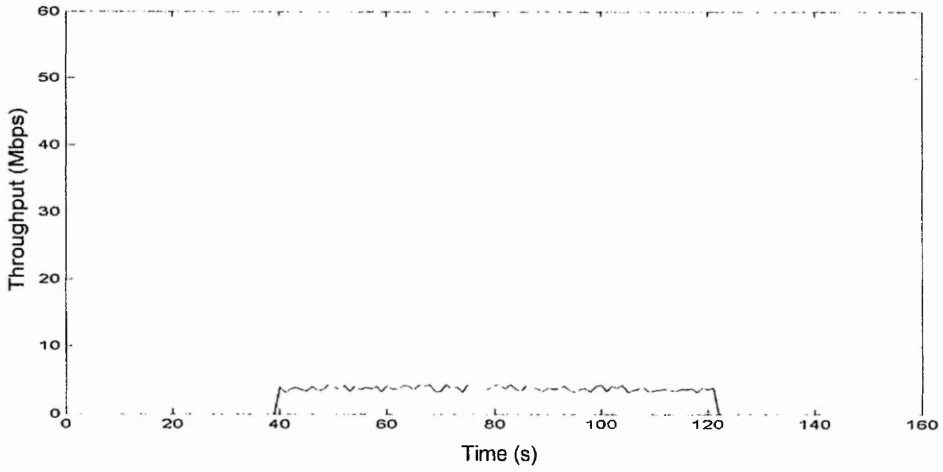


Figure 4-33: Experiment 4 Trace - UDP with Disabled Ethernet flow control

As can be seen from the trace, the majority of the packets were lost or in error due to buffer overflow. Unlike TCP, UDP is an unreliable transport protocol and does not have mechanism to retransmit lost packets. The packet loss statistic is obtained by comparing the number of UDP messages transmitted by Host A against those received by Host B. The observed statistics were 340137 packets sent and 319728 lost. Hence, without Ethernet flow control, the UDP only managed to utilise 6% of the available capacity provided by the point-to-point MWS link. The average throughput reported by IPERF of 3.13 Mbps tallies with this result.

However, when the results of experiment 4 are compared with experiment 2, it is realised that UDP actually performs better than TCP when Ethernet flow control is not present. This comparison provides further evidence that attempted congestion control by TCP had worsened the throughput. The amount of impact can be quantified to be 51% $(\frac{1.59}{3.13}\times 100\%)$ of further throughput degration.

Summary of Section 4.4

This section detailed the construction of two 56 Mbps MWS Subscriber Interface prototypes as well as the development and calibration of a point-to-point MWS experimentation platform. The primary results obtained through experimentation and analysis done using the developed hardware can be summarised as follows:

- The calibration of the point-to-point MWS experimentation platform to measure round trip time up to 10 microsecond accuracy.
- The cumulative maximum end-to-end latency performance below saturation level of a point-to-point MWS was determined to be less than 1.0ms or less than 1.1 ms above saturation level.
- The cumulative maximum end-to-end jitter performance below saturation level of a point-to-point MWS was determined to be less than 30 μs.
- Practical utilization of 94% achieved for both unoptimised TCP/IP and UDP/IP stacks were determined with throughput of *45.95 Mbps* and *46.66Mbps* respectively.

Chapter 4 conclusions

The design of an MWS Subscriber Interface incorporating the Byte Stuffing encoder/decoders was described in industry standard VHDL. While using VHDL allowed vendor independence, generic parameters integrated into the MWS Subscriber Interface intellectual property further increases its value by enabling a high degree of flexibility and configurability. For these reasons, the design stage was considered successful which resulted in high quality source code.

To consolidate the findings of simulation and experimental work done in this chapter, the performance latency results from simulations (section 4.3) and experiments (section 4.4) are combined to obtain a 'performance profile' plot of the MWS Sub Interface. Using the performance profile plot, an accurate estimate of the expected end-to-end latency can be derived from the current utilisation level and size of the frame being sent. The performance profile of a 56 Mbps point-to-point MWS is shown in Figure 4-34.

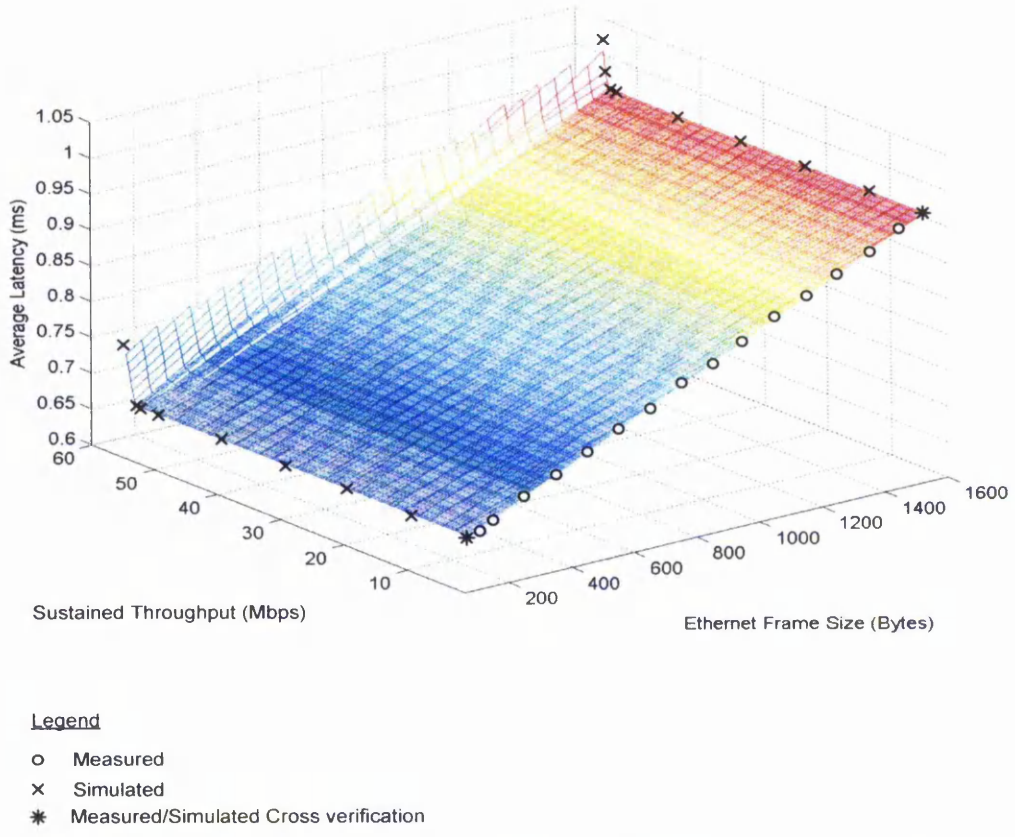


Figure 4-34: Performance profile Plot of MWS Subscriber Interface

Positive results from cross verification work between hardware tests and simulation provided evidence for: the integrity and precision of the design and prototyping flow; successful prototyping on FPGA and the validity of the VHDL test-bench simulation models. In conclusion, it is found that the close integration of the MWS Interfaces with Ethernet technology combined with the Byte Stuffing encapsulation enables:

- Low latency, as layer-2 Ethernet forwarding and Byte Stuffing allowed cut-through forwarding at the MWS Nodes.
- Simultaneous services of MPEG broadcast and Ethernet access, as the proposed integration of the MWS system into Ethernet and DVB frameworks was efficient.
- High utilisation of available MWS capacity, as the Ethernet flow control mechanisms of the MWS interface worked extremely well with default TCP/IP and UDP/IP stacks implemented in the Host PCs.

5 ETHERNET-HUB-EMULATION BASE STATION FOR MWS

"The basic communication functionality is provided by protocol layers 1 and 2; since all stations connected to the broadcast channel can receive directly from any other station. Hence, there is no true layer 3 protocol necessary..." [156].

The Ethernet-hub-emulation (EHE) Multimedia Wireless System (MWS) is a novel low-cost layer-2 approach to the IP routing problem caused by the point-to-multipoint topology of MWS systems. Most previous systems had either adopted an IP layer approach or ATM approach to the problem. These solutions often required the use of manually defined routes, either by creating static IP subnets or ATM virtual channels (VC), to a known router. Whilst these methods are sufficient for small scale demonstration networks, the administrative complexity of manual IP address management and the creation of many IP subnets can become impractical and wasteful when the number of subscriber nodes is increased. Further, as these solutions depend on end-to-end management at the IP layer or ATM virtual channels, more sophisticated layer-3 equipment would be required at each subscriber node which can significantly increase the cost of the overall network. Since the Ethernet infrastructure is sufficient to sustain a network the size of an MWS (about one thousand nodes [97]), the Ethernet-hub-emulation (EHE) MWS layer-2 approach to the problem was explored to diminish the need of static IP route management and layer-3 hardware. The results from design, simulation, synthesis and hardware prototypes show that the EHE MWS was indeed an administratively flexible system, and can be implemented in low-cost FPGA with faster operational speed and lower latency than the equipment employed in previous BFWA demonstrators.

5.1. Ethernet-Hub-Emulation MWS Architecture

5.1.1. Motivations and aims for EHE MWS

The motivation of the EHE MWS was to investigate a solution to the IP routing problems caused by the MWS topology. The MWS topology describes a broadcast downlink and point-to-point uplinks which are caused by the directional subscriber antennas used in these systems. This topology has been found to induce IP routing problems in previous BFWA trials [81, 88 ,89, 94] which is caused by the lack of direct communication (layer-2 path) between subscriber nodes.

In view of an IP network, the "remote versus local assumption" of the IP protocol will be invalid because basic layer-2 connectivity are only present between base station and subscriber nodes, and not between any two subscriber nodes [157]. The routing issues associated with NBMA media that will be discussed in the following paragraphs are adapted from a reference book [157]. IP routing is based on the 'local versus remote assumption': if the destination IP address is outside the source host's own subnet, i.e. different subnet ID prefix, the packet will be forwarded by the default router. In a normal network where there is full layer-2 connectivity, packets that are destined to a host in the same IP subnet can reach the correct destination without going through the router. However, NBMA medium violates the basic IP assumption of full Layer-2 connectivity within a subnet (Figure 5-1-a). The IP based solution to this is shown in Figure 5-1-b where each host on the NBMA medium is allocated as an IP subnet.

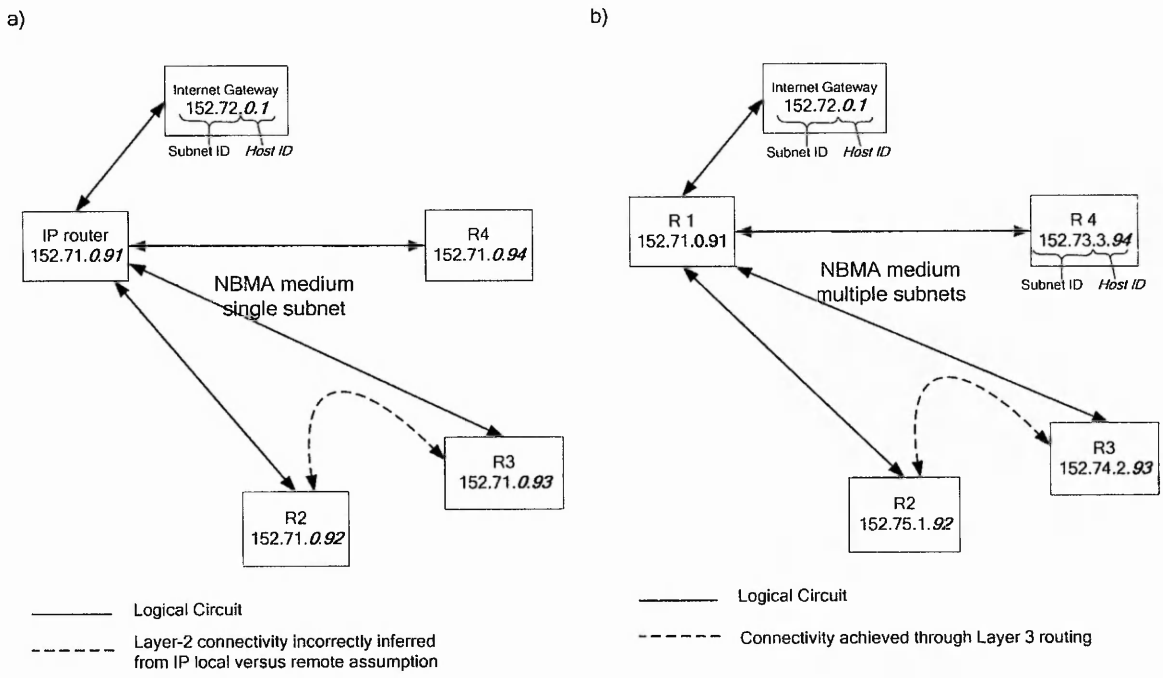


Figure 5-1: IP routing issues over Non-Broadcast Multiple Access (NBMA) medium

The survey done in section 2.3 identified a number of recent BFWA systems that have novel solutions to the IP routing problem. However, it is viewed that these systems have drawbacks of inflexibility and/or high equipment costs in the context of addressing the IP routing problem. The table below compares flexibility and equipment costs of the systems that are involved in overcoming the IP routing problem. Flexibility of each system is measured here by the number of mandatory IP subnets that have to be created in order for IP routing to work. Cost is measured by the mandatory equipment to maintain the IP subnets and encapsulate IP traffic. While it is understood that the inflexibility and equipment costs might be intrinsically linked to other mechanisms required in the systems, in most cases the potential to replace entire PC workstations with low-cost FPGAs is substantial.

BFWA System	IP Subnets	Equipments
CRABS	3+N	IP to DVB Gateway workstation Uplink Router workstation, N x Subscriber routers.
NTU Trial	2	Cable Headend INA, N x Subscriber cable Modems.
AT&T LABS	2	Cable Headend INA, N x Subscriber cable Modems
Cambridge Trial	2	ATM base station Router, N x Subscriber ATM Routers.
EMBRACE	2	MPLS Router workstation, N x MPLS Router workstation,
EHE MWS	1	Base Station FPGA, N x Subscriber FPGAs

Table 5-1: Flexibility and cost comparison of BFWA systems

The Ethernet-hub-emulation (EHE) MWS architecture disguises an MWS to mimic the functional operation of an Ethernet internetworking device by internally forwarding and filtering frames. To an external network entity, all networks connected to the MWS, i.e base-station and subscriber premises LANs will appear to have full-layer-2 connectivity between them. As full layer-2 connectivity is achieved, the "remote versus local assumption" rule will be satisfied and IP routers can establish routes as normal. The forwarding and filtering mechanisms within EHE MWS are detailed in the next subsection, 5.1.2.

Unlike other BFWA architectures, an EHE MWS deployment does not require sophisticated layer-3 Base station and subscriber node premises equipment such as MPLS routers, Interactive Network Adaptor (INA) or specialized IP routers. Instead the functions of these internetworking devices are performed by layer-2 Base station and subscriber interface designs which were synthesized on low-cost FPGA. The reduction of some layer-2 functions and elimination of layer-3 can substantially reduce equipment costs. Moreover, the EHE MWS layer-2 solution also enables the MWS network to be more flexibly managed as layer-3 IP routers are not required to be placed at specific points in the system. For example, all hosts on subscriber premises and Base station premises LANs can be addressed and managed by a single IP subnet since the EHE MWS combines all LANs connected to it to form a larger layer-2 network. The designs and synthesis of the low-cost and administratively flexible layer-2 EHE MWS Base station are detailed in section 5.2.

Layer-2 emulation also enables protocol flexibility since network traffic is not restricted to only IP protocol. "Its advantages are that it is agnostic to layer 3, in that it can transport multiple protocols" [158]. Any type of protocol that works over Ethernet can be forwarded by the EHE MWS. There are many protocols such as Microsoft Browser protocol, AppleTalk, and CISCO switching protocols that are supported by Ethernet but are not IP. These protocols are traditionally used by services within a LAN. However, forwarding such traffic can allow services such as, network printing, network directory file sharing and remote desktop to span between all LANs connected to the MWS. Such LAN services might appeal to MWS deployments in a private but geographically large network such as a university campus [4]. In an IP based solution, such services usually require a Virtual Private Network (VPN) connection or a tunnelling approach to be set up before they can be used [159]. The flexibility of protocol support for both Web-based and LAN-based applications was demonstrated in experiments conducted on a prototype EHE MWS detailed in section 5.3.

5.1.2. EHE MWS Forwarding and Filtering Mechanisms

The EHE MWS performs full layer-2 forwarding between all Ethernets connected to base station and subscriber nodes. The layer-3 routing and layer-2 forwarding essentially perform the same service level task, which is to ensure frames from the source host will eventually reach their intended destination(s). However, the fundamental difference is that, while IP routing tells each frame where to go, Ethernet forwarding tells each frame where not to go. Figure 5-2 below illustrates the forwarding and filtering mechanisms within subscriber nodes and base station. In this abstracted model, other functions besides the SID tagging and filtering are not shown in the Base station and subscriber nodes since the flow of data is of importance.

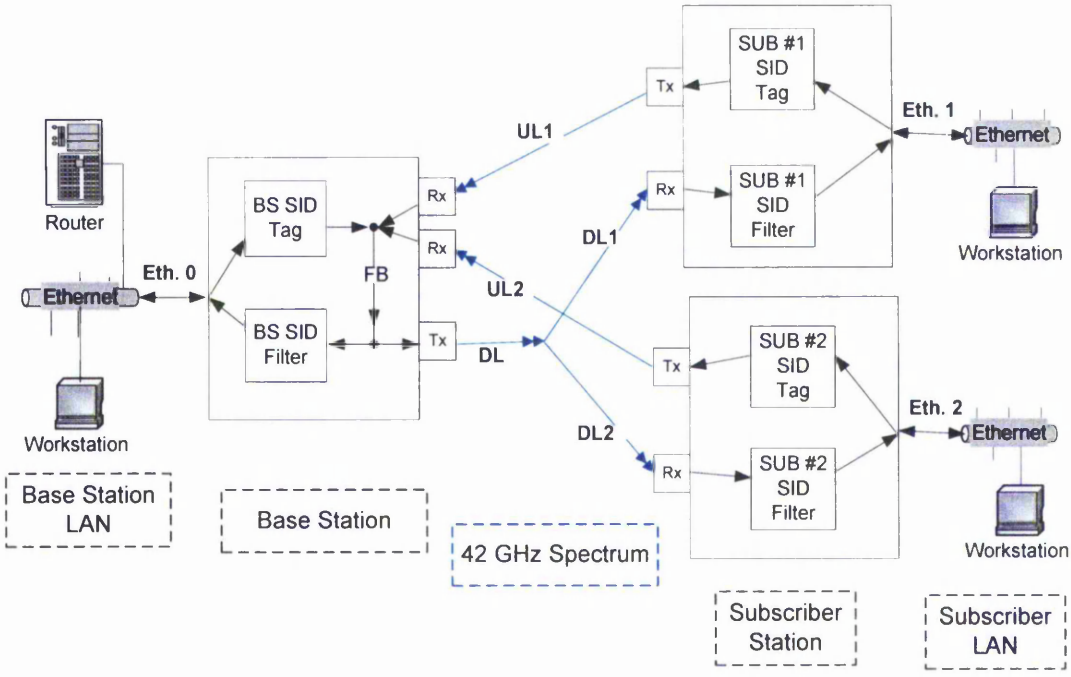


Figure 5-2: Data flow in Ethernet-hub-Emulation architecture

Full-layer-2 connectivity can be established by Ethernet hubs that apply the basic rule to "forward a frame to all ports except the port which it came from". The base station and subscriber nodes will cooperatively process Ethernet frames entering the MWS on any of the Ethernet ports to achieve this basic rule. To aid understanding, the basic rule can be restated as two rules that apply to an MWS:

- Rule 1: Ethernet frames entering from the base station's Ethernet port are forwarded to all subscriber node's Ethernet ports.
- Rule 2: Ethernet frames entering a Subscriber node are forwarded to base station's and all other subscriber node's Ethernet ports.

An addressing mechanism, termed the Station Identity (SID), is introduced in the EHE MWS. The SID tagged Ethernet format only exists internally to the MWS system and is transparent to the networks it is interfaced to. An SID address adds exactly one byte of overhead per Ethernet frame. Consequently, the number of possible unique SIDs are two hundred and fifty six (N = 256). This limits the maximum number of subscriber stations to two hundred and fifty five (N-1 = 255) per MWS sector, with one SID reserved for the base station. While the size of the SID address space could be increased, the number of

subscribers addressable by a one byte SID is considered to be sufficient since it is equal to that of HIPERACCESS [65]. Whilst the IEEE suggests no more than 1024 nodes should be placed in a single Ethernet network, there are no technical limitations in the Ethernet technology which explicitly restricts the number of nodes on a network [97].

Referring to Figure 5-2, function of the 'SID tag' module is to attach a SID to each Ethernet frame entering the MWS while the 'SID filter' module prevents frames that have the same SID as itself from exiting the MWS. The MWS satisfies Rule 1 by ensuring that Ethernet frames from the Base station Ethernet, Eth.0, are forwarded only to the subscriber node's Ethernets, Eth.1 and Eth.2. Ethernet frames entering the Base station Ethernet port, Eth.0, will be tagged with the base station SID. The frames are forwarded by the internal fabric bus, FB, to both the base station SID filter module, and the SPI output interface, Tx. The base station SID filter will discard these frames since they possess the base station SID. The copies of Ethernet frames at the SPI output interface are eventually broadcast on the downlink path, DL. The subscriber node will receive the frame on their SPI input interfaces, Rx. The SID filters of the subscriber nodes, SUB1 SID filter and SUB2 SID filter modules detect that the frames originated from the Base station based on their SID tags and forward the frames to the subscriber node's LANs.

In a similar manner, Rule 2 will be satisfied by ensuring that frames from a subscriber Ethernet port are only forwarded to the Base Station and other subscriber Ethernet ports. An Ethernet frame entering from a subscriber node, say SUB 1, will follow the path from its' Ethernet port; tagged by SUB1 tag module, forwarded via the uplink UL1 and received by the Base station's SPI input, Rx. The frame will flow to the Base station internal fabric bus and be copied to both the downlink interface and the HUB SID filter modules. Since the frame is tagged with the SUB1 SID, the HUB SID filter will allow the frame to pass and eventually forward it to the Base station's LAN. The copy forwarded by the downlink will reach both subscriber nodes SUB1 and SUB2. While SUB2's SID filter will let the frame pass and forward it to its own LAN, SUB1 SID filter will discard the frame since it has a SID 1 tag on it.

The Ethernet hub emulated by the EHE MWS is compared to the two kinds of standard Ethernet hub devices which are repeaters and multiport bridges as explained in

subsection 2.4.3. However, the Ethernet hub emulated by EHE MWS has unique traits and cannot be categorised in either types. Table 5-2 below summarises these differences.

Device	Ethernet switching	Single speed technology	Full-Duplex capability	Half-Duplex capability	Store-and-forward latency
Repeater	No	Yes	No	Yes	0
Multiport bridge	Yes	No	Yes	Yes	0 or 1
EHE MWS	No	Yes	Yes	No	1 or 2

Table 5-2: Comparison of EHE MWS properties with standard Ethernet hubs

In the Store-and-forward latency column, a value of 1 is given in to represent one Ethernet frame delay while a value of 0 represents no-delay cut-through-forwarding capabilities. The EHE MWS has 1 equivalent store-and-forward delay between Base station and subscriber nodes, and 2 equivalent store-and-forward delays between two subscriber nodes. In an EHE MWS, these store-and-forward delays are not introduced by layer-2 conditioning performed in Ethernet bridges, but are due to the 'DVB to Ethernet' (decoding) conversion processes. As the EHE MWS employs OBS encapsulation, 'Ethernet to DVB' (encoding) conversion does not incur store-and-forward delay. While the EHE MWS does incur more store-and-forward delay compared to standard Ethernet hubs, the delay is an improvement over previous BFWA systems where FLM based encapsulation is employed. Since FLM based encapsulation will incur store-and-forward latency for both its encapsulation and de-capsulation processes, a delay of 2 is incurred between Base station and subscriber nodes, and 4 between two subscriber nodes.

5.2. Design and Synthesis of EHE MWS Base-Station

The Design and synthesis of the EHE MWS Base station show that the electronics required for layer-2 frame forwarding can be implemented in low-cost FPGAs, and have operation speeds that exceed the maximum bit rates of commercial DVB-S equipment. Details of two Base Station designs of are presented in this section. The first design investigated an integrated solution whereby the EHE MWS Base station was successfully synthesized into one Field Programmable Gate Array (FPGA) chip. However, synthesis results of this design revealed that the hardware resource

requirement for this implementation was too high for a prototype to be realised on the FPGA facilities available. Hence, the second Base Station design was investigated that allowed the base station electronics to be partitioned into two FPGAs without compromising performance. This enabled a hardware prototype of a complete EHE MWS to be built.

5.2.1. Design considerations for Base Station Integration with DVB framework

In a usual MWS sector deployment, the role of the Base station is to manage and provide core services to the subscriber nodes connected to it. Since the EHE MWS is based on the DVB infrastructure, MPEG-TS broadcast based digital television services are inherently supported. When an EHE MWS is deployed with simultaneous digital television and Ethernet data services, the digital television and Ethernet data subsystems will have to converge at the protocol and physical level at some point to share the MWS infrastructure. The integration of the proposed EHE MWS base station to the DVB framework was considered carefully so that the new Ethernet services was not only compatible, but also did not impose heavy restrictions to the widely adopted, standardised MPEG-TS digital television service. A possible integration of EHE Base Station in the DVB framework is shown in Figure 5-3 below:

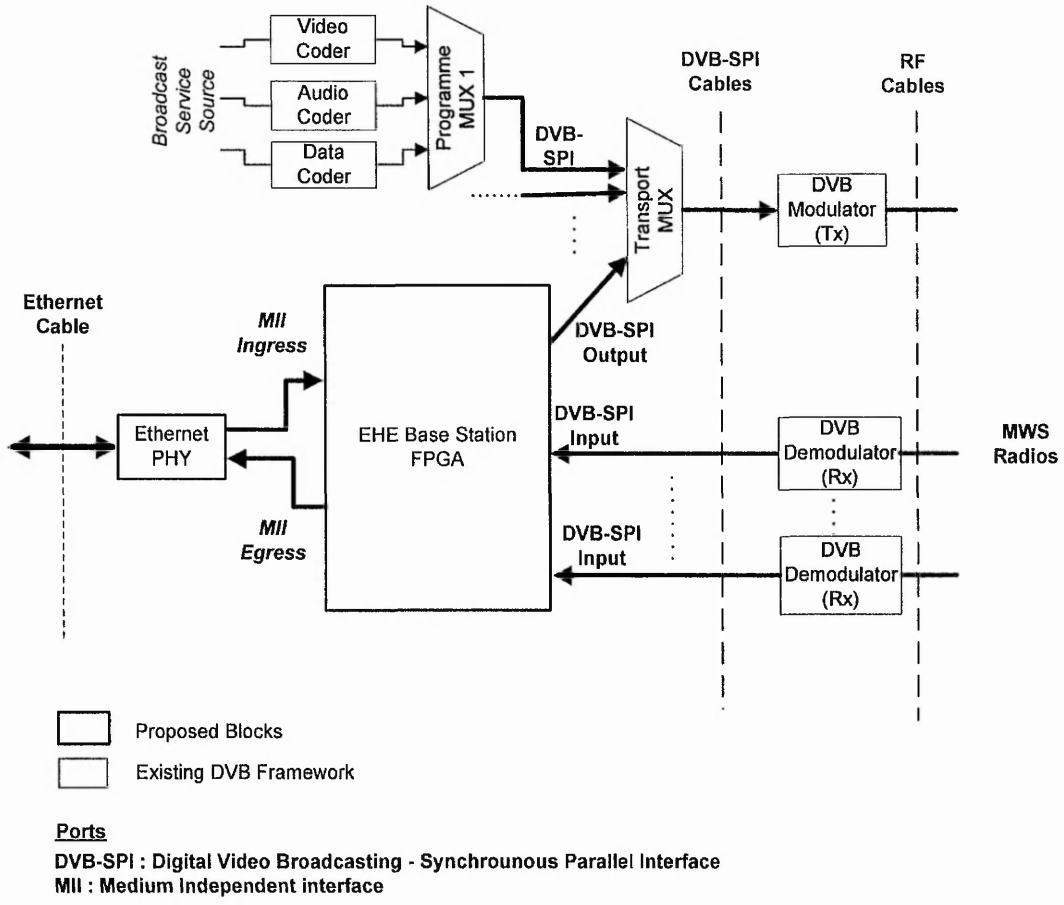


Figure 5-3: Possible integration of EHE Base Station in DVB Framework

At the protocol level, the EHE MWS requires the registration of only one PID address in a MPEG-TS multiplex. In the DVB framework, each MPEG program stream in a multiplex is allocated a unique PID address which serves as a virtual television channel. The audio/video decoders in a DVB-S set-top-box can tune in to and display digital television channels for which the customer has paid subscription for. In the EHE MWS, the registration of one PID address would ensure that the DVB-S set-top-box would not attempt to decode and display the proprietarily encoded MPEG-TS. For experimental purposes the EHE MWS uses the 0x1FFF PID, which is safely ignored by the DVB-S audio/video decoders, since this PID is already reserved as 'test stream' in the DVB frame work.

At the physical level, transport multiplexer equipment can be used to combine MPEG-TS streams from DVB MPEG-TS service sources and the EHE MWS DVB-SPI output to form a MPEG-TS multiplex. Transport multiplexer equipment was not available to

this research project to demonstrate simultaneous digital television and Ethernet data services. However, by manually swapping between the EHE MWS DVB-SPI output and an MPEG-TS digital television multiplex directly to the DVB Modulator, both services were successfully demonstrated individually.

In the MWS investigated, the radio side physical layers use standard DVB-S broadcast equipment for both its downlink and uplink paths. Although the previous Nottingham Trent University (NTU) MWS trial used DVB-C equipment, it was learnt that the DVB-C modems was not suitable for reliable long term use as they could not tolerate the frequency drift due to aging of the crystals used to generated 42 GHz carrier signals. Similar limitations of DVB-C equipments for use in BFWA systems were also reported in the AT&T trial system [84]. DVB-S equipment, on the other hand, is more tolerant to the frequency drift. The stability of DVB-S satellite has been demonstrated in a long term measurement campaign conducted as part of the NTU MWS trial [82].

The use of DVB-S equipment also allows the maximum raw data rate to be increased to 56 Mbps from the 13 Mbps limitation of the DVB-C equipment. The DVB-S equipment used for downlink and uplink paths in the proposed MWS was actually meant for the broadcast downlink path in a satellite system. Therefore, the MWS in this research was restricted to a Frequency Division Multiple Access (FDMA) return path technique, which can be constructed with the available DVB-S equipment. Other return path techniques that were used in previous point-to-multipoint BFWA systems include: Time Division Multiple Access (TDMA) or a combination of FDMA and TDMA - Multiple Frequency Time Division Multiple Access (MF-TDMA). However, the MWS topology is still inherent to such systems where the benefits of the EHE MWS can still be applied.

5.2.2. Base Station Design 1

Base Station Design 1 is a single core solution whereby the EHE MWS Base Station hardware is implemented on one FPGA. In total, the Base Station has exactly one Ethernet port, one DVB-SPI output port, and N number of DVB-SPI input ports to connect to DVB and Ethernet networks. N is the number of DVB-SPI input ports installed on the MWS Base Station. In view of constructing an MWS demonstrator as proof of concept, the design of the MWS Base station is fitted with two DVB-SPI input

interfaces (i.e. N=2), which enables the Base Station to support up to that number of subscriber stations. As with all designs and prototyping in this research, the MWS Base Station follows the VHDL design flow as described in subsection 3.3.1. The top-level block diagram of Base Station Implementation 1 is detailed in Figure 5-4 below:

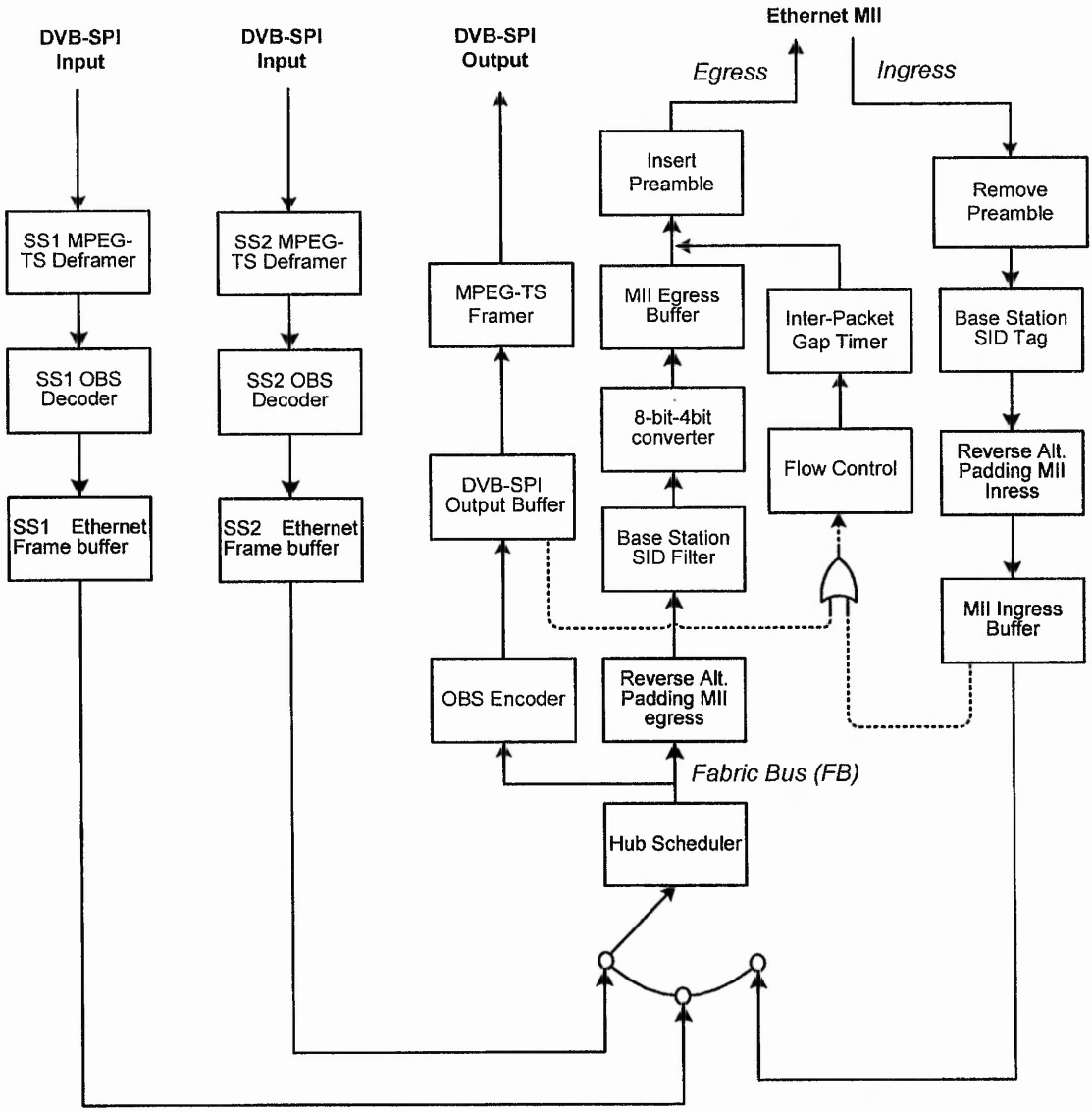


Figure 5-4: Base Station Design 1 Block Diagram

While the SID based forwarding operations of the Base Station have been already explained in subsection 5.1.2; scheduling, buffering and bus implementations that aims to maximize downlink capacity efficiency, and overall latency over a multi-clock domain environment are detailed here. At the Base Station, Ethernet frames can arrive

from subscriber nodes via DVB-SPI input ports, or a Base Station's LAN via the Ethernet MII Ingress port. Each port defines a clock domain which is not synchronised with each other. Another clock domain is defined by the internal Fabric bus that links all input and output ports. The buffers shown in the top-level block diagram are dual-clock First-In-First-Out (FIFO) memories which are placed at boundaries of the clock domains to manage data flow from one clock domain to another.

Ethernet Frames arriving from the DVB-SPI inputs ports are received preformatted in MPEG-TS frames and encoded by OBS encapsulation. Hence, at the base station, they are processed by a dedicated 'MPEG-TS De-framer' and 'OBS Decoder' before they are stored in the 'Ethernet Frame Buffer'. Raw Ethernet frames from the MII ingress interface of the Ethernet port are processed by 'Preamble remover' and 'HUB SID tag' modules before they are stored in 'MII ingress buffer'. 'Ethernet frame buffers' and 'MII ingress buffer' store frames in the same intermediate format whereby the Ethernet frames are SID tagged and are without preambles. However, since the MII port's data rate is always faster than the 'Fabric bus', the 'MII ingress buffer' implements cut-through forwarding which allows the 'Hub scheduler' to read a frame from it before the frame is fully stored. 'Ethernet frame buffers', on the other hand, implement a store-and-forward discipline since the DVB-SPI rate is slower than the 'Fabric bus'. Therefore, frames are fully stored in 'Ethernet frame buffers' before they are allowed to be forwarded by the 'Hub scheduler'. 'Ethernet frame buffers' and 'MII ingress buffer' are instantiations of a generic VHDL design detailed in Module 15, where the slight functional differences are selected using generic parameters.

Since there are two buffers ('MII ingress buffer' and 'DVB-SPI output buffer') in the path from MII ingress port to the DVB-SPI output, the occupancy of both of these buffers is monitored by the 'Flow control' module. The 'Flow control' module (detailed in subsection 4.2.2, Module 12) will generate flow control frames on the egress MII interface whenever either of the buffers reach their respective thresholds. A simulation based investigation into optimal threshold values for the buffers is detailed in section 5.3.

The Base Station design reused many modules initially designed for the Subscriber Interface. These modules are already explained in Sections 3.3 and 4.1. However, the new modules introduced in Base Station Design 1 which are the Hub Scheduler, Packet

Buffers, and Alternate-Pad De-formatting are explained at the operational level in the following paragraphs.

Module 15 Ethernet Frame / MII Ingress Buffers

Whilst MII Ingress Buffers and Ethernet Frame instantiations use the same underlying Altera Dual-Clock FIFO intellectual property core to instantiate FIFO memory, the status indicators of the Altera Dual-Clock FIFO were not suitable for the Ethernet Frame buffer application. The Ethernet Frame Buffer requires the capacity indicator to keep track of the buffer occupancy using frames as units. However, the Altera Dual-Clock FIFO core can only indicate its buffer occupancy as byte units. Hence, custom logic was designed around the Altera core to generate the appropriate signals. The simplified schematic diagram in Figure 5-5 below illustrates the additional blocks designed for the Ethernet frame buffer.

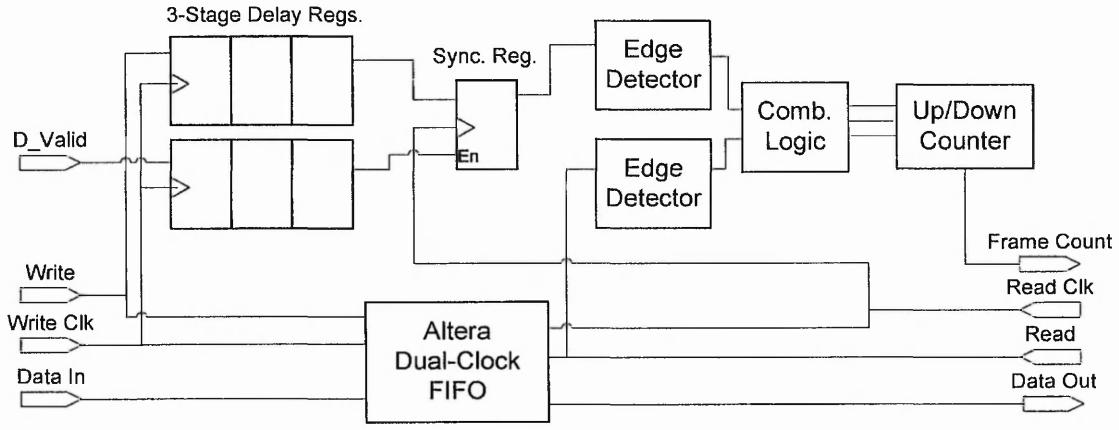


Figure 5-5: Ethernet frame buffer Block Diagram

The additional logic includes two Edge Detectors which generate a pulse whenever a frame enters or exits the Altera Dual-Clock FIFO. Combinatorial logic is used to resolve the pulses into appropriate signals to control an up/down counter which keeps track of the buffer occupancy as number of frames in the buffer. The additional logic in effect transforms a normal byte based FIFO to a frame based FIFO.

The most complex aspect of the Ethernet frame buffer design was to ensure synchronisation of the module that resides in two clock domains. The ingress side interface is in the DVB-SPI input clock domain, while the egress is on the fabric bus

clock domain. As can be seen in the simplified schematic diagram, three clock-cycle delay registers as well as a synchronisation register was utilised to ensure synchronisation between the two clock domains. Basically, the altera_Dualclock_FIFO core required some time to store and refresh its own status to be valid in both clock domains [144]. A three cycle delay was determined through simulation to be more than sufficient for the refresh to be performed. Hence, the potentially hazardous condition was successfully avoided whereby the Ethernet Frame buffer indicates the availability of a frame in memory before it is ready to be read.

Module 16 Hub Scheduler

The function of the Hub Scheduler Core is to control access to the fabric bus. The Hub Scheduler services the MII cut-through frame buffer, and all DVB-SPI Input frame buffers using a priority queuing discipline. Priority is assigned to Uplink ports (DVB-SPI inputs) and remainding bandwidth is given to the MII Ingress port. The Hub Scheduler resolves contention between the uplink ports by assigning a priority index starting from 1 to N subscriber interfaces. The Hub Scheduler concurrently determines (in one clock pulse) the availability of frames in each Ethernet Frame Buffer and services the buffer with the highest priority index first. MII Ingress port has a priority index of 0 and hence will be served only if no frames are waiting in any of the uplink ports. This is a simple scheduling mechanism, yet it enables the hub scheduler to efficiently use the downlink capacity by adapting to the varying traffic levels of the uplinks.

In the current stage of development of the Ethernet hub emulation MWS architecture, there is no mechanism for the Base Station to indicate to a subscriber node to slow down its uplink speed. Hence, the total uplink capacity of all subscriber nodes must not be greater than the downlink broadcast of the MWS or else there will be a possibility of frame losses due to overload. It is proposed that the total uplink capacity of all Subscriber nodes be about equal to half of the downlink capacity so that there will be a significant portion of downlink capacity reserved for data traffic from Ethernet MII ingress interface to provide Ethernet services. At any instant, the available Ethernet service bandwidth, provided via the Ethernet MII port can be expressed as:

$$E_t = DL - \sum UL_N$$

Where,

E_t is the Ethernet service bandwidth.

DL is the overall downlink capacity of the MWS.

UL_N is the instantaneous uplink utilisation of N number of subscriber nodes.

For example, if a Base Station has 50 Mbps downlink capacity and, at the instant of measurement, two subscribers utilise 3 Mbps and 5 Mbps respectively the available bandwidth for Ethernet services provided by the Base Station's LAN can be calculated as follows:

Given,

DL = 50 Mbps;

UL_1 = 3 Mbps;

UL_2 = 5 Mbps

Hence,

$$\begin{aligned} E_t &= DL - \sum UL_N \\ &= DL - (UL_1 + UL_2) \\ &= 50 \times 10^6 - (3 \times 10^6 + 5 \times 10^6) \\ &= 42\ Mbps \end{aligned}$$

Module 17 Reverse Alternate Padding

The Reverse Alternate Padding core translates frames from the 'alternate padding format, as described in subsection 4.2.1, to a normal (unpadded) format. This module is used in two places in the Base station; between 'HUB SID Tag' and 'MII Output Buffer' and between the 'Fabric Bus' and 'Hub SID filter'. Both 'HUB SID Tag' and 'Fabric Bus' generate output in alternate padded format, but the 'MII Output Buffer' and 'Hub SID filter' require frames in unpadded format. Hence the Reverse Alternate Padding core was utilised as a translator. There is a slight requirement difference in the translation

required by frames on the 'HUB SID Tag' and 'Fabric Bus'. The 'HUB SID Tag' module uses a pad-space when inserting the tag while 'Fabric Bus' does not. The Reverse Alternate Padding core has two architectures defined to compensate for the slight difference and the appropriate architecture can be bounded as required.

5.2.3. Base Station Design 2

The Base Station Design 2 is a dual FPGA solution. The primary reason for partitioning of Base Station 1 into two FPGAs was because Base Station Interface Implementation 1 was not able to fit into the a Cyclone FPGA device available to the research. However, the opportunity was taken to investigate a useful partitioning where fixed service ports i.e. DVB-SPI output and Ethernet interfaces are placed in one FPGA (Base Station Downlink FPGA), while DVB-SPI input ports on the other (Base Station Uplink FPGA). Partitioning the design in this manner can be advantageous to future developments when more DVB-SPI inputs are added to the system since the changes will be isolated to the Uplink controller FPGA. When the number of required supported nodes are increased, only the Uplink FPGA design needs to be upgraded. In contrast, if more DVB-SPI input ports are required in Base Station Design 1, the entire base station design has to be re-synthesized.

Except for the 'Hub Scheduler' module, Base Station Design 2 retains all the modules and functionality of Base Station Design 1. In Base Station Design 2, the 'Hub Scheduler module' is split into three cores. These are: 'Downlink Scheduler', 'Uplink Scheduler' and 'Fabric Bus Access Controller'. The 'Downlink Scheduler' and 'Fabric Bus Access Controller' are placed in Base station Downlink FPGA, while 'Uplink scheduler' is placed in the Base station Uplink FPGA. A twelve-pin external interface that physically links the two Base station FPGAs consist of signals required between the Uplink Scheduler and the Downlink Scheduler or Scheduler controller. A top-level block diagram of Base Station design 2 is illustrated in Figure 5-6 below:

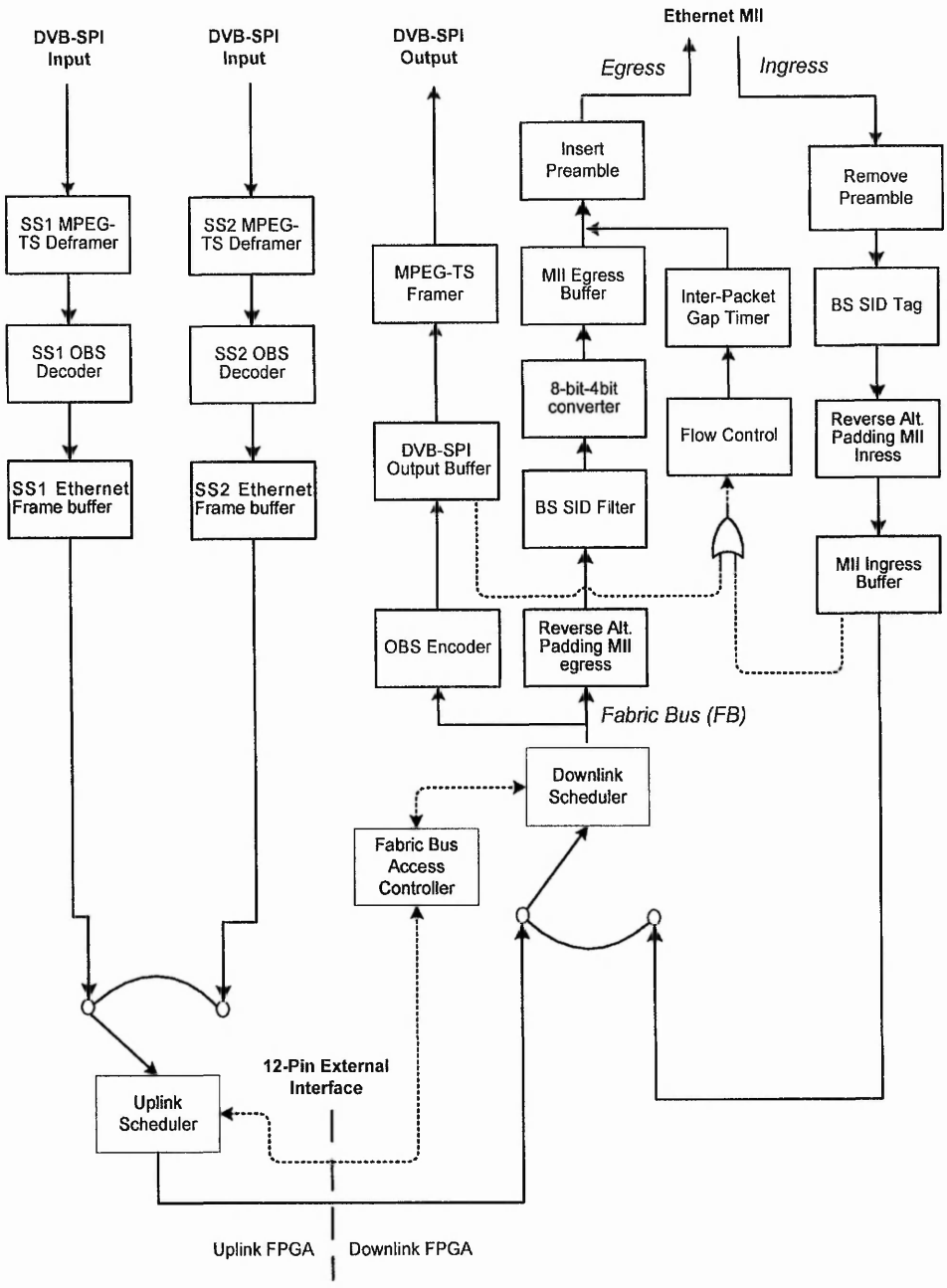


Figure 5-6: Base Station Design 2 Block Diagram

Module 18 Fabric Bus access controller

The Fabric Bus (FB) Access Controller ensures that the Downlink Scheduler and Uplink Scheduler do not simultaneously attempt to access the Fabric Bus. A simple request and grant mechanism is employed where the uplink or downlink schedulers can request access to the FB by sending a request pulse to the FB access controller. The FB access

controller will issue a grant pulse to one of the schedulers, allowing it transmit exactly one Ethernet frame.

There are three situations where a request will be denied to Schedulers. The first is when the FB is currently in use by the other scheduler. The second is when the FB has been used and released for less than twenty four byte time. The second rule ensures that the minimum inter-frame-gap defined by the Ethernet standard is met. Finally, the third is when both Schedulers simultaneously send a request pulse; neither of the schedulers will be granted access. To guarantee avoidance of another request collision, the Uplink and Downlink schedulers were designed to back-off for one and two clock times respectively before re-requesting access. However, in review of the back-off timing, a more efficient method can possibly be implemented by immediately granting access to Uplink while Downlink is backed-off for only one clock cycle.

Module 19 Uplink Scheduler

The function of the Uplink scheduler is to request access to the Fabric Bus from the Fabric Buss Access Controller whenever frames are available in the Ethernet frame buffers. The operation of the Uplink Scheduler can be explained by examining its state machine (Figure 5-7). The start state is SCAN, where the state machine waits for a frame to be available in a buffer. Once a frame is available at a buffer the state machine will attempt to obtain the medium by transitioning to GETMED state. In GETMED state, the state machine will request access to the Fabric Bus from the Fabric bus access controller. If a collision of request occurs, i.e. the Downlink Scheduler produces a request at the same instant, the state machine will back off (BACKOFF state) for one clock pulse. When the access is eventually obtained the state machine will transmit one frame from a non-empty Ethernet Frame buffer with highest priority index. The procedure to transmit a frame requires the state machine to cycle through a few states. In the diagram below the transmit-frame procedures are abstracted as TXS1FRAME or TXS2FRAME to represent a frame transmission from DVB-SPI input 1 or 2 respectively. The Uplink scheduler can be upgraded to support additional DVB-SPI inputs ports by adding more TXFRAME transitions between GETMED and SCAN states.

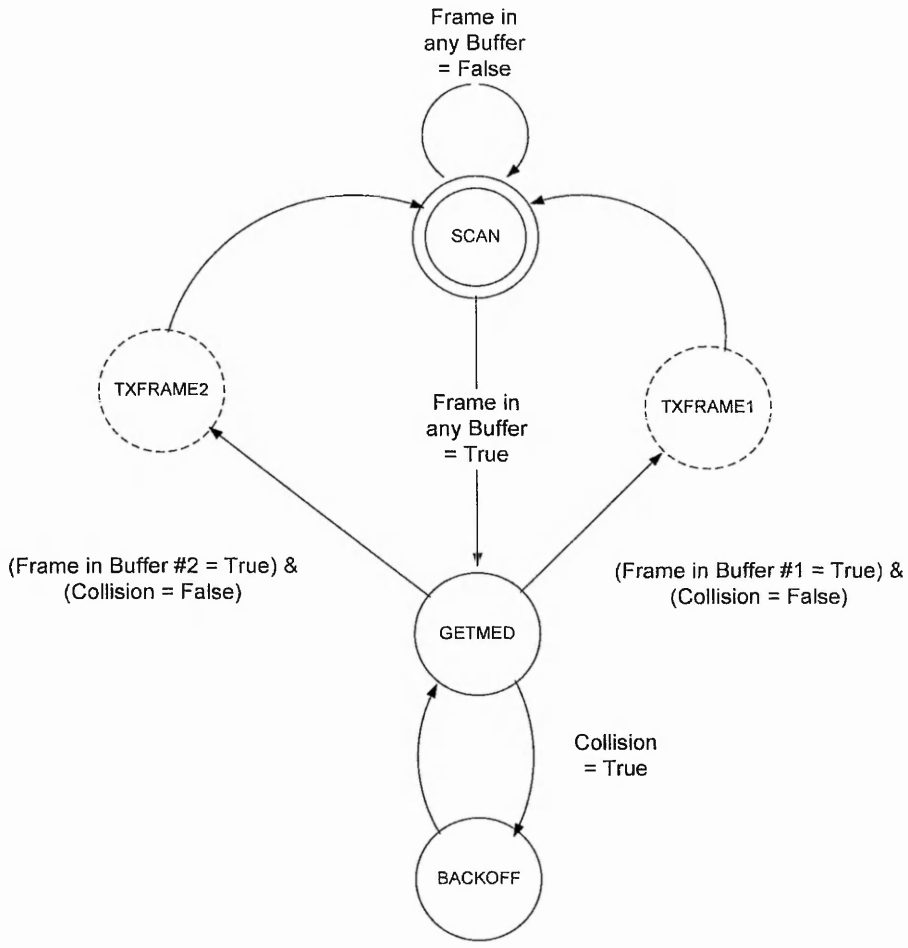


Figure 5-7: Simplified state-machine for Uplink Scheduler

Module 20 Downlink Scheduler

The Downlink scheduler module is responsible for requesting access to the Fabric Bus so that the MII ingress interface can be served. Similar to the Uplink scheduler, the Downlink Scheduler will check, using the SCAN state, if there are available frames in the MII buffer. Note that MII buffer is a cut-through buffer which will be ready to transmit a frame even if it only holds a partial frame. When the Downlink Scheduler is not using the Fabric bus, it will connect the 12-pin external interface to the Fabric bus so that data can flow from the Uplink scheduler directly to the Fabric bus. The Downlink Scheduler's state machine has two back-off states which are used in a request denial to the Fabric Bus. The TXMIIFRAME path is taken when access is granted to transmit one Ethernet frame from the MII to the Fabric bus.

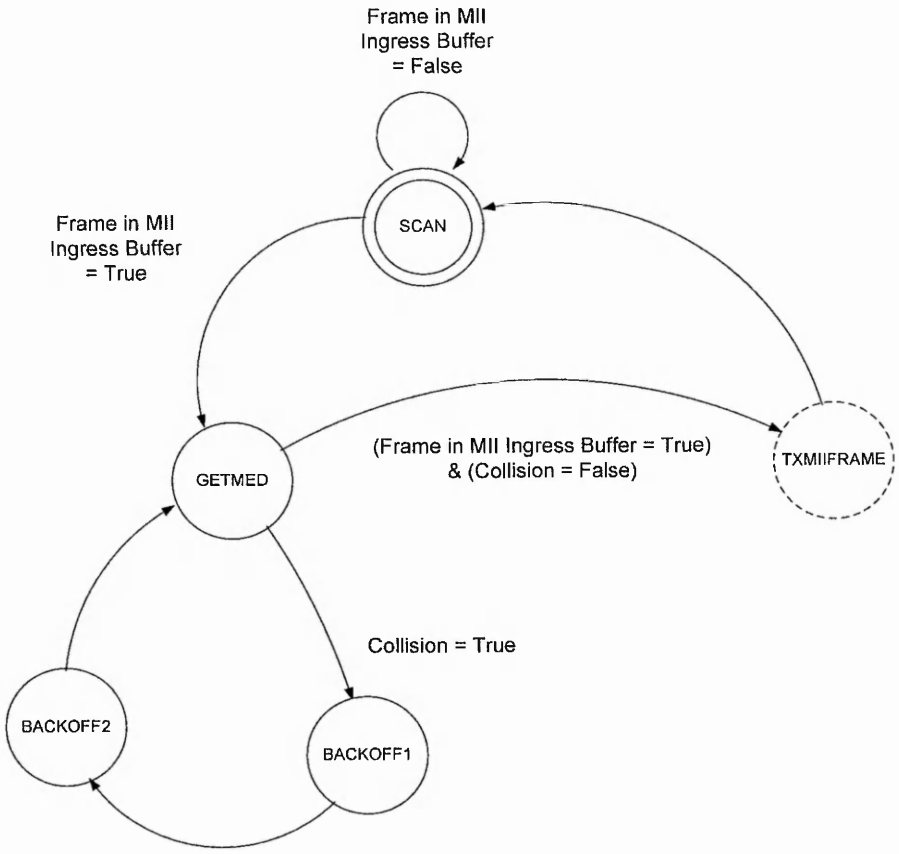


Figure 5-8: Simplified State machine for Downlink Scheduler

5.2.4. Synthesis Results

Base Station Design 1 and 2 were targeted onto Altera FPGA devices for synthesis and analysis to investigate resource usage, performance and scalability of the cores. The synthesis analyses were done for the Altera Cyclone FPGA device family since EPC1C6 Cyclone devices are readily mounted on development boards used in this research. Altera Cyclone devices are not as fast and do not have the advanced features of a higher performance devices such as Altera Stratix. Therefore, designs have to be more sensitive to the stricter timing requirements and limitations of a basic feature set. Lack of support for show-ahead FIFOs is a good example of a feature limitation that explicitly required time consuming work-arounds using custom logic. However, an advantage of using Altera Cyclone is that they are significantly lower-cost than the advanced FPGAs and are suitable for "high-volume, price-sensitive applications" [160]. The current range of the Altera Cyclone device family consists of eleven devices with capacities ranging from 2,910 Logic-Element and 59,904 memory bits; to 68,416 Logic elements and 1,152,000 memory bits.

As described in subsections 5.22 and 5.2.3, Base Station Design 1 is a single FPGA solution while Base Station Design 2 is a dual-FPGA solution. The successful synthesis of Base Station Design 1 shows that the electronics required for an EHE MWS Base station with support for two subscriber nodes can be implemented in a low-cost Altera Cyclone FPGA. The EP1C12 Cyclone device was the closest match for the Base station design 1 solution. Base Station Design 2 was also successfully synthesized into two smaller EP1C6 Cyclone FPGAs which allowed hardware prototypes to be built. Table 5-3 summarises the synthesis resource usage results for Base Station Design 1 and 2.

	Base Station 1	Base Station 2:	
		Downlink FPGA	Uplink FPGA
Device	EP1C12	EP1C6	EP1C6
Speed Grade	-6	-6	-8
Logic Element	1,622 / 12,060 (13%)	1,022 / 5,980 (17%)	665 / 5,980 (11%)
Memory Bits	151,296 / 231,616 (63%)	77,568 / 92,160 (84%)	73,728 / 92,160 (80%)
PLL	1 / 2 (50%)	1 / 2 (50%)	0 / 2 (0%)
Pins	44 / 185 (23%)	44 / 185 (23%)	41 / 98 (41%)
Cost*	£29.55	£15.77	£10.50

*Prices quoted in unit quantities in www.altera.com on 06/06/05

Table 5-3: Base Station FPGA resource usage

Altera devices are rated with a speed grade number (higher is better) which helps determine the maximum operating frequency of a design. The EP1C6 device available on the FPGA prototyping facilities for the Downlink FPGA is rated at -6 speed grade which should generally perform better than the device available for Uplink FPGA with a -8 speed grade. The use of a -8 speed grade FPGA made the design synthesis of the Uplink FPGA more challenging. The synthesis and place & route processes of the design cycle had to be iterated many times before a 'fit' without timing errors were found. The synthesis tool was very useful for this process since it allowed the requirements for each clock domain to be easily adjusted so that the compiler could give priority to higher frequency clock domains, while trading off speed on lower frequency clock domains where excess speed is not required.

Once clocking requirements are defined, the Place & Route compiler displays results of operating performance as slack time. Slack time is the leeway above the clocking requirements entered in the synthesis tool. The Place & Route compiler uses slack time instead of maximum operating speed when it decides that the design has complex clocking. A negative slack time on any of the clocks mean that the compiler had failed to find a successful fit. However, the magnitude of negative slack times reveal how much more speed is required to a successful fit, which helps fine tune the parameters for the next synthesis attempt. Table 5-4 below summarises the clocking requirements entered to the synthesis tool (synthesized speed) that produced the successful fit, as well as the reported slack times.

Clocks	Required Speed	Base Station Design 1		Base Station Design 2	
		Synthesized Speed	Slack time	Synthesized Speed	Slack time
MII Ingress	25 MHz	25 MHz	35.379 ns	30MHz*	29.292 ns
MII Egress	25 MHz	25 MHz	17.471 ns	30MHz*	27.772 ns
Reference Clock	20 MHz	25 MHz	63.975 ns	20MHz*	34.861 ns
Fabric Clock	14 MHz	25 MHz	34.622 ns	14 Mhz*	66.153 ns
				14 MHz**	34.367 ns
DVB-SPI Input 1	1.25 MHz	25 MHz	33.101 ns	7 MHz**	66.115 ns
DVB-SPI Input 2	1.25 MHz	25 MHz	34.265 ns	7 MHz**	69.445 ns

*Downlink FPGA **Uplink FPGA

Table 5-4: Synthesized speeds of Base Station Design

In the Frequency Division Multiple Access (FDMA) MWS investigated, the number of DVB-SPI input ports on the Base Station core must be equal to the maximum number subscriber stations supported. Increasing the number of supported subscriber nodes also increases the complexity of the base station design as each DVB-SPI port requires dedicated blocks between the port and the Fabric bus. Therefore scalability of the Base station designs can become an issue. Since this research does not intend to implement a wide scale hardware prototype, the scalability of the system is estimated from the detailed resource usage from the available synthesis results (see Table 5-5 and Table 5-6 below):

Module Name	Logic Elements	RAM bits	Mod. No.
DVB-SPI Output Buffer	383	18,944	5
SS2 Ethernet Frame buffer	239	36,864	15
SS1 Ethernet Frame buffer*	*239*	*36,864*	*15*
MII Ingress Buffer	233	36,864	15
MII Egress Buffer	92	20,480	11
Flow Control	57	1,280	12
Hub Scheduler	54	0	16
SS2 OBS Decoder	51	0	2
SS1 OBS Decoder *	*51*	*0*	*2*
MPEG-TS framer	44	0	6
OBS Encoder	26	0	1
Preamble Generator	20	0	14
SS2 MPEG-TS Deframer	20	0	8
SS1 MPEG-TS Deframer *	*20*	*0*	*8*
Glue Logic	15	0	
Base station SID Filter	17	0	9
Header Suppression	15	0	3
Base station SID Tag	12	0	4
Reverse alt pad MII ingress	12	0	17
8-bit-4bit converter	9	0	10
Reverse alt pad MII egress	9	0	17
Phased Lock Loop	4	0	7
Base Station Design 1	**1,687**	**151,296**	

Table 5-5: Resource Usage of Base Station Design 1

Module Name	Logic Elements	RAM bits	Mod. No.
Downlink FPGA	**1,022**	**77,568**	
DVB-SPI output Buffer	383	18,944	5
MII Ingress Buffer	233	36,864	15
MII Egress buffer	92	20,480	11
FB Access Controller	48	0	18
Flow Control	45	1,280	12
Glue Logic and Registers	41	0	
MPEG-TS framer	35	0	6
OBS Encoder	28	0	1
Downlink Scheduler	23	0	20
Preamble Generator	20	0	14
Base station SID Filter	17	0	9
Preamble Remover	15	0	3
Base station SID Tag	12	0	4
Rev. Alternate Padding MII	12	0	17
Rev. Alternate Padding MII	9	0	17
Phase Locked Loop	6	0	7
8bit-4bit converter	3	0	10
Uplink FPGA	**665**	**73,728**	
SS2 Ethernet Frame buffer	239	36,864	15
SS1 Ethernet Frame buffer *	239	36,864	15
SS2 OBS Decoder	51	0	2
SS1 OBS Decoder*	51	0	2
Uplink Scheduler	31	0	19
SS2 MPEG-TS Deframer	19	0	8
SS1 MPEG-TS Deframer*	19	0	8
Glue Logic	16	0	
Base Station Design 2	**1,622**	**151,296**	

Table 5-6: Resource Usage of Base Station Design 2

For each additional DVB-SPI input port supported, the Base station requires a dedicated MPEG-TS deframer, OBS Decoder and Ethernet Frame Buffer as highlighted in asterisks in the tables. In this FDMA based MWS the advantage of using the OBS encapsulation becomes very significant because an OBS decoder requires only 51 logic elements (see summary of section 3.3 for comparison with GFP). Assuming the additional complexity added to the schedulers are negligible, each additional DVB-SPI input port at the hub station is 309 (239+19+51) logic elements and 36,864 bits of memory. The limit for the number of subscriber nodes for the current EHE MWS implementation is 255 which is the maximum number of addressable subscriber nodes using an 8-bit SID field. A base station that requires 255 DVB-SPI inputs has an estimated device usage of about 79K (255 x 309) logic elements and 9.4 Mb (255 x 36,864) of memory. The current largest Altera FPGA device has capacity close to those

figures with 72K logic elements and 9 Mb of memory and priced at about £2,200 pounds [161].

Section 5.2 Summary

The design of the Base station design 1 was successful because the objective to explore the feasibility of a low-cost single FPGA solution was shown to be operational up to at least 56 Mbps at post synthesis stage. Unfortunately, a single FPGA prototype could not be realised due to the unavailability of FPGA development boards that could support the resource requirement. The limiting resource was not the Logic Elements (LE) in the FPGA but the memory bits (RAM). This resulted in a large requirement upgrade from EP1C6 to EP1C12 device that left high amounts of unused resources. With the introduction of the new range of Altera Cyclone II devices, the EP2C8 FPGA will become available and a more optimal fit for Base station design 1 could be realised. Nevertheless, Base station design 2 that is a two FPGA implementation of the base station was also successfully achieved. As Base station design 2 required two smaller FPGAs rather than one large one, a prototype was realised within the limits of the available facilities. Details of experiments done on a complete EHE MWS using real applications are left to the next sections. Here, this section is concluded with a summary of the key achievements:

- In the context of a low-cost and flexible MWS, the layer-2 Ethernet Hub Emulation (EHE) concept is introduced and justified.
- Successful designs of two possible implementations of the Base station hardware were detailed for the proposed EHE MWS architecture.
- Post synthesis results of Base station design 1 and Base station design 2 with operational speeds of at least 56 Mbps will able to operate at the limits of available DVB equipment in the EHE MWS prototype.
- Total LE/RAM resource requirements for Base station design 1 and Base Station design 2 with support for two subscriber nodes were determined to be 1,687 / 151,296 and 1,622 / 151,296 respectively.

5.3. Ethernet Hub Emulation MWS Prototyping

Using the Base Station design 2 and Subscriber interfaces, a prototype of the proposed EHE MWS was investigated within the limits of the available facilities. The prototyping stage involved a complete system simulation and experiments in actual hardware. DVB equipments were set to its maximum capabilities while VHDL test-bench simulation was used to obtain matching optimal parameters for optimal latency performance. The final EHE MWS prototype had two operational subscriber nodes and achieved 56 Mbps downlink and 12.5 Mbps uplink, saturating the DVB-S equipment. On the EHE MWS prototype platform; latency, throughput and resilience tests were performed using application level software to determine the realistic performance of the system.

5.3.1. EHE MWS Simulation Platform

A virtual VHDL test bench platform was coded to simulate a complete MWS system. The primary purpose of the test bench is focused on the functional verification and optimisation of the Base Station and Subscriber nodes. Therefore, a few 'safe' abstractions of the system's physical layers were done that eased the design of the test bench as well as increase simulation speed. Since the simulation maintained register transfer logic level behaviour of the cores, a good preliminary behaviour of the system was able to be determined at this stage. The resulting MWS simulation platform was therefore well suited for this research and was able to greatly aid in removing design bugs, and fine tuning of the intellectual property cores. A top-level block diagram of the MWS simulation platform is shown in Figure 5-9 below:

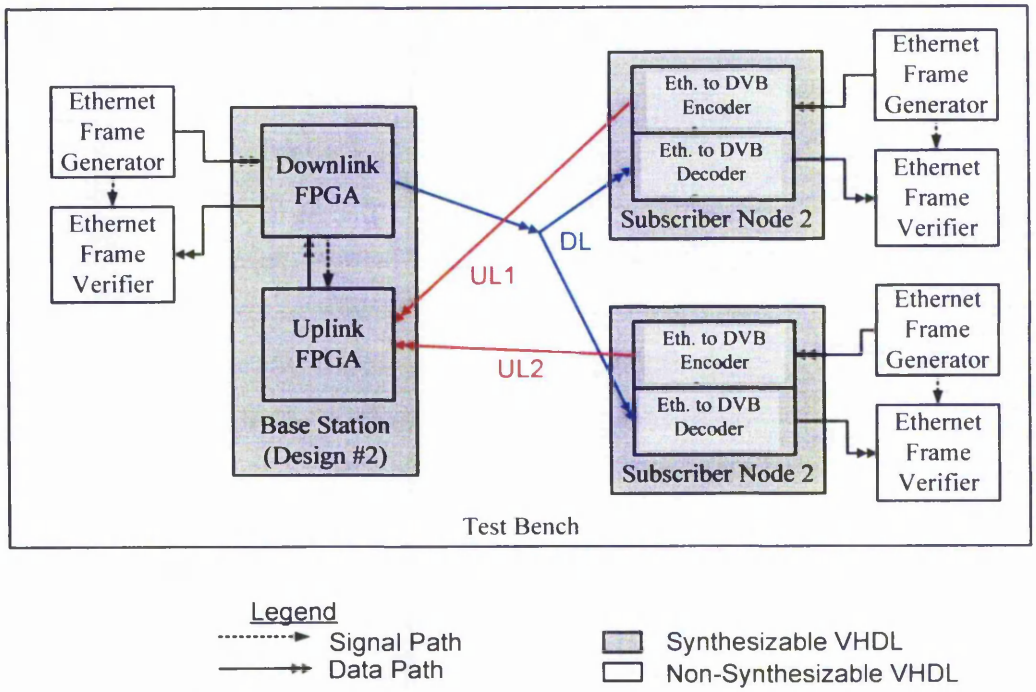


Figure 5-9: MWS Simulation platform

The physical layers can be safely abstracted without impacting the functional behaviour of the cores because they would only appear as latency to the paths to the Base Station and Subscriber interface. In simulation, the DVB system is emulated by direct connections (DL,UL1 and UL2 paths) between DVB-SPI input and output ports and without any latency. The operations of the Ethernet PHYs and host PCs are also abstracted in simulation. MII interfaces of the base station and subscriber nodes are connected directly to Ethernet frame generator and Ethernet frame verifier modules. As described in 4.2.1, The Ethernet frame generator and Ethernet frame verifier can emulate the behaviour of an Ethernet MAC for flow control reactivity, and also generate any type of Ethernet traffic. However, the verification of Ethernet traffic is more complex than the test bench described in 4.2.1 since a received Ethernet frame can originate from multiple sources. A simplified flow chart that is used in all three Ethernet frame verification modules is shown in Figure 5-10 below.

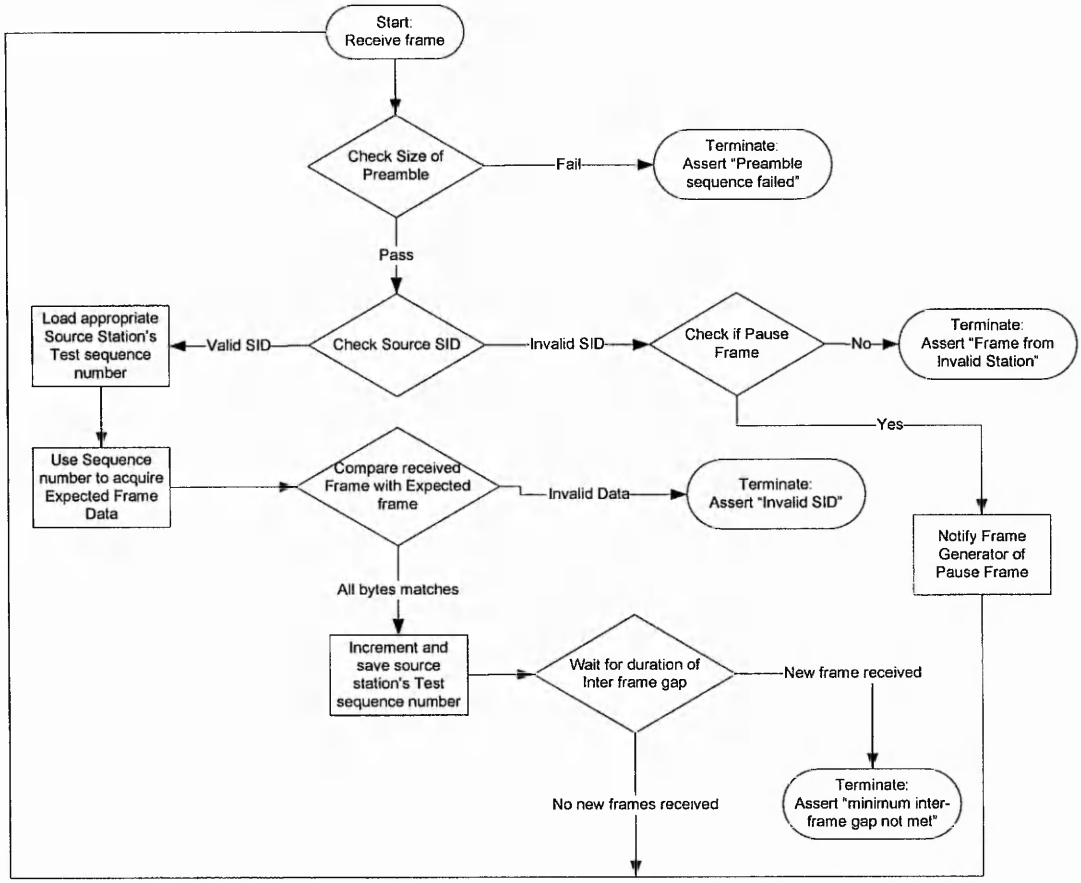


Figure 5-10: Flow chart of Ethernet frame verification module

Since generation and verification of test vectors are done automatically, large amounts of test vectors (in the order of thousands of frames) were simulated. The simulation performed included a stress test that saturated the Ethernet ports at maximum throughput for a runtime of 12 hours (real time). The base station passed the simulated stress without firing any of the assertions. However, the complexity of the MWS simulation platform meant that the actual simulated time was about 8 seconds. The table below compares the time taken for an identical set of test vectors on two different workstation facilities available to the research:

	CPU	Simulated Time	Time
Workstation 1	PentiumIII 1 GHz	50 ms	12 Min 55s
Workstation 2	Pentium4 2.8 Ghz	50 ms	4 Min 9s

Table 5-7: Comparison of Simulation time on different workstations

5.3.2. Buffer Threshold parameter optimization

Optimisation for buffer threshold and buffer size for the base station design was achieved using the MWS simulation platform. Buffer threshold is the amount of bytes stored in the buffer above which, the buffer will be considered congested. A congested buffer will alert the Flow control modules of the congested state. The Flow control modules will send special Ethernet Pause frames which will produce a backpressure effect at the Ethernet port. The threshold parameters of the buffer are crucial in influencing the behaviour of the buffers. On one hand, if the threshold values are set too high, the latency and hardware memory usage will be unnecessarily increased. On the other hand, if the threshold values are set too low, the maximum downlink capacity will be affected.

The problem of determining optimal threshold becomes non-trivial since the Base station design has two buffers in the downlink data path; each with its own threshold parameter that has to be globally optimised as a single system. Further, the latency between a 'congestion alert' until the reaction of the far-end Ethernet MAC is non-constant since there is possibility of contention with data from DVB-SPI input ports An abstracted closed-loop model of the Base Station downlink flow control is shown in Figure 5-11 below. Modules not directly involved in flow-control process are shown as constant data paths to aid clarity of the diagram, but in the test bench simulation, all modules of the complete MWS are used.

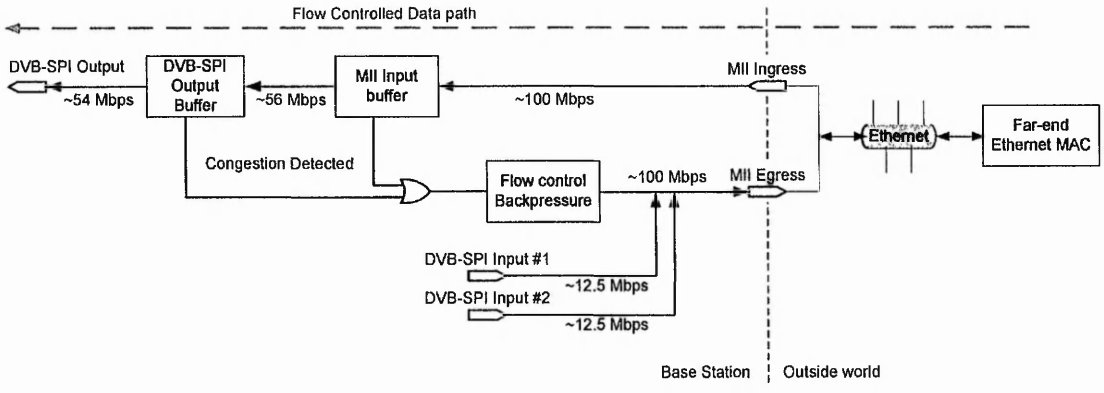


Figure 5-11: Base Station Flow control mechanisms

The fabric bus capacity is designed to be roughly equivalent to, but slightly more than, the DVB-SPI Output to minimise latency at the DVB-SPI port output buffer. Since the fabric bus capacity is almost equivalent to the downlink capacity, the amount of bytes stored in the buffer will be near-zero. Yet cut through forwarding is possible since the input rate of the DVB-SPI output buffer would be slightly more than its output rate. By maintaining cut-through forwarding and a near-zero 'DVB-SPI output buffer' occupancy, the queue-time from the DVB-SPI inputs are minimised even in situations where the MII ingress input port is saturated. While it may seem at first that the buffer can be removed by connecting the fabric bus directly to the egress port of the output buffer, buffer elasticity is required since there is a possibility of frame expansion due to OBS encapsulation, which can slow down the effective downlink rate.

The downlink path from the MII Ingress port to the DVB-SPI Output is of concern in this investigation for which the limiting capacity is about 54 Mbps. Hence, the task here was to determine the minimum threshold parameters for the DVB-SPI Output buffer and MII Input buffer to maintain the 54 Mbps capacity.

To avoid generating excessive Ethernet Pause frames, the flow control mechanism design limits output generation to one Pause Frame every 44 µs (1120 clock time at 25MHz) which was a good compromise. The flow control mechanism also employs a static pause quantum since Pause frames are copied from ROM instead of being generated. The pause quantum was chosen to be 34 which will stop the Far-end Ethernet MAC temporarily from starting a new frame for a short amount of time. A 34 pause quantum will cause a pause of 169 µs since $(34-1)\times 64\times 80ns$ = 169 µs, taking into account the time to transmit a Pause frame (service time) on to Ethernet. This value was chosen since it is smallest value that is able to pause the interface for one maximum frame size plus the maximum latency it might take from a congestion alert until acknowledgement of the Pause frame by the far end Ethernet MAC i.e. 1120 + 3052 = 4172 clock times. The equation includes the time of a maximum size frame since there is a high probability that a maximum size frame from a DVB-SPI input interface will 'hog' the MII Egress port during which time the flow control mechanism may not generate further flow control frames to relieve the congestion.

Through simulations adopting trial and error of different parameters for the buffer thresholds, the optimal threshold parameters were found. To achieve the boundary case maximum buffer occupancy in simulation, the far-end Ethernet MAC is programmed to behave as an infinite source of traffic which aggressively sends as much maximum-frame-size traffic as possible along the down link path. DVB-SPI Input 1 and 2 are programmed to send a small amount of maximum sized frames, to simulate contention on the Pause frame path, but would not significantly affect the downlink throughput. The simulation was conducted many times using different pairs of buffer thresholds. The throughput performance is measured by determining the number of frames that are managed to be sent by the far end Ethernet MAC given 50 ms of simulation time. Although the simulation time was not long enough to accurately determine the achievable downlink capacities of each simulation, the number of frames sent was good enough to reveal optimized threshold values. The table below compares 47 simulation results obtained. As can be seen 170 frames in 50 ms is the maximum that can be achieved on the downlink path.

		MII Buffer Threshold								
		1000	1100	1200	1300	1400	1500	1600	1700	1800
DVB-SPI Output Buffer Threshold	900				153	158	162	165	167	167
	1000				158	165	168	167	**170**	169
	1100				165	169	169	169	170	170
	1200			168	168	169	**170**	170	170	
	1300			168	169	**170**	170	170	170	
	1400		167	169	**170**	170	170			
	1500	168	169	**170**	170	170				
	1600	169	**170**	170	170					
	1700	169	170	170						

Table 5-8: Simulation Results for threshold optimisation

From the table the value pair of 1000 / 1700 is chosen for DVB-SPI Output buffer and MII Input Buffer threshold respectively. Other threshold pairs such as 1300/1400 and 1200/1500, highlighted in bold, are equally optimal in terms of RAM resource usage since they also produce the lowest threshold sum of 2700. However, as the 1000 / 1700 value pair has a lower value for the MII Input buffer threshold, it was considered a slightly better choice, when all else is equal. A lower MII Input buffer threshold induces

a lower latency on other paths in the system, i.e. between DVB-SPI inputs to DVB-SPI output, which makes it preferable.

5.3.3. EHE MWS Experimentation Platform

A prototype of the proposed MWS system was constructed that enabled investigation of the MWS performance in a realistic networking environment where the mechanisms of higher layer protocols are present. Experiments conducted on the prototype platform reveal performance of higher layer protocols which were not simulated in VHDL test bench. Such a VHDL emulation would be very difficult and probably require impractical simulation time scales. The prototype platform also allowed experiments conducted on previous BFWA systems to be repeated on the prototype platform and hence service level comparison results were acquired. Improvements contributed by the proposed architecture can therefore be proven in a realistic environment.

Except for the custom FPGA PCBs, all physical connectors, cables, Ethernet and DVB equipments used to construct the prototype are standard components. Using standard equipments from mature technologies is advantageous as they are already commercially available, and hence the cost of the prototype is dramatically reduced. A schematic diagram of the system is shown in Figure 5-12 below while photographs of separate modules can be found in Appendix C.

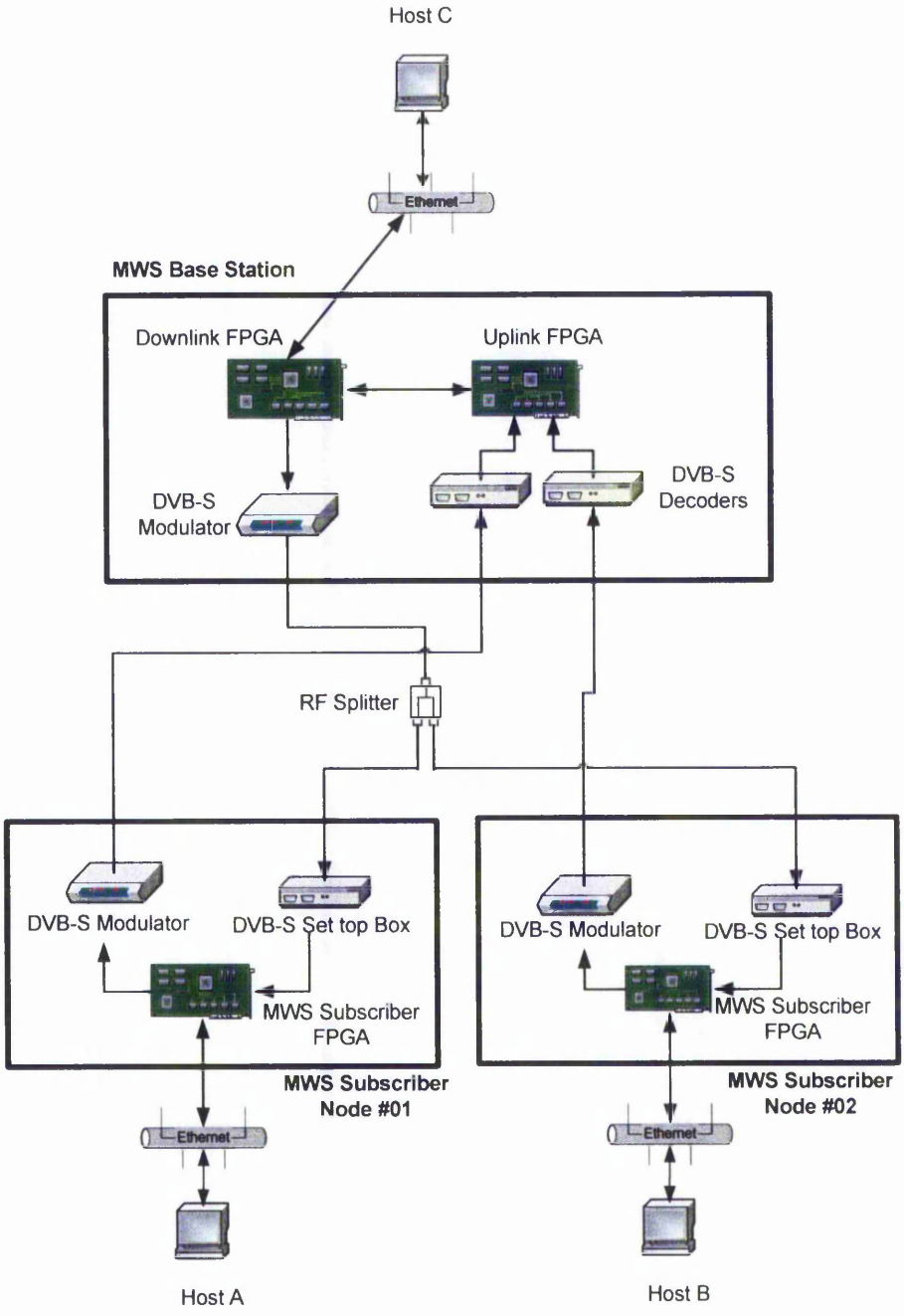


Figure 5-12: MWS Experimental Platform Set-up

The MWS prototype platform uses MWS Subscriber nodes based on the 12.5 Mbps configuration as described in section 4.3.1. Downlink and Uplink FPGA cores of Base station design 2 were programmed into EP1C6-6 and EP1C6-8 Altera Cyclone FPGAs respectively. The FPGA development boards were mounted onto another PCB that routed the general purpose input output pins to physical connector interfaces forming the required DVB-SPI, MII, Fabric bus and Debug ports.

The Radio Frequency (RF) output of the DVB-S modulator is split using an RF splitter to emulate a downlink broadcast to the DVB-S decoders of the Subscriber nodes. On the uplink direction, individual RF cables are connected from the Subscriber node's DVB-S modulator to the DVB-S decoders of the Base station to emulate an FDMA uplink path.

For interface to the DVB-S Modulators, the FPGAs drive external TTL-to-LVDS line drivers that form a LVDS compatible DVB-SPI output port. The LVDS compatible DVB-SPI ports are then connected to the DVB-S modulators using standard 25 pin sub-d printer port cables. For the DVB-S decoders, the copper tracks on the PCB are physically soldered to 25 pin sub-d connectors to form the DVB-SPI ports. The signals on the 25 pin sub-d connectors are TTL level and hence can be directly connected to the MWS subscriber FPGA development board via a printer port cable. MII-to-Ethernet development boards [162] are used to connect the MII connector on the MWS FPGAs to Ethernet. An MII-to-Ethernet development board consists of an Ethernet PHY chip and transformer which converts signal from its MII connector to its RJ-45 connector and vice versa.

5.3.4. TCP/IP and UDP/IP throughput tests

Six experiments were conducted to test throughput of TCP and UDP over the Ethernet Hub Emulation MWS prototype. In the Base station design, the Downlink scheduler, Uplink scheduler and Scheduler controller modules are responsible for controlling access to the available bandwidth while maintaining fair and efficient use. The experiment was designed to test the functionality of these modules by saturating paths between subscriber nodes and base station. Paths were tested individually as well as simultaneously to reveal their capacities in unobstructed and contended scenarios. The diagram below shows a model illustrating the logical paths taken by Ethernet frames in MWS prototype. To aid understanding of the logical paths, this model represents the DVB-S modulator of the Base Station (highlighted with asterisks) as two separate blocks.

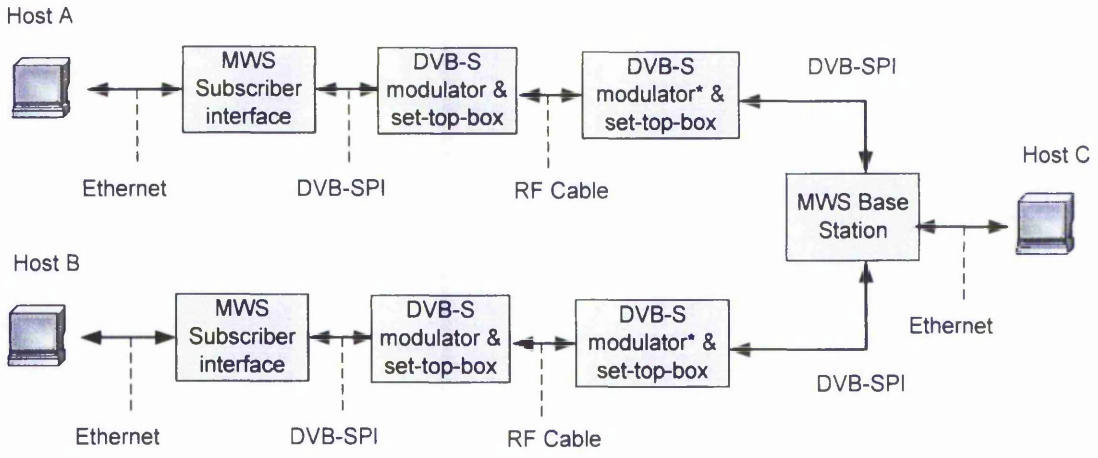


Figure 5-13: Experimental set-up for measurements

Host A, B and C are modern PCs installed with Windows XP operating system. Host A is a 600MHz Pentium III Laptop, Host B is a 500 MHz Celeron desktop and Host C is a 1 GHz Pentium III Desktop. Each path created between two hosts required one host to run the PCATTCP application as a client and the other as server. In PCATTCP the transport protocol can be selected to be either TCP or UDP. The server will behave as an infinite source of data which is continuously streamed at maximum possible rate to the specified client. It should be noted that PCATTCP only establishes a single stream to pipe the data, as opposed to parallel streaming techniques used by some programs such as IPERF to increase utilisation [155]. Each experiment was conducted five times for both TCP and UDP and the average taken. The results of the six experiment sets are summarised in Table 5-9: below.

Exp.No.	Paths	Hosts	TCP/IP User Rate	UDP/IP User Rate
1	Hub to Sub.	C to A	44.9 Mbps	46.76 Mbps
2	Sub.2 to Hub	B to C	11.66 Mbps	11.78 Mbps
3	Sub.1 to Sub.2	A to B	11.65 Mbps	11.78 Mbps
4	Hub to Sub.2 &	C to B	33.75 Mbps	34.94 Mbps
	Sub.1 to Sub.2	A to B	11.64 Mbps	11.78 Mbps
5	Sub.1 to Hub &	A to C	11.66 Mbps	11.77 Mbps
	Sub.2 to Hub	B to C	11.66 Mbps	11.79 Mbps
6	Sub.1 to Hub &	A to C	11.66 Mbps	11.78 Mbps
	Sub.2 to Hub &	B to C	11.66 Mbps	11.78 Mbps
	Hub to Sub.1	C to A	22.21 Mbps	22.98 Mbps

Table 5-9: PCATTCP Throughput experiments result

Each case of the experiments produced results close to what is expected. For example, experiment 1 showed 44.9 Mbps of user throughput using single stream TCP, when the theoretical limit was estimated to be 49.8 Mbps for this protocol. This means that TCP is 90 % efficient in utilising the available Downlink capacity. This result was similar to the result obtained for the same experiment repeated over a normal 100 Mbps Ethernet network link, where an average of 92% efficiency was observed.

Interestingly, the utilisation of the downlink capacity for the TCP protocol is marginally better for simultaneous path experiments than for individual path experiments. This can be most likely explained by the effects of parallel TCP streaming since the simultaneous experiments effectively create parallel TCP streams on the downlink. A slight improvement is observed when comparing the 44.9 Mbps downlink utilisation of experiment 1, with 45.39 Mbps (33.75 + 11.64) of total downlink utilisation of experiment 4. Furthermore, the total utilisation is also improved, but by a smaller margin, in experiment 6 that established three parallel streams on the downlink achieving 45.53 Mbps (11.66 + 11.66 + 23.21). UDP tests however, did not show much difference in downlink utilisation whether parallel streams were established or not.

5.3.5. Ping Experiments

Unlike IP based BFWA systems, the proposed Ethernet hub emulation MWS will transparently forward LAN traffic of adjacent networks. As the Ethernet hub emulation MWS will inherently support Ethernet-based services, its latency performance becomes a critical factor. While IP web applications are usually designed to operate over networks with slow links and large variation in delays, Ethernet-based services may assume the underlying network to have comparatively low latencies and jitter.

The Ethernet hub emulated by the proposed MWS is compared with a mid range 100Mbps Cisco Ethernet switch to determine the latency overhead caused by the underlying MWS. The Ping program was used to measure the round-trip time between two hosts connected to the MWS experimental platform. Referring to the experiment set up in Figure 5-13, the first set of experiments measures the round trip time between a host on a Subscriber node and one on the Base Station network i.e from Host C to Host A. The average of two hundred round-trip-time measurements are taken for each frame

size between 64 and 1526 inclusive, with 100 byte increments. The same procedure was repeated for the second set of experiments, except the DVB modulators and demodulators were removed from the system. By comparing the first and second set of experiments, the latency introduced by the DVB modulators/demodulators and FPGA processing are isolated. Finally, the experiments were repeated again with the MWS was replaced by the Cisco Ethernet switch. The results are plotted in the graph below. The first, second and third experiments are labelled as DVB mod./demod., FPGA processing and Ethernet switch respectively.

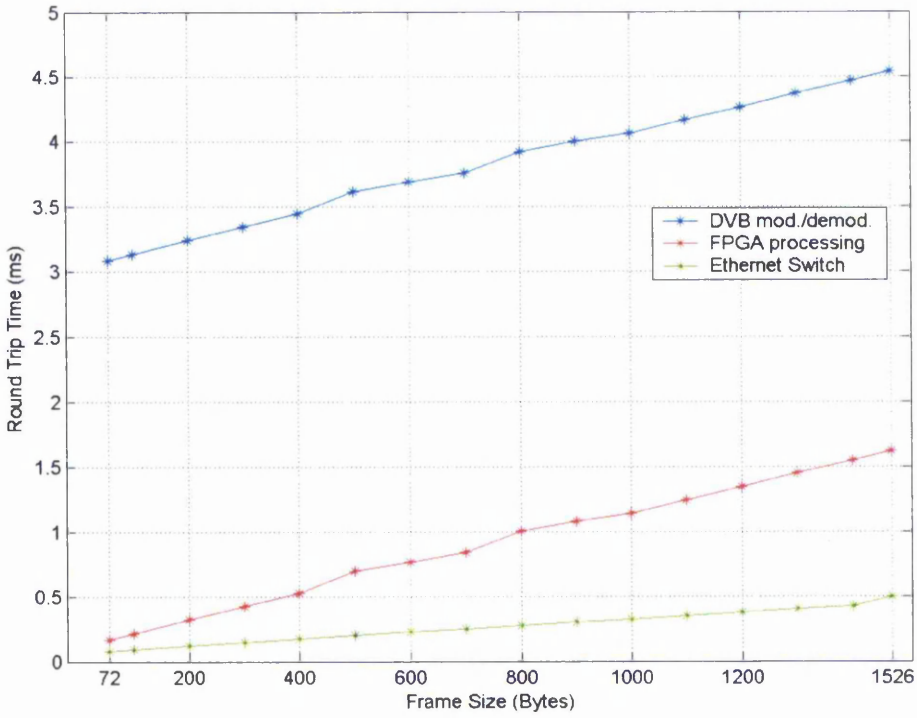


Table 5-10: Round trip time between Subscriber and Base Station

As can be seen from the plots, the extra latencies introduced by FPGA processing is between 0.1 ms and 1.1 ms increasing with frame size, when compared to the Ethernet switch. This can be considered a good result considering limitations of the underlying DVB MWS layer. The latency introduced by FPGA processing includes; conversion of Ethernet frames to MPEG-TS cells (twice), conversion from MPEG-TS cells to Ethernet (twice), store-and-forward latency (twice), and restricted bus speeds of 54 Mbps downlink and 14 Mbps uplink. As the OBS coder and decoder operated at line speed, the

store-and-forward latencies become the most significant contributor to latency which also has an increasing effect for larger frames.

As the EHE MWS achieved a low FPGA processing latency, the DVB modulation and demodulation is seen to be the dominant contributor. The latency induced by the DVB modulation and demodulation is constant at about 3 ms for all frame sizes. The 3 ms latency in the DVB is attributed to the forward error correction as well as 2k byte buffers in the modulators which are observed to be consistently 50% full. Including the latency of DVB modulator and demodulator, the total round-trip-time of the system between the Base station and subscriber node was between 3.2 ms (minimum frame size) and 4.5 ms (maximum frame size).

The latency results between hosts of two subscriber nodes (Hosts A and B) of the EHE MWS were also measured. The experiments were conducted using the same procedures to measure round-trip-time between the Base station and subscriber node explained earlier. The round trip times between two subscriber nodes are expected to be greater than that from a subscriber to base station because the frame has to first travel via the hub in each direction. Namely, the latency incurred was due to; conversion of Ethernet frames to MPEG-TS cells (four times), conversion from MPEG-TS cells to Ethernet (four times), store-and-forward latencies (four times) and DVB modulation and demodulation (four times). The results of the experiments are plotted in the graph shown in Table 5-11 below.

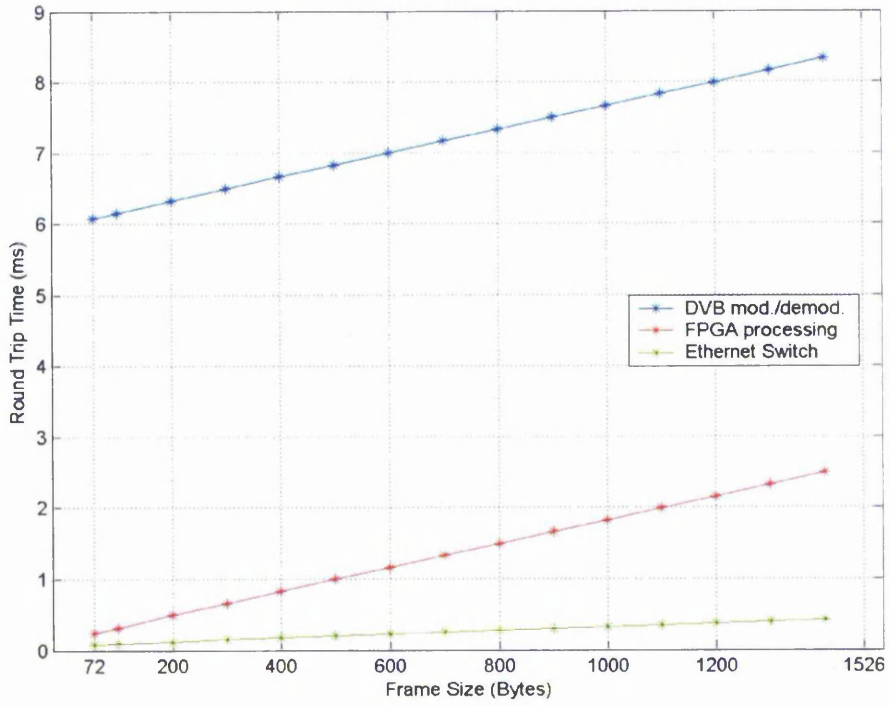


Table 5-11: Round trip time between two Subscribers

5.3.6. Resynchronization and Resilience Experiments

For practical purposes, the MWS investigated can be in one of two states, either quasi error-free or have a very high error rate that is unusable for data transport. The transition from unusable to the quasi-error-free state requires the receiver hardware to resynchronize with the incoming signal. During resynchronisation of the Optimised Byte Stuffing (OBS) decoder false starts can occur, especially when the incoming data stream emulates a starting sequence. False starts may at best cause an errored output, but in the worst case may cause the receiver hardware to crash and must be avoided. The experiments conducted in this subsection test the resynchronization and resilience mechanisms designed into the OBS decoder core.

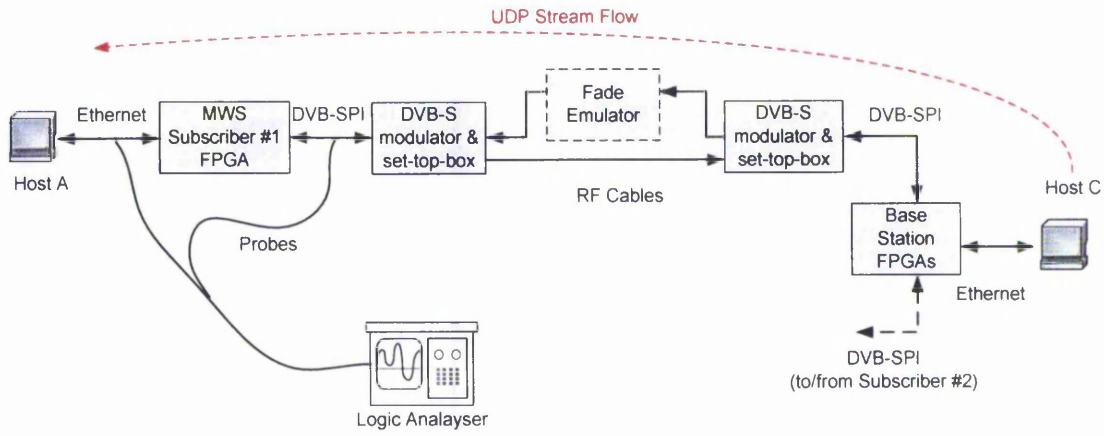


Figure 5-14: Experimental set up for resynchronisation and resilience tests

A rain fade emulator and logic analyser were added to the downlink path in the prototype MWS as shown in Figure 5-14 to measure the resynchronisation and resilience behaviour of the hardware under a heavy rain fade. Unlike some other wireless systems [163] whose link quality can have a gradual degradation, the concatenation of forward-error-correction techniques employed by the DVB-S equipment produce wireless links with a very steep degradation behaviour. A real MWS system is usually deployed with a link budget to achieve a certain availability rating. The availability rating (which is typically 99.9% or 99.99%) represents the percentage of time the MWS signal strength will be strong enough for the DVB-S equipment to ensure quasi error-free wireless links. Although at a small percentage, the availability rating suggests that link break down usually caused by rain attenuation must be expected and designed for.

The experiment methodology was to use the logic analyser to observe the input of the DVB-SPI interface and output of the Ethernet MII interface of the subscriber node at the instant the DVB-SPI interface assumes lock on the incoming signal. Firstly, a consistent UDP stream was established using the PCATTCP program to saturate the downlink path and produce a regular train of Ethernet frames between the base station and subscriber node 1. When the downlink path is saturated, the fade emulator is used to increase attenuation the radio frequency (RF) path to emulate a heavy rain fade until the DVB decoder looses lock on the incoming signal. At this point, the logic analyser is armed to trigger on the first valid MPEG-TS cell by detecting a positive edge on the SPI Data

Valid signal. Attenuation of the fade emulator is decreased and the logic analyser triggered to record the resynchronisation event. The output of the logic analyser is shown in the screenshot below:

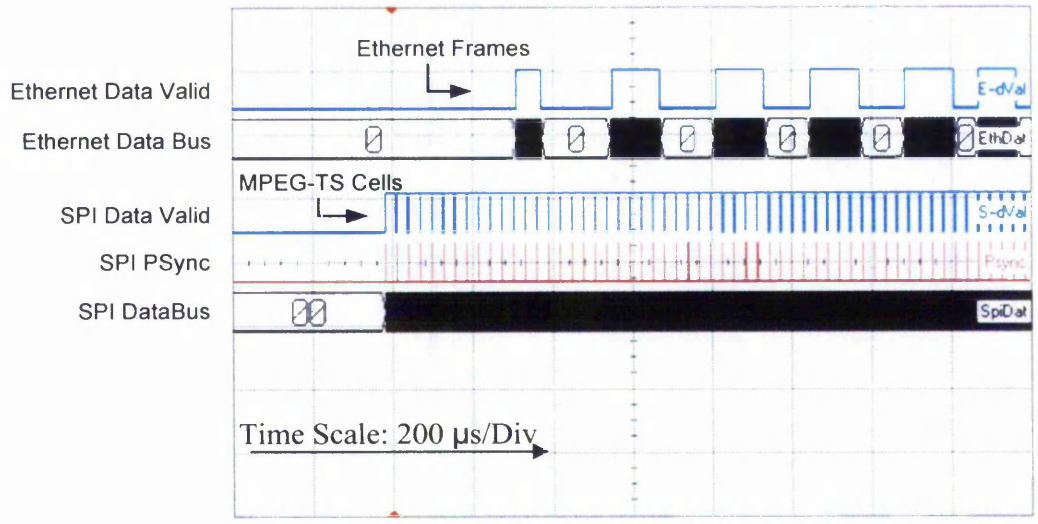


Figure 5-15: Logic analyser output of resynchronisation and resilience tests

The first Ethernet frame output of the subscriber node is an example of a false start. As can be seen, the Ethernet Data Valid signal reveals that the first frame is actually smaller than the expected size of the data train. Upon closer inspection of the SPI DataBus, the false start occurred because the third MPEG-TS cell (third pulse on SPI Data Valid) holds a partially encapsulated frame which had one data byte of 0x32 which emulated a start_of_frame indication of the OBS algorithm. However, the first complete encapsulated Ethernet that started in the tenth MPEG-TS cell was successfully decoded.

Another experimental result was obtained for the case where the end_of_frame delimiter was missing. In that experiment the Optimised Byte Stuffing (OBS) decoder resilience counter successfully detected the error and stopped the reassembly of an Ethernet frame when it reached 1526 bytes. When the experiment was repeated again with resilience counter mechanism disabled, the OBS decoder continued to reassemble the frame indefinitely. Eventually, a buffer over flow event occurred which lead to a fatal hardware failure. It was therefore known from this that the resilience mechanisms were working and that it was essential for robust resynchronisation. It is noted that, although resynchronization in the prototype hardware might produce defective frames onto

Ethernet, its presence on a healthy network is perfectly acceptable as Ethernet MACs are designed to handle defective frames that may occur due to other natural reasons such as; a collision or abrupt disconnection of a machine [98].

5.3.7. Demonstrations of Ethernet-Based Services over MWS

The prototype EHE MWS was linked to heterogeneous Ethernet and Wi-Fi networks to demonstrate its service-level capabilities. In this set-up, Host C was installed with two Ethernet Network Interface Cards (NICs) and is software-configured to behave as an Ethernet bridge. One NIC was connected to the Nottingham Trent University (NTU) network and the other to the MWS Base Station which effectively combines the two networks. Host A, B and C are given valid NTU-registered IP addresses and the default gateway configuration is set appropriately (152.71.0.66) so that the hosts can have internet connection via the NTU network.

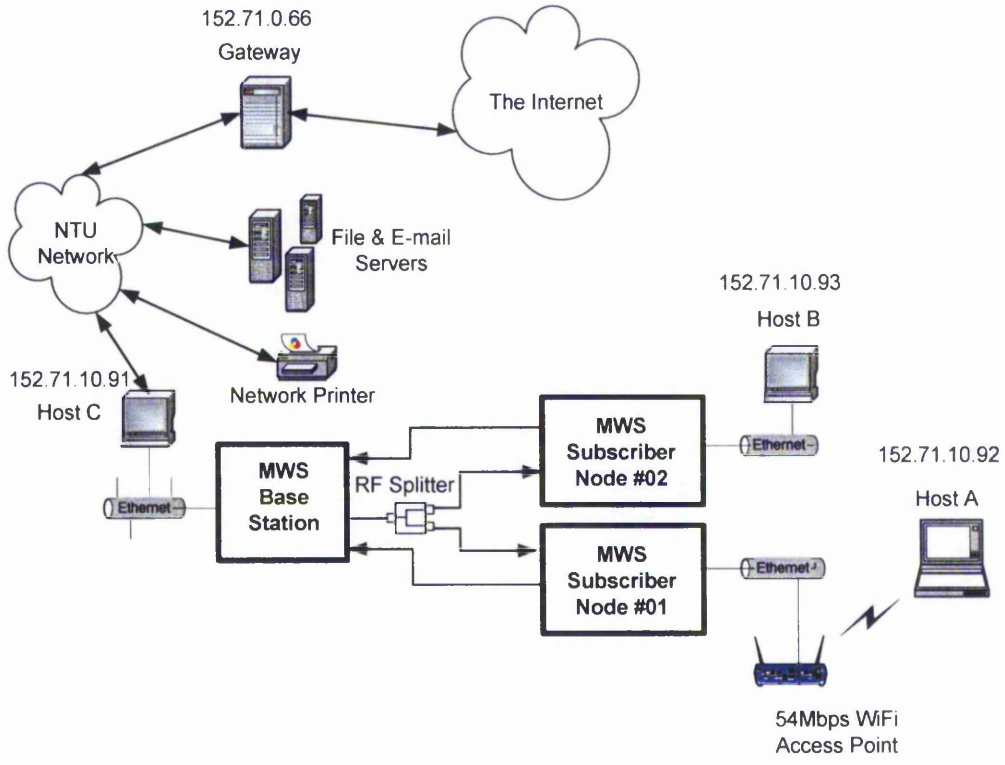


Figure 5-16: Ethernet Hub Emulation (EHE) Multimedia Wireless System (MWS) Demonstrations

A 802.11g 54 Mbps WiFi Access point is connected to MWS Subscriber 01 and is configured as a repeater. As MWS Subscriber Node 01 can be about 5 km physically apart from the MWS base station if 42 GHz antenna were used, this set-up demonstrates the feasibility of using EHE MWS as a backhaul for a remote WiFi hotspot. Host A is installed with a USB (Universal Serial Bus) 802.11g 54 Mbps WiFi adaptor and is set-up to connect to the WiFi Access point. Maximum bit rate from Host C to Host A using TTCP test program was measured at 23.7 Mbps TCP user throughput. The throughput bottleneck is attributed to the WiFi Access point as the 23.7 Mbps speed was also observed in a control experiment, where TTCP was run between a host connected via Ethernet and another wirelessly to the same WiFi access point. Maximum throughput experiments from Host A to Host C was measured at 11.65 Mbps which was limited by the EHE MWS uplink path.

Similar IP/web-based application experiments done in previous MWS systems was demonstrated on Host A and B over EHE MWS. These experiments include legacy Internet services such as; telnet, file transfers using FTP (File Transfer Protocol), World Wide Web (WWW) browsing, streaming media download, and voice using Voice Over Internet Protocol (VOIP). For these demonstrations, appropriate clients were installed in the hosts which interacted with remote servers or hosts from the Internet. As these applications would not stress the MWS links the service-level observed were good - as if Host A and B were Ethernet-connected to the NTU network. A particularly interesting experiment was done using the Skype VOIP client [164] to establish a three-user conference call using Host A, Host B and a user from the Internet. The perceived service-level in this experiment was satisfactory for all three users.

In addition to IP/web-based applications, the EHE MWS is able to support Ethernet-based services which were not demonstrated in previous MWS systems. The NTU network largely employs Microsoft Server System solutions [165] to provide enterprise services to students and staff which can usually only be accessed within the campus Ethernet. The Microsoft Server System in NTU is considered an Ethernet-based service since it uses non-IP proprietary protocols which are meant to work only within the organization's network. Previous BFWA systems that have IP-only connectivity would not be able to provide Ethernet-based services in a straightforward manner. Among the applications demonstrated (without using VPN) from subscriber hosts are access to the

university's Exchange email server, Remote Desktop connection, network printing, and Windows Storage Server. These experiments demonstrate the feasibility to use EHE MWS to physically extend the organisations network and also the Ethernet-based services provided. In the university environment, a possible use of EHE MWS is for the provision of Ethernet connectivity from main campus network to remote university buildings and student halls of residence.

In another experiment, the WiFi Access point was given an NTU registered IP address and configured as a layer-3 IP router instead of a layer-2 repeater. This set-up puts Host A in a separate IP subnet which was automatically assigned to 192.168.1.100 IP address by the built-in DHCP server in the WiFi Access Point. In this set-up, Host A was able to have IP/web services but not Ethernet-based services. The reason is that Host A is now separated from the organisation's LAN and, as it is behind an IP router, only IP/web services worked.

Section 5.3 Summary

In this section, the successful prototyping of a complete Ethernet-Hub-Emulation (EHE) MWS using Base station design 2 and two subscriber nodes was described. At the simulation level, new VHDL models had to be defined that could stress test the EHE MWS designs developed. While the simulation platform allowed design bugs to be revealed and fixed, the complexity of a system level simulation limited the practical simulation run time to a few seconds. Using available facilities, a hardware prototype of the EHE MWS was constructed with MPEG-TS downlink and uplink speeds of 56 Mbps and 14 Mbps respectively. The software test programs Ping and PCATTCP were used to measure achievable behaviour using typical protocol stacks. The key latency and throughput performance results are:

- Application layer round-trip-time between base station and subscriber node was determined to be 3.2 ms.

- Application layer data throughputs of 44.9 Mbps and 46.76 Mbps downlink, and 11.66 Mbps and 11.778 Mbps uplink, were consistently achievable for TCP/IP and UDP/IP throughputs respectively.

- The OBS decoder has been demonstrated to be resilient to start_of_frame emulation that can occur during resynchronisation from rain fade events. The robust OBS decoder was also shown experimentally to successfully recover the first error-free Ethernet frame.

- IP/web-based and Ethernet services were demonstrated over a heterogeneous network with EHE MWS, Ethernet, WiFi, NTU campus intranet, and the Internet. Perceived service-level on hosts on EHE MWS was satisfactory as if directly connected to NTU campus intranet.

Chapter 5 conclusions

This chapter detailed the novel MWS Base station interface designs, simulation, synthesis and hardware prototyping that incorporated the proposed mechanisms for Ethernet Hub Emulation (EHE) architecture. A key mechanism that enabled EHE was the internal SID (Station IDentity) tag/filter which was implemented primarily at the MWS Base Station interface. This mechanism allowed the MWS interfaces to cooperatively ensure that EHE forwarding rules are followed while maintaining the high speed processing and forwarding. Two working VHDL Base Station designs were detailed. The first design (Base Station design 1) is an integrated solution whereby the base station interface was successfully synthesized into a single FPGA chip. The second design (Base Station design 2) introduced new modules that allowed the base station interface to be partitioned into two smaller FPGAs so that a hardware prototype could be constructed with the facilities available to this research project.

At the simulation stage, system level simulation was done using new test bench models described for functional verification of four FPGAs (two for Base Station design 2 and two for subscriber interfaces). Known design bugs, revealed by the automated verification models, were eliminated. Further, configurable parameters for a specific (56Mbps-downlink/14 Mbps-uplink) Base station design 2 were optimised for a hardware prototype.

Synthesis results of Base Station design 1 and 2 showed operational speed exceeding the 56Mbps design specifications. Total FPGA logic element and RAM resource usage of 1687:151 and 1622:151,296 for Base Station design 1 and 2 respectively was achieved on a low-cost FPGA architecture. The low-cost FPGA devices targeted was the Altera Cyclone family where the resource usage for the base station designs was within constraints of small to medium sized chip of this product range.

A complete EHE MWS prototype was built and tested with a heterogeneous set-up which included Windows XP and Linux hosts, Ethernet switches and 802.11g WiFi access points. Connecting the experimental network to the Nottingham Trent University (NTU) campus network allowed legacy Internet connectivity and IP/web-based

applications to be demonstrated. New Ethernet-based services were demonstrated which reveal an advantageous capability of the EHE architecture over previous BFWA systems that were limited to IP/web-based services. Application layer data throughput of 44.9 Mbps and 46.76 Mbps downlink and 11.66 Mbps and 11.78 Mbps uplink were sustained using TCP/IP and UDP/IP protocols respectively. Robustness of the EHE MWS prototype was also shown where the first error free Ethernet frame can be recovered after resynchronization.

Application layer round-trip-times over the EHE MWS prototype were determined to be 3.2 ms and 4.6 ms depending on Ethernet frame size. To conclude this chapter, the 3.2 ms smallest frame size round-trip-time result is compared (Table 5-12) with performance of previous BFWA systems that were surveyed in chapter 2. This service-level comparison is particularly challenging for the EHE MWS as its cut-through forwarding advantages are less significant for small frame sizes. Nevertheless, the EHE MWS had achieved a relatively large latency performance margin compared to previous MWS systems. Despite the underlying DVB-S layer which is known for incurring a significant amount of delay, the EHE MWS latency performance comes close to the performance of LMDS and MMDS systems which usually have lower latency interface due to less channel encoding required. Latency performance of a version of the EHE MWS, which was configured and synthesized with a 7 Mbps and 625 Kbps downlink and uplink bit rates respectively, is included in the table for comparison.

BFWA System	Downlink bitrate	Uplink bitrate	RTT	BFWA Class
EHE MWS	51 Mbps	12.5 Mbps	3.2 ms	MWS
CRABS	35 Mbps	2.112 Mbps	110.0 ms	MWS
AT&T LABS	27 Mbps	650 Kbps	10.0 ms	LMDS
Cambridge (ATM)	25 Mbps	25 Mbps	1.2 ms	MMDS
Cambridge (Eth.)	25 Mbps	25 Mbps	2.5 ms	MMDS
CABSINET	8 Mbps	1 Mbps	85.0 ms	MWS
EHE MWS (Slow)	7 Mbps	625 Kbps	22.9 ms	MWS

Table 5-12: Round-trip-time comparison of Ethernet Hub Emulation (EHE) MWS with other systems

6 DISCUSSION, CONCLUSION AND FURTHERWORK

6.1. Discussion and Conclusion from Work Done

The research detailed within this thesis has focused on the investigation and development of subscriber and base station interfaces for the NTU MWS that can efficiently encapsulate Ethernet frames and emulate the underlying MWS as an Ethernet hub. The MWS interfaces are designed in VHDL, implemented in low-cost FPGAs following a modern Electronic Design Automation (EDA) flow and prototyped to show the claimed service-level internetworking flexibility, support for Ethernet-based services, latency and throughput performance improvements compared to previous BFWA trials.

Literature review of relevant prior work from a broad scope of BFWA and standardization literature identified two primary areas that were addressed to achieve the research aims. Firstly, existing encapsulation algorithms are not optimised for (or not capable of) the transport of Ethernet frames, and where it is supported, the partial software implementation limits throughput and increases latency. Secondly, the broadcast downlink and point-to-point uplink topology of the BFWA medium had required; cumbersome workarounds when IP routing or ATM technologies were used for internetworking, and the use of sophisticated costly networking equipment.

An efficient byte stuffing type encapsulation algorithm that can be entirely and cheaply implemented in dedicated hardware was investigated. This prompted a unique byte-level acquisition and subsequent analysis of 'live' Ethernet frames to address several prevailing criticisms against its use. One primary concern was that Byte stuffing algorithms introduces large amounts of service-level unpredictability and jitter due to its factor-of-two worst case overhead. However, measurements undertaken in this work revealed that unpredictability and jitter are insubstantial. The outcome of the investigation was the creation of an Optimised Byte Stuffing (OBS) algorithm VHDL design that is tailored for this application to achieve higher average encapsulation efficiency, lower jitter and lower latency. The innovations employed by the OBS algorithm include; unique method of frame encapsulation and synchronisation

mechanisms, statistically optimal delimiter byte-value, minimisation of header count while reserving header space for additional functionality.

The analysis of network traces revealed that frame-size-distribution is dependent on the method of delivery e.g. TCP/IP or UDP/IP, while byte-value-distribution is dependent on the nature of the content e.g. uncompressed text, compressed files, control traffic. An observation can be made about the use of Byte Stuffing algorithms from the statistical analysis of byte-value-distribution. It was found that unpredictability and jitter in Byte Stuffing is inversely related to uniformity of the byte-value distribution. Therefore, since compression and encryption mechanisms were shown to result in more uniformly distributed byte values, they would be complementary when used in conjunction with Byte Stuffing encapsulation. It is expected that future traffic would reflect more application layer compression and encryption as processing power in the average PC increases and information security becomes a wider issue, the unpredictability and jitter of byte stuffing will be further diminished.

Another criticism against the use of Byte stuffing encapsulation was due to its implementation; which was thought to be 'very complex' as it requires processing at the byte-level and large amounts of buffer space at the receiver due to possible worst-case frame expansion. It was therefore previously believed that byte stuffing was limited to low-speed interfaces such as telephone modems. However, the innovative pipelined VHDL design of the OBS encoder and decoder algorithm also shows that it can be implemented at very high speeds and very efficiently in FPGA requiring few logic element and memory resources. Also, since the decoder can process input at interface rate, worst-case expanded frames would be decoded to its original size before being stored into a buffer. Hence, the OBS decoder at the receiver would require the same amount of buffer as the commonly employed frame-length marking algorithms used in previous BFWA interfaces.

An investigation and subsequent development of a high-speed and low-latency 'Ethernet to DVB/MPEG Encoder and Decoder' based on OBS encapsulation was undertaken. The 'Ethernet to DVB/MPEG Encoder and Decoder' physically interfaced to Ethernet MII and DVB-SPIs ports while performing other functions for Ethernet and MPEG-TS conversion including; header framing, format conditioning, buffering and flow control.

Similar functions performed in other BFWA interfaces were a throughput bottle-neck and the source of large latencies which impacted end-to-end TCP/IP and UDP/IP performance. In this research, the 'Ethernet to DVB/MPEG Encoder and Decoder' was successfully prototyped following complete process from verification, VHDL design, test-bench simulation, synthesis and programming into Altera Cyclone FPGAs using an integrated Electronic Design Automation (EDA) design flow. As the 'Ethernet to DVB/MPEG Encoder and Decoder' form the building blocks for an MWS subscriber interface, working prototypes were formed by integrating the programmed FPGAs into modified DVB equipment and other supporting hardware components that were developed as part of this research. Using results from simulation and experimentation, it was shown that the 'Ethernet to DVB/MPEG Encoder and Decoder' had met and surpassed the design specifications of at least 56 Mbps encoding/decoding rates and 5 ms latency which can saturate the DVB equipments used and maintained the performance of TCP/IP and UDP/IP applications.

The significant improvements of 'Ethernet to DVB/MPEG Encoder and Decoder' over previous interfaces are attributed to the processing at the layer-2 for encapsulation into MPEG-TS and its full implementation in dedicated FPGA hardware. While detailed implementations of previous BFWA interface were not available, a probable reason for slow throughput in those interfaces is the IP encapsulation into MPEG performed that requires packet parsing up to at least layer-3. Further, the previous BFWA interfacing equipment are typically implemented in software run on a PC workstation with hardware Ethernet and MPEG network interface cards installed. These can also be probable causes of the high latencies and encoding bottleneck in those systems.

It was previously claimed that the advantage of software implementation is that it enabled rapid prototyping of the interface where framing and encapsulation functions can be easily modified. However, from the experience of prototyping in VHDL and FPGA in this research, it is argued that the integration of EDA tools used also facilitated a rapid prototyping environment. The VHDL designs, also described as source code, albeit at a lower level, can be easily modified similar to software programming languages. Recompilation, synthesis, place and route and reprogramming the FPGA, i.e a chip 're-spin', typically took less than 20 minutes on a modest PC workstation.

An investigation was carried out to determine a method to make the underlying MWS emulate an Ethernet hub which minimises internetworking complexity and enables the MWS to provide transparent Ethernet-based services. The proposed solution was called the Ethernet-Hub-Emulation (EHE) architecture primarily implemented in the base-station interface design. This emulated the broadcast-like nature of Ethernet networking. The advantage of the EHE architecture over previous architectures is the fundamental minimisation of complexity by exploiting the possibility of internetworking at layer-2. Seamless layer-2 internetworking is made possible due widespread use of Ethernet in the LAN environment which are assumed to be used in the service provider (base-station) and (subscriber) customer premises. Hence to the external Ethernet networks, the internetworking complications associated with the topology of the MWS medium is abstracted. A key innovation in the EHE architecture was the introduction of the internal Station Identity (SID) addressing scheme which provides the means for the MWS base station and subscriber interfaces to perform internal tagging and filtering of frames that maintains high-speed encoding rates and sub-5ms latencies. Prototypes of two subscriber interfaces, one base station and DVB-S equipment were combined to formed a complete point-to-multipoint EHE MWS. The prototype EHE MWS was networked with existing Ethernet and WiFi networks where legacy IP/web-based services and newer Ethernet-based services were demonstrated. Results from simulation and experiments, as well as, architecture and service-level comparisons with previous BFWA systems provided evidence for the claimed improvements in latency, throughput and internetworking flexibility of EHE MWS.

The contributions of this research demonstrated the provision of low latency, high-speed and low-cost IP/web-based and Ethernet-based data services over MWS that is in high demand for Metropolitan Area Network (MAN) access. When combined with the existing DVB infrastructure of the NTU MWS, the prototype MWS was also shown to provide support for MPEG-based broadcast services. The contributions can make MWS a strong alternative to existing MAN access technologies for businesses and homes where dominance of Ethernet networking and DVB digital television is already seen. With efficient support for the so-called triple-play services and low-cost architecture, the outcome of new MWS interface designs has successfully achieved the research aims and corroborates leading work in this field.

6.2. Further Work

The framework presented in this research will serve, hopefully, for further research work. The suggested areas are as follows:

Widespread field trials

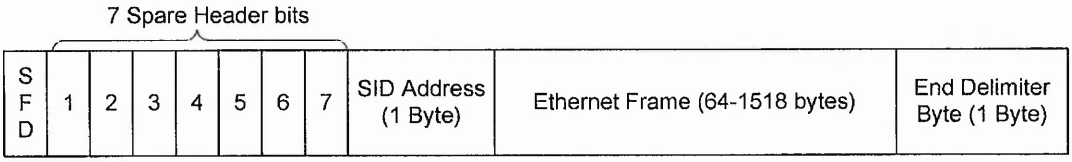
The base station and subscriber MWS interface prototypes developed was implemented using low-cost FPGA and hence is suitable for an investigation into a larger scale experimental deployment. Such an investigation might provide empirical evidence of performance and flexibility of the EHE architecture when scaled.

Protocol Enhancements

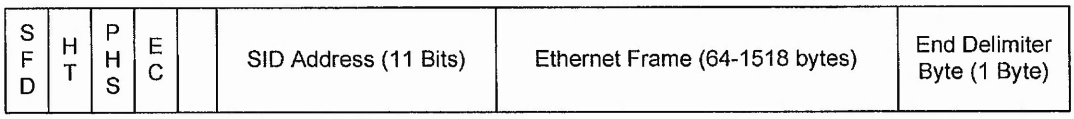
The OBS Encapsulation frame structure employed by the MWS interfaces provides space for seven spare header bits which are currently unused (reserved). Although the spare header bits can not be easily used to carry user Ethernet frame payload which comes in bytes, it still can be utilised as header flags to signify the presence of other data link layer mechanisms or internal protocols. These header flags effectively extend protocol functionalities when needed whilst maintaining a low minimum-header count.

Some header flag assignments are suggested to address some immediate shortcomings of the current protocol (Figure 6-1). The current Ethernet Hub Emulation (EHE) MWS does not have a mechanism to pass internal control messages, such as bandwidth request or flow control, between base station and subscriber interface. Similar to the 802.16 frame structure (see 2.2.7), a Header Type (HT) flag is suggested where setting this flag to '1' signifies that the current encapsulated frame is an internal control message (i.e. not a user Ethernet Frame). The presence of other data link layer mechanisms can also be indicated by flags for example encryption (EC) and payload header suppression (PHS). Besides header flags, the spare header field can be used to extend the SID address field. For example, the 8 byte SID field of the current implementation could be expanded using three spare header bits to obtain an 11 bit SID field. This would result in an address space expansion from 255 to 1023 subscriber stations per MWS sector.

Current implemetation



Suggested implementation



Legend
SFD: Start Frame Delimiter
HT: Header Type
PHS: Payload Header Supression
EC: Encryption

Figure 6-1: Suggested Frame Structure of Ethernet Hub Emulation (EHE) MWS

Return channel Improvements

The return channel (uplink) solution employed is based on a Frequency Division Multiple Access (FDMA) scheme. However, Time Division Multiple Access (TDMA) is usually viewed to be more suitable for BFWA as it can better handle the dynamic bandwidth demands of data services. Unlike most of the MPEG based encapsulation techniques surveyed, the OBS encapsulation developed does not depend on any of the MPEG-TS header (e.g. the PUSI bit). This opens an avenue to explore non-MPEG based return channel mechanisms provided the current support for MPEG transport in the uplink is viewed to be redundant.

Ethernet Switch emulation

In the current Ethernet hub emulation, frames are re-broadcast onto the downlink although it is destined to a host that can be found on the Base station Ethernet. A possible layer-2 solution, commonly employed in modern Ethernet switches, is to filter frames that match any of the MAC addresses of known Hosts that reside on the base-

station's Ethernet. An IP based Layer-3 solution could be used to filter frames that has the address of known IP routers that reside on the base-station's Ethernet.

List of Publications

Mun, W.Y., Germon, R., "An Efficient Ethernet over DVB Encapsulation Technique for MWS", PG Net 2004 : The Convergence of Telecommunications, Networking and Broadcasting, conference paper at The Liverpool John Moores University, Liverpool, UK, 27th-28th June 2004.

Mun, W.Y., Germon, R., Lam, E.B., "Multimedia Wireless System for Broadband Access", SET for Britain, poster presentation at The House of Commons, London, UK, 15th December 2003.

Mun, W.Y., Germon, R., "Data Interface for 42GHz DVB Multimedia wireless system (MWS)", PREP2003 International Conference, conference paper at The University of Exeter, Exeter, UK, 14th-16th April 2003.

References

[1] Park, Soo-Jin., et al., "Fibre-to-the-Home Services Based on Wavelength-Division-Multiplexing Passive Optical Network", Journal of Light Wave Technology, vol. 22, no.11, Nov. 2004, pp.2582-2591.

[2] Smith, Clint., "LMDS Local Multipoint Distribution Service", McGraw-Hill Professional Telecom Publication, Aug.2000.

[3] European Radio communications Committee, "ERC Decision of 1 June1999 on the designation of harmonised frequency band 40.5 to 43.5 GHz for the introduction of Multimedia Wireless Systems (MWS), including Multipoint Video Distribution Systems (MVDS)", Jun. 1999.

[4] Nordbotten, A., "LMDS systems and their application", IEEE Communications Magazine, Jun. 2000, pp.150-154.

[5] Tardy, Isabelle., Grøndalen, Ole., "On the Role of Future High-Frequency BFWA Systems in Broadband Communication Networks", IEEE Communications Magazine, Feb. 2005, pp.138-144.

[6] The WiMax Fourm. Available at: http://www.wimaxforum.org, Online, [Accessed 14 Sept 04].

[7] Koffman, I., Ramón, V., "Broadband Wireless Access Solutions based on OFDM Access in IEEE 802.16" IEEE Communications Magazine Apr. 2002, pp. 56-63.

[8] Germon, R., et al., "42 GHz Multimedia Wireless System campus trial", IEE Seminar New Access Techs., IEE, 2000, pp.6/1-5.

[9] Lam, E. B., Germon, R., "Deployment of a 42GHz MWS broadband fixed wireless access campus network trial", 3rd Conf. on Postgrad. Research in Electronics, Photonics, Comms. And Software (PREP 2001), University of Keele, U.K., Apr. 2001.

[10] Germon, R., Lam, E.B., "42GHz MWS campus Network Trial", European Microwave Week 2001, Broadband Wireless Workshop, London, UK., Sept. 2001.

[11] ISO/IEC 13818-2, "Information Technology – Generic Coding of Moving Pictures and associated Audio Information – Part 2: Audio", International Organisation for Standardisation, 1994.

[12] ISO/IEC 13818-2, "Information Technology – Generic Coding of Moving Pictures and associated Audio Information – Part 3: Video", International Organisation for Standardisation, 1994.

[13] G. Fairhurst, et al., "Performance Issues in Asymmetric TCP Provision Using Broadband Satellite", IEE Proceedings in Communications, vol. 148., No.2., pp.95-99, 2001.

[14] Bushmitch, Dennis., et al., "Supporting MPEG Video Transport on DOCSIS-Compliant Cable Networks", IEEE Journal on selected areas in communications, vol.18, no. 9, Sept. 2000, pp. 1581-1596.

[15] Noro, R., Hubaux, J.P., "Clock Synchronization of MPEG-2 Services over Packet Networks", 1998.

[16] Lam, E.B., "42GHz Multimedia Wireless System: Measurement and Analysis", Ph.D. dissertation, The Nottingham Trent Univ., Nottingham, UK, 2004.

[17] Bolcskei, H., et al., "Fixed Broadband Wireless Access: State of the Art, Challenges, and Future Directions", IEEE Communications Magazine, pp.10-108., 2001

[18] QGirish Chiruvolu, et al., "Issues and Approaches on Extending Ethernet Beyond LANs", IEEE Communications Magazine, Mar. 2004, pp. 80-86.

[19] Nenov, G.P.C. , Bigini, G., Valente, F., "Transporting Ethernet services in metropolitan area networks (MANS)", Proceedings. 12th IEEE International Conf. on Networks, (ICON 2004), Nov. 2004, pp.53-59.

[20] International Organization for Standardization, "Information Processing Systems - Open Systems Interconnection – Basic Reference Model", ISO 7498, International Organisation for Standardization (ISO), 1984.

[21] Chandranmenon, G.P., "Reconsidering Fragmentation and Reassembly", Proceedings of the seventeenth annual ACM symposium on Principles of distributed computing, ISBN:0-89791-977-7, Puerto Vallarta, Mexico, 1998, pp. 21 – 29.

[22] Heijenk, G., El Zarki, M., Niemegeers, I.G., "Modelling Segmentation and Reassembly Processes in Communication Networks", Proc. of ITC-14, North-Holland, pp. 513-24, 1994.

[23] Cheshire, Stuart., "Consistent Overhead Byte Stuffing", Ph.D. Thesis, Computer Science Department, Stanford, 1997.

[24] Stallings, W., "Data and Computer Communications", Prentice Hall, Seventh Edition, ISBN: 0-13-084370-9. 2000.

[25] Cavanaugh, J.D., T. J. Salo, "Internetworking in ATM WANs", Advances in Local and Metropolitan Area Networks, IEEE Computer Society Press, 1994.

[26] Angelopoulos, J.D. et al., "Efficient transport of packets with QoS in an FSAN-aligned GPON", IEEE Communications Magazine, vol 42, no. 2, Feb 2004, pp-92-98.

[27] Rege, K., "QoS Management in Trunk-and-Branch Switched Ethernet Networks", IEEE Communications Magazine, vol. 40, no. 12, Dec 2002, pp.30-36.

[28] M. Shreedhar and G. Varghese, "Efficient Fair Queuing Using Deficit Round-Robin," IEEE/ACM Trans. on Networking, vol. 4, no. 3, 1996.

[29] Mathis, Matthew., Mahdavi, Jamshid., "Forward Acknowledgement: Refining TCP Congestion Control", Symposium on Communications Architectures and Protocols, (SIGCOMM), Aug. 1996.

[30] Forouzan, B.A., "Data Communications and Networking", McGraw Hill, Third Edition, ISBN 0-07-251584-8, 2004.

[31] Wechta, J., Eberlein, A., Halshall, F., Spratt. M., "Simulation Based Analysis of the Interaction of End-to-End and Hop-by-hop flow control Schemes in Packet Switching LANs", Proc. of the Fifteenth UK Teletraffic Symp. on Performance Engineering in Information Systems, UK 1998.

[32] Wechta J., Eberlein A., Halsall F. and Spratt M. "An Investigation into the Performance of Switched LANs", Proc. of the Conf. on Networks and Optical Communications, UK, 1998.

[33] Glover, I,A, Grant, P.M., "Digital Communications", 2nd Edition, ISBN: 0-13-089399-4, 2003. pp 295.

[34] Knutsson, B., "Increasing Communication Performance via Adaptive Compression", Proc. of the Seventh Swedish Workshop on Computer Systems Architecture, Gothenburg, Sweden, 1998.

[35] I. Papaefstathiou, *"Low level Hardware Compression for Multi-Gigabit Networks",* Journal of Circuits, Systems and Computers, special issue on VLSI Architectures for Multimedia Applications.

[36] Tye, C.S., Fairhurst, G., "A Review of IP Packet Compression Techniques", Fourth annual Postgraduate Symp. on the convergence of telecommunication, networking and broadcasting, (PGNET2003), 2003, pp. 6-12.

[37] Perkins, S.J., Mutka, M.W., "Low-Bandwidth Access: An evaluation of application level protocol compressibility", Proceedings of the 4th International Conference on Telecommunication Systems, Modelling and Analysis, Nashville, TN, 1996.

[38] Lilley, J., Yang, J., Balakrishnan, H., Seshan, S., "A Unified Header Compression Framework for Low-Bandwidth Links", International Conference on Mobile Computing and Networking (MobiCom). ACM, August 2000.

[39] Mellor, J., Karamat, T., "Wireless LAN security issues and proposed preventive measures", Fifth annual Postgraduate Symp. on the convergence of telecommunication, networking and broadcasting PGNET 2004, pp. 123-127.

[40] Arbaugh, W.A., Shankar, N., and Wang, J., "Your 802.11 Network has no Clothes", Proc. of the first IEEE International Conference on Wireless LANs and Home Networks, 2001.

[41] MPEG working group, Available at: http://www.chiariglione.org/mpeg/about_mpeg.htm, Online, [Accessed on 28 June 2005].

[42] DVB Standard of the Digital world, Available at http://www.dvb.org /index.php?id=31, Online, [Accessed on 28 June 2005].

[43] The ATM Forum. Available at: http://www.atmforum.com/standards/anchorage .html, Online, [Accessed on 28 June 2005].

[44] Davic Digital Audio Video Council, Available at: http://www.davic.org/, Online, [Accessed on 28 June 2005].

[45] CableLabs - Revolutionising Cable Technology, Available at: http://www.cablemodem.com, Online, [Accessed on 28 June 2005].

[46] IEEE 802.16 Backgrounder, Available at: http://wirelessman.org/pub/ backgrounder.html, Online, [Accessed on 28 June 2005].

[47] ETSI Press Release, Available at: http://www.etsi.org/pressroom/previous /2002/etsi%2Dhyperaccess.htm, Online, [Accessed on 28 June 2005].

[48] Nordbotten, Agne., "System requirements" EMBRACE deliverables D2. IST-1999-11571, Nov. 2000.

[49] ETSI "Interaction channel for local multipoint distribution systems (LMDS)", EN 301 199, Jun. 1996.

[50] Clausen, H.D., Linder, H., Nocker, B., "Internet over Direct Broadcast Satellites", IEEE Communications Magazine, vol. 37, no. 6. June 1999.

[51] ISO/IEC CD 13818-1, "Information Technology – Generic Coding of Moving Pictures and associated Audio: systems Recommendation H.222.0", International Organisation for Standardisation, pp.22-23, 33-36, 44, 47, 52, 130-131, 1994.

[52] ETSI, "Specification for the Transmission of Data in DVB bitstreams", TS/ETS 301 192.

[53] Clausen, H.D., Nocker, B., Eder, T., Linder, H., "MPEG-2 AS a Transmission system for internet traffic", IEEE Magazine, ISBN: 0-7803-4468-5/98, 1998, pp.101-107.

[54] ETSI, "Digital Video Broadcasting (DVB); Specification for Service Information (SI) in DVB systems", ETS 300 480.

[55] Sties, P., Kellerer, W., "Radio Broadcast Networks enable Broadband Internet Access for Mobile Users", EUNICE'99 - Fifth EUNICE Open European Summer School, Barcelona, 1999.

[56] Collini-Nocker, Bernhard., Fairhurst, Godred., "ULE versus MPE as an IP over DVB Encapsulation" Second International Working Conference on Performance Modeling and Evaluation of Heterogeneous Networks (HETNET04). Jul. 2004.

[57] Linder, H., Clausen, H.D., Colini-Nocker, B., "Satellite Internet Services Using DVB/MPEG-2 and Multicast Web Caching", IEEE Communications Magazine, vol.32, no.6., Jun. 2000, pp.156-160.

[58] Filali, F., Aniba, G., Dabbous, W., "Efficient Support of IP Multicast in Next-Generation of GEO Satellites", IEEE JSAC Special Issue on Broadband IP Networks via Satellites, 2004.

[59] Nordbotten, Agne., "System requirements" Efficient Millimetre Broadband Radio Access for Convergence and Evolution (EMBRACE), deliverable D2 IST-1999-11571, Nov. 2000.

[60] Vacca, J.R., "Wireless Broadband Networks Handbook 3G, LMDS and Wireless Internet", Mc Graw Hill, ISBN0-07-213031-8, pp.395, 2001.

[61] Cable Labs, "Data-Over-Cable Service Interface Specifications : Radio Frequency Interface Specification", DOCSIS 2.0, CM-SP-RFIv2.0-I06-040804., 2004, pp. 105-108.

[62] Cable Labs, "Data-Over-Cable Service Interface Specifications: Baseline Privacy Plus Interface Specification", DOCSIS 1.2, SP-BPI+-I11-040407, 1999, pp. 13-18.

[63] IETF Internet Engineering Task Force, Available at http://www.ietf.org. [Accessed on Sept 04].

[64] DAVIC, "High and Mid Layer Protocols", DAVIC 1.2 specification, part 7. 1998, pp. 16-18.

[65] ETSI, "Broadband Radio Access Networks (BRAN); HIPERACCESS; DLC protocol specification", ETSI TS 102 000, 1997.

[66] ETSI, "Broadband Radio Access Networks (BRAN); HIPERACCESS; PHY protocol specification", ETSI TS 101 999, 2002.

[67] Kercheval, B., "TCP/IP over ATM: A no-nonsense Internetworking guide", Prentice Hall, ISBN 0-13-768599-8, pp.59-64, 1997.

[68] The ATM Forum, "Multi-Protocol Over ATM Version 1.1", AF-MPOA-0114.000, May 1999.

[69] Mähönen, P., Saarinen, T., Shelby, Z., Muñoz, L., "Wireless Internet over LMDS: Architecture and Experimental Implementation", IEEE Communications Magazine,. vol. 39 , no. 5, May 2001, pp. 126 – 132.

[70] Kuri, J., Gagnaire, M., "ATM Traffic Management in LMDS Wireless Access Network", IEEE Communications Magazine, vol. 39, no. 9, Sept. 2001, pp. 128 - 133.

[71] Bonnarigo, D., deMarco, M., Leonardi, R., "A comparison of back off and Ternary Tree collision resolution algorithms in HFC access networks", Proceedings of Globecom'98, Sydney, 1998.

[72] Kos, A., Pustisek, M., Bester, J. "Characteristics of Real Traffic Captured at Different Network Locations" EUROCON, Ljubljana, Slovenia, 2003.

[73] IEEE, "Part 16: Air Interface for Fixed Broadband Wireless Access Systems", IEEE Std 802.16-2001, Dec 2001.

[74] Ghosh, A., Wolter, D.R., Andrews, J.G., Chen, R., "Broadband Wireless Access with WiMax/802.16: Current Performance Benchmards and Future Potential", IEEE Communications Magazine, vol.43, no.2, Feb 2005, pp.129-136.

[75] Mehaoua, A., Boutaba, R., Pujolle, G., "A Picture Quality Control Framework for MPEG video over ATM", Proc. 5th Workshop on Protocols for High Speed Networks, October 1996. pp. 49-59.

[76] ATM Forum SAA SWG, "Audiovisual Multimedia Services Video on Demand", specification 1.0, ATM-Forum/ AF-SAAA-0049.000, 1995.

[77] Wenwu Zhu; Hou, Y.T.; Yao Wang; Ya-Qin Zhang, "End-to-end modelling and simulation of MPEG-2 transport streams over ATM networks with jitter", IEEE Transactions on Circuits and Systems for Video Technology, vol. 8, no. 1, Feb. 1998, pp.9 – 12.

[78] Kweon, S.-K., Shin, K.G., "Real-time transport of MPEG video with a statistically guaranteed loss ratio in ATM networks" IEEE Transactions on Parallel and Distributed Systems, vol. 12, no. 4, April 2001, pp. 387 – 403.

[79] G. Alberico, "Interactive Broadband technology trials", Cellular Radio Access for Broadband Services (CRABS), deliverable D4P4, Jan. 1999.

[80] ETSI DVB, "Digital Systems for Television, Sound and Data Services; Framing Structure, Channel coding and modulation for 11/12 GHz Satellite Services", EN 300 421 v1.12, 1997.

[81] Germon, R., Downes, D., Lam, E.B., Oswald, C. and Williams, K., "42 GHz multimedia wireless system campus trial", IEE Colloquium New Access Network Technologies, London, UK, 2000, pp.6/1-5.

[82] Lam, E.B., "42GHz Multimedia Wireless System: Measurement and Analysis", Ph.D. dissertation, The Nottingham Trent Univ., Nottingham, UK, 2004.

[83] ETSI DVB, "Interaction channel for Cable TV distribution systems (CATV)" ETS 300 800, July 1998.

[84] Kim,B.J., et. al., "The AT&T Labs Broadband Fixed Wireless Field Experiment" IEEE Communications Magazine, Oct 2001.

[85] Kim,B.J., et. al., "The AT&T Labs Broadband Fixed Wireless Field Experiment", IEEE Communications Magazine, Oct 1999, pp. 56-62.

[86] Liao, R., Brown, M., Mapp, G., Wassell, I., "The Cambridge Wireless Broadband Trial", 1999 IEEE International Workshop on Mobile Multimedia Communications (MomuC), November 1999.

[87] Kercheval, B., TCP/IP Over ATM : A no-nonsense internet working guide, Prentice Hall, ISBN 0-13-768599-8, 1997.

[88] Mahonen, P., Saarinen, T., Shelby, Z., Munoz, Luis., "Wireless Internet over LMDS: Architecture and Experimental Implementation", IEEE Communications Magazine, pp126-132. May 2001.

[89] Mahonen, P., et. al., "40GHz LMDS System Architecture Development", ICT 1998, p.422-428.

[90] Saarinen, T., Jamin, A., Mohonen, P., Shelby, Z., T. "Modelling Internet Access Through LMDS Network". OPNETWORK'99 user conference, Washington, DC. Aug 30 - Sep 3, 1999.

[91] P. Mähönen, Z. Shelby, T. Sukuvaara., "Wireless Internet and Multimedia Services by Two-Layer LMDS System". International Conference on Multimedia & Telecommunications Management (ICMTM), Dec. 1998.

[92] P. Mahonen et al., "Wireless Internet over LMDS: Architecture and Experimental Implementation," IEEE Commun. Mag., May 2001, pp. 126–32.

[93] Schmitdt, M., Koudelka, O., et al. "The EMBRACE system Demonstrator", the EMBRACE project, 2001. IST Mobile and Wireless Communications Summit, Thessaloniki, June 2002. pp.17-19.

[94] IST-1999-11571 EMBRACE D2 "System Requirements", EMBRACE project deliverables. pp.80-81.

[95] IST-1999-11571 EMBRACE D11 "Medium Access Control for mixed traffic", EMBRACE project deliverables.

[96] Metcalfe, R.M., Boggs D.R. "Ethernet: Distributed Packet Switching for Local Computer Networks", Communications of the ACM, vol. 19, no. 5, July 1976. pp.395- 404.

[97] Institute of Electrical and Electronic Engineering (IEEE), "CSMA/CD access method and physical layer", IEEE 802.3 Part 3, 2000, pp.33-34, 496-525, 698.

[98] Ferrero, Alexis, "The Eternal Ethernet", Addison-Wesley, ISBN: 0-201-36056-X, 1999.

[99] Bedell, P., "Gigabit Ethernet for Metro Area Networks", McGraw-Hill Networking professional, 2002.

[100] Spurgeon, C. E. "Ethernet the Definitive Guide", O'Reilly and Associates, -56592-660-9, 2000, pp.76-78,82-84.

[101] Forouzan, B.A., "Data Communications and Networking", Mc Graw Hill, ISBN 0-07-251584-8, 3rd ed, 2004, pp.333-353.

[102] Standard Micro Systems (SMSC) "LAN83C180: 10/100 Fast Ethernet PHY Transceiver", SMSC datasheet, 2000.

[103] Rosa, P., "Standards watch: Ethernet breaks free of shackles", Light wave, Penn Well Publishing Co., 2004, pp. 14.

[104] Scholten, M., et al., "Data Transport Applications Using GFP", IEEE Communications Magazine, 2002, pp. 96-103.

[105] S. Bradner, ed., "Benchmarking Terminology for Network Interconnection Devices", Network Working Group, IETF RFC 1242, 1991, pp.1-12.

[106] Fairhurst, G., "ULE for transmission of IP datagrams over an MPEG-2 Transport Stream", IETF Draft, 2003.

[107] Jouni Korhonen, "MosquitoNet - Consistent Overhead Byte-Stuffing", Research Seminar on Middleware for Mobile Computing, 2002.

[108] Doshi, B., et. al., "A Simple Data Link (SDL) Protocol for Next Generation Packet Network", IEEE Journal on Selected Areas in Communications, vol. 18, no.10, Oct 2000, pp. 1825-1837.

[109] Cavendish, D., et al., "New Transport Services for Next-Generation SONET/SDH Systems", IEEE Communications magazine, May 2002, pp.80-87.

[110] Cheshire, S., "Consistent Overhead Byte Stuffing", IEEE/ACM Transactions on Networking, vol.7, no.2, Apr. 1999, pp. 1-15.

[111] Calusen, H. D., Linder, H., "Issues in Link-Level Encapsualtion for Broadband Wireless Channels", IST Mobile Communications Summit, Barcelona, 2001, pp.1-6.

[112] W. Simpson, ed., "RFC 1662 - PPP in HDLC-like Framing", Network Working Group, IETF RFC, July 1994.

[113] Williamson, C., "Internet traffic measurement", IEEE Internet Computing, Nov 2001, pp.70-74.

[114] Choi, Y., Lee, H.N., Garg, A., "Measurement and Analysis of Wide Area Network (WAN) traffic", The 2000 Symposium on performance evaluation of computer and Telecommunication Systems (SPECTS 2000), July 2000.

[115] M.T. Lucas, D.E. Wrege, B. Dempsey, A.C. Weaver, "Statistical Characterization of Wide-Area IP Traffic", In Proceedings of Sixth International Conference on Computer Communications and Networks (IC3N'97), Sept. 1997.

[116] Kos, A., Pustisek, M., Bester, J., "Characteristics of real packet traffic captured at different network locations". The IEEE Region 8: computer as a tool (EUROCON 2003), vol. 1, 2003, pp. 164-168.

[117] Fraleigh, C., et al. "Packet-Level Traffic Measurements from the Sprint IP Backbone", IEEE Network, vol. 17, no.6, Nov 2003, pp. 6-16.

[118] Beverly, R., Claffy, K.C. , "Wide Area IP Multicast traffic characterization", IEEE Network, vol. 17, no.1, Jan 2003, pp. 8-15.

[119] Cheshire, S.; Baker, M,. "A wireless network in MosquitoNet", IEEE Micro, vol.16, no.1, Feb 1996, pp.44-52.

[120] Beverly, R., Claffy, K.C. , "Wide Area IP Multicast traffic characterization (extended version)", Available at http://www.mit.edu/~rbeverly/papers/ mcastworkchar-tr.pdf, Online, Accessed on 19 Sept 2003.

[121] D. Morato, J. Aracil, M. Izal., "Analysis of Internet services in IP over ATM networks", IEEE Communications Magazine, vol.37, no.12, Dec. 1999, pp.92-97.

[122] Smith, F. D., Campos, F. H., Jeffay, K., Ott, D., "What TCP/IP Protocol headers can tell us about the web", ACM SIGMETRICS Performance Evaluation Review, vol.9, no.1, Jun. 2001, pp.245-256.

[123] A. Wolman, et al., "Organization-based analysis of Web-object sharing and caching". In Proc. of the USENIX Symp. on Internet Technologies and Systems, 1999.

[124] Tang, D., Baker, M., "Analysis of a local-area wireless network", Proc. of the 6th Annual int. Conf. on Mobile Computing and Networking (MobiCom '00), August 2000.

[125] Comer, D. E., Stevens, D.L., "Internetworking with TCP/IP Volume II Design, Implementation and internals", 3rd Ed., Prentice Hall,1995.

[126] EventHelix, "TCP/IP Sequence Diagrams : TCP Slow Start Sequence Diagram", Online, Available at http://www.eventhelix.com/RealtimeMantra/Networking/, Accessed on 5th Jun. 2004.

[127] Clausen, H.D., Linder, H., Nocker, B.C., "Internet over direct Broadcast Satellites", IEEE Communications Magazine, pp.146-151, June 1999.

[128] Musciano, C., Kennedy, B., "HTML: The Definitive Guide", 3rd Ed., O'Reilly, 1998, pp. 7-8.

[129] Cheshire, S., Baker, M., "Consistent Overhead Byte Stuffing" IEEE/ACM Transactions on Networking, vol.7, no.2, Apr. 1999, pp.159 - 172

[130] Druker, L., "Verification Metrics – How you know when you're done", IEE Electronics Systems and Software, vol. 1, no. 2, 2003, pp. 22-25.

[131] Altera, "Quartus II Development Software handbook v5.0", Online, Available at http://www.altera.com/literature. Accessed on 19 Sept 2004.

[132] Horowitz, P., Hill, W., "The art of electronics", Cambridge University Press, 1989.

[133] Schoberl, M., "JOP: A Java Optimized Processor for Embedded Real-Time Systems", PhD. Dissertation, Institute for Computer Engineering, Vienna University of Technology. 2005.

[134] Altera, "Cyclone Device Handbook", Online, Available at http://www.altera.com/literature. Accessed on 19 Sept 2004. pp.26.

[135] Xelic, "XCGFPFM – Frame Mapped GFP Core", fact sheet, Online, Available at http://www.xelic.com, Accessed on 19 Sept 2004.

[136] Intec, "GFP-F Processor IP Core (GFPF-GbE)", Online, Available at http://www.intec2000.com/GFPF-GbE.html. Accessed on 19 Sept 2004.

[137] Qiao, W., Ni, L.M., Rokicki, T., "Adaptive-Trail Routing and Performance Evaluation in Irregular Networks Using Cut-Through Switches", IEEE Trans. Parallel Distrib. Syst. vol.10, no.11, Nov. 1999, pp. 1138-1158.

[138] Chan, L. L., Lin W., "Performance Analysis of General Cut-Through Switchingon Buffered MIN Switches", Journal of Information Science and Engineering, vol.18, no.5, Sep. 2002, pp.745-762.

[139] Chiruvolu, Girish., et al., "Issues and approaches on Extending Ethernet beyond LANs", IEEE Communications Magazine, vol.42, no.3, Mar. 2004, pp.80-86.

[140] Partridge, C., Shepard, T.J., "TCP/IP Performance over Satellite Links", IEEE Network, pp.44– 49,1997.

[141] Angelopoulos, J. D., et al., "Efficient Transport of Packets with QoS in an FSAN-Aligned GPON", IEEE Communications Magazine, vol.42, no.2, Feb 2004, pp.92-98.

[142] ITU Rec. G.983.1, Study Group 15, "Broadband Optical Access Systems Based on Passive Optical Networks (PON)," Oct. 1998.

[143] Kiran Rege et. Al. " QoS Management in Trunk-and-Branch Switched Ethernet Networks" IEEE Communications Magazine, vol. 40, no.12., Dec. 2002, pp.30-36.

[144] Altera, "Single- & Dual-Clock FIFO Megafunction: User Guide", Dec. 2004, Online, Available at www.altera.com/literature, Accessed on 5th Jan. 2005.

[145] Altera, "Reed-Solomon Compiler: User Guide", Jul. 2004, Online, Available at http://www.altera.com/literature/ug/rs-compiler_ug.pdf, Accessed on 5th Jan. 2005.

[146] Chuah, A., "Cost-Effective implementation of Digital Video Broadcasting Channel Encoder", Mphil. dissertation, Nottingham Trent University, Apr. 2005.

[147] Pol, M., Teunissen, R., Veenendal, E., "Software Testing: A guide to TMAP Approach", Addisson Wesley, ISBN 0 201 74571 2, 2002, pp.202-205.

[148] PCA USA, "PCATTCP Utility", Online, Available at http://www.pcausa.com/Utilities/pcattcp.htm, Accessed 19th Sept 2003.

[149] McVoy, L., Staelin, C. "lmbench: Portable tools for performance analysis", Proceedings of the 1996 USENIX Technical Conference, 1996.

[150] Jin, G., Tierney, B., "Netest: A Tool to Measure the Maximum Burst Size, Available Bandwidth and Achievable Throughput", International Conference on Information Technology: Research and Education, 2003. Proceedings. ITRE2003. pp.578- 582, 2003

[151] Siyan, K.S., Parker, T., "TCP/IP Unleashed", 3rd ed., Sams publishing, ISBN. 0-672-32351-6, 2002. pp.317,221.

[152] EventHelix, "TCP/IP Sequence Diagrams : TCP Congestion Avoidance Sequence Diagram", Online, Available at http://www.eventhelix.com/ RealtimeMantra/Networking/, Accessed on 5th Jun. 2004.

[153] Chaskar, H. M., Lakshman, T.V., Madhow, U., "TCP over wireless with link level error control: Analysis and design methodology", IEEE/ACM transactions on networking, vol. 7, no. 5, 1999, pp.605-615.

[154] Lakshman, T.V., Madhow, U., "The performance of TCP/IP for networks with high bandwidth-Delay products and random loss", IEEE Trans. On Networking, 1997, pp.336-350.

[155] Tirumala, A., et al., "IPERF: Internet Performance", Online, Available at http://dast.nlanr.net/Projects/Iperf/, Accessed 19th Sep. 2003.

[156] Linder, Hilmar, Clausen, Horst D., Collini-Nocker, Bernhard., "Satellite Internet Services Using DVB/MPEG-2 and Multicast Web Caching", IEEE Communications Magazine, June 2000, pp. 156 – 161.

[157] Berkowitz, C.H., "Designing Routing and Switching Architectures for enterprise networks", Macmillan Technical Publising U.S.A. ISBN:1-57870-060-4, 1999.

[158] Sonia D. Bot, "Key Technical Considerations When Using Ethernet Solutions in Existing ATM and Frame Relay Networks" IEEE Communications Magazine, March 2004. pp.96-102.

[159] Cohen, R..,Kaempfer, G., "On the Cost of Virtual Private Networks", IEEE/ACM Transactions on Networking, vol. 8, no. 6, Dec 2000, pp.775-784.

[160] Altera, "Cyclone: The lowest cost FPGA ever", Online, Available at http://www.altera.com/products/devices/cyclone/cyc-index.jsp, Accessed on 4th Jul. 2005.

[161] Altera, "Altera E-store", Online, Available at http://www.altera.com/buy/devices/buy-devices.html, Accessed on 4th Jul. 2005.

[162] Micrel, "KS8721B/BT 2.5V 10/100BasTX/FX MII Physical Layer Transceiver", Rev. 2.2, Online, Available at http://www.micrel.com/, [Accessed on 24th Sept 2004].

[163] Kumar, A., "Comparative performance analysis of version of TCP in a Local Network with a lossy link", IEEE Trans. on networking, 1998, pp.485-498.

[164] Skype, "Skype: The world can talk for free", Online, available at www.Skype.com, [Accessed on 27th Sept 2004].

[165] Microsoft, "Microsoft Windows Server System Platform", Online, Available at http://www.microsoft.com/windowsserversystem/overview/overview.mspx, [Accessed on 8th June 2005].

APPENDICES

Appendix A

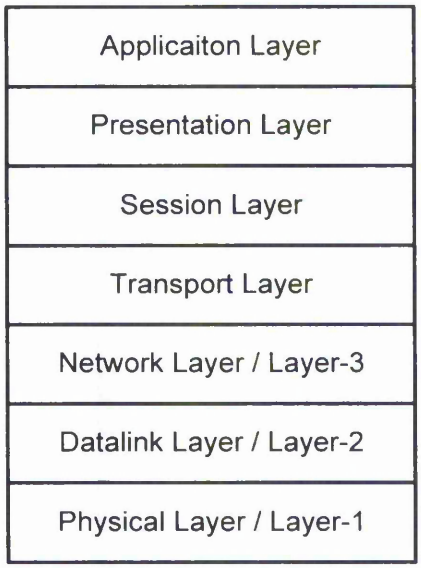


Figure A-1: International Organization for Standardisation (ISO) protocol Stack Model

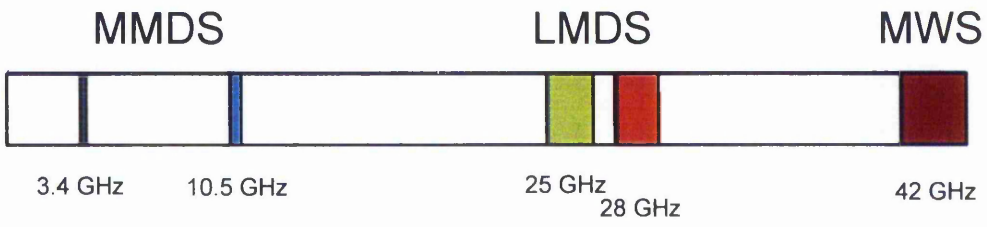


Figure A-2: Allocation of spectrum for different types of BFWA

Appendix B: SSL content Analysis

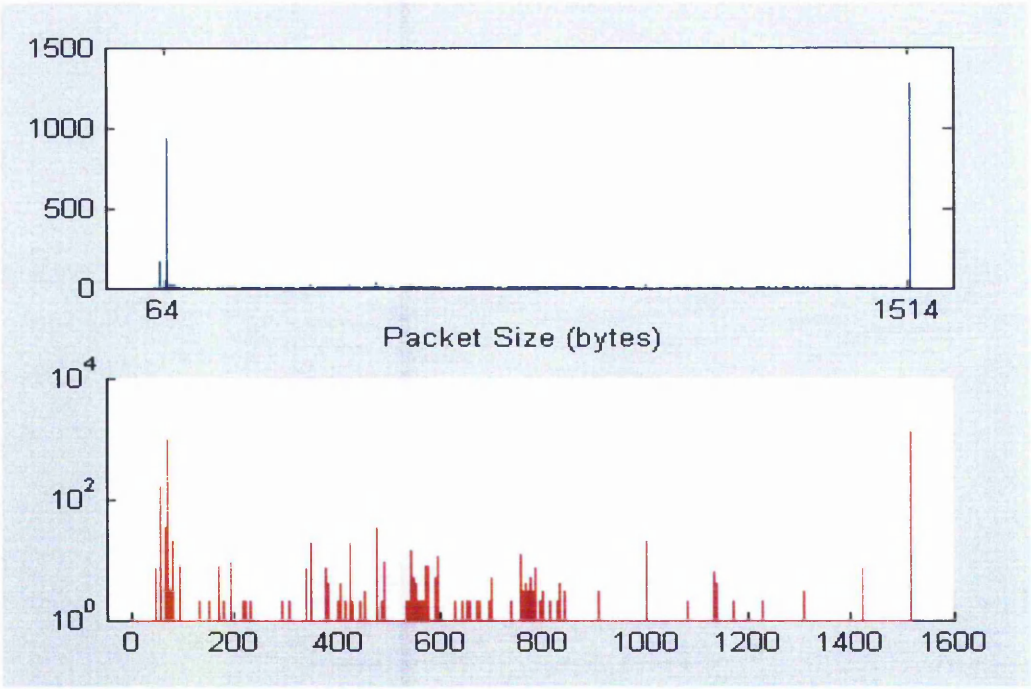


Figure B-1: Frame Size distribution of Secure Socket Layer (SSL) web content

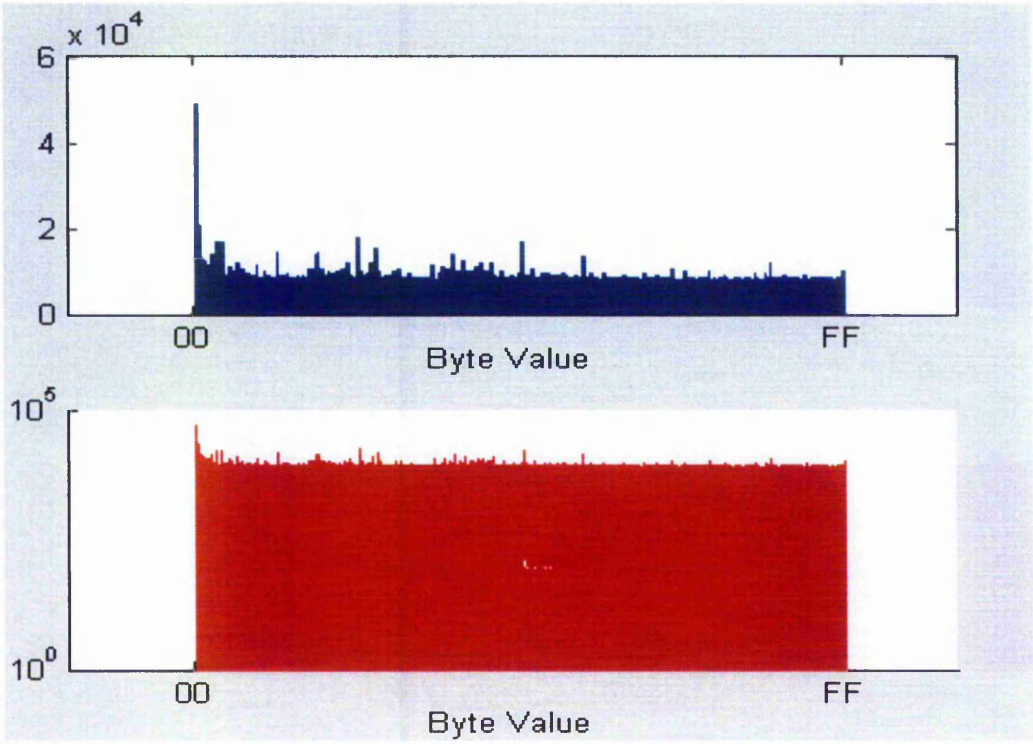
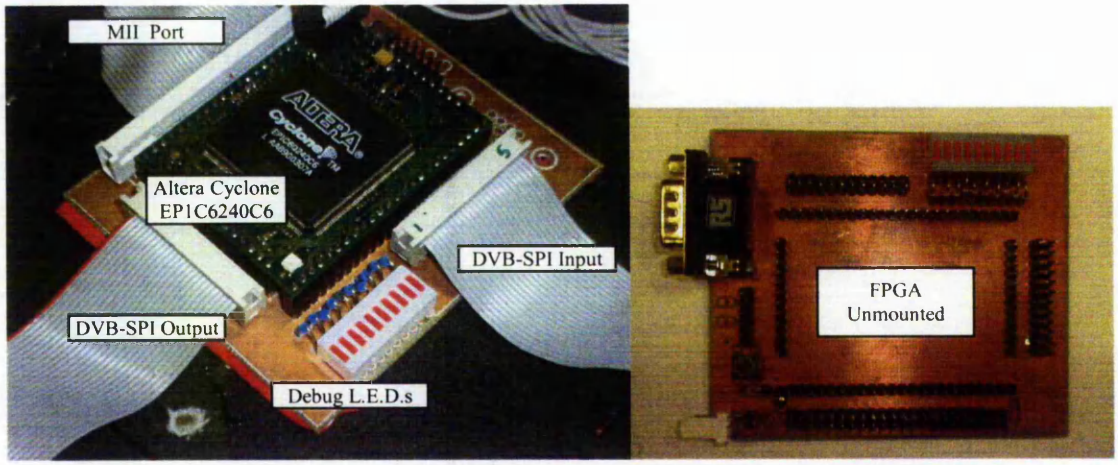
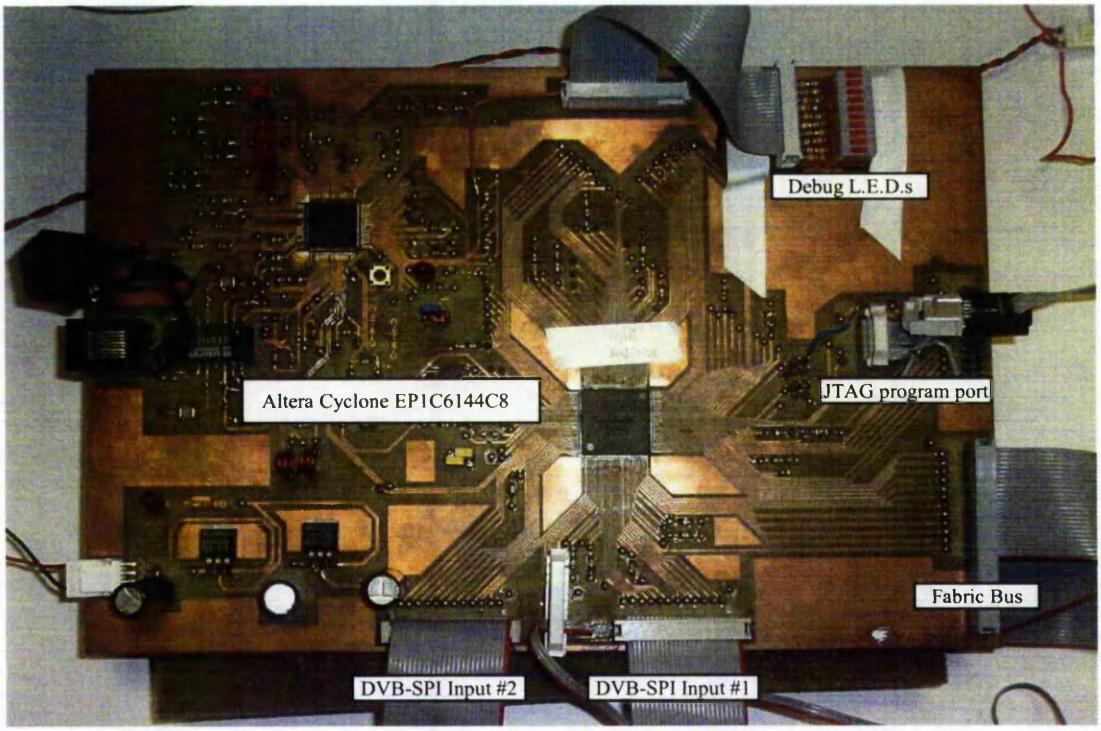


Figure B-2: Byte Value distribution of Secure Socket Layer (SSL) web content

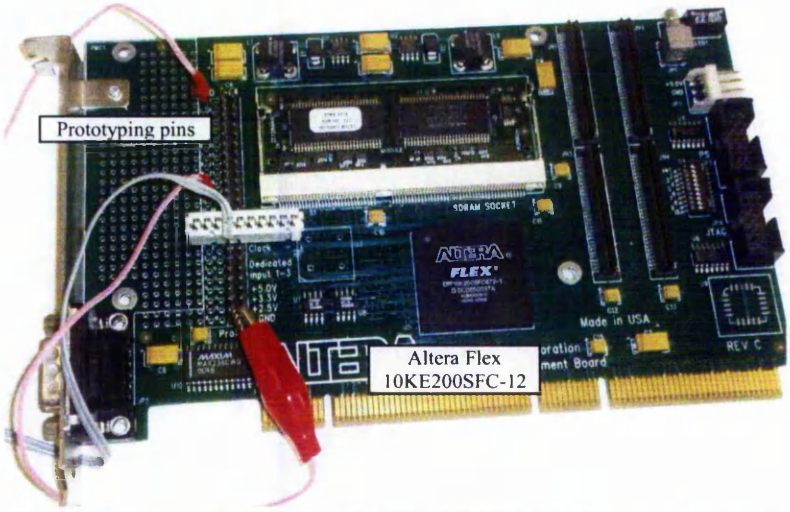
Appendix C: Photographs of Hardware



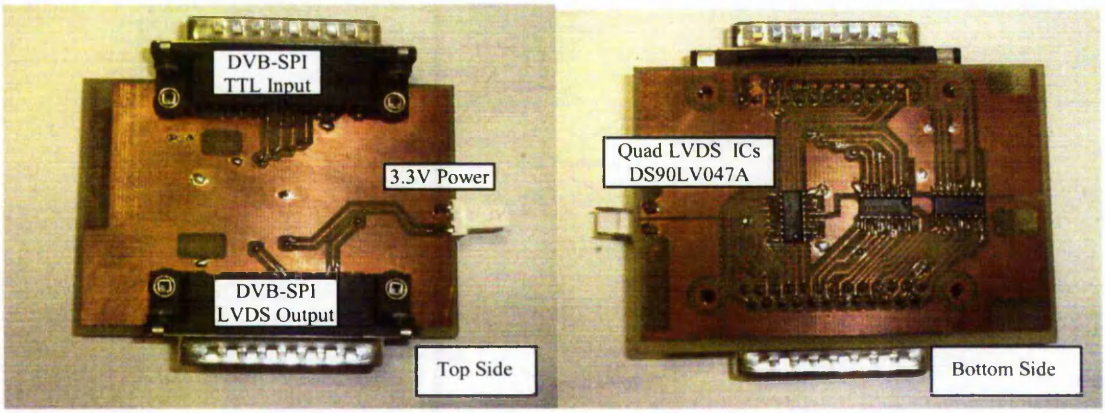
Photograph C-1: FPGA Development boards for Subscriber and Base Station Downlink Interfaces (one shown)



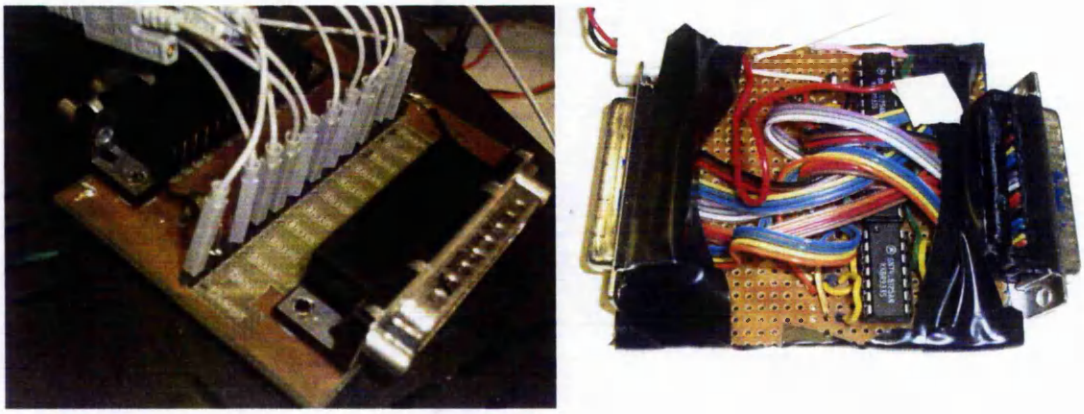
Photograph C-2: Base Station Uplink FPGA Development board



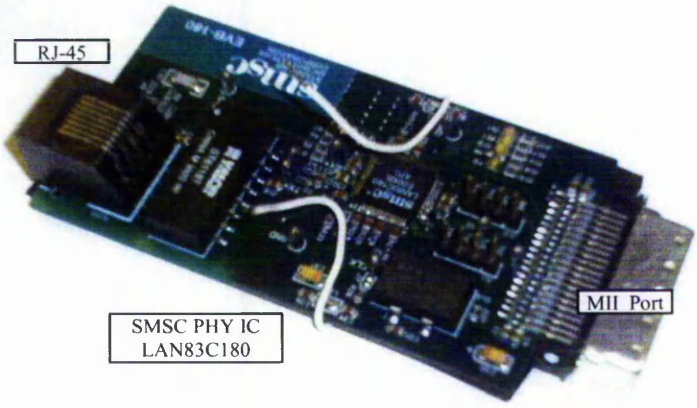
Photograph C-3: Altera FLEX 10KE Development board



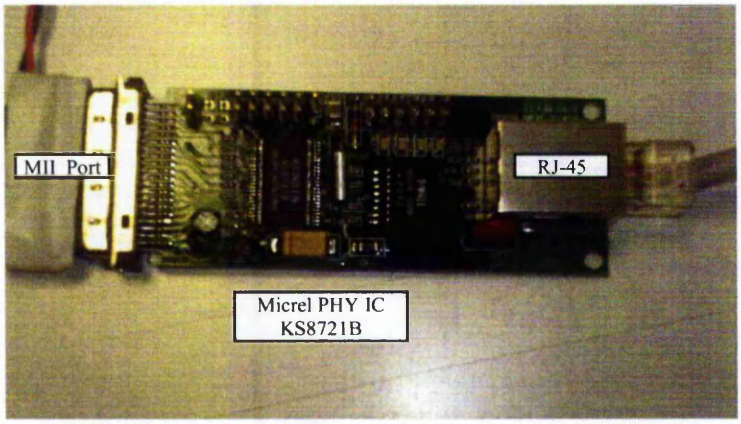
Photograph C-4: 'TTL to LVDS' DVB-SPI converter module



Photograph C-5: DVB-SPI Probe Module (left); DVB-SPI Isolator (right)

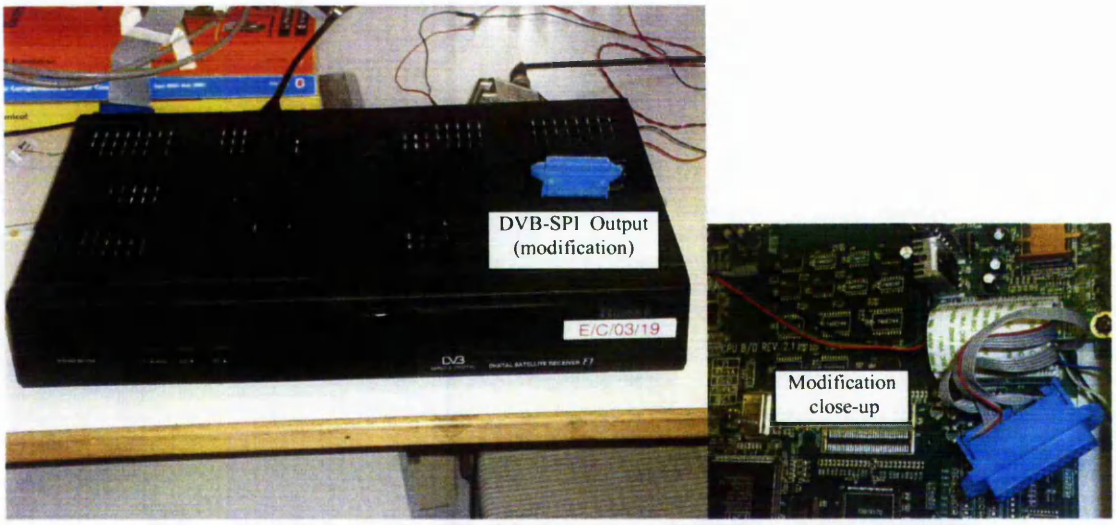


Photograph C-6: SMSC MII to Ethernet module



Photograph C-7: Micrel MII to Ethernet module

Photograph C-8: DVB-S Modulators



Photograph C-9: Modified DVB-S Set-Top-Boxes (one shown)

Photograph C-10: Digital Logic Analyzer test Instrument

Photograph C-11: 802.11g WiFi Access Points

Appendix D: Ping Calibration Measurements

```
[root@localhost root]# ping 152.71.10.64 -c 200 -i 0.15
PING 152.71.10.64 (152.71.10.64) from 152.71.10.34 : 56(84) bytes of
data.
64 bytes from 152.71.10.64: icmp_seq=1 ttl=64 time=5.00 ms
64 bytes from 152.71.10.64: icmp_seq=2 ttl=64 time=4.99 ms
64 bytes from 152.71.10.64: icmp_seq=3 ttl=64 time=4.98 ms
64 bytes from 152.71.10.64: icmp_seq=4 ttl=64 time=4.99 ms
64 bytes from 152.71.10.64: icmp_seq=5 ttl=64 time=4.95 ms
                                    .
                                    .
                                    .
64 bytes from 152.71.10.64: icmp_seq=196 ttl=64 time=4.91 ms
64 bytes from 152.71.10.64: icmp_seq=197 ttl=64 time=4.90 ms
64 bytes from 152.71.10.64: icmp_seq=198 ttl=64 time=5.00 ms
64 bytes from 152.71.10.64: icmp_seq=199 ttl=64 time=5.02 ms
64 bytes from 152.71.10.64: icmp_seq=200 ttl=64 time=4.99 ms

--- 152.71.10.64 ping statistics ---
200 packets transmitted, 200 received, 0% loss, time 30333ms
rtt min/avg/max/mdev = 4.849/4.958/5.070/0.054 ms
```

Figure D-1: Linux console screenshot for Ping Measurement (Tp)

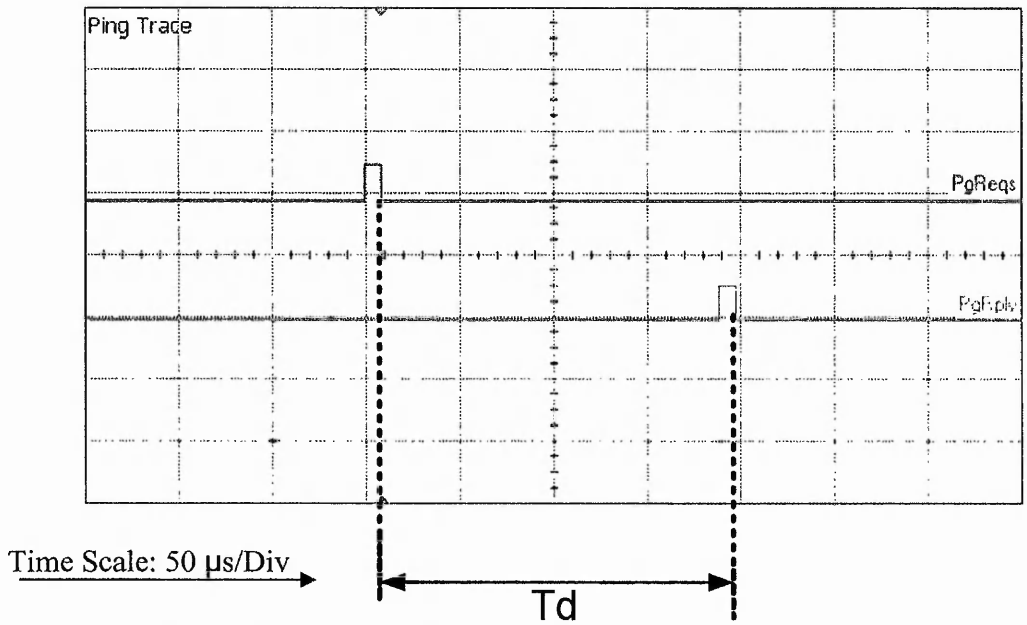


Figure D-2: Logic Analyzer screenshot for Ping Measurement (Td)

	72 Bytes		500 Bytes		1000 Bytes		1526 Bytes	
Exp.	Ping (ms)	Logic Analyzer (ms)	Ping (ms)	Logic A. (ms)	Ping (ms)	Logic A. (ms)	Ping (ms)	Logic A. (ms)
1	0.271	0.186401	0.493	0.360055	0.771	0.592157	0.994	0.775603
2	0.278	0.19025	0.493	0.368978	0.751	0.573502	1	0.775464
3	0.281	0.191357	0.503	0.384774	0.749	0.568851	1	0.774222
4	0.273	0.182696	0.51	0.377291	0.749	0.571889	1.01	0.78184
5	0.269	0.186329	0.5	0.369992	0.748	0.56832	1	0.779079
6	0.29	0.201705	0.501	0.37422	0.746	0.570557	1	0.778545
7	0.278	0.188305	0.492	0.366324	0.749	0.5725	1.01	0.784725
8	0.259	0.174565	0.5	0.370223	0.753	0.579747	1	0.778736
9	0.279	0.193997	0.503	0.375973	0.749	0.57051	1	0.773617
10	0.265	0.182782	0.514	0.38765	0.756	0.579315	1.01	0.78466
11	0.28	0.197408	0.499	0.372395	0.758	0.57547	1	0.772209
12	0.275	0.190582	0.493	0.368447	0.759	0.572406	1.01	0.777652
13	0.267	0.179743	0.505	0.378173	0.769	0.581825	1	0.773153
14	0.277	0.186578	0.473	0.356413	0.75	0.573808	1.01	0.780313
15	0.276	0.184855	0.487	0.358336	0.76	0.592243	1.02	0.77978
16	0.266	0.186011	0.499	0.375543	0.757	0.575714	1	0.780118
17	0.278	0.19222	0.5	0.371784	0.746	0.578309	1	0.779362
18	0.279	0.19238	0.487	0.358812	0.753	0.572268	1	0.779949
19	0.268	0.180829	0.5	0.371184	0.764	0.585495	1.01	0.786459
20	0.294	0.207629	0.516	0.38874	0.76	0.5829	1.01	0.78671
21	0.293	0.206522	0.498	0.368172	0.746	0.577394	1	0.777356
22	0.301	0.214517	0.492	0.366786	0.748	0.577071	1.01	0.780642
23	0.277	0.191759	0.5	0.372169	0.751	0.572457	1.01	0.777248
24	0.269	0.183907	0.495	0.372646	0.751	0.570436	1	0.77673
25	0.272	0.189242	0.512	0.383417	0.765	0.581404	1	0.781292
26	0.276	0.189642	0.498	0.371402	0.751	0.572752	1	0.777668
27	0.271	0.185238	0.507	0.378177	0.763	0.581482	1.01	0.783595
28	0.272	0.187639	0.502	0.371237	0.754	0.573918	1.01	0.781083
29	0.277	0.190058	0.499	0.370403	0.738	0.561836	0.997	0.76655
30	0.274	0.189311	0.505	0.375411	0.759	0.579658	1	0.779949

Table D-1: Recorded Logic analyzer vs. Ping Measurement (Td vs. Tp)