AAL2 Signalling Framework to Support a Gigabit AAL2 Switching Node

Prepared by:
André van Zyl

Supervised by:
Neco Ventura

Department of Electrical Engineering
University of Cape Town
2003

This dissertation is submitted to the University of Cape Town in fulfilment of the academic requirements for the degree of Masters of Science in Engineering

15 February 2003

Declaration

I declare that this thesis is my own work. Where collaboration with other people has taken place, or material generated by other researchers is included, the parties and/or material are indicated in the acknowledgements or references as appropriate.

This work is being submitted for the Masters of Science Degree in Electrical Engineering at the University of Cape Town. It has not been submitted to any other university for any other degree or examination.

_____ _____

André van Zyl Date

Acknowledgements

This research could not have been possible if not for the support of the following people:

Mr. Neco Ventura, who initiated and guided this project. Thank you for your advice throughout the duration of this study. Your dedication and enthusiasm towards your work is truly admirable.

Sven Shepstone, who developed and implemented the first phase of the AAL2 project. Thank you for your expertise in so many aspects of this project.

Dale How and Stuart Doyle, who provided technical support for the implementation phases of this study. Thanks guys for your practical help and advice during the many lines of code.

Gavin Teague, who proof read this thesis. Gav, you're a machine, thanks for all the help in this regard.

To my wife Thia. Thank you for all your love, support and prayers throughout the duration of this thesis. I owe so much to you for always being there for me.

Finally, to my Lord and Saviour. Thank you for Your guidance as well as Your unfailing love and support during my many years of study.

Synopsis

Although Asynchronous Transfer Mode (ATM) Adaptation Layer type 2 (AAL2) has only been in existence for a few short years, it is widely being adopted as the technology of choice in VoDSL, VTOA trunking, as well as in UMTS wireless networks. The two most important concerns when transporting voice in a packet based network, are end-to-end delay and the efficient use of available bandwidth.

AAL2 transports voice in the form of small, variable length packets, called CPS packets. Each packet is fitted with a unique identifier referred to as the CID. Multiple CPS packets, from various sources, are then multiplexed into a single ATM cell in order to reduce the packetisation delay. Although this method utilises the local loop more efficiently, it also introduces a drawback when not all the CPS packets are intended for the same destination. This problem can be addressed by introducing an additional switching level on top of ATM, called AAL2 switching. Thus switching of AAL2 CPS packets will be performed based upon the CID values of individual AAL2 CPS packets. This switching technique utilises the core network resources more efficiently. The device capable of achieving this switching functionality is called an AAL2 switching node.

However, ATM is a virtual circuit based technology and requires the establishment of an end-to-end connection prior to the transfer of any user traffic. As a result, prior to two AAL2 users transmitting traffic, an AAL2 connection first needs to be established. This process of connection setup is done using the procedures of signalling. AAL2 signalling (Q.2630.1) was recently standardised by the ITU-T and provides the ability to dynamically establish, maintain and release AAL2 connections over existing ATM virtual circuits. The combination of the AAL2 signalling and switching components would result in the development of an AAL2 switching node, which in practise would form part of a hybrid ATM / AAL2 network.

Research commenced into the design of an AAL2 switching node at the University of Cape Town in the year 2000. An experimental ATM research switch, called the Washington University Gigabit Switch (WUGS) is being utilised to implement this AAL2 switching node design. The complete design will comprise three distinct yet integrated components, namely an AAL2 user-plane switching component that performs the underlying switching of individual CPS packets, an AAL2 control-plane signalling component that is responsible for the establishing and releasing of end-to-end AAL2 connections and finally a management-plane component.

This study is concerned with developing and integrating a modular AAL2 signalling protocol framework, which resides in the control-plane of the AAL2 switching node and on various AAL2 end-points. In order to evaluate the functionality and performance of the signalling framework, two evaluation platforms were proposed and implemented. Firstly, the signalling framework was integrated into a modified AAL2 transport entity to emulate the operation of two peer AAL2 end-point nodes establishing and tearing down AAL2 channels between them. Secondly, the signalling framework was integrated into the previously developed AAL2 switching node to perform the signalling functionality. The second evaluation platform thus comprises three AAL2 nodes, namely two end-points and a switching node.

The results obtained from conducting a set of experiments on both evaluation platforms will be presented. Particular focus will be placed on the scalability and efficiency of the developed AAL2 signalling protocol framework. Scalability refers to the ability of the system to process multiple simultaneous AAL2 connections, whereas efficiency implies the signalling delays required to establish new AAL2 connections.

Table of Contents

List of Figures

List of Tables

Glossary of Terms and Acronyms

A2S	AAL2 Signalling Stack
AAL1	ATM Adaptation Layer 1 defined in IUT-T I.363.1. The type of ATM adaptation principally used for circuit emulation services over an ATM network.
AAL2	ATM Adaptation Layer 2 defined in ITU-T I.363.2. A new type of ATM adaptation used for variable bit-rate services.
AAL5	ATM Adaptation Layer 5 defined in ITU-T I.363.5. The type of ATM adaptation principally used for frame and packet transport over an ATM network.
ATM	Asynchronous Transfer Mode. A cell-relay service that transports multiple traffic types over a common communications medium.
CBR	Constant Bit Rate
CAS	Channel Associated Signalling. Also known as In-Band signalling
CCS	Common Channel Signalling. Also known as Out-of-Band signalling
CPE	Customer Premises Equipment
CPS	Common Part Sub-Layer
ECF	Establish Confirm
ERQ	Establish Request
FPX	Field Programmable Port Extender
GST	Generic Signalling Converter
ITU-T	International Telecommunication Union-Telecommunication Standardisation Sector. An international standards organisation of more than 150 countries, founded in 1865, which sets communications standards.
Kbps	Kilobits per second
ms	Milliseconds
PCM	Pulse Code Modulation. The basic modulation scheme for transporting voice channels in 64 Kbps time slots.
PSTN	Public Switched Telephone Network
PVC	Permanent Virtual Circuit. A permanently configured or static ATM virtual connection.

QoS	Quality of Service
REL	Release Request
RLC	Release Confirm
rt-VBR	Real Time-Variable Bit Rate
SPC	Smart Port Card
SVC	Switched Virtual Circuit. A dynamically allocated ATM virtual connection, established by signalling, as needed by the end-user devices.
SSCS	Service Specific Convergence Sub-Layer
STC	Signalling Transport Converter
TDM	Time Division Multiplexing. The technology of the traditional PSTN, designed for voice.
UBR	Unspecified Bit Rate
UNI	User Node Interface
UMTS	Universal Mobile Telecommunication System
VBR	Variable Bit Rate
VC	Virtual Circuit or Virtual Connection
VCI	Virtual Circuit Identifier
VPI	Virtual Path Identifier
VoDSL	Voice over Digital Subscriber Line
VTOA	Voice and Telephony Over ATM. Applications that transports voice services, including value-added features, over an ATM infrastructure. Also a working group of the ATM Forum.
VXT	VPI/VCI Translation Table
WUGS	Washington University Gigabit Switch

Chapter 1

Introduction

1.1 Background Information

Telecommunications: the Oxford English Dictionary describes this word as "communicating over a distance" and for many years voice was the predominant traffic type being transported over public telecommunication networks. However, with the rapid increase of data traffic, newer network types such as the public Internet, corporate intranets and virtual networks are becoming dominant. Although the rate of data traffic growth is 10 times greater than that of voice traffic, voice still accounts for between 80 to 90 percent of telecommunications service providers' revenue [1]. Voice and data traffic have traditionally been transported over two distinct networks. Voice is predominantly transported over a circuit switched Time Division Multiplexing (TDM) based network, while data is predominantly transported over a packet based network. However, the current trend in the telecommunication industry is towards a convergence of these two network types into a single efficient and reliable network. This converged telecommunications network will be a packet based network with the ability to integrate different types of traffic such as voice, video and data seamlessly.

1.1.1 Disadvantage of Traditional Voice Networks

Voice has traditionally been transported by means of a TDM technique in which a user is assigned a specific time slot during which voice traffic can be transmitted. This time slot allocation technique allows multiple calls to be placed or multiplexed into a single Pulse Code Modulation (PCM) frame. Whenever an ordinary voice call is made over the Public Switched Telephone Network (PSTN), the analogue voice signal is

first converted to a digital format. This is accomplished through sampling the analogue voice at 8 000 times per second. Each sample is an 8-bit digital representation resulting in a 64 Kbps digital signal. This PCM digital signal can then be switched and routed in the telephone network. The digital voice signal is then converted back to analogue form before being presented at the caller's destination. The digitised voice from a specific user is transported in a particular time slot of the PCM frame for the duration of the call. Even during moments of silence, when the connection between the two calling parties is temporarily idle, that time slot is still statically dedicated to the user. Transmission and switching resources in the TDM network are thus wasted during this idle period, which amounts to approximately 50 to 60 percent of the total connection time [2].

In a packet based network, this waste of resources is eliminated by dynamically allocating transmission and switching resources as required by the user. Idle traffic transmissions are thus eliminated. Traffic information is divided into packets, each of which is handled as a separate connection. Each packet is then fitted with a header that indicates where the packet originated from, what its destination is and other relevant information, to be interpreted by the network nodes along the route. Voice traffic can thus be placed into packets and transported over a packet based network more efficiently than over a traditional TDM network due to the multiplexing gains.

ATM, which is a packet based network, is an ideal candidate for the converged network since the nature of a user application can be matched to the specific features of an ATM network.

1.1.2 Asynchronous Transfer Mode (ATM)

Asynchronous Transfer Mode (ATM) is a cell-switching and multiplexing technology that combines the benefits of circuit switched TDM networks, guaranteed capacity and constant delay, with those of packet switching, flexibility and efficiency. Due to the inherent limitations of traditional TDM networks, ATM was standardised by the International Telecommunication Union-Telecommunication Standardisation Sector (ITU-T) to support converged service types. Although ATM was originally conceived

as a high speed transfer technology for voice, video and data over public networks, the ATM Forum extended the ITU-T's vision of ATM for use in both public and private networks [3]. ATM is connection-oriented meaning that a logical end-to-end connection, called an ATM Virtual Connection (VC), must first be established between sender and receiver, prior to the transmission of any user data. ATM transports data in small, fixed-sized packets called cells. An ATM cell is 53 bytes in length, with a 5 byte header and a 48 byte payload.

ATM is based on the efforts of the ITU-T Broadband Integrated Service Digital Network (B-ISDN) standard and has the ability to integrate multiple traffic types, such as voice, video or data onto a single medium. In contrast to Internet Protocol (IP) and Frame Relay, ATM uses short fixed-length cells that can be switched in hardware instead of software, making it less susceptible to network delays. Thus, the time delays required to firstly fill cells with user traffic, such as variable bit-rate voice, and secondly to transmit the user traffic is significantly reduced. This allows ATM to provide a guaranteed Quality of Service (QoS) to user applications. The amount of bandwidth required by a user application can be provided without adversely impacting on the performance of other applications on the network.

For these reasons, ATM has emerged as the technology of choice for transporting voice over a packet network. Chapter 2 will discuss some of the technical challenges involved in transporting voice over a packet based network.

1.1.3 ATM Reference Model

The ATM architecture uses a logical model to describe the functionality it supports. The ATM reference model, as illustrated in Figure 1-1, consists of the following planes, which cover all layers:

- **User Plane**

  The User Plane is responsible for managing the transfer of any user traffic originating from higher layer applications.

- **Control Plane**

  This plane is responsible for generating and managing signalling requests, which are required to establish, maintain and tear-down ATM VCs.

- **Management Plane**

  This plane contains two components:

  - Layer Management manages layer specific functions, such as the detection of failures and protocol related problems.
  - Plane Management manages and coordinates functions related to the entire system.

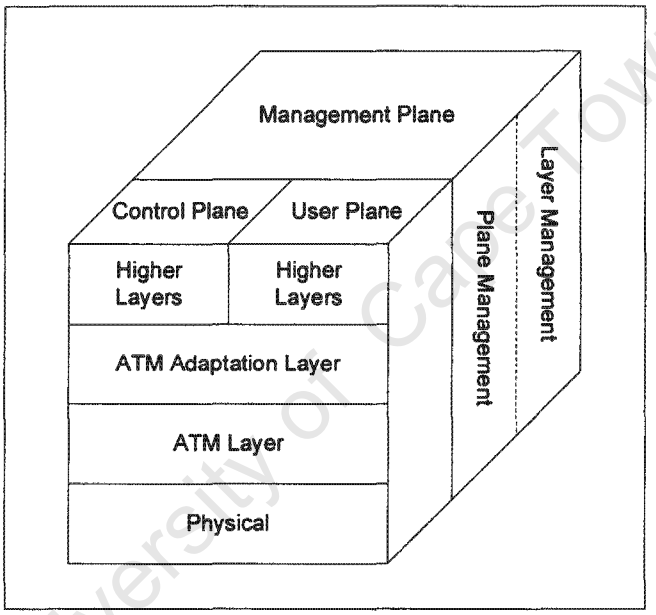

Figure 1-1 ATM Reference Model

The ATM reference model is also composed of the following ATM layers:

- **Physical Layer**

  The ATM physical layer is analogous to the physical layer of the well known OSI reference model. It manages issues relating to the transmission of data on the physical transmission medium.

- **ATM Layer**

  Combined with the ATM adaptation layer, the ATM layer is roughly analogous to the data link layer of the OSI reference model. The ATM layer is

responsible for establishing connections and transporting the ATM cells through the ATM network, based on information contained in the ATM cell header.

- **ATM Adaptation Layer (AAL)**
  The AAL is responsible for isolating higher layer protocols from the details of the ATM processes. It also segments user application data into small fixed size payloads of 48 bytes at the sender, and reassembles these payloads at the receiver.

It is important to understand that not all the layers of the ATM reference model are applicable to all ATM network elements. ATM switches located in the network only consist of the ATM and Physical layers, whereas network elements located at the Customer Premises Equipment (CPE) consist of all the ATM layers since user applications are directly supported. In order to gain a better understanding of the ATM technology, its architecture and its practical implementations, the reader is advised to read [3] and [4].

1.1.4 ATM Adaptation Layer (AAL)

In order for an application's traffic to be carried over an ATM network, the application must first be adapted to ATM transport. The ATM Adaptation Layer (AAL) provides this mapping service, at the edge of the network, between the service required by the application and the service offered by the ATM network. The AAL performs functions required by the user, control and management planes, of the ATM reference model. Due to the variety of service types that can be supported by ATM, several AAL types have been standardised by the ITU-T. The most commonly implemented are AAL1 [5] and AAL5 [6]. AAL1 supports connection-oriented services and is used primarily for handling circuit emulation applications, such as voice and video conferencing. AAL5 is used principally for transporting packet data, such as Frame Relay or Internet Protocol (IP), over an ATM network and supports both connection-oriented and connectionless data [3].

1.1.5 Voice over ATM Adaptation Layer type 1 (AAL1)

Voice has traditionally been transported over ATM as uncompressed 64 Kbps Circuit Emulation Service (CES), utilising AAL1[1]. AAL1 provides a Constant Bit Rate (CBR) service for isochronous[2] traffic, thus making it ideally suited for circuit emulation. Circuit Emulation provides a transparent transport mechanism for leased-lines TDM voice traffic over an ATM network, as illustrated in Figure 1-2. An advantage of this technique is that no change is required to the existing TDM or PBX network.

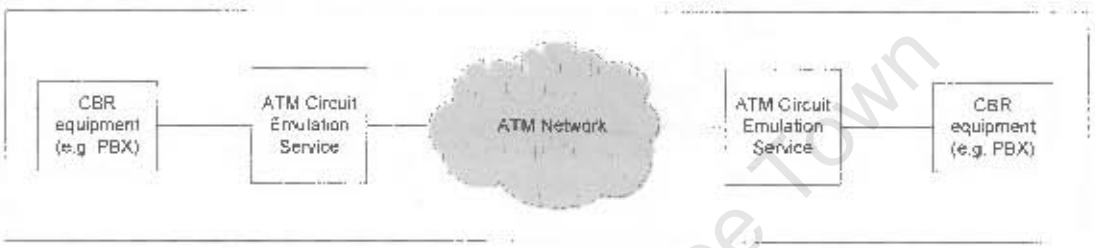

Figure 1-2 Circuit Emulation Framework

Two circuit emulation techniques are available:

- **Unstructured Circuit Emulation**

  This technique establishes a full T1/E1 (1.544 Mbps / 2.048 Mbps) or T3/E3 (44.736 Mbps / 34.368 Mbps) emulated circuit between two points in the network. This kind of connection would typically carry end-to-end transparent circuits between TDM voice switches or digital Public Branch Exchanges (PBXs).

- **Structured Circuit Emulation**

  This technique enables emulated circuits to be established at the level of *N* x 64 Kbps. It is therefore not necessary to dedicate a full T1/E1 link across the ATM network.

[1] As defined by the ATM Forum in af-vtoa-0078.000
[2] Delay sensitive

Although voice over ATM, through either method of AAL1 circuit emulation, is simple to deploy, it is bandwidth inefficient since it utilises a constant bit-rate service. This implies that cells will continue to be transported on the ATM virtual circuit even when there is no data to be sent [7]. An analogy can be drawn between AAL1 circuit emulation and the functionality of a conveyer belt connected transparently through the ATM network between the two ATM Service Emulation Service blocks of Figure 1-2. This conveyer belt is exclusively reserved for a single dedicated user. Therefore, during periods when no user traffic is available for transportation from the single user, other users are unable to utilise the dedicated ATM VC, or conveyer belt in the analogy.

The ATM Forum addressed this limitation of CES and developed a new standard called Dynamic Bandwidth Circuit Emulation Service (DBCES)[3]. This standard enables dynamic bandwidth utilisation by detecting which time slots of a TDM circuit are active and which are inactive. When an inactive state is detected in a specific time slot, the time slot is dropped from the next ATM circuit emulation data structure and the bandwidth it was using may be re-allocated to other services. While this technique removes inactive time slots from an ATM circuit emulation data structure, the utilisation of AAL1 for the transportation of voice over ATM remains inefficient, since neither method of circuit emulation exploits the statistical multiplexing advantage of ATM.

Circuit emulation also requires additional overhead in the ATM cell and AAL1 layer. The connection therefore requires 12 to 15 percent more bandwidth than that of the actual circuit being transported [1]. Further, AAL1 suffers from unacceptable packetisation delay when applied to low-bit-rate, compressed voice traffic. For example, using AAL1, only 6 ms is required to fill a 48 byte ATM cell payload at 64 Kbps voice. However, if the voice is compressed to 8 Kbps, the packetisation delay increases to 48 ms [8]. Significant advances in voice compression technology in the last decade have necessitated the need for a new AAL type.

[3] As defined by the ATM Forum in af-vtoa-0085.000

Another limitation of AAL1 is observed in mobile networks where low-bit-rate compressed voice traffic is a common occurrence. The mobile terminal itself performs some form of voice compression before the voice traffic is transmitted to the mobile network. If an AAL1 circuit is utilised to transport this mobile connection in the fixed portion of the mobile network, the compressed voice first needs to be uncompressed to 64 Kbps before entering the AAL1 circuit. This is required since AAL1 does not support any compressed voice. At the exit point of the AAL1 circuit the uncompressed voice traffic is again compressed to the original format before being delivered to the remote mobile terminal. These conversions are clearly inefficient as it introduces additional transmission delays to the delay sensitive voice traffic.

1.1.6 Evolution of Voice over ATM Adaptation Layer type 2 (AAL2)

AAL1, which utilises a fixed size cell, has limitations when applied to small data packets. A method using partially filled AAL1 cells could be used to reduce the end-to-end transfer delay, but this method would be bandwidth inefficient. Chapter 2 will discuss some of the primary transfer delay requirements concerning voice. A second approach utilising AAL1, could be to allow multiple small, fixed size packets originating from different users, to be multiplexed into a single 48 byte ATM cell payload. For example, 16 users could be allowed to send packets of 3 bytes each to be packetised into the ATM cell payloads of a single ATM connection. The position of the small 3 byte packets within the payload of the ATM cell would provide a method by which each individual user's packets can be identified. This identification method is referred to as position-based multiplexing and delineation [9]. This method could serve a maximum of 48 users per ATM connection, with each user sending packets of 1 byte each to be placed in the ATM cell payloads. While this approach is efficient, it suffers from a lack of flexibility. It is only effective for constant bit-rate applications and a low number of users, but voice is Variable Bit Rate (VBR) by nature. However, if each user employs a different voice codec, this approach will create significant packetisation inefficiencies within the fixed ATM cell payload. If variable bit-rate voice codecs with silence suppression are employed by the users, then either variable length packets would be generated or variable packetisation delays would be incurred

while generating fixed length packets. In either case, the position-based multiplexing method is inflexible and cannot be applied with ease.

The primary need for a new AAL layer is to provide a mechanism to transport small data packets over an ATM network efficiently, while ensuring that the transfer delay across the ATM network is kept to a minimum. In 1997 the ITU-T addressed this need and standardised AAL type 2 to provide bandwidth efficient transmission of low-bit-rate, short and variable length packets for delay sensitive applications [10]. The objective of AAL2 is to multiplex several low-bit-rate voice channels, each using a small bandwidth and originating from multiple users, into a single ATM virtual connection. This is accomplished by allowing multiple small user data packets, called AAL2 CPS packets[4], to be multiplexed into a single ATM cell, thus limiting the packetisation delay for the compressed voice channels carrying delay sensitive applications. Chapter 2 will discuss the AAL2 cell structure in more detail to illustrate how multiple AAL2 CPS packets are placed inside an AAL2 cell.

AAL2 supports the VBR class of service that enables statistical multiplexing for the higher layer requirements demanded by voice applications, such as voice compression, silence detection and suppression and idle channel removal. AAL2 has since been used extensively for Loop Emulation Service (LES), by trunking compressed voice as a cost effective alternative to AAL1. As a summary, Table 1-1 provides a brief comparison between AAL1 and AAL2.

AAL1	AAL2
Single user channel per ATM VC	Multiple user channels on a single ATM VC
Bandwidth is used even when there is no traffic	Bandwidth efficient and supports statistical multiplexing
Only supports a CBR class of service, suitable for circuit-switching	VBR ATM class of service, suitable for packet switching
No support for voice compression, silence suppression, idle channel removal	Extensive standardisation to provide support for voice compression, silence suppression, idle channel removal

Table 1-1 Comparison between AAL1 and AAL2

[4] Originally called mini-cells

1.2 Problem Statement

AAL2 was originally intended for trunking[5] purposes, of which a simplistic example is illustrated in Figure 1-3. Various low-bit-rate voice channels, in the form of AAL2 cells, would first be multiplexed at concentrator 1 before being passed unaltered via ATM switches, A and B, to their destination at concentrator 2. In this figure a channel set is simply a set of individual voice channels destined for the same destination.

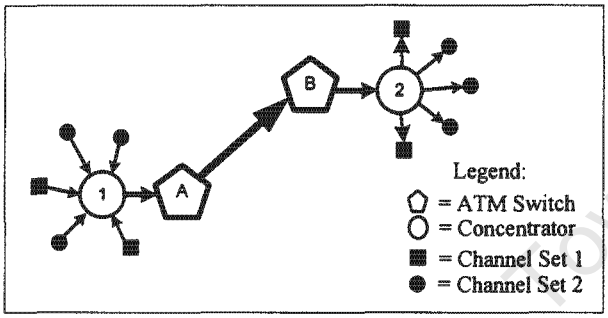

Figure 1-3 Simple AAL2 Trunking Scenario

However, this approach has a bandwidth inefficiency limitation, which needs to be addressed. Figure 1-4 illustrates a configuration of five ATM switches, three AAL2 concentrators and various voice sources and receivers. AAL2 concentrator 1 receives five different voice channels to be multiplexed for transmission. The channels have been arbitrarily divided into two groups, to emphasize two different destinations for the various voice channels.

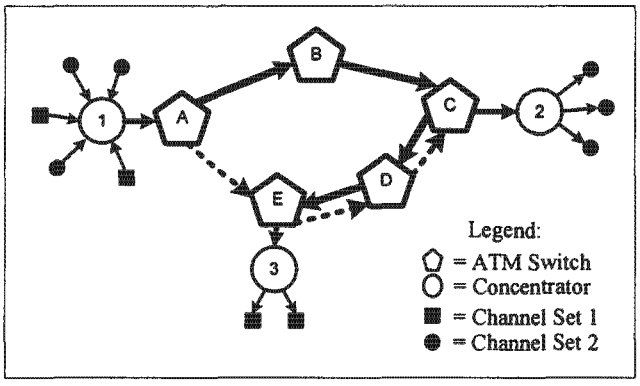

Figure 1-4 Complex AAL2 Trunking Scenario

[5] The tunnelling of voice traffic across a network between two fixed points.

Three channels are destined for concentrator 2 and the remaining two channels are destined for concentrator 3. With AAL2 trunking, a decision must be made at ATM switch A: either switch the five voice channels along the solid line from switch A to switches B, C, D and finally to switch E, thus undergoing four ATM switch hops, or alternatively traverse the dotted line resulting in a total of three hop counts. Regardless of which route is taken, each concentrator receives five voice channels and ends up discarding at least two.

This trunking scenario depicted in Figure 1-4 is far more likely to appear in a typical ATM network than the idealised trunking example of Figure 1-3, where each and every voice packet carried on the entire route is intended for the same destination. This problem can be addressed by introducing an additional switching level on top of ATM, called AAL2 switching.

AAL2 transports voice in the form of small, variable length packets, called CPS packets. Each packet is fitted with a header that contains a Channel Identifier (CID). AAL2 switching will thus be able to switch these small CPS packets based on the CID value of the individual AAL2 CPS packet [11]. Where conventional ATM switching is performed based upon the VPI / VCI value contained in the ATM cell header, AAL2 switching will be performed based upon the VPI / VCI / CID values of individual CPS packets. This switching technique utilises the core network resources more efficiently. The device capable of achieving this switching functionality is called an AAL2 switching node.

Research commenced on the design of an AAL2 switching node at the University of Cape Town in the year 2000. An experimental gigabit ATM research switch, called the Washington University Gigabit Switch (WUGS) is being used to implement this AAL2 switching node design[6]. The current design comprises of two distinct components, namely an AAL2 user-plane switching component and an AAL2 control-plane signalling component. The user-plane switching component, which has been successfully implemented, performs the underlying switching of individual AAL2 CPS user packets [12].

[6] More information regarding the gigabit ATM switch can be found at http://www.arl.wustl.edu/gigabitkits

1.3 Objective of this Thesis

This study will focus on developing a control-plane signalling component to provide the necessary signalling functionality required for establishing, maintaining and tearing down end-to-end AAL2 connections. Focus will also be placed on integrating the signalling component with the already developed user-plane switching component as discussed in [12]. ATM is a virtual circuit based technology and requires the establishment of an end-to-end connection prior to the transfer of any user traffic. As a result, prior to two AAL2 users transmitting traffic, an AAL2 connection first needs to be established. This process of connection setup is performed using signalling procedures. AAL2 signalling was recently standardised by the ITU-T and provides the ability to dynamically establish, manage and release AAL2 connections over existing ATM virtual circuits [13]. This standard is often viewed as an extension of normal ATM signalling, but this is not the case. However, AAL2 signalling does require the existence of a pre-established ATM VC, provisioned by ATM signalling, prior to AAL2 signalling creating any AAL2 connections within the ATM VC.

In this study we will develop an AAL2 signalling protocol framework, based on various available standards, which will reside in the control-plane of an AAL2 switching node and on various AAL2 end-points. The AAL2 signalling protocol framework is responsible for dynamically establishing new AAL2 channels, end-to-end, without negatively impacting on existing AAL2 channels, as well as maintaining those channels and tearing them down. Not only should the AAL2 signalling framework perform correctly, it should also be scalable, in terms of the number of simultaneously connections it is able to support, and efficient, in terms of signalling delays. Therefore, various design techniques will be evaluated to determine how the critical AAL2 signalling delay, associated with AAL2 channel establishment, could be decreased. The process required to integrate the AAL2 signalling framework and the AAL2 switching module, which resides on both the AAL2 switching node and the AAL2 end-points, will also be presented and discussed. Of particular interest will be to determine how well the signalling framework for both the end-points and the switch node scale with an increase in the number of users. Designing and implementing the AAL2 signalling framework will form the second phase of a three phase project aimed at designing a fully functional AAL2 switching node, which

would form part of a hybrid ATM / AAL2 network. The first phase of the project was designing and implementing the AAL2 switching component, which has already been completed and presented in [12].

Although AAL2 minimises the packetisation delay when transporting voice traffic, the feasibility of deploying AAL2 switching nodes in an AAL2 network will also be investigated as they introduce an additional delay into the already critical end-to-end voice transfer delay, as will be discussed in Chapter 2.

1.4 Scope and Limitations

The scope of this study is limited to developing a functional AAL2 signalling framework for implementation on both the Washington University Gigabit Switch (WUGS) and AAL2 end-points, as well as the integration of this AAL2 signalling framework with the already developed AAL2 switching module [12]. The design of the AAL2 signalling framework was based on a modular design to ease development and isolate it from the underlying transport layer. Even though many techniques could be employed to enhance the performance of the signalling framework, this study could not address every issue that could impact the performance. AAL2 signalling is highly specialised and no commercial tools are available to aid in implementing a suitable design. To date no commercial AAL2 switching node is available.

As mentioned in the previous section, AAL2 signalling is unable to establish an end-to-end connection without the existence of a previously established ATM VC. This ATM VC is established via ATM signalling. This dependence of AAL2 channels on ATM channels is a potential problem because of the independence between these two signalling protocols. It is impossible for a user to request a new AAL2 connection without an already existing underlying ATM VC. The AAL2 signalling protocol is informed by the layer beneath it whether an ATM VC exists between the desired AAL2 nodes, but the AAL2 signalling protocol itself cannot request the establishment on a new ATM VC. This limitation becomes apparent when the ATM VC is congested and no more AAL2 connections can be supported. Although ITU-T recommendation Q.2963 allows ATM connections to renegotiate their bandwidths

while in the active state, this could not be implemented since no communication exists between the AAL2 signalling and ATM signalling protocols. The AAL2 signalling protocol is thus unable to request an increase in ATM VC bandwidth from the ATM signalling protocol.

Even though the AAL2 signalling protocol framework was designed to optimise the AAL2 channel establishment process, the physical hardware utilised in the implementation provided some limitations. The evaluation platform consisted of two general purpose PCs for each AAL2 end-point and a single embedded computer situated inside the WUGS. This embedded computer called the Smart Port Card (SPC) is physically situated between the link interface and the ATM switching fabric allowing it to intercept incoming cells. The SPC functions as an active processing module and contained a software implementation of both the AAL2 switching and signalling components. However, since the SPC only operates at 166 MHz its operation is slower in comparison to that of a commercial hardware implementation.

Washington University in St. Louis, who designed the WUGS switch, the SPC embedded computer as well as the accompanying network interface cards, chose NetBSD 1.4.1 as the Operating System (OS) for their hardware modules. However, NetBSD is not a real-time operating system. Therefore, the AAL2 signalling protocol framework design and, more importantly, the interface design required to integrate the AAL2 signalling framework and the AAL2 switching framework, were influenced by this limitation.

1.5 Development of Thesis

The remainder of this document is organised as follows:

Chapter 2 will start by introducing voice and discuss some of the technical challenges faced when transporting voice over a packet network. Different voice compression algorithms will also be discussed to highlight their contribution to the transfer latency experienced by transporting voice across a packet based network. An overview of the AAL2 protocol will be given to illustrate the technique of multiplexing small AAL2

CPS packets into ATM cells. This is necessary to understand the AAL2 signalling protocol presented in Chapter 3. The relationship between AAL2 switching and ATM switching will also be discussed, after which the importance of AAL2 and AAL2 switching in commercial telecommunication networks will be presented.

Chapter 3 will introduce the AAL2 signalling protocol stack, its architecture and features and investigate the need and benefits of this new signalling protocol. A comparison between AAL2 signalling and ATM signalling will also be discussed to determine similarities between these two independent protocols. The choice of service class utilised to establish an ATM VC containing AAL2 channels will also be presented and discussed. The AAL2 signalling life cycle will also be looked at, after which the chapter will conclude by discussing current unresolved challenges of the AAL2 signalling protocol.

Chapter 4 presents the design of the AAL2 signalling protocol framework and discusses the importance of minimising signalling delays associated with the AAL2 channel establishment process. Various optimisation techniques that could be utilised to accomplish this are described. It also presents various design issues that need to be considered relating to the design of the AAL2 signalling framework and presents a high-level AAL2 signalling protocol framework design for implementation on the WUGS switch and various AAL2 end-points.

Chapter 5 will discuss the process of integrating the AAL2 signalling framework into firstly, an AAL2 end-point design and secondly, into an AAL2 switching node architecture. The primary components of the end-point node will be discussed as well as the modifications required to support the signalling framework. The individual steps required to establish and end-to-end AAL2 connection between two end-point nodes are also presented, after which the integrated AAL2 switching node implementation on both the SPC and a stand-alone development station is discussed.

Chapter 6 will present and discuss results obtained from evaluating the functionality and efficiency, in terms of signalling delays, of the AAL2 signalling framework. Firstly, the framework is evaluated as a stand-alone entity to ensure functional correctness and secondly, as being integrated into an AAL2 end-point and finally as

being integrated into an AAL2 switching node. Although the link speed of each port on the WUGS switch is 1.2 Gbit/s, the bottleneck in the WUGS switch is definitely the SPC card, which contained an integrated software-based AAL2 switching and signalling module. The drawbacks that the SPC card introduce into the network will also be analysed. Finally, the feasibility of deploying AAL2 switching nodes in an AAL2 network will also be investigated.

Chapter 7 will provide a set of conclusions based on the research presented and results obtained in this study. These conclusions will be presented in the order in which the research was conducted. Important limitations will also be mentioned.

Chapter 8 will provide recommendations based on the limitations discovered during this study. Some of these are protocol related, whereas some are specific to this evaluation. Future projects involving AAL2 and AAL2 signalling are also presented.

Finally, a set of appendices are presented for completeness to provide enhanced information regarding the AAL2 signalling protocol. Development specific details are also provided.

Chapter 2

Background Theory and Literature Review of AAL2

2.1 Technical Challenges in Transporting Voice over ATM

In terms of the transportation of various network traffic types, real-time voice is one of the most challenging. The two most important concerns when transporting voice over a packet based network are end-to-end transit delay and the efficient use of available bandwidth. Some of the technical challenges associated with the transportation of voice over an ATM network that need to be addressed, are discussed below. Most of these challenges are not confined to ATM networks only, but are present in any packet based network.

2.1.1 Delay

In order to provide an acceptable QoS to users, the end-to-end transfer delay incurred in an ATM network should be minimised. The following are sources of delay in an ATM network:

- **Encoding / Decoding Delay**
  This delay occurs as a result of the voice encoding from an analogue signal to a digital format and the reverse decoding process. This typically becomes more significant at lower bit rates. Section 2.1.4 provides an overview of some of the most popular voice compression algorithms used in a packet network.

- **Packetisation Delay**

  This delay is also called cell construction delay and is the time required to fill a cell with data prior to being transmitted across the ATM network. Normal PCM encoded voice, also referred to as ITU-T standard G.711, samples are transmitted at a rate of 64 Kbps. Therefore, approximately 6 ms is required to fill a fixed size 48 byte ATM cell payload [14]. This delay is directly proportional to the level of voice compression employed, which means greater compression algorithms result in greater delay. The packetisation delay could be reduced by either transmitting partially filled cells or by multiplexing several voice packets into a single ATM virtual circuit connection (VCC). The latter is the technique that AAL2 utilises.

- **Buffering Delay**

  In order for voice to be transported across an ATM network, the voice traffic must first be segmented and placed into ATM cells for transmission. Once the cells are received at the destination, the cells are reassembled to form the original voice traffic. This segmentation and reconstruction process of ATM cells is managed by the Segmentation and Reassembly (SAR) function. The reconstruction of packets must be done in real-time to avoid any noticeable distortion by the received user. If there is a delay variation in the transmission of cells, the SAR function might not have any voice traffic to process. This is known as under-run and results in gaps during a telephone conversation.

  To prevent these gaps from occurring, the receiving SAR function accumulates a buffer of information prior to commencing the reconstruction of the voice traffic from the received cells. The size of this buffer must exceed the maximum predicted delay to ensure that no under-runs occur. However, the size of this buffer translates into delay, as each packet needs to progress through the buffer on arrival. The cell delay variation therefore needs to be strictly controlled, as it accounts for a significant portion of the total end-to-end network delay [15].

2.1.2 Echo

The increase of end-to-end delay in an ATM network deteriorates the voice quality and gives rise to echo which disrupts the normal end-to-end interactive telephone conversation.

In a 4-wire to 2-wire conversion hybrid circuit, located between a telephone handset and the communication network, the transmitted voice signal is reflected back due to an impedance mismatch between the handset and the network. This reflected voice signal results in an echo, which is transmitted to the ear piece of the telephone handset. If the end-to-end delay in the network is minimal, this echo is not detected. The maximum acceptable end-to-end delay for the transportation of voice, without the use of echo cancellation, is 25 ms. However, when echo is adequately controlled in the network, an end-to-end delay of up to 150 ms is acceptable [4]. ITU-T recommendation G.165 defines performance requirements that echo cancellers need to adhere to.

2.1.3 Silence Suppression

The characteristics of voice can be used to the advantage of making optimum use of available bandwidth. Voice, in its inherent form is variable and voice communication across a telephone network is half-duplex[7] by nature. A significant reduction in bandwidth utilisation can be accomplished by employing silence suppression at the edge of the ATM network. The normal flow of conversation, between two active users, consists of pauses between sentences and periods of silence when the corresponding user is active. These two characteristics of voice communication amount to approximately 50 to 60 percent of the total connection time [2]. Silence suppression exploits these two characteristics and prohibits the transmission of packets across the network during these silence periods.

[7] One person is silent while the other person speaks.

2.1.4 Voice Compression Algorithms

The purpose of employing a voice compression algorithm at the customer premises equipment (CPE), for the transportation of voice over AAL2, is to reduce the bandwidth required to support voice transportation. The ITU-T has defined several encoding standards to encode voice at lower bandwidths. The choice of voice codec employed is often dependent on the voice quality required, transmission delay constraint and the cost of implementation.

Table 2-1 indicates four commonly used voice encoding schemes that may be used for transporting voice over AAL2. It can be seen that as each encoding scheme is utilising larger data packets, the required network bandwidth, to transport the voice traffic, decreases and thus the network becomes more efficient. This is because the size of the packet overhead becomes less significant in larger packets. However, as the packet size increases, the packetisation delay increases correspondingly. In Table 2-1 the acronym SS refers to 50% Silence Suppression employed.

		10 Byte Packet		20 Byte Packet		30 Byte Packet		40 Byte Packet	
		Bandwidth in Kbps	Delay in ms	Bandwidth in Kbps	Delay in ms	Bandwidth in Kbps	Delay in ms	Bandwidth in Kbps	Delay in ms
G.711 at 64 Kbps	Without SS	93.8	1.25	83.0	2.5	79.4	3.75	77.3	5
	With SS	46.9	1.25	41.5	2.5	39.7	3.75	38.8	5
G.726 at 32 Kbps	Without SS	46.9	2.5	41.5	5	39.7	7.5	38.8	10
	With SS	23.4	2.5	20.7	5	19.8	7.5	19.4	10
G.728 at 16 Kbps	Without SS	23.5	5	20.7	10	19.8	15	19.4	20
	With SS	11.7	5	10.5	10	9.9	15	9.7	20
G.729 at 8 Kbps	Without SS	11.7	10	10.4	20	9.9	30	9.7	40
	With SS	5.9	10	5.2	20	5.0	30	4.8	40

Table 2-1 AAL2 Compressed Voice Encoding Algorithms

- **G.726 ADPCM at 32 Kbps**

  Adaptive Differential Pulse Code Modulation (ADPCM) is a widely used voice codec in the local PSTN particularly for international voice connections.

It is a single algorithm with relatively low delay and has a well defined robustness against tandem coding, where multiple stages of compression and decompression across the network cause progressive deterioration of the voice quality.

- **G.728 LD-CELP at 16 Kbps**

  Low Delay Code Excited Linear Prediction (LD-CELP) is a low delay, high quality voice codec operating at 16 Kbps. Although this combination of features makes it an ideal codec for transporting voice over AAL2, the cost involved in utilising this voice codec is an issue. It requires significant processing power approximately three times that needed for ADPCM 32 Kbps and is subject to royalty payments since it is based on patented technology [16].

- **G.729 CS-ACELP at 8Kbps**

  The Conjugate Structure Algebraic-Code Excited Linear Prediction (CS-ACELP) voice codec is widely used for Internet telephony. Its advantages include relatively low complexity and excellent voice quality, but its major disadvantage is delay. Unfortunately, the combination of a high encoding delay and a high packetisation delay, even when a smaller packet size is used, renders this voice codec unacceptable for most access network applications.

2.1.5 Voice Codec Mean Opinion Score (MOS)

Each voice compression algorithm listed in Table 2-1 provides a certain quality of speech and is awarded a Mean Opinion Score (MOS). Each voice codec is measured by means of a subjective test in which a wide range of listeners judge the speech quality and assigns a score to a particular voice codec. The scores range between the values 1, which is considered bad, to 5, which is considered excellent. A score of 4 and above is regarded as toll quality. The scores are averaged to provide the MOS for that voice codec. Table 2-2 indicates the relationship between the voice codec and the MOS scores [17].

Voice Codec	Bit Rate	MOS Score
G.711 PCM	64 Kbps	4.1
G.726 ADPCM	32 Kbps	3.85
G.728 LD-CELP	16 Kbps	3.61
G.729 CS-ACELP	8 Kbps	3.92

Table 2-2 Relation between Voice Codec and MOS

Although the transportation of compressed voice over a packet network is bandwidth efficient, it does exhibit two major disadvantages. Firstly, the voice codec utilised introduces additional delay, which increases as the voice codec bit-rate decreases. Secondly, the voice signal might become distorted as a result of multiple encodings. This is referred to as tandem encoding. For example, when a G.729 voice signal is tandem encoded three times, its MOS score decreases from a very good level of 3.92 to an unacceptable level of 2.68 [17].

2.2 Use of AAL2 in Transporting Voice

The evolution of voice traffic in recent years, from an almost constant bit-rate of 64 Kbps to a variable low bit-rate traffic type, has necessitated the need for a mechanism to efficiently transport voice traffic. Therefore, in order to overcome the problems of high packetisation delay and inefficient utilisation of available bandwidth when transporting voice over AAL1, the ITU-T standardised AAL2, as described in Section 1.1.6. AAL2 achieves high bandwidth efficiency and low packetisation delay simultaneously by multiplexing several low-bit-rate user information voice packets, originating from multiple users, into a single ATM cell payload. This makes it ideal for low bit-rate applications, such as compressed voice.

The AAL2 layer, as specified in ITU-T recommendation I.363.2, consists of two sub-layers, namely the Common Part Sub-layer (CPS) and the Service Specific Convergence Sub-layer (SSCS), as illustrated in Figure 2-1. The CPS layer is responsible for packing small variable length packets into ATM cells, also called AAL2 cells, and provides error correction. The SSCS functionality depends on the application to be supported. The first AAL2 recommendation, I.363.2, regarded the

SSCS sub-layer as being null. Subsequent recommendations regarding the SSCS sub-layer and its functionality will be discussed in Section 2.3.

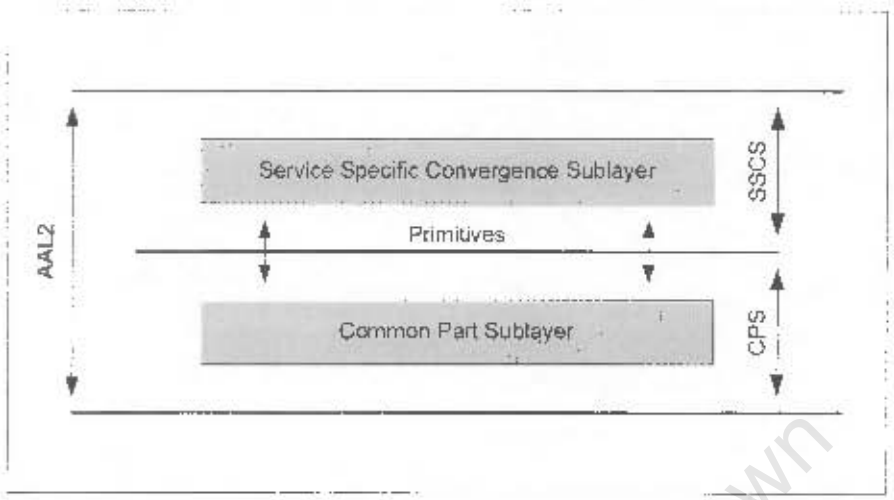

Figure 2-1 AAL2 Layer Structure

2.2.1 AAL2 CPS Packet Format

The AAL2 layer performs a two step operation in order to package information into ATM cells. For this reason, it uses two packet formats, namely the CPS packet and the CPS Protocol Data Unit (CPS-PDU). As indicated in Figure 2-2, each CPS packet consists of a three byte header and a variable length payload.

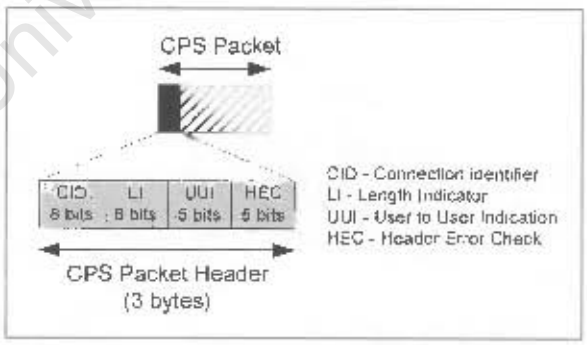

Figure 2-2 CPS Packet Format

The function of each field in the CPS packet header is as follows, starting with the leftmost field:

- **Channel Identifier (CID) Field**

  The CID field consists of 8 bits, or one byte, and is used to identify specific AAL2 users of a multiplexed channel at the AAL2 level. This 8-bit field thus enables a potential maximum number of 256 individual AAL2 user channels to be multiplexed onto a single ATM VC. However, it is important to note that not all 256 CID values are available to be used for AAL2 channel assignment. The value CID = 0 is not used for channel assignment, as it is used to activate the padding function by indicating that all subsequent objects are coded with the value zero. The value CID = 1 is used to indicate that the CPS packet is intended for peer-to-peer layer management functions and does not carry any higher layer user information. In addition, values CID = 2 to 7 are reserved for future enhancements and therefore unavailable to AAL2 users at present. The remaining values CID = 8 to 255 can be assigned to individual AAL2 users. Consequently, a maximum of 248 AAL2 user channels can be multiplexed onto a single ATM VC with each individual AAL2 channel being bidirectional. Therefore, the same CID value is used to identify both directions of the bidirectional channel.

- **Length Indicator (LI) Field**

  This 6-bit field indicator is used to indicate the length of the CPS packet payload, which may vary for each AAL2 channel. The LI value is binary encoded with a value that is one less than the number of bytes in the CPS packet payload. That is, the value of LI will be LI = n -1 bytes, where n is the number of bytes in the CPS packet payload. The maximum length of the CPS packet payload may be provisioned to be either 45 or 64 bytes, with the default length being 45 bytes [10]. If the maximum length chosen is 45 bytes, then LI values between 45 and 63 are not permitted.

- **User-to-User Indication (UUI) Field**

  The 5-bit UUI field is used to send information between peer SSCS sub-layers, above the CPS sub-layer. This information could either be destined for peer SSCS entities itself or for peer Layer Management entities. The UUI field is also used to distinguish between the different SSCS entities and the different

Layer Management users of the CPS. The ITU-T has defined 32 binary UUI codepoints for the 5-bit UUI field. Binary values 0 up to 27 are used to identify a particular SSCS protocol, whereas binary values 28 up to 30 are reserved for future enhancements. Finally, binary value 31 is used to send Layer Management information regarding the AAL2 layer. A more detailed examination of the UUI codepoint assignments can be found in [18].

- **Header Error Check (HEC) Field**

  This 5 bit field is intended for error detection of the CPS packet header, using a particular generator polynomial $G(x) = x^5 + x^2 + 1$. This detailed algorithm used for the HEC function is outside the scope of this thesis but may be found in ITU-T Recommendation I.366.1. When an error is detected, the particular CPS packet is discarded and the Layer Management is informed [19].

2.2.2 AAL2 Cell Assembly Process

The AAL2 cell payload, also referred to as the CPS-PDU, consists of a Start Field (STF) and multiple CPS packets. These CPS packets might all be sent from the same user, or from several distinct users. Figure 2-3 illustrates the process of assembling multiple CPS packets, originating from different sources, into a single AAL2 cell stream. It can also be seen in the diagram that a CPS packet may be split over two consecutive AAL2 cells in order to utilise the bandwidth more efficiently.

In the event that an AAL2 cell is lost, during cell transmission, and a CPS packet was split over itself and the following AAL2 cell, a technique is required to locate the start of the first complete CPS packet within the following AAL2 cell. For this reason each CPS-PDU contains a one byte header or Start Field (STF), as indicated in Figure 2-3.

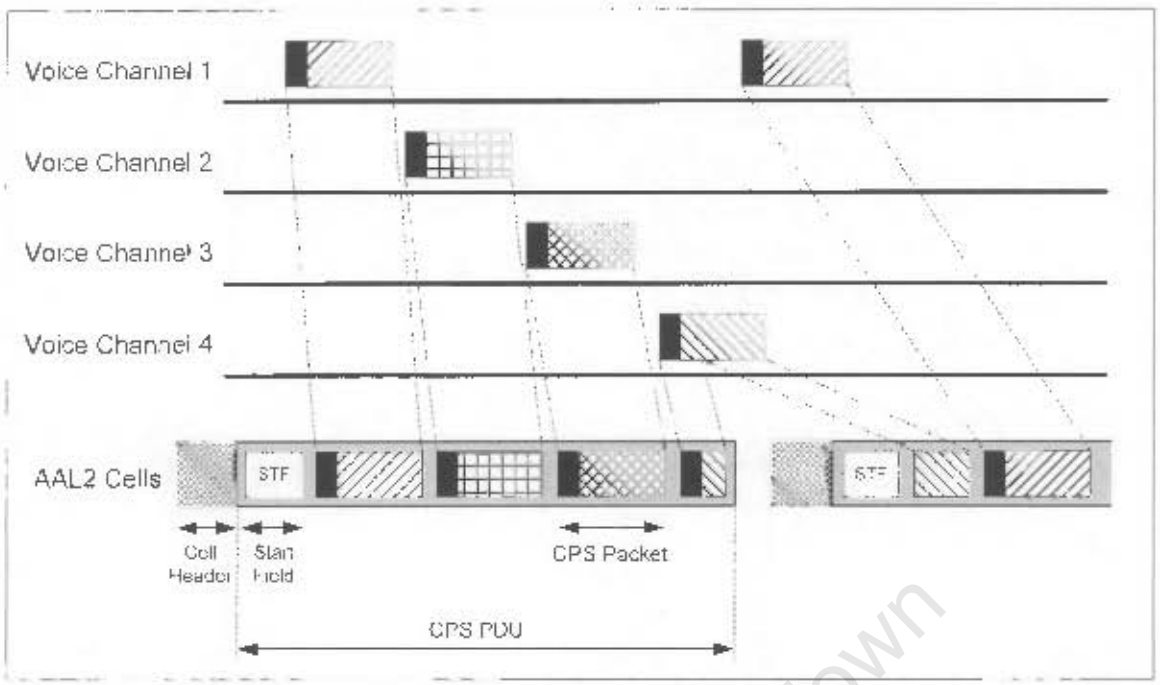

Figure 2-3 AAL2 Cell Assembly Process

2.2.3 Components of the AAL2 Start Field (STF)

The function of each field in the one byte CPS-PDU header or start field, as illustrated in Figure 2-4, is:

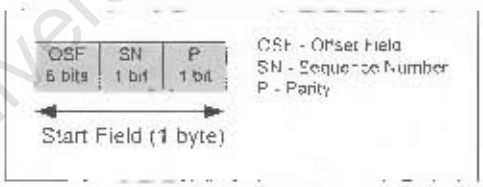

Figure 2-4 AAL2 Start Field

- **Offset Field (OSF)**

  The purpose of the 6-bit offset field is to indicate the remaining length, in bytes, of a CPS packet that possibly started in the preceding AAL2 cell and is continuing in the current AAL2 cell. The OSF thus points to the start of the new CPS packets and thereby provides a mechanism of boundary recovery in the event of loss of packet delineation resulting from the preceding AAL2 cell being lost.

- **Sequence Number (SN)**

  This 1-bit SN field provides a modulo 2 sequence number to be assigned to consecutive CPS packets in order to enable detection of lost or miss-inserted information. If a sequence number error is detected, the CPS packet may be discarded and the management system at the receiver is informed of the error condition via AAL2 Layer Management.

- **Parity (P) Field**

  The 1-bit parity field is used to detect errors in the STF. In the event that an odd parity violation is detected, the CPS packet is discarded and the error condition is reported via AAL2 Layer Management to the management system at the receiver.

Since the end-to-end delay for the transportation of voice needs to be strictly controlled, the AAL2 multiplexing unit, at the sender, will transmit AAL2 cells even if the AAL2 cells are not optimally packed with CPS packets. This is to prevent the first CPS packet in the AAL2 cell from being delayed for too long a period and thus violating its end-to-end delay. A timing module, referred to as the Timer Composite User (CU), is used to measure the period from when the first CPS packet arrives at the AAL2 cell until such a time that the AAL2 cell needs to be transmitted. Whenever the timer CU expires, the rest of the AAL2 cell is simply padded with zeros and transmitted. This padding functionality is achieved by assigning the value CID = 0, as mentioned in Section 2.2.1.

2.3 AAL2 Service Specific Convergence Sub-layer (SSCS)

The initial AAL2 recommendation, I.363.2, regarded the SSCS sub-layer as being null. The primary need for this recommendation was to provide a technique that would enable the multiplexing of small low-bit-rate voice packets onto a single ATM virtual connection. However, this recommendation did not specify a standard method by which true PSTN voice, with all its features such as in-band tones, dialled digits and signalling messages, could be transported over an AAL2 connection. The ITU-T addressed this shortcoming and standardised recommendation I.366.2 to enable an

AAL2 connection to provide the functionality associated with a PSTN connection. The ITU-T also introduced a complimentary specification, recommendation I.366.1, to define a method of transporting very large data packets (65 568 bytes) and out-of-band signalling information over an AAL2 connection [20].

The combination of ITU-T recommendations I.366.1 and I.366.2, in collaboration with the AAL2 CPS, provide a comprehensive solution for transporting a variety of media types over AAL2, such as uncompressed / compressed voice, circuit mode digital data, narrowband ISDN messages, demodulated facsimile, etc. To gain a better understanding on how these functionalities are accomplished, the avid reader is advised to examine [18].

2.4 Relationship between ATM Switching and AAL2 Switching

ATM is a connection-oriented technology, which means that a logical end-to-end connection first needs to be established, between the sender and receiver, prior to any user data transfer. This logical end-to-end connection is called an ATM Virtual Circuit (VC) and is identified by means of a Virtual Path Identifier and a Virtual Channel Identifier (VPI/VCI) value. In AAL2 terminology, each ATM VC is also referred to as an AAL2 path. Since AAL2 packets are multiplexed into an ATM VC, each ATM VC consists of a number of AAL2 VCs, called AAL2 channels. Each AAL2 channel is identified with a unique VPI/VCI and Channel Identifier (CID) value. AAL2 switching is thus performed, by an AAL2 switch, based on the VPI/VCI/CID value of each individual AAL2 CPS packet. In contrast, traditional ATM switching is performed, by an ATM switch, based on the VPI/VCI value of each ATM cell. Figure 2-5 illustrates the difference in stack architecture between an AAL2 switch and an ATM switch. As can be seen from the diagram, ATM switches are not aware of the content of any ATM cells and are therefore not even aware if AAL2 cells are switched through them. AAL2 cells are simply treated like normal ATM cells and are switched based upon the VPI/VCI value in the AAL2 cell header. However, an AAL2 switch would disassemble each received AAL2 cell into its individual CPS packets. CPS packets destined for a common AAL2 switching node or

AAL2 end-point are then reassembled into new AAL2 cells and transmitted to their desired destination.

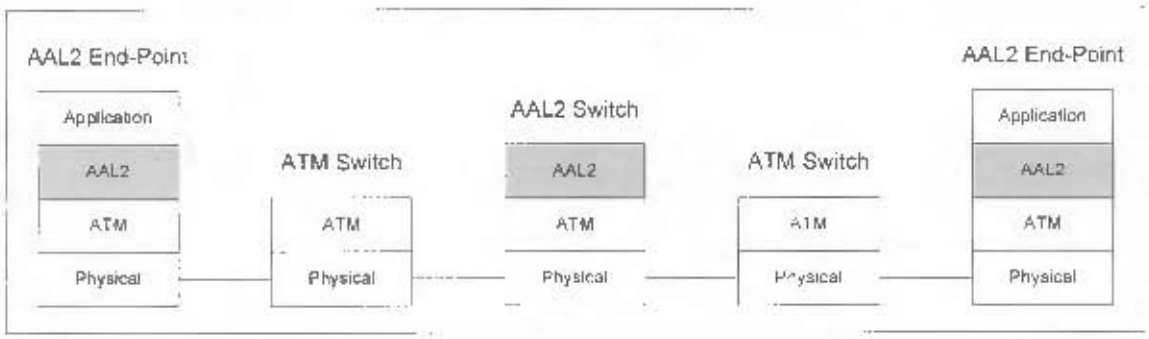

Figure 2-5 AAL2 Switch and ATM Switch Architecture

2.5 Practical Applications for AAL2

Although AAL2 has only been in existence for a few short years, it is widely being adopted as the technology of choice in Voice and Telephony over ATM (VTOA) applications, such as Voice over DSL (VoDSL) [16], ATM trunking [21], and Universal Mobile Telecommunication System (UMTS) wireless networks [22].

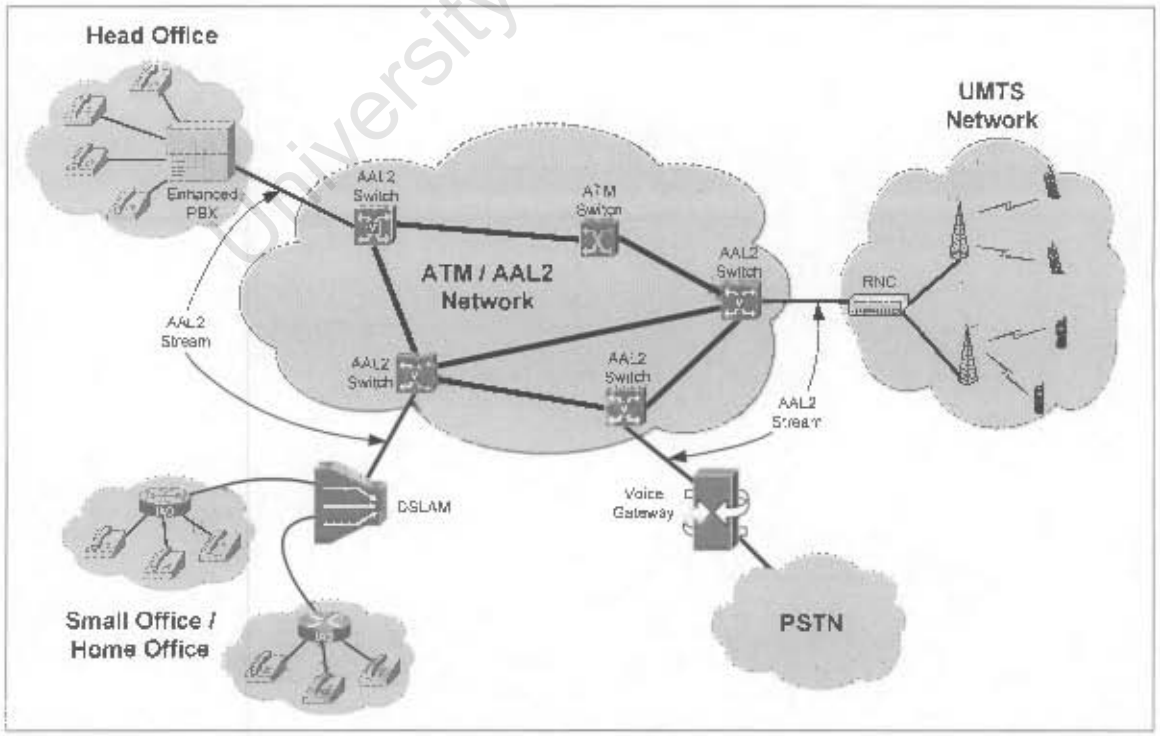

Figure 2-6 AAL2 Application Scenarios

Figure 2-6 illustrates all three of these application scenarios interconnected with each other via a hybrid ATM / AAL2 network.

2.5.1 Voice over Digital Subscriber Line (VoDSL)

A popular application for AAL2 is Voice over DSL (VoDSL), also referred to as ATM Loop Emulation Service (LES) using AAL2[8]. The VoDSL application is illustrated in the bottom left-hand corner of Figure 2-6 and is an extension of the better known Asynchronous DSL (ADSL). It consists of a number of normal telephones, Integrated Access Devices (IADs) and a common Digital Subscriber Line Access Multiplexer (DSLAM). At each office the telephones are connected to a single IAD. Each IAD is then connected, via local copper loops, to a centralised DSLAM located in the nearest local exchange. Although the transport mechanism to implement VoDSL can be either IP or ATM, the DSL Forum has standardised the use of ATM's AAL2 due to its inherent advantages, which includes the efficient use of limited bandwidth. The idea of VoDSL is to extend the local copper loop from the customer premises through an ATM access network. The DSLAM in turn, streams AAL2 voice traffic into the ATM / AAL2 network via an AAL2 switch. Independent research into the quality of voice calls transported over a VoDSL access network and voice calls carried over the traditional PSTN network concluded that no perceptible difference exists [23].

2.5.2 ATM Trunking

A second commercial application for AAL2 in the area of VTOA is ATM trunking. This is illustrated in the upper left-hand corner of Figure 2-6. This application is based on circuit emulation, as discussed in Section 1.1.5, but instead of utilising traditional AAL1, voice could be transported over the more efficient AAL2 as a cost effective alternative. This is an appropriate mechanism for connecting PBXs located in corporate offices to the ATM / AAL2 network. ATM trunking can also be referred to

[8] As defined by the ATM Forum in af-vmoa-0145.000

as ATM Trunking using AAL2 for Narrowband Services[9], which is merely an adaptation of ITU-T recommendation I.366.2 as described in Section 2.3.

2.5.3 Universal Mobile Telecommunication System (UMTS) Networks

A third commercial application, and the original driving force behind the standardisation of AAL2, is in the fixed wire-line portion of a UMTS network commonly referred to as the UMTS Terrestrial Radio Access Network (UTRAN). Figure 2-6 illustrates a simple UMTS network consisting of only one Radio Network Controller (RNC) and two base stations, also referred to as Node Bs. A realistic UMTS network would typically consist of a number of these network elements, depending on the size of the overall UMTS network. AAL2 is utilised as the transport technology between the base stations and the RNCs and between peer RNCs to utilise the transmission resources as efficiently as possible.

2.5.4 Necessity for AAL2 Switches and AAL2 Signalling

Figure 2-6 illustrates a hybrid ATM / AAL2 network, consisting of both ATM and AAL2 switches, interconnecting each of the above mentioned AAL2 application scenarios. The location of these AAL2 switches is commonly considered to be at the edge of the hybrid network, as well as a few strategic locations within the network. In Figure 2-6 each application scenario is streaming AAL2 voice traffic into the hybrid network and is thus connected to an AAL2 switch. The AAL2 switch would then switch these individual AAL2 user packets to their desired destinations through the network and utilise the network bandwidth efficiently.

However, these AAL2 applications require AAL2 signalling to first establish on-demand AAL2 connections between two AAL2 end-points, through the hybrid ATM / AAL2 network, prior to the transportation of any user traffic. Chapter 3 will discuss the AAL2 signalling protocol and investigate the need for and benefits of this new signalling protocol.

[9] As defined by the ATM Forum in af-vtoa-0113.000

Chapter 3

Literature Review of the AAL2 Signalling Protocol

3.1 Purpose of a Signalling System

A signalling system's purpose is to transfer control information between the network elements of a telecommunications network. These elements would typically be switches or operation centres and the control information would be the signalling messages required to establish or tear-down connections. In turn, these connections could be either voice connections or data connections [24]. This Chapter will focus on the establishment of voice connections via the AAL2 signalling protocol. It will also highlight and discuss the importance of signalling in an AAL2 network.

3.2 Necessity for AAL2 Signalling

The primary objective of AAL2 signalling is to establish, maintain and tear down AAL2 connections between peer AAL2 nodes. An AAL2 node can be either an AAL2 end-point or an AAL2 switch. AAL2 is the only AAL layer that has its own signalling protocol. The primary reasons for this are as follows:

- Firstly, ATM has many signalling protocols, such as the ATM Forum's UNI and PNNI, the ITU-T's Q.2931, Q.2971, B-ISUP, and so on. Extending all these protocols to accommodate AAL2 would be very difficult.

- Secondly, having two separate signalling protocols has its advantages. The decoupling of the two signalling protocols would allow multiple AAL2 overlay networks to operate over a single ATM network. Each network would operate with its own routing and addressing schemes [25]. This separation becomes crucial when separate operators own and manage the two different networks.

- Finally, AAL2 has the ability to multiplex many AAL2 channels into a single ATM virtual connection. In order to establish and release these AAL2 channels, a dynamic protocol is required. This is accomplished by means of AAL2 signalling.

3.3 AAL2 Signalling Protocol Architecture

As illustrated in Figure 3-1, the AAL2 signalling protocol is limited to the control of the AAL2 layer. Although the AAL2 signalling protocol bears some similarities to the ATM signalling protocol, it is important to note that they operate completely independent of each other. This approach speeds up the AAL2 connection establishment process since intermediate ATM switches do not cause delay by storing and forwarding AAL2 signalling messages.

The AAL2 signalling protocol assumes the existence of an established ATM virtual channel between two peer AAL2 nodes before AAL2 channels can be established inside the ATM VC. This ATM VC could be a Permanent Virtual Circuit (PVC), created by management system provisioning, or a Switched Virtual Circuit (SVC), created by an on-demand ATM signalling procedure. However, how these ATM virtual channels are established is not within the scope of the AAL2 signalling protocol specification [13]. Consequently, ATM VCs perform the same function for AAL2 CPS packets as ATM Virtual Paths (VPs) perform for ATM VCs, namely providing trunks between peer AAL2 nodes for the multiplexing of CPS packets.

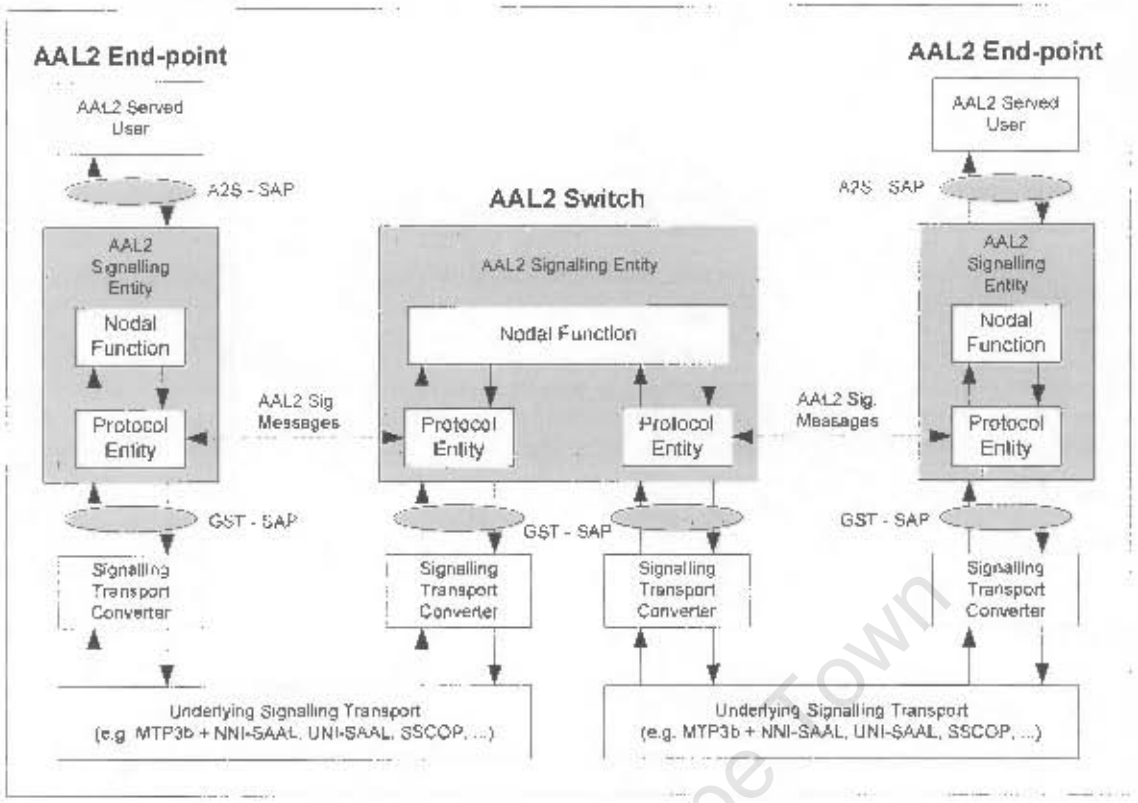

Figure 3-1 AAL2 Signalling Protocol Reference Model

The diagram above, adapted from ITU-T recommendation Q.2630.1, illustrates the AAL2 signalling protocol architecture. It contains three AAL2 nodes consisting of two AAL2 end-points and one AAL2 switch. An AAL2 end-point provides services such as connection establishment and release and the AAL2 switch provides the switching and routing support. Above the AAL2 signalling entity, in each AAL2 end-point, an AAL2 served user is located. This would typically be the user application requiring AAL2 signalling support, or as is the case in a UMTS network, the radio resource management and handover control entity [25].

The Signalling Transport Converter (STC), also referred to as the Generic Signalling Transporter (GST), situated below the AAL2 signalling entity provides independence between the AAL2 signalling protocol and the underlying signalling transport layer. This is accomplished by providing a generic set of primitives that the AAL2 signalling protocol utilises when exchanging AAL2 signalling messages with a peer signalling entity. Therefore, the STC hides all the differences between the signalling protocol and the signalling transport mechanism. The type of STC used will depend

on the underlying signalling transport mechanism being utilised. Currently, two STC standards exist. The ITU-T's Q.2150.1 is used if the transport mechanism is broadband MTP3b, whereas Q.2150.2 is used if the transport mechanism is User Network Interface (UNI) – Signalling AAL (SAAL). Services provided by the STC include the assured (error free) transfer of data and in sequence delivery of data [26].

The AAL2 Signalling Entity is comprised of two distinct components, namely the Nodal Function and the Protocol Entity, as illustrated in Figure 3-1 on the previous page.

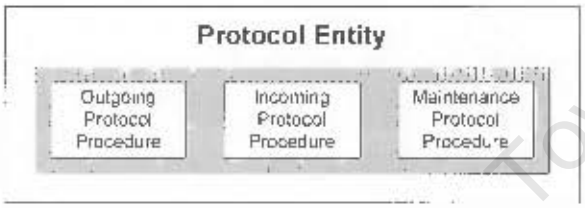

Figure 3-2 Internal Structure of the Protocol Entity

The Protocol Entity defines the interaction between two peer adjacent AAL2 nodes and is illustrated above in Figure 3-2. The outgoing protocol procedure contains the mechanisms to initiate an AAL2 connection, while the incoming protocol procedure is applied when a request for a new AAL2 connection is received. Either of these procedures can release an AAL2 connection. The maintenance protocol procedure provides the mechanism to align the AAL2 resources status within the two adjacent AAL2 nodes, as well as the procedures to block or unblock an AAL2 path. The nodal function provides a bridge between the incoming and outgoing protocol entities. It is also responsible for the routing functionality and keeps a record of the AAL2 path resources.

The mapping of these components into the AAL2 signalling framework design implemented in this study will be discussed in Chapter 4. Appendix A provides an overview of the interface between the AAL2 signalling entity and its neighbouring layers. The Layer Management entity, not indicated in Figure 3-1, is also illustrated in Appendix A. This entity was excluded from the AAL2 signalling protocol reference model illustration in Figure 3-1 to simplify the diagram. This was done because the features supported by the layer management entity fall outside the scope of this study.

For more detail regarding the AAL2 signalling stack architecture, the reader is advised to examine ITU-T recommendation Q.2630.1.

3.4 Relationship between AAL2 Signalling and ATM Signalling

As stated in Chapter 1, AAL2 is a connection oriented protocol. Therefore, prior to the transportation of any user data across an AAL2 network, an end-to-end AAL2 channel first needs to be established between the sending and receiving AAL2 nodes. However, AAL2 signalling is unable to establish this end-to-end connection without the existence of either a previously established ATM VC or multiple ATM VCs between the relevant peer AAL2 nodes, depending on the network topology. These ATM VCs are established via ATM signalling. This dependence of AAL2 channels on ATM channels is a potential problem because of the independence between these two signalling protocols, as stated earlier in Section 3.2. Consequently, it is impossible for a user to request a new AAL2 connection without an pre-established underlying ATM VC. The AAL2 signalling protocol is merely informed by the STC below it whether an ATM VC exists between the current AAL2 node and the neighbouring AAL2 node en-route to the destination node. But the AAL2 signalling protocol itself cannot request the establishment on a new ATM VC if one does not already exist.

It is important to understand the concept of ATM channel establishment in the context of an AAL2 network. The ATM channel in question is not a transparent end-to-end connection spanning the entire hybrid ATM / AAL2 network in order to connect the two remote AAL2 end-points. This would imply that AAL2 channels are merely multiplexed onto this end-to-end ATM channel. This technique would not only be inefficient, but it would also cause AAL2 switching to be redundant, since it simply offers a trunking solution. Rather, ATM channels are established on a point-to-point manner between peer AAL2 nodes as illustrated in Figure 3-3.

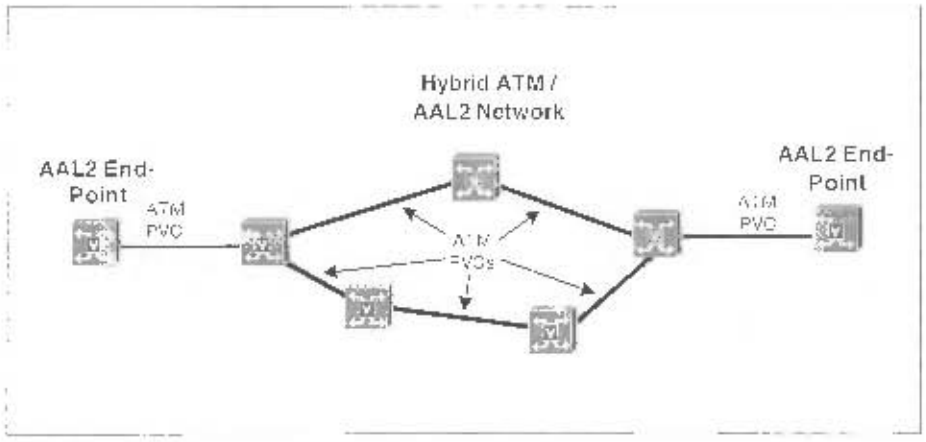

Figure 3-3 ATM PVCs within an AAL2 Network

In the diagram above all the switches located inside the network are AAL2 switches and all connections between them are point-to-point ATM connections. These connections could be either PVCs or SVCs as discussed in Section 3.3. However, PVCs are illustrated above. Only once these ATM connections are provisioned can point-to-point AAL2 channels be established onto these ATM VCs. Therefore, multiple AAL2 connections are required to establish a functional end-to-end AAL2 connection between the two AAL2 end-points as illustrated in Figure 3-1.

The sections that follow investigate and discuss the procedures required to establish an end-to-end AAL2 connection. Firstly, it considers and discusses the procedures involved in establishing an ATM VC, via ATM signalling. It will then examine the AAL2 signalling procedures required to establish an AAL2 channel within the pre-established ATM VC. This is done to highlight certain similarities and distinct differences between the two signalling protocols.

3.4.1 Procedures Required for ATM Channel Establishment

An ATM virtual connection is established by means of ATM signalling. The ITU-T and ATM Forum have specified and standardised many ATM signalling protocols. The ITU-T's recommendation Q.2931 and Q.2971 define the procedures required to establish point-to-point and point-to-multipoint ATM connections respectively. A point-to-multipoint connection is commonly referred to as multicasting. However, the

most popular ATM signalling specification is ATM Forum's User Node Interface (UNI), which is employed in the access part of the network, and Private Network Node Interface (PNNI), which is employed in the core of the network. Various versions of each specification exist, but the most recent are UNI 4.0 and PNNI 1.0. ATM signalling messages are transported on a dedicated ATM VC, namely VPI/VCI 0/5, by means of AAL5 between peer ATM users. As a result, ATM signalling is referred to as Common Channel Signalling (CCS), also known as Out-of-Band signalling.

The ATM virtual connection channel establishment process is initiated by means of a SETUP ATM signalling message sent from the initiating ATM user to the desired destination ATM user. Each ATM signalling message contains information in the form of parameters referred to as Information Elements (IEs). The information contained in the parameters of the SETUP signalling message corresponds to the desired channel characteristics of the requested ATM VC [27]. Some of the basic parameters contained within an ATM signalling SETUP message are listed below in Table 3-1.

ATM SETUP Message Parameters	**Description of Parameter**
ATM *Destination Address*	This uniquely identifies the desired ATM destination
Broadband Bearer Capacity	Contains the traffic type, e.g. CBR, VBR, UBR, etc to be supported
Traffic Descriptor	Specifies information relating to the Peak Cell Rate (PCR), Minimum Cell Rate (MCR), etc
QoS parameters	Specifies the end-to-end transit delay required for the chosen *Broadband Bearer Capacity*
AAL parameters	Specifies the AAL type to be supported by the ATM virtual connection

Table 3-1 ATM SETUP Message Parameters

For the purpose of this study only the *AAL* parameter will be discussed in detail. For detailed information regarding any other SETUP parameter the reader is advised to examine ITU-T recommendation Q.2931.

- **The AAL Parameter**

  Each parameter within the SETUP signalling message contains specific information regarding the requested ATM VC. In order to establish AAL2 channels within an ATM VC, the ATM VC's AAL parameter should be configured as follows: a) the AAL type selected must be AAL2; b) the maximum CPS packet size must be either 45 or 64 bytes, as specified in Section 2.2.1; c) the maximum number of multiplexed AAL2 channels within the ATM VC must be specified and d) the SSCS type selected must be either ITU-T recommendation I.366.1 Segmentation and Reassembly (SAR), ITU-T recommendation I.366.2 trunking, as described in Section 2.3 or the NULL option. Each SSCS type, except the NULL option, has additional parameters that need to be specified. If the SSCS type chosen is I.366.1 SAR then the primary parameter is the maximum packet size. However, if the SSCS type chosen is I.366.2 trunking then the primary parameter is the audio profile chosen. The function and significance of this audio profile will be discussed in Section 3.4.3.

The following section will discuss the AAL2 signalling procedures required to establish an AAL2 channel within a pre-established ATM VC, or AAL2 path. One of the limitations of this two-step approach to AAL2 channel establishment is that all the AAL2 channels established within a specific ATM VC inherit some of the characteristics of that VC, such as the specified QoS. Even though the QoS requirements of different services are distinctly different at the user application level, the IUT-T has standardised one QoS class at the AAL2 path level. The AAL2 network operator therefore needs to know beforehand the typical QoS and bandwidth requirements needed by an AAL2 user. This knowledge would enable the AAL2 operator to request the required QoS and traffic parameters from the ATM operator who will need to provide the necessary ATM VCs.

This limitation has been investigated elsewhere [28] and a method has been proposed by which different QoS conditions could be accommodated within a single ATM VC. Since different applications requiring different QoS levels would need to be supported by UMTS wireless networks, this method could be used to control the QoS for each application at the AAL2 level, instead of the ATM level. It was argued that this

method would lead to more efficient utilisation of network resources. A simulation study was performed to validate this argument. However, Kawakami et al merely concluded that this method could improve bandwidth utilisation.

The question of which ATM service class should be assigned to an AAL2 path, namely CBR, Real Time-VBR (rt-VBR), or Unspecified Bit Rate (UBR), has been investigated in [29]. A mathematical analysis addressed this question comparing the effectiveness of each service class from a bandwidth management point of view. The simplest service class is CBR, which only specifies a Peak Cell Rate (PCR) for each ATM VC. When a new AAL2 connection is established, Call Admission Control (CAC) is performed at the AAL2 node to determine whether there is enough bandwidth available for the incoming call. If the ATM VC is unable to support the requested PCR, the AAL2 connection request is denied and the call is simply terminated. The second possible service class is rt-VBR. The bandwidth of an rt-VBR connection is specified by a PCR, a Sustainable Cell Rate (SCR) and either a Burst Tolerance (BT) or a Maximum Burst Size (MBS). Since the rt-VBR service class can achieve a higher statistical multiplexing gain than the CBR service class it was found that rt-VBR performs better than CBR. This advantage increases as the number of AAL2 connections increases. However, it is far more complicated to implement. The third possible service class is UBR, which results in bandwidth management being performed at the ATM virtual path level instead of the ATM virtual channel level. AAL2 connections are thus accepted or rejected based on the available bandwidth of the virtual paths. This method is the most bandwidth efficient technique.

The use of AAL2 switches for all three service classes was also investigated in [29]. It was found that AAL2 switches were very effective for CBR and a small number of rt-VBR ATM VCs, but had less impact as the number of VBR VCs increased. The impact of AAL2 switches in UBR was also very small for any number of VCs, since a large statistical multiplexing gain had already been achieved by utilising UBR without the use of AAL2 switches.

This study, comparing the effectiveness of AAL2 switches and UBR ATM VCs, was extended in [30] by means of mathematical analysis. It was found that AAL2 switches are effective with CBR or VBR VCs if only small traffic streams were considered.

This is because the partially-filled AAL2 cells resulting from low traffic, which deteriorate performance, could be removed with AAL2 switching nodes, by re-packaging the partially-filled cells with AAL2 CPS packets destined for a common destination. However, as the traffic increases, so the UBR alternative becomes more effective as the statistical multiplexing gain becomes larger than under CBR or VBR. Overall, the utilisation of UBR showed a similar performance in bandwidth efficiency to that achieved with AAL2 switches utilising either CBR or VBR.

Although UBR is the most bandwidth efficient service class, as discussed above, for the purpose of transporting delay sensitive compressed voice over an ATM connection, as is the case with voice over AAL2, UBR it is not the most effective. UBR is a best-effort service class used primarily for data transfers. UBR connections are unable to support any form of bandwidth guarantees, nor is it able to support any form of Cell Transfer Delay (CTD), as required by voice traffic. Therefore, based on the above discussion it can be concluded that ATM connections supporting AAL2 voice channels should be established as rt-VBR connections. This is because although rt-VBR is not the most bandwidth efficient service class, it is however able to support both bandwidth guarantees as well as CTDs as required by AAL2 connections.

Figure 3-4 below illustrates the ATM connection establishment process as a three-way handshaking procedure. Handshaking simply refers to the basic exchange of signalling messages along with their parameters.

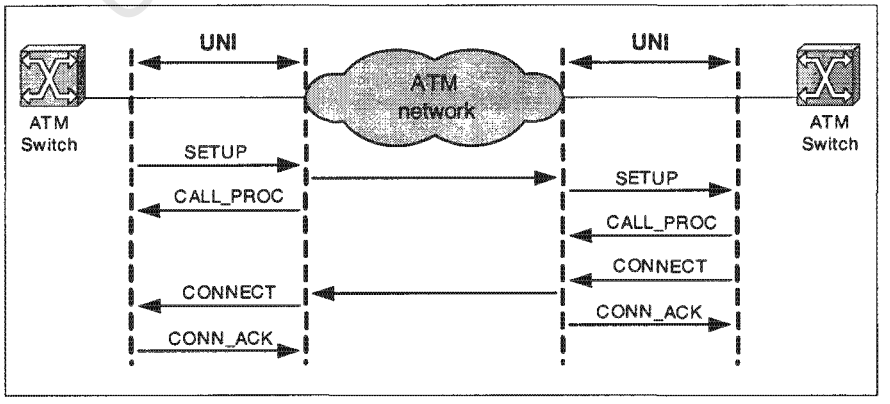

Figure 3-4 Connection Establishment Procedure in ATM Signalling

The process is referred to as three-way handshaking, because of the three primary ATM signalling messages, namely SETUP, CONNECT and CONN_ACK, which need to be transported across the UNI network in order to request a new ATM connection.

3.4.2 Procedures Required for AAL2 Channel Establishment

Unlike ATM signalling, AAL2 signalling protocol is the same for the user-network or the network-network interface. AAL2 signalling thus establishes AAL2 connections on a point-to-point basis. As an example, AAL2 connections would typically be established between a base station and the Mobile Switching Centre (MSC) for trunking in a UMTS network, or between two PBXs or international AAL2 switches for landline trunking. An AAL2 connection is bidirectional and the same CID value is used in both directions of the connection. However, each AAL2 connection is also asymmetrical. Therefore, different traffic and QoS parameters can be supported in either direction of the connection. The AAL2 signalling protocol is a symmetrical protocol, consequently either interconnected peer AAL2 node can initiate a new AAL2 connection or tear down an existing connection.

Once ATM signalling has created a logical end-to-end ATM VC, comprising multiple point-to-point ATM VCs connecting the relevant AAL2 switches, the AAL2 signalling protocol is able to establish additional AAL2 connections over this ATM connection, as requested by AAL2 users. This is done using the AAL2 signalling Establish Request (ERQ) message [13]. Some of the parameters contained within an AAL2 signalling ERQ message are listed in Table 3-2.

AAL2 ERQ Message Parameters	Description of Parameter
AAL2 *Destination Address*	This uniquely identifies the desired AAL2 destination node
Originating Signalling Association Identifier (OSAID)	Required to uniquely identify the originating AAL2 node
Connection Element Identifier (CEID)	Specifies the AAL2 path identifier and the Channel Identifier (CID)
AAL2 Link Characteristics (ALC)	Specifies the requested bandwidth for an AAL2 connection
Service Specific Convergence Sub-layer (SSCS) Information	Indicates the service specific information to define the type of SSCS residing over the AAL2 layer

Table 3-2 AAL2 ERQ Message Parameters

The initiating AAL2 node proposes the establishment of a new AAL2 connection with two primary parameters. Firstly, the employment of a specific CID value that is not yet in use and secondly, a particular bandwidth is requested for either direction of the asynchronous connection. The destination AAL2 node could either accept or reject the connection request, as it is the responsibility of the two nodes to guarantee the resources required within the ATM connection to support this new AAL2 connection. As stated previously in Section 3.4, the independence between the ATM signalling protocol and the AAL2 signalling protocol results in a few drawbacks. One of these is that an AAL2 connection request can either be accepted or rejected. If the available bandwidth of a particular ATM VC becomes limited and a new AAL2 connection request requires a link bandwidth greater than the residual ATM connection bandwidth, the AAL2 connection request is simply rejected. Further, the AAL2 signalling protocol is currently unable to request a change in ATM connection bandwidth resulting from limited available capacity.

3.4.3 AAL2 Encoding Format Profile

During the ATM connection establishment process, described in Section 3.4.1, the initiating ATM user is required to specify the AAL parameter in the ATM SETUP signalling message. As part of the AAL parameter, the SSCS type requested also needs to be specified. As stated in Section 3.4.1, if the SSCS type chosen is ITU-T

recommendation I.366.2 trunking, then a specific audio profile needs to be chosen. A profile consists of a set of entries, where each entry comprises:

- An *Encoding Format*, which specifies a particular voice encoding algorithm
- A *Length*, which specifies the length of the packet
- A *Packet Time* value, which indicate the inter-packet departure / arrival period
- An *UUI Range*, which is used for packet sequencing

A simplified example of a profile is indicated in Table 3-3. Each row of the profile represents a separate entry. The acronym SS refers to 50% Silence Suppression being employed.

UUI Codepoint Range	Packet Length (Bytes)	Description of Encoding Format	Packet Time (ms)
0-15	40	PCM, G.711, 64 Kbps, generic	5
0-15	20	CS-ACELP, G.729, 8Kbps	20
0-15	10	CS-ACELP, G.729, 8Kbps	10
0-15	2	CS-ACELP, G.729, 8Kbps, SS	10

Table 3-3 SSCS Profile Using G.729

The ITU-T has pre-defined 10 profiles for use by AAL2 voice traffic, which are depicted and explained in Annex P of ITU-T recommendation I.366.2. Each profile is specified by means of a profile identifier, which is indicated during the ATM connection establishment phase. The default profile entry is PCM G.711 at 64 Kbps. This acts as a precautionary feature in situations where the initiating and destination AAL2 users' equipment do not support the same voice encoding / decoding algorithms. With G.711 as the default voice encoding algorithm in all profiles, it is guaranteed that at least this algorithm will be supported. Once a profile has been adopted between the transmitting and receiving AAL2 users, the transmitter can select any entry in the adopted profile and it will be accepted by the receiver.

3.5 The AAL2 Signalling Life Cycle

The AAL2 signalling protocol was designed to be a simpler protocol than the traditional ATM signalling protocol. One of the areas in which this simpler design approach is apparent, is in the two-way connection establishment phase of AAL2 signalling, depicted in Figure 3-5. This is in contrast to the three-way connection establishment procedure of ATM signalling, illustrated in Figure 3-4.

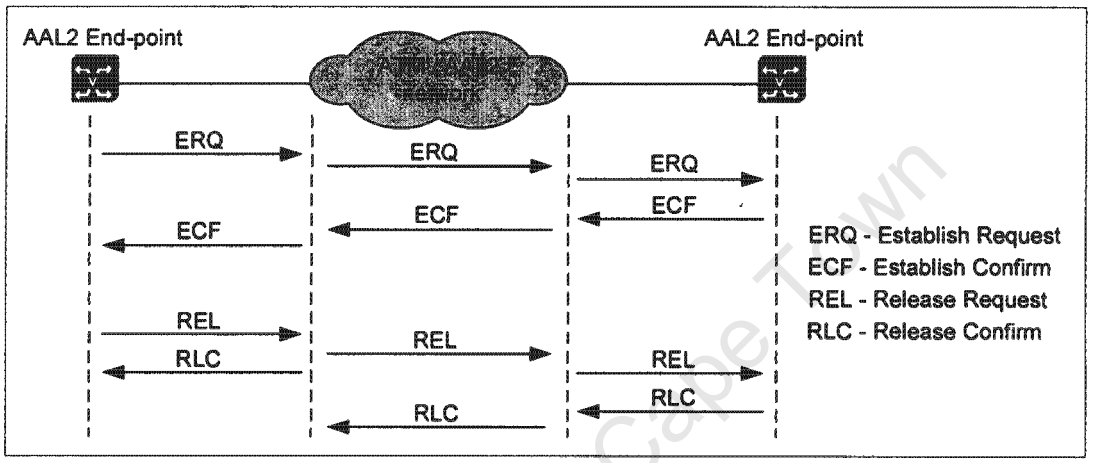

Figure 3-5 Channel Establishment and Release Procedures in AAL2 Signalling

This figure illustrates both the procedures for connection establishment and release in AAL2 signalling. The specific format and encoding technique defined for each AAL2 signalling message can be found in [13]. The AAL2 connection or AAL2 call life-cycle consists of three phases, namely:

- The *Establishment* phase, initiated by an AAL2 signalling Establish Request (ERQ) message
- The *Active* phase, where the connection remains connected and AAL2 user data can be transferred across the AAL2 channel. The duration of the connection is referred to as the *hold time*
- The *Release* phase, initiated by an AAL2 signalling Release Request (REL) message. The process of releasing an AAL2 channel is also called *tear down*

Each of these phases will now be discussed in greater detail.

- **Establishment Phase**

  The *Establishment* phase consists of two AAL2 signalling messages. The initiating AAL2 end-point starts the establishment phase by sending an ERQ message, via the ATM / AAL2 network, to the receiving AAL2 end-point. On reception, the receiving AAL2 end-point responds by sending an Establish Confirm (ECF) message back to the initiating AAL2 end-point. Only after both these AAL2 signalling messages have travelled through the ATM / AAL2 network, is the establishment phase complete. This entire phase is known as connection setup.

- **Active Phase**

  Once the establishment phase has completed successfully, an AAL2 connection enters the *Active* phase. More specifically, the receiving AAL2 end-point enters the active phase once it sends the ECF message and the initiating AAL2 end-point enters the active phase on reception of the ECF message.

- **Release Phase**

  The active phase of each AAL2 end-point ends whenever a REL message is initiated and sent from either AAL2 end-point or is accepted by a receiving AAL2 end-point. This process initiates the *Release* phase. The release phase of each AAL2 end-point ends whenever a Release Confirm (RLC) message is returned from either the ATM / AAL2 network to the initiating AAL2 end-point or from the receiving AAL2 end-point to the ATM / AAL2 network. This process is also known as connection release or connection tear-down.

- **Pending and Rejected Connections**

  After the initiating AAL2 end-point sends an ERQ AAL2 signalling message and before the establishment phase is completed successfully, the connection is referred to as *pending*. If the establishment phase does however not complete successfully, the connection is referred to as *rejected*.

Besides the four AAL2 signalling messages explained above, AAL2 signalling utilises a total of 11 signalling messages. Four of them are utilised for connection

setup and tear down while the remaining 7 messages are utilised mainly for management related information transfers. A complete set featuring the available AAL2 signalling messages is illustrated in Appendix A.

3.6 Choice of AAL to Support AAL2 Signalling

AAL2 signalling messages can be transported between peer AAL2 nodes over either AAL5, as is the case with ATM signalling messages, or AAL2. Utilising AAL5 would result in either some sort of Channel Associated Signalling (CAS) or CCS, also known as In-Band or Out-of-Band signalling respectively, being employed. In CAS both ATM and AAL2 signalling messages would be transported in the same dedicated ATM VC, namely VPI/VCI 0/5. In contrast, separate dedicated ATM VCs for both ATM and AAL2 signalling messages could be employed in CCS, such as VPI/VCI 0/5 for ATM and VPI/VCI 0/6 for AAL2.

Even the utilisation of AAL2 as a transport mechanism for AAL2 signalling messages could result in either a CAS or CCS technique being used. In CAS, both the AAL2 user data and the AAL2 signalling messages would be transported within each AAL2 connection for that particular connection. However, in CCS a dedicated AAL2 channel would be reserved for all AAL2 signalling related information pertaining to a specific AAL2 path, or ATM VC. The question, which AAL type to utilise for this transportation purpose will be addressed and discussed in more detail in Chapter 4.

3.7 Unresolved Challenges of the AAL2 Signalling Protocol

The most important issue still unresolved in the AAL2 signalling protocol is the acquisition and distribution of AAL2 routing information. Unlike the ATM signalling protocol PNNI that incorporates a routing protocol, the AAL2 signalling protocol currently does not incorporate a separate routing protocol. The AAL2 signalling protocol supports hop-by-hop routing based on the addressing information and requested link characteristics specified. The AAL2 signalling standards, however, do not define a technique whereby AAL2 switches could exchange AAL2 routing information. As a result, most AAL2 links are statically provisioned and this limits

AAL2 signalling to small domains. In order for a completely independent overlay AAL2 network to operate over an existing ATM network, an AAL2 routing protocol is essential. The routing protocol's primary function would be to update the addressing / routing tables in the AAL2 nodes in accordance with the network topology of the AAL2 network.

As an example, consider Figure 3-6 as an overlay AAL2 network operating over an existing ATM network. All the switches illustrated in the figure are AAL2 switches and interconnected with ATM VCs as discussed in Section 3.4. An AAL2 routing protocol would enable the AAL2 end-point on the left-hand side of the figure to send voice traffic to the AAL2 end-point situated on the right-hand side of the figure based on the optimal route through the AAL2 network. Once determined, the routing tables of each AAL2 switch along the traffic route would be configured accordingly. This would result in all AAL2 user data traversing the AAL2 network along this predefined route.

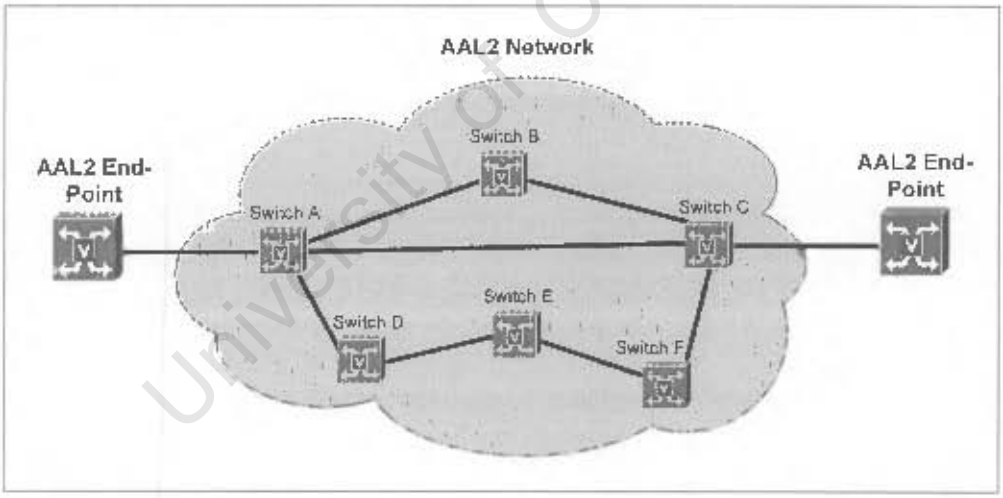

Figure 3-6 Example of an AAL2 Routing Protocol

In Figure 3-6 the optimal route might appear to be the direct link between AAL2 switches A and C, but the direct link might be unable to support the requested AAL2 link characteristics. As a result, the AAL2 routing protocol would then calculate a suitable route through the AAL2 network by means of evaluating each possible route through the network. Each route would be considered based on the available bandwidth of each link and the end-to-end time delay constraint of the requested

AAL2 connection. Factors relating to load balancing of each link in the network, as an attempt to avoid congestion in any specific link, would also be considered. As soon as a suitable route is discovered, the routing table's entries of each affected AAL2 switch along the chosen route would be configured accordingly.

Although the example above consists of a small AAL2 network populated with a small number AAL2 switches, the need for an AAL2 routing protocol becomes more apparent in larger networks, especially when using AAL2 and AAL2 signalling in commercial applications such as VoDSL access networks, ATM trunking networks and UMTS wireless networks, as described in Section 2.5.

Chapter 4

Design of the AAL2 Signalling Framework

In designing the AAL2 signalling framework, issues relating to AAL2 signalling delay need to be identified and addressed. This Chapter discusses certain techniques which can be employed to minimise the signalling delay after which the AAL2 signalling framework design will be presented and discussed. The most important design considerations were:

- **Modularity**
  This technique allowed individual modules of the signalling framework to be developed and tested in isolation
- **Scalability**
  The signalling framework needs to be scalable regarding the number of simultaneous connections that can be supported
- **Efficiency**
  The AAL2 signalling framework needs to be efficient in terms of the delays incurred during the AAL2 connection establishment process

4.1 Requirement for Minimising AAL2 Signalling Delay

AAL2 has been chosen as the transport technology for the access network in third generation UMTS wireless networks, as discussed in Section 2.5. The access network is required to support the Soft Handover (SHO) functionality, which requires a fast connection setup and tear-down. A SHO is initiated by a mobile terminal whenever a

user moves from a base station's area of coverage to the area of coverage of a neighbouring base station. An objective of the signalling protocol is to minimise the service disruption time during the SHO. The service disruption time is defined as the period during which the mobile user is not connected to the network and thus unable to receive any user data. Therefore, data destined for the mobile user needs to be buffered and stored in the network, until such time as the mobile user can again start receiving data. Where this buffering of user data should occur, either in the base station or in the Radio Network Controller (RNC), is an area of study that falls outside the scope of this dissertation, but needs to be addressed in future. The disadvantage of buffering the user data in the UMTS network is that an additional delay is introduced into the network. The data rate at which the mobile user receives data also influences the buffer occupancy, since the higher the data rate, the larger the buffer required for a fixed disruption time. Thus, the maximum buffer size required to prevent any data loss due to buffer overflow is influenced by the following two factors:

- The maximum data rate of a single connection destined to the mobile user
- The largest signalling delay that could occur during the SHO process

The AAL2 signalling protocol, located in the base stations, the RNCs and all other AAL2 switches in the fixed portion of a UMTS network, would thus need to be efficient in terms of delays incurred during the AAL2 connection establishment process. This can be accomplished by minimising the signalling delay associated with establishing a new AAL2 channel. This would result in both minimising the service disruption time during SHO and guaranteeing the required QoS of the AAL2 voice connection.

4.2 Choice of AAL to Support AAL2 Signalling

Either AAL2 or AAL5 can be utilised to transport AAL2 signalling messages between AAL2 nodes, as introduced in Section 3.6. Further, both AAL options can be utilised for Channel Associated Signalling (CAS) or Common Channel Signalling (CCS). However, neither method of CAS would be a viable option. Utilising AAL5 in CAS would result in both AAL2 signalling and AAL5 signalling messages being

transported in a common ATM VC. This technique would require each ATM and AAL2 switch to interrogate each traversing ATM signalling cell to determine whether it contains AAL2 or AAL5 signalling information. This method would be time consuming. However, utilising AAL2 in CAS would result in both AAL2 user data and AAL2 signalling messages being transported in a common AAL2 channel. This technique, in turn, would require each AAL2 switch to interrogate each AAL2 cell to determine whether user data or signalling information was contained within each cell. Consequently, neither AAL option results in an efficient solution when channel associated signalling is utilised.

A simpler technique, resulting in faster processing at the switching nodes, is to separate the signalling information from the user data. This is achieved by common channel signalling where, for AAL5, a separate ATM VC is utilised or, for AAL2, a separate AAL2 channel is utilised.

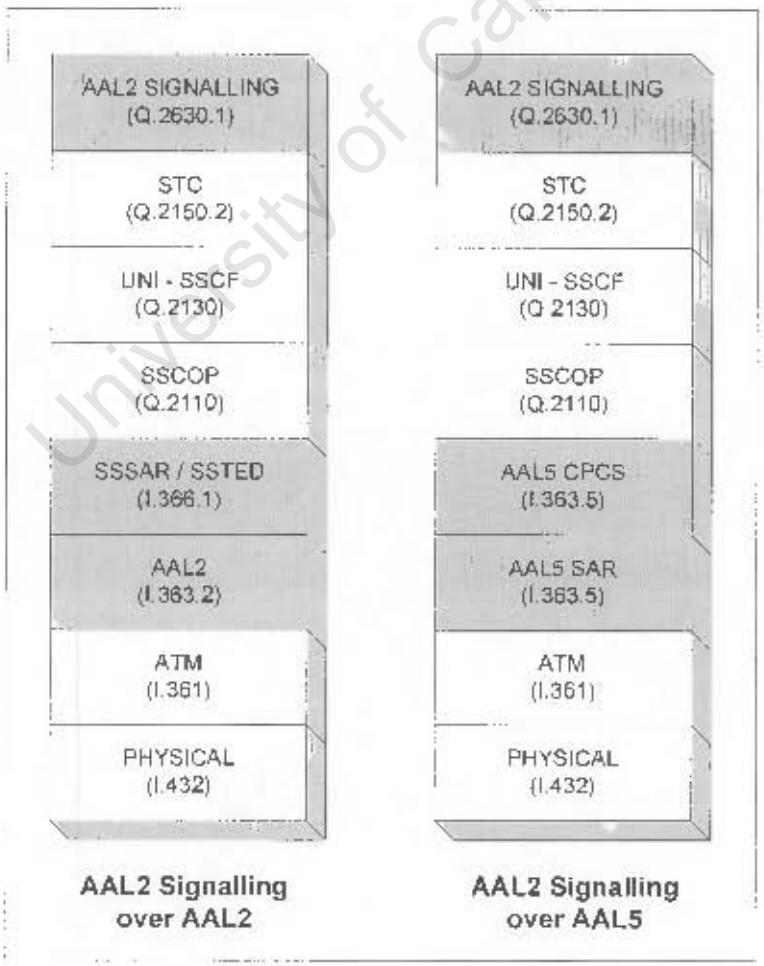

Figure 4-1 AAL2 Signalling Transported over AAL2 or AAL5

Figure 4-1 depicts the complete AAL2 signalling stack to illustrate the difference between transporting AAL2 signalling over AAL2 or AAL5 in a User-to-Network Interface (UNI) network. Each layer in the figure also contains the equivalent ITU-T recommendation number. The reader is advised to examine Appendix B in order to gain a better understanding of each layer in Figure 4-1.

The following questions could thus be asked:

- Which AAL type should be utilised to transport AAL2 signalling messages? And if AAL2 is chosen,
- What are the implications of transporting AAL2 signalling messages inside AAL2 CPS packets?

These two questions were considered and investigated in [31] by means of simulation. The analysis conducted assumed the same overhead for the AAL2 signalling messages over both AAL2 and AAL5. It was found that transporting the signalling messages over AAL5 resulted in less delay variation, due to the AAL5 frame structure. However, transporting the signalling messages over AAL2 resulted in less overall signalling delay and thus a higher throughput when the network traffic density was low.

With regard to the second question, the same simulation study also concluded that it was more beneficial to use two separate queues inside each AAL2 node, instead of a single FIFO queue, when utilising AAL2 signalling messages in CCS. One queue would be dedicated to AAL2 signalling messages and the other to normal AAL2 user traffic. The signalling queue would then be assigned a higher priority than the user traffic queue. Consequently, the user traffic queue would only be serviced once the AAL2 signalling queue was empty.

4.3 Enhanced Techniques for Reducing AAL2 Signalling Delay

The previous Section has already discussed two techniques that would contribute to the lowering of the AAL2 signalling delay, namely transporting AAL2 signalling

messages over AAL2 in CCS and employing separate prioritised queues for the signalling and user data respectively.

It has also been proven, via simulation, that the AAL2 connection establishment phase can be even further reduced when prioritised handling of individual AAL2 signalling messages is introduced [32]. The signalling messages that should receive higher priority are Establish Request (ERQ) and Establish Confirm (ECF) since these two messages are responsible for creating an AAL2 connection. Two separate signalling queues were used in this simulation. The first queue collected the ERQ and ECF signalling messages while the second queue would collect the remaining signalling messages. Only once the first queue was empty, would the second queue be serviced. This resulted in a 12% faster connection establishment phase than when a single FIFO queue was utilised for all AAL2 signalling messages. However, this technique also resulted in an 11% increase in the connection release phase. Therefore, the overall handling time did not change significantly when signalling priorities were introduced. The different techniques discussed that can be utilised to lower the AAL2 signalling delay during AAL2 channel establishment are summarised as follows:

- AAL2 signalling messages should be transported in CCS over AAL2
- AAL2 signalling data should receive a higher priority than AAL2 user traffic
- ERQ and ECF messages should receive a higher priority than any other AAL2 signalling messages.

4.4 Choice of AAL2 Signalling Stack Interface

Interfacing of the AAL2 Signalling Stack (A2S) with the AAL2 Served User (SU) and the Signalling Transport Converter (STC) can be performed either by means of a functional or message-based interface. Figure 4-2 below illustrates these two possibilities in greater detail by only considering interfacing between the SU and A2S entities.

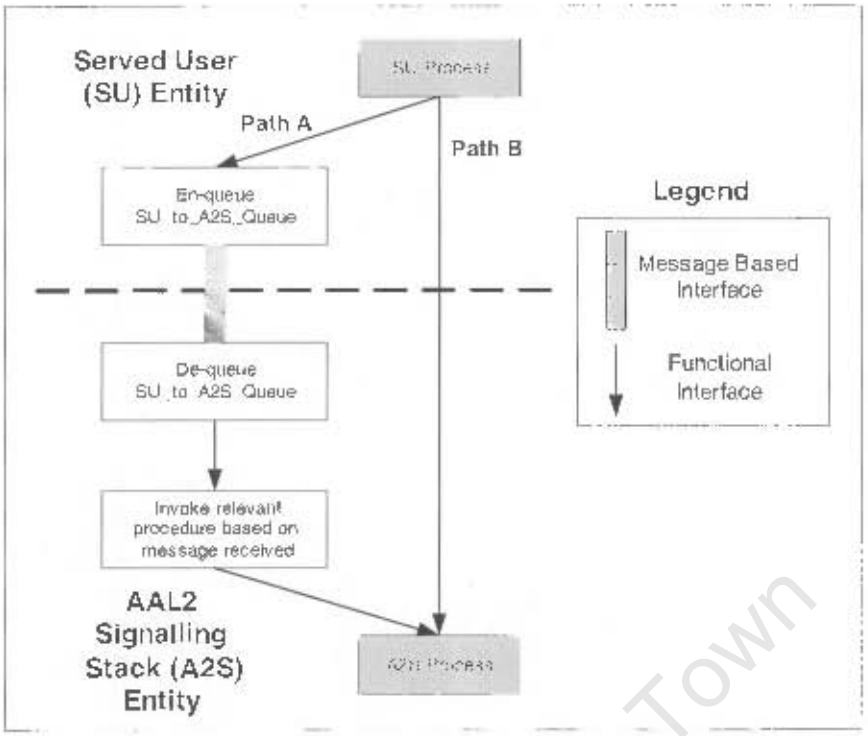

Figure 4-2 Interfacing with the AAL2 Signalling Stack

When a SU process needs to send information to an A2S process, the information can be transported either via path A, which is a message based interface, or via path B, which is a functional interface. Utilising path A results in the information from the SU process first being encapsulated in a message. This message is then en-queued by the SU process in a queue separating the two entities from one another. The A2S entity would then de-queue the received message, extract the relevant information and send it to the appropriate A2S process. In contrast, utilising path B results in the SU process simply sending the information to the relevant A2S process directly.

Although the functional interface may appear to be a simpler approach, the message based interface was chosen for the AAL2 signalling protocol framework design. This is because the message-based interface would enhance the modular design of the signalling framework and allow the framework to be designed as a stand alone entity. Although only one queue is indicated in Figure 4-2, four queues were utilised in the design. Two queues were utilised for interfacing between the SU and the A2S and the other two queues were utilised between the A2S and the STC. Two queues per interface were required since each queue only operated in a unidirectional manner.

4.5 High-Level AAL2 Signalling Framework Design

This Section illustrates and discusses a high-level design of the AAL2 signalling protocol framework along with its various components. The framework will be implemented at each AAL2 node contained within an AAL2 network. The methodology of the design is structured around a modular framework design. This ensures that any future changes due to technological or protocol related updates can be confined to small sections of the overall design.

The Protocol Entity, which defines the interaction between peer adjacent AAL2 nodes, is illustrated in Figure 4-3. It is divided into several modules in order to implement the desired functionality, as discussed in Section 3.3. These modules include:

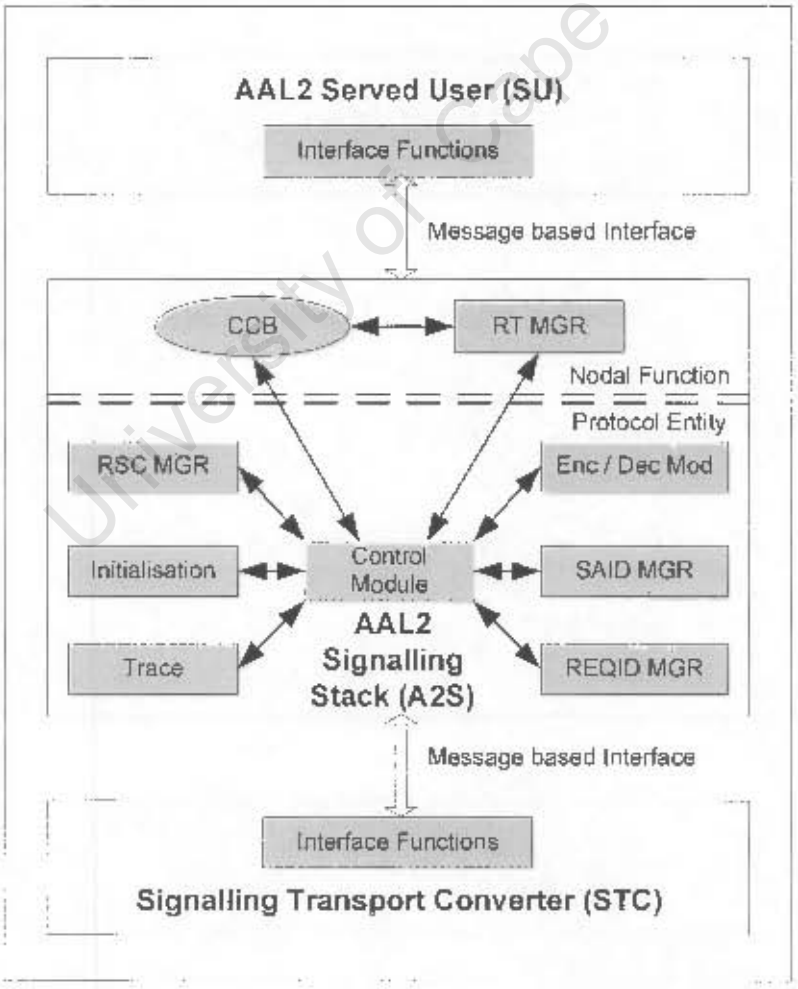

Figure 4-3 AAL2 Signalling Software Architecture

- **Signalling Association ID Manager (SAID MGR)**

  A Signalling Association Identifier (SAID) is utilised by peer AAL2 entities to identify a common AAL2 connection that exist between them. The allocation, release and management of these SAIDs are performed by the SAID Manager.

- **Request ID Manager (REQID MGR)**

  As two peer AAL2 entities communicate utilising a SAID, the individual modules within the AAL2 signalling stack communicate by means of a Request Identifier (REQID). The allocation, release and management of these REQIDs are performed by the REQID Manager.

- **Encoder / Decoder Module (Enc / Dec Mod)**

  This module is responsible for the encoding and decoding of AAL2 signalling messages. The encoder is invoked when a signalling message to be sent to a peer entity is formed. The decoder is invoked when a signalling message is received from a peer AAL2 entity.

- **Initialisation Module**

  The Initialisation module is utilised to initialise the AAL2 signalling stack at start up time. It allocates memory for various internal data structures and initialises these data structures with appropriate values.

- **Resource Manager (RSC MGR)**

  The Resource (RSC) Manager is responsible for collecting statistics relating to a specific AAL2 path. These statistics include the number of connections being processed by the specific path, the number of active connections, as well as the amount of bandwidth allocated to those active connections.

- **Trace Module**

  This module provides a tracing feature to facilitate in the debugging of the system during the development phase. It is also utilised to identify any errors that might arise during the operation of the signalling stack.

- **Control Module**

  Lastly, the Control module is central to the operation of the AAL2 signalling stack. It ensures that all the different modules mentioned above can operate independently of each other by providing a suitable common interface between these different modules.

As stated previously, the Nodal Function is responsible for handling routing queries, allocating resources and providing a mapping between the incoming and outgoing protocol entities. This is accomplished by dividing the nodal function into two separate modules, as indicated in Figure 4-3.

- **Connection Control Block (CCB)**

  For each AAL2 connection, the CCB module maintains an account of any resources allocated to the connection. These resources are then de-allocated when the connection is released. The CCB also maintains a primitive connection state to ensure that duplicate request messages are not sent with the same REQID value.

- **Routing Manager (RT MGR)**

  The Routing manager is a key component of the AAL2 signalling stack as it maintains routing information and handles all routing queries. This is accomplished by locating routes that can satisfy the resource requirements of each connection request.

Although no dedicated error handling module was defined during the design phase, error handling did form part of the design of each individual module. The following sections will discuss the most important modules mentioned above in greater detail. The particular order in which they are discussed is based on the flow of information through the AAL2 signalling framework.

4.5.1 Request ID Manager

The exchange of any connection related information between the AAL2 Served User (SU), AAL2 Signalling Stack (A2S) and the Signalling Transport Converter (STC) is performed by means of a unique identifier referred to as the request ID. This unique ID is used to identify individual connections between the different modules of the A2S. The Request ID Manager is responsible for allocating unique request identifiers to each request initiated by the SU or received by the STC and for releasing previously allocated request identifiers when a connection is terminated.

4.5.2 Routing Manager

Even though the AAL2 signalling protocol does not currently incorporate a separate routing protocol, a need still exists in order to identify addresses that can be reached from a specific AAL2 node, as well as a means of reaching them. Thus, the two most important functionalities residing in the routing manager are the identification of reachable addresses and the provision of adequate resources along this particular path. This process is also referred to as QoS based routing. QoS based routing locates a route through the network that will satisfy the resource requirements of the requested connection. Although the AAL2 signalling protocol as defined in Q.2630.1 provides support for both E.164 and NSAP addressing plans, a simpler proprietary addressing scheme was utilised in this study to identify individual AAL2 nodes and AAL2 users.

The primary structure of the routing manager is the routing table, as depicted in Figure 4-4. This table is used to search for a given AAL2 End-Point Address (A2EA) upon receiving an establish request. The table is a collection of A2EAs arranged in a sorted manner. Each A2EA entry consists of a corresponding set of Interface Identifiers (IFID) arranged in increasing order of cost. The cost value refers to the number of AAL2 nodes that need to be traversed in order to reach the desired destination.

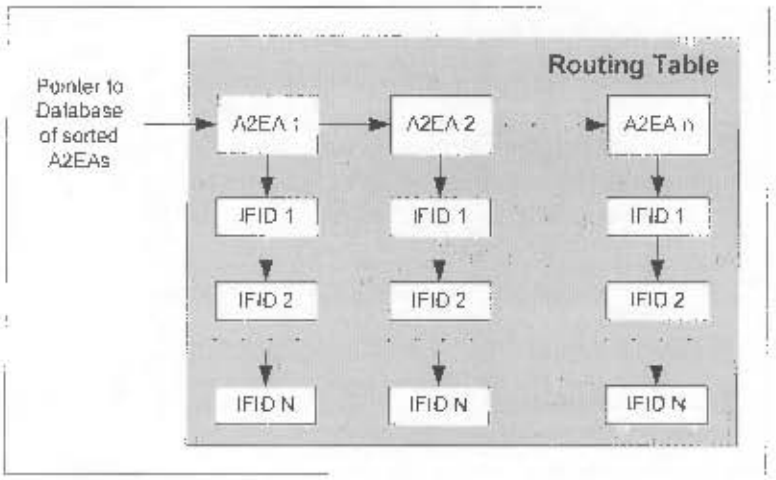

Figure 4-4 Diagram of the AAL2 Routing Table

Interfaces are maintained at node level where each node contains multiple interfaces. Each interface in turn contains multiple AAL2 paths and each path contains multiple AAL2 channels. A diagram illustrating the above mentioned hierarchy is depicted in Figure 4-5.

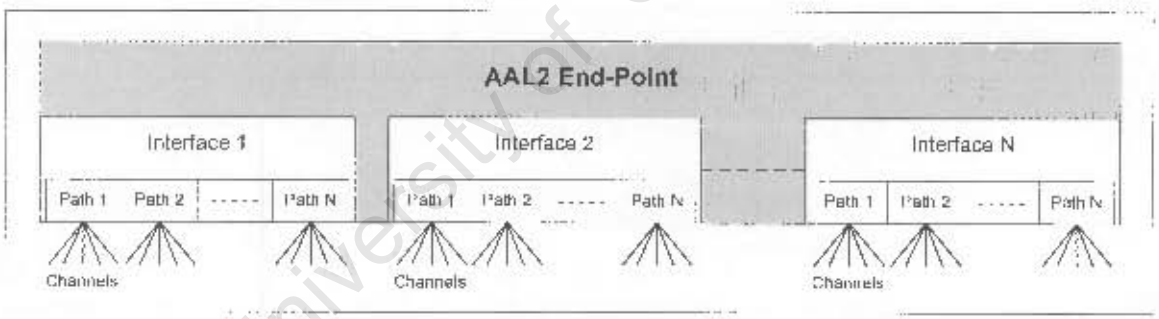

Figure 4-5 Interface, Path and Channel Hierarchy

Each AAL2 end-point contains one interface for every neighbouring node connected to it. Each interface, in turn, contains information relating to the paths contained in it. Finally, each path contains information relating to the path capacity and channel allocation, which is used for granting connection requests.

Whenever an establish connection request is received for a particular A2EA, a query is performed to determine whether the requested address is reachable through this specific node. The steps followed are:

- Firstly, the routing table is traversed to locate the requested A2EA. If this A2EA is not located then an error is indicated. However, if the address is found then the query proceeds to the second stage.

- Secondly, the first interface in the set of interfaces under this specific A2EA is selected, since the interfaces are arranged in increasing order of cost.

- Thirdly, the set of paths specific to this interface are traversed to locate a path that would be able to support the requested link characteristics.

If no paths are able to support the link characteristics for this particular interface, then an error is indicated and the next interface of the set, which has a higher cost value, is examined until a suitable path has been located. However, if no suitable path can be located for the connection request, then a release confirm (RLC) signalling message is returned to the requested user with a reason for the failed connection.

Connection Admission Control (CAC) is performed in order to determine whether a selected path is able to support the requested AAL2 link characteristics. The CAC algorithm ensures that the resource requirements of the new connection can be supported without negatively affecting any existing connections in the path. The algorithm is based on a two step procedure. Firstly, the actual bandwidth required is calculated based on the average and maximum link characteristics of the connection request. This is performed by means of the following formula:

$$BW_{n+1} = ave\_rate + \alpha(peak\_rate - ave\_rate)$$

In the formula BW_{n+1} is the bandwidth required for the new connection and α is a constant representing the statistical multiplexing gain employed. If α is set to 0 then the bandwidth is based on the average rate only and a possibility for congestion exists. However, if α is set to 1 then there is no statistical gain. Typically, a value between 0.1 and 0.4 is chosen. Once the bandwidth is calculated, the second step is to determine whether the bandwidth can be supported on the specific path. This is performed using the formula:

$$\text{if}\,(BW_{n+1} < (BW - \sum_{i=1}^{n}(BW_i)))$$

Accept

else

Reject

In the above formula BW refers to the total bandwidth of the specific AAL2 path.

4.5.3 SAID Manager

The exchange of AAL2 signalling message information between peer AAL2 nodes is performed by utilising a unique identifier referred to as the Signalling Association Identifier (SAID). These identifiers uniquely identify signalling messages between two specific AAL2 nodes. All signalling messages are required to be associated with a unique SAID. When a connection request is processed, a unique Originating SAID (OSAID) is generated for the signalling message for a given request ID. On reception of the signalling message at the remote node, a corresponding Destination SAID (DSAID) is allocated to the signalling message. The SAID Manager ensures that these identifiers are randomly generated with no duplicates. This is recommended by the ITU-T and done as a precautionary measure.

4.5.4 Connection Control Block Database

The Connection Control Block (CCB) database maintains a record of any resources allocated to each connection being processed by the A2S. This insures that the A2S can manage its internal and external resources in the event of a normal or abnormal termination of a connection. Connection related information includes:

- OSAID generated for a request
- DSAID received for a request
- Interface ID where path is located
- Path and channel IDs
- AAL2 end-point address for the connection

- Connection status
- Timer status

4.6 Message Flow within the AAL2 Signalling Stack

This section will discuss the flow and processing of information within the AAL2 signalling stack for two situations. The first will consider the processing of an establish connection request message from the SU entity. The second will discuss the reception of an establish connection request message from the STC entity at the remote node.

4.6.1 Initiating an Establish Connection Request

Based on the message-based interface method employed by the AAL2 signalling framework, the processing of a connection request, from the SU entity, will commence with the de-queuing by the A2S process of a request message from the queue between the SU and the A2S. This connection request is initiated by a user located in the SU entity. Figure 4-6 is a diagram depicting the process of information flow through the various modules of the AAL2 signalling framework. If the request is successfully processed, then an ERQ signalling message is formed and en-queued in the queue between the A2S and the STC by the A2S process, which is then sent to the destination AAL2 node. Although not indicated in Figure 4-6, once an ERQ signalling message is transmitted a timer function is initialised. This is done as a precautionary measure to ensure that the initiating user does not unnecessarily occupy system resources in the event that the peer AAL2 node is not functioning as expected. Therefore, if an appropriate response, in the form of either an ECF or a RLC message, is not received from the peer AAL2 node within a specified period, the connection request is terminated. An ECF message will signal a successful connection establishment to the initiating node, whereas a RLC message signals that the connection request was unsuccessful.

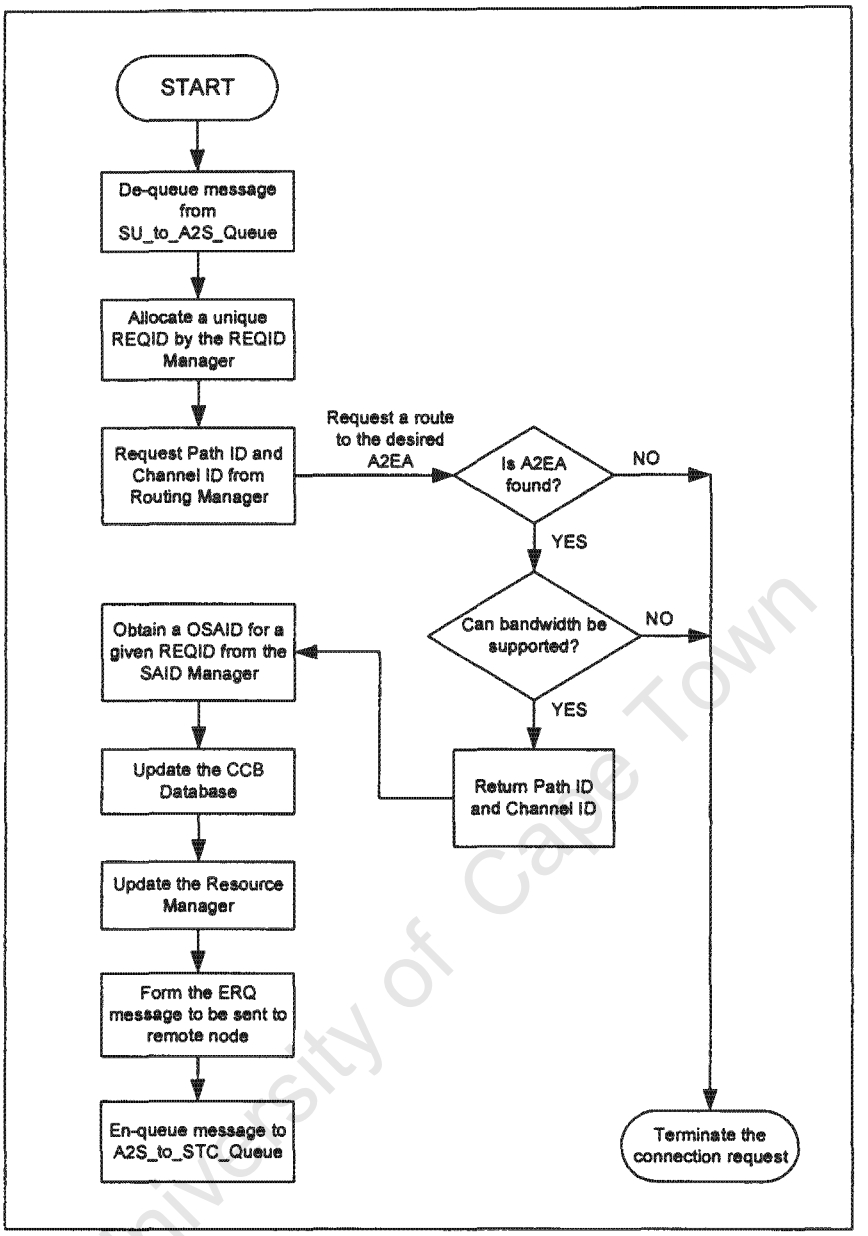

Figure 4-6 Flow of Processes for an Initiating ERQ Message

4.6.2 Receiving an Establish Connection Request

The remote AAL2 node commences the processing of a connection request message by de-queuing the request message from the queue located between the STC and the A2S. Figure 4-7 illustrates the processing of the connection request message by the various modules of the AAL2 signalling stack.

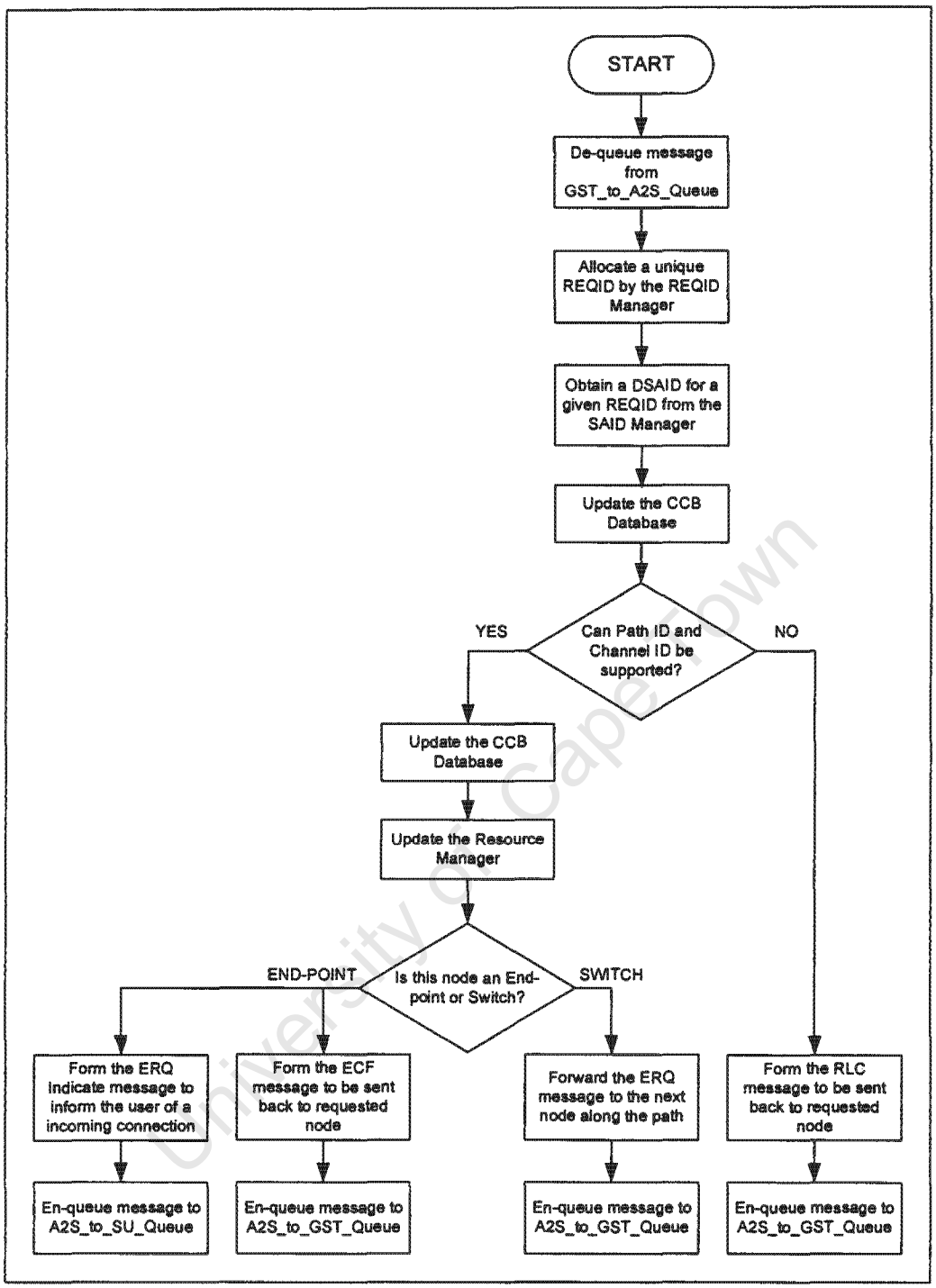

Figure 4-7 Flow of Processes for an Received ERQ Message

If the requested connection element identifiers, such as the path ID and the channel ID are already in use, the request is unsuccessful and a RLC message is sent back to the initiating node. However, if the connection element identifiers are available, then the request is successfully processed. If the current node processing the request is an AAL2 switch and not the desired AAL2 end-point, then the data structures of the

various modules in the A2S are updated to reflect a connection traversing the switch and the ERQ message is forwarded to the next node along the path. If the current node is in fact the desired end-point and the request was successful, then two messages are formed and en-queued. The first of these messages is an ECF message sent back to the initiating node informing it that the connection request had successfully been processed. This message is en-queued in the queue between the A2S and the STC. The second message is destined for the SU entity and is en-queued in the queue between the A2S and the SU. This message indicates that a new connection has been established for a specific user located in the SU entity.

Chapter 5

Architecture of the AAL2 Signalling Framework

The development of this Chapter commences with a logical scenario to illustrate how AAL2 switching could be accomplished in a network by means of AAL2 signalling. This is followed by two distinct sections discussing the primary implementation phases in this study. The first section concentrates on the implementation required to establish an end-to-end AAL2 connection via the AAL2 signalling protocol framework. The design and architecture of this framework was described in Chapter 4. The second section of this Chapter will discuss the AAL2 switching node architecture, with emphasis placed on the implementation of this signalling framework on the AAL2 switching node architecture. It is essential that the overall system architecture meets as many of the design specifications as possible. These specifications, as stipulated in Chapter 4, include modularity, scalability and efficiency. This will ensure that the evaluation platform functions as intended within the signalling framework design.

5.1 Logical AAL2 Switching and Signalling Scenario

A scenario requiring both AAL2 signalling and AAL2 switching to transfer information between AAL2 users, by means of a hybrid ATM / AAL2 network, is illustrated in Figure 5-1. The diagram includes a number of users all connected to various AAL2 concentrators. All of these users are considered to be AAL2 users. In turn, the AAL2 concentrators are connected via the hybrid network, consisting of ATM and AAL2 switches.

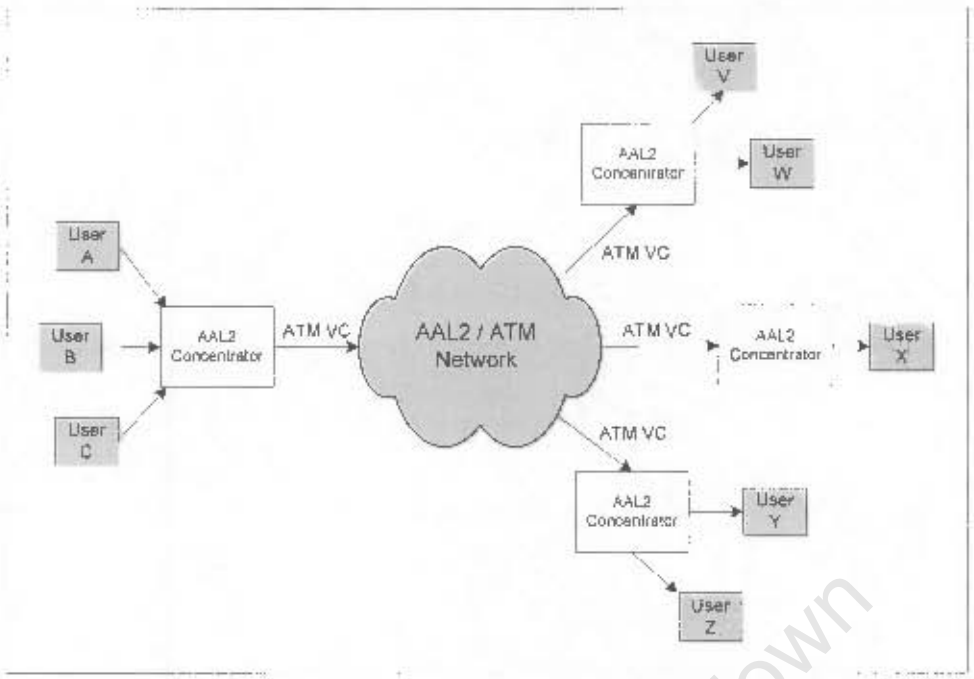

Figure 5-1 Logical Test-Bed Implementation

Each AAL2 concentrator and its associated AAL2 users are located at different physical locations. An AAL2 signalling protocol entity is also located in each AAL2 concentrator as well as in each AAL2 switch.

Suppose AAL2 users A, B and C need to transfer AAL2 traffic to AAL2 users W, X and Z respectively. Since AAL2 users A, B and C are connected to the same concentrator, only one ATM VC or AAL2 path is required between its concentrator and the ATM network. But three individual ATM VCs are required between the hybrid network and the AAL2 concentrators connected to AAL2 users W, X and Z. Firstly, ATM signalling is utilised by the ATM network operator to provide the relevant ATM VCs with the required QoS and bandwidth characteristics. Once completed, the AAL2 network operator is then able to provide the necessary AAL2 connections. Each initiating AAL2 user then stipulates its bandwidth requirements at each AAL2 concentrator and AAL2 switch in the path, via AAL2 signalling. If the ATM VCs are able to support these bandwidth requirements, then AAL2 signalling would establish these AAL2 connections inside the ATM VCs so as to connect the corresponding AAL2 users together. Thus, one incoming ATM VC, from AAL2 users A, B and C, would contain three multiplexed AAL2 channels. The AAL2 traffic contained in that ATM VC would then be separated by means of an AAL2 switch,

located in the hybrid network, to direct the correct AAL2 traffic to its corresponding destination.

The sections that follow provide more detailed information regarding the implementation of the above mentioned scenario on a practical evaluation platform.

5.2 Practical AAL2 Switching and Signalling Evaluation Platform

Designing and implementing a control-plane AAL2 signalling framework forms the second phase of a larger three phase project aimed at designing a fully functional AAL2 switching node. This switching node would form part of a hybrid ATM / AAL2 network. An experimental gigabit ATM research switch developed by Washington University in St. Louis, called the Washington University Gigabit Switch (WUGS), is being used to implement this AAL2 switching node design. The first phase of the project was concerned with designing and implementing the user-plane AAL2 switching component [12] [33]. The third and final phase of this project, which is the management-plane functionality, is necessary to support the AAL2 signalling framework. This final phase will be discussed in Chapter 8. The signalling framework therefore had to be integrated with the existing switching component. However, the integration process was hampered by certain challenges and limitations, since the AAL2 switching component had to be enhanced to support the full functionality of AAL2 signalling framework. These challenges and limitations will be discussed throughout this Chapter.

A simplified subset of the logical scenario illustrated and explained in Section 5.1 is indicated in Figure 5-2. The figure consists of three distinct components, namely two AAL2 end-points and an AAL2 switching node.

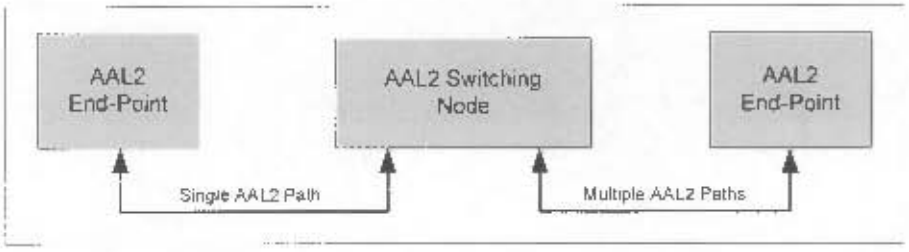

Figure 5-2 Simplified Test-Bed Implementation

Each AAL2 end-point consists of a few user instances, an AAL2 signalling entity and an AAL2 concentrator module. The AAL2 concentrator module multiplexes the AAL2 user traffic unto a single ATM VC. Once the AAL2 traffic is transported to the AAL2 switch, the switch performs the AAL2 switching functionality and forwards the AAL2 traffic to the second AAL2 end-point using multiple AAL2 paths. Multiple AAL2 paths are necessary, because although all the destination AAL2 user instances are contained in the same destination AAL2 end-point, they are logically considered to be located at different physical locations. The second AAL2 end-point will thus provide the illusion of being a number of separated AAL2 end-points with its own AAL2 users. The provisioning for these AAL2 channels will be performed by means of AAL2 signalling.

As mentioned previously, the first phase in developing a fully functional AAL2 switching node involved designing the underlying switching functionality required to switch individual AAL2 CPS packets, as discussed in Section 2.4. Therefore, although simplified end-points were developed in [12], its primary focus was in developing and implementing an AAL2 switching node with the required modules. The end-points were merely necessary to send and receive AAL2 user data to and from the switching node respectively, in a unidirectional manner. The primary focus was on bandwidth efficiency and data integrity. Subsequently, various voice codecs, with varying packet lengths, were utilised to build AAL2 cells that were transmitted from one end-point, acting as a client, to a second end-point, acting as a server, via the AAL2 switching node. The switching node, in turn, would disassemble each received AAL2 cell into its individual CPS packets. All CPS packets destined for a common AAL2 end-point were then reassembled into new AAL2 cells and transmitted to their desired destinations. At the destination end-point, the data were evaluated for data integrity to

ensure that both end-points and switching node were functioning correctly. Also, various timer options at the client AAL2 end-point, concerned with when to transmit AAL2 cells, were evaluated. This was done in order to ensure the required QoS of the supported user applications and to determine an ideal timer range for bandwidth efficiency.

The AAL2 end-point development therefore had to be modified extensively to support the requirements of the AAL2 signalling framework. As an example, bidirectional AAL2 connections are required since requesting signalling messages are complimented by means of replying signalling messages that need to be transmitted on the same AAL2 connection.

5.3 AAL2 End-Point System Architecture

This section will focus on the modified architecture of the AAL2 end-point. Specifically, it will highlight the functionality of the various components required in an AAL2 end-point in order to support the AAL2 signalling framework. As discussed in Chapter 4, AAL2 signalling messages were to be transported over AAL2 as common channel signalling, in the same ATM VC as the AAL2 user traffic. This simplified the implementation by not having to be concerned with the complexity involved in utilising two separate AALs, namely AAL2 and AAL5.

As illustrated in Figure 5-3, the architecture of an AAL2 end-point consists of three distinct components, namely various AAL2 Users, an AAL2 Signalling Entity and an AAL2 Transport Entity. The signalling entity is responsible for establishing, maintaining and releasing AAL2 channels as requested by the various AAL2 users. The transportation of the required AAL2 signalling messages to remote nodes is the responsibility of the AAL2 transport entity.

The AAL2 end-point architecture presented in Figure 5-3 is merely a simplified version of the complete AAL2 signalling stack as discussed throughout this study and included in Figure 5-3.

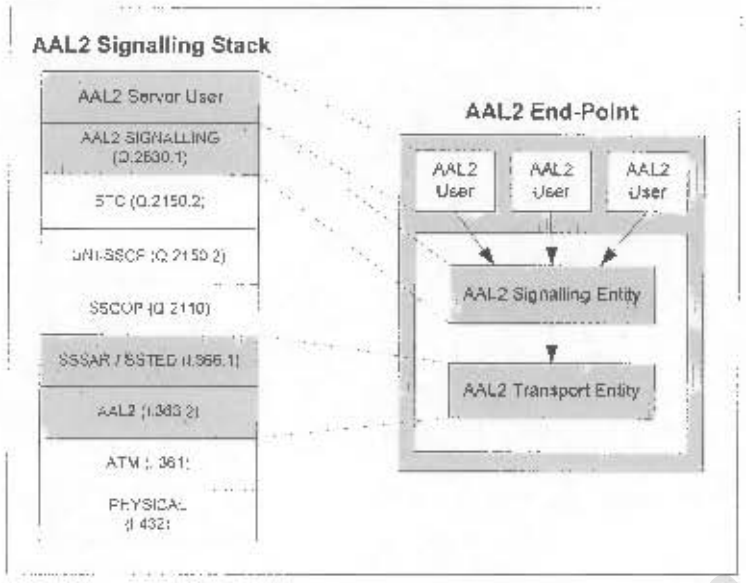

Figure 5-3 System Architecture of an AAL2 End-Point

In contrast to the complete AAL2 signalling stack, which consists of a total of six layers, excluding the ATM and physical layers, the AAL2 end-point only consists of the four primary layers as illustrated in the diagram. These layers are the served user layer, the AAL2 signalling layer, the Segmentation and Reassembly (SAR) layer and the AAL2 transportation layer. The layers omitted from the end-point design are responsible for providing services such as error recovery, flow control and connection control. However, since the end-point design was evaluated in a controlled environment these layers were not essential to the end-point system architecture and were hence omitted from the design. Both the served user and the AAL2 signalling entities are based on ITU-T recommendation Q.2630.1. However, the AAL2 transport entity is a combination of ITU-T recommendations I.363.2 and I.366.1. This is required since recommendation I.363.2 only provides a means by which small, 45 or 64 byte, AAL2 CPS packets can be transported across an AAL2 channel[10]. The inclusion of I.366.1 in the AAL2 transport entity provides a method of transporting larger data packets[11]. This is necessary since the size of the AAL2 signalling messages, to be transported to remote AAL2 nodes, are larger than what was catered for in recommendation I.363.2.

[10] See Section 2.2 for further details.
[11] See Section 2.3 for further details.

The AAL2 signalling entity is a direct implementation of the signalling framework, as presented and discussed in Chapter 4, with all its components included. The design and implementation of the AAL2 transport entity falls outside the scope of this study, as it was presented and discussed extensively in [12]. However, various modifications to the transport entity were required to support AAL2 signalling. These modifications are discussed in the following Section.

5.3.1 Modifications to the Transport Entity to Support AAL2 Signalling

The primary modification performed on the AAL2 transport entity, as discussed earlier, was to provide support for AAL2 signalling by means of bidirectional AAL2 connections. This is required since signalling messages need to be transported to and from peer end-points on a common AAL2 connection.

A further enhancement required by the AAL2 transport entity was the ability to transport the larger AAL2 signalling messages, which exceeds the maximum allowable CPS packet size of 64 bytes. A subset of ITU-T recommendation I.366.1 therefore had to be implemented and integrated into the transport entity in order to cater for these larger signalling messages. However, once completed, it was found that AAL2 signalling messages did not arrive at their desired end-point destination as expected. Instead, the signalling message information was incomplete and therefore corrupt. Upon investigation it was found that the finite state machine utilised by the AAL2 transport entity did not operate as expected. The initial purpose of the transport entity was merely to transmit many small fixed sized AAL2 CPS packets inside AAL2 cells. However, transmitting the final CPS packet of a large segmented file resulted in corrupt data. Since only large files were transmitted across the unidirectional connections in [12] in order to evaluate issues relating to bandwidth efficiency, the final CPS packet was always discarded. Therefore, in order to successfully transmit and receive an AAL2 signalling message between peer nodes without any data corruption a subset of I.366.1 had to be implemented. This modification to the finite state machine enabled it to handle the final CPS packet of an AAL2 signalling message correctly.

Also, when packaging CPS packets into AAL2 cells a timer module was utilised to determine when these AAL2 cells should be transmitted. This was required since although AAL2 cells should be filled with as much data as possible for reasons of bandwidth efficiency, care needs to be taken so as not to exceed the end-to-end delay associated with AAL2 voice traffic. Therefore a timer module would signal to the transmitter module when to transmit AAL2 cells, even if these cells are not completely filled with CPS packets containing user data. However, signalling information needs to be transmitted from source to destination end-points as speedily as possible in order to minimise signalling delays. Subsequently, the timer module in the transport entity was adapted so as only to be active for AAL2 connections transporting CPS user packets and inactive for AAL2 channels transporting signalling information.

5.4 AAL2 Connection Establishment Procedure

This section will discuss the procedures and functionality required to establish an end-to-end AAL2 connection between two peer AAL2 end-points. Firstly, the logical flow of information between two peer AAL2 end-points will be examined. This will be followed by a discussion describing how the various entities of an end-point are interconnected in order to achieve this functionality.

5.4.1 Flow of Information Required to Establish an AAL2 Channel

The various steps required in establishing an end-to-end, non-switched, AAL2 connection between two peer AAL2 end-points are illustrated in Figure 5-4 below. In the figure AAL2 user 1, located in the originating AAL2 end-point, initiates the establishment of a new AAL2 connection to AAL2 user 4, located in the destination AAL2 end-point. For ease of description the individual steps required are numbered and listed as follows:

1) AAL2 user 1 sends an Establish Request message for an AAL2 connection which is en-queued.
2) AAL2 signalling entity de-queues the Establish Request message for processing.

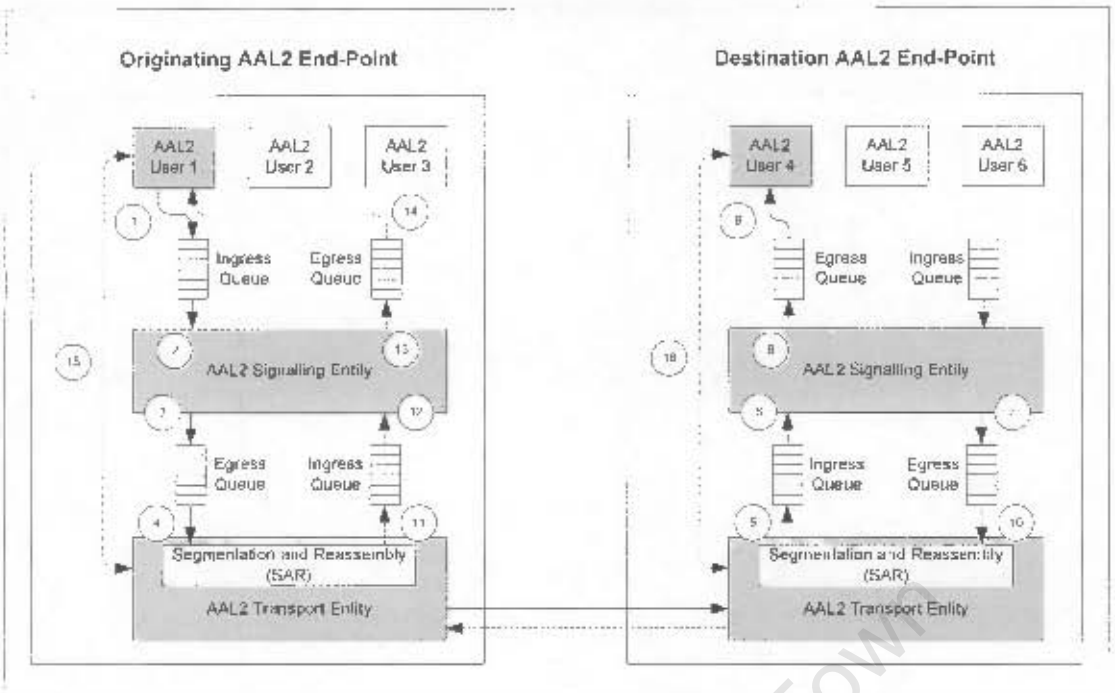

Figure 5-4 End-to-End AAL2 Channel Establishment Process

3) AAL2 signalling entity forms an ERQ message destined for the destination AAL2 end-point and en-queues the message.
4) The AAL2 transport entity de-queues the ERQ message and then segments this large message into multiple smaller CPS packets. It then sends these CPS packets to the destination AAL2 end-point.
5) On reception of these CPS packets, the destination AAL2 transport entity reassembles the small CPS packets to form the original ERQ message and en-queues the message.
6) The AAL2 signalling entity then de-queues the request message and takes appropriate action on the basis of the request.
7) The AAL2 signalling entity responds to the Establish Request message by sending a confirmation message back in the form of an ECF message, which is en-queued.
8) An Establish Indicate message destined for AAL2 user 4 is also formed by the AAL2 signalling entity and en-queued.
9) AAL2 user 4 de-queues the Establish Indicate message and is informed that the remote AAL2 user 1 is creating a new connection to it.

10) The AAL2 transport entity de-queues the confirmation message and then segments this large message into multiple smaller CPS packets. It then sends these CPS packets to the originating AAL2 end-point.
11) On reception of these CPS packets, the AAL2 transport entity reassembles the small CPS packets to form the original confirmation message and en-queues the message.
12) The AAL2 signalling entity then de-queues the confirmation message to be processed.
13) If the request is successful, an Establish Confirmation message is sent to AAL2 user 1 and en-queued.
14) AAL2 user 1 de-queues the confirmation message and is informed that the end-to-end AAL2 connection has successfully been established.
15) This enables AAL2 user 1 to send user data directly to the AAL2 transport entity, which is intended for the destination AAL2 user by means of the newly created AAL2 channel.
16) At the destination AAL2 end-point, the received user data will simply be forwarded directly to AAL2 user 4, without passing through the AAL2 signalling entity.

An end-to-end AAL2 connection has thus successfully been established between two peer end-points and AAL2 user data can thus be transported between these end-points. It is important to note that the AAL2 signalling protocol at the destination end-point acknowledges the connection establishment request in step 7 prior to informing the served user of the incoming connection in step 8. This technique contributes to lowering the AAL2 connection establishment delay. Although not indicated in Figure 5-4, the connection release procedure required to tear-down the AAL2 connection follows a very similar flow as that illustrated for creating an AAL2 connection.

5.4.2 Message-Based Queuing Implementation

The four unidirectional queues illustrated at each end-point are based on the message-based interface technique employed by the AAL2 signalling framework so as to isolate it from its neighbouring layers. This technique, as presented in Section 4.4,

added to the modular design methodology of the signalling framework. Consequently, the author was able to develop and test the signalling framework in isolation. Various request messages originating from AAL2 users were en-queued and processed by the signalling framework. This enabled emphasis to be placed on the scalability of the signalling system. Each module within the signalling framework was tested to determine if the data structures utilised by the individual modules scaled correctly as the number of simultaneous connections was increased. After processing a specific request message, appropriate signalling messages were then formed and en-queued to be sent to the underlying transport entity.

Although Section 4.3 discusses the utilisation of multiple priority queues instead of single FIFO queues for sending information between the AAL2 signalling entity and its neighbouring layers, priority queues were not utilised in this study. This is because the benefits of priority queues are only noticeable in a highly loaded system where many different signalling messages are all contending for processing time from the signalling entity. For the purpose of this study, only four of the eleven AAL2 signalling messages were utilised and implemented. The remaining signalling messages are management related messages, which are not concerned with establishing or tearing down AAL2 connections. Consequently, the four unidirectional FIFO queues implemented at each AAL2 end-point proved adequate for this study as the primary deliverable was to develop a functional system. Each FIFO queue was implemented as a circular FIFO buffer of signalling messages. These signalling messages, or elements, were en-queued and de-queued to and from the FIFO buffer by means of a memory copy technique.

5.4.3 ATM Connection Establishment Implementation

As discussed previously, prior to the AAL2 channel establishment process, an ATM VC or AAL2 path needs to be established between peer AAL2 nodes. Section 3.4.1 discussed the choices of ATM service classes that could be utilised for the ATM VCs supporting AAL2 traffic, namely CBR, rt-VBR or UBR. It was concluded that although UBR is the most bandwidth efficient service class, it is however not ideally suited for the transportation of delay sensitive voice traffic. It was therefore decided

that ATM VCs transporting AAL2 traffic, consisting of both AAL2 signalling information and user traffic, should be provisioned as rt-VBR connections. This is because rt-VBR supports both bandwidth guarantees, of compressed voice traffic and signalling information, as well as cell transfer delays, as required by AAL2 traffic enabling it to adhere to strict end-to-end delays. Also, ATM switches prioritise different traffic streams based on the service class of the ATM connection. Therefore, rt-VBR VCs would receive a higher priority than UBR VCs, thus resulting in lower delays for rt-VBR connections.

However, the physical hardware utilised in the platform evaluation does not support any particular ATM service class. Each AAL2 end-point was implemented on a general purpose PC and thus required a Network Interface Card (NIC) to connect to the experimental ATM research switch utilised for the AAL2 switching node implementation. The research switch, called the WUGS switch, was connected to each PC by means of a fibre optic cable. The NIC, located in each PC, contains a high performance network interface chip called the ATM Port Interconnect Chip (APIC). In this study, ATM VCs were provisioned as Permanent Virtual Circuits (PVCs) by means of the APIC chip. The type of VCs supported by the APIC chip are referred to as *low delay*, *paced* and *best effort* and resemble the ATM service classes CBR, rt-VBR and UBR respectively. The platform evaluation utilised the *best effort* option, although in a realistic implementation environment where many VCs are supported, the VCs supporting AAL2 traffic should be provisioned as *paced*, which resemble the rt-VBR service class. Care should however be taken when assigning the specific traffic characteristics of these ATM VCs, since the link characteristics of all subsequent AAL2 channels, established inside these ATM VCs, need to be supported.

5.4.4 AAL2 Signalling Channel Assignment

After the provisioning of the individual ATM PVCs between the AAL2 nodes, AAL2 signalling messages are assigned a dedicated AAL2 channel in each ATM VC. Each node would thus send and receive signalling messages in a common channel of a specific AAL2 path. This study utilised the value CID = 8 for this purpose. The AAL2 signalling entities located in each node would then assign new CID values to each

new connection request starting with a CID value of 16. This CID value assignment complies with the ATM Forum's Loop Emulation Service Using AAL2[12]. Since AAL2 connections are bidirectional, the same CID value was used to identify both directions of the channel.

5.5 AAL2 End-Point System Integration

One of the design approaches decided upon during the development of the AAL2 end-point was to develop a complete end-point entity that could function as either an initiating end-point or destination end-point. An initiating end-point refers to an end-point that would initiate the establishment of a new AAL2 connection, whereas a destination end-point would receive the request from the initiating end-point and reply with an appropriate response. Thus the same AAL2 end-point entity would need to operate in either initiating or destination mode.

5.5.1 Building of the AAL2 Node Routing Table

Although this approach would simplify the deployment process of AAL2 end-points at the edge of a network, it provided the author with a number of challenges. Upon investigation it was discovered that the only signalling module contained within an AAL2 node, whether end-point or switch, that differed from other nodes in a network was the contents of the routing module. This is because each node is assigned a unique node address and each node occupies a unique location within a network. Therefore, each node's routing table entries must reflect the network topology as seen from that specific node. Since the AAL2 signalling protocol does not include a routing protocol, as discussed in Section 3.7, the routing tables of each node had to be populated and updated statically. Consequently, at system start-up, each AAL2 node, and hence each end-point, populated its routing table by loading a uniquely preconfigured parser file. This parser file contained specific information regarding which AAL2 End-point Addresses (A2EA) could be reached from a particular node through a specific interface identifier and by means of a particular AAL2 path[13].

[12] As defined by the ATM Forum in af-vmoa-0145.000
[13] The routing table architecture was presented and discussed in Section 4.5.2.

Therefore, wherever an AAL2 endpoint or AAL2 switch is placed anywhere within an AAL2 network, the AAL2 network operator only needs to assign the node a unique address and configure its parser file according to the position of this new node within the network topology.

5.5.2 AAL2 End-Point Integration through Threading

At any one time during the operation of an AAL2 end-point, AAL2 traffic in the form of either signalling messages or user data need to enter and leave the end-point. Subsequently, a second challenge faced in developing a complete AAL2 end-point entity is that the end-point needs to be able to operate in both a sending mode and a receiving mode simultaneously. Therefore, in addition to each AAL2 end-point consisting primarily of an AAL2 signalling entity and an AAL2 transport entity, two primary threads, namely a *server thread* and *user thread* were utilised to implement this feature. Threads were required since NetBSD 1.4.1, which is the Operating System (OS) employed by the hardware components utilised in this study, is not a real-time operating system. Threads are essentially lightweight processes that enable multiple operations to be performed on a single CPU. Although a CPU can only perform one task at a time, it appears as if the CPU is able to execute many different tasks simultaneously as each task receives a portion of processing time by the CPU in typically a round-robin manner. Utilising threads in the AAL2 end-point resulted in a non-preemptive cooperative system. Each thread is given full control of the CPU once active and must relinquish control back to the scheduler to give another thread the opportunity to be executed. Consequently, each thread needs to yield control back to the scheduler in a cooperative way.

The following section will discuss the thread execution utilised in the AAL2 end-point in more detail.

5.5.3 AAL2 End-Point Thread Execution Procedure

Figure 5-5 illustrates the various threads utilised at each AAL2 end-point for both the initiating end-point, which initiates the creation of new AAL2 channels, and the

destination end-point, which receives these requests. It is important to note that the two diagrams depicting the initiating AAL2 end-point and the destination AAL2 end-point are symmetrical. This was done to synthesise the fact that each end-point is effectively the same end-point system merely functioning in two different capacities and on separate development stations. Although the two end-points are in fact peer entities they are referred to as being an initiating-destination model in this example. It should also be noted that the following example illustrates the connection establishment procedure for multiple AAL2 connections. For ease of explanation, the various threading procedures required to establish these end-to-end non-switched AAL2 connections are numbered.

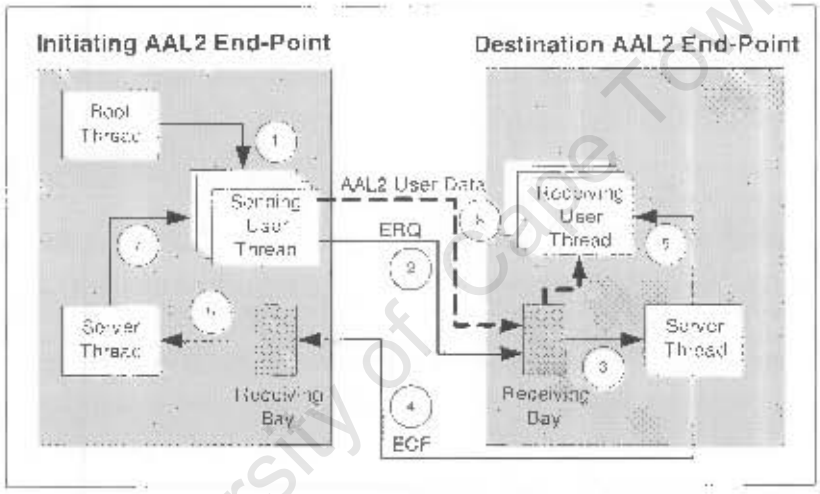

Figure 5-5 Thread Execution Procedure for AAL2 Channel Establishment

Whenever the system is initialised, the initiating end-point spawns a *boot thread* and a *server thread*, whereas the destination end-point only spawns a *server thread*. The *boot thread* then spawns a number of *sending user threads*, depending on the number of AAL2 users to be supported by the system. Each *sending user thread* comprises of both signalling and transport instances as it is able to form and transmit signalling messages. Further, each *server thread*, located on either end-point, also comprises both signalling and transport instances as it is able to receive, form and transmit signalling messages. After being spawned, each *sending user thread* generates an individual ERQ signalling message to be sent to the destination end-point. Each *sending user thread* is then suspended as it awaits a reply from the destination end-point.

As traffic arrives at the destination end-point, the receiving bay determines the type of traffic received. If the received traffic is signalling information it will be forwarded to the *server thread* otherwise, the AAL2 user data will simply be forwarded to the relevant *receiving user thread.* In this case, the information sent was signalling messages, which are forwarded to the *server thread* in the destination end-point. Each request is individually processed and if the connection request was successful then an ECF confirmation signalling message is generated and sent back to the initiating *sending user thread* on the initiating end-point. In addition, a new *receiving user thread* is spawned on the destination end-point to receive any future AAL2 user data sent from the corresponding *sending user thread.*

After the ECF message is received by the initiating end-point, the receiving bay once again determines the traffic type. Since signalling information was sent, the message is forwarded to the *server thread* on the initiating end-point. Once this information has been processed, the *server thread* informs the relevant *sending user thread* that the channel has successfully been established. Only then is the *sending user thread* able to transmit AAL2 user data to the remote *receiving user thread,* as indicated with the dashed lines in Figure 5-5. After the relevant AAL2 user data has successfully been transmitted, the initiating *sending user thread* commences the release procedure, which again follows a similar process as above to release the AAL2 channel. Each end-point will terminate its corresponding user threads once the connection is released.

Since each end-point system is implemented on a cooperative multitasking OS, each thread receives execution time from the CPU in a round-robin manner. Therefore, in order to maximise the performance of the system, each *sending user thread* wishing to transmit user data needs to yield control back to the scheduler on a regular basis. This method aims to ensure fairness among the respective transmitting *sending* and *receiving user threads.* A good scheduler mechanism therefore needs to be employed to guarantee the specified QoS for each supported application. If this was not done, each *sending user thread* would simply transmit all of its user data before any other *sending user thread* is able to. This would negatively influence the performance of the system and violate the end-to-end transit delay requirements of the supported

application, as the last *sending user thread* would need to wait until all other user threads have transmitted their data before it receives an opportunity to do the same.

In the event that only one *sending user thread* is spawned by the *boot thread* at system start-up, only one AAL2 connection will be established and only one *receiving user thread* will be spawned. However, all subsequent steps as illustrated in Figure 5-5 and described above will stay unchanged.

5.6 AAL2 Switching Node Architecture

Any generic switching node consists of three distinct yet integrated components, namely the control-plane component, the user-plane component and the management-plane component. The control-plane component is responsible for creating end-to-end channels through the carrier network by means of signalling. After the successful creation of such a channel, the user-plane component is responsible for transporting the relevant user data across the newly created end-to-end connection. Finally, the management-plane module is responsible for creating and updating the content of the switching node's routing table. The allocation of resources to new channels, by means of connection admission control, is also the responsibility of the management-plane component.

The second phase of the evaluation platform developed in this study required the control-plane AAL2 signalling framework to be integrated with the user-plane AAL2 switching node architecture [12]. As discussed previously, the AAL2 signalling framework forms the second phase of a three phase project aimed at developing a functional AAL2 switching node. A logical diagram illustrating the complete architecture of the AAL2 switching node is illustrated in Figure 5-6.

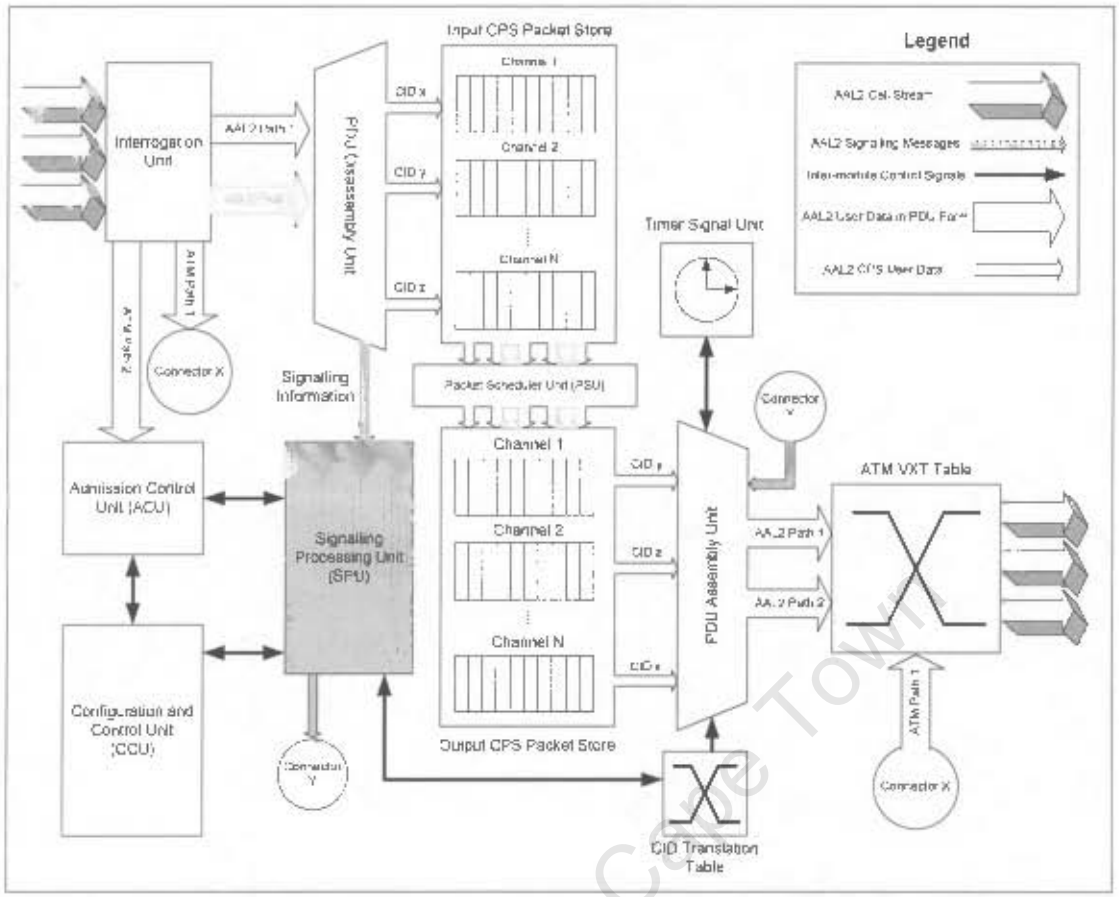

Figure 5-6 Modular Architecture of the AAL2 Switching Node

From the diagram it can be seen that AAL2 traffic traverses the switching node in a unidirectional manner from left to right. Due to the complexity of the switch design, the switch was only developed to process traffic in a unidirectional manner. The different information streams that flow inside the switch are identified in the legend. The modular design approach utilised in developing the switching node is evident in the diagram as each module communicates with its neighbouring modules in a well defined manner. In addition, the whole system was implemented as a cooperative pre-emptive multi-threaded system. A detailed description regarding the functionality of each module in Figure 5-6, as well as the multi-threaded design approach followed, can be found in [12] and [33].

As AAL2 traffic enters the switch, the Interrogation Unit determines whether the received traffic is AAL2 traffic or normal ATM traffic. AAL2 traffic is then forwarded to the PDU Disassembly Unit where the AAL2 signalling information is separated from normal AAL2 user data. AAL2 signalling messages are then

forwarded to the control-plane Signalling Processing Unit (SPU) for processing. Once processed, the newly formed AAL2 signalling messages are sent to the PDU Assembly Unit to be packaged into AAL2 cells. Finally the new AAL2 cells containing AAL2 signalling messages are simply switched through the switch like normal ATM cells. The switching node was designed to assign three different priority levels to various traffic streams, depending on the traffic type supported. Hence, AAL2 signalling traffic always received the highest priority to aid in minimising the end-to-end signalling delays.

Figure 5-6 also illustrates the SPU module interfacing with the management-plane Admission Control Unit (ACU). The ACU module would need to support the SPU module by providing Connection Admission Control (CAC) to any new AAL2 connection requests sent from the SPU. The ACU would also be required to handle ATM signalling requests sent from the Interrogation Unit. However, due the independence between the AAL2 signalling and ATM signalling protocols, the SPU module is not able to request the establishment of a new ATM VC or AAL2 path from the ACU. Finally, the ACU would also be required to handle any routing queries sent from the SPU to locate destination AAL2 end-points in the network or to assign new CID values to new AAL2 channels. However, since the ACU module is not yet implemented all CAC and routing queries are currently handled by the AAL2 signalling framework in the SPC module. Future development on the switching node design will address this need. The only module that will be discussed in detail for the remainder of this Chapter is the SPU.

5.6.1 AAL2 Signalling Processing Unit (SPU)

The SPU module, as stated previously, is responsible for receiving incoming AAL2 signalling messages, processing these messages and generating the required response to be sent to the remote AAL2 nodes. Before the SPU is initialised, ATM VCs are statically provisioned by the ATM network operator to provide the required ATM connections between the current AAL2 node and all its neighbouring nodes. This ATM connection establishment procedure has been discussed previously in Section 5.4.3.

At system start-up, the SPU module executes an initialisation procedure which populates its AAL2 routing table by loading a uniquely preconfigured parser file. This is a similar procedure to that described in Section 5.5.1 for the AAL2 end-point. This routing table is however separate from the normal ATM routing table. The AAL2 routing table is only utilised to determine the routes to requested AAL2 nodes, whereas the ATM routing table is utilised to determine the outgoing route for ATM cells.

The SPU module is a direct implementation of the AAL2 signalling framework, as presented in this study, with the exception that an additional mapping service is provided to map the incoming AAL2 connections to their corresponding outgoing AAL2 connections. This is required since the AAL2 signalling protocol establishes an end-to-end AAL2 connection by means of multiple point-to-point connections spanning a number of nodes, depending on the AAL2 network topology. Each AAL2 node contains an AAL2 signalling entity responsible for performing the required signalling procedures. Therefore, each AAL2 switch is required to monitor its incoming and corresponding outgoing AAL2 connections since they are regarded as separate VCs. Consequently, the connection control block (CCB) database module in the signalling framework was extended for the switching node to contain information regarding both the incoming and outgoing values of the following entities for each AAL2 connection:

- OSAID generated for a request
- DSAID received for a request
- Interface ID where path is located
- Path and channel IDs
- AAL2 end-point address for the connection

Upon receiving an incoming establish request signalling message from a remote node, the request is initially processed as if the switching node was a normal end-point and therefore follows the procedure described in Section 4.6.2. Once successfully processed, a corresponding outgoing connection request procedure, as described in Section 4.6.1, is followed. This is necessary to determine the next point-to-point

connection required to eventually establish a transparent end-to-end AAL2 connection from the initiating end-point through the network to the eventual destination end-point.

Whenever an incoming release request message is received for an established connection, the request is processed and forwarded to the next destination end-point. Subsequently, a release confirmation message is also generated and returned to the node initiating the release request. Only once a release confirmation message is received from the forwarding node is the AAL2 connection terminated and its data structures released. This procedure was presented and illustrated in Section 3.5.

5.6.2 Modification to the Switching Node to Support AAL2 Signalling

In order to support the full functionality of the AAL2 signalling framework, various modules of the AAL2 switching node had to be modified. The first modules that required modification were the transmitting and receiving modules. This was needed since, as was the case with the AAL2 end-point, neither of these modules were capable of transporting the larger AAL2 signalling messages since they could only transport small CPS packets. As a result, the segmentation and reassemble (SAR) functionality needed to be included in both the transmitting and receiving modules, as defined in ITU-T recommendation I.366.1.

However, the initial implementation of the SAR functionality did not operate as expected as it resulted in the received AAL2 signalling information being corrupted. It was determined that this was a result of the SAR functionality not being able to distinguish between segmented AAL2 signalling messages received from different AAL2 connections. Consequently, received signalling segments from various connections were joined together in order to produce the *original* signalling message sent on a particular connection. Instead, the result was a signalling message consisting of various different segments and was therefore meaningless. Subsequently, the SAR functionality was modified to be channel specific so that only segments from a particular channel could be joined together to reproduce the original signalling message at the switching node.

5.7 Various AAL2 Signalling Evaluation Platform Configurations

This section discusses the various evaluation platform configurations utilised in this study to assess the functionality of the integrated AAL2 signalling framework in both the AAL2 end-point design and the AAL2 switching node architecture. The signalling framework was implemented as part of the SPU module in the switching node architecture.

5.7.1 Implementing AAL2 Signalling on the SPC

Figure 5-7 illustrates the physical devices, namely two PCs and a single WUGS switch, which serve as the test-bed implementation. Each PC serves as an AAL2 end-point and the WUGS switch is populated with a single Smart Port Card (SPC), on which the AAL2 switching node architecture was implemented. The SPC is an embedded computer situated inside the WUGS switch between the link interface and the ATM switching fabric allowing it to intercept incoming cells. The SPC thus functions as an active processing module and contains a software implementation of both the AAL2 switching and signalling components.

It is important to note that the SPC only operates at 166 MHz. Subsequently, its operation is slower in comparison to that of a commercial hardware implementation. The diagram also demonstrates the interconnection required between the two AAL2 end-points and the WUGS switch in order to send AAL2 signalling messages between the two end-points in a bidirectional manner. Only ports 4 and 5 are shown in the diagram, as they were the only ports utilised. For ease of explanation, the input and output ports of the WUGS switch are depicted on opposite sides of the switch. In the centre of the switch, the VPI/VCI Translation Table (VXT) is shown which is used to determine where ATM cells should be sent as they traverse the WUGS switch. Each entry in the VXT table consists of an input route and a corresponding output route. Each route value contains three numbers, where the first is the port number, the second the VPI and the third the VCI number.

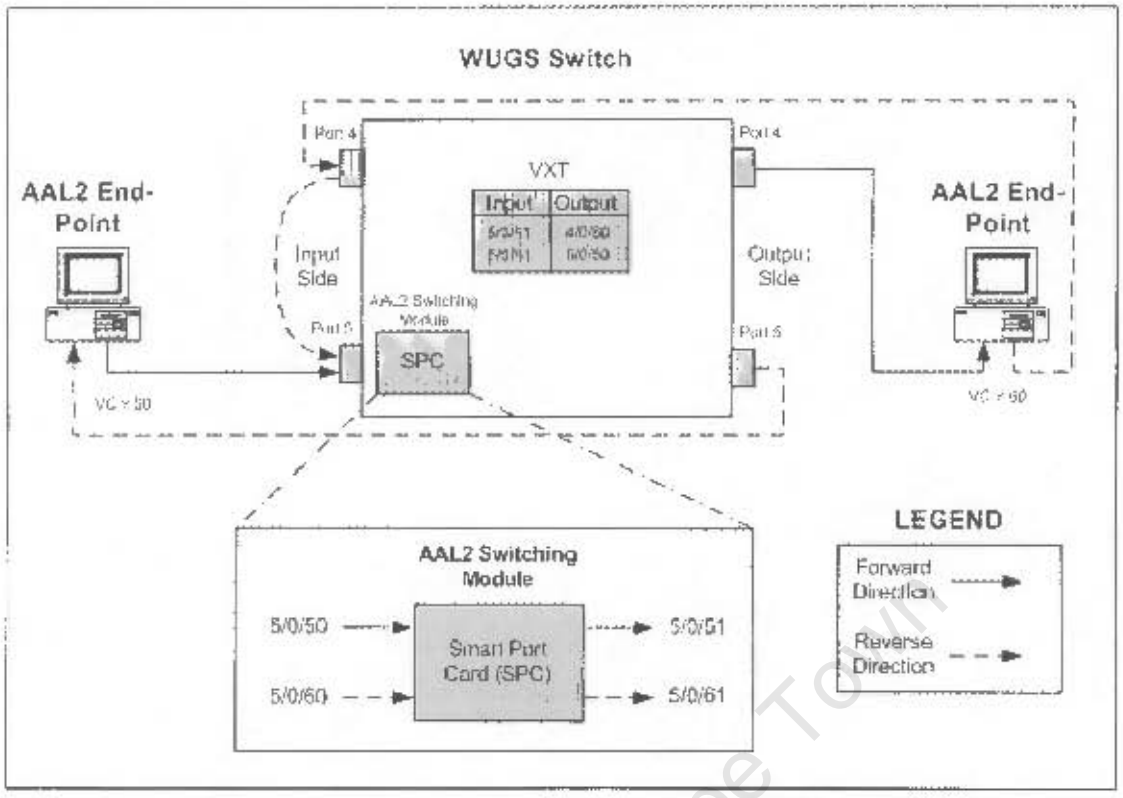

Figure 5-7 Implementing the AAL2 Switching Node on a Single SPC

In order for the WUGS switch to perform AAL2 switching based upon the VPI/VCI/CID values of individual AAL2 CPS packets, the CPS packets must be intercepted, as they enter the WUGS switch. The AAL2 switching module, implemented on the SPC card, performs this functionality. The primary problem is that the AAL2 switching module is a unidirectional module, because of its complexity and because it is physically located between the input port and the main switching fabric of the WUGS switch. In theory, each input port of the WUGS switch should be populated with a SPC card containing an AAL2 switching module. This would result in the WUGS switch being a fully integrated AAL2 switch with distributed processing on a per port basis. Unfortunately, due to hardware limitations this distributed approach was never implemented.

The AAL2 end-point, on the left-hand side of Figure 5-7, would initiate the establishment of a new AAL2 connection by sending an Establish Request (ERQ) signalling message to the second AAL2 end-point, on the right-hand side, through the switch, as illustrated with the solid lines. The second AAL2 end-point would then respond to the request by sending an Establish Confirm (ECF) message back to the

initiating AAL2 end-point through the WUGS switch, as illustrated with the dashed lines. Both these signalling messages must be sent through the unidirectional AAL2 switching module, as AAL2 signalling messages are transmitted on a hop-by-hop basis from one AAL2 node to another.

Therefore, in order to distinguish between the forward and backward directions of the signalling messages sent through the AAL2 switching module, two separate paths were utilised, as illustrated in Figure 5-7. In the forward direction, the ERQ message is sent to port 5 on VPI/VCI 0/50 and intercepted by the AAL2 switching module. After being processed, a new ERQ message is sent out of the AAL2 switching module on VC 0/51 to the WUGS' switching fabric. There the VXT table is consulted to determine the output path through the switch. In this example the ERQ message is sent to output port 4 on VC 0/60 which is connected to the receiving AAL2 end-point. The receiving AAL2 end-point then sends an ECF message as a reply to port 4 on VC 0/60, but because the ECF message must traverse the unidirectional AAL2 switching module located on port 5, the ECF message is recycled to port 5 on VC 0/60. After being processed by the AAL2 switching module, the ECF message is sent out on VC 0/61 and routed through the switch to port 5 on VC 0/50. Although this solution to the unidirectional AAL2 switching module might seem complicated, its benefit is that both AAL2 end-points view this configuration as being connected simply by means of a single bidirectional VC.

This implementation, however, could not be implemented due to hardware related problems in the SPC. The SPC card utilises the same APIC chip, described in Section 5.4.3, that is found in the network interface card located in each PC or AAL2 end-point. Although not indicated in Figure 5-7, the SPC was designed to have two input ports and to two output ports, enabling it to send and receive data originating from either the link interface side or the switch side. The SPC can therefore handle data in a bidirectional manner. In the example depicted above, data is sent from the end-point on the left into the SPC on port 5 with VC = 50. After being processed by both the AAL2 switching node and the remote end-point on the right-hand side of the diagram, the switching node again needs to process the replying signalling message and finally send that to the initiating end-point on the left-hand side of the diagram by means of port 5 again with VC = 50. The problem is that the SPC intercepted incoming cells on

port 5 with VC = 50, but then as data is attempting to exit the WUGS switch on port 5 with VC = 50, the SPC card would simply intercept the data again. Upon investigation it was discovered that the APIC driver was designed with this limitation. After correspondence with the developers at Washington University (WU) attempts were made to correct this shortcoming, but at the time of writing developers at WU were still in the process of modifying the APIC driver. Thus, due to time constrains, the full AAL2 switching node module was never successfully implemented on the SPC.

5.7.2 Implementing AAL2 Signalling on a Development Station

Since the AAL2 switching node is a software implementation, the author was able to port this software module to a normal PC to act as a stand-alone AAL2 switch. Figure 5-8 below illustrates a logical scenario in which two end-points are able to connect to one another by means of an ATM / AAL2 network.

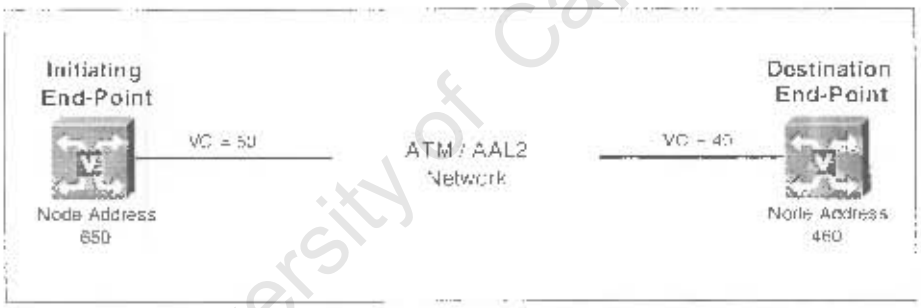

Figure 5-8 Logical End-to-End AAL2 Scenario

This logical scenario was practically implemented as illustrated in Figure 5-9. Two PCs were utilised to implement the three AAL2 nodes as depicted in the diagram. The switching node was implemented on a single PC, whereas both end-points were implemented on the second PC, but as two separate processes. One process was required for the initiating node, whereas a second process was required for the destination node. Each VC connecting an AAL2 end-point to the WUGS switch is illustrated as two separate unidirectional VCs so as to illustrate the required unidirectional VPI/VCI channel mappings needed in the WUGS switch. These

mappings are statically provisioned before system start-up by means of a scripting language developed by WU called Jammer[14].

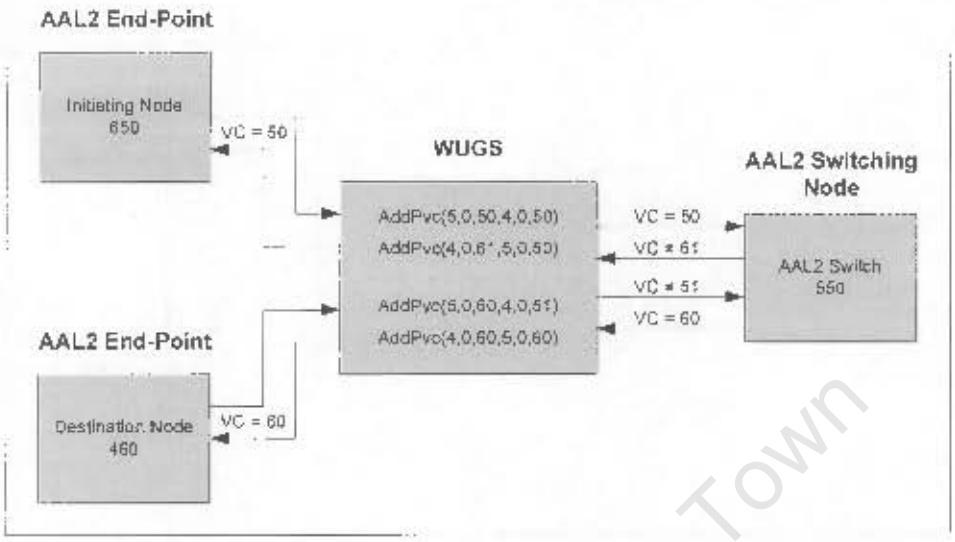

Figure 5-9 Practical End-to-End AAL2 Scenario

The interconnection between the WUGS and the AAL2 switching node on the second PC is illustrated as four unidirectional VCs, representing two bidirectional VCs, since the AAL2 switching node is a unidirectional module, as discussed previously.

As illustrated in the diagram, each node receives a unique node address at system start-up. Although the two AAL2 end-points are peer entities, they operate in an initiating-destination manner through the AAL2 switching node. The initiating end-point merely initiates the establishment of new AAL2 connection to the destination end-point, after which AAL2 user data can traverse the network in either direction.

Although the illustrations above only cater for a simple end-to-end scenario with two end-points, it could be extended to operate with three AAL2 end-points, as indicated in Figure 5-10.

[14] More information regarding Jammer can be found at http://www.arl.wustl.edu/gigabitkits

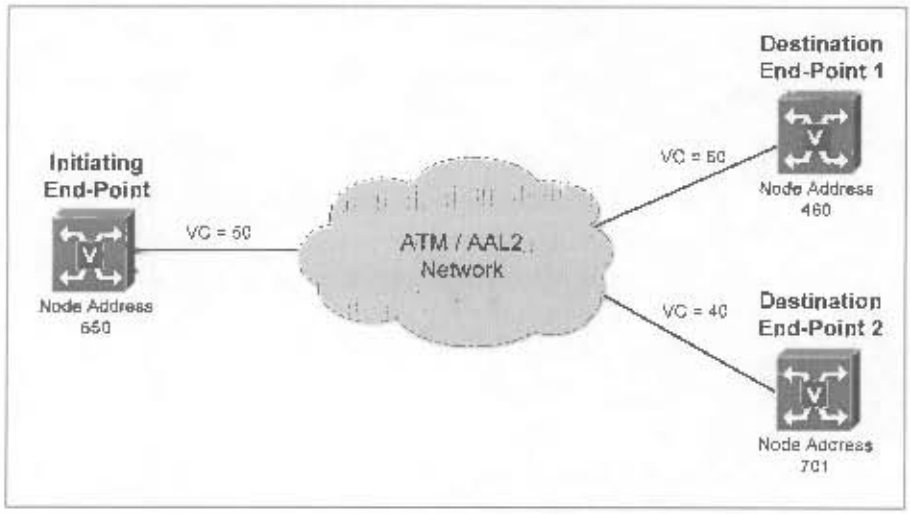

Figure 5-10 Logical End-to-End AAL2 Scenario Utilising Four Nodes

The practical implementation of the above logical scenario is illustrated below in Figure 5-11.

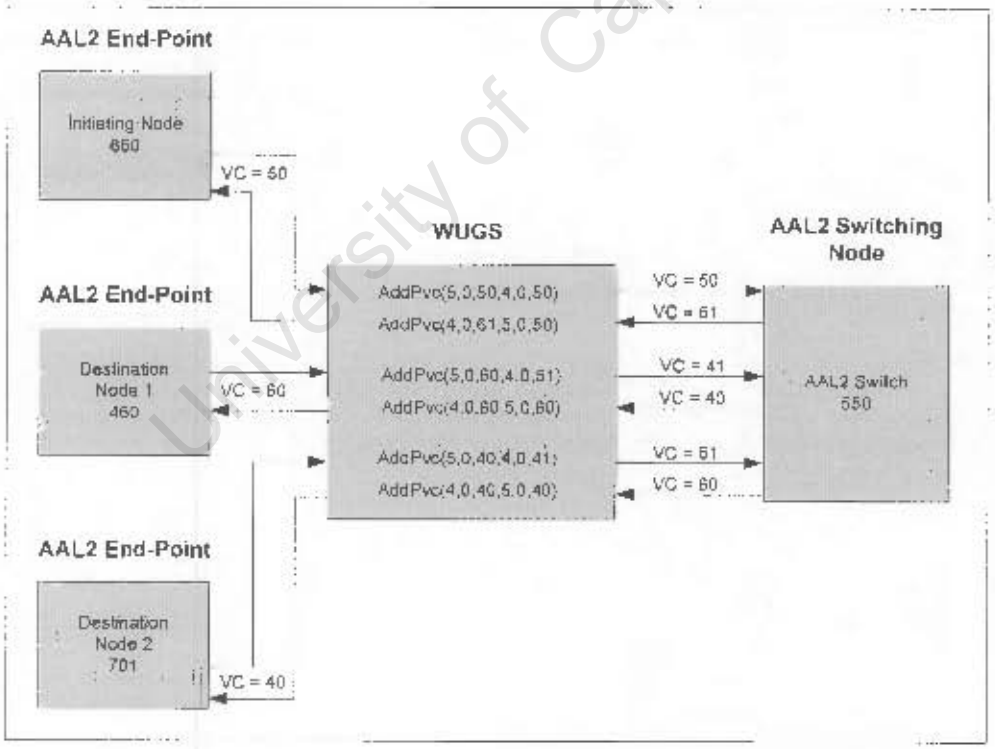

Figure 5-11 Practical End-to-End AAL2 Scenario Utilising Four Nodes

In this scenario, the AAL2 switching node is again implemented on a single PC, whereas all the AAL2 end-points are implemented on a single PC as three separate

processes. Again, the initiating-destination model is utilised merely to illustrate that the initiating end-point would initiate the creation of a new AAL2 channel to either destination end-point. The required VCs and channel mappings required in the WUGS switch are also included for completeness.

Chapter 6

Evaluation of the AAL2 Signalling Framework

The AAL2 signalling framework was designed to be modular in order to ease the development process and to isolate it from the layers above and below. This Chapter evaluates the functionality and efficiency, in terms of signalling delays, of the AAL2 signalling framework. Firstly, the framework is evaluated as a stand-alone entity to ensure functional correctness, secondly as being integrated into an AAL2 end-point and finally as being integrated into an AAL2 switching node. Even though many techniques could have been employed to enhance the performance of the signalling framework, due to time constraints this study could not address all of these issues.

6.1 AAL2 Signalling Framework

Due to the modular design approach utilised to develop the AAL2 signalling framework, the full functionality of each of the framework's modules was extensively and individually tested. Thereafter, as each module was systematically integrated into the framework, care was taken to ensure that firstly the module functioned correctly and secondly, that any problems that may have occurred during the integration process were identified. Once all the various modules were successfully integrated into the AAL2 signalling framework an abstraction layer, simulating the AAL2 transport entity in a limited capacity, was developed to test the full functionality of the entire framework, as shown in Figure 6-1.

The abstraction layer served two purposes, namely to sink signalling messages and secondly, to source signalling messages. As illustrated in Figure 6-1, an AAL2 user would form a request message and send it to the AAL2 signalling framework via the message-based interface. The signalling framework would then process the request and generate the corresponding AAL2 signalling message to be sent to the neighbouring AAL2 node en-route to the final destination node. This signalling message would then be sent to the abstraction layer, via the egress queue. Once received by the abstraction layer, the signalling message would firstly be studied to verify the integrity of the generated message. Secondly, each module of the signalling framework involved in generating the message would be studied to determine whether they were operating correctly. Also, while the request was being processed by the various entities, certain key variables were displayed on the PC screen in order to verify that each module handled the specific signalling request in the desired manner.

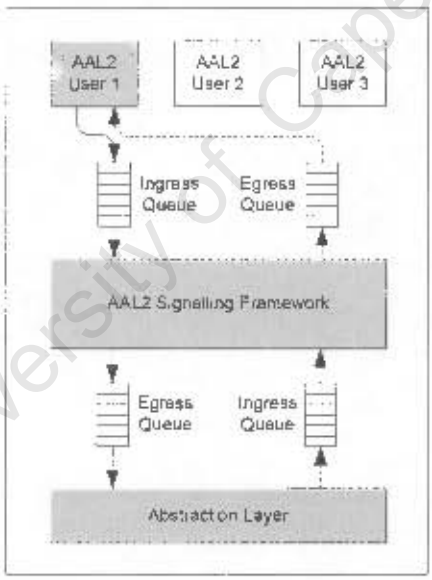

Figure 6-1 AAL2 Signalling Framework with Abstraction Layer

After all suitable request signals that could be generated by an AAL2 user were tested and verified, the abstraction layer would then generate certain signalling messages and send them to the AAL2 signalling framework. This was done to simulate the process of receiving a remote signalling message as oppose to initiating a message. Once again the above mentioned verification procedure was performed to ensure that the AAL2 signalling framework functioned correctly whether initiating a signalling message or receiving a signalling message.

6.2 AAL2 Signalling Integrated into an AAL2 End-Point

The framework was systematically integrated with the AAL2 transport entity once the AAL2 signalling framework was fully tested. The transport entity, which had previously been developed and discussed in [12], was extensively modified in this study to form part of the eventual AAL2 end-point design, as discussed in Section 5.3. Once the AAL2 end-point architecture was completed and a functional system in place, two end-point nodes were implemented on two separate PCs and interconnected via the WUGS switch as illustrated in Figure 6-2. This formed a platform with which the AAL2 signalling framework could be evaluated. Since the two end-points are interconnected via the WUGS switch, only a single mapping in the WUGS' switching table was required to transmit data transparently through the switch in a bidirectional manner between the two end-points.

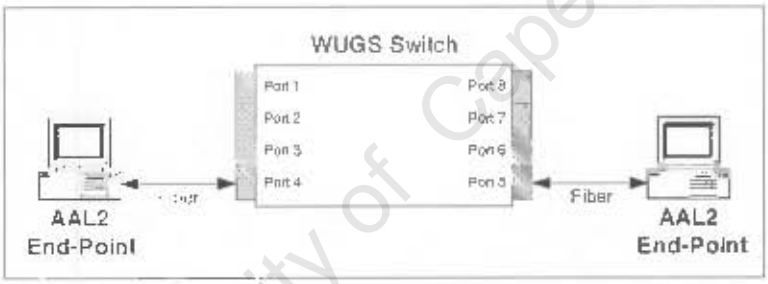

Figure 6-2 Interconnection between Two AAL2 End-Points

Besides switching the AAL2 traffic, the WUGS switch itself did not process the traversing traffic in any way. Therefore, the switch did not contribute to the signalling delays of the results obtained and presented in this Section. It should also be noted that the two PCs utilised in the platform evaluation, presented in this Chapter, are general purpose machines executing a number of different processes at any given moment. One of these processes was the evaluation conducted.

6.2.1 Establishing and Releasing a Single AAL2 Connection

Since general purpose machines were utilised in the evaluation platform, the results presented fluctuated slightly with each experiment. This fluctuation is evident in Figure 6-3 which illustrates ten separate trials. Each trial illustrates the signalling

delay or duration required to firstly, establish an end-to-end AAL2 connection and secondly, the signalling delay or duration to release that connection. Note, however, that this end-to-end connection merely consists of a single point-to-point connection, since only two AAL2 end-points were utilised. Section 5.4 illustrated and described the connection establishment and connection release procedures in detail.

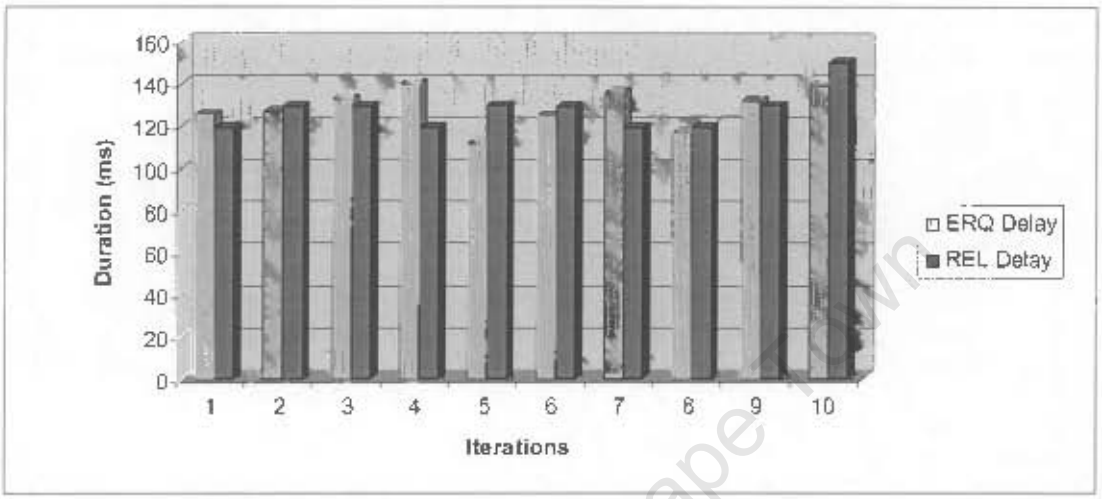

Figure 6-3 Fluctuation in System Processing for a Single AAL2 Connection

In the diagram the *ERQ Delay* is the period from when the connection request is formed by the initiating end-point, sent to the destination end-point to be processed and an eventual confirmation signalling message is received back at the initiating end-point. The *REL Delay* follows the same procedure as described above, except that the connection is now being released and all relevant data structures assigned to that connection are freed. The average AAL2 signalling delay thus required to establish a single AAL2 connection, based on the network topology of Figure 6-2, is 128.7 ms, whereas the average signalling delay required to tear-down the same connections is 127.7 ms.

6.2.2 Establishing and Releasing Four Simultaneous Connections

The AAL2 end-point architecture is based on a multi-threaded design, enabling it to support multiple AAL2 users simultaneously, as discussed in Section 5.5.3. The number of users at the initiating end-point was thus increased from one to four to

determine how the system would behave with an increase in AAL2 users, which translates to an increase in the number of simultaneous requests for new AAL2 connections. The four users were thus spawned at system start-up as individual sending user threads on the initiating end-point. Each sending thread would then simultaneously request an AAL2 connection to a corresponding receiving user thread situated on the second AAL2 end-point.

After each connection had been established, each sending user thread would then transmit a fixed number of AAL2 CPS packets, containing user traffic, to the corresponding receiving user thread via its newly created AAL2 connection. This process was simplified by allowing each sending user thread to send the same user data. Also, the file containing the AAL2 user data had already been read into memory during the initialisation phase of the AAL2 end-point system. Therefore, after each AAL2 connection had been successfully established, the same user data originating from a common file were transported across the various AAL2 connections. However, the CID value of each individual CPS packet was modified to reflect the specific CID value assigned to that particular AAL2 connection. Each receiving user thread was also assigned a matching CID value allowing it to only receive CPS packets with the correct CID value. Therefore, although the same AAL2 user data was transmitted to each receiving user thread, each receiving user thread only received user data destined for it. This technique served to simplify the process of transmitting user data to individual receiving user threads. Increasing the amount of AAL2 traffic handled by the evaluation platform provided a means of studying the signalling delays in a loaded system.

The signalling delays required to establish and release the four simultaneous AAL2 connections are illustrated in Figure 6-4. It can be seen that the period required to establish an AAL2 connection increases as the number of concurrent users grows. This is because the duration of each operation in the AAL2 signalling framework increases slightly as the framework becomes more utilised. An example of this is the signalling association identifier (SAID) module. For each new connection a unique originating SAID (OSAID) value must be generated and assigned. The purpose of this was explained in Section 4.5.3. As the number of active connections being processed by the signalling framework grows, so the time required to assign a unique OSAID

value to a new connection increases. This is because the newly generated OSAID value must first be compared to all the existing values in the database to check whether the new value is indeed unique. However, if the number has already been assigned to an existing connection, then the whole process would need to be repeated.

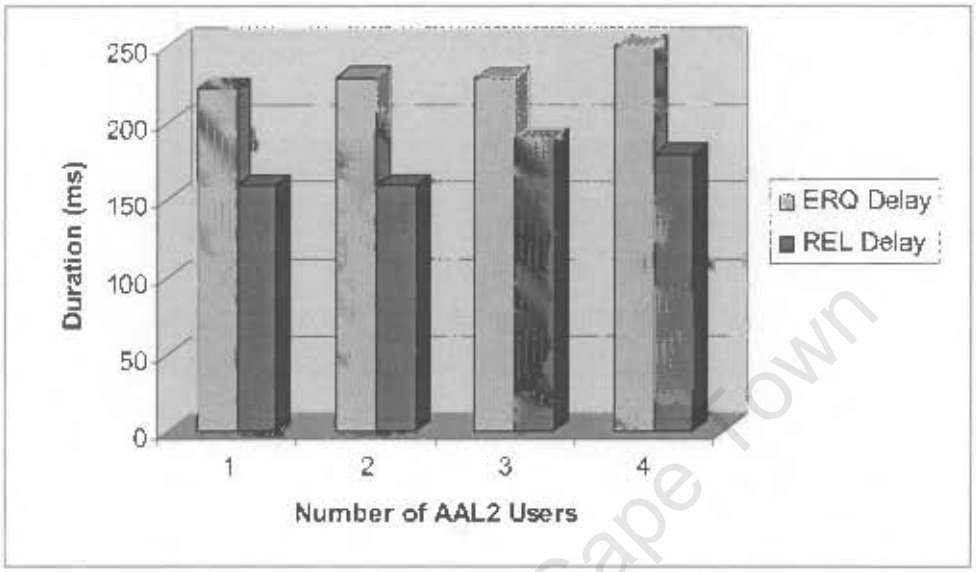

Figure 6-4 Creation and Release of AAL2 Connections with Four AAL2 Users

Since this is one of the many operations performed by the signalling framework on a newly created AAL2 connection, it is expected that both the average connection establishment signalling delay and the average connection release delay would increase as the system becomes more loaded. This is confirmed in practice since the average signalling delay required to establish an AAL2 connection, when four users are simultaneously spawned, is 232.9 ms, whereas the average signalling delay required to tear-down one of the four connections is 172.2 ms.

6.2.3 Establishing and Releasing Eight Simultaneous Connections

The maximum number of simultaneous AAL2 users, and thus connections, that can be successfully supported by the configured system as illustrated in Figure 6-2 is eight. Any number above this results in a system synchronisation failure between the two AAL2 end-points. This is due to a limitation of the AAL2 transport entity, which is an integrated component in the overall AAL2 end-point design. The transport entity

utilised a finite state machine to send and receive AAL2 user data. At system start-up the state of these finite state machines of both AAL2 end-point systems is at a known state. However, after a number of user threads are spawned on each end-point, signalling messages are sent and received in an unpredictable and bidirectional manner. It is therefore impossible to predict the exact behaviour of the multi-threaded system as the scheduler decides which thread is allowed to gain control of the system resources for execution. This limitation was not discovered during the evaluation stages of the AAL2 transport entity [12], since the system was only required to send user data in a predictable unidirectional manner. Therefore, although the AAL2 signalling framework was designed to process an initial maximum of 500 simultaneous connections, this could not be achieved.

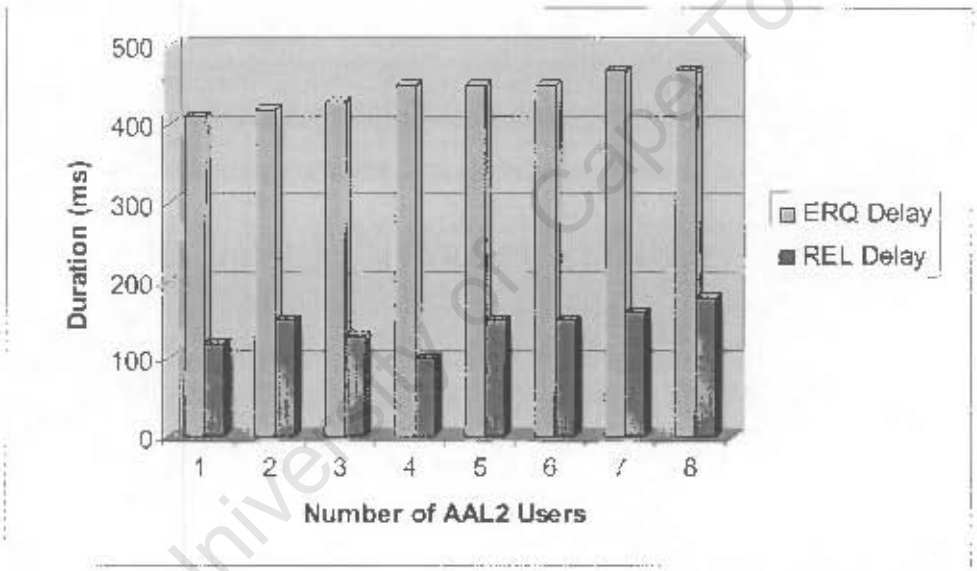

Figure 6-5 Creation and Release of AAL2 Connections with Eight AAL2 Users

Figure 6-5 illustrates signalling delays required to establish and release eight concurrent AAL2 connections for eight individual users. It can be seen that this figure follows the same trend as Figure 6-4, in that the period required to establish an AAL2 connection increases as the number of users supported by the system increases. The average signalling delay therefore required to establish an AAL2 connection, for eight simultaneously users, is 433.5 ms, whereas the average signalling delay required to tear-down one of the eight AAL2 connections is 142.1 ms.

It can be seen that as the number of concurrent users increases from one, to four, to eight, so the average establishment delay per AAL2 connection also increases linearly from 128.7 ms, to 232.9 ms, to 433.5 ms, respectively. It therefore appears as if the individual establishment delay per AAL2 connection exhibits a linear growth rate as the number of concurrent users increases. However, due to the relatively low number of users that could simultaneously be supported, this assumption could not be verified in this evaluation.

In contrast, the average time required to release an AAL2 connection remains relatively constant at 127.7 ms, 172.2 ms, 142.1 ms for one, four and eight concurrent users respectively. This is because at system start-up when a number of sending user threads is spawned, the only AAL2 traffic type present in the initiating end-point is AAL2 signalling. Each new user is therefore contending for common system resources, but as time progresses so the traffic type changes from only signalling messages to only AAL2 user data. As each sending user thread completes its cycle of sending user data, a new signalling message is created to release the AAL2 connection. Two types of AAL2 traffic then exist in the evaluation system, namely user data originating from other sending user threads and the release signalling information from the current user thread. Therefore, since the AAL2 end-point system was designed to assign a higher priority to signalling messages than normal user data, the delays required to release an AAL2 connection remained relatively constant irrespective of the number of users supported by the system.

6.3 AAL2 Signalling Integrated into an AAL2 Switching Node

After the AAL2 signalling framework was evaluated on a platform consisting of only two AAL2 nodes, this platform was expanded to incorporate a third node. In particular, an AAL2 switching node was placed between the two end-points. Therefore, each end-to-end AAL2 connection would now need to consist of two point-to-point channels. The first point-to-point channel would extend from the sending user threads' end-point to the switching node, whereas the second point-to-point channel would extend from the switching node to the receiving user threads' end-point. Figure 6-6 illustrates the newly configured evaluation platform.

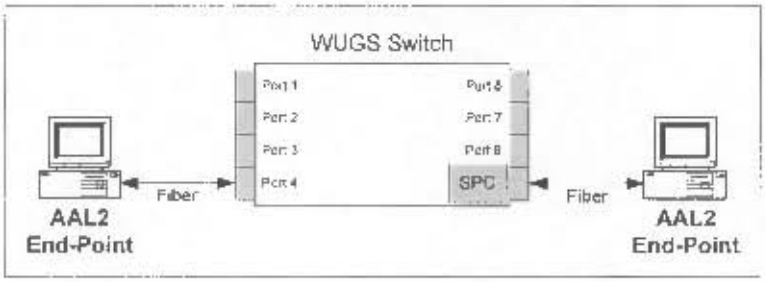

Figure 6-6 Interconnection between Three AAL2 Nodes

The SPC card located inside the WUGS switch would contain a software implementation of the AAL2 switching node enabling it to intercept incoming AAL2 traffic in the form of AAL2 cells. All signalling messages would be processed by the signalling processing unit (SPU) and all AAL2 user data would be processed by the relevant modules. Section 5.6 presented and discussed the AAL2 switching node architecture. However, due to a driver limitation of the APIC chip on the SPC, discussed in Section 5.7.1, this implementation was not possible.

Since the AAL2 switching node is a portable software module, it was however possible to implement the switching node design on a stand-alone development machine, as discussed in Section 5.7.2. The development station utilised for this purpose in this study was an 850 MHz PC. The configuration required to connect the two AAL2 end-points to the AAL2 switching node was presented in Section 5.7.2. Based on the new evaluation platform consisting of two AAL2 end-points and an AAL2 switching node, the following results were obtained.

6.3.1 Establishing and Releasing Four Simultaneous Connections

Similar to Section 6.2.2, four simultaneous AAL2 sending user threads were created on the initiating end-point. Each user thread then requested the establishment of a new AAL2 connection to the remote end-point via the AAL2 switching node. The results obtained from this evaluation are illustrated in Figure 6-7.

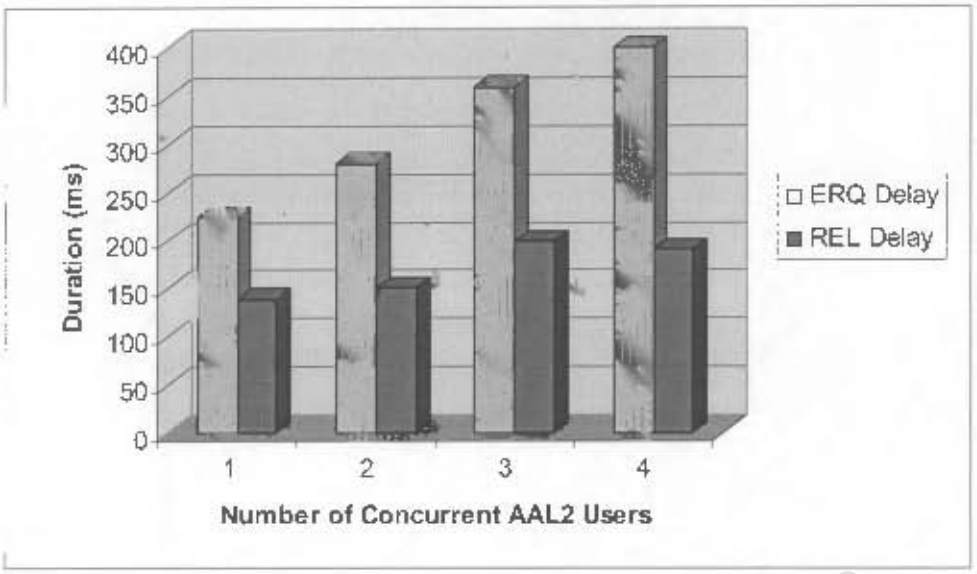

Figure 6-7 Establishing and Releasing Four AAL2 Connections

From the above figure it can be seen that the average signalling delay required to establish an end-to-end AAL2 connection, when four users are simultaneously created in a network utilising an AAL2 switching node, is 315.6 ms. The average signalling delay required to tear-down one of the four connections is 169.7 ms. These results were then compared to the results obtained in Section 6.2.2, which supported the same number of concurrent AAL2 connections without the utilisation of an AAL2 switching node. It was discovered that the average signalling delay required to establish a connection increased from 232.9 ms without the switching node, to 315.6 ms with the switching node. This increase in signalling delay was expected since the AAL2 switching node processed each traversing signalling message and hence added additional processing delay to the evaluation platform.

Further, it was also discovered that although the average establishment delay increased in the above comparison, the average signalling delay required to release an AAL2 connection remained virtually unchanged, irrespective of whether or not a switching node was utilised. This result was also expected since a local release confirmation message is returned from the neighbouring switching node to the initiating end-point, whereas the establish request confirmation message is returned from the remote end-point to the initiating end-point.

It can therefore be concluded that the AAL2 signalling delay required in establishing an end-to-end connection across an AAL2 network increases as the number of point-to-point connections increases. However, the connection release delay will remain unchanged irrespective of the number of point-to-point connections required to establish the end-to-end AAL2 connection.

6.3.2 Establishing and Releasing Eight Simultaneous Connections

The number of simultaneous AAL2 users, in the form of sending user threads, was increased at system start-up from four to eight. The result obtained for eight users are illustrated in Figure 6-8.

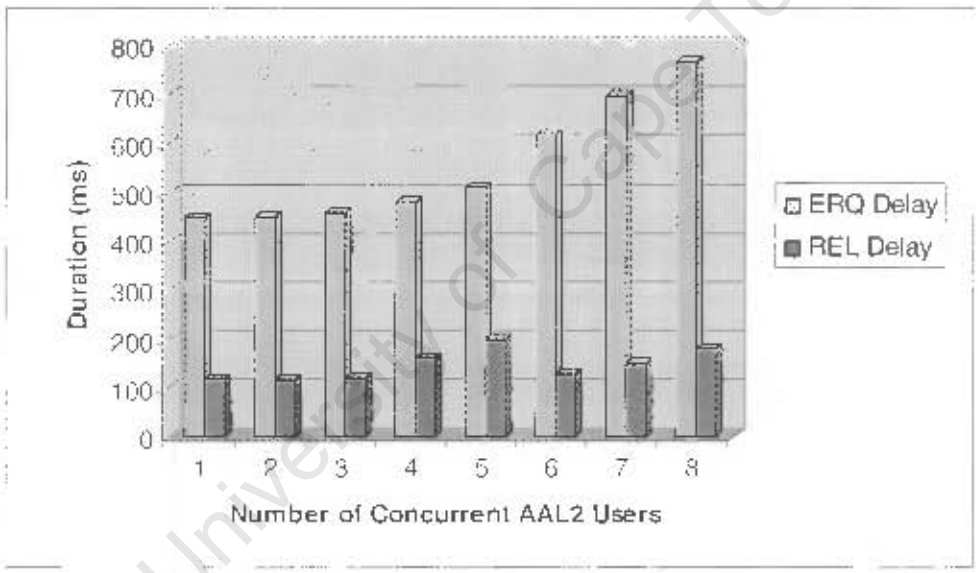

Figure 6-8 Establishing and Releasing Eight AAL2 Connections

From the figure it is clear that the individual signalling delay required to establish an AAL2 channel increases exponentially as the number of concurrent users in the initiating end-point increased. This confirms the question posed in Section 6.2.3 regarding exponential increase of AAL2 signalling delay with an increase in the number of users. Further, it is only the establish request signalling delay that increases exponentially. The signalling delay associated with tearing down each connection remains relatively constant. From the above trial, the average signalling delay

associated with establishing the eight AAL2 connections is 555.1 ms. The average delay required to tear-down the same connections is 146.8 ms.

6.3.3 Establishing and Releasing Ten Simultaneous Connections

The maximum number of simultaneous sending user threads that could be supported in the evaluation platform utilising two end-points and an AAL2 switching node was limited to ten. Any number above this limit resulted in a system synchronisation failure of either one of the AAL2 end-points or the switching node. This is due to a limitation of the finite state machine utilised in the transport entity of both the end-points and the switching node. This limitation was previously discussed in Section 6.2.3. The results obtained from ten concurrent AAL2 users are indicated in the figure below.

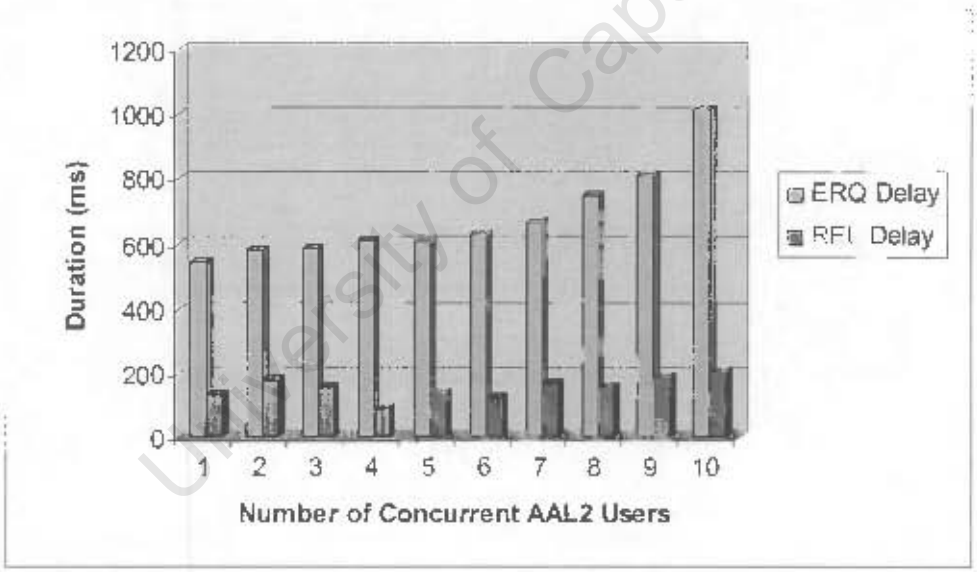

Figure 6-9 Establishing and Releasing Ten AAL2 Connections

It can be seen that the establish request delay duration for each AAL2 user increases exponentially as the number of users increased. This confirms the observation of the previous Section that only supported eight simultaneous users. The average signalling delay associated with establishing a single AAL2 connection when supporting ten concurrent connections is 681.1 ms, whereas the average delay required to tear-down the same connection is 155.2 ms.

6.4 Performance Comparison of the SPC

Although the link speed of each port on the WUGS switch operates at 1.2 Gbit/s, the bottleneck in the WUGS switch is definitely the SPC card operating at only 166 MHz. An attempt was made to implement the software-based AAL2 switching node module on the SPC card, but due to the SPC driver limitation, discussed in Section 5.7.1, the portable switching node software module did not function as desired.

It was however possible to send a single ERQ signalling message from the initiating AAL2 end-point to the AAL2 switching module, located on the SPC, to be processed. This enabled a comparison between the processing duration required by the switching node implemented on the SPC to the processing time required by the switching node implemented on the 850 MHz stand-alone PC. To simplify the comparison, only the duration required to process the signalling request by the AAL2 signalling framework in each implementation was compared. It was found that the signalling framework implemented on the 850 MHz stand-alone PC required 5.5 ms to process the signalling request, whereas the same signalling framework, implemented on the 166 MHz SPC, required 2148 ms to process the same signalling request. Since the SPC card required a much longer duration to process the same signalling request, it is clear that the SPC is the bottleneck in the system.

6.5 Performance of the AAL2 Signalling Framework

The duration required to successfully process a single establish request signalling message end-to-end, by only the AAL2 signalling framework located on each of the three AAL2 nodes, was evaluated. When the switching node was implemented on the 850 MHz PC, the establish request signalling process totalled 21.9 ms. Since the SPC required 2148 ms to process a signalling request that only required 5.5 ms by the 850 MHz general PC, it could therefore be extrapolated that the evaluation platform utilising the SPC for the switching node would require 3216 ms to successfully process a single establish request signalling message. This signalling duration would also increase as the number of concurrent AAL2 connections per SPC increases. Based on the above mentioned assumptions, it could therefore be concluded that the

SPC is not a suitable network element on which to implement an AAL2 switching node.

Washington University has subsequently developed a second active network device called the Field Programmable Port Extender (FPX) card. The FPX is designed to be placed inside the WUGS switch at the same location where the SPC card would normally be located. The FPX card contains a Field Programmable Gate Array (FPGA) chip enabling the FPX card to contain a hardware implementation of the AAL2 switching node. Therefore, the processing limitations of the SPC could be eliminated by replacing the software implementation of the switching node on the SPC with a hardware implementation of the switching node on the FPX card. However, due to time constraints this implementation was never attempted.

6.6 Feasibility of Deploying AAL2 Switching Nodes

Throughout this study, AAL2 switching has been discussed to be more bandwidth efficient than normal AAL2 trunking, especially in large complex networks. However, this study has also proven that AAL2 switching adds additional signalling delays to AAL2 signalling messages traversing the hybrid ATM / AAL2 network. This is because each AAL2 switching node needs to process each traversing signalling message. As the number of simultaneous users and hence the number of concurrent AAL2 connections in the system increases, so does the signalling delays required to establish and tear-down a single connection.

Care should therefore be taken when deciding where to place AAL2 switching nodes inside a network as too many switching nodes in an end-to-end connection may cause the signalling delays to become unacceptable. As an example the deployment of AAL2 switching nodes in a UMTS network can be considered. During the process of soft handover (SHO) the signalling delays associated with establishing a new connection need to be minimised so as to establish the new connection as fast as possible. If however, the new connection need to traverse too many AAL2 switching nodes the signalling delays might become unacceptable.

Chapter 7

Conclusions

Throughout this study, the benefits of AAL2 switching, in terms of bandwidth efficiency, as opposed to normal AAL2 trunking, have been presented and discussed. It has also been shown how AAL2 is becoming a popular technology in commercial applications such as VoDSL and UMTS wireless networks. However, these applications require a dynamic signalling protocol capable of establishing, maintaining and tearing-down the AAL2 connections as required by these applications. A functional AAL2 signalling protocol framework was therefore designed and developed. This signalling framework is capable of dynamically creating and tearing down on-demand end-to-end AAL2 connections between AAL2 end users.

In order to implement this AAL2 signalling framework, various functional modules were required. Firstly, the author implemented a *Request ID Manager* to facilitate inter-module communication within the signalling framework, as well as between the signalling framework and its neighbouring layers. This was required to provide a means of identifying a particular connection irrespective of the module processing that connection. In turn, the *SAID Manager* was developed to facilitate communication between adjacent AAL2 nodes. This enabled peer nodes to identify a particular AAL2 connection between them. The next module to be implemented was the *Routing Manager*. The two most important functions of the *Routing Manager* are firstly, to identify remote AAL2 node addresses that can be reached from the current node and secondly, to provide adequate connection resources along this particular route. In order to keep an accurate account of all the system resources utilised by a particular connection, as well as the status of that connection at any given moment,

the *Connection Control Block* (CCB) *Database* and *Resource Manager* was designed and completed. Other modules also completed and integrated into the signalling framework are the *Initialisation Module*, the *Encoder / Decoder Module*, the *Tracer Module* and the *Control Module*. The completion of all these modules into an integrated signalling framework provided a scalable system capable of dynamically establishing and releasing AAL2 connections. These connections are established by means of connection admission control based on QoS resource allocation. Although some of the functions performed by the *Routing Manager* are actually management related functions, such as connection admission control, these functions were incorporated into the Routing Manager due to the current lack of a management module.

Further, it was critical for future work that the framework developed be based on a modular design methodology ensuring that future research conducted at UCT could employ this framework with minimal modification effort. The author successfully achieved this goal by selecting a modular design approach in developing the signalling framework. This design methodology was further enhanced by use of a message-based interface, as opposed to a normal functional-based interface, between the signalling framework and its neighbouring layers. Four FIFO queues, implemented as circular FIFO buffers, were utilised for this purpose.

Having developed an AAL2 signalling framework, it was necessary to design an evaluation platform to verify the performance of the system. The author achieved this goal by employing two general purpose PCs connected to the WUGS switch. This allowed the system to be evaluated in two different phases. The first phase required the author to integrate the signalling framework into a functional AAL2 user end-point design. This multi-threaded end-point design was then ported to each PC to simulate an environment consisting of only two AAL2 nodes. In this evaluation phase the WUGS switch did not process the traversing AAL2 traffic in any way.

Further, the author was also required to implement the newly developed signalling framework on a previously developed AAL2 switching node architecture. This formed the second phase of the framework evaluation. This was accomplished by integrating the signalling framework into the signalling processing unit (SPU) of the

switching node. By achieving this, the author successfully enhanced the functionality of the AAL2 switching node by integrating a control-plane signalling component into the user-plane switching component of the existing architecture. The evaluation platform was thus expanded to incorporate three nodes, namely two end-points and a switching node. Although this switching node was originally designed to be implemented on a SPC located in the WUGS switch, this could not be accomplished due to a driver limitations. However, due to the portability of the switching node software, the author was able to implement this software module on a stand-alone development station.

After successful implementation of each evaluation phase, the AAL2 signalling framework was tested to determine the functional correctness within the developed system. The systems were also tested for scalability, in terms of the number of simultaneous connections that it is able to support, and efficiency, in terms of AAL2 signalling delays.

It was found that the maximum number of concurrent AAL2 connections that could be supported by the evaluation platform consisting of only two end-point nodes was eight. Any number higher than this resulted in the system losing synchronisation due to a limitation in the underlying transport entity. Although the AAL2 signalling framework was designed to support a higher number of concurrent connections, eight connections were sufficient in proving the functional correctness of the signalling framework. It also proved that the signalling framework with all its components did actually scale as desired. The author also discovered that as the number of concurrent connections being processed by the system increased, so did the signalling delay associated with establishing each AAL2 connection. As an example, four concurrent connections resulted in an average signalling delay of 232.9 ms. Subsequently, if this number was increased from four to eight concurrent connections, the average signalling delay increased to 433.5 ms. In contrast, the average signalling delay required to release an AAL2 connection remained relatively constant at 172.2 ms and 142.1 ms for four and eight concurrent users respectively. This is because at system start-up only signalling information is present in the system, but as time progresses the traffic type in the system changed from only signalling information to both signalling and user traffic. Therefore, since the end-point system assigned a higher

priority to signalling information than normal user data, the average signalling delay associated with releasing a connection remained almost constant irrespective of the number of simultaneous connections supported in the system.

For the second phase of the signalling framework the evaluation platform consisted of two developed AAL2 end-point nodes interconnected via the AAL2 switching node. It was determined that with this new network topology, a maximum number of ten concurrent connections could be supported. Any number above this resulted in the same synchronisation limitation as experienced in the previous evaluation phase. The results obtained again indicated that the average signalling delay required to establish an AAL2 connection increased as the number of concurrent connections increased. As an example this average signalling delay increased from 315.6 ms to 555.1 ms for four and eight connections respectively. It was also confirmed that this increase experienced an exponential growth in signalling delay. In contrast, the average connection release signalling delay remained relatively constant with an increase in the number of concurrent connections. This is because the connection release procedure utilised a local release signalling procedure to tear-down the AAL2 connection.

Although a management module is still lacking in both the end-point design and the switching node architecture, the signalling framework was able to dynamically establish and release AAL2 connections. It was proven that the signalling framework is applicable to both AAL2 end-points and AAL2 switching nodes.

Although the resulted delays presented in this study could be significant in real networks, many of the delay reduction techniques possible in such networks were not implemented on the evaluation platform. These techniques include the use of dedicated embedded systems to process certain functionalities that could be performed in a hardware implementation. Also, utilising a real-time operating system would enhance the efficiency of the signalling framework thus resulting in lower signalling delays.

Chapter 8

Recommendations and Future Projects

Although a functional AAL2 signalling framework was successfully designed and implemented in this study, a number of changes could be made to enhance the functionality of the system. Some of these enhancements are protocol related, whereas others are directly related to the signalling framework implementation. The following recommendations could therefore be made based on discussions in preceding Chapters as well as the conclusions drawn and presented in the previous Chapter. Several on these enhancements could in fact lead to possible future research projects, which are also presented in this Chapter.

8.1 Recommendations

The limitation imposed by the finite state machine of the AAL2 transport entity utilised in this study needs to be overcome. Currently, both evaluation platforms developed in this study are only able to support a limited number of concurrent AAL2 connections. Without the switching node this limit is eight, whereas utilising the switching node this limit increases to only ten connections. Additional work therefore needs to be performed to overcome this limitation imposed by the finite state machine of the AAL2 transport entity, as described in Section 6.2.3. Although this enhancement seems trivial to implement ITU-T recommendation I.363.2, which stipulates the transportation mechanism employed by the finite state machine, currently does not provided a mechanism to overcome this synchronisation limitation.

The FPX card should be utilised in conjunction with the SPC to improve system performance of the AAL2 switching node. Although the full functionality of the AAL2 switching node could not be implemented on the 166 MHz SPC, the processing limitation of the SPC was however proven by means of a simple experiment. It is therefore recommended that various components of the portable AAL2 switching node module be implemented on the hardware based FPX, since it promises to overcome the processing limitation of the SPC. Certain components, however, such as the routing manager, which contains the routing table of the switch, would still need to be implemented on the software based SPC since the routing table entries would need to be updated continuously, depending on the network topology and network utilisation.

Communication between the ATM and AAL2 signalling protocols need to be established. A current limitation of the AAL2 signalling protocol is that there is no means of communicating with the underlying ATM signalling protocol. Even though an ATM connection is required between peer AAL2 nodes prior to the establishment of any AAL2 channels, the AAL2 signalling protocol is currently unable to request the creation of this new ATM connection if one does not already exist. Even when the available resources of an existing ATM connection becomes limited, the AAL2 signalling protocol is unable to either request an additional ATM connection, or request an increase in bandwidth of the current ATM connection enabling it to support a larger number of AAL2 channels. Although ITU-T recommendation Q.2963 allows ATM connections to renegotiate their bandwidths while in the active state, this could not be implemented since no communication exists between these two signalling protocols. Future research could therefore address this need by providing some sort of mechanism to establish communication between these two currently independent signalling protocols.

A functional AAL2 routing protocol needs to be developed and integrated into the existing AAL2 signalling framework. A further limitation of the AAL2 signalling protocol is the lack of an associated routing protocol. Any AAL2 routing protocol, like the one developed for this study, is therefore implementation specific. The routing protocol's primary function would be to dynamically update the content of the routing table of each AAL2 node located in the network in accordance with the

current network topology. Due to the lack of a functional AAL2 routing protocol, current commercial implementations of AAL2 signalling are therefore limited to small domains. Future research should thus address this limitation.

8.2 Future Projects

The third and final phase of the AAL2 switching node project, as implemented on the WUGS switch, involves designing a management-plane module for the switching node. Past research has addressed the need for both a user-plane switching component and a control-plane signalling component. The management-plane module would thus form the required management component of the overall AAL2 switching node design. The management module could be responsible for dynamically creating and continuously updating the content of the AAL2 switching node's routing table in accordance with the AAL2 network. Connection admission control (CAC), which is currently performed by the control-plane signalling component would also be the responsibility of the management component. It could also be accountable for traffic management requirements and policing of the individualised AAL2 channels, a feature that currently does not exist. A centralized manager, such as the Call Agent illustrated in Figure 8-1, could also be designed and implemented to gather certain management related information, for billing and fault management purposes, from each node in the AAL2 network However, the management module of each AAL2 node in the network would need to have an interface platform enabling the Call Agent to interface to each node and thereby allowing it to exchange relevant information. This management module could also be integrated into the existing AAL2 end-point architecture to enhance its functionality.

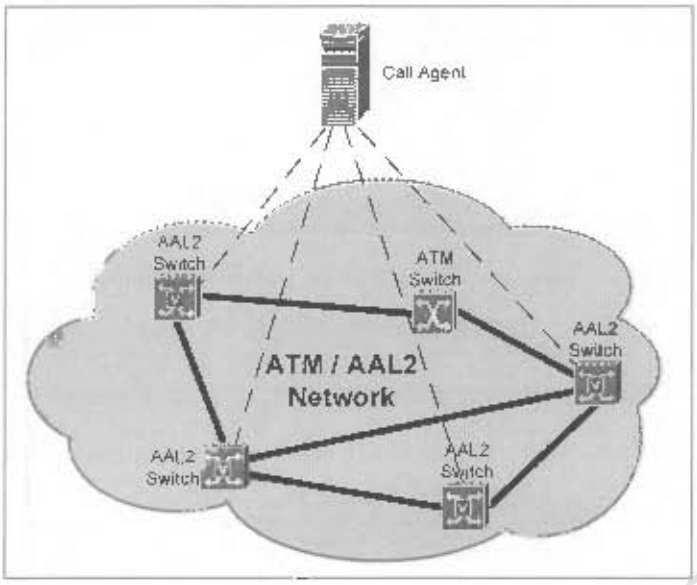

Figure 8-1 Implementing a Call Agent in the AAL2 Network

In order to extend the current functionality of the switching node design, support for basic narrowband AAL2 trunking services could also be added. A popular commercial application for AAL2 is narrowband carrier trunking as defined in ITU-T Recommendation I.366.2. This ITU-T recommendation defines a means by which an AAL2 trunk could be implemented to behave like a normal PSTN line with all the supporting functionality it requires. These features include support for:

- Voice (analogue / ISDN, optionally compressed voice)
- Voice-band data (modem, fax calls)
- Circuit mode data (ISDN data)
- Secondary messaging (Dialled Digits (DTMF), Fax Demodulation, etc.)

Adding this trunking functionality to the existing AAL2 switching node would provide the node with the unique capability of switching narrowband voice calls while being trunked to their respective destinations. This is a feature not supported by current commercial equipment.

A final commercial application for AAL2 and, in particular, voice over ATM is in the area of Voice over DSL (VoDSL), also referred to as ATM Loop Emulation Service (LES) using AAL2, as defined by the ATM Forum and illustrated in Figure 8-2. The

idea of VoDSL, as introduced in Section 2.5.1, is to extend the local copper loop from the customer premises through an ATM access network. Extending the AAL2 switching node to support AAL2 LES would enhance the node with the unique capability of being able to switch narrowband voice calls originating from VoDSL derived phones. This feature is also not currently supported by commercial ATM equipment.

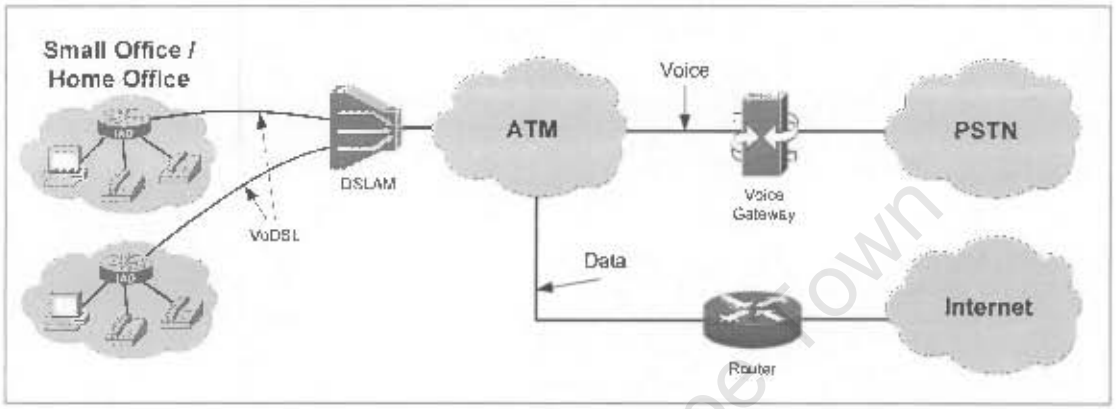

Figure 8-2 VoDSL Application Scenario

References

[1] Nortel Networks, "Packet Voice Convergence Using ATM Adaptation Layer – 2 (AAL-2) Protocol", White Paper

[2] Gil Biran, "Voice over Frame Relay, IP and ATM – The case for Cooperative Networking", http://www.protocols.com/papers/voe.htm

[3] Cisco, "Asynchronous Transfer Mode (ATM) Switching", White Paper http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/atm.htm

[4] Martin, Chapman and Leben, "Asynchronous Transfer Mode: ATM Architecture and Implementation", Prentice Hall PTR, New Jersey, 1997

[5] ITU-T, "B-ISDN ATM Adaptation layer specification: Type 1 AAL", ITU-T Recommendation I.363.1, August 1996

[6] ITU-T, "B-ISDN ATM Adaptation layer specification: Type 5 AAL", ITU-T Recommendation I.363.5, August 1996

[7] Mike McLoughlin and John O'Neil, "Adapting Voice for ATM Networks – An AAL2 Tutorial", General DataComm White Paper, 1997

[8] Hiroshi Saito, "Performance Evaluation of AAL2 switch Networks", IEICE Transaction Communication Magazine, Vol. E82-B No. 9 September 1999

[9] John H. Baldwin, Behram H. Bharucha, Bharat T. Doshi, Subrahmanyam Dravida and Sanjiv Nanda, "AAL2 – A New ATM Adaptation Layer for Small Packet Encapsulation and Multiplexing", Bell Labs Technical Journal, Spring 1997, pg. 114 - 131

[10] ITU-T, "B-ISDN ATM Adaptation layer specification: Type 2 AAL", ITU-T Recommendation I.363.2, September 1997

[11] Sven Shepstone, Neco Ventura and Andre van Zyl, "A Gigabit Switch for Low Variable Bit Rate Services", SATNAC Conference, September 2001

[12] Sven Shepstone, "AAL2 Switching Node to Support Voice Services in 3rd and 4th Generation Networks", MSc Thesis University of Cape Town, 2002

[13] ITU-T, "AAL Type 2 Signalling Protocol – Capability Set 1", ITU-T Recommendation Q.2630.1, December 1999

[14] Hughes Software Systems, "Voice over ATM – A stepping stone to converged broadband packet network", White Paper, May 2001

[15] ATM Forum, "Speaking Clearly with ATM – A practical guide to carrying voice over ATM", http://www.atmforum.com/atmforum/library/practical_voiceover.html

[16] CopperCom, "Mastering Voice over DSL: Network Architecture", A CopperCom Technology White Paper, 1999

[17] Cisco, "VoIP – Understanding Codecs: Complexity, Support, MOS and Negotiation", http://www.cisco.com/warp/public/788/voip/codec_complexity.html

[18] ITU-T, "AAL type 2 Service Specific Convergence Sublayer for Trunking", ITU-T Recommendation I.366.2, February 1999

[19] Khalid Ahmad, "Sourcebook of ATM and IP Internetworking", IEEE Press Series on Network Management

[20] ITU-T, "Segmentation and Reassembly Service Specific Convergence Sublayer for the AAL type 2", ITU-T Recommendation I.366.1, June 1998

[21] ATM Forum White Paper, "Speaking Clearly with ATM - A practical guide to carrying voice over ATM", URL: http://www.atmforum.com/pages/library/whitepapers/2.html

[22] NTT DoCoMo, "At the Core of 3G Mobile", IEEE Communications Magazine, Vol. 38 No. 12 December 2000

[23] Alcatel, "Voice over DSL Quality Study", Technical Paper, URL: http://www.cid.alcatel.com/doctypes/techpaper/html/VODSLqos.jhtml

[24] Uyless Black, "ATM Volume II – Signalling in Broadband Networks", Prentice Hall Series in Advanced Communications Technologies, New Jersey, 1998

[25] Goran Eneroth, Gabor Fodor, Gosta Leijonhufvud, Andras Racz, Istvan Szabo, "Applying ATM/AAL2 as a Switching Technology in Third-Generation Mobile Access Networks", IEEE Communications Magazine June 1999 pg 112 – 122

[26] Sumit Kasera, Pitambar Sahoo, Anil Sinha, Kuldeep Singh, "AAL2 Signalling: A White Paper", URL: http://skasera.tripod.com/papers.htm

[27] ITU-T, "Broadband Integrated Services Digital Network (B-ISDN) – Digital Subscriber Signalling System No. 2 (DSS 2) – User-Network Interface (UNI) Layer 3 Specification for Basic Call/Connection Control", ITU-T Recommendation Q.2931, February 1995

[28] Hiroshi Kawakami, Fumiaki Ishino and Hideaki Yumiba, "QoS Management of AAL2 in IMT-2000 Networks", IEICE Trans. Fundamentals, Vol. E84-A, No. 7, July 2001

[29] Hiroshi Saito, "Bandwidth Management for AAL2 Traffic", IEEE Transactions on Vehicular Technology, Vol. 49, No. 4, July 2000

[30] Hiroshi Saito, "Effectiveness of UBR VC Approach in AAL2 Networks and Its Application to IMT-2000", IEICE Trans. Communications, Vol. E83-B, No. 11, November 2000

[31] Shi Lu, Richard Thompson, "Improving VTOA AAL2 Signalling Transmitter Performance", Globecom 2001, San Antonio, Texas

[32] Istvan Szabo, Sandor Szekely, Istvan Moldovan, "Performance Optimisation of AAL2 Signalling for Supporting Soft Handoffs in UMTS Terrestrial Radio Access Networks", Proceedings of the Fifth IEEE Symposium on Computers & Communications 2000

[33] Sven Shepstone and Neco Ventura, "Active AAL2 Switching Node to Support 3rd and 4th Generation Voice Services in Fixed ATM Networks", SATNAC Conference, September 2002

Appendix A

Interfacing with the AAL2 Signalling Protocol

This Appendix consists of two sections. The first section provides an overview of the interfacing between the AAL2 signalling entity and its neighbouring layers. The second section features a complete set of all the available AAL2 signalling messages and their parameters.

A.1 Interfacing with the AAL2 Signalling Entity

This section provides an overview of the interfacing between the AAL2 signalling entity and the AAL2 Served User, the Signalling Transport Converter (STC) and the Layer Management, as depicted in ITU-T recommendation Q.2630.1. It also describes the interaction between the various AAL2 signalling primitives and the Service Access Point (SAPs), as illustrated in the Figure A-1.

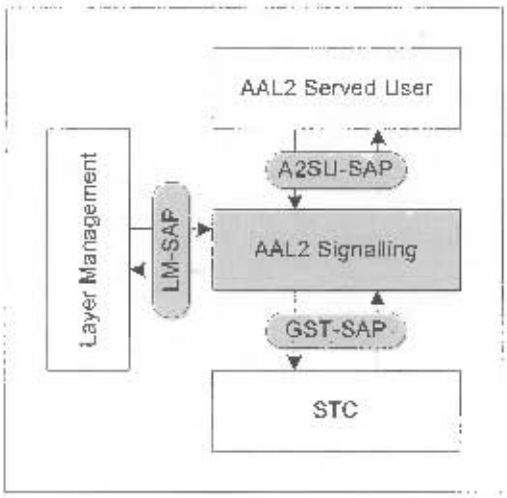

Figure A - 1 Interface between the AAL2 Signalling Entity and its Neighbouring Layers

Information is transferred between the AAL2 signalling entity and its neighbouring layers by means of primitives and parameters described below.

A.1.1 Interface between the AAL2 Signalling Entity and the AAL2 Served User

1. Service provided by the AAL2 signalling entity:

- Establish an AAL2 connection
- Release an AAL2 connection

2. A2SU-SAP primitives are used by the:

- Originating served user to initiate an AAL2 connection and to initiate a release request from either the originating or the destination user.
- AAL2 signalling entity to indicate an incoming connection request to the destination served user and to notify the served user of a connection release.

3. The primitives.type and parameters:

- ESTABLISH.request (A2EA, SUGR, SUT, TCI, SSCS, ALC)
- ESTABLISH.indication (SUGR, SUT, TCI, SSCS)
- ESTABLISH.confirm ( )

- RELEASE.request (cause)
- RELEASE.indication (cause)
- RELEASE.confirm (cause)

Parameter definition:

- A2EA – AAL2 service end-point address
- SUGR – Served user generated reference
- SUT – Served user transport
- TCI – Test connection indication
- SSCS – SSCS information
- ALC – AAL2 signalling Link characteristics
- Cause

A.1.2 Interface between the AAL2 Signalling Entity and the STC Layer

1. Service provided by the STC:

- Independence from the underlying signalling transport protocol

2. The GST-SAP primitives.type and parameters:

- IN-SERVICE.indication (Level)
- OUT-OF-SERVICE.indication ( )
- CONGESTION.indication (Level)
- TRANSFER.request (Sequence control, Message)
- TRANSFER.indication (Message)

Parameter definition:

- Message – Contains a complete signalling message
- Level – Value between 0 and 10 to indicate congestion level
- Sequence control – Allows the STC to perform load sharing between several signalling transports

A.1.3 Interface between the AAL2 Signalling Entity and the Layer Management

1. Service provided by the layer management:

- Provides the interface to the network management system

2. The LM-SAP primitives.type and parameters:

- BLOCK.request (ANI, A2P)
- BLOCK.confirm (cause)
- UNBLOCK.request (ANI, A2P)
- UNBLOCK.confirm (cause)
- RESET.request (ANI, CEID)
- RESET.indication (ANI, CEID)
- RESET.confirm ( )
- STOP-RESET.request (ANI, CEID)
- ADD-PATH.indication (ANI, A2P, ownership)
- REMOVE-PATH.indication (ANI, A2P)
- ERROR.indication (ANI, CEID, cause)

Parameter definition:

- A2P – AAL2 path identifier
- CEID – Connection element identifier. Identifies all AAL2 paths, a particular AAL2 path or a particular AAL2 channel on a certain path between adjacent AAL2 signalling nodes
- Ownership – indicates if the AAL2 signalling entity is the owner or its peer
- ANI – Adjacent AAL2 node identifier

A.2 AAL2 Signalling Messages and Parameters

Table A-1 below lists all the available AAL2 signalling messages and their acronyms. The minimum and maximum length of each signalling message is also presented, as indicated by ITU-T recommendation Q.2630.1.

Message	Acronym	Min Length (Bytes)	Max Length (Bytes)
Block Confirm	BCL	4	129
Block Request	BLO	13	13
Confusion	CFN	6	129
Establish Confirm	ECF	8	8
Establish Request	ERQ	12	321
Release Confirm	RLC	4	4
Release Request	REL	6	129
Reset Confirm	RSC	4	129
Reset Request	RES	13	13
Unblock Confirm	UBC	4	129
Unblock Request	UBL	13	13

Table A - 1 AAL2 Signalling Messages

Not all the parameters in Table A-2 need to be included in the AAL2 signalling messages. A detailed description of each AAL2 signalling message and signalling parameter can be found in ITU-T recommendation Q2630.1

AAL2 Signalling Message Parameter	Acronym
Cause	CAU
Connection Element Identifier	CEID
Destination E.164 Service Endpoint Address	ESEA
Destination NSAP Service Endpoint Address	NSEA
Destination Signalling Association Identifier	DSAID
AAL2 Link Characteristics	ALC
Originating Signalling Association Identifier	OSAID
Served User Generated Reference	SUGR
Served User Transport	SUT
Service-Specific Information (Audio)	SSIA
Service-Specific Information (Multi-rate)	SSIM
Service-Specific Information (SAR)	SSISA
Test Connection Indicator	TCI

Table A - 2 AAL2 Signalling Message Parameters

Appendix B

AAL2 Signalling Stack Architecture

This Appendix provides a brief overview of the different layers required to transport AAL2 signalling messages over either AAL2 or AAL5 in a User to Network Interface (UNI) network.

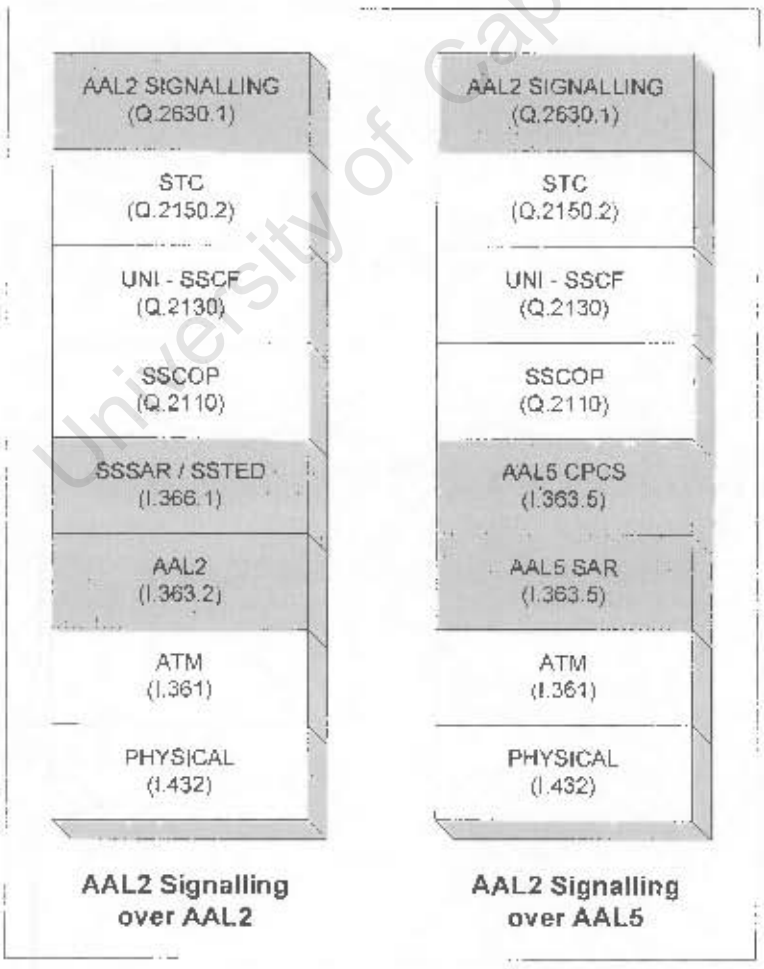

Figure B - 1 AAL2 Signalling Transported over AAL2 or AAL5

The choice of which AAL layer to utilise in order to transport the signalling messages, is discussed in Section 4.2. It was concluded that AAL2 be utilised for this purpose. Each of the layers of Figure B-1 will be explained briefly commencing with the AAL2 Signalling over AAL2 Stack.

B.1 AAL2 Signalling Transported over AAL2

Q.2630.1, AAL2 Signalling Protocol

This ITU-T recommendation specifies the procedures required to establish, maintain and tear-down AAL2 point-to-point connections. This AAL2 signalling protocol can be utilised in both the switched and non-switched environments and can operate in public or private networks. The protocol can also operate over multiple signalling transport protocol stacks.

Q.2150.2, AAL2 Signalling Transport Converter on SSCOP

This recommendation provides the specification required in the B-ISDN ATM environment for the provision of a signalling transport service. In particular, this protocol provides a generic signalling transport service that is utilised by the AAL2 signalling protocol. The AAL2 signalling transport converter on SSCOP utilises the Service Specific Connection Oriented Protocol for assured data transfer. This AAL2 signalling transport converter can be deployed on any protocol stack that supports SSCOP, such as AAL2 or AAL5.

Q.2130, B-ISDN Signalling ATM Adaptation Layer - Service Specific Coordination Function for support of signalling at the User-Network Interface (SSCF at UNI)

The ATM Adaptation Layer (AAL) is defined to enhance the service provided by the ATM layer to support the functions required by the next higher layer. Various AALs are defined to support AAL service users. One particular AAL service user is the

Signalling AAL (SAAL) defined in Q.2100. The SSCF maps the services provided by the SSCOP to the requirements of the user of the SAAL.

Q.2110, B-ISDN ATM Adaptation Layer - Service Specific Connection Oriented Protocol (SSCOP)

The SSCOP is a peer-to-peer protocol utilised above the AAL layer. It provides services such as error recovery, the transmission of user data with sequence integrity, flow control and connection control. This recommendation also describes the necessary elements for layer to layer communication.

I.366.1, Segmentation and Reassembly (SAR) Service Specific Convergence Sub-layer (SSCS) for the AAL type 2

This recommendation specifies the Segmentation and Reassembly Service Specific Convergence Sub-layer (SAR SSCS) of AAL2 which provides a means of transporting larger data packets across the AAL2 layer. The optional transmission error detection and assured data transfer are also defined.

I.363.2, B-ISDN ATM Adaptation Layer (AAL), type 2 Specification

This recommendation describes a method for bandwidth-efficient transmission of low-bit-rate, short, and variable length packets in delay sensitive applications. Each AAL2 connection can carry multiple AAL2 type information, such as compressed voice, originating from various sources.

I.361, B-ISDN ATM Layer Specification

This recommendation describes the ATM cell structure and the ATM protocol procedures. Two different cell structure formats are defined for the User-Network Interface (UNI) and the Network-Node Interface (NNI) respectively. It also defines the contents of the various fields of the ATM cell header for each format.

I.432, B-ISDN User-Network Interface - Physical Layer Specification

This recommendation covers the physical layer characteristics for transporting ATM cells at various bit rates of the B-ISDN UNI network. However, this recommendation was replaced with the new I.432.1, I.432.2, I.432.3, and I.432.4 specification with each specification defined for different bit rates and applications. These bit-rates range from 1.544 Mbps to 622.080 Mbps.

B.2 AAL2 Signalling Transported over AAL5

The only difference between the two protocol stacks illustrated in Figure B-1 is the AAL transport layer. This subsection will only discuss the AAL5 layer, since all the other layers are common to the AAL2 layer stack and have already been discussed.

I.363.5, B-ISDN Adaptation Layer (AAL), type 5 Specification

This recommendation provides a means for large data packets to be transported over an ATM connection. The combination of the CPCS and the SAR sub-layers are referred to as the Common Part of the AAL5 layer. This recommendation is applicable to both UNI and NNI networks.

Appendix C

Downloading a Kernel to the SPC

This Appendix presents an informal guide to the main steps required to build and download a NetBSD kernel to the Smart Port Card (SPC). It also demonstrates how to modify the filesystem for the SPC, so that it is possible to place additional software on the SPC.

Each input port of the WUGS switch could be populated with a SPC card, which would be located between the line card and the WUGS' switching elements. In order to perform the steps below, only one SPC card is required. The different software systems required in order to download a NetBSD kernel to the SPC will be mentioned, as well as the steps required in order to achieve a working system. This is not a comprehensive step-by-step 'fool-proof' instruction manual by any means but it intends to serve as guidelines to make the process less painful. For basic commands and new concepts, refer to information presented at the Washington University web site, READMEs, manual pages etc. Note that a great deal of this information is specific to the Communications Research Group (CRG) laboratory environment at the University of Cape Town (UCT). It is therefore recommended to adjust the below information accordingly, as required.

Figure C - 1 is a typical network topology that could be utilised. Two PCs, installed with NetBSD, are connected to the WUGS switch via optical connections. The port, on the WUGS, to which Inca is connected, is also the port in which the SPC card is fitted. An additional serial connection is also fitted between the serial port on Inca and the SPC card. Most of the configuration setup work in this Appendix will be performed on Inca.

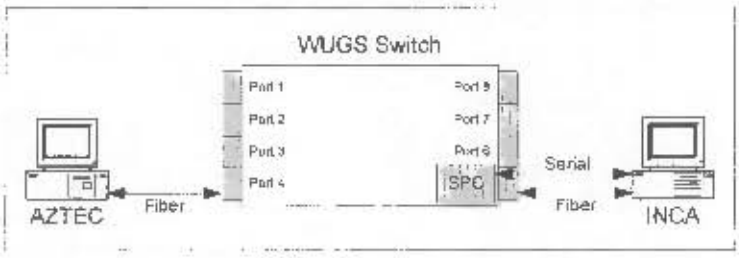

Figure C - 1 Physical Hardware Layout

The successful completion of this section involves the following steps:

1. Obtaining and installing the SPC source files
2. Initialise the supporting systems
3. Adding supplementary functionality to the kernel
4. Compiling the new SPC kernel
5. Downloading this new or modified kernel to the SPC

C.1 Obtaining and Installing the SPC Source Files

Aztec is the official CRG CVS mirror, which is updated every now and again. Locate the file '*wu-arl.tar.gz*' inside the directory '*/usr/home/crg-uct*' on Aztec. Ftp this file to your user directory, e.g. '*/home/andre*' on Inca and uncompress the file using the command '*tar –xvzf wu_arl.tar.gz*'. The source will now be inside the directory '*/home/andre/wu-arl*'. Since this directory name is not very descriptive, rename the directory to '*/home/andre/SPC1*'.

Before the new NetBSD kernel can be compiled, there are a few configuration details that need to be noted. The first ensures that a file can be mounted as a directory under '*/mnt*'. This step is already taken care of on Inca, but is required when NetBSD is installed on a new workstation that will interface to a SPC.

Go to the directory '*/etc*' and run the command '*vi disktab*' to edit the file with the *vi* editor. Make sure the '*spc24MB*' entry below is located in this file, if not then add it.

spc24MB|spc24MBkernSPC Kernel FileSystem Disk:

:ty=simulated:se#512:nt#16:rm#300:ns#36:nc#80:

:pa#46080:oa#0:ba#4096:fa#512:ta=4.2BSD:

:pb#46080:ob#0:

:pc#46080:oc#0:

The second configuration option allows a serial connection to be established between the SPC and a NetBSD workstation. In this Appendix example the workstation is Inca. In the directory '*/etc*' run the command '*vi remote*' and verify that the entry below is included.

spc0:dv=/dev/ty00:dc:br#9600:pa=none

Once these configuration details have been taken care of, go to the directory '*SPC1/utilities*' and run the command '*make install*'. This will create all the utilities needed to compile the kernel for the SPC, download data to the SPC and also reset the SPC.

C.2 Initialising the Supporting Systems

Before working on the SPC however, a few things need to be set up first:

1. Gigabit Network Switch Controller (GBNSC) must be running on one of the NETBSD machines (Inca or Aztec). This program controls the WUGS switch. In order to load GBNSC on Inca, go to the directory '*/home/crg-uct/gbnsc*' and run the command '*./setup_gbnsc_on_port_5*'.

2. The APIC card on Inca must also be set up to have ATM VCs connected to the SPC card. On Inca, go to the directory '*/home/crg-uct/spc_scripts/configuration*' and run the command '*sh config.apic.connections*'.

3. Finally Jammer must be started in the WUGS itself. On Inca go to the same directory as in the above step and run the command '*sh doit.sh*'. This will start Jammer with the configurations for the SPC card by means of the Jammer script '*SPC_Control.js*'.

To check whether GBNSC is running, run the command '*ps –aux*' on Inca and search for GBNSC in the process list that follows. To check that the APIC card is up and running, run the command '*ifconfig apic0*'.

C.3 Adding Supplementary Functionality to the Kernel

To change or add some functionality to the SPC kernel and download the SPC kernel to the SPC card, the following steps need to be performed:

Suppose we want to add the alias '*la*' for the command '*ls –la*' to the SPC kernel. Because the shell of the SPC kernel is *csh* and not *bash* like on Inca, all the shell configuration commands are placed in different files.

First mount the filesystem file '*SPC24MB.fs*'. To do this use the '*su*' command to become root. Go to the directory '*/mnt*' and create a new user directory where to mount the filesystem to, e.g. '*/mnt/andre*'. Go to the directory '*/home/andre/SPC1/spc_working_dirs/root_wdir*' and run the command '*vi ./mountit*'. Make sure that the device name to be mounted is right and that it will be mounted in the correct user directory. Now and run the command '*./mountit*'. This will mount the file '*SPC24MB.fs*' as device '*vnd0d*'. Run the command '*df*' to verify that the command was executed correctly.

Now go to mounted filesystem, located at '*/mnt/andre*' to perform any required changes. To add the alias '*la*', go to the directory above, become root and edit the file '*.cshrc*' with the command '*vi .cshrc*'. After this change has been performed, the filesystem needs to be un-mounted. Go back to the directory '*/home/andre/SPC1/spc_working_dirs/root_wdir*' and run the command '*./unmountit*'. Now exit as root to be a regular user.

C.4 Compile the New SPC Kernel

Now the new kernel is ready to be compiled. The next steps are as follows: First a symbols only version of the kernel is created, a filesystem is added to the kernel and then an image version of the new kernel is downloaded to the SPC card after a BSS segment was added.

To compile the new kernel, go to the directory *'/home/andre/SPC1/crossbow/NetBSD/usr/scr/sys/arch/i386/compile/SPC_24MB_with_Crossbow'* and run the command *'../mk.1'*.

Use the command *'su'* to become root, go to the directory *'/home/andre/SPC1/spc_working_dirs/root_wdir'* and run the command *'vi mk.2'* and make sure that the mounted directory and mounted device are specified correctly. Now run the command *'./mk.2'* and exit as root.

Go back to the previous directory *'/home/andre/SPC1/crossbow/NetBSD/usr/scr/sys/arch/i386/compile/SPC_24MB_with_Crossbow'* and run the command *'../mk.3'*. The kernel is now ready to be downloaded to the SPC. It is located in the directory *'/home/andre/SPC1/spc_working_dirs/download'* and is called *'netbsd.bss'*.

Note that *'mk.1'* and *'mk.2'* only needs to be run then the kernel needs to be recompiled. If one simply needs to add more functionality to the filesystem, then simply mount the filesystem add the new feature. When finished, unmount the filesystem and only run *'mk.3'*.

C.5 Download this New or Modified Kernel to the SPC

Open a new terminal on either Inca or Aztec, depending on which machine is directly connected to the SPC card (In this case it is Inca). Logon to Inca, on the second terminal, and run the command *'tip spc0'*.

Go back to the first terminal on Inca and go to the directory '*home/andre/SPC1/spc_working_dirs/download*' and run the command '*sh resetit.sh*' to reset the SPC. A program will start executing and after the program wrote the line '*wrote cell 2*' it should hang. Press <*ctrl-c*> and run the command '*sh doit.sh*'. The line '*Sending entire kernel……*' should then appear on the screen.

After 2 to 4 minutes, the second terminal on Inca will start booting. It will start in single user mode. So press <*ENTER*> and type the command '*exit*' to boot in multi-user mode. Now logon to the SPC as root by typing the command *'root'* and start playing around on the new kernel.

Appendix D

AAL2 Evaluation Platform Code Overview

This Appendix provides a brief overview of the C code develop and utilised during this study. The Appendix is divided into three sections to highlight the three different code segments utilised this study. Firstly, the AAL2 signalling framework code is presented. Secondly, this code is integrated with the modified AAL2 transport entity code to develop a multi-threaded AAL2 End-Point. Thirdly, the signalling framework is integrated into the modified AAL2 switching node code to complete the second phase of the overall three phase AAL2 switching node architecture.

D.1 AAL2 Signalling Framework Code

Each component contributing to the high-level AAL2 signalling framework design, as presented in Chapter 4, was developed as a separate module and is:

"global.h"
Main AAL2 Signalling header file. Contains structures for the AAL2 signalling messages that are sent and received between various nodes. It also contains the structure for the routing table of each node as well as the structure of the call control database.

"init.c"

AAL2 Signalling Initialisation file. This file is executed at system start-up to initialise the required modules used by the signalling framework.

"reqmgr.c"

Required ID Manager Module. This module assigns a unique ID to each connection being processed by the AAL2 signalling framework. This enables inter-module communication.

"msghand.c"

Message Handler file. This is the messages handler module that manipulates the received signalling message once received by the signalling framework. Thereafter the data is sent to the control module.

"a2s_ctrl.c"

AAL2 Signalling Control Module. This module is the heart of the AAL2 signalling framework and performs a scheduling function between the various modules of the signalling framework and the message based interface.

"ccbdbase.c"

Call Connection Control Database. This module is effectively a database containing information regarding each connection being processed by the AAL2 signalling framework.

"romgr.c"

Routing Manager Module. This module determines whether a particular node is reachable from the current node. It also determines whether the requested bandwidth can be supported by the existing ATM channel.

"rscmgr.c"

Resource Manager Module. This module calculates the available resources of a specific AAL2 path. This information includes the number of active connections and the path bandwidth utilised by these active connections.

"saidmgr.c"

Signalling Association Identifier (SAID) Manager Module. The module assigns a unique ID to each established AAL2 connection. This enables peer nodes to identify a specific AAL2 connection.

"encoder.c"

Encoder module. This module encodes each AAL2 signalling message before being sent out of the signalling framework to the below STC layer.

"show.c"

This file is essentially the Tracer Module that is responsible for identifying and outputting any errors that might arise during the operation of the signalling stack.

D.2 Integrated AAL2 End-Point System

The complete AAL2 end-point system is an integrated system comprising of the complete AAL2 signalling framework, the previously developed AAL2 transport entity modules as well as additional supporting software modules developed during this study. Section 5.3 presented the complete integrated end-point system. Only new modules developed for this study will be presented below.

"user.h"

This is the User header file. It contains information concerning the state of a user's connection as well as the CID channel value that was assigned to the specific user connection.

"su_cac.c"

Served User Connection Module. This module reads in the specified values sent from an AAL2 user, located in the served user layer, to facilitate in forming a signalling message destined for the AAL2 signalling framework.

"su_tx_msg.c"

Served User Transmit Message Module. This module supports the previous module to form a signalling request message that is sent to the AAL2 signalling framework via the message based interface.

"a2s_fifo.c"

AAL2 Signalling FIFO Buffers. This module forms the message based interface between the signalling entity and its neighbouring layers. It is implemented as a circular FIFO queue.

"a2s_interf_550_conf"

AAL2 Interface Configuration file. This file contains information regarding the current network topology. It specifies the interface IDs, path IDs and available link bandwidth between this node and its neighbouring nodes. The number 550 is the local node address assigned to the AAL2 switching node as illustrated in Figure 5-11.

"a2s_parser.c"

AAL2 Signalling Parser file. This file reads in the above configuration file and populates the routing table entries for a specific AAL2 node.

"interfaces.h"

Interfaces header file. This is the header file containing a structure that specifies all the necessary information required to link two neighbouring nodes.

"end_point.c"

AAL2 End-Point Node. This is the main function of the AAL2 end-point system. It performs the initial configuration of the end-point system, which depends on the specific mode of the end-point. It can be configured as initiating node or destination node. It also spawns a new user thread for each individual user requesting a new AAL2 connection.

"wrapper.c"

This file performs a wrapping function to transform an AAL2 signalling message into a compatible format to be used by the AAL2 transport entity. It also performs the

reverse function when data is sent from the transport layer to the signalling framework.

"i366-1.c"
This is the Segmentation and Reassembly (SAR) file. It contains a finite state machine enabling the AAL2 transport entity to transport large signalling messages as small CPS packets.

"i366-1.h"
This is the header file of the above .c file. It contains the structures required by the SAR state machine.

"l2s.c"
Larger to Small. This is the specific code required to implement the I.366.1 segmentation and reassembly function.

"aal2packet.in"
AAL2 Packet Input. This file contains all the AAL2 user data that will be sent to a remote node after the AAL2 required AAL2 channel was successfully established.

"a2s_file.c"
AAL2 Signalling File. This file is responsible for reading in the specified AAL2 user data to be sent from the initiating AAL2 end-point to the receiving AAL2 end-point.

D.3 Integrated AAL2 Switching Node System

The complete AAL2 switching node system was presented in Section 5.6. The AAL2 signalling framework was integrated into the SPU module which is the signalling module of the switching node. No new modules were developed to facilitate this functionality. Therefore, no code will be presented below. However, various signalling modules and switching node modules were modified to enable the correct functionality of the overall integrated switching system.

Appendix E

Supplementary CD-ROM

This text is accompanied with a CD-ROM that contains information relating to this study. This information is categorised as follows:

- **Source Code**

  All source code developed in this study can be found under the "Source Code" directory. This directory is divided into two sections, namely the AAL2 end-point code and the AAL2 switching node code.

- **Conference Proceedings**

  All SATNAC conference papers written and presented by both the author and Sven Shepstone on the design and development of the AAL2 switching node project can be found under the "Conference" directory.

- **Research Literature**

  Electronic copies of some research literature utilised in this study can be found under the "Research" folder. This literature ranges from certain standards developed by international standards organisations to whitepapers as well as presentations.

- **WUGS Information**

  All WUGS related information is located under the "WUGS" folder. This folder is sub-divided into WUGS specific source code and general WUGS specific literature.