Real-Time Feedback Control Mechanism on Packet Switching Network

by

Yam Kam Tai

Chief Supervisor Dr. C. K. Li

Industrial Supervisor Mr. H. L. Yiu

10 June 2003
Department of Electronic and Information Engineering
The Hong Kong Polytechnic University,
Hunghom, Kowloon,
Hong Kong

A THESIS SUBMITTED FOR THE DEGREE OF MASTER OF PHILOSOPHY TO THE HONG KONG POLYTECHNIC UNIVERSITY

Abstract

This project is a joint research between Motorola (HK) Semiconductor Product Sector Limited Company and the Hong Kong Polytechnic University. In this project, Internet Telephony is studied as an example of real time traffic communication in Internet.

In recent years, due to the popularity of Internet and many applications that are associated with it, more emphases are focused on the QoS of delivery to Internet-users. At the beginning of the thesis, background research on packet voice is presented. It includes:

(i) Several classic packet voice models.

(ii) Real Time Protocol (RTP) and Real Time Control Protocol (RTCP) protocols.

(iii) Definition of QoS and traffic management algorithm in Internet communications.

The real time communications issue in packet switching network is addressed. This study providing an overview on the current technology on what one should consider in real time packet communications.

Base on above studies, a new traffic control algorithm, namely the Real Time

Feedback Control Mechanism (RFCM) is developed for Internet real time traffic management. The mechanism provides real time monitoring and scheduling base on individual channel characteristic. The proposal RFCM improves channel utilization in both of real time and non-real time data distribution. The main advantage of the RFCM is its provision of dynamic adjustment on the bandwidth utilization for different types real time applications. Network Simulations (NS2)[24] is used to validate the supervisor properties of the proposed RFCM.

Besides on the theoretical analysis of traffic management, a new systematic and flexibility design methodology in central office communications equipment - Common Communication Platform (CCP) is introduced. It is a generic hardware and software platform on how to design network equipment. By using such model, it reduces the cycle time and reuses the previous knowledge in designing network equipment. Also, a high density DSP array board is designed in this platform. Such board contains 64 DSPs in Compact PCI 6U form factor. It uses data hub (designed by FPGA) to interconnect the board's processor and the DSP farm. By using this board, it can increase the number of voice channel to support in a single networking system. Hence, with this hardware, the proposed RFCM and other algorithm can be implemented for further investigation.

About the Motorola Semiconductors HK Ltd

Motorola Inc. is one of the world's leading providers of electronic equipment, systems, components and services produced for worldwide markets. Motorola products include two-way radios, telephones systems, semiconductors, defenses and aero-space electronics, automotive and industrial electronic equipments, data communications and information processing and handing equipments. The Corporate Headquarters is in Chicago, Illinois of the United States.

By specific product line, Motorola serves its customers as six groups - Semiconductor Products Sector, General Systems Sector, Land Mobile Products Sector, Messaging, Information and Media Sector, Government and Systems Technology Group and Automotive, Energy and Controls Group

The main role of Semiconductor Products Sector is to design, produce and distribute a broad line of discrete semiconductors and integrated circuits, including microprocessors, RF devices, microcomputers, memories and sensors.

Acknowledgments

The author wishes to convey his sincere thanks to his supervisor, Dr. C. K Li for his excellent supervision of his project.

Thanks also to Motorola for providing me the financial support to the Teaching Company Scheme for his project and to Mr. H. L Yiu, General Manager of Network and Communications System Group of Motorola for his technical advice and guidance. The author would also like to thanks all the team members (Jeffrey Ho, Kevin Lam, Alex Wong, Joshua Leung, Wilson Lo, Fat, Matthew Liong and Lewis Tse) in Networking and Computing System Department (NCSD) for great support on this research project.

Last but not the least, I would like to thanks to my family for their love and kindness. Without any of the above people this research project and my thesis will never come to existence.

Contents

List of Figures

List of Tables

List of Symbols and Abbreviations

ADPCM	Adaptive Differential Pulse Code Modulation
ALT	Adaptive Layered Transmission Scheme
API	Application Programming Interface
ATM	Asynchronies Transfer Mode
BSP	Board Support Package
CCP	Communications Common Platform
Compact PCI	Compact Peripheral Component Interface
CPM	Communications Processor Module
CPU	Center Processing Unit
CTMC	Continuous-Time Markov Chain Model
DAA	Direct Adjustment Algorithm
DBA	Dynamical Bandwidth Arbitrator
DMA	Direct Memory Access
DSP	Digital Signal Processor
DTMF	Dual Tone Multi Frequency
FCR	FPGA Control Register
FEC	Forward Error Correction
FPGA	Field Programmable Gate Array
FTP	File Transfer Protocol
G.711	Pulse Code Modulation (PCM) of Voice Frequencies [ITU Recommendations]
G.729	Coding of Speech at 8 kbit/s using Conjugate-structure Algebraic-code-excited Linear-prediction (CS-ACELP) [ITU Recommendations]
G.723.1	Dual Rate Speech Coder for Multimedia Communications Transmitting at 5.3 and 6.3 kbit/s
GMII	Gigabit Media Independent Interface
GPCM	General Purpose Chip-select Machine
H.110	Hypothetical Reference Connections for Videoconferencing Using Primary Digital Group Transmission
H.323	Packet-based Multimedia Communications Systems
HI08	8-bit Parallel Host Interface

IDMA	Independent Direct Memory Access
IP	Internet Protocol
LAN	Local Area Network
LPAS	Linear Prediction Analysis by Synthesis
LPC	Linear Predictive Coding
MAC	Media Access Control
MAN	Metropolitan Area Network
MIPS	Million-Instruction Per Second
MMPP	Markov Modulated Poisson Process
MMU	Memory Management Unit
NS	Network Simulator
OSI	Open System Integration
Otcl	Object-Oriented Tool Command Language
PABX	Public Automatic Branch Exchange
PCM	Pulse Code Modulation
PCI	Peripheral Component Interconnect
PHY board	Physical board
POST	Power On Self Test
PSOS	Real Time Operating System for Embedded Microprocessors
RFCM	Real Time Feedback Control Mechanism
RTOS	Real Time Operation System
PSTN	Public Switch Telephone Network
QoS	Quality of Service
RLM	Receiver-Driven Layered Multicast
RMII	Reduced Media Independent Interface
RTCP	Real Time Control Protocol
RTOS	Real Time Operating System
RTP	Real Time Protocol
SAD	Speed Activity Detector
SDRAM	Synchronous Dynamic Random Access Memory
SIU	System Interface Unit
SLIC	Subscriber Line Integrated Circuit
SMP	Semi-Markov Process Model
SNMP	Simple Network Management Protocol
SSRC	Synchronization Source
TDM	Time-Division Multiplex
TFTP	Trivial File Transport Protocol
UAS	Uniform Arrival and Service Model
UISA	User Instruction Set Architecture
UTOPIA	Universal Test and Operations PHY Interface for ATM
VAD	Voice Activity Detector
VCI	Virtual Channel Index
VoIP	Voice over IP
VPI	Virtual Path Index
VxWork	Real Time Operating System for Embedded Microprocessors
WAN	Wide Area Network

CHAPTER 1

Introduction

IP Telephony is one of multimedia communications applications that allow several users to share the same physical transmission media using packet-switching communications method. Nowadays, the technology of IP Telephony is growing faster and mutual. Most of its applications are used for long-distance call. The reason is that it offers lower price to end-users as well as less maintenance cost. Within the framework, it also provides integration of voice and data multimedia communications. However, the lack of QoS control in today's Internet restricted the applications of any serious real time multimedia communications. Data transmission over Internet only operates on a "best effort" packet delivery basis, with no prioritization on which packets get delivered first or which packets are less likely to be lost. Due to congestion at routers within a network, unpredictable delay and unrecoverable packet loss in real time network is not acceptable but today's Internet technology cannot provide a solution with sufficient QoS control.

This issue affects the growth of IP Telephony and must be faced in future multimedia Internet development.

Traditionally, voice communications uses circuit-switching method such as Public Switching Telephone Network (PSTN). PSTN does not provide any method for voice compression and silence detection. Each dedicated call must reserve 64kbps bandwidth between two end-users. Although there is no traffic during silence period, the bandwidth still occupied. Until the call is finished, the line is free up for other users.

The most significant difference between Internet Telephony, also known as Voice over IP (VoIP), and traditional PSTN is bandwidth utilization for each connection. Internet Telephony is packet-based and does not reserve any bandwidth for two-end users. It means that the channel may not reserve sufficient bandwidth for voice transmission when comparing with circuit-switching network. Also, additional processes have to be done in Internet Telephony before transmitting out voice signal such as voice coding, compression and packetization. On the other side, there are issues that affect any real time transmission such as packet collusion, packet loss and packet delay. Those issues are the main factors that seriously affect the corresponding QoS.

With the reference to the original objective of PSTN and Internet, they are totally different. PSTN is designed for voice (real time) communications as Internet (packet switching network) is used for data (non-real time) communications. They have different protocols to support their nature of data transmission such as packet encoding; network signaling, packet switching and terminal addressing method. The challenge is to integrate and handle real time data on Internet environment to provide multimedia service.

Traditional problems, such as packet delay and packet loss, should be addressed in order to guarantee the performance of Internet telephony application. New protocol design is also needed to handle voice packet in term of call setup signaling and timing measurement. Due to above reasons, it has to re-engine and re-design the Internet technology to enhance its functionality and reliability for multimedia service in Internet.

1.1 Currently Algorithm

Network traffic control is one of main task to provide QoS for different applications. In packet-switching network, it needs to provide different types of traffic control mechanisms so as to delivery different level of QoS according to different situation. Currently, research effort indicates that there are two systematic and realistic approaches for traffic control. They are,

1. Direct Adjustment Algorithm (DAA)
2. Adaptive Layered Transmission scheme (ALT)

1.1.1 Direct Adjustment Algorithm (DAA)

The Direct Adjustment Algorithm is based on both additive increase/multiplicative decrease approach under the same round trip time, packet size and loss ratio. It uses Real Time Control Protocol (RTCP) control message content to reflect the channel loss rate and round trip delay.

When the sender receives the control message, it calculates the smooth loss ratio L_S. If L_s is greater than 16%, then the sending rate is reduced by half. If L_s is smaller than 16%, then the sender sets its transmission rate R_A which depending on the Additive Increase Factor (AIF).

1.1.2 Adaptive Layered Transmission Scheme (ALT)

With the Adaptive Layered Transmission Scheme approach, the sender dynamically redistributes data stream on different layers based on the feedback of the receivers. In addition, ALT allows for dynamic discovery of the appropriate transmission rate for each layer. Receivers notify the sender of their view of the network congestion state using the Real Time Control Protocol (RTCP). The sender reacts to the reported loss rates by shifting bandwidth between the layers and keeping the overall sending rate constant. The scheme tries to provide all connected receivers on all layers with the best possible quality.

1.2 Proposed Algorithm

In this research, a new mechanism is proposed for traffic control using in real time applications, called Real Time Feedback Control Mechanism (RFCM). This algorithm can improve loss ratio and jitter on the network. It reserves dedicated bandwidth for different applications with different QoS requirements according to current network status. The key features of this mechanism are shown as follow:

- Provide real time feedback control on bandwidth reservation and management
- Provide level of service according to different QoS requirement
- Predict the traffic on dedicated channel

1.3 Scope of Thesis

Following the introduction chapter, the thesis is organized with the remaining seven chapters as follows.

In chapter 2, a general overview of current Internet Telephony technology is presented. It includes critical design issues of Internet Telephony such as packet delay, packet jitter, packet loss, echo in transmission line and voice coder. A brief description of Real Time Protocol (RTP) and Real Time Control Protocol (RTCP), which is used for real time packet traffic, is detailed. The characteristic of packet voice will be examined using different packet voice models.

In chapter 3, QoS requirement in Internet Telephony is introduced. It defines the amount of bandwidth that is desired for voice channels as well as what QoS requirements in packet voice communications over Internet. On the other side, different types of architecture and network solution is outlined showing how to provide a realistic solution for real time packet transmission and management.

In chapter 4, the detail of two systematic approaches - Direct Adjustment Algorithm (DAA) and Adaptive Layered Transmission Scheme (ALT) is discussed in detail. As the proposed Feedback Control Mechanism (FCM) is quite similar to the DAA. This provides the basic idea on how to control the flow of packet in such schemes.

A new traffic control mechanism for real time multimedia applications is proposed in chapter 5. This jitter and loss ratio reduction algorithm composes of three models. They are:

(i) Service model

(ii) Real time control model

(iii) Channel model

The objective of the algorithm is to adjust the transmission rate on different channels with according to the specific QoS requirements. A simulation tool, Network Simulators (NS), is used to analyze the Real Time Feedback Control Mechanism (RFCM). The simulation focuses on the bandwidth utilization, packet jitter and loss ratio on real time traffic as well as non-real time traffic. The result is analysis in this chapter.

Implementation of real time control scheme in Communications Common Platform (CCP) is discussed in chapter 6. Integration of hardware components including communications processor, Digital Signal Processor (DSP) and line card is presented. The performance of any real time applications will be affected by the designed network equipments as well as the control software implemented on the network equipments. The influence of performance of the Feedback Control Mmechanism (FCM) is mainly due to the packet processing delay, which in term affected by factors such as signaling, channel setup and DSP encode/decodes speed, etc.

In chapter 7, high density of Digital Signaling Processor (DSP) design under Communications Common Platform (CCP) is presented. There are total 64 DSPs built in Compact PCI 6U form factor with 233.5x160mm through two data hub (design using Field Programmable Gate Array (FPGA)) control. The main task of data hub is to handle Direct Memory Access (DMA) transfer in between the network processor and the DSPs.

Finally, the last chapter concludes the thesis and discusses further work.

CHAPTER 2

QoS Control Through Traffic Management

Nowadays, Internet Telephony becomes popular in long distance call because of its lower cost comparing with the traditional Public Switching Telephone Network (PSTN). There are two approaches to analyze voice quality of Internet Telephony. They are:

1. Quality of Service (QoS) Traffic control
2. Performance requirement in system design consideration.

The first approach in Quality of Service (QoS) traffic control needs to use of theoretical analysis in order to obtain the characteristic of packetized network. The task of theoretical analysis is to model voice packet behaviors. Different types of voice packet model have its constraints and limitations such as Uniform Arrival and Service Model (UAS) which ignores high frequency variations in buffer content as compared to the real system [1] and M/D/1 model assumes to provide large number of voice source. In previous research [2], most of packet voice model analysis use MMPP as a common and simple model to study packet voice characteristic. On the other hand, network characteristic is another main factor that affect QoS provided by Internet such as packet loss, delay and jitter. These issues cannot be avoided because it is the nature phenomenon in IP network but needed to guard and control through traffic management. For instance, some real time applications have to suffer a number of packet losses in order to satisfy packet delay or jitter constraint. For these reasons, packet voice model and network characteristic have to be studied and then an appropriated scheme for traffic management is applied.

The second approach in real time applications is to analysis network performance requirement in system design consideration. Networking equipments can be separated into three segments, namely the core segment, the edge segment and the access segment. To implement real time service over packet network, specific system architecture design has to be considered within these three types of network segment. The performance of such network equipments affect QoS that the network can be provided such as encoding, signaling and addressing data processing. Unlike traditional PSTN network, VoIP system uses packet switching network to transmit voice signal. Such network can provide other service on top of voice service in between PSTN and Internet. It includes how to receive, process and transfer voice packet to circuit switching network and reverse.

In this chapter and the following chapter, basics introduction on packet voice network is outlined and problems on some outstanding issues are identified.

One major advantage of using packet-switching network for voice transmission is that instead of having a group of engineer and manager to operate a data network and another group to manage a single packet backbone network, a single network management software system can be employed. Furthermore, additional features can be added such as instant messaging, web integration, collaboration, video telephony and fax messaging. However, on the other side, the change of voice transmission from circuit switching to packet switching introduces several issues such as signaling, voice compression and how to interconnect traditional PSTN network to IP network together.

The most important issue in Internet Telephony that must be resolved is the real time consideration. In traditional Internet design, no real time concept is considered but today many applications require a certain real time specification. Real time data means that data transmission from source to destination must be completed within a defined time interval. Otherwise, such data is of no use. In fact, packet switching network is never designed for real time applications. To support real time applications in packet switching network, traffic control mechanism must be incorporated in order to handle such applications. Besides, different services in packet network have their own characteristic in terms of delay, loss ratio, etc. Those types of requirement can be consolidated as "QoS". In fact, well-defined QoS and its consideration should be studied in detail before designing packet network to perform real time applications.

2.1 QoS Parameter Measurement

There are four outstanding parameters that must be considered in measuring QoS of IP telephony. They are jitter, delay, packet loss ratio and voice compression ratio (voice codec). For these considerations, there is always trade-off when deploying those four QoS parameters. Gaining the improvement in one of these QoS parameters may degrade the other QoS parameters. For instance, either using packet retransmission or using FEC (Forward Error Correction) to recover the error can improve packet loss ratio, but it introduces processing and packetization delay. It should be keep in mind that the aim of traffic control cannot ignore any of those four phenomenon, but it has to guard or reduce such effect to a certain accept level so as to meet individual application's requirement. Before going into system design on packetized voice application in the next section, those four QoS parameters are described in detail.

Another issue arises in packet voice transmission is echo. Different from above four QoS parameters, echo can be overcome or removed by matching its line impedance. Currently, several echo cancellation methods are proposed and implemented to solve the problem. In later section, method on how echo affect packet voice application and how it can be compensated.

2.1.1 Jitter

The occurrence of jitter is due to several factors such as queuing delay, variable packet length, and relative load on the intermediary links and routers. Every voice packet must arrive within a bounded interval to avoid being too late. In the receiver side, the introduction of jitter buffer is one of a commonly used method to compensate jitter effect. This jitter buffer must be carefully tuned to provide an optimal packet-delivery rate while minimizing delay. The real time applications such as voice and video are sensitive to jitter limitations. However, high delay variations introduced to real time applications can lead to high loss to the applications. Obviously, high delay variation requires large buffer size and the receiver design will also be more expensive due to additional memory requirements.

There are more problems while using jitter buffer. Firstly, the buffer requirements may make the design more expensive to implement. Secondly, there might not be a known maximum delay variation on the network, which make size of buffer difficult to determine. Thirdly, the pre-buffer delay may not be acceptable to certain applications. These problems imply that we need to understand a more realistic model of delay variation exceeding the receiver buffer constraints to avoid underflow or overflow.

2.1.2 Delay

Delay is caused by several reasons such as coding/encoding delay, packetization delays, propagation delays and jitter compensation delay. It relates to bandwidth reservation when it comes to QoS parameters. If more bandwidth reservation is available for real time applications, it can greatly improve delay through the network. Different types of real time applications may accept different amount of delay. But on the other side, non-real time applications can accept more delay than its real time counterpart. Quite often, packet delay time is unpredictable and it may depend on several factors that generated during transmission. In Internet Telephony application, there are three causes of delay. They are propagation delay, queuing delay and processing delay.

Propagation delay is due to the speed of light in the channel. Such delay is limited by the transportation media and is difficult to minimize. It is measured by the distance between source and destination that the packet has to passing through.

Queuing delay occurs when there are more packets in the network than the system cannot process immediately at a given interval. When dealing with this type of delay, real time and non-real time application should be consider separately since they have different requirements in delay consideration.

Processing delay includes encode/decode (compression) delay, packetization delay and jitter buffer delay. The advantage of voice compression can optimize bandwidth utilization. μ-law and A-law are the two basic variations of 64kbps Pulse Code Modulation (PCM) commonly used. A 12 to 13 bit linear PCM quality is compressed into 8 bits. In order to achieve a higher compression ratio, another two common compression methods are used, they are the "Adaptive Differential Pulse Code Modulation" (ADPCM) and the "Linear Predictive Coding" (LPC). In general, increasing compression ratio also increases the processing time. In some system, they have to accumulate certain number of packet and then process it to prevent the jitter effect. The system used this method to prevent jitter effect will introduce jitter buffer delay because it has wait for a number of packet to arrive before start processing. Besides, during packet transmission, packetization is to add an appropriate header together with payload according to different protocol specification. The process includes data segmentation, checksum and packet length calculation. All this process consumes time, hence introduce more delay.

2.1.3 Packet Loss

There are two reasons that cause packet loss in multimedia applications - occurrence of noise during transmission and excessive delay time. It is also impossible to re-transmit lost packet in real time applications.

Internet do not guaranteed packet delay because it only uses "best afford" transmission methodology, but on the other side, due to the stringent delay requirements on real time interactive applications, reliable transportation protocols, such as TCP, is not suitable to be used. It seems that packet loss is unavoidable, but it can be compensated by several error detection and error correction methods such as the Forward Error Correction (FEC). In fact, packet loss control is a two-step processes. The first step is error detection and the second step is error recovery. Packet retransmission is one of error recovery method but it is not suitable for real time applications.

The most popular method to recover packet loss in real time communications is the forward error correction (FEC). This scheme has been proposed to alleviate loss busts of a small number of packets. However, the drawback of FEC-based loss recovery is that in order to recover packet *n*, we need to receive packet *n+1* successfully. Thus, it will be subjected to at least one extra frame delay in addition to the processing delay of the FEC encoding and decoding. This additional delay may cause the recovered packet to arrive too late so that it is lost its usefulness anyway.

Another common method to recover the loss packet is adding copies of the previous k frames in the packet containing frame *n*. Like the FEC schemes, it is the most effective scenario in which a large receiver buffer is needed for several frames storage.

The other approach is to build redundancy into the speech coder itself. In this way, algorithms can be designed for each specific coder to recover from such loss but this method is less practical. Speech coders are applied to many different environments such as Internet, Asynchronous Transfer Mode (ATM) and wireless network. The codecs must therefore designed to meet general requirements for loss recovery, which are bother inter and intra operative among these different mediums. However, this end up with a non-optimal codec for any single media. Moreover, loss behavior varies with time, which makes no sense if the behavior cannot be adequately characterized.

Another approach is diversity, which sends multiple copies of the same thing over several paths, and hope that at least one gets there. This approach is very inefficient and generally considered being a very poor method in the Internet environment.

Delay and losses are two main parameters affecting QoS control. For Internet Telephony application normal conversion, users accept a delay of 200ms or less while more tolerant users satisfies with the delay of 300-800ms[3]. Users also require a clear speech coming out from the speaker. There are trade-offs between these two parameters such that it can sati factorize defined in the QoS requirements.

2.1.4 Voice Coder (Voice Compression)

There are two main purposes to do voice coding – digitization and compression. Digitization is the conversion of analog signal into digital signal. Compression is performed in the transmit side to reduce the bandwidth utilization without degrading the voice quality. Currently, the most popular voice coding protocol is the G.711, the G.729 and the G.723.1.

The G.711 is as 64kbps Pulse Code Modulation (PCM) including both A-law and μ-law. It does not involve another kind of voice compression.

The G.729 is a 8-kbps Conjugate Structure Algebraic Code Excited Linear Prediction (CSACELP) speech algorithm providing good speech quality. It is designed for low-delay applications, with a frame size of only 10ms, a processing delay of 10ms and a lookahead of 5ms. This total to a 25-ms contribution [4] to end-to-end delay.

The G.723.1 is a 6.3- kbps/5.3-kbps vocoder for multimedia communications that was originally designed for low-bit-rate videophones. The system encodes the voice signal into frames based on Linear Prediction Analysis by Synthesis (LPAS) coding. A coder is capable of producing two rates of voice traffic

a. 6.3kbps for the high-rate

b. 5.3kbps for the low-rate

The high-rate coder is based on Multipulse Maximum Likelihood Quantization (MP-MLQ), and the low-rate coder is based on Algebraic-Code-Excited Linear Prediction (ACELP).

2.1.5 Echo

Echo is the signal reflections of the speaker's voice from the far end telephone equipment back into the speaker's ear. There are two types of echo to impact packet voice network. The first one is the usual far-end echo caused by the four-wire to two-wire hybrid conversion. End users will hear their own voice signal bouncing off the remote central office's line-card hybrid. The second form of echo occurs when a free-air microphone and speakers are used, as is the case for most PC endpoints. The remote user's voice signal produced by the speakers is picked up by the microphone and echoed back to the remote user. The microphone may receive the PC speaker signal from multiple paths. Echo becomes a significant problem when the round trip delay is greater than 50 milliseconds. Packet voice must address the need for echo control and implement some means of echo cancellation.

2.2 Real Time Protocol Design

Real Time Protocol (RTP) and Real Time Control Protocol (RTCP)[5] are commonly used for real time packet transmission. The purpose of these protocols allows receivers to generate report and such report can be used by applied algorithms in transmitter to compensate jitter and consequence introduced by packet network. RTP is used for real time data transmission and RTCP is the companion control protocol for RTP.

Real Time Protocol (RTP)

It is typically used on the top of UDP, which provides the notion of port and checksum. Figure 2-1 shows the packet format of RTP. Below is some useful fields description of RTP header:

- Sequence Number - It is used to number each RTP packet. At the receiver, the sequence number is used to reconstruct the whole message due to different arrival rate of packet

- Timestamp - It can be used to count the time taken for each RTP packet from source to destination in order to maintain the time information
- Payload type - It identifies the type of message in the RTP stack. It can be used to classify the level of QoS that should be provided during transmission
- SSRC - It identifies the source of RTP stream. All RTP packets within a common SSRC have a common time and sequencing reference.

Real Time Control Protocol (RTCP)

RTCP is used to control packets transmission from time to time regarding a particular RTP session. Basically, RTCP packet types include Sender Report (SR), Receiver Report (RR), source description RTCP packet (SDES), BYE and APP. Below is the description of each reports:

- SR: Sender reports contain transmission and reception information for active senders
- RR: Receiver reports contain reception information for listeners who are not also active senders
- SDES: Source descriptions describe various parameters about the source, including the CNAME
- BYE: Send by a participant when he leaves the conference
- APP: Functions specific to an application

Through these control messages report, network operator can provide appropriate traffic management mechanism base on the result of that reports.

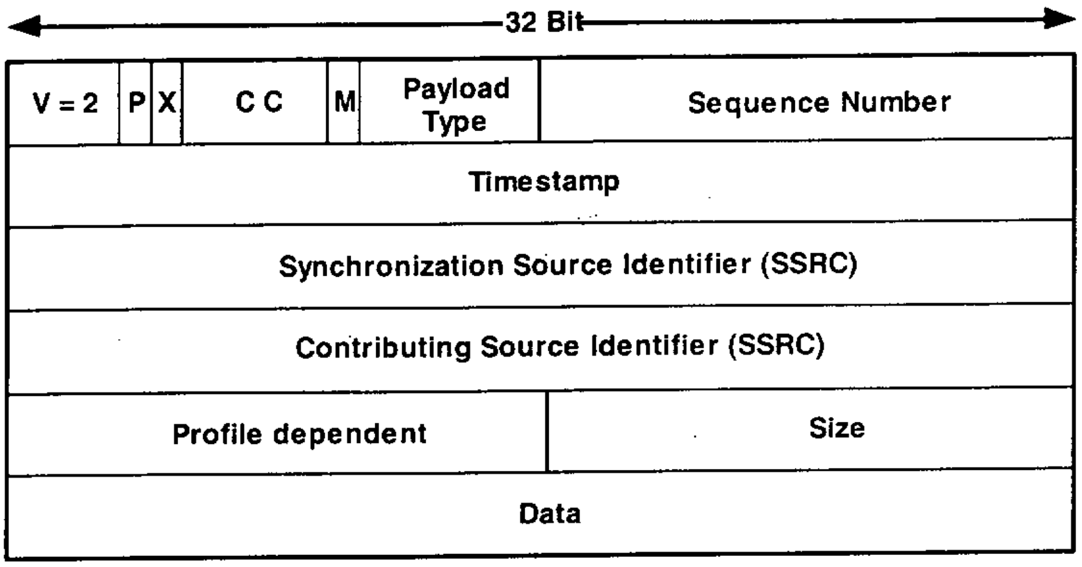


Figure 2-1 RTP Protocol Stack

2.3 Common Voice Model Analysis

In 70s' and 80s', most of the research and analysis on packet voice modeling had started. Some outstanding models are given out in this period [6-9]. Most of research is focused on the voice packet model, queuing management on voice packet network analysis and studying voice packet delay and loss characteristic.

In early state, voice can be modeled as the on-off source model [10]. In talk spurt period, ones are speaking and the source is in 'ON' or active state. At this time, the source generates voice data. In the silence period, the source is in 'OFF' or inactive state and no voice data are produced. In normal conversation, the duration of active period fits the exponential distribution reasonably well while the duration of inactive period is approximated less well by the exponential distribution.

In packet voice communications, voice is modeled as the embedded Markov chain which queue length is modeled as the "overload/underload" cycle based on the phase process [6]. Then the further development on the packet voice queue model is modeled as the bivariate Markov chain embedded at instants of phase state changes, queue increments, and queue decrements. Following is the mutual and common used for packet voice models.

2.3.1 On-Off Source Model

In this voice model, voice conversation is divided into six states according to Brady 's[10] description. It assumed that voice source is based on Poisson distribution. It defined the voice conservation into 'solitary', 'interrupted', 'interruptor', 'pause', and 'alternate' and 'interrupt'. During change of the state, one of the signals is given out to fall into next state.

2.3.2 Semi-Markov Process Model (SMP)

The voice source of this model is assumed to be an alternating renewal process. It means that active and inactive period is exponentially distributed. These approximations adopted for this model ignore the "high frequency" fluctuations present in the system being modeled. The second implication of the approximations is that a renewal occurs at the instants of phase transition. That is, no fractional queue length growth is accrued to account for packets that are in the process of arriving between the time of the latest queue length change prior to a given phase transition and the time of the phase transition.

From the renewal theory, $p_{i,j} = \lim_{t \to \infty} P\{l(t) = i, \wp(t) = j\}$ where *l(t)* is the number of packets in the queue at time *t* and $\wp(t)$ is the number of active sources at time *t*. So, probability of the queue length:

$$\pi_i = P\{l = i\} = \sum_{j=0}^{N} p_{i,j} \tag{2-1}$$

where $p_{i,j} = \dfrac{q_{i,j} m_{i,j}}{\sum_{k=0}^{\infty} \sum_{l=0}^{N} q_{k,l} m_{k,l}}$ is obtained by from the renewal theory.

$q_{i,j}$ is the equilibrium probabilities for the embedded Markov chain, $m_{i,j}$ is the expected sojourn time in state *(i,j)* of the process, *N* is the total number of active source

2.3.3 Continuous-Time Markov Chain Model (CTMC)

This model assumed the generation of packets of the active voice is a Poisson Process. The generation of packets by an active voice source occurs according to a Poisson process with rate β rather than at a constant rate of one packet every 1/Vs as in the system being modeled. Also, the service times are exponentially distributed with parameter υ.

In both real system and CTMC model, the average rate of growth in system occupancy is zero, but the length of time required to observe this zero average growth rate would be quite different. The CTMC model experiences higher frequency fluctuations in system occupancy than does the real system. It is then expected that the CTMC model would yield conservative estimates for probability of buffer overflow and waiting time. That is, if the system were engineered to achieve a certain probability of overflow based on the CTMC model, then the probability of overflow in the system would probably be less than that predicted by the model.

Probability of the queue length is:

$$\pi_i = P\{l = i\} = P\{s = i+1\} = \sum_{j=0}^{N} p_{i+1,j} \qquad (2\text{-}2)$$

where $p_i = [p_{i,0}, p_{i,1}, \ldots\ldots, p_{i,N}]$

N is the number of active sources, s is the number of packets in the system and i is the queue length.

2.3.4 Uniform Arrival and Service Model (UAS)

The active voice source is generated in the buffer at a uniform rate and the buffer removes the packet at a rate not to exceed the link capacity. The approximation of this model is to ignore "high frequency" variations in buffer content as it compares to the real system. That is, in the real system, information does not enter the transmission buffer, and therefore it cannot be transmitted, until a particular active source completes the generation of a packet. In the UAS model, it is possible for the information to be transmitted while it is being received. It would appear that the effect of this approximation on the UAS model's accuracy would be slight since the difference between the behavior of this model and that of the real system is greatest only when the buffer content is low and the number of active sources is below the server's capacity.

Define the complementary queue length distribution for this system, *P{l>i}*, to be probability that the buffer occupancy exceeds $(\alpha/V)i$ units of information. Then, the complementary queue length distribution is:

$$P\{l > i\} \equiv G(\frac{\alpha}{V} i) \qquad (2\text{-}3)$$

where V = packet generating rate, α is the frequency of the active period and *G(x)* is the complementary occupancy distribution.

A thorough discussion of the details of the implementation of the solution is given in [11].

It shows that SMP and UAS models are reasonable for the engineering purposes. It can use to predict the buffer capacity needed to ensure that the fraction of packets lost due to buffer overflow is less than some maximum value.

2.3.5 Markov Modulated Poisson Process Model (MMPP)

This is the most common model to simulate voice packet queue. Since the RFCM assumes such model as the voice packet source, the explanation on this model will be given.

This model is defined to be one having *n* states in general, with the process, while in any state i, $1< i < n$, behaving as a Poisson Process with a state-dependent rate parameter λ. Transitions between states are governed by an underlying continuous-time Markov chain. In packet voice modeling, MMPP is used to analysis the characterization, modeling, and analysis of N voice sources multiplexed at the buffer.

Figure 2-2 shows the MMPP model for N multiplexed voice source with infinite buffer. λ represents the rate of transition from the silent state to talk spurt, and α be the rate in the reverse direction. Also, a utilization parameter ρ, the condition of capacity v and Poisson average arrival rate β is defined. In Figure 2-3, it is easy to find out that the average number of cells per second entering the queue is $N\beta/(\lambda/\alpha+\lambda)$. This must be less than the capacity v, so that:

$$\rho=\left(N\beta\left(\frac{\lambda}{\alpha+\lambda}\right)/v\right)<1 \tag{2-4}$$

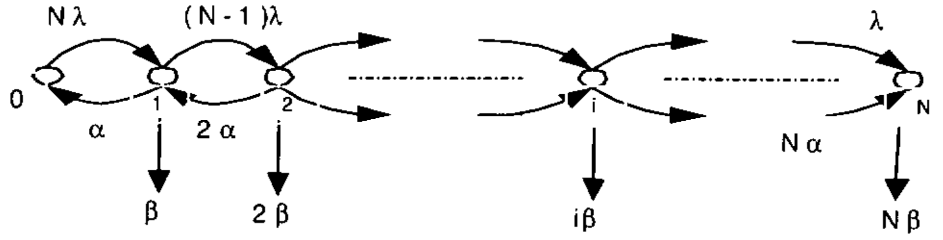


Figure 2-2 MMPP Model, N Multiplexed Voice Sources

Now, the probability distribution of buffer occupancy can be found. Let the integer value $i > 0$ represents the queue state; j represents the number of sources in talk spurt. Also, p_{ij} is defined as the joint probability that the queue length is i with j sources on p_{ij} = P[queue length = i, j source on]

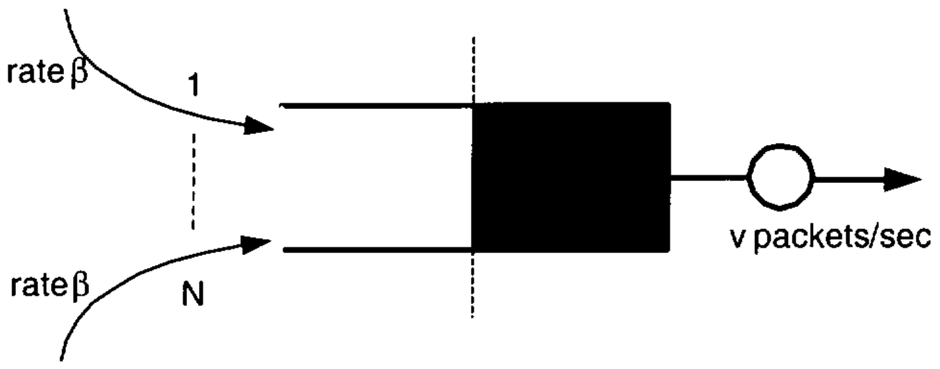


Figure 2-3 Statistical Multiplexer for Input Model

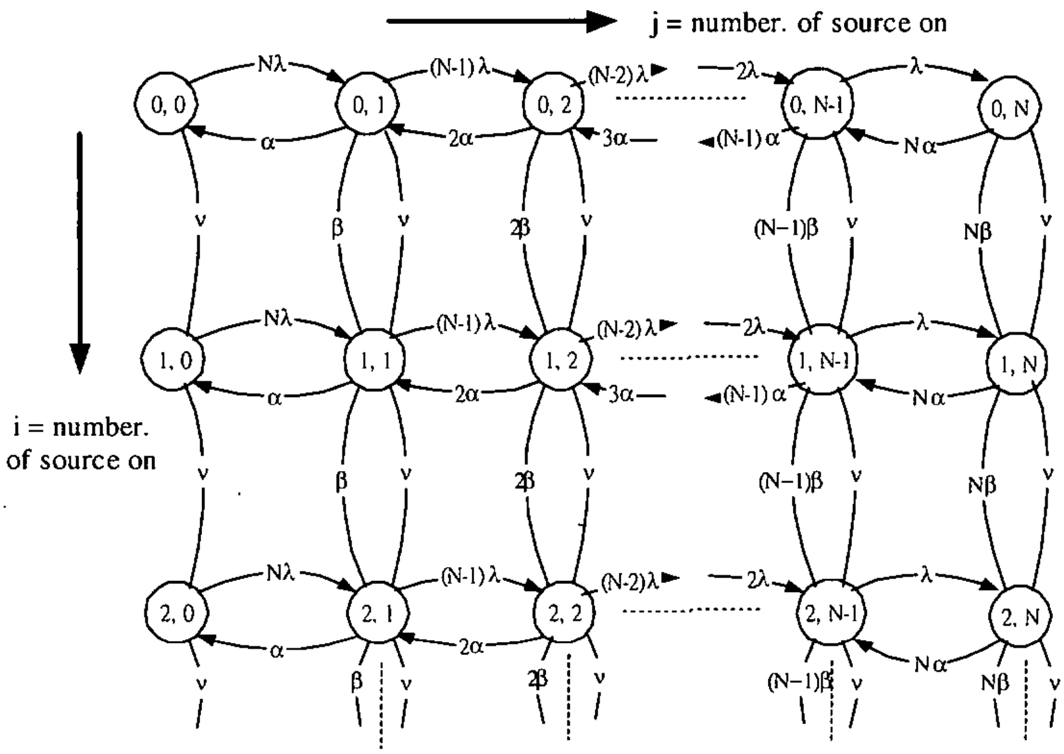


Figure 2-4 Two-dimensional State Space

Figure 2-4 shows the two-dimensional state space for p_{ij}. Three two-dimensional balance equations for this state space.

$$p_{00} = (1 - N\lambda)p_{00} + \alpha p_{01} + v p_{10} \tag{2-5}$$

$$p_{0j} = [N-(j-1)]\lambda p_{0j-1} + [1-(N-j)\lambda - j\alpha - j\beta]p_{oj+1} + vp_{1j} \qquad (2\text{-}6)$$

$$p_{0N} = \lambda p_{0N-1} + [1-N\alpha - N\beta]p_{0N} + vp_{1N} \qquad (2\text{-}7)$$

Then, the equivalent compact matrix vector equation can be written as:

$$P_i = P_o B_o + P_1 B_1 \qquad (2\text{-}8)$$

$$B_0 = \begin{bmatrix} 1-N\lambda & N\lambda & 0 & \dots\dots & 0 \\ \alpha & [1-\alpha-(N-1)\lambda-\beta] & (N-1)\lambda & \dots\dots & 0 \\ 0 & 2\alpha & [1-2\alpha-(N-2)\lambda-2\beta] & \dots\dots & 0 \\ 0 & 0 & 3\alpha & \dots\dots & 0 \\ . & . & 0 & \dots\dots & . \\ . & . & . & \dots\dots & . \\ . & . & . & \dots\dots & . \\ . & . & . & \dots\dots & . \\ . & . & . & \dots\dots & \lambda \\ 0 & 0 & 0 & \dots\dots & (1-N\alpha-N\beta) \end{bmatrix} \qquad (2\text{-}9)$$

$$B_1 = diag[v, v, \dots\dots v] \qquad (2\text{-}10)$$

$$P_i = [P_{i,0}, P_{i,1}, P_{i,2} \dots\dots P_{i,N}] \qquad (2\text{-}11)$$

P_i is a set of (N+1)-element row vector.

From above calculation, it shows that the characteristic of voice source depend on several factor - number of source, buffer length, arrival rate of each source and service rate on each server.

To determine the voice characteristic and the convergence of queue length of the integrated data/voice in separated queue for today's packet switch network, such model should be studied so as to adjust the service rate on servers and define the queue length of voice source in feedback control mechanism and can be enhanced to provide traffic prediction in further analysis.

CHAPTER 3

System Consideration and Performance Requirement

In previous chapters, a numbers of QoS measurement parameters and theoretical description on packet voice are introduced. The characteristic of packet voice can be known through this model. On the other side, network products design technology and implementation algorithm must be in consideration because it affects actual overall performance of whole networking system. Nowadays, the complexity of system networking design requires system level point of view to analysis the network. Highly integrated networking design can provide high efficient and also can fully utilize network resource. Also, the challenge on packet switching network and traditional circuit switching network is on how to interconnect PSTN (circuit switching) and Internet (packet switching) together to provide multimedia application including voice, video and data. For this reason, signaling, addressing and routing technique need to be enhanced such that it provides highly efficient bandwidth utilization and traffic management.

Before designing packet voice network, specifying QoS requirement of performance is needed such that system performance can be measured based on those QoS performance requirements. In this section, it mentions what requirement is needed in packet voice network service, how to defined desired voice quality and how to measure performance. The last topic of this chapter will provide some information about the system architecture on the most common used packet voice networking equipment.

3.1 Performance Requirement for Internet Telephony

Comparing with PSTN, packet voice transmission has to share its bandwidth with other users. In social point of view, the fundamental performance requirement is that the end user can listen voice clearly and they can communicate with each other smoothly. But this requirement is not enough to satisfy our desire and it is not acceptable from technology point of view. In fact, the purpose of using packet voice communications is to reduce bandwidth utilization to deliver higher quality of voice. In other word, it has to use the lowest cost to get high quality of voice communications service. In fact, it has some conflict on resource and quality requirement. There are some questions are raised in performance requirement definition.

3.1.1 How Much Bandwidth Should be Assigned for One Channel

Bandwidth can be defined as a number of bits per second that is available on network. It is a main variable resource in network communications. For those applications, it needs to determine how much bandwidth it desires to provide the appropriate QoS. But on the other hand, bandwidth is a limited resource. The main concern in packet voice application is on how to share limited bandwidth to one or more voice channels. In traditional circuit switching network, the data transmission rate for each voice channel is 64kbps. If the rate is used in packet switching network, the bandwidth utilization in term of channel is very high and inefficient. In fact, there is not need to transmit voice signal at silence period. The first thing need to do is to reduce unnecessary bandwidth consumption for each channel. Two common methodologies are adopted to reduce bandwidth utilization for voice channels – silence detection and voice compression. In silence detector, most of systems are equipped with Speed Activity Detectors (SAD) at each user node such that packets are only generated while users are in a talkspurt period. Since the typical mean duration of the talkspurt and silence period is about the same [10], SAD can potentially double channel utilization. Also, voice compression is another types of method to minimize voice traffic. By applying voice compression, bandwidth utilization can be reduced from 64kbps to lower data rate depending on the applied algorithm. Different compression algorithms can provide different compression ratio. Since bandwidth utilization for each voice channel is based on compressed ratio, one has to choose appreciate voice compression algorithm to give desired QoS.

Currently, the most popular methods for voice compression is G.723.1, G.729 and G.729A. Table 3-1 shows some figures on different types of codec. G.729 and G.729A with the smallest frame size allowing low-latency encoding but a significant overhead are added if only one frame is encapsulated in as RTP packet. This implies that G.723.1 would be favorable to home PC users, who must share what little bandwidth they have with data traffic. Corporate users with direct access to Ethernet or T1 media may refer to G.729A for its favorable delay characteristics.

CODEC	G.723	G.729	G.729A
Bit rate	5.3/6.4 kb/s	8 kb/s	8 kb/s
Frame Size	30ms	10ms	10ms
Processing Delay	30ms	10ms	10ms
Lookahead delay	7.5ms	5ms	5ms
Frame Length	20/24 bytes	10bytes	10bytes

Table 3-1 Codec Bit Rates, Delays

3.1.2 How to Define Desired Quality

There are three main factors that affect voice quality - loss (transmission error), delay and jitter (delay variance).

The statistical measurement and analysis carried out by Guy Almes[12] and Sanghi et. al., [13] showing that the losses in the current Internet are in the range of 2%-10%. Losses have a direct correlation with delay.

Also, experiments conducted at Bellcoe show that the busy hour average packet loss rate on random cross- Internet connections can be as high as 16%. In busy hour, the round trip delay between well-connected sites varies within 100 to 300ms

The above statistical factors reflect the general characteristics of the current Internet.

On the other side, there are some expected performance requirements [14] for Internet Telephony. Telephony applications can be described as relatively tolerant to a small amount of packet losses (about 1% to 2%), but it depends on a small network delay. For end to end delay, lower than 100 or may be 150 ms is generally deemed compatible. If end-to-end delay is between 150 and 350 ms, it is generally considered mediocre. On the other hand, it is not acceptable if end-to-end delay is larger than 350ms. The delay factor included coding and packetization delays, network transmission delays, jitter compensation delay, decoding and play out delay.

3.1.3 How to Measure Performance

In [15], it encloses some approach to measure network performance such as service/transmission approach, call-phase approach, ad hoc approach and combinational approach.

In call-phase approach, it considers each call with three phases - access phase, information exchange phase and disengagement phase. By dividing the call period into separate phases, this approach attains completeness in the criteria as well as network independence from user's viewpoint.

In service/transmission phase, it considers the offer and the actual transmission quality. It associated with the supply, support, and the operation of the service which are separated from the quality of the actual received information. It considers under the functions required at the particular time.

In Ad Hoc approach, it is similar to the call-phase approach but it concern on the non-technical user. In user point of view, they do not concern all the technical knowledge to so how to do that. So the difficult is the satisfaction of user desired.

In Combinational approach, it combined all the above three approaches. The problem of this approach is that the number of criteria may increase significantly, the combining or cross listing gives the resulting criteria less generality, and the criteria may still not encompass all user concerns.

3.2 Architecture and Network Solution on IP Phone

Nowadays, several types of network architectures are interconnecting together to provide data communications service. Some of them are standardized but some of them are not. Providing real time data transmission in packet switching network is one of difficult task to achieve. When designing a high efficiency network, it has to forecast its maximum channel utilization and loading. Besides, scalability is one of main factor to increase the functionality and the processing power. Moreover, the system has to interconnect different types of network such as ATM, IP Ethernet, and frame relay. The interconnected network equipments have to provide the services including to convert packet header format, signaling and handshaking. As a result, traffic management must be considered and implemented in order to manage data transmission and monitor network performance. In fact, processing power of board design and system architecture determines the overall system performance and limited its functionality. In general, when considering real time network design, some factors should be noticed:

a . Connectivity - It is an important factor when determining how to design stable network in order to support a number of terminal or connection in real time network. Intranet estimation of network connection is easier than Internet. Most of network designs are limited by number of port and addressing scheme.

b. Channel capacity - the maximum bandwidth of each path should be known in between source and destination. In general, network is divided into Wide Area Network (WAN), Local Area Network (LAN) and Metropolitan Area Network (MAN). Most likely, WAN is a backbone of the network; it needs high channel capacity because it has to provide high speed of traffic and support huge amount of data transmission. LAN is a small network with a small group of user such as a small company up to 30 people. So bandwidth is not too high but it is important to interconnect with different LANs and enter into WAN. The MAN is the intermediate between LAN and WAN. The covered area may be within a building or a country. To determine channel capacity, maximum loading of the channel should be well known. It is just like a water pipe in our life.

c. Supporting protocol – Different types of protocols has their own network design topology. For instance, Ethernet can support more than 500m in a segment. For Internet, it uses gateways and routers to separate the subnet. protocol design restricts network topology design and limit the overall network performance. Also, WAN and LAN has own specific protocols to support different traffic rate. On the other side, routing algorithms are another types of protocol used to determine the path of packet. Intelligence routing can be used for load balancing to against overloading in real time applications.

d. Service integration - There are a number of service that the network has to provide. It can classify into two different categories - real time and non-real time. Real time applications are time critical data that cannot tolerate much delay but it can accept certain amount of loss ratio. The requirement of these types of service is highly depend on the demand of bandwidth. But, on the other hand, most of non-real time applications have to reliable data transmission. It can tolerate much of delay but the error rate must be kept much low. Packet re-transmission is one of the most popular methods to compensate packet loss. A systematic design of network and proper traffic management mechanism is needed to integrate such types of data.

3.3 Gateway Architecture on VoIP

Gateway, router and switch are three major networking equipments in the Internet. They have their own functionality and definition. In order to handle packet voice traffic, additional functions are needed in these types of equipments such as voice compression, protocol conversion. The architecture of such network equipments directly affects network performance, functionality and quality. Currently, a number of network architecture solutions are provided. Here shown a common used architecture of packet voice transmission system solution in both server side and client/terminal side.

3.3.1 Common Packet Voice Gateway Solution on Server Side

Figure 3-1 shows a common used system architecture for multimedia communications in server side. The system controller in this architecture is a processor based on embedded computer, capable of hosting a real time operating system such as PSOS and VxWork. The combination of hardware and operation system is open, powerful and easily able to host applications available in the desktop computing world.

The gateway design based on Compact PCI architecture. Besides the main PCI bus, the system has the H.100 bus for Time Division Multiplexing (TDM) traffic. This bus is capable of carrying a number of PCM channels. Also, the other bus hosts the non-bused I/O for E1/T1, etc. Besides, another bus is meant for user I/O and Ethernet can be routed on this.

The DSP boards are Compact PCI boards based on several DSP processors and also have a PCI bridge and for hosting networking protocol. The boards have PCI Mezzanine Connector slots on the secondary PCI bus which can host daughter cards for E1/T1, Ethernet or ATM I/O processing. In addition to the PCI bus, these connectors also have connectivity with the TDM bus and Telecom I/O.

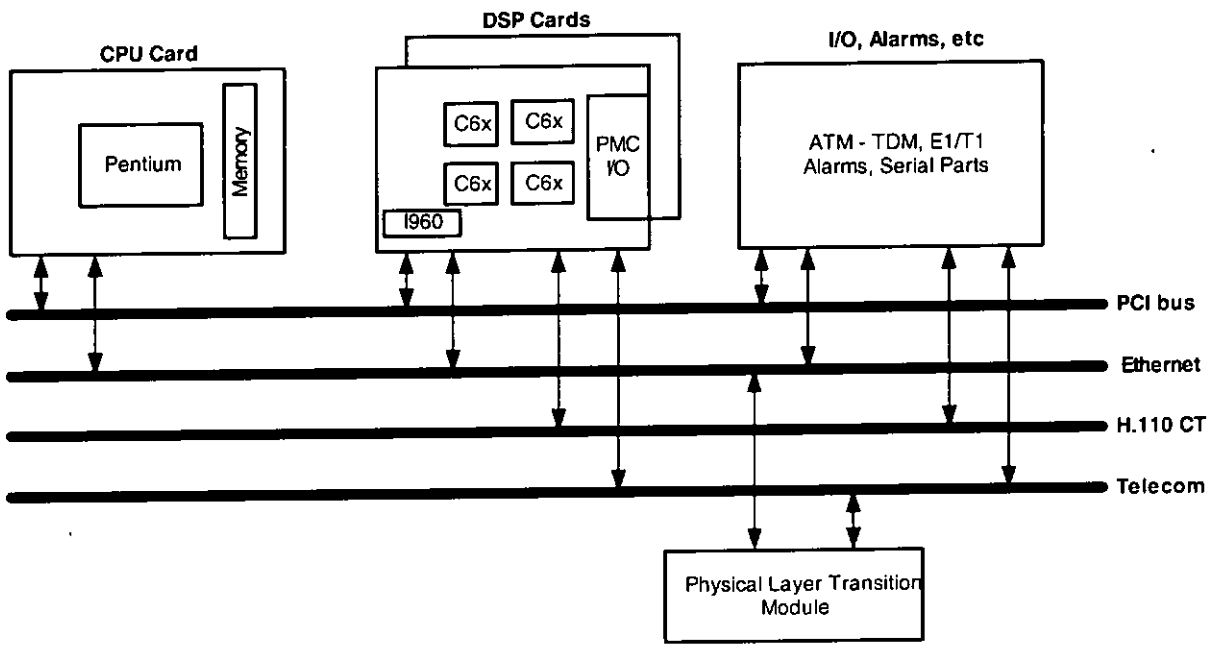


Figure 3-1 Server Side Solution for Voice over IP Gateway

3.3.2 Common Packet Voice Solution on Client/Terminal Side

Figure 3-2 shows a block diagram of a reference design of an IP telephone. It divided into hardware and software design. In hardware design, it consists of the following components

1. User Interface - It provides the traditional user interface functions of a telephone

2. Voice Interface - It provides the conversion of analog voice into digital samples. Speech signals from the microphone are sampled at a rate of 8kHz to create a digitized 64kbps data stream to the processor via a pulse code modulation.

3. Network Interface - It allows transmission and reception of voice data packets from/to the telephone.

4. Processor Core - It performs voice processing, call processing, protocol processing, and network management software functions of the telephone

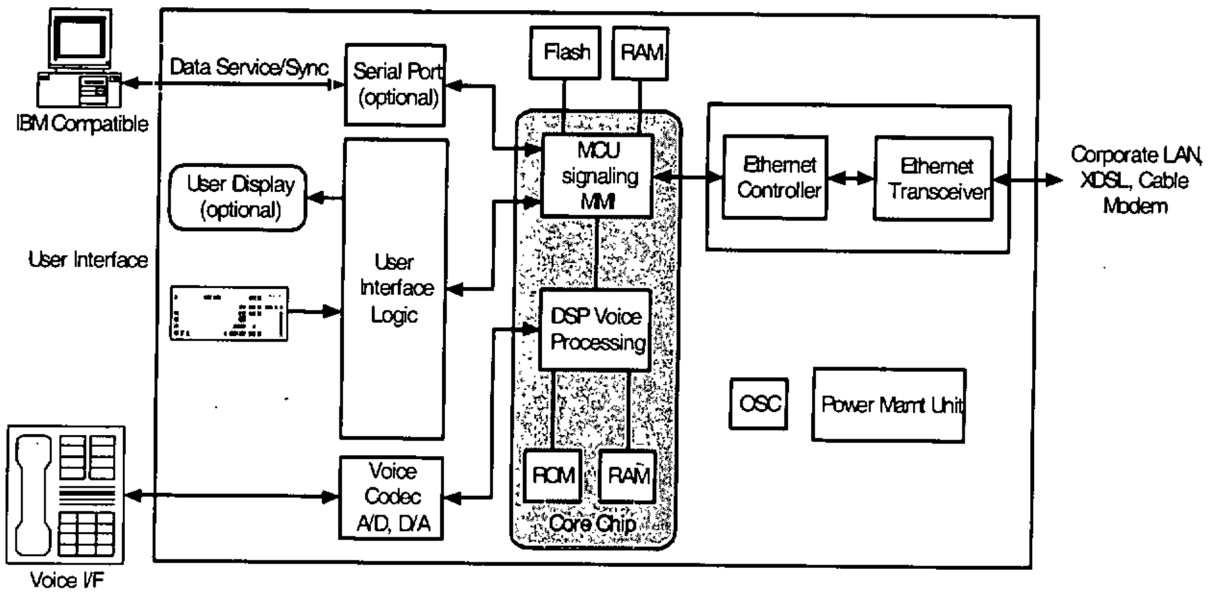


Figure 3-2 IP Telephony Reference Design on Client Side

Besides the reference design for IP telephony, it concerns on the software architecture on IP telephony including system service, signaling gateway, network management, voice processing and network interface protocol. Figure 3-3 shows how these modules are interactive with each other and their functions.

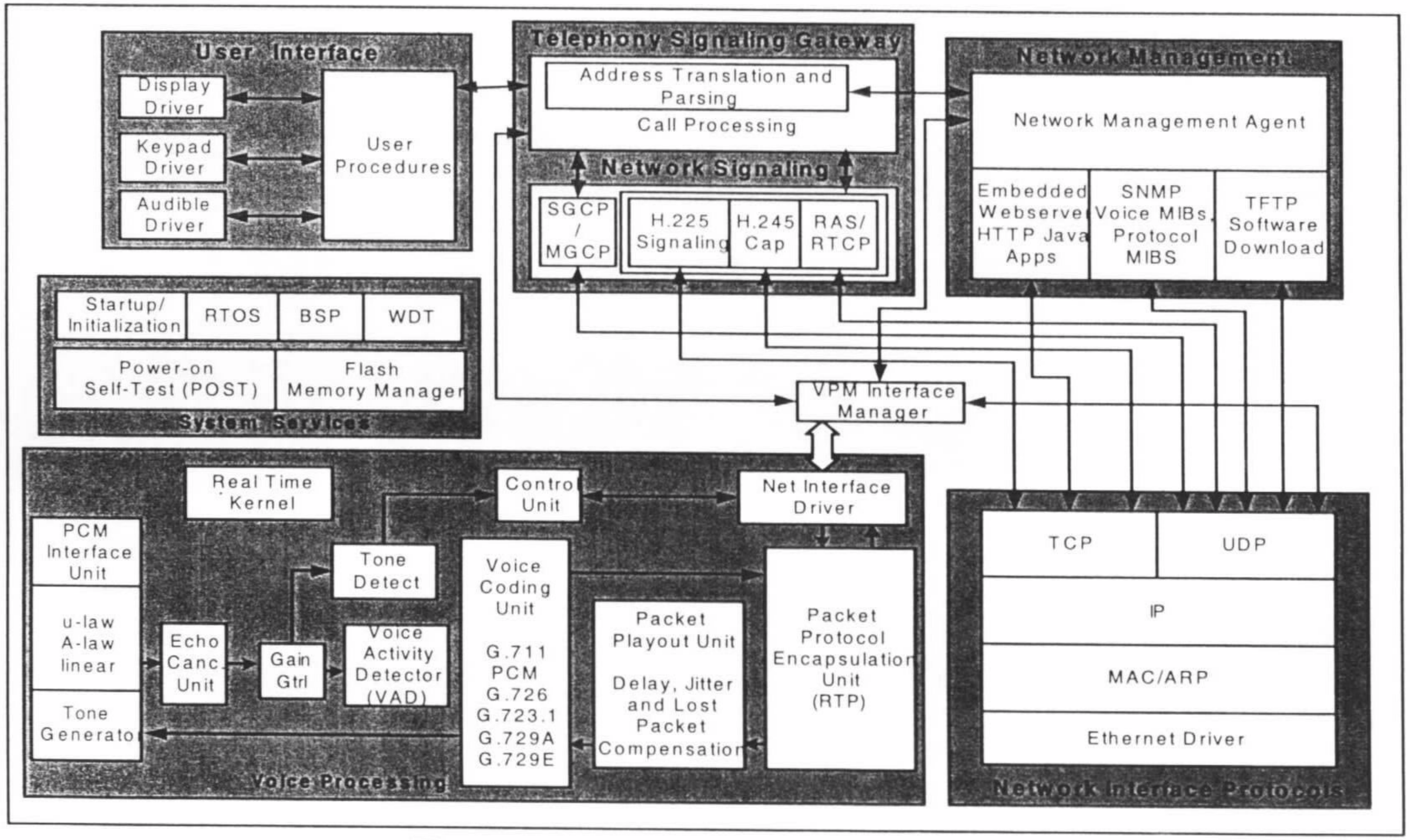


Figure 3-3 IP Telephone Software Architecture

1. User Interface - It provides the software components that handles the interface to the user of the IP telephone and consists of display driver, keypad driver, audible driver and user procedure. User procedures control the information displayed by the display driver and process user key inputs and converts them into primitives for call processing.
2. Voice Processing - It consists of PCM interface unit, tone generator, line echo canceller unit, acoustic echo canceller, Voice Activity Detector (VAD), voice codec unit, packet playout unit, packet protocol encapsulation unit, voice encryption and control unit. These entire modules are used to control and perform voice-processing function. The main function of these modules is to perform voice compression and echo cancellation.
3. Telephony Signaling Gateway - It contains call processing, address translation and parsing, network signaling and H.323 protocols. Most of these functions are to establish, maintain and terminate a call.
4. Network Management - It supports remote administration of the IP telephony by a network management system. It consists network management agent, embedded web server, Simple Network Management Protocol (SNMP) and Trivial File Transport Protocol (TFTP).

5. Network Interface Protocol - It supports communications over the LAN and the module consists of TCP, UDP, IP, Media Access Control (MAC), and Ethernet Driver.

6. System Service - It consists of startup, Power-On Self-Test (POST), Real Time Operating System (RTOS), Board Support Package (BSP), watchdog timer driver, flash memory manager and DSP interface manager. Most of these functions are directly interactive with hardware. It isolates the hardware and upper layer function. Though this function, upper layer function becomes hardware independent.

CHAPTER 4

Deficiencies on Current Method

Nowadays, there are so many algorithms to provide traffic management in order to support different types of multimedia applications in packet switching network. Such algorithms have their pros and cons. But, in general, most of them cannot provide systematic and realistic approach to handle real time packet transmission. In this chapter, two realistic and systematic approaches are studied to handle real time traffic - Direct Adjustment Algorithm (DAA) and Adaptive Transmission Scheme (ALT).

4.1 Direct Adjustment Algorithm (DAA)

4.1.1 Theoretical Description

The direct adjustment algorithm is based on both additive increase and multiplicative decrease approach as well as directly using the bandwidth to share a TCP connection under the same round trip delay, packet size and loss ratio. This scheme achieves high bandwidth utilization, low loss and a fair distribution of the available bandwidth among connections with the same round trip times, but it is unfair with longer round trip times.

During a RTP session, each receiver reports the percentage of lost data to its control packets since sending the last control packet. At the sender site, the RTCP packets are processed depending on the loss values reported within the RTCP packets, the sender can increase or decrease its sending rate. With the reception of each RTCP packet the sender needs to do the following:

1. Calculate the round trip delay to the reporting receiver and determine the propagation delay. The RTCP receiver report includes the timestamp of the last received report from this sender and the time elapsed since receiving this report and sending the corresponding receiver report. Knowing the arrival time *T* of the RTCP packet, the end system can calculate the round trip time (*RTT*).

$$RTT_i = T - DLSR - LSR \tag{4-1}$$

where *LSR* is last received report from sender and *DLSR* is the processing time between the receiving the sender report and sending the corresponding receiver report.

2. In the RTCP receiver report, it contains the value of the average packet loss measured for the sender in the time between sending two consecutive RTCP

packets at the reporting receiver. To avoid reactions to sudden loss peaks in the network, the sender maintains a smoothed loss ratio L_s for each receiver.

$$L_s = (1-\alpha)L_s + \alpha L \qquad (4\text{-}2)$$

with L as the loss value reported by RTCP packet and α as a smoothing factor.

3. Using the smoothed loss ratio and round trip delay, the sender can calculate the maximum bandwidth R_{TCP} share by using [16] and [17]

$$R_{TCP} = \frac{1.22xMTU}{RTTx\sqrt{L_s}} \qquad (4\text{-}3)$$

with MTU as the maximum packet length

4. If the smoothed loss ratio smaller then 16% (simulations run in [16] show that it only applies up to loss ratio of 16%), the sender sets its transmission ratio to (min[R_m , R_A] with

$$R_A = R + \frac{AIF}{m} \qquad (4\text{-}4)$$

with AIF as the additive increase factor, R as the current transmission rate and m is set to (min[members, h]). The number of members in a session can be determined using the RTCP packets and the scaling threshold (h) can be calculated as

$$h = \frac{\text{min. Interval x } B_R}{\text{RTCP Packet Size}} \qquad (4\text{-}5)$$

with B_R as the bandwidth dedicated for RTCP and is set to 5% of the bandwidth used by the session. R_m is the minimal R_{TCP} calculated for all receivers.

5. With smoothed loss ratio greater than 16%, the sending ratio is reduced by half. For the reduction to take effect, all RTCP packets are ignored for five seconds, which is approximately the minimum interval between the sending of two consecutive RTCP packets or until another RTCP packet from the receiver, who reported the loss is received.

4.1.2 Performance Measurement on Direct Adjustment Algorithm

The topology in Figure 4-1 is setup to measure the performance of the Direct Adjustment Algorithm (DAA). The model consists of 15 connections sharing a bottleneck router. All connections deploy the direct adjustment algorithm and are persistent sources. That is, they always have data to send with the maximum allowed rate. They all have the same round trip times and are similar in their requirements and characteristics. The router contains 200kbytes of buffer size. In all simulations, it has to set the packet size to 1kbyte, which is typical voice packet size. Table 4-1 shows the average utilization results achieved for different round trip times and different link bandwidth. The results shown in Table 4-1 and Figure 4-2 reveal that using direct adjustment algorithm, bandwidth utilization between 60% and 99% is possible and the bandwidth is equally distributed among all connections sharing the same link. Also, while connection 15 starts 100 seconds later than the other connections it soon reaches the same bandwidth level.

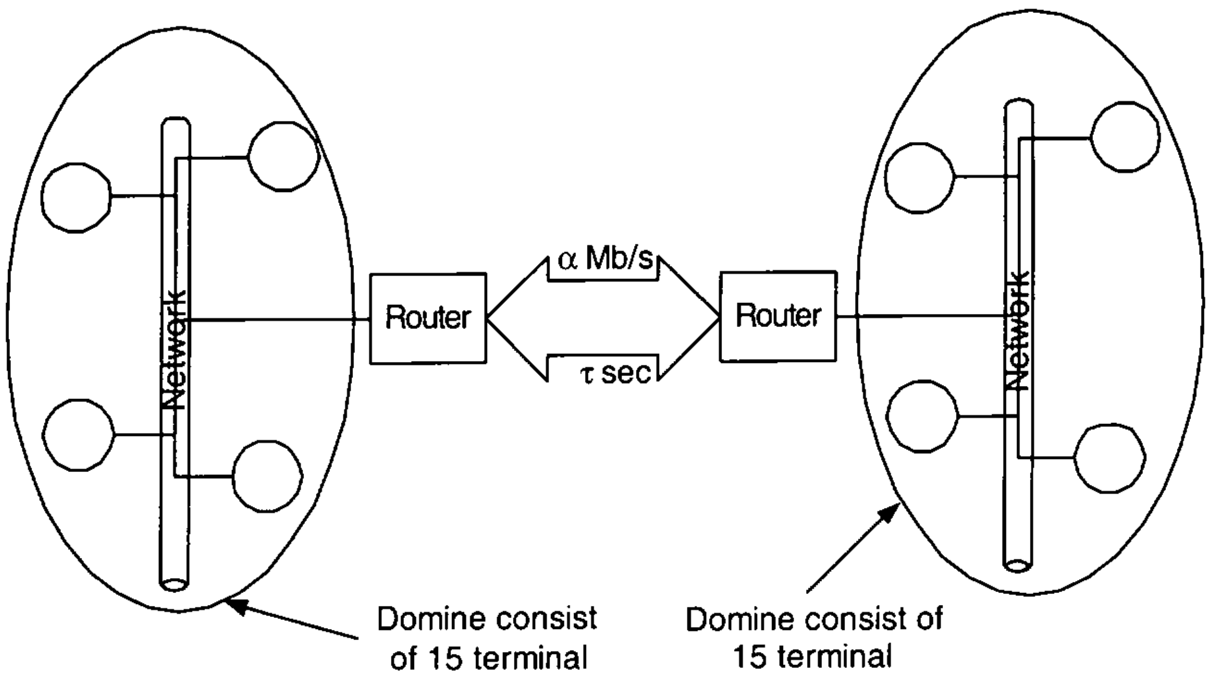


Figure 4-1 Direct Adjustment Algorithm Performance Testing Topology

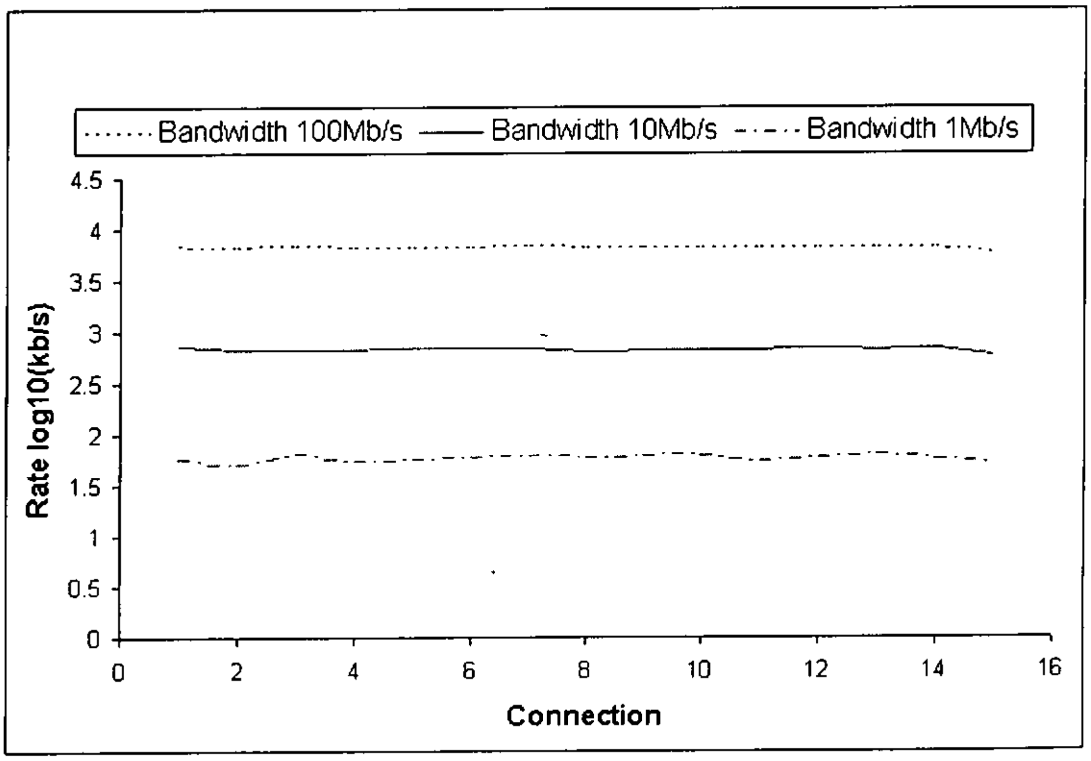


Figure 4-2a Rate of Single Connection of DAA under 5μ Second Round Trip Times and Link Bandwidth

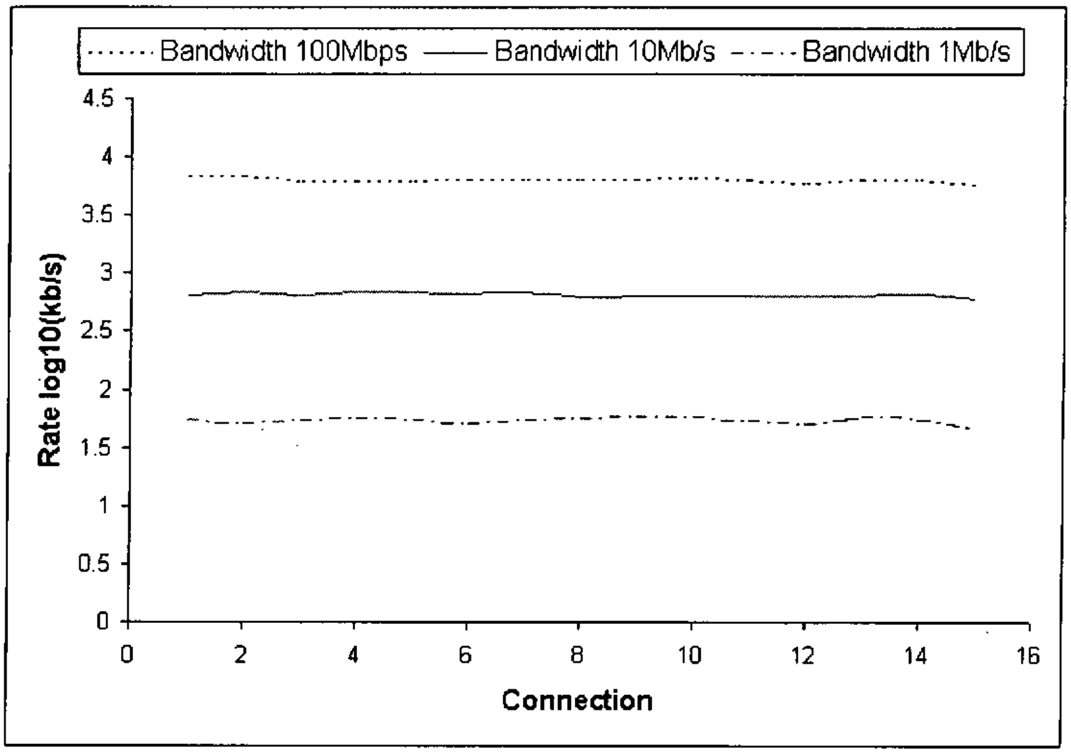


Figure 4-2b Rate of Single Connection of DAA under 5m Second Round Trip Times and Link Bandwidth

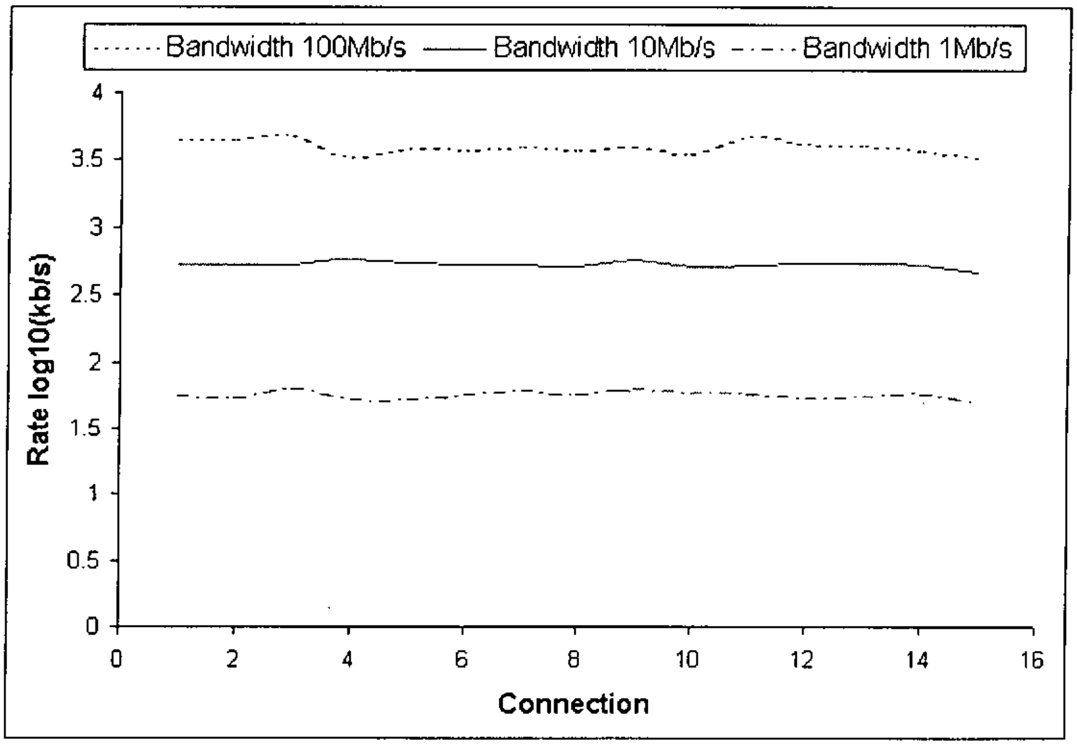


Figure 4-2c Rate of Single Connection of DAA under 100m Second Round Trip Times and Link Bandwidth

Propagation Delay (τ)	α=1 Mb/s	α=10 Mb/s	α=100 Mb/s
5 μsec	0.89	0.99	0.99
5 msec	0.82	0.97	0.95
100 msec	0.85	0.8	0.6

Table 4-1 Link Utilization with the Direct Adjustment Algorithm for Different Propagation Delays (τ) and Link Rates (α)

4.2 Adaptive Layered Transmission Scheme (ALT)

4.2.1 Theoretical Description

With ALT, the receiver notifies the sender of the congestion state upon which the sender reduces its transmission rate on the congested layer and increases it on a higher one such as layer exits. Thereby, the overall transmission rate may stay constant and the end systems receiving all layers will not be affected by the rate changes.

The ALT scheme consists of three major parts: the feedback information, an adaptation scheme and the mechanisms of the dynamic redistribution of the bandwidth among the different layers.

A. Adaptation Scheme

The adaptation scheme is based on the RTP. It uses the RTCP packet information to get the status of each session member

With the adaptive bandwidth adjustment scheme, the sender reduces its transmission rate by a multiplicative decrease factor after receiving feedback from the receiver that indicates losses above a certain threshold call the upper loss threshold. With losses below a second threshold call the lower loss threshold, the sender can increase its transmission rate additively. For the case that the feedback information indicates losses in between the two thresholds, the sender can maintain its current transmission level. Reducing the rate multiplicatively allows for a fairer reaction to congestion. That is, connections utilizing a disproportionately large bandwidth share are forced to reduce their transmission rate by a large amount.

B. Mechanism of the Dynamic Redistribution

The Dynamical Bandwidth Arbitrator (DBA) is used to distribute the overall transmitted rate among the different layers based on the transmission rates for each layers. The operation method is shown as follows:

1. The sender starts transmitting data with an initial rate R_i on each layer L_i.
2. Receivers join the first layer, and if no losses were noticed they join the second layer and do so on.
3. If losses were noticed on a layer, then on the contrary to the approach presented in bandwidth adjustment scheme, the receiver does not need to leave the congested layer, but it should inform the sender about the loss. This notification is done by sending the periodic reports to the senders with RTCP.

4. Upon receiving a loss notification at the sender site the adaptation instance of layer L calculates a new reduced transmission rate R_i and informs the arbitrator about the new transmission rate.

5. To maintain an overall constant transmission rate, the DBA tries to shift the reduced transmission rate of layer L_i up to layer L_{i+1}.

6. If the layer L_{i+1} was transmitting data at a rate (R_{i+1}) less than its initial rate R_{i+1}* which indicates congestion on that layer, then the rate should be shifted directly from layer L_i up to L_{i+2} or up the first higher layer L_x transmitting at a rate $R_x > R_x$*

7. In the case that a layer (L_i) is underutilized, the adaptation instance on that layer would signal the DBA to increase the transmission rate (R_i). This is done either by increasing the overall transmission rate by the requested amount or by reducing the transmission rate of the highest layer. The first approach would lead to a QoS improvement at all members. The second approach is suitable for the cases, where the send data stream is constant. In this case, only members listening up to layer (L_i) would benefit from the rate shifting.

8. To avoid over reducing the rate on a certain layer, a receiver should leave a layer if he is facing losses higher than a certain threshold.

9. In the common case, a low value over a layer (L_i) would result in an increased transmission rate (R_i). However, as the sender adjusts its transmission rate down to

the worst receiver level, the only way for a receiver listening to this layer and having no losses to improve its QoS is by joining a higher layer. In this, the receiver should follow a scheme such as Receiver-Driven Layered Multicast (RLM) [18] or the reservation-based scheme [19]. It would be valid for the case that the sender is sending with R_i equal to the maximum transmission rate for layer L_i.

10. Finally, in the IP multicast model, the multicast data would most probably traverse the same paths until reaching a splitting point. If congestion happens on a link traversed by all the layers, the ALT scheme would try to readjust the bandwidth distribution, by sending more on the highest layer and less on the lower layers. However, in this case the highest layer is congested as well. With the adaptation applied on the highest layer, the sender would reduce its transmission rate. If the transmission rate was reduced below the minimum transmission rate of highest layer, the sender should just quit this layer and only restart it when the lower layers seem to be satisfied. This applies then recursively to all lower layers as well.

4.2.2 Example for ALT Scheme Implementation

A three layers model is used to show how to implement the concept of the ALT scheme. Figure 4-3 shows the implementation model. Two main components are the adaptive sender and the controller.

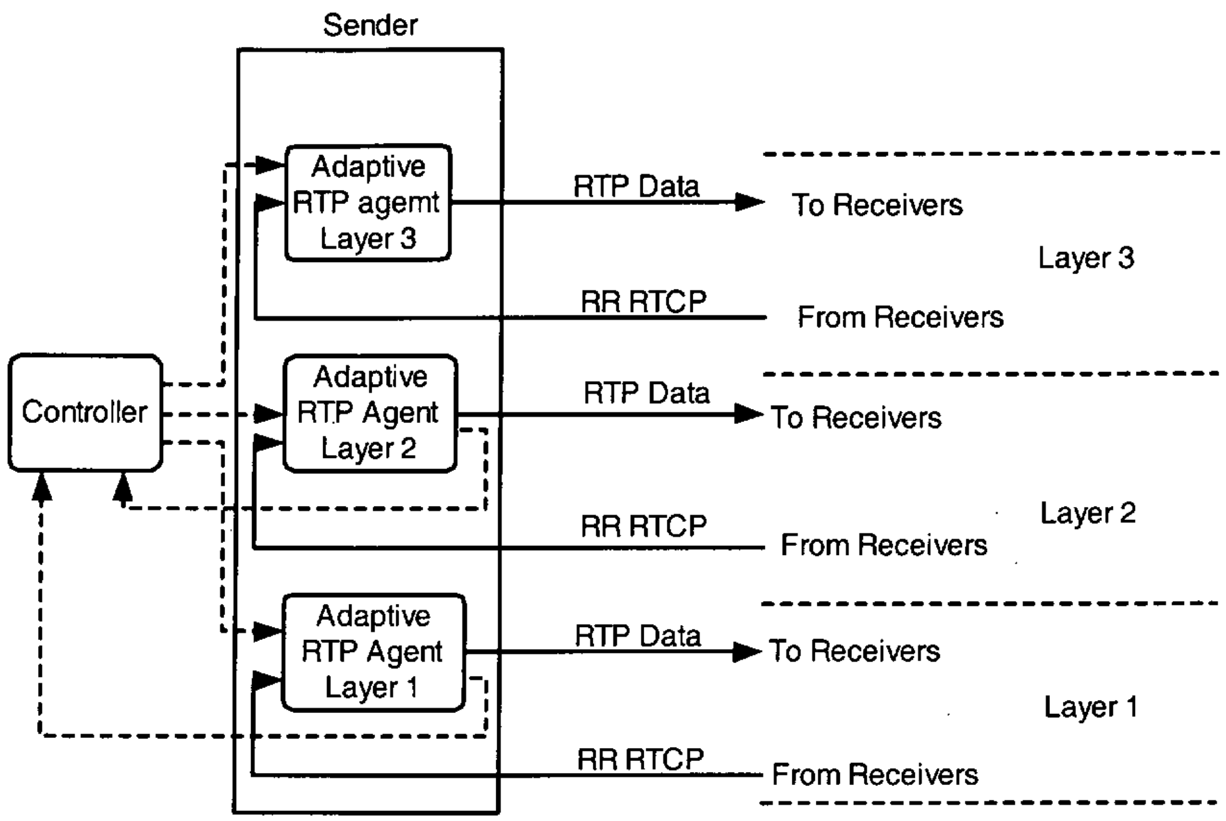

Sender
Adaptive RTP agemt Layer 3
Adaptive RTP Agent Layer 2
Adaptive RTP Agent Layer 1
Controller
RTP Data
To Receivers
RR RTCP
From Receivers
Layer 3
Layer 2
Layer 1

Figure 4-3 Diagram of the Control Scheme with Three Layers

The sender consists of three instances of the adaptation scheme with each instance controlling the bandwidth of one layer. The sender starts transmitting data with the initial rate for each layer. The receivers are listening to a certain layer that send back RTCP packets containing Receiver Report (RR) about the received data over of that layer. According to the reported loss rates, the adaptation instance of that layer makes an adjustment decision. The amount of bandwidth needed to be increased or decreased on a certain layer is communicated to the controller. According to the bandwidth calculated on layer 1 and 2, the controller recalculates the bandwidth distributions. This is done in the following way:

1. The controller is waiting for a message from the adaptation on layer 1 or layer 2 which asking for a rate change.
2. The layer of the adaptation instance asking for a rate change is called the "actual layer".
3. If the bandwidth of the actual layer is to be decreased, the amount of the desired decrease is subtracted from the current rate of the actual layer and added to the rate of the next higher layer. The controller then redistributes the overall used bandwidth rate among the three layers.
4. On the other hand, if the bandwidth on the actual layer is to be increased, the controller decreases the rate of the highest layer (layer 3).
5. After redistributing the bandwidth the actual layer is set to the next higher layer that may ask for decrease or increase. That is, the controller waits for a message from the adoption instance on the new actual layer. Thereby, before a layer can change its rate again, the influence of the previous change on the other layers will be considered.

4.2.3 Performance Analysis

Figure 4-4 shows the experiment configuration of the topology where sender is transmitting data over three layers to their receivers. One of the receivers is only receiving one layer, another is receiving two and the final one is receiving all of the layers. As background traffic, it used an active - idle source [20], see Figure 4-5.

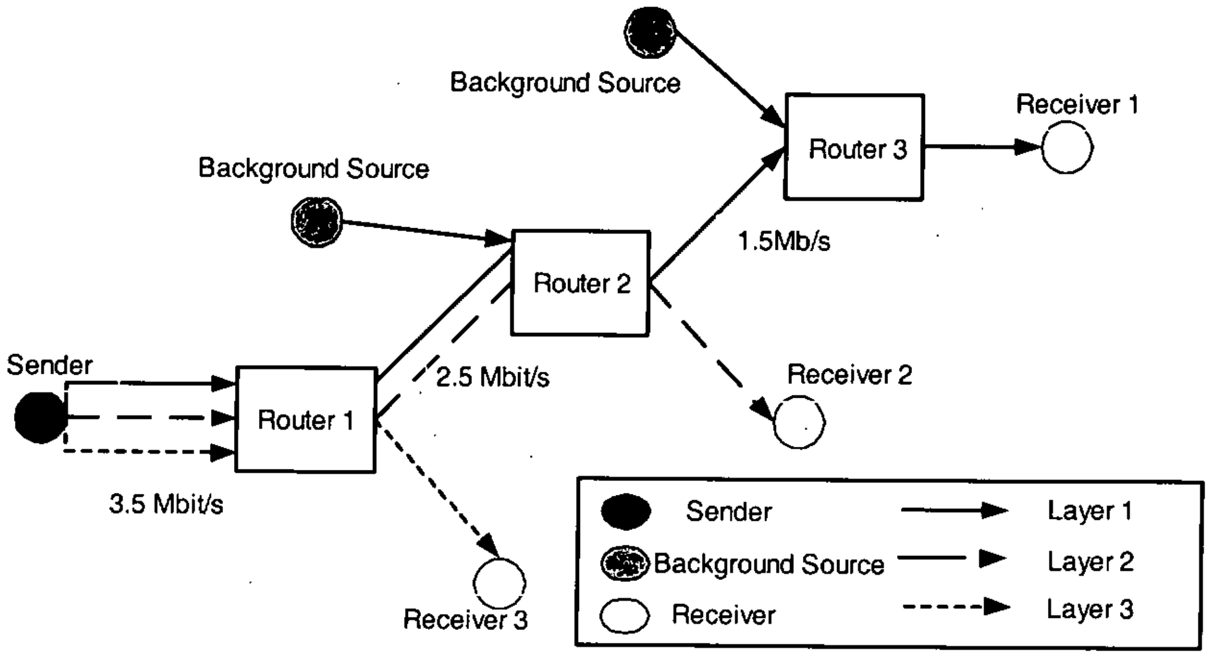


Figure 4-4 Diagram of the Control Scheme with Three Layers

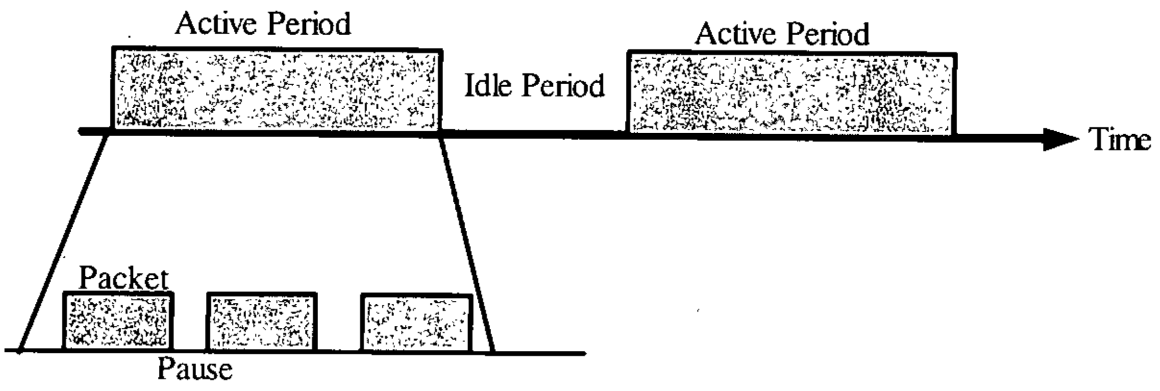


Figure 4-5 A Three State Active-idle Source

Now, let the number of packet generated at the rate 50, the pause between two packets during active state set to 2m seconds and idle state set to 5m seconds. Table 4-2 shows the parameter value for the adaptation instances. Also, background traffic is turned on at router 3, thus it affects the traffic of the first layer. To increase the congestion state, additional background traffic is added to the router 2 starting from time 500 seconds and thereby affecting both layer 1 and layer 2. When the transmission rate of a certain layer is increased, the transmission rate of the highest layer is reduced.

Parameter	Value
Initial Bandwidth	64 kbit/s, 500 kbit/s, 1.5Mbit/s
Lower Loss Threshold	5%
Upper Loss Threshold	10%
Additive Increase	50kbit/s
Multiplicative Decrease	0.875
Minimum transmission rate	50kbit/s
Maximum transmission rate	10Mbit/s

Table 4-2 Parameter Values for the Adaptation Instances

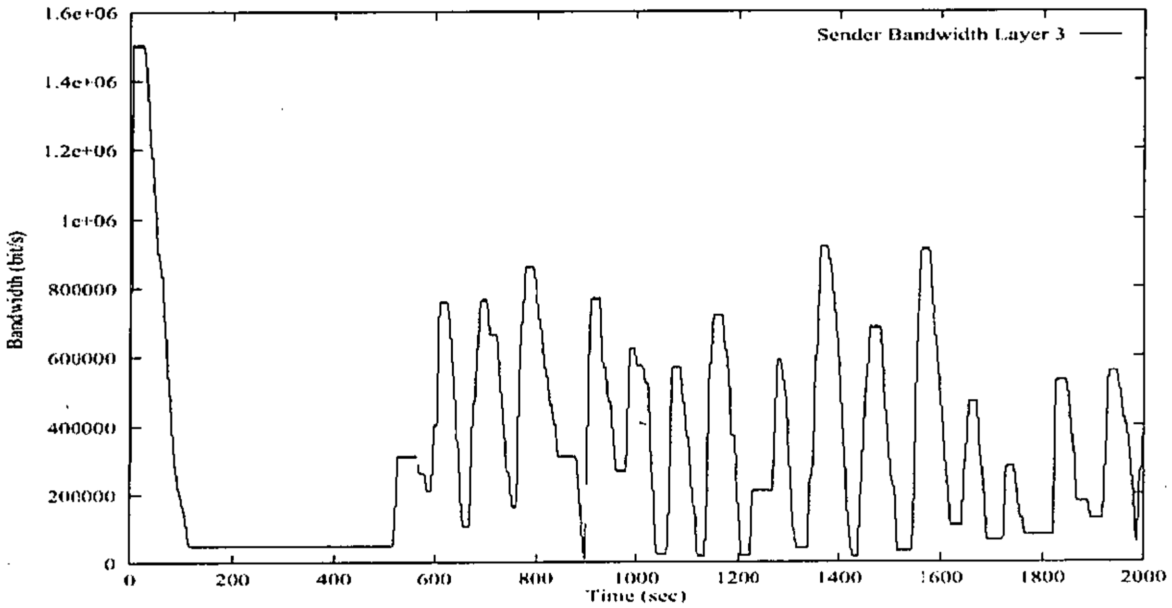


Figure 4-6 Sender Bandwidth Layer 3

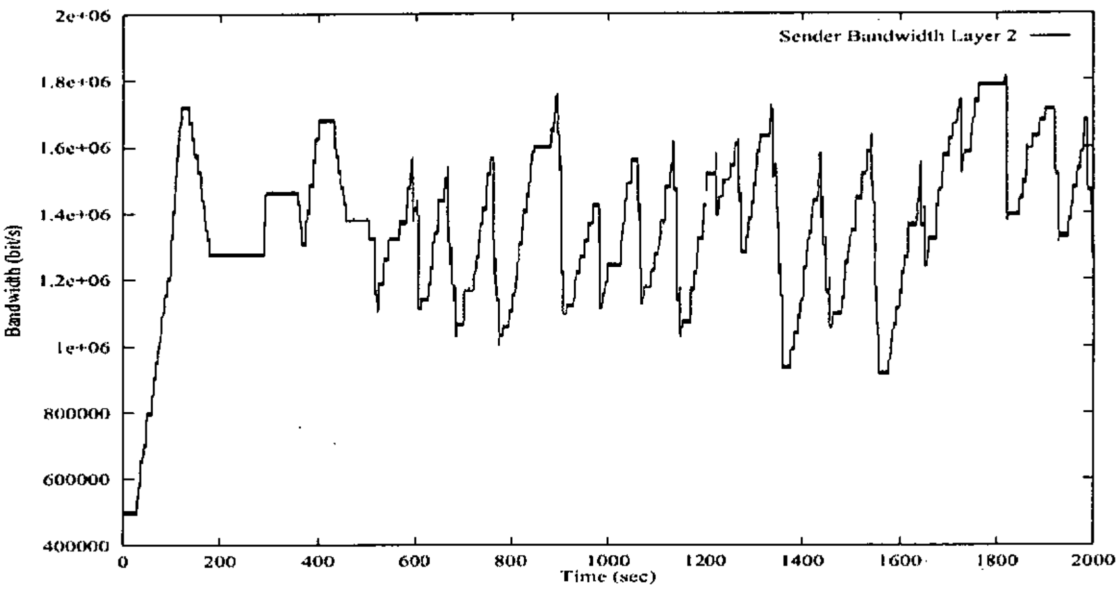


Figure 4-7 Sender Bandwidth Layer 2

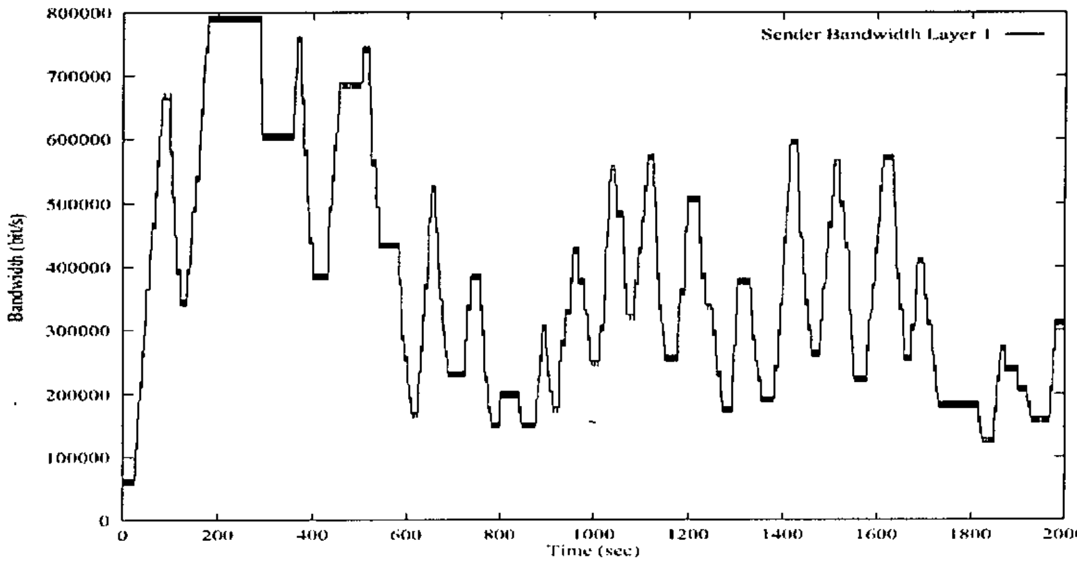


Figure 4-8 Sender Bandwidth Layer 1

The result in Figure 4-8 reveal that while the sender starts transmitting data on the first layer with an initial rate of 64kbit/s, it finally oscillates around 500kbit/s. Figure 4-7 depicts a similar result for the second layer, that is the source starts with 500kbits/s and reaches 1.5Mbits/s on the average. For the third layer, the sender actually decreases the transmission rate. As the rate gets reduced down to the minimum transmission rate, the user could actually drop this layer. While this would slightly reduce the overall data amount arriving at the receivers, it might improve the QoS at the end systems as the receivers would only need to consider two layers which results in easier resynchronization of the data and possibly lower delays.

The results show that with the ALT scheme, the sender adapts the transmission rate for each layer in accordance with the available network resources. Thereby, the QoS of the different receivers improves considerably. Receiver number 1, for example, does not only get the 64bits/s initial rate chosen for the base layer, but can actually benefit from around 500kbits/s. Also while receivers number 1 and number 2 suffer from losses due to the background traffic, receiver number 3 which listens to all layers gets a constant data stream and is not affected by the rate shifting procedure.

4.3 Open Issues for These Networks

1. Oscillatory behavior in DAA - Multimedia application that could benefit from adaptation schemes requires steady rate streams. Notifying end systems of the specified bandwidth would not require for the end systems to detect the appropriate transmission rates and hence would reduce those oscillations considerably.
2. Scalability in ALT - In multicast sessions with various numbers of participants when increasing the number of session members increases the number of received

RTCP packets at the receivers. Therefore, the additive increase factor needs to the scaled with the number of control packets. Another possible problem with the scalability of RTCP is that, for large numbers of members, the interval between two RTCP packets for a single source increases. If a receiver reported high losses and had only a low rate, that is throughput according to the TCP model, the sender would not be able to increase its transmission rate above this limit until another RTCP message indicating lower losses arrives from that receiver. For sessions having a large number of members such an update might take a long time ranging from several minutes to several hours.

3. Insufficient bandwidth for input source - If the maximum available bandwidth cannot handle the total arrival rate of sources, i.e. it cannot reserve more bandwidth for particular channel, violation of delay constraint still occurs and QoS requirement cannot be met. In practice, the total transmission rate is limited by the channel capacity. If total bandwidth in the channel cannot handle the total arrival rate from the source, the packets accumulate in transmitted queue until the channel has available bandwidth for transmission. In this condition, DAA and ALT are not able to reserve sufficient bandwidth to guarantee constant delay.

CHAPTER 5

Traffic Control on Real Time Data Communications

5.1 Background

There are two types of categories for data communications when taking timing consideration - real time and non-real time packet. Internet telephony and teleconferencing applications are the examples of real time packet transmission. E-mail and FTP is considered as non-real time packet transmission, belongs to the latter category. Most of real time applications are time critical but it can tolerate certain level of loss. On the other hand, non-real time applications are loss sensitive but it can accept a long delay during transmission.

This chapter investigates the Real Time Feedback Control Mechanism (RFCM) in data and voice-integrated network based on its QoS requirement. The mechanism provides real time traffic control and adaptive adjustment in bandwidth management. It consists three models, namely:

- Service model - Service model separates real time and non-real time data source.
- Real time control model - Real time control model schedules and prioritizes packet according to the feedback parameter coming from RTCP report and queue length provided from service model.
- Channel model. - Channel model specifies channel characteristic such as channel capacity.

These models interconnected together and used to control data flow in packet switching network so as to provide traffic management by adaptive bandwidth reservation.

Voice is a continuous, real time, non-reproducible data source in packet switching network. Different applications involved voice transmission has their constraint in defining minimum time of delay. In Internet Telephony, the expected voice delay in between source and destination should be less than 40ms. The most critical users required delays of 200ms or less while more tolerant delays of 300-800ms [21] could be accepted. In early state, voice can be modeled as the on-off source model [10]. In talk-spurt period, ones are speaking and the source is in 'ON' or active state. In this period, the source generates information. In the silence period, the source is in 'OFF' or inactive state and no information is generated from the source.

On the other hand, to simply analysis on feedback control mechanism, it assumes that the rest of data resided in the channel can be classified as continuous, non-real time and loss sensitive information. Most likely, such non-real time data is modeled as a Poisson Process (random source). File Transfer Protocol (FTP), file sharing or web access applications are examples of non-real time data.

In this chapter, multiplexing voice and data issue in time delay and buffer payout relationship are explored and reported in more detail. It uses three models (service model, channel model and real time control model) to evaluate and analyze these relationships. Base on these three separated models analysis, real time control mechanism, which will achieve the QoS requirement on the real time data transferring in packet network, was explored.

5.2 Real Time Feedback Control Mechanism (RFCM)

There are three types of model in Real Time Feedback Control Mechanism (RFCM) - service model, real time control model and channel model. These three models are interconnected together to provide traffic management over Internet. The purpose of the service model separates voice and data source into different priority queues. It provides number of queue to store and classify different types of data according to the application QoS's requirement in analyzing packet voice characteristic. The second model (real time control model) makes use of RTP/RTCP to setup control mechanism to prioritize the flow of packets in the network. The third model (channel model) specifies channel characteristic such as channel capacity.

5.2.1 Service Model

Figure 5-1 shows the simple architecture of service model. It consists of two queues, voice queue and data queue, which is used to separated real time source and non-real time source. Generally, the queues can be further decomposing in several queues based on the specific applications QoS requirement. In this analysis, it only uses two different characteristic sources - real time and non-real time data as an example to realize traffic management.

Such two queues has different source of parameters. Voice source is assumed as a two-state Markov Model. Two-state are the period of activity and the period of silence. In each state, it may be in the same state or going to the other state in next sampling time. On non-real time application, the data is modeled as the Poisson source.

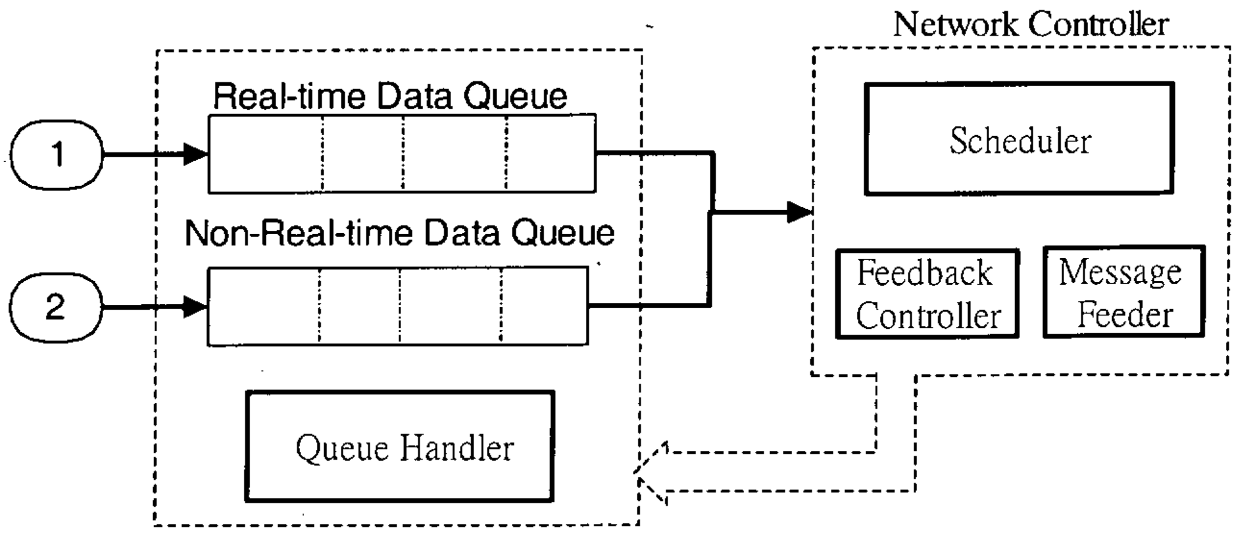


Figure 5-1 Service Model

In traditional analysis, most of popular model for voice is Markov Modulated Poisson Process (MMPP) [22], The equation for the waiting time for real time channel is:

$$W_a = W_M + \frac{uh}{\rho(1-\rho)} \tag{5-1}$$

$$\text{where } W = \frac{\lambda_t h^{(2)}}{2(1-\rho)}$$

u is the index of waiting time, ρ is the utilization factor, $h^{(2)}$ is the second moment of service time and λ_t is the total arrival rate.

In non-real time data source, it is modeled as the Poisson distribution. The waiting time on the packet network is calculated using the Little's Law:

$$W_c = \frac{L}{\lambda_d} \tag{5-2}$$

Non-real time data traffic is burst source in nature and tolerates substantial delay but real time traffic is time critical. It assumes that non-real time data traffic is regularly and uniformly distributes with time. The arrival rate of non-real time data is random and is queued in non-real time queue. The service rate of non-real time data queue is determined by network controller in the real time control model such that the arrival rate of real time data is equal to the service rate in the node. It makes sure that the arrival rate of real time packet can be handled immediately and packet delay can be minimized due to queuing. The remaining bandwidth is reserved for non-real time data transmission. Also, the buffer size for each queue is depended on the input rate and the output rate in the system. The output is controlled by the network controller and assumed that the non-real time data is following the Poisson distribution.

5.2.2 Real time Control Model

In first stage, service model separated non-real time and real time data into two queues. Base on the previous classification of data source, real time control model is used to investigate the bandwidth utilization for each queue in the system. It uses RTP/RTCP protocols to handle the packet switching network. One of purpose of real time control model is to handle feedback information and control the data rate in the network. The diagram of the real time control model is illustrated in Figure 5-2. Two main measurement parameters are used to control bandwidth utilization for different types of application 1. Generation rate of RTCP message. 2. RTCP message content.

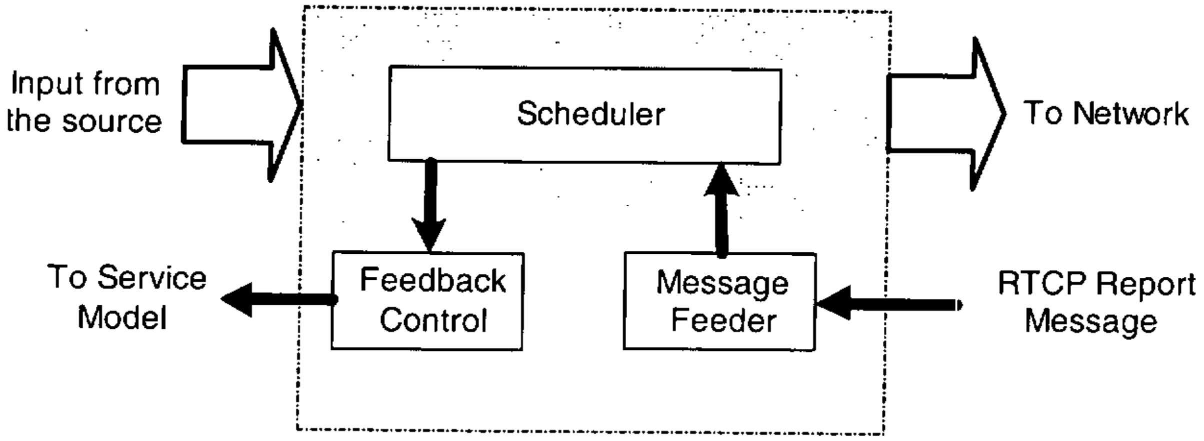


Figure 5-2 Real time Feedback Model

At certain time interval, each terminal, which is included in the connected session, receives RTCP message. Those RTCP message contains status of network and provides useful information in the session such as RTP timestamp, inter-arrival jitter, and delay in last sender report, etc. After receiving the message, the real time controller determines the traffic loading based on the equation describe in [23]. The result given out from the feedback controller determines the ratio of real time and non-real time data traffic and the time to place such information into actual transmit buffer. The relationship between the voice and data ratio is calculated from the following equations.

To control the non-real time data rate on the network, the input data rate of the network should follow Poisson distribution:

$$\lambda_d = \frac{\lambda}{\rho + 1} \tag{5-3}$$

where λ is the maximum input rate on the channel, ρ is the total loading on the network and λ_d is the non-real time data input rate.

On behalf of adjusting non-real time data rate, the other issue is the flooding of RTCP message in a session. RTCP message consumes a certain percentage of bandwidth for reporting if huge of terminals in the same session. As a result, generation of RTCP message should also be in consideration in order to prevent flooding of control message. In feedback control mechanism, it adopts dynamic RTCP message rate for better bandwidth utilization.

The philosophy of dynamic RTCP message rate is that when channel, within the session, is stable, it does not require RTCP message to reflect the channel state frequently. But, when the received terminals find sudden long delay in RTP packet reception in the destination and the QoS requirement of the application is not met, RTCP message will increase its reporting rate to reflect such change to the transmitter so as to reverse more bandwidth for real time data over the network. Until transmission delay goes down to the threshold, which is defined in each connection setup, the rate of RTCP message generation will be decreased. The generation rate of RTCP message is valued according the network stability. And the relationship between RTCP and data ratio should be:

$$\lambda_c = \rho\lambda_d + C \tag{5-4}$$

where λ_c is the arrival rate of control message in receiver and C is the threshold of control message rate

5.2.3 Channel Model

The task of channel model is used to model traffic loading in the network. It can be use to measure traffic loading on the network. In this model, it classifies three types of data in the channel – control message (RTCP), voice (RTP) and data (TCP and UDP). Control message (RTCP) contains the information to reflect the state of terminals within the same session. Refer to the RTCP specification, bandwidth utilization of RTCP message in a session should not be greater than 5%. It has to make sure that control message only utilizes small amount of bandwidth and maximize bandwidth for voice and data transmission. Regarding to the channel model, there are two possible reasons that make channel unstable

1. A large amount of users joining into the session within smaller time interval. As a result, it increases RTCP message-generation rate. The rate of the message is increased exponentially because each terminal within the same session sends a multicast message to other terminals. Every terminal joining in this session receives RTCP report. It consumes huge amount of bandwidth to pass control message and cannot provide sufficient bandwidth for another types of data transmission.
2. The loading of traffic increases suddenly. It is due to a huge amount of data "pumping" into network within smaller time interval such as file transfer application or video on demand application. As a result, congestion and packet (including voice and data) drop occurs frequently. Regarding to the TCP connection based data transmission, re-transmission of loss data occurs frequently and delay experienced from the destination will be increased.

Regarding on determine traffic loading on different types of message transmission, assume λ_v be the voice packet arrival rate, L_v be the voice packet length, C be the channel capacity, β be the loss probability, t_m be the measured round trip delay which obtains from RTCP message and t_x be the expected round trip delay. Traffic loading can be defined as :

$$\rho = \frac{\lambda L_v}{C(1-\beta)} x \frac{t_m}{t_x} \tag{5-5}$$

For purely non-real time traffic, the traffic loading is

$$\rho_d = \frac{N_d \lambda_d L_d}{C} \tag{5-6}$$

In purely real time traffic, since the real time traffic is modeled as a MMPP model, the voice traffic loading is:

$$\rho_v = \frac{N_v L_v}{W_v C} \tag{5-7}$$

For the RTCP control message, it is non-real time packet, but the rate of RTCP message is controlled by two factors - number of terminal in the session and network stability. So, here is the relationship in between the rate of RTCP message generation and traffic loading:

$$\rho_m = (N_v - 1)\frac{L_m \lambda_m}{C} \tag{5-8}$$

From above calculation, the total traffic loading on the network is:

$$\begin{aligned} \rho &= \rho_m + \rho_d + \rho_v \\ &= (N_v - 1)\frac{L_m \lambda_m}{C} + N_d \frac{L_d \lambda_d}{C} + N_v \frac{L_v \lambda_v}{C} \end{aligned} \tag{5-9}$$

For the network stability, it assumes that the RTCP control message should not be greater than 5% of the total bandwidth. Besides, in order to control voice quality, instead of controlling non-real time packet by the service model, the generation rate of RTCP control message is another factor to indicate the stability of channel. The service model counts the received rate of RTCP control message and makes sure that it does not validate the ratio of RTCP control message in channel model.

5.3 Simulation on Real time Feedback Control Mechanism (RFCM)

5.3.1 Introduction to Network Simulator (NS)

Network Simulator (NS) [24] is an object-oriented simulator, written in C++, with an Object-riented Tool Command Language (Otcl) interpreter as a front-end. The simulator supports a class hierarchy in C++, and a similar class hierarchy within the OTcl interpreter. The two hierarchies are closely related to each other.

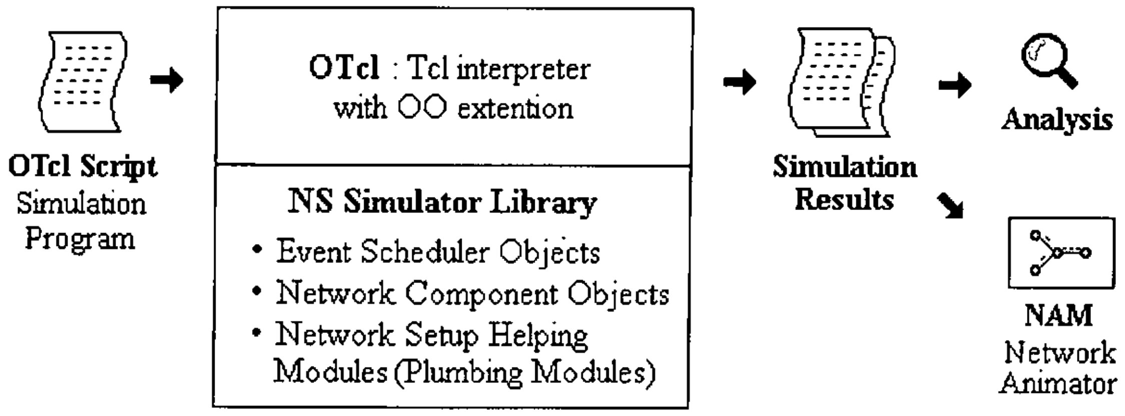

OTcl Script
Simulation Program
OTcl : Tcl interpreter with OO extention
NS Simulator Library
• Event Scheduler Objects
• Network Component Objects
• Network Setup Helping Modules (Plumbing Modules)
Simulation Results
Analysis
NAM
Network Animator

Figure 5-3 User's View of NS

As shown in Figure 5-3, in a simplified user's view, NS is OTcl script interpreter that has a simulation event scheduler and network component object libraries, and network setup (plumbing) module libraries (actually, plumbing modules are implemented as member functions of the base simulator object). To setup and run a simulation network, user should write an Otcl script that initiates an event scheduler, set up the network topology using the network objects and the plumbing function in the library, and tells traffic sources thee time start and stop transmitting packets through the event scheduler.

Figure 5-4 shows the general architecture of NS. In this figure a general user (not an NS developer) can be thought of standing at the left bottom corner, designing and running simulations in Tcl using the simulator objects in the OTcl library. The event schedulers and most of the network components are implemented in C++ and available to OTcl through an OTcl linkage that is implemented by tclcl. NS is formed by gathering all the components together, it is a extended Tcl interpreter with network simulator libraries.

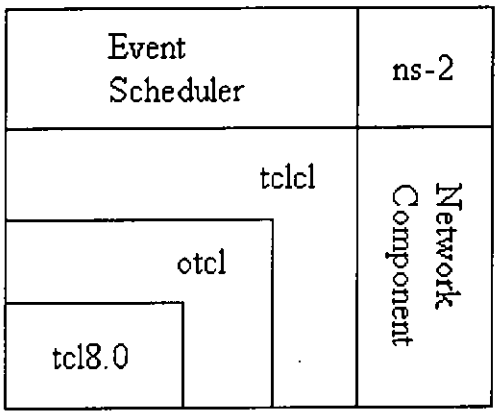


Figure 5-4 Architectural View of NS

5.3.2 Simulation on Two Types of Network

The simulation of feedback control mechanism uses network simulator (NS) to analysis multiplexing of voice and data packet. Delay on the multiplexing voice and data packet with different traffic loading is studied. In the simulation, some initial parameters is outlined as below:

1. Channel capacity is 5Mbps
2. Assume talkspurt and silence ratio is 6:4.
3. Voice packet length is 48 bytes and data packet length is 64k bytes
4. Voice sampling time is 8kbps

In this simulation, two types of network topologies is choose to analysis the performance on Real Time Feedback Control Mechanism (RFCM). In first network topology, there is only one link connected between source and destination. The second topology uses four terminals and each terminal connects to two sources/destinations. These two network topologies are built to verify the functionality of RFCM in such types of basic network topology.

First Network Simulation

<u>Network Topology Design</u>

Figure 5-5 is a simple network topology. Node 1 is a purely non-real time data source. The data is transferred from node 1 to node 4 using TCP protocol. Real time data, which carried by RTP protocol, is generated in node 6, 8, 0, 7, 5 and 9. The real time RTP packet transmits from node 6, 8 and 0 to node 7, 5 and 9. Node 2 and 3 connect together to route the packets from sources to destinations or reverse. The feedback control mechanism is implemented in these nodes. Besides, RTCP control message is generated and returned to the node 2 to reflect the status of network.

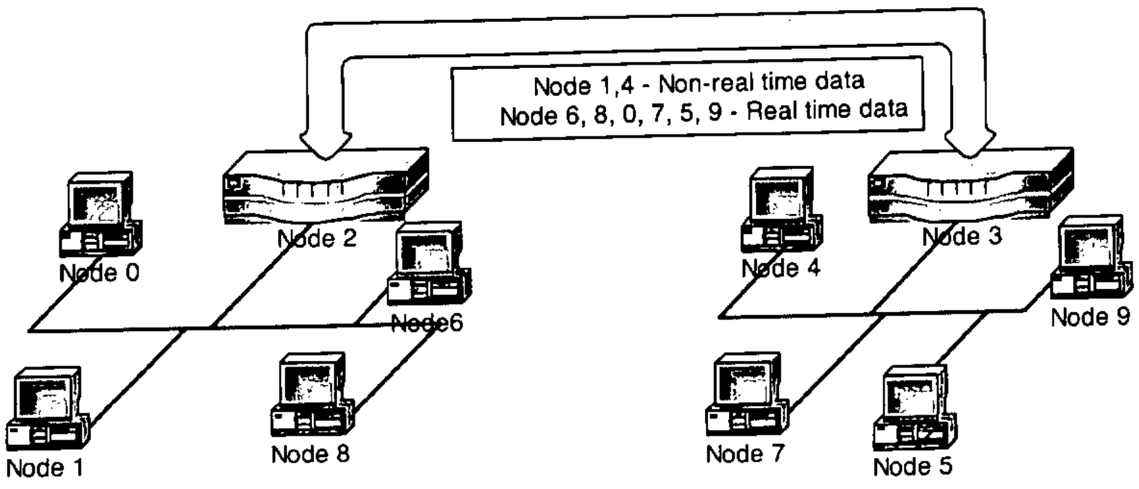

Node 1,4 - Non-real time data
Node 6, 8, 0, 7, 5, 9 - Real time data
Node 0
Node 2
Node6
Node 1
Node 8
Node 4
Node 3
Node 9
Node 7
Node 5

Figure 5-5 Network Topology

The rate of RTCP message can use to reflect loading on link 2-3. If the rate of RTCP message is increased, it means that traffic loading is increased in this link and indicates that the link becomes unstable. To maintain the quality of service on real time application, node 2 has to decrease its scheduling rate of non-real time data. As a result, the stability of this link is maintained.

Result Description

Figure 5-6 is the result on the traffic loading with the time using feedback control mechanism.

Initially, node 1 occupies about 3Mbps bandwidth on link 2-3 to transfer non-real time data to node 4. At the same time, node 6 starts to transfer real time data to node 7 at time 0s. It occupies about 500kbps bandwidth.

After 200s, node 8, 0, 5 and 9 join into the session. Node 8 and 0 generate real time data to node 5 and 9 respectively. In Figure 5-6, channel capacity against time, the real time traffic is increased to occupy 2Mbps bandwidth. A sudden changed in the bandwidth utilization on the real time data forces non-real time data rate to decrease. At 800s, real time data rate remains constant.

In node 2, feedback control on node 2 has two parameters to maintain real time data 1. rate of RTCP, 2. RTCP message content.

In Figure 5-7, it shows that jitter on the real time data traffic is improved by reserving more bandwidth. By statistical analysis, the approximated curve is plotted. Packet delay and jitter can improved 15% of jitter effect on channel.

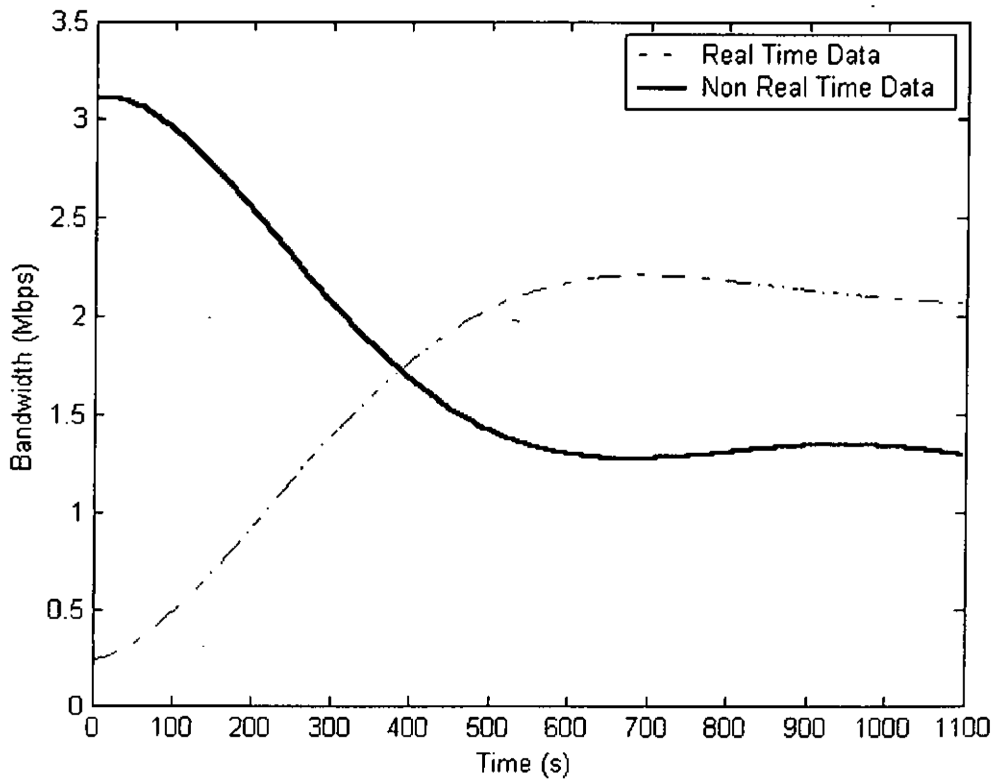


Figure 5-6 Channel Capacity against Time

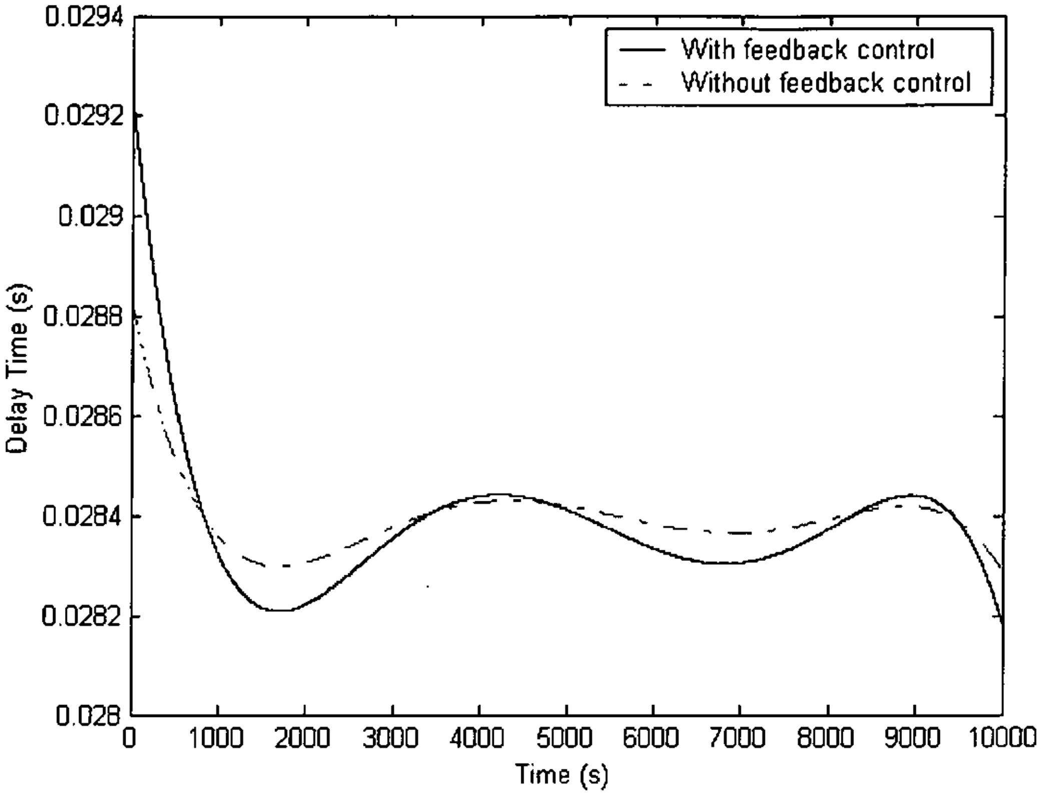


Figure 5-7 Delay against Time

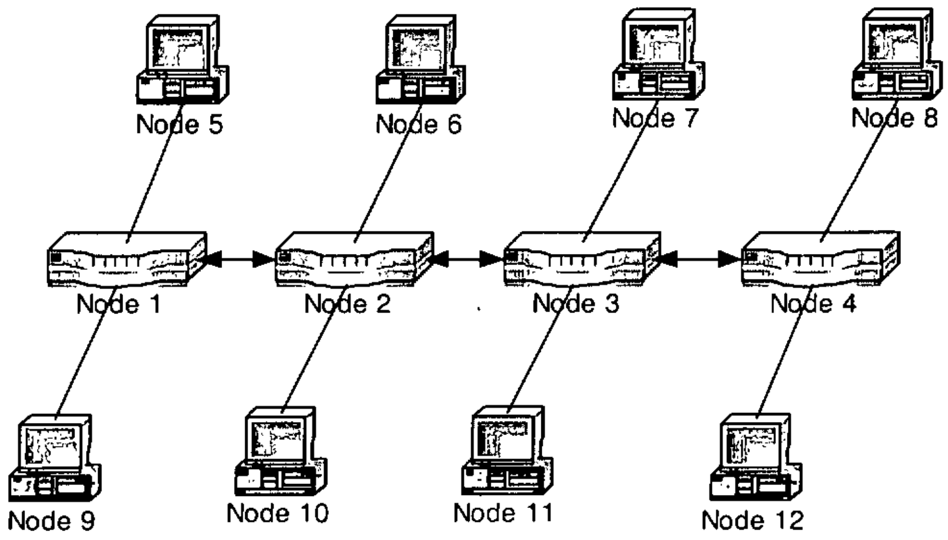


Figure 5-8 Bus Network Topology

Second Network Simulation

Network topology design

Figure 5-8 is the second network topology; all nodes are shared with a single transmission media. This topology is more complex than the first one and several feedback control mechanism is implemented on different nodes.

Result Description

In this topology simulation, node 5 and node 8 generate real time data to node 11 and node 10 respectively. On the other side, non-real time data is generated in node 6, 7, 9 and 12 in both directions. There are two non-real time data sources (node 9 and node 12) and two real time data sources (node 5 and node 8). The feedback controller is implemented in node 1, 2, 3 and node 4.

Link 1-2 has two data streams, real time and non-real time data. In Figure 5-9, bandwidth utilization is decreased for non-real time data stream at the beginning. On the other side, the data stream for real time is increased. Due to the symmetry of link 1-2 and link 3-4, the result of link 3-4 and link 1-2 are the same. The result shows that real time data get more bandwidth for transmission by applying feedback control mechanism.

In link 2-3, there are four data streams, two real time data and two non-real time data. Traffic on this channel is doubled comparing with link 1-2. Figure 5-11 shows that the link is fully utilized by two real time data streams and there is no non-real time data traffic in this channel. Also, in Figure 5-10, it shows delay on this two real time data streams and jitter is reduced on this link by apply RFCM. In this simulation, it shows that feedback controller can be applied in bi-direction system to provide traffic management.

As a result, by adding feedback control mechanism in node 1, 2, 3 and 4, only non-real time data is queued and real time data is passed through link 2-3 directly without queuing in the system to introduce delay. It shows that, QoS on multimedia application can be maintained since real time traffic get higher priority for transmission under real time adjustment on the bandwidth utilization.

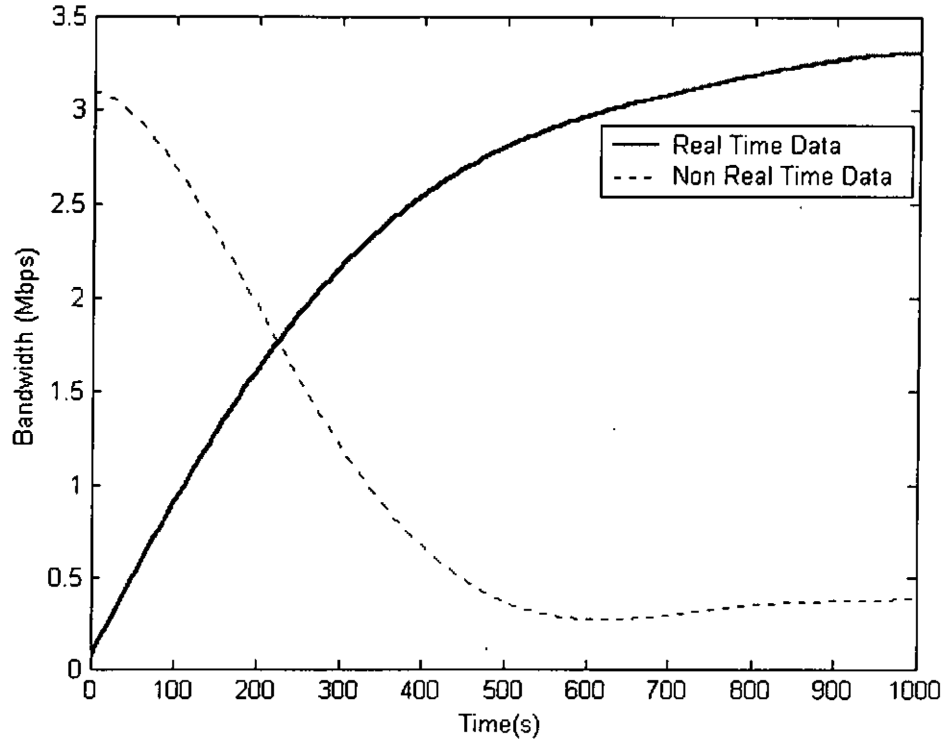


Figure 5-9 Channel Utilization against Time

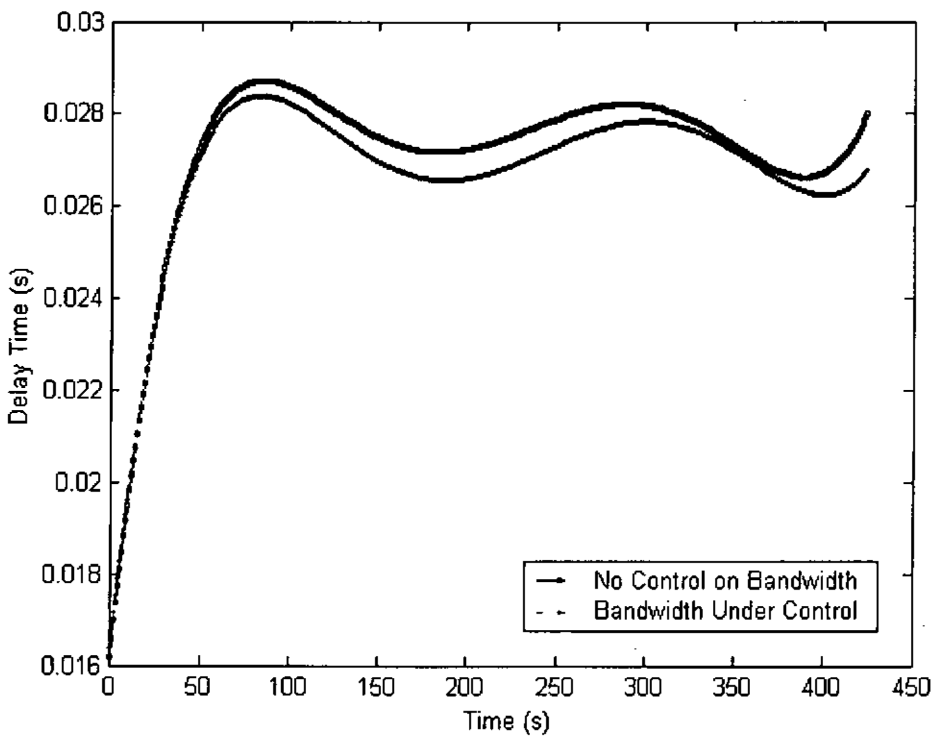


Figure 5-10 Delay against Time

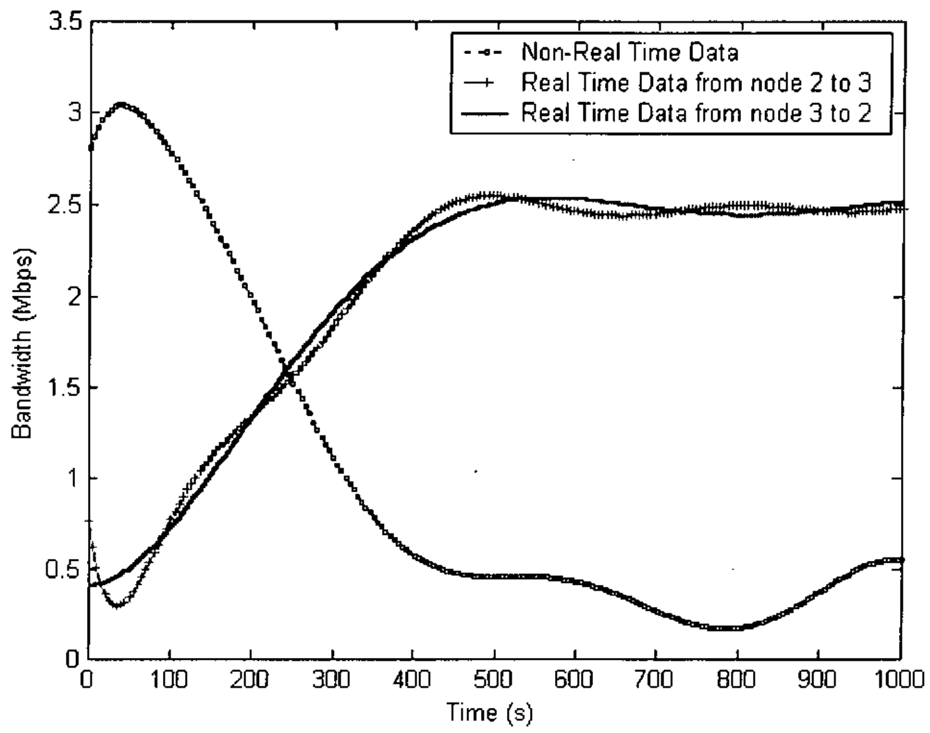


Figure 5-11 Bandwidth against Time on Link 2-3

The result shows that non-real time data is under control by the feedback controller on node 1 and node 4. At node 2 and node 3, non-real time data is queued such that the channel can reserve more bandwidth for real time data transmission. For further enhancement, feedback controller can provide different priority level for different types of service.

5.3.3 Result Analysis

Comparing with Direct Adjustment Algorithm (DAA) and Real Time Feedback Control Mechanism (RFCM)

In Direct Adjustment Algorithm (DAA), it shows that if the algorithm is applied to the terminal, it can utilize more bandwidth because it reduces collusion rate by additive increase and multiple decreases the data rate. But it does not change the ratio of real time and non-real time data in the network. It only achieves high bandwidth utilization on the network without consider the nature of data.

In real time feedback mechanism, it provides high bandwidth utilization and QoS consideration with different types of data. If compare those two algorithms, the connection has to increase in feedback control model and take as the average value for each test.

Delay Comparison in Two Different Topology

The delay without feedback control is smaller initially comparing with the system adopted feedback control mechanism. It is because they have different initialization values. Initially, the feedback control need pre-defined initial value to determine the transmission rate. As the controller cannot determine whether the channel is busy or not at the very beginning, it assumes that the channel is fully occupied. Until the controller gets the status of the channel, it starts to adjust its transmission rate according to the channel condition. In figure 5-10, initial value of two systems is set to zero. In figure 5-7, the initial value of delay with feedback control is 0.9 and without feedback control is 0.5

Since there is no feedback control in the network, the channel cannot maximize its channel utilization and delay at the beginning is high.

By applying the feedback control in the system, the channel can utilize more bandwidth and the traffic rate for real time and non-real time is under control, it may achieve higher bandwidth utilization. On the other side, the feedback controller is sensitive to the following factors:

1. Delay of message reporting – Propagation delay on the channel makes the feedback controller cannot get the actual status in the session.

2. Scheduling rate of packet –The scheduling rate affects the variation of bandwidth utilization. It seems that if the scheduling rate is too high, the channel may become various.

As a result, it should be further studies on these two relative factors so as to stabilize bandwidth utilization.

Delay With and Without Feedback Controller

In Figure 5-10, the delays between with and without feedback control are relatively small because it is the average delay in the system. The figure shows that packet delay is improved with feedback controller. In Figure 5-11, it can be proved that real time data gains more bandwidth for transmission after a certain amount of time period. One the other side, non-real time data releases its bandwidth. The channel utilization is remained constant and only the adjustment on the ratio in between real time and non-real time data.

5.3.4 Result Conclusion

From the simulation, it seems that feedback control mechanism provides real time dynamic adjustment on channel bandwidth arrangement. It concludes that:

- Two parameters are used to adjust real time and non-real time data traffic ratio - RTCP generation rate in the same session and RTCP message reporting.
- Controlling RTCP generation rate can reduce flooding of RTCP message while

huge number of terminals is joined into the same session.

- Jitter in the network is improved by increasing scheduling rate on real time queue in service model.

Comparing with the traditional method, feedback control mechanism provides real time channel bandwidth adjustment according to different QoS requirements on multimedia communications. The limitation on feedback control mechanism is delay time for the RTCP feedback message. In long delay, the RTCP feedback report may not reflect the real situation of the network. If a long delay is happened in unstable channel, it requires further analysis to estimate feedback message in order to maintain the desired bandwidth for real time traffic. One of the solutions is to increase the RTCP message feedback rate, which increasing traffic loading in RTCP message reporting and turn out decreasing bandwidth for actual data transmission. It should take further analysis of this tradeoff and delay on feedback message that affects the performance of the controller.

CHAPTER 6

Implementation Issues on Internet Telephony

In previous chapter, it provides some theoretical analysis on voice characteristic and feedback control mechanism to provide QoS management in packet switching network. In this chapter, practical implementation is studied for Internet Telephony technology.

6.1 Communications Common Platform (CCP) Architecture

Internet Telephony gateway, it can be separated into hardware part and software part to analysis the system. Hardware part concerns the flow of data in the network. It includes buffer size allocation, processing power, board-to-board interface, data bus bandwidth, number of peripheral can be supported, etc. Refer to the OSI model, most of hardware design is implementing layer 1 and part of layer 2 of OSI model. Software part concerns on the management of data flow in the system including real time operating system supporting, board initialization, signaling, routing, process control, etc. The range of software design is more flexible depending on what types of service should be provided in the system. Besides, operating system is a major part of software design to provide efficient task scheduling for application management and transparent hardware design to provide upper level application support.

In this chapter, Communications Common Platform (CCP) development concept is given to show how to design a flexibility platform to integrate several types of applications, such as PSTN and VoIP internetworking development.

6.2 Hardware Design on Communications Common Platform (CCP)

The basic element of hardware design is the semiconductor chips with other active components such as resister, capacitor, etc. Hardware designer has to choose different types of chips to perform different types of function. In designing a Communications Common Platform gateway, some basic components should be involved as standard parts:

1. Embedded Communication Processor – It provides central processing unit in each system boards. The service it provides including memory management, protocol encode/decode, I/O control, etc. The performance of system is determined by its processing power. Also, the functionality of the system is determined by the feature of the processor. In communications field, there are two types of processor

- control processor and data processor. Control processor is used for traffic, memory, peripherals and co-processor management on the board. Some of protocol is handled by control processor such as Signaling System 7 (SS7) and Simple Network Management Protocol (SNMP). On the other side, the main task for data plan processor is for data processor including packets segment and reassembly, classification, queuing, forwarding and address lookup, etc. The processing power of data processor should be large enough to handle huge data rate traffic.

2. Common Interface Unit – The purpose of this interface is to interconnect different types of board together to connect different boards with different functions. They share several types of resource in a system such as board memory, information and traffic. Using common interface for system design gives more flexibility for developer in system integration by using separated functional boards. Some common standards for board-to-board communication is provided such as Compact Peripheral Component Interconnect (PCI) and H.110. H.110 uses for data stream passing from one board to another and Compact PCI uses for memory sharing for board's control.

3. Digital Signal Processor (DSP) for voice encoder/decoder - The function of DSP is to encode and decode voice data according to some of voice coding standard such as G.723.1 and G.729.

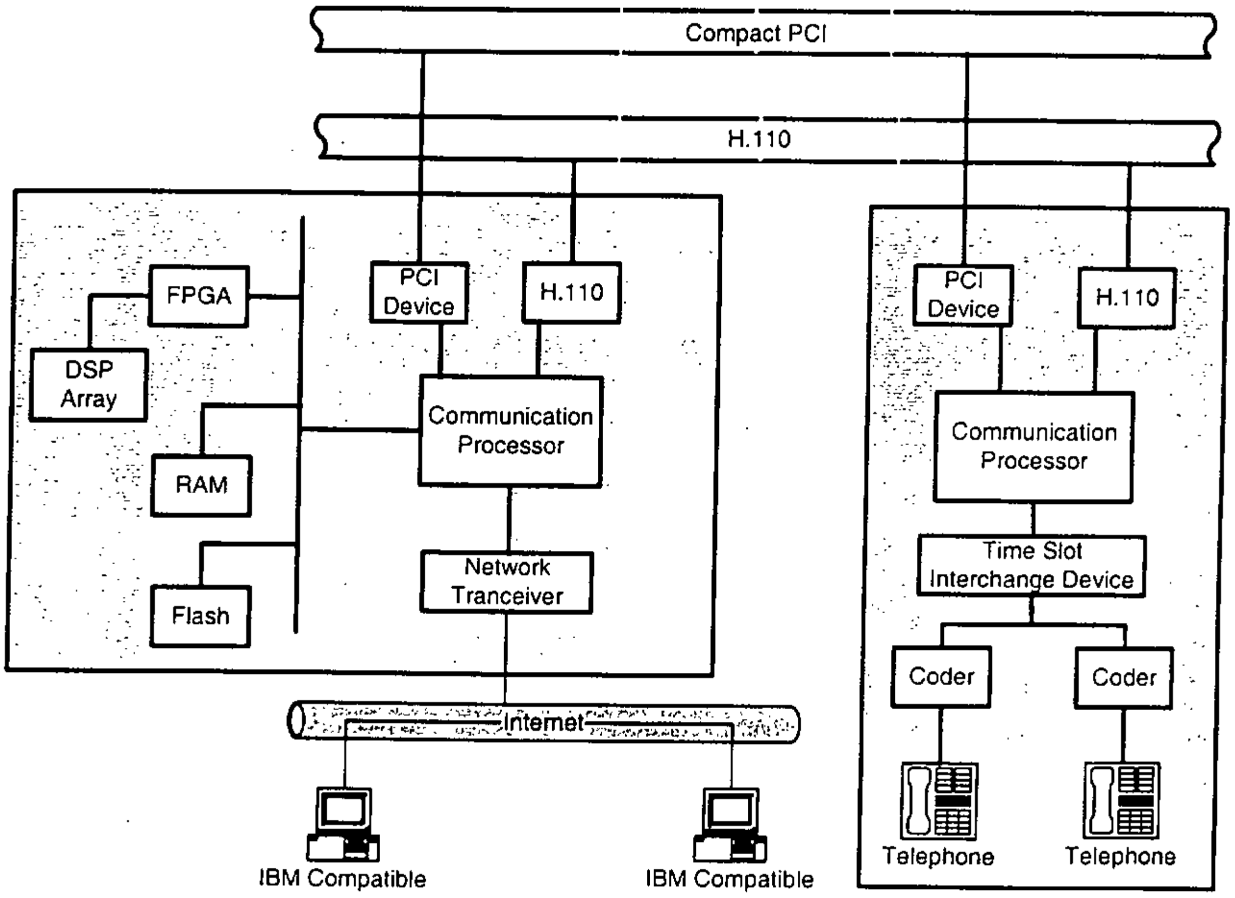


Figure 6-1 Hardware Design Architecture for VoIP Equipment

Figure 6-1 shows an example on how to use CCP concept to design VoIP gateway. The gateway interconnects both of circuit switching and packet switching network together to provide voice communications. There are several types of boards with different functionality interconnected together using some of common interface standard such as Compact PCI and H.110. The system can be divided into three parts – analog line card, DSP and Internet array board and T1/E1 WAN interface board.

6.2.1 Analog Line Card Architecture

It is a traditional codec-based analog line card that interfaces analog plain phone with common platform backplane. It converts analog signal into digital signal using PCM (A-law or μ-law). Also, it has Compact PCI and H.110 interface for PCM data transfer between boards. The board should contain following components:

1. Communication Processor - It is a central processing unit to control all resources in the board such as memory, input and output device, etc. Currently, most of the embedded processor can provide this function such as Motorola MPC860, MPC8260 and C-5, Intel IXP 220 and IXP 225. They have different processing power to satisfy different analog line card performance requirement.
2. Compact PCI device - It supports industrial Compact PCI standard. It acts as a bridge between local bus and Compact PCI bus. The Compact PCI bus is used as an interconnect mechanism between highly integrated peripheral controller components, peripheral add-in boards, and processor/memory systems.
3. H.110 device - It gives a low-latency communications traffic between the boards within the computer, independent of computer's I/O and memory buses. The voice data can directly transport between boards that can reduce board-to-board transmission time.

4. Codec, DTMF and SLIC controller - These parts are the basic components of the standard analog line card. Codec obeys A-law and μ-law and used for A/D and D/A conversion in between SLIC and backplane bus. DTMF is used for DTMF tone decoding in every in-bound channel. SLIC is used for interfacing the 2-wire subscribe line with the board 4-wire side. Besides, it has following parts to line card design: battery feed, AC impedance matching for loop terminations, hooks witch detection, adjustable transmit, receive and, transhybrid gains, and fault indication.

6.2.2 DSP Array and Internet Board Architecture

The function of this board is used to compress PCM signal, transmit/receive Internet data and provide a Compact PCI and H.110 interface for board-to-board communication. Figure 6-2 shows the simple structure of DSP array board with 64 DSPs. The detail design and implementation of the high density of DSP array board will be discussed in next chapter. Basically, it contains following components:

1. Digital Signal Processor (DSP) farm - It is an array of DSPs that provide compression function for PCM signal. Increasing number of DSPs in a system can increase the supporting voice channel. Currently, most of DSP can handle 3 to 4 voice channels. It is challengeable to integrate large number of DSPs in a system

since the numbers of DSP in the board determines the number of voice channel that it can support.

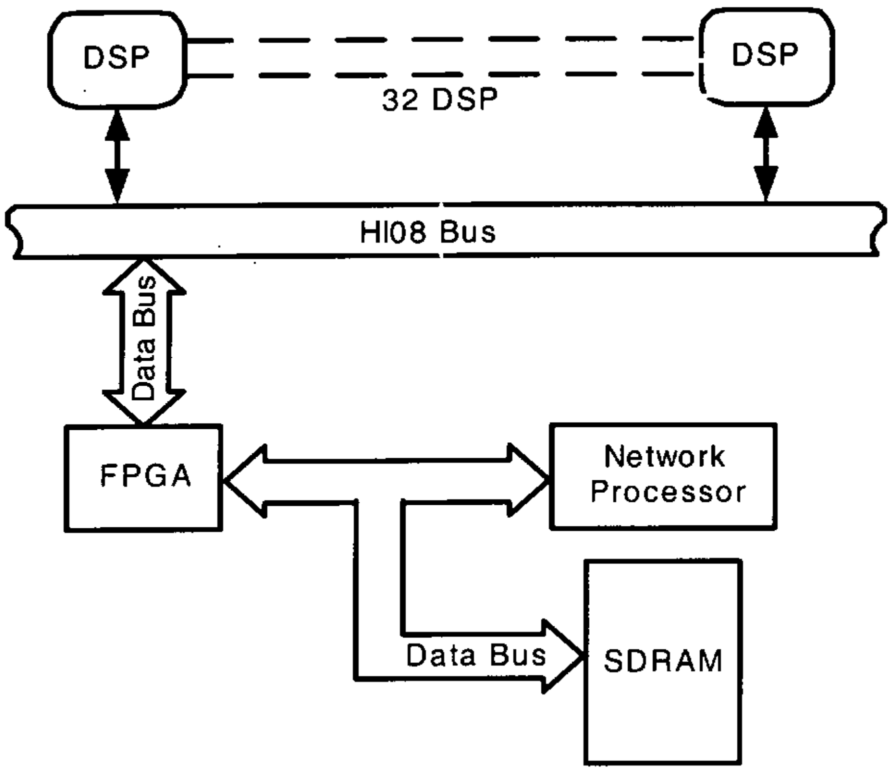


Figure 6-2 Architecture of DSP Array Board

2. Data Hub – It is designed by a Field Programmable Gate Array (FPGA) that interconnects network processor to large amount of DSP. The function of data hub is to provide a control path between network processor and DSPs and also data path between memory and DSPs. It acts as the interface between DSPs' bus and network processor bus, DMA peripheral of network processor for data transfers between DSPs and processor. In addition, processor can access all of the DSPs register via data hub. The advantage using data hub is to simplify the design of a high capacity system.

3. Common Interface Device - It provides an interface for board-to-board communication. It follows the popular industrial standard - Compact PCI and H.110. Network processor does the initialization and configuration of such device.

6.2.3 T1/E1 WAN Interface Board

The Dual T1/E1 WAN Interface Board supports two independent channels of T1/E1 by T1 single-chip transceiver and E1 single-chip transceiver. The network processor is used for board control. Figure 6-3 shows the block diagram of Dual T1/E1 WAN Interface Board and following is the list of some main components:

1. Integrated Communication Processor - This is a board controller that used to control the entire device such as PCI Bridge, T1/E1 transceiver, etc. Also, the processor has its own SDRAM and flash that provide a memory space for program execution and buffer storage.
2. T1/E1 Transceiver - The transceiver connects with the host processor and T1/E1 line interface. It provides frame synchronization and TDM clock to the line. Data will move from H.110 interface through T1/E1 transceiver in 1.544Mbps or 2Mbps.
3. Compact PCI Interface Bridge – The function of the bridge is to connect the host processor and PCI interface to provide board-to-board communication.

4. H.110 Interface Bridge - This interface should provide a time-slot interchange mechanism and TDM line in the backplane such that voice data could transfer through this interface from one board to another board.

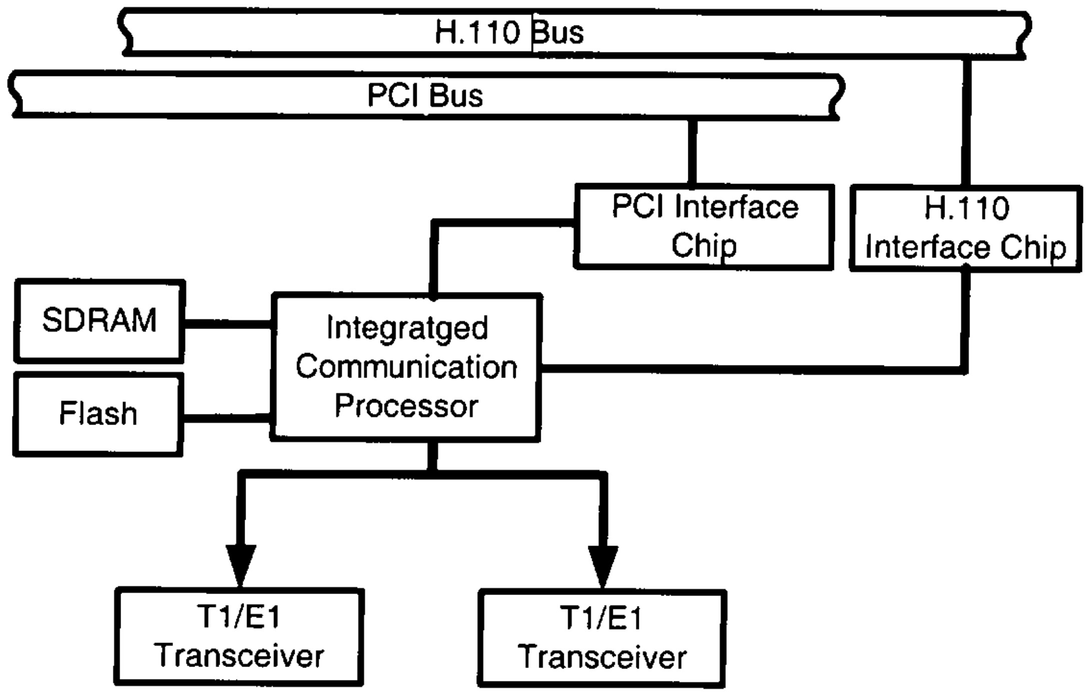


Figure 6-3 Architecture of T1/E1 Line Card

6.3 Software Design on Communications Common Platform (CCP)

Software design in multimedia communications has to consider quality of service for different architectures. Layering design is proposed in Communications Common Platform (CCP) in order to transparent hardware dependence. Besides, different types of applications have its QoS requirement in CCP. Software design concept for the CCP is separated into three directions - service provider axis, QoS handler axis and layering designer axis. Those three axes are considered as three-dimension software architecture used for applications running in Communications Common Platform (CCP) and form three-separated plane. The three planes are quality control plane, service specification plane and layering management plane. This three planes model is used for application mapping by integrated several software model in each plane. Figure 6-4 shows software architecture for CCP. Here is break down of three-dimension model for discussion.

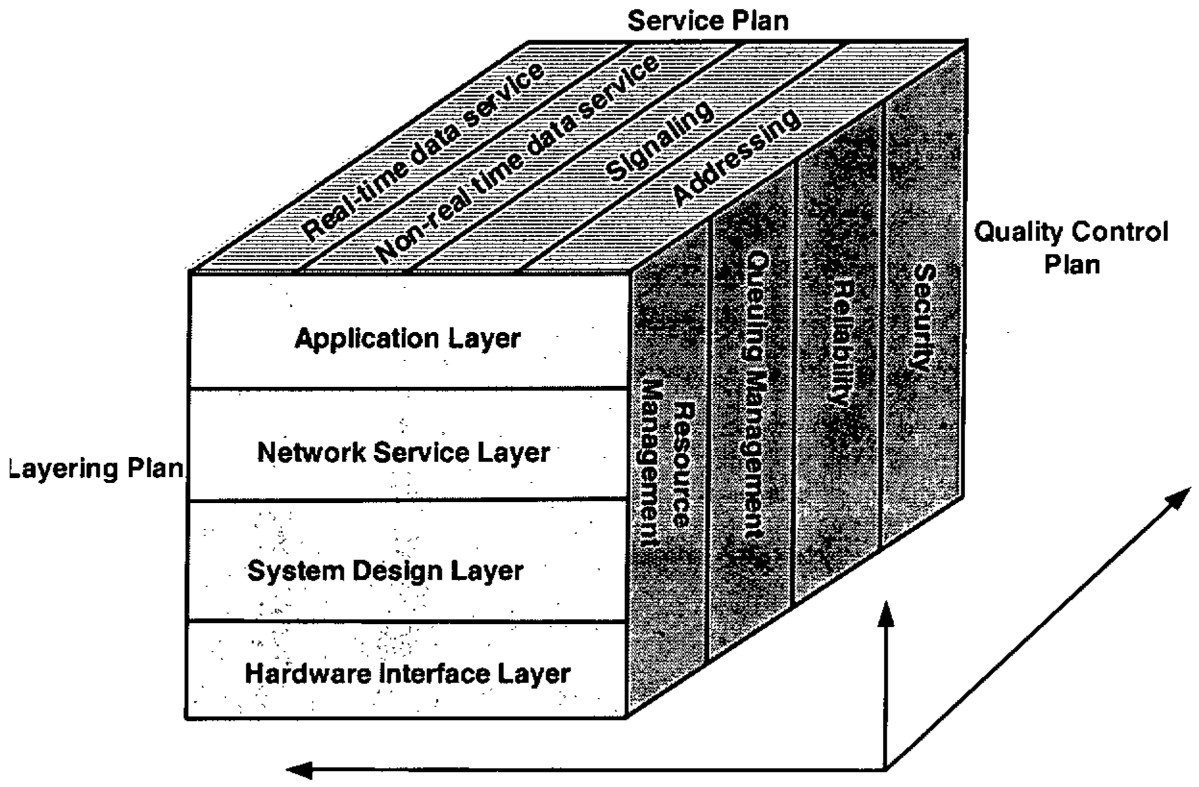


Figure 6-4 Multimedia Application Software Design Architecture

6.3.1 Layering Plane

This plane handles hardware and software implementation to build multimedia communications applications. It consists of four layers - hardware dependence layer, system level layer, network service layer and application layer. Lower layer tasks provide service to upper layer tasks. Through these four layers, one can develop its application by sending request to lower layer. Also, each layer has its corresponding API design to passing request to lower layers and returning response to upper layers. Figure 6-5 shows these four layers.

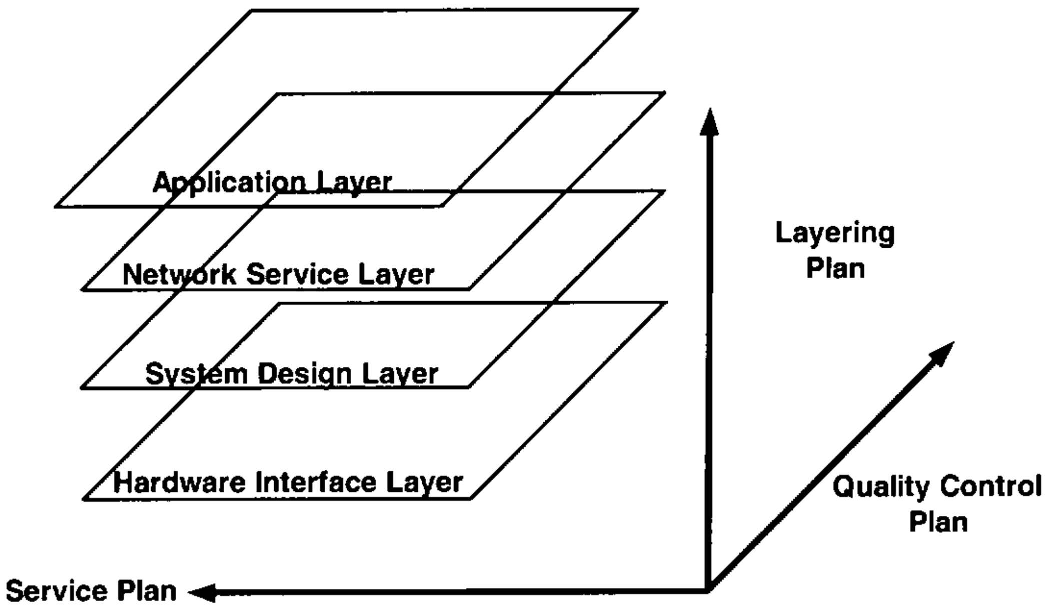


Figure 6-5 Layering Plane

Hardware Interface Layer

The main function of this layer is to provide single board design management. It contains real time operating system to manage hardware configuration of each board such as memory management, chip select on different devices and exception handling routines. These layers contain the following components to provide service to upper layers:

Board Support Package (BSP) for specific board design - Each board has its own BSP to control hardware since different board has their hardware components.

Real Time Operating System (RTOS) - It centralizes all the resource on the board and provides management such as interrupt handling, memory management, and processor control. pSOS and VxWork are the examples of RTOS.

Diagnostic software for self-functional test - Each board contains its self-testing program to guarantee the basic functionality of the board. It is a simple testing program and execute without operating system support for hardware verification.

System Design Layer

Most of the network equipments have to adopt some common interface standard such as Compact PCI and H.110 to interconnect different types of functional boards in order to form a complex system solution for a particular network solution. The common interface standards are used to provide board-to-board communications including control message or data stream passing. The purpose of this layer is used to handle such information passing within different boards. On the other side, it also tries to transparent hardware configuration to the upper layer such that upper layer do not need to know the hardware detail.

Network Service Layer

Network service layer provides networking service to upper application layer such as driver of protocol stack. The application layer can use the API provided by the system design layer to develop networking service. This layer consolidates the necessary protocols and forms a group of driver. For instance, H.323 protocol including signaling (Q.931), RTP/RTCP and channel setup protocol is implemented as a group of driver for upper layer to develop the VoIP or video conferencing application. Instead of implementing protocol stack in this layer, it also can implement different types of network traffic control algorithm for traffic management. This layer provides a certain level of transparency of protocol detail to upper layer. System design layer provides API to upper layer and isolates lower level hardware configuration to the application layer.

Application Layer

Application layer allows network developers to design their applications and provide a user-friendly API for users. In most of gateway, the applications include routing table configuration software, FTP and telnet server software for remote configuration, etc. For instance, one can issues a request to open socket in network service layer for remote access control and get the response from the network service layer to see whether the request is success or not.

6.3.2 Quality Control Plane

In another axis, quality control plane is defined to classify network quality and performance measurement. This plane is tried to monitor and maintain the QoS that the network provided. In order to maintain desired quality of the network, traffic monitoring and control software is a main task of this plane. This plane specifies four network qualities - queuing management, resource management, reliability management and secure management. Figure 6-6 shows the structure of quality control plane

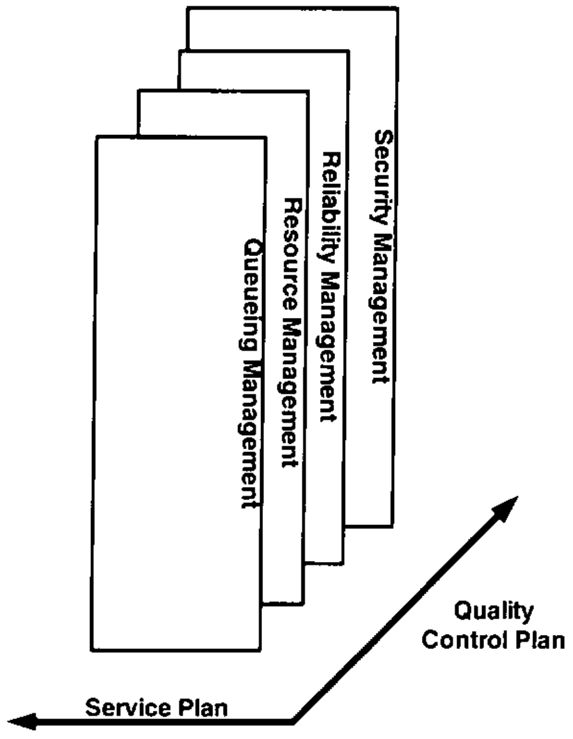


Figure 6-6 Quality Control Plane

Queuing Management

This plane handles network delay, jitter and loss. One can see that different services in different layers have to fulfill different quality requirements. In hardware dependence layer, queuing concerns buffer size configuration, number of queue that can be supported to implement the traffic control mechanism. In system design layer, network performance is determined by system complexity and its limitation such as bus latency and tasks execution time. If system considers as a black box, latency of such system is the time between input data rate and output data rate. In network service layer, it should provide several types of traffic control algorithm to manage data flow such as real time feedback control model as proposed before. Based on the requirement of application, network service layer should provide several appropriate traffic management methods for end user to choose.

Resource Management

Two types of resource in networking software design have to be concern – resource of network and resource inside network equipments. Resource of network is available bandwidth for transmission. It affects QoS management and limits network resource. Resource of network equipments is those components in the system such as buffer size, MIPS of processor and peripherals. MIPS determines capability of packet handling. Task scheduling is needed to prioritize the critical tasks execution. For example, packets using RTP/RTCP protocols for transmission should handle first. On the other side, the availability of peripherals determine what network interface support in the system. For example, Gigabit Ethernet needs GMII or RMII to interconnect with PHY device. ATM needs UTOPIA interface to interconnect with PHY device. This plane provides a structured platform for the developer to maximum system resource.

Reliability Management

Software reliability is one of factors that affect system stability. Most of the failure is due to improper programming, incorrect variable definition and incorrect interrupt handling. Besides, stability of operating system design is another factor that affect the reliability of system in software development. Most of system uses real time operation system (RTOS) in their network equipments because of timing consideration. RTOS can execute the task within scheduled time. During transmitting or receiving packet, system cannot waste too much time in buffering and should be handled as soon as possible.

In order to prevent sudden system halt, the most useful and popular method is tried to use redundant server. Once system fails, it can switch to the redundant server and continues its operation to provide service to the user. But the cost is increased to build such redundant server since the redundant server is idled in most of time. So, another application for those redundant servers is used for load balancing.

Secure Management

Nowadays, network security is a new topic and becomes more important in data transmission. Most of transaction and e-commerce activities in Internet requires highly secured data encapsulation. Currently, some protocols and methodologies are used for data security transmission such as SSL, private key and public key, classification in packet routing. The level of security depend on what types of security protocol is used in the network. In network service layer of layering plane, some of security protocols can be implemented such that applications can select their security level. In lower layer, the security policy can be implement in system design layer in order to provide security service to upper layer.

6.3.3 Service Plane

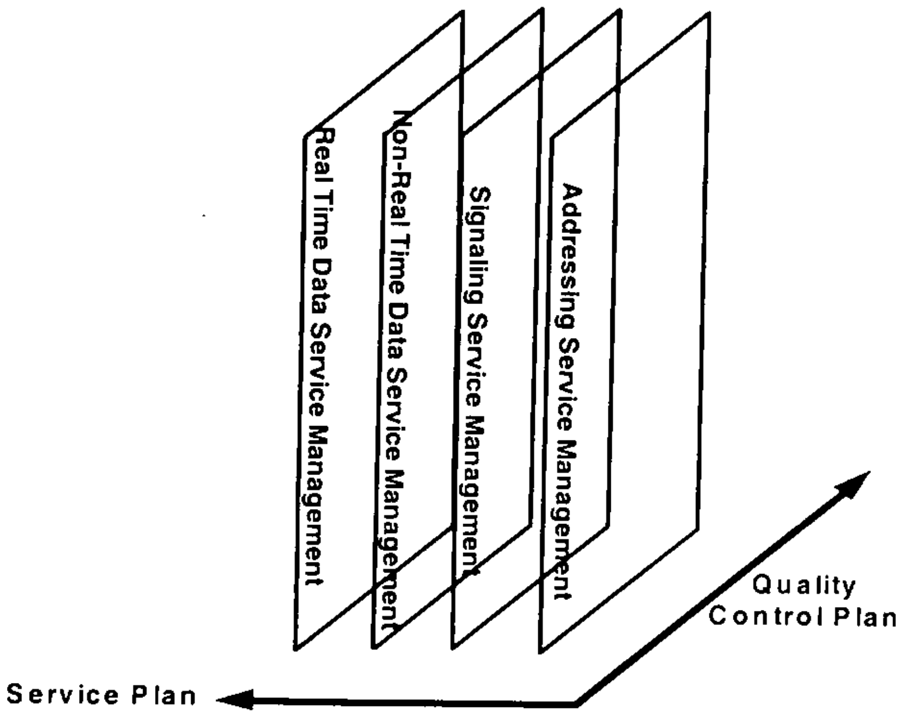

Real Time Data Service Management
Non-Real Time Data Service Management
Signaling Service Management
Addressing Service Management
Quality Control Plan
Service Plan

Figure 6-7 Service Plane

The purpose of service planes is to define what service provided in the system and separate them into individual modules. Different services provided in the system for multimedia applications require different types of QoS support. In order to handle those of network applications, it is necessary to group the services into separated modules.

These service planes specify what requirement should be provided for each layer. Four sub-layers are defined - real time data service management, non-real time data service management, signaling service management and addressing service management. Ones can create their separated service on different planes. Figure 6-7 shows the service planes structure.

Real time Data Service Management

This plane is to handle time sensitive data. Delay and jitter is the most important factors in system design for real time applications. In hardware dependence level, RTOS is needed to support real time applications. In system point of view, delay comes from packet processing time in network equipments, propagation delay on the transmission line. The performance measurement is needed to know the worst case for data in and out time. Since system may contain several types real time applications, this plane is needed to coordinate different types of applications.

Non-real Time Data Service Management

This plane is to handle the non-time sensitive data. Such data can tolerate a certain amount of delay. But in other point of view, this type of non-real time application is sensitive to packet loss and network collusion. In fact, most of non-real time applications require reliability for data transmission. In this case, this plane should provide service to reduce data loss such as re-transmission of loss packets, error rate reporting, etc.

Signaling Service Management

Other than data transmission, network requires information exchange in between network equipments. This plane is used to manage signaling information such as passing QoS parameter and routing information, connection setup, etc. On the other side, after received signaling information, this plane has to determine what tasks to be executed or just route to other destination for further process. For example, the network equipment has needed to know how to process the SS7 signaling message.

Addressing Service Management

This plane is to maintain address lookup mechanisms for different types of network, such as hash table, and provides address mapping in the system. Different types of addressing scheme are used for some system to identify the target terminal. Addressing management plane determines how packet route to destination and perform table lookup such that system can know how to handle the received packet. For instance, Internet uses IP address to identify its communication terminal; But ATM uses VPI/VCI to specify the dedicated path.

CHAPTER 7

High Density Architecture Design in VoIP Gateway

In previous chapter, the concepts of Common Communications Platform (CCP) hardware and software platform are proposed. For VoIP application, the number of voice channel, that it can support, is determined by the processing power of the hardware. In general, DSPs are used, consider each can handle up to 4 voice channels, in order to increase the number of voice channels, it has to increase the number of DSPs in a system. As a result, high density DSP array board is necessary. In this chapter, a methodology to integrate multiple DSPs in a single board that in 6U form factor is described.

7.1 Design Architecture

There are three main components in the board - Motorola MPC860 communications processor, data hub implemented by Xilinx Field Programmable Gate Array (FPGA) and Motorola DSP56307 digital signal processor. A brief architecture of high density DSP array board is introduced in the previous chapter and more detail on the board design is outlined here.

Introduction to Motorola MPC860 and DSP56307

The MPC860 [25] communications processor is a versatile one-chip integrated microprocessor and peripheral combination that can be used in a variety of controller applications. It particularly excels in both communications and networking systems. The MPC860 is a 32-bit PowerPC implementation that incorporates Memory Management Units (MMUs) and instruction and data caches. It comprised of three modules that each use the 32-bit internal bus: The Power core, the System Interface Unit (SIU), and the Communication Processor Module (CPM). The core is compliant with the User Instruction Set Architecture (UISA) portion of the PowerPC architecture. It is fully static design that has an integer unit and a load/store unit. System interface unit provides power management functions, reset control, PowerPC decrementer, PowerPC time base and the real time clock. CPM contains features that allow the MPC860 to excel in communications and networking products.

The DSP56307 [26] digital signal processor is a 24-bit general-purpose digital signal processor. The family architecture features a modular chip layout with a standard central processing module which supports memory expansion options, different on-chip peripherals, and different package size. It is built on a standard central processing module, which includes data buses, address buses, data arithmetic logic unit, address generation, program control unit, memory expansion, on-chip emulator circuitry and phase-locked loop based clock circuitry.

Two Transmission Modes

The communications processor MPC860 is the controller for the data hub. The data hub acts as the interface between DSPs' 8-bit Parallel Host Interface (HI08) bus interface and MPC860 bus interface. Up to 32 DSP HI08 ports may be connected to each data hub. The data hub also acts as an Independent Direct Memory Access (IDMA) peripheral of MPC860 for transferring data between DSPs and MPC860. In addition, MPC860 can access all DSPs HI08 registers via the data hub. The recommended interface block diagram is shown in Figure 7-1.

There are two main data transfer modes within the data hub - By-pass mode and IDMA mode.

1. By-pass mode - It is mainly used for register configuration, but it can also be used for data transfer, which is backward compatible with the DSP array board. Besides, this mode can provide instance data path without any setup, which is useful for small size DSP code downloading. For register configuration, the General Purpose Chip-select Machine (GPCM) memory controller of the MPC860 is used to configure the internal registers of the data hub (designed by FPGA) and the registers in the DSP. Both groups of register are memory-mapped to the MPC860.

2. IDMA mode - After the configuration, the data link between the DSP and the MPC860 should be setup. In transmit mode, 32 bit data would be transferred from the external memory, for instance SDRAM, to the data hub. Then the hub would unpack 32 bit data into 4 byte sequential data, and send them to the DSP through the HI08 interface. In receive mode, 8 bit data would be transferred from the DSP to the data hub. Then the hub would pack the 4-byte data into 32 bit data, and send it to the SDRAM. The system consists of the Motorola MPC860 communications, the data hubs (designed by FPGA) and the DSP array.

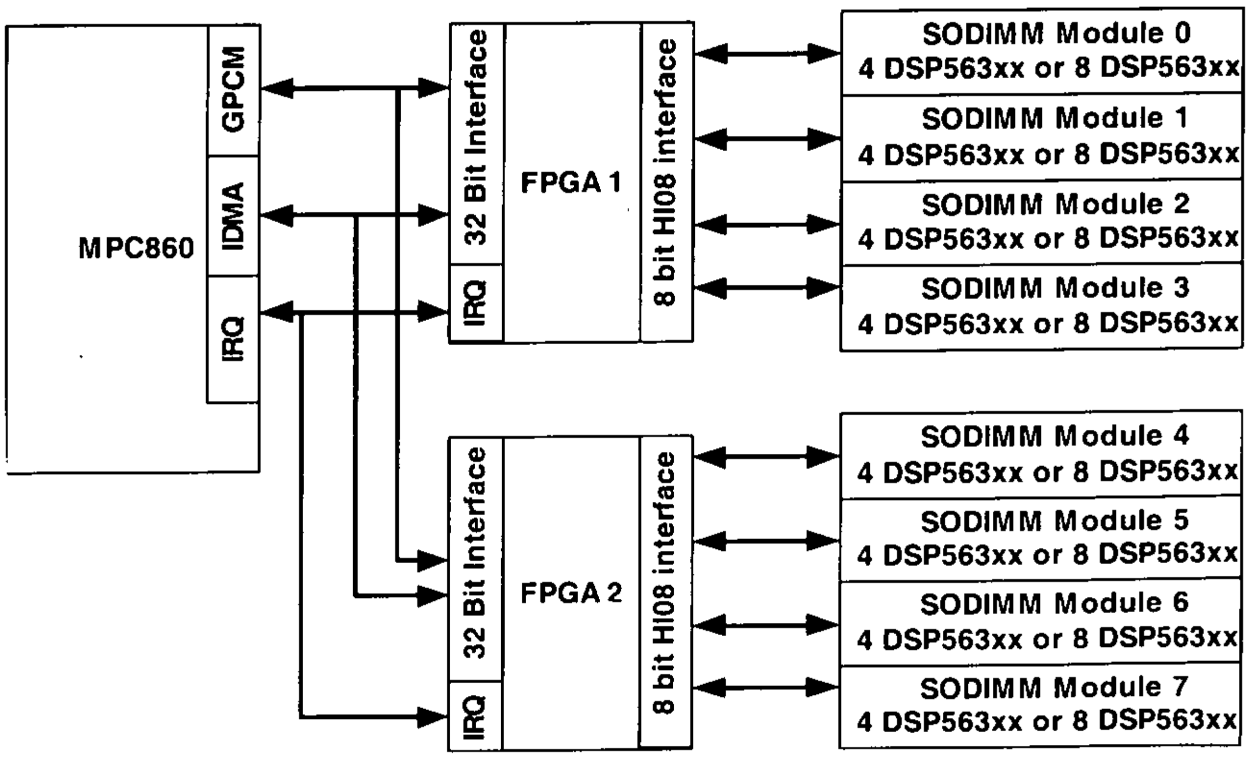


Figure 7-1 64 DSP Configuration Using Data Hub Control

7.2 Functional Description

This interface allows all the usual access of HI08 interface by MPC860 through the memory mapped HI08 registers. The functions include:

1. Download program-using HI08 bus to DSPs.
2. Move data to/from DSPs through HI08 bus.

1. Downloading Program to DSPs Using HI08 Bus Interface

Since all DSPs HI08 registers are memory mapped to MPC860 through 8-bit (General Purpose Chip-select Machine) GPCM memory controller, DSP software can be downloaded from MPC860 to DSPs using the normal HI08 access. The procedure is:

1. Reset the DSP
2. Specify the length of the program to be downloaded.
3. Defines the start address where the program should be downloaded in the DSP's program memory.
4. Write the user program word-by-word to the HI08 Rx register.

After the program is downloaded to DSP, the DSP will reset itself and start program execution from the start address. Several programs download may require if different compression standard is used.

2. Data Transfer Between DSPs and MPC860 Through HI08

The main function of the data hub (designed by FPGA) is to accelerate the data transferring between MPC860 and DSPs. HI08 is an 8 bits interface. MPC860 needs to poll the status bit of the HI08 register to make sure that the DSP HI08 FIFO will not be over-run/under-run. When a large amount of data is transferred between MPC860 and DSP, the bus bandwidth consumption will be excessive. MPC860 does not have enough interrupt pins to connect to each DSP, which will require extra polling to identify an interrupting DSP. Due to this limitation of HI08, fewer DSPs can be hooked up to a MPC860. By using the data hub design, the HI08 bus is physically isolated from MPC860. The polling of DSP status register will be done by the data hub, the 8-bit data to/from DSPs will be unpack/pack into a 32-bit word before the data hub requests an IDMA transfer. This reduces bus usage and increases the number of DSPs which can be hooked up to MPC860.

The detailed functions of the data hub are discussed below. This includes two types of data transfer - data moving from MPC860 to DSP and data moving from DSP to MPC860.

(TX) MOVING DATA FROM MPC860 TO DSPs

This transfer contains two stages. Stage one is to transmit the header information, setup the IDMA channel in MPC860 and setup of the data hub. Stage two is the enabling the actual IDMA data transfer.

The following steps list the procedure in a block data transfer. Steps 1 to 4 are the stage one header read and setup; steps 5 to 7 are the stage two actual data transfer.

1. MPC860 have to ensure that there is no on-going traffic on the DSP HI08 busses.
2. MPC860 transfers the message header byte by byte to the addressed DSP.
3. MPC860 configures the IDMA channel, set the byte count, target DSP to receive and the direction in data hub.
4. MPC860 issues start command to the data hub.
5. The IDMA controller will service the DMA request by reading the SDRAM. A state machine in the hub transfers the 32-bit word byte-by-byte into an 8-bit FIFO.
6. At the same time, another state machine in the hub is responsible for reading the 8 bit FIFO and writing the data to the selected DSP host port.
7. When the FIFO is empty, the DMA transfer is completed and the DONE bit is set in the hub status register.

(RX) MOVE DATA FROM DSPS TO MPC860

This transfer contains two stages. In stage one, when the DSP generate request and MPC860 get the request from the hub by doing polling hub register. MPC860 then configures IDMA channel and data hub. In stage two, the data transfer takes place.

1. When DSP generates request, the data hub logs the DSPn and the type of transfer into the hub pending register. Then, the data hub interrupts MPC860.
2. MPC860 reads the data hub pending register to decode the DSPn and type of transfer.

3. MPC860 checks the status of data hub to ensure that there is no ongoing HI08 access in the hub. MPC860 then starts the header transfer by reading the target DSP's 24 bit header using direct HI08 access.
4. MPC860 configures the IDMA channel using the header information (block size).
5. MPC860 configures the data hub byte count, write the target DSP number and transfer direction
6. MPC860 issues start command to start data transmission.
7. The data hub reads the data byte-by-byte from the DSP's HI08 interface into the FIFO.
8. On every latch of 8 bit data into the FIFO, the byte count is decremented by one. When the byte count reaches zero, no more data is read from the HI08 port.
9. On the other side, when 32 bits of data is accumulated in the FIFO, the data hub reads four bytes from the 8-bit FIFO and packs them into a 32-bit word. The IDMA transfer 32 bit data to SDRAM.
10. When the byte count reaches zero and the FIFO is empty, the DMA transfer is completed and the data hub returns to ready status.

7.3 System Driver Control

The previous section describes how to implement and design the data hub to increase the transmission rate in between DSPs and MPC860. Here will describe how to design system driver in this high density DSP array board.

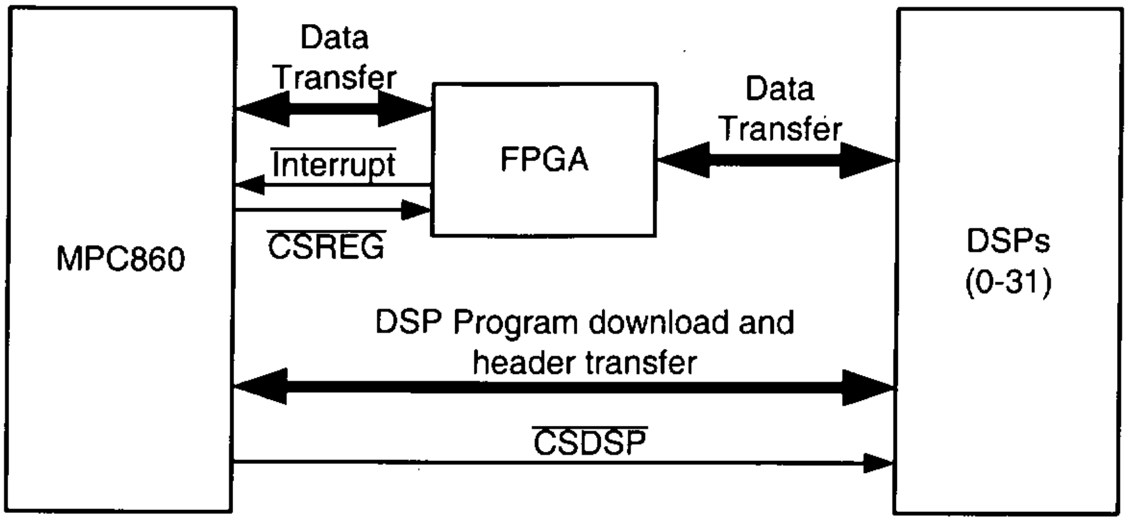


Figure 7-2 HI08 Bus Interface

The HI08 driver contains 2 interfaces between MPC860 and DSPs:

1. Direct Interface (8-bit port) - This interface is used for DSP program download, header (byte count of the data) transfer and DSP HI08 registers read/write.
2. Data Hub Interface (32-bit port) - This interface is used for IDMA data transfer.

7.3.1 Data Hub Interface Design Overview

Each data hub supports up to 32 DSP's HI08 ports. Two data hubs interfaces should be provided for 64 DSPs for IDMA data transfer.

One of the HI08 device driver supports upper 32 DSP (0-31). The other HI08 device driver supports lower 32 DSPs (32-63). These two HI08 device drivers have the same function but they operate independently. MPC860 read data from or write to DSPs is through the corresponding data hub transmit and receive FIFO. Each HI08 device driver has its own semaphore and message queue.

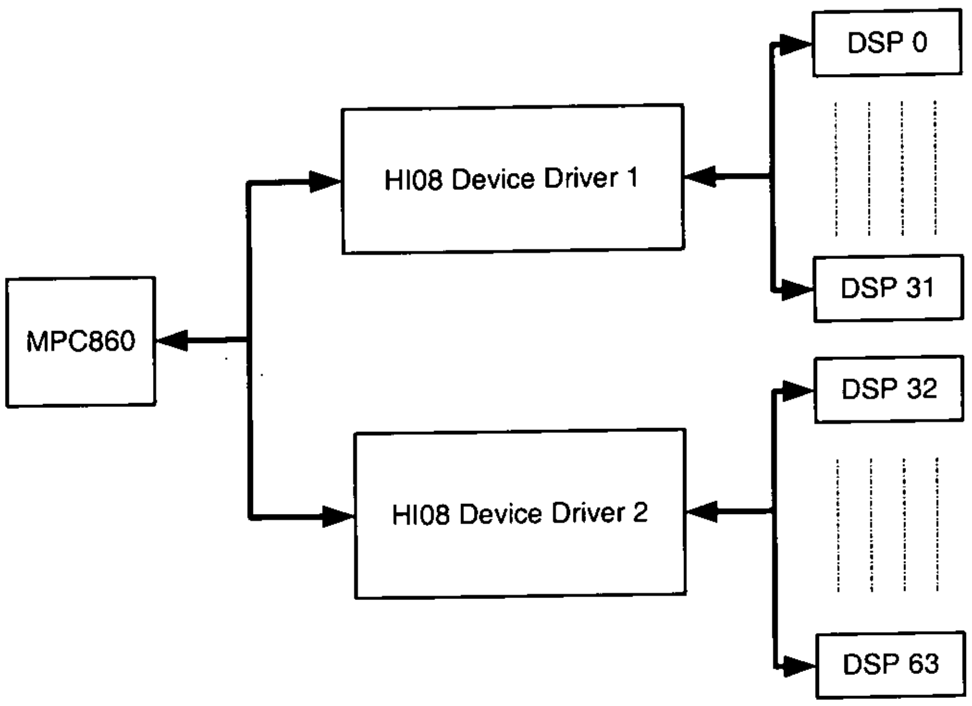


Figure 7-3 Dual Data Hub Interface

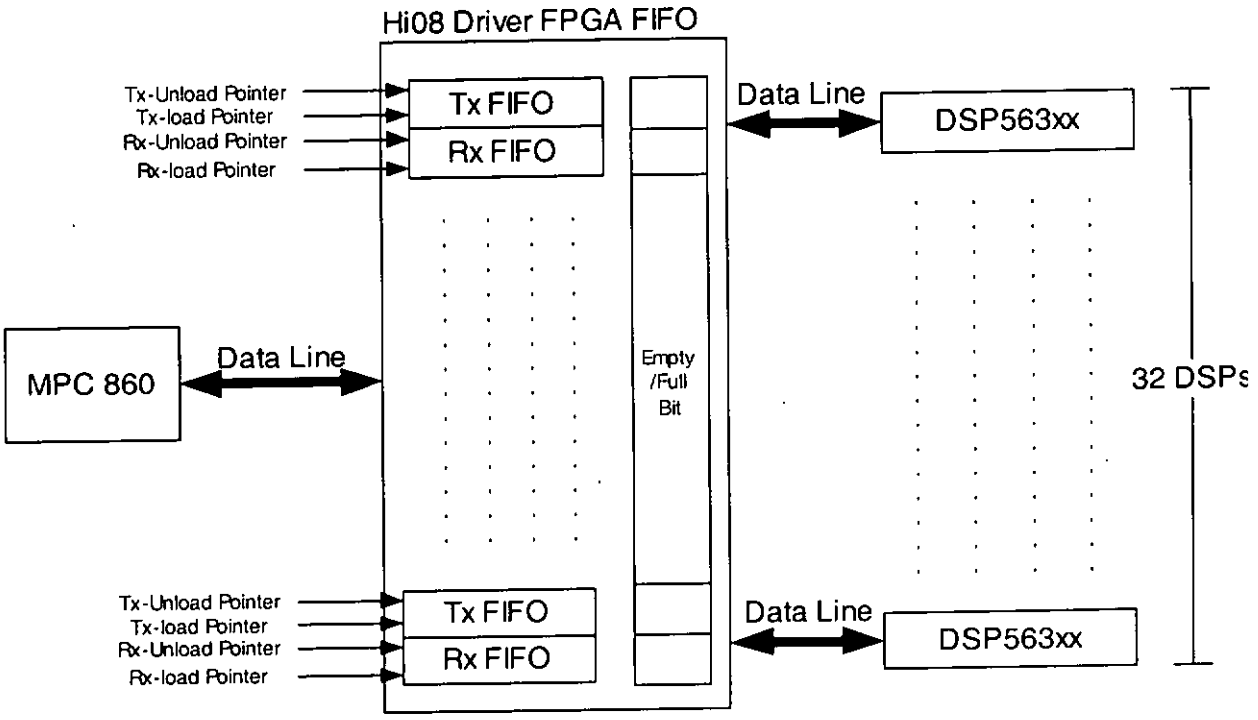


Figure 7-4 Data Hub (FPGA) FIFO Structure

7.3.2 Data Hub Transmit and Receive FIFO

All data read from or write to DSPs via the data hub interface is put in a transmit FIFO or receive FIFO. Each DSP has a transmit FIFO and receive FIFO. There are a total of 128 FIFOs since there are a total of 64 DSPs. Each FIFO has 2 pointers, one load pointer and one unload pointer.

Each transmit and receive FIFO has a bit to indicate the FIFO full or empty. When the bit is set, the FIFO is full. When the bit is clear, the FIFO is empty.

The full/empty bit should be set when the load pointer is passed from the bottom of the FIFO to the top of the FIFO. The full/empty bit should be clear when the unload pointer is passed from the bottom of the FIFO to the top of the FIFO.

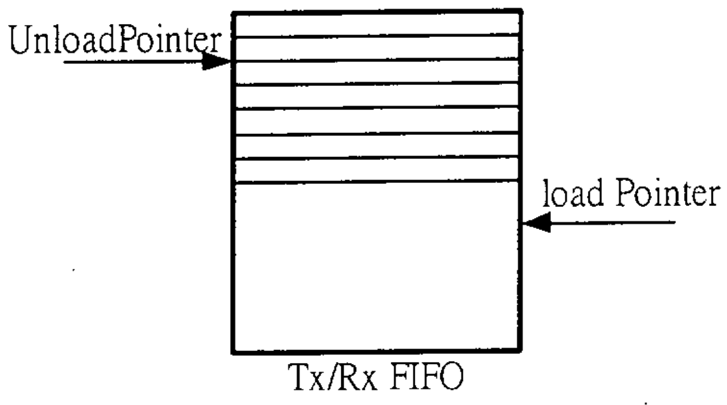


Figure 7-5 FPGA Tx/Rx-FIFO

During the system started up, the load pointer is at the same location as the load pointer. Also, the full/empty bit should clear to indicate the FIFO empty. The FIFO can be viewed as a circular buffer. As the FIFO is loaded, the load pointer is incremented by one. As data is unload from the FIFO, the unload pointer is incremented by one. Two condition are generated - FIFO overflow and empty.

In a transmit FIFO, the unload pointer points to the location of the first data which is ready to send. The load pointer points to the location of the last data which is ready to send.

After data is sent to the DSP, the unload pointer will move towards the load pointer. If all the data is sent, the unload pointer is at the same location as the load pointer and the full/empty is clear. If the FIFO is full, the unload pointer is at the same location as the load pointer and the full/ empty is set.

In a receive FIFO, the unload pointer points to the location of the first data received. The load pointer points to the location of the last data received.

After the received data is read by the MPC860, the unload pointer will move towards the load pointer. If all received data is read by the MPC860, the unload pointer is at the same location as the load pointer and the full/empty is clear. If the FIFO is full, the unload pointer is at the same location as the load pointer and the full/empty is set.

7.3.3 DSP Status Structure

A 32-entries structure named DSP [32] in the data hub is used to save the status of the DSPs, Figure 7-6 shows the status of DSP structure in the data hub:

- DSPOnOff - Stores the on/off and present status of all DSPs.
- FIFO Mode – Save the FIFO full handling.

	[31]		[6]	[5]	[4]	[3]	[2]	[1]
DSPOnOff	1	— · —	0	2	0	1	2	0
FIFO Mode	1	— · —	1	1	1	0	1	1
	32/63		5/37	4/36	3/35	2/34	1/33	0/32

DSP Mode : 0 - that old data in FIFO is overwirtten
1 - that new data is discarded

DSP OnOff : 0 - that DSP is off and present
1 - that DSP is on and present
2 - that DSP is not present

Figure 7-6 DSP Status Structure

7.3.4 Structure of HI08 Interface Driver

The HI08 interface driver contains of 3 parts - application I/O code, device independent code in driver directory and device dependent code in BSP. Figure 7-7 shows the structure of HI08 interface driver and the detail of each function can refer to Appendix B.

1. Application I/O Code

System calls used in the application is for calling the I/O device driver. The calls include de_init, de_open, de_close, de_read, de_write and de_cntrl.

2. Device Independent Code in Driver Directory

Driver code which does not depend on the information of board. The functions include H03Init, H03Open, H03Close, H03Read, H03Write and H03Cntrl.

3. Device Dependent Code in BSP.

Driver code which uses board-specific parameters. The parameters are used by the device dependent code in driver directory.

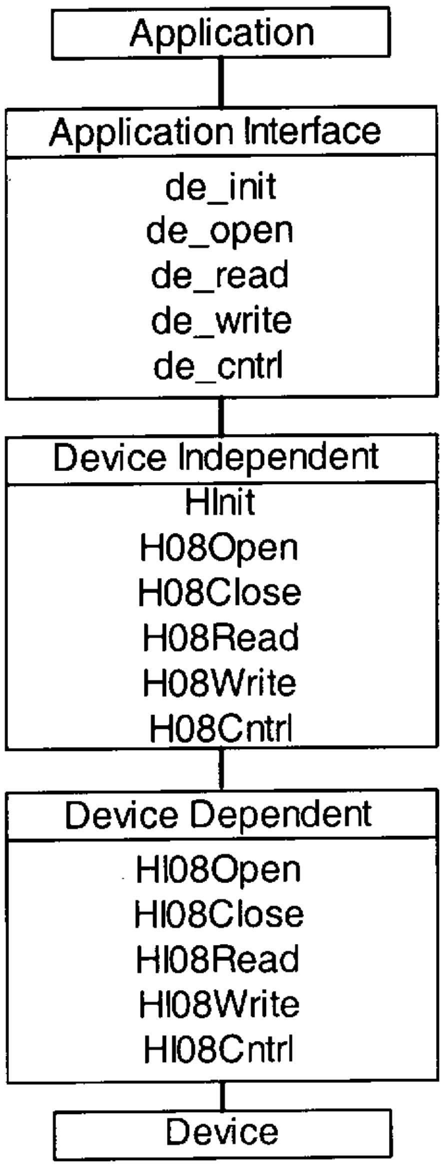


Figure 7-7 Structure of HI08 Interface Driver

7.4 DSP Array Board Performance Measurement

The objective of this test is to measure the performance on by-pass mode and IDMA mode. The maximum bandwidth available for the data hub and the loading of

IDMA channel should be measured. There are two major tests that will be performed

1. Data hub FIFO switching time test - It measures the delay time for switch response for the gate to handle a request in both directions.
2. Data hub stress test – It measures the maximum bandwidth of IDMA channel and the performance of data hub in order to handle huge number of DSPs.

The testing environment setup is:

1. The system clock of DSP array Board is equal 33MHz
2. The ratio of CPU clock and bus clock is equal one
3. Xilinx FPGA XCS40 is used as the data hub design
4. Direction is port pin control so that delay is added to the system to give higher stability

7.4.1 Data Hub Switching Response Testing

Test Methodology

The purpose of this test is to measure data hub switching time. Switching time is read and write test of data hub. During read and write of MPC860, the gates in the hub have to take some time to switch its direction. It is the response time of the data hub. The following two steps show how to get the response time:

1. Send 16 bytes of data from MPC860 to DSPs and send 16 bytes data from DSPs to MPC860. Repeating the process 1000 times. Figure 7-12 shows the flow of the testing procedure.
2. Send 32 bytes data from MPC860 to DSPs and send 32 bytes data from DSPs to MPC860. Repeating the process 1000 times. Figure 7-11 shows the flow of the testing procedure.

After the test, switching response time, send time and receive time is gotten. In each test send time and receive time is recorded. Figure 7-8 shows time flow. (Send time ⇒ switching response time ⇒ receive time ⇒ switching response time ⇒ send time .....).

Switching time of data hub can be calculated as follows:

1. In Figure 7-8, it shows a time sequence of 16 bytes data transfer. If multiply total time by 2, the time is doubled which shows in Figure 7-9.
2. In Figure 7-10, it shows the time sequence of 32 bytes data transfer. It can assume that sending and receiving time for 16 and 32 bytes data transfer is equal.
3. Finally, switching time for one data transfer can be calculated by using the following equation:

$$\text{Switching Time} = \left| \left(\frac{2\text{x} \,(16 \text{ bytes transfer time}) - (32 \text{ bytes transfer time})}{\text{Number of data transfer}} \right) \right| \qquad (7\text{-}1)$$

Notes: To minimize the measurement deviation, the pair of bytes should be as large as possible.

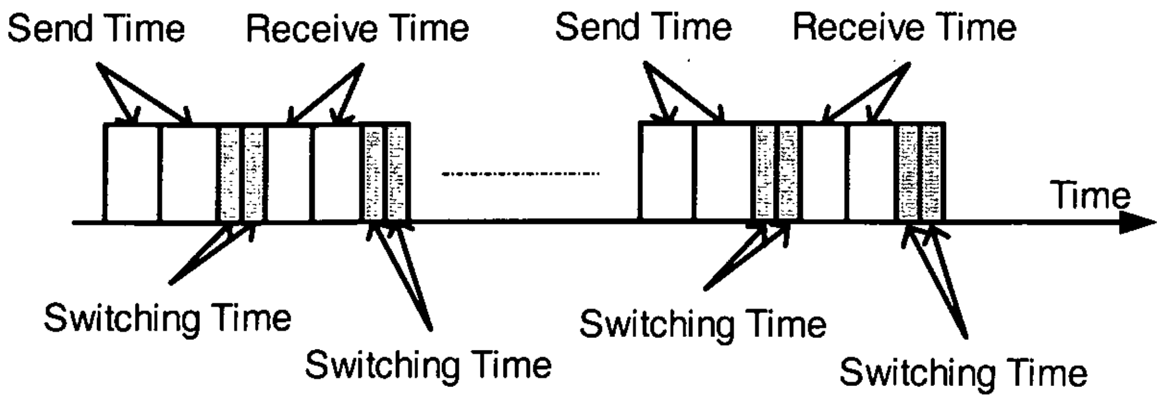


Figure 7-8 Time Diagram for 16 Bytes of Data

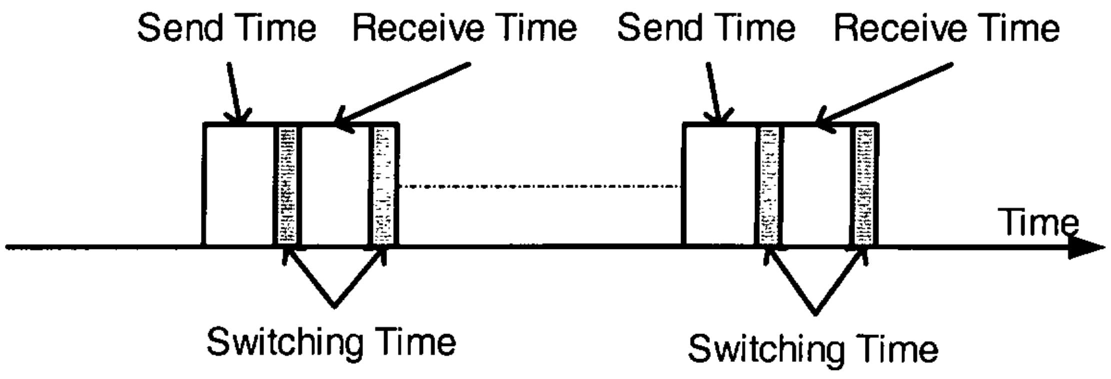


Figure 7-9 Doubling Time Diagram for 16 Bytes of Data

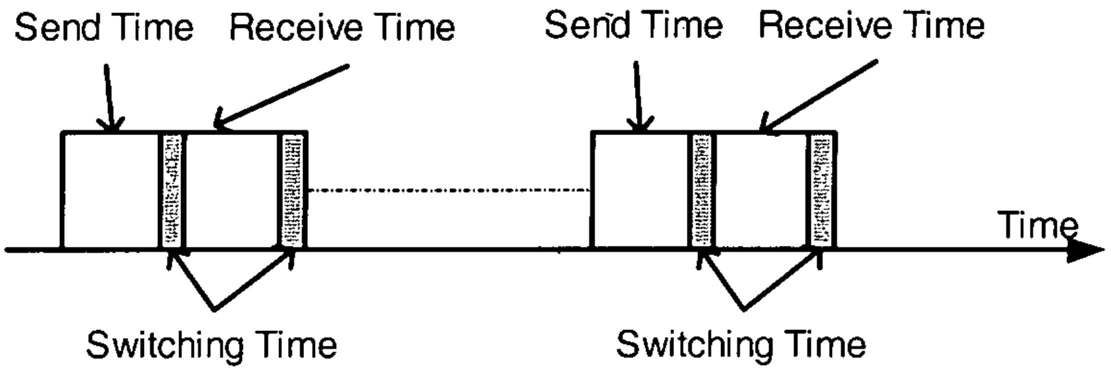


Figure 7-10 Time Diagram for 32 Bytes of Data

Result of Switching Response Test

The switching response time can be calculated from Equation 7-1. In the test, the number of times for gates switching to send and receive data is 800 times. Also, from the measurement, the total time for reading and writing between MPC860 and DSP for 512 bytes packet size is equal 2127.935μs. And the total time for reading and writing between MPC860 and DSP for 1024 bytes packet size is equal 3930.92μs. By using equation 7-1, the switching response time can be calculated as follow:

$$\text{The switching for data hub} = \left(\frac{2x2127.915-3930.92}{800}\right)\mu s = 0.406\mu s \qquad (7\text{-}2)$$

The result shows that the response time of data hub is sufficient to support several number of voice channels because the switching response time is relatively smaller than the time for voice steaming. But in actual situation, gate switching will occur frequently since the data transfer size of voice stream is about hundred of bytes, which depends on what compression ratio is used. The response time have to be improved by increasing the operational frequency of FPGA or reducing the number of gates used in FPGA.

7.4.2 Data Hub Stress Testing

Test Methodology

1. Transmission Test – Figure 7-11 shows the testing flow. Firstly, MPC860 sends data to DSPs continuously. At each time, data size will be increased by 256 bytes. The data pattern should be 1, 2, 3 ... 256, 1, 2, 3 ... In DSP side, it checks the

received data to see whether correct or not. If error occurs, transfer is adopted. The maximum data size is 65535 bytes.

2. Receive Test - Figure 7-12 shows the testing flow. Firstly, DSP sends data to MPC860 continuously. At each time, data size is increased by 256 bytes. The data pattern is 1, 2, 3, ... 256, 1, 2, 3 ... At each end of data received by MPC860, it checks data pattern to see whether it has error or not. If it is corrected, the testing is continuous until the end of the test. Otherwise, the test will be ended.

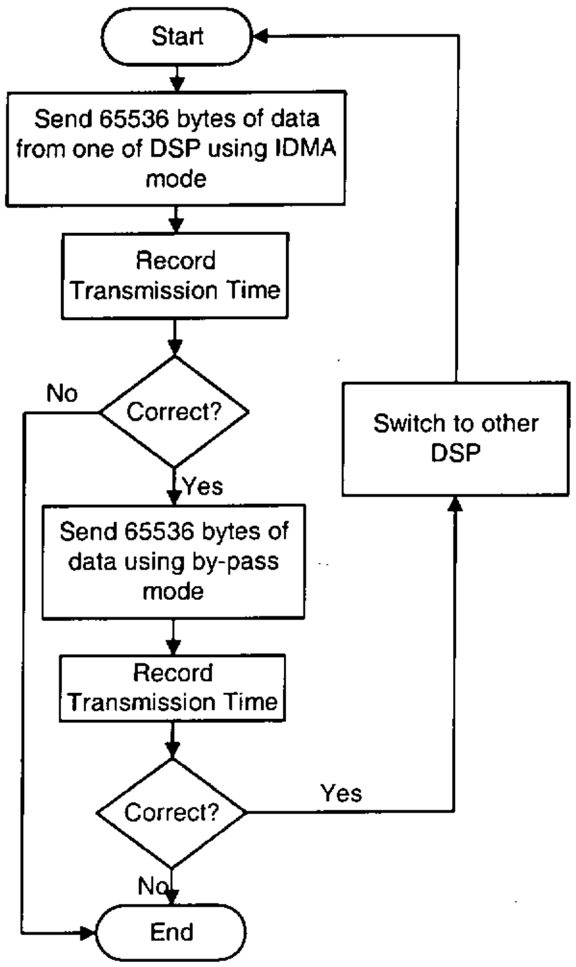


Figure 7-11 Continuous Send Data from Different Number of DSP

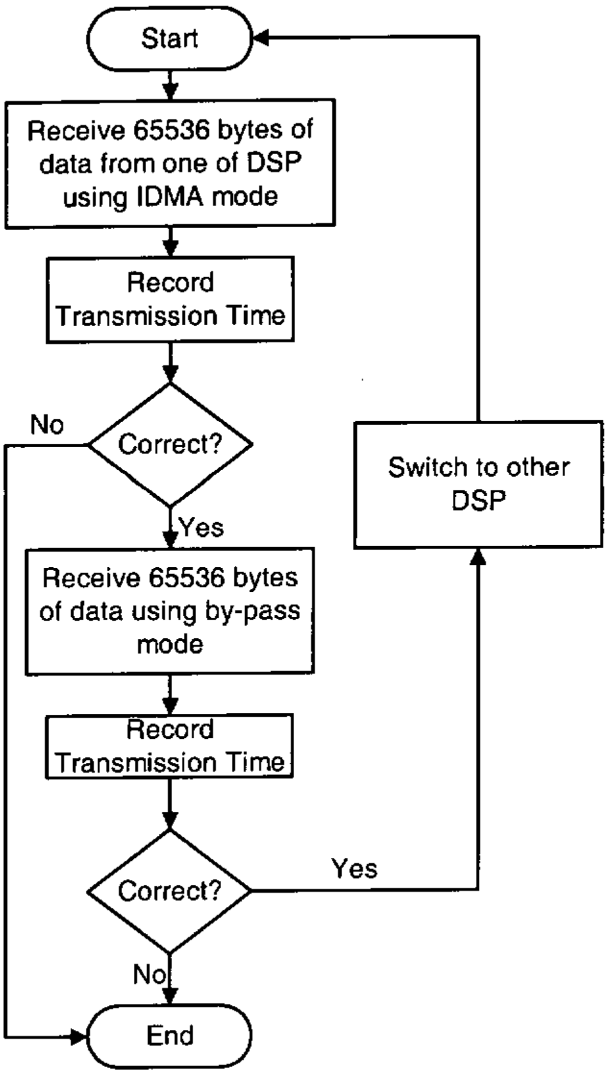

Start
Receive 65536 bytes of data from one of DSP using IDMA mode
Record Transmission Time
Correct?
No
Yes
Receive 65536 bytes of data using by-pass mode
Record Transmission Time
Correct?
Yes
No
End
Switch to other DSP

Figure 7-12 Continuous Receive Data from Different Number of DSP

Test Result of Data Hub Stress Test (33MHz System Clock)

There are some results and comparison can be shared by using by-pass mode and IDMA mode. The system clock is 33MHz.

1. Comparison on By-pass Mode and IDMA Mode in Both Directions

Figure 7-13 and Figure 7-14 show time difference between by-pass mode and IDMA mode on transmission in both direction. Transfer rate of by-pass mode and IDMA is the slope of the graphs. Table 7-1 is the transmission rate using by-pass mode and

IDMA mode in both of directions. It seems that delay is linear proportional to the data size.

	DSP to MPC860	MPC860 to DSP
Transmission rate using by-pass mode (M bytes/sec)	51.64	46.9
Transmission rate using IDMA mode (M bytes/sec)	304	223.35

Table 7-1 Parameter Values for the Adaptation Instances

2. Performance Comparison on By-pass Mode and IDMA Mode in Transmission and Receive Test

Figure 7-15 and Figure 7-16 show that the direction of IDMA from MPC860 to DSP is faster than the IDMA from DSP to MPC860. In these Figures, it proved that delay in both of directions of IDMA data transfer is almost the same. The ratio of DMA-Tx and DMA-Rx is 3 to 2. The data rate sent from MPC860 to DSP is faster than the data rate sent from DSP to MPC860. There are two possible implications on this result:

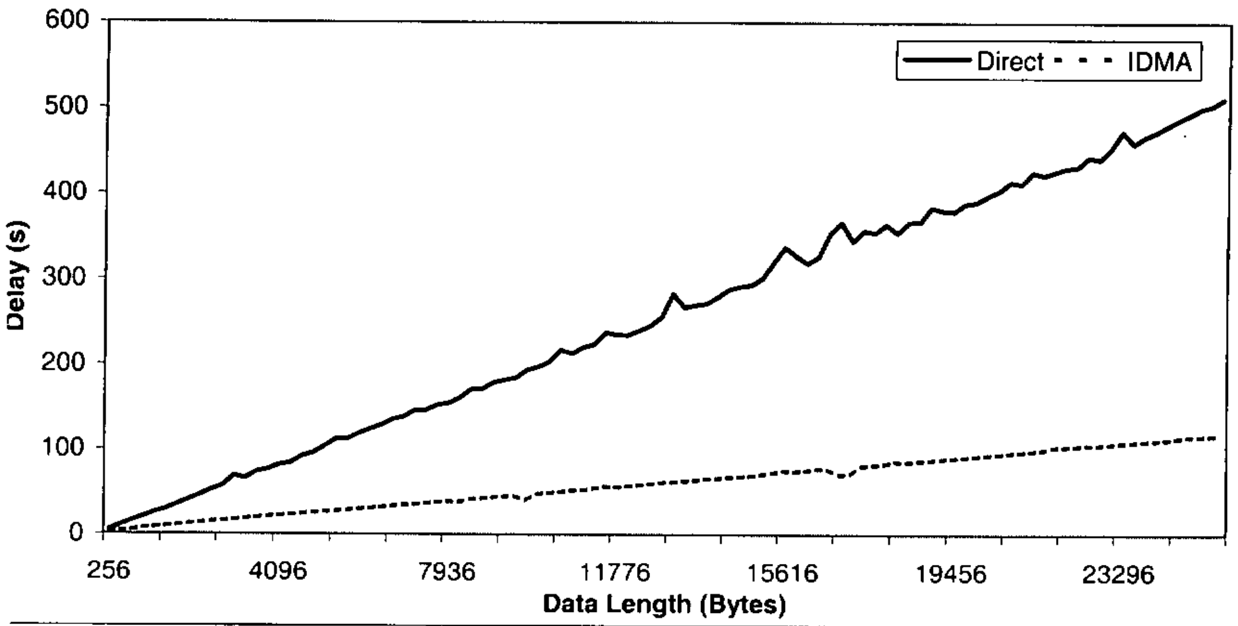
1. The ratio of the gate in FPGA used in these two directions is 3 to 2.
2. The response time of the gate is different in both of transmit side and receive side.

From above observation, it seems that data rate in transmit path and receive path is not balanced resulting in generation of wait status in MPC860 side since the direction from DSP to MPC860 is slower than reverse direction. In order to reduce the wait status in MPC860 side, the best case of this ratio should be one to one such that both of devices (DSP and MPC860) can experience the same amount of delay over these two directions.

In addition, Figure 7-17 is the comparison on delay ratio (By-pass : DMA) in both of transmission and receive directions. It seems that :

1. IDMA mode can improve data rate above 6 times than by-pass mode in direction from DSP to MPC860
2. In direction MPC860 to DSP, IDMA mode using data hub improves data rate about 4 times than by-pass mode.

It seems that the performance on data hub performance in DSP to MPC860 is better in direction MPC860 to DSP.



igure 7-13 Average Delay in By-pass Mode and IDMA Mode from MPC860 to DSP Direction

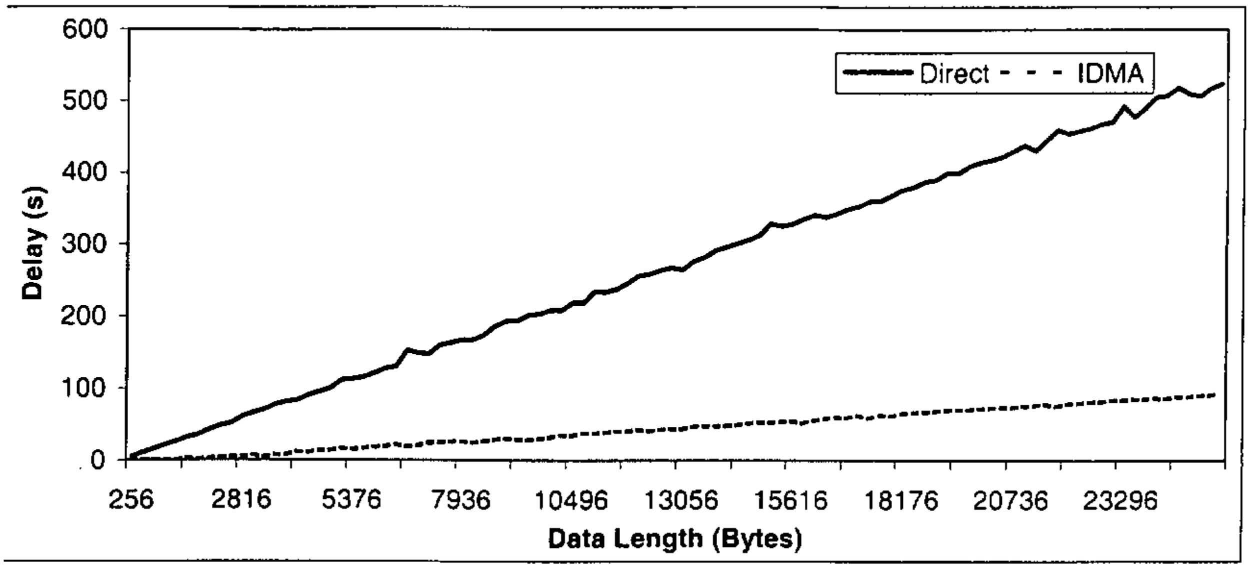


Figure 7-14 Average Delay in By-pass Mode and IDMA Mode from DSP to MPC860

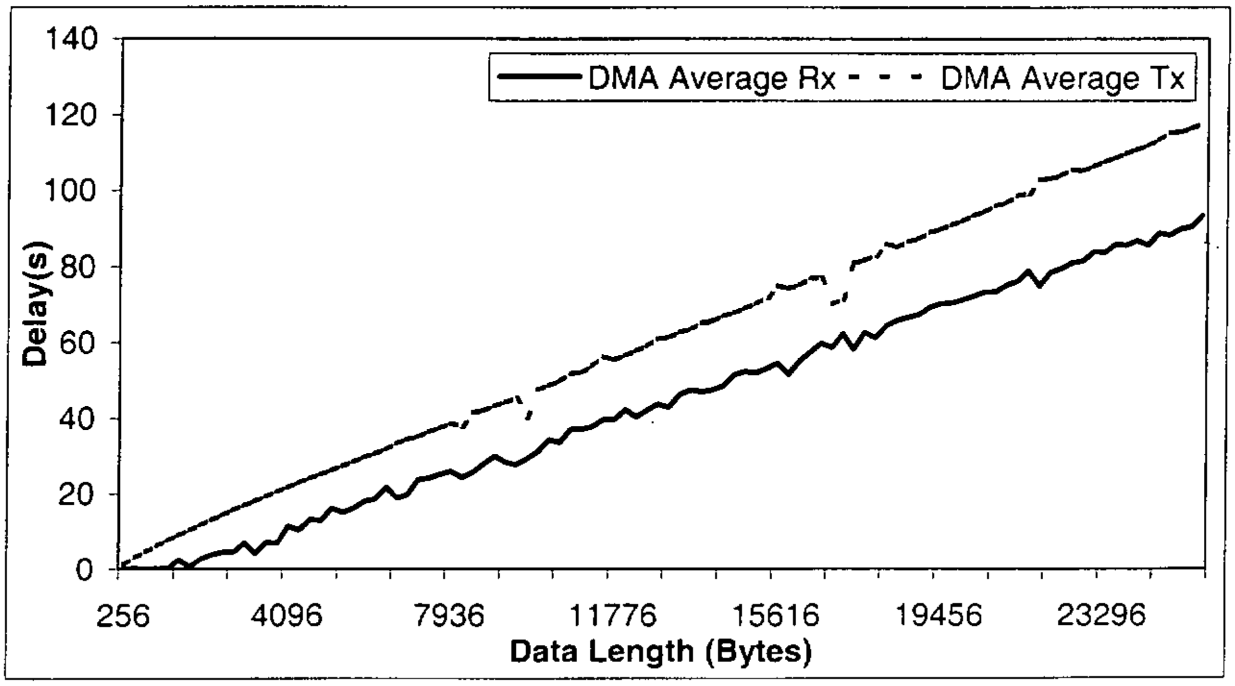


Figure 7-15 Average Delay Using IDMA Mode on Both Directions

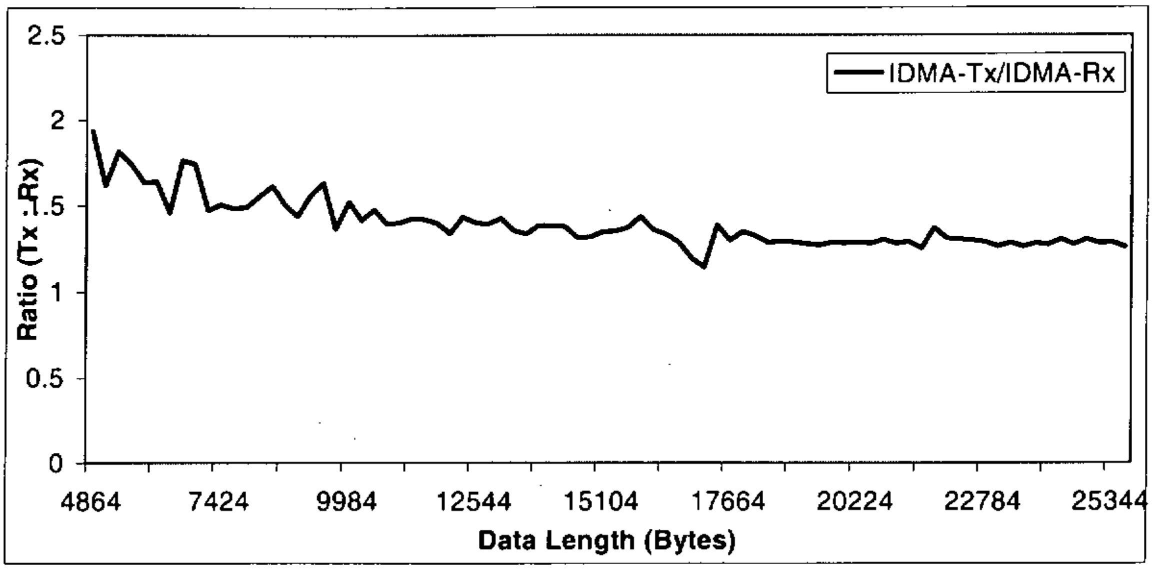


Figure 7-16 IDMA Mode Ratio on Both Directions (DMA-Tx/DMA-Rx)

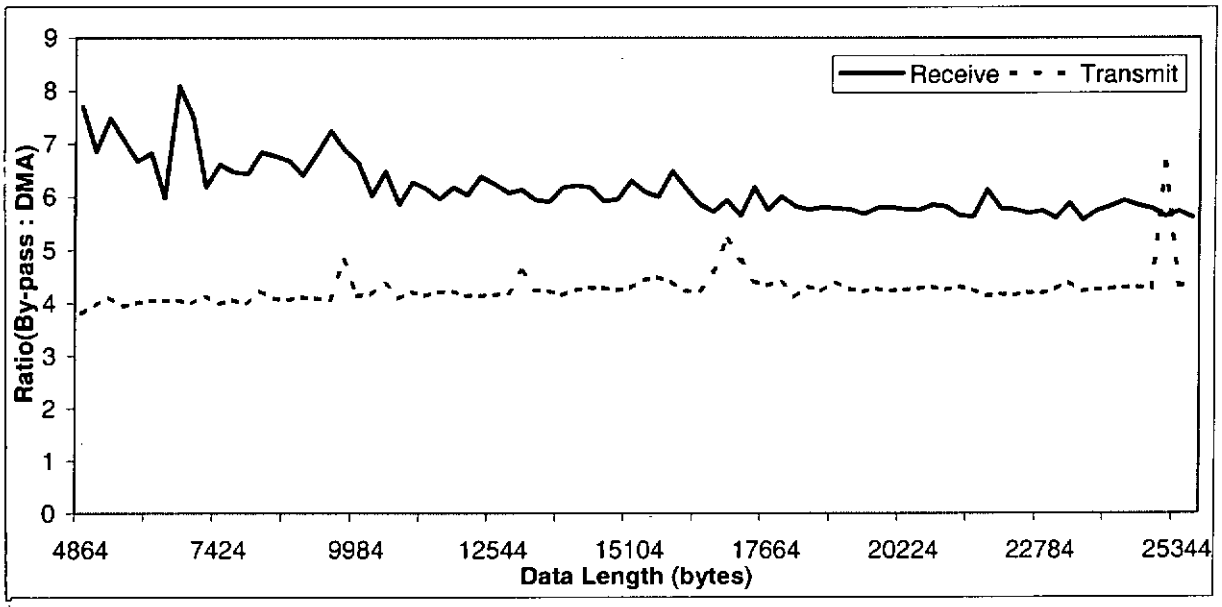


Figure 7-17 Delay Ratio on Both Directions (By-pass Mode /IDMA Mode)

CHAPTER 8

Conclusion and Further Work

8.1 Conclusion

The research is divided into three parts - background research, theoretical consideration and implementation consideration.

The background research gives general overview of current Internet Telephony technology such as QoS requirement on voice communications, issues faced by Internet Telephony, introduction of RTP/RTCP protocol and its feature on how to support real time communications and packet voice service.

Basic on the background research, some QoS parameters and measurement are studied. The QoS parameters include delay, jitter, packet loss and voice compression. Voice compression determines how much bandwidth is required to reserve for each voice channel. Real time voice packets can suffer a number of packet losses, but more sensitive on delay and jitter effect. Another issue is echo but it does not define as in QoS parameter because current technology on echo cancellation compensates most of the echo effect.

Also, another background research covered packet voice model analysis and presented some theoretical analysis on these models. Those models characterize packet voice networks. There are five models to be presented - On-off source, Semi-Markov Process Model (SMP), Continuous Time Markov Chain Model (CTMC), Uniform Arrival and Service Model (UAS) and Markov Modulated Poisson Process Model (MMPP). On-off source model divides voice conversation into six states. Semi-Markov Process Model overestimates probability that queue is empty because it ignores high frequency. Continuous Time Markov Chain Model conservatively estimates the probability of buffer overflow and waiting time. Uniform Arrival And Service Model assumes that information does not enter the transmission buffer until a particular active source completes generation of packet. It would appear that the effect of this approximation on the UAS model's accuracy is small. MMPP model is the most popular model in current research for packet voice because it provides the most accurate result.

Moreover, the thesis specifies the requirement of QoS for the Internet Telephony application. Two issues is addressed

1. How much bandwidth should be assigned to one voice channel?
2. How to define the desired quality?

In question 1, bandwidth requirement is based on the compression ratio of the voice. In question 2, the worst delay should not exceed 350ms and the packet loss should be smaller than 1% to 2%. If the transmission media violate these values, the voice quality considers as unacceptable.

From the background research, it provides useful information and guideline to implement Real Time Feedback Control Mechanism (RFCM) and design VoIP gateway such that it can meet those QoS requirements.

On the other side, in theoretical consideration, two QoS control algorithms are studied - Direct Adjustment Algorithm (DAA) and Adaptive Layer Transmission Scheme (ALT). Those two algorithms do not consider the total bandwidth available in the channel. If the maximum bandwidth is reached, it cannot reserve any more bandwidth for particular channel. Also, the oscillatory of traffic and scalability of network size are the other issues that have not been considered.

The proposed Real Time Feedback Control Mechanism (RFCM) separates different types of arrival source, dynamically assign channel bandwidth for different applications, monitor traffic on the network and prioritize network traffic. The server implemented RFCM can obtain the information about bandwidth utilization by receiving RTCP message and uses the feedback control model to adjust the transmission rate dynamically. Moreover, the issue of RTCP message flooding is also considered in RFCM. When the bandwidth utilization is stable, the generation rate of RTCP message is reduced such that it prevents too much RTCP messages generated in the session if the number of terminal joining into the session is so large or the network is stable. From the simulation, it seems that it can maintain the constant rate of traffic. i.e. there are no oscillatory effect comparing with DAA and ALT by alternating the scheduling rate.

The scheme uses three network models - channel model, service model and real time control model to provide a systematic control mechanism on packet transmission. The service model is to prioritize each source and distribute different class of data into different types of queues. Real time control model is used to gather RTCP messages and adjust the transmission rate of the packet. Channel model is used to model traffic loading and separates the traffic into three types – RTP data traffic, TCP data traffic and control message.

The simulation shows that bandwidth can be dynamically adjusted. Comparing with ALT and DAA model; it provides systematic and realistic traffic control in packet switching network.

The third area of the research is the implementation consideration. The performance of multimedia gateway implementation is another factor that influence overall system performance. Common Communications Platform (CCP) gateway implementation methodology proposes a standardized platform for developer to design their system so as to speed up cycle time of product design and increase system flexibility.

The platform includes hardware and software designs.

In hardware design, it uses Compact PCI as the common target board control and H.110 standard for voice stream passing. It uses VoIP gateway as an examples to implement CCP concept. Actually, different combination of board configurations can provide different types of applications. Developer uses their previous designed board for new application and does not need to redesign the whole system. For example, line card can be used as a standard interface to connect telephone, developer does not need to redesign a new line card in another application such as PSTN. The advantage of this methodology can reduce the cycle time of product design and reuse the resource.

In software design of CCP, three-dimensions model on multimedia software development is used. It separates different types of requirement in three layers (layering plane, service plane and quality control plane) such that developer can change their software to support different types of application such as wireless local loop. Layering plane is used to isolate the application and hardware dependence parameter. Quality control plane considers on what quality requirement is needed in the application. Service plane considers how to manage different types of data in a single application such as signaling information, real time and non-real time data traffic management. Also, service plan can be used to implement traffic control mechanism.

Other than that, it presents how to implement 64 DSPs into Compact PCI 6U form interface board. It is a high-density board design to increase the number of voice channel in a single system. FPGA design is used as a data hub to isolate the information passing between network processor and DSPs such as byte length conversions, interrupt handling. There are two modes - By-pass mode and IDMA mode. By-pass mode is used to download DSPs codec and provides single byte read/write in between network processor and DSPs. IDMA mode provides another faster path to pass a large amount of data from system memory to DSPs or reverse. The performance measurement shows that IDMA mode reduces the system delay time for data moving in a single system. It means that processing delay is reduced.

8.2 Further Improvement and Research Direction

There are some works that can be further developed:

1. Feedback message delay analysis - In real time feedback control feedback controller; it assumes that there is no delay on the RTCP feedback message. But in actual situation, it experiences certain amount of propagation delay. Such delay may affect the stability of feedback control. Further delay analysis is needed to investigate such delay in order to provide an acceptable boundary to limited the delay.
2. Expansion of QoS control using laying approach – The service model in RFCM can be further develop using layering approach to handle different class of traffic

characteristic. One of the solutions is to increase the number of queue in the service model. But it will increase buffer size of the server. The other solution is to dynamically relocate the resource queue in system level configuration. To maintain the simple and distributed control purpose, servers have to maintain self-layering design according to their application.

3. Implementation of RFCM into CCP platform - Another work has to do is to implement RFCM into the CCP platform. The control algorithm should be implemented in real time traffic control layer of the service plan inside CCP software model design.

4. Enhance the feedback control model for traffic prediction by using neural network approach -The design of feedback control model in real time control feedback mechanism can be further enhanced for traffic prediction. The feedback controller records bandwidth utilization in different time period. And then, feedback controller uses those statistical records to predict bandwidth utilization and re-routes the packets to other servers for load balancing purpose. In fact, the method used to implement traffic prediction by using feedback controller is opened. It needs more research to choose the best method to do traffic prediction.

References

[1] D. Anick, D. Mitra, and M. Sondhi, "Stochastic Theory of a Datahandling System with Multiple Sources, " in Conf. Rec. 1980 Int. Conf. Commun., vol 1, Seattle, WA, 1980, pp. 13.1.1-5.

[2] Harry Heffes, and David M. Lucantoni, "A Markov Modulated Characterization of Packetized Voice and Data Traffic and Related Statistical Multiplexer Performance", in IEEE Journal on Selectēd areas in Communications, Vol. SAC-4, No. 6, Sept 1986.

[3] ITU Rec. G.114, "One-way Transmission Time," Feb. 1996

[4] R. V. Cox, "Three New Speech Coders from the ITU Cover a Range of Applications," IEEE Commun. Mag., Volume: 35 Issue: 9 , Sept. 1997,pp 40-47.

[5] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP : A Transport Protocol for Real Time Applications," Request for Comments (Proposed Standard) 1889, Internet Engineering Task Force, Jan. 1996

[6] J.N. Daigle and J. D. Langford, "Model for Analysis of Packet Voice Communication Systems," IEEE Journal on Selected Areas in Communications., vol. SAC-4, pp.847-855, Sept. 1986.

[7] Chin Yuan and John A Silvester, "Queueing Analysis of Delay Constrained Voice Traffic in a Packet Switching System", IEEE Journal on Selected Areas in Communications, vol 7, No 5, June 1989, pp 729-738.

[8] K. Almeroth and M. Ammar, "Collecting and Modeling the Join/leave Behavior of Multicast Group Members in the Mbone," in Proceedings of the Symposium on High Performance Distributed Computing, (Syracuse, NY), pp. 209-216, IEEE, Aug. 1996.

[9] K. Sriram and W. Whitt, "Characterizing superposition arrival processes in packet multiplexers for voice and data," IEEE Journal on Selected Areas Communications., vol. SAC-4, pp.833-846, Sept. 1986.

[10] Brady, P.T., "A Model for Generating ON_OFF Speech Patterns in Two-way Conversations." Bell System Technology Journal, Vol.48. pp 2445-2472, Sept, 1969.

[11] J. N. Daigle and J. D. Langford, "Models for Analysis of Packet Voice Communication Systems," in IEEE Journal on Selected Areas m Communications, Sept. 1986, pp. 847-855.

[12] http://www.advanced.org/surveyor/, Advanced Network & Services, Inc.

[13] D. Sanghi et.al., "Experimental Assessment of End-to-End Behavior on Internet", Proc. IEEE INFOCOM '93, Vol.2 March 1993, pp 867-874

[14] "One-Way Transmission Time. Transmission Systems and Media". ITU-T recommendation G.114, Rev. 1. International Telecommunication Union, 1993.

[15] John G. Gruber and H. LE, "Performance Requirement for Integrated Voice/Data Networks." IEEE Journal on Selected Areas Communications, Vol SAC-1, no 6, pp 981-1005, Dec. 1983

[16] S. Floyd and F. Kevin, "Router Mechanisms to Support End-to-end Congestion Control", Technical Report, Feb. 1997.

[17] T. Ott, J. Kemperman, and M. Mathis. "Window Size Behavior in TCP/IP with Constant Loss Probability". in the Fourth IEEE Workshop on the Architecture and Implementation of High Performance Communication Systems (HPCS'97), Chalkidiki, Greece, June 1997.

[18] S. McCanne, V. Jacobson, and M. Vetterli. "Receiver-driven Layered Multicast". In SIGCOMM Symposium on Communications Architectures and Protocols, Palo Alto, California, Volume 26 Issue 4, pp1-14, August 1996

[19] D. Hoffman and M. Speer. "Hierarchical Video Distribution over Internet-style Networks". in Proceedings of the IEEE International Conference on Image Processing, Lausanne, Switzerland, Sept. 1996, pp. 5-- 8.

[20] L. Wojnaroski. "Baseline Text for Traffic Management Sub-working Group". Technical Report 94-0394r5, ATM Forum, Oct. 1994.

[21] J.G. Gruber, "Delay Related Issues in Integrated Voice and Data Networks," IEEE Trans. in Communications., 29, 1981, pp. 786-800.

[22] S.Q. Li and Jon W. Mark, "Traffic Characterization for Integrated Services Networks", in IEEE Trans. in Communications. 38, 1990, pp. 1231-1243.

[23] K. Sriram and W. Whitt, "Characterizing superposition arrival processes in packet multiplexers for voice and data" IEEE Journal on Selected. Areas Communications, SAC-4, 1986, pp 883-846

[24] Kevin Fall, Kannan Varadhan, "NS-Network Simulator User Manual", UC Berkeley, LBL, USC/ISI, and Xerox PARC, The VINT Project http://www.isi.edu/nsnam/ns/doc/ns_doc.pdf

[25] MPC860 PowerQUICC II Users Manual, Motorola, 1996, http://e-www.motorola.com/brdata/PDFDB/docs/MPC860UM.pdf

[26] DSP56300 Digital Signal Processor Family Manual, Motorola. 1996 http://e-www.motorola.com/brdata/PDFDB/docs/DSP56300FM.pdf

[27] K. T. Yam, C.K.Li and T.P.J. To, "Real Time Network Control of Internet Telephony using RTP/RTCP", Proceedings of the 4th IASTED International Conference in Internet and Multimedia Systems and Applications (IMSA'2000), Las Vegas, USA, November 19-23, 2000, pp145-150, 2000.

APPENDIX A

H.323 and SIP Reference Model

A.1 H.323 Description

The H.323 is an umbrella recommendation from the ITU. It sets the standards for the multimedia communications over Local Area Networks that do not provide a guaranteed Quality of Service. It provides a foundation for audio, video, and data communications across IP-based networks. It supports the conference and point-to-point communications.

Protocol stack

H.323 includes different types of protocol. It specified how to combine these protocols together to achieve multimedia communications.

1. Real Time Protocol (RTP) - This protocol used a time stamp, sequence count and payload type to achieve the real time data transfer. It is generally used in conjunction with the UDP, but it can make use of any packet-based lower-layer protocol.

2. Real time Control Protocol - Mostly, it uses with the RTP to control the packet flow. It sends the message between the terminals periodically. The message includes the sender reports, receiver report and source descriptor. In a session, each terminal can obtain the state of the other terminals and the simple session control.

3. H.225 Control Protocol - The main function is to format the transmitted video, audio, data and control streams into message passing to or getting from the network interface. It includes logical framing, sequence numbering, error detection and error correction.

4. H.245 Call Signaling Protocol - It used to negotiate the channel usage and capabilities. The function includes capabilities exchange, opening and closing of logical channels, mode preference requests flow control message and general commands and indications.

5. Audio Codec - It supports G.711, G.723 and G.729 format. It includes digitalized and coded speech. These codec can provide low bit rate, up to 8kbps, for packet network.

6. Video Codec - It support H.261 and H.263 standard. These codec used to code and digitized the motion video.

7. Data Interface - It used T.120 to achieve data transmission. The data includes pictures, facsimile, documents, computer files and other data streams.

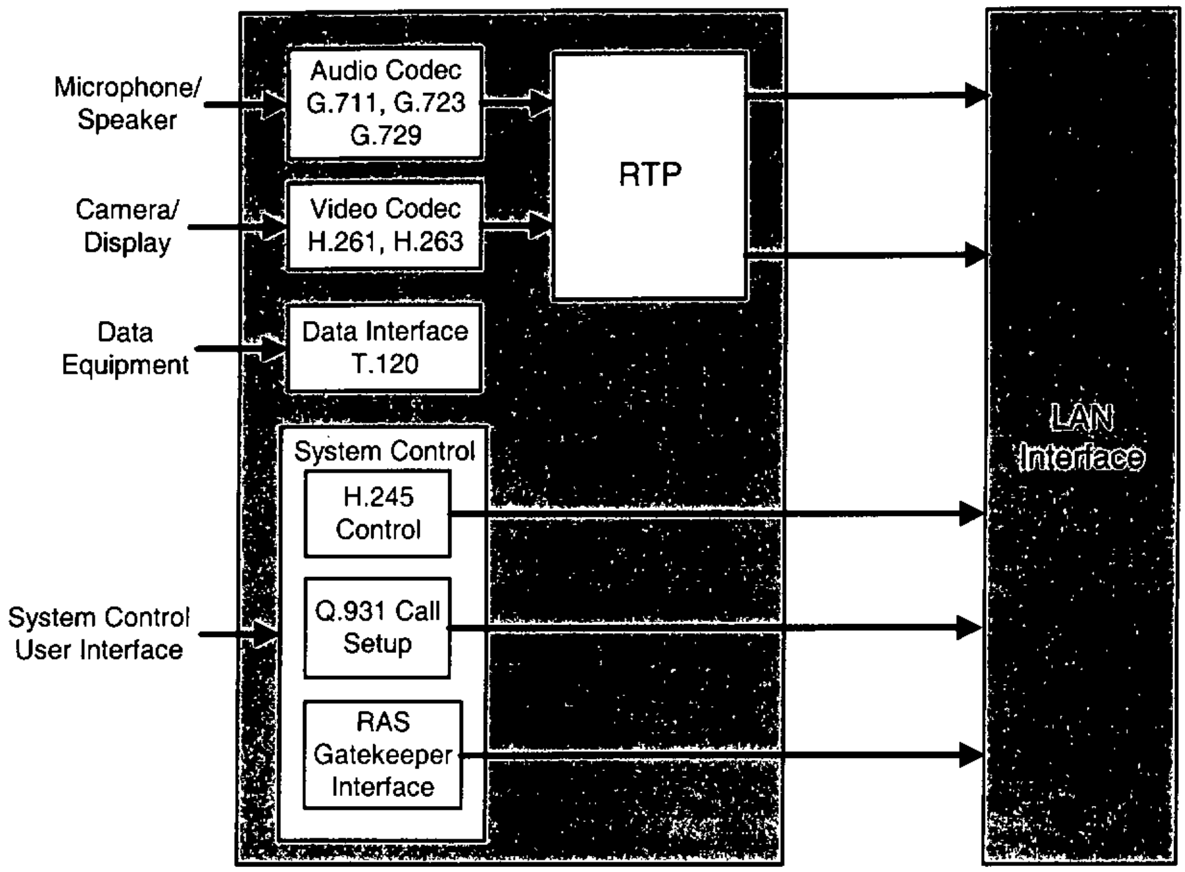


Figure A- 1 Protocol Stack

Participant

H.323 has four main components for network-based communications systems - Terminals, gateways, gatekeepers, and multipoint control units.

1. Terminal - It provides an interface for user to communicate with the other side. Each terminal has to support several type of protocol to achieve the multimedia communications. It provides a bi-direction audio, video and data communications.
2. Gateway - It acts as an interface for both H.323 equipment and non-H.323 equipment. It translates the H.323 data format to the other type of protocol such as

Ethernet. It also performs the call setup and clearing on the LAN and the switched circuit network.

3. Gatekeeper - It acts as a central point for all calls within its zone and provides call control services to registered endpoints. The function of gatekeeper includes address translation, admissions control, and bandwidth control and zone management. Through the gatekeeper, the bandwidth reservation and the call registration can be under control

4. Multipoint Control Units. - It supports multipoint conferences. The multipoint control unit contains a multipoint controller (MC) and zero or more multipoint processor (MP). Multipoint controller handles H.245 negotiations between all terminals to determine the common capabilities for audio and video processing. Multipoint processor is used for the audio mixing, data distribution, and video switching/mixing functions. Also, MP can be used to convert the codec and bit rates to the required format.

Signaling

H.323 have several signaling procedures. In normal connection, it has to follow the procedure below:

1. Call Setup - At this stage, the terminal uses the Q.931 to make the connection between two terminals. It supports direct call signaling and routed call signaling. The connection may go through the gatekeeper or not.
2. Initial Communication and capability exchange - It follows the H.245 control protocol to setup the communications channel. The operation in this stage includes capability exchange and master-slave determination.
3. Establish of audiovisual communication - The main function is to setup a logical channel by using H.245. The end-to-end point will setup different types of channel according to the user specific. That audio and video use unreliable protocol for data transmission and data uses the reliable protocol.
4. Call Service - After above three steps, the data can be transmitted in this stage. Also, the gatekeeper and the gateway will initiate the control signal to manage the system during the communication. The control signal includes bandwidth change and keeps the status of each connected end-point.
5. Call Termination - In terminating the communication, the audio, video and data logical channel will be closed immediately. Then, H.245 control channel will be closed. Finally, the call signaling channel will be closed to terminate the communications. The call termination may go through the gatekeeper or not.

A.2 SIP description

Session Initiation Protocol (SIP) is a proposed standard to the IETF and being under developed. The application on this protocol includes Internet multimedia conferences, Internet telephone calls and multimedia distribution. It is an application-layer control protocol for creating, modifying and terminating sessions with one or more participants. It is a client - server protocol. The call setup and the protocol stack are similar to the HTTP protocol. It can use either UDP or TCP as a transport protocol.

Participant

Four components are included in this protocol.

1. User agent - It is an application programs that are used by the caller or callee to initiates the SIP request. It contains a user agent client and user agent server. The user agent client is the calling user agent and the user agent server is the called user agent. Through the user agent, ones can join the conference.
2. Proxy Server - The purpose of the proxy server is used to receive the request from the user agent and generates outgoing request. The server may be stateful or stateless. Through the address field, the proxy server can generate the request to the destination.

3. Redirect Server - It can accept a SIP request to map the address into zero or more new address to the client. It does not initiate its own SIP request or accept a call. IT maintains transaction state for the whole SIP transaction.

4. Location Server - It used to locate the caller's possible location(s).

SIP Message

General speaking, SIP message can be divided into two groups - request and response. The request is sent from a client to a server. Response is generated from the server to give the feedback to the caller request.

- Request - The client used a method to generate a request. A method is a message format to initiate the SIP request. It involves INVITE, ACK, OPTIONS, BYE, CANCEL, REGISTER.
- Response - After receiving the request from the client, the server will response the client using the response message. The response types include Informational, Success, Redirection, Client Error, Server Error, Global Failure.

Header field structure

The header field is similar to the structure of HTTP header in both syntax and semantics. It gives four types of header field - general header, request header response header and entity header. These header fields follow the same generic header format. The main information in the header field includes Accept, Allow, Authorization, Call ID, Contact, Content, Common Sequence, Date, Encryption, Expires, organization, etc.

Feather of SIP

1. HTTP - like format i.e. text-base format.
2. Server and client architecture
3. Run on the top of the UDP and TCP.
4. Support Multi-party call - To add one or more user, SIP uses the ALSO header in the INVITE call to add the additional user to the conference. The multi-party call is distributed due to the server client relationship.

SIP basic operation

There are two basic operations on the SIP signaling. SIP uses this connection procedure to support the IP mobility.

A. Through proxy server

1. The caller issues the "INVITE:people@polyu.edu.hk" to the proxy server at domain 'polyu.edu.hk'.
2. The proxy server sends the "people" to location server. Then, the location server will locate the computer name, which is used by the "people".
3. The location server return the computer name to the proxy server
4. Then the proxy server will send "INVITE: people@garfield" request to the computer "Garfield".

5. the "Garfield" gives the response to the proxy server

6. The proxy server returns the response to the caller

7. Then, caller gives the acknowledge to the proxy server

8. The proxy server sends the acknowledge to the destination computer "Garfield"

B. Through the relocation server

The procedure 1, 2 and 3 are same as the connection through the proxy server.

1. The relocation server returns the destination computer name "Garfield" to the caller

2. The source client sends the "INVITE: people@garfield.polyu.edu.hk" request to the destination

3. Computer "Garfield" sends a response to the source client

4. Finally, computer "ghost" sends the acknowledgement to end up the connection procedure.

After making a connection, different communications protocol can be used to transfer the data depending on different operations. It supports SDP, RTP, RTCP, etc.

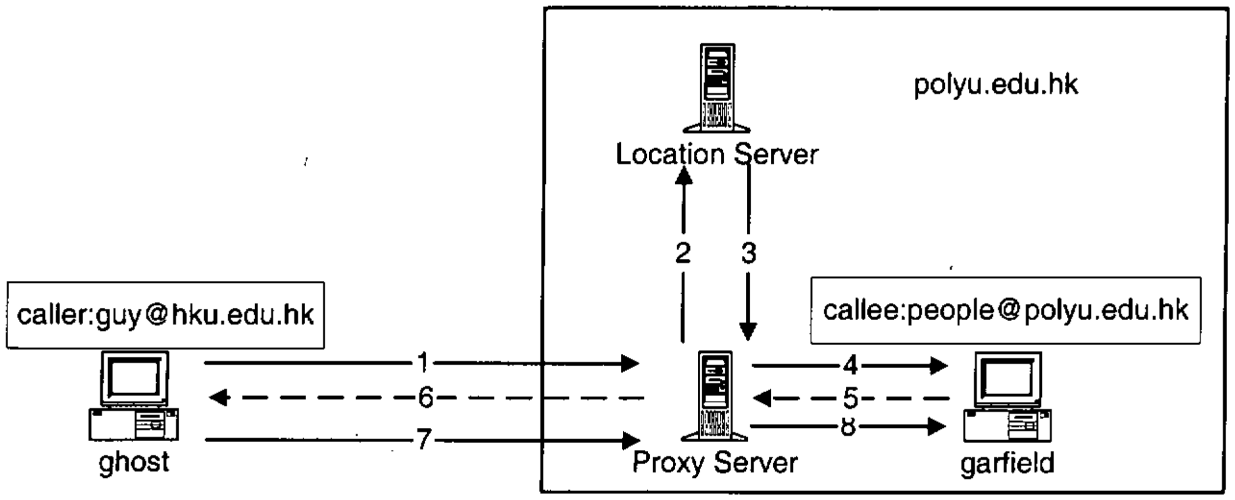


Figure A- 2 Through Proxy Server

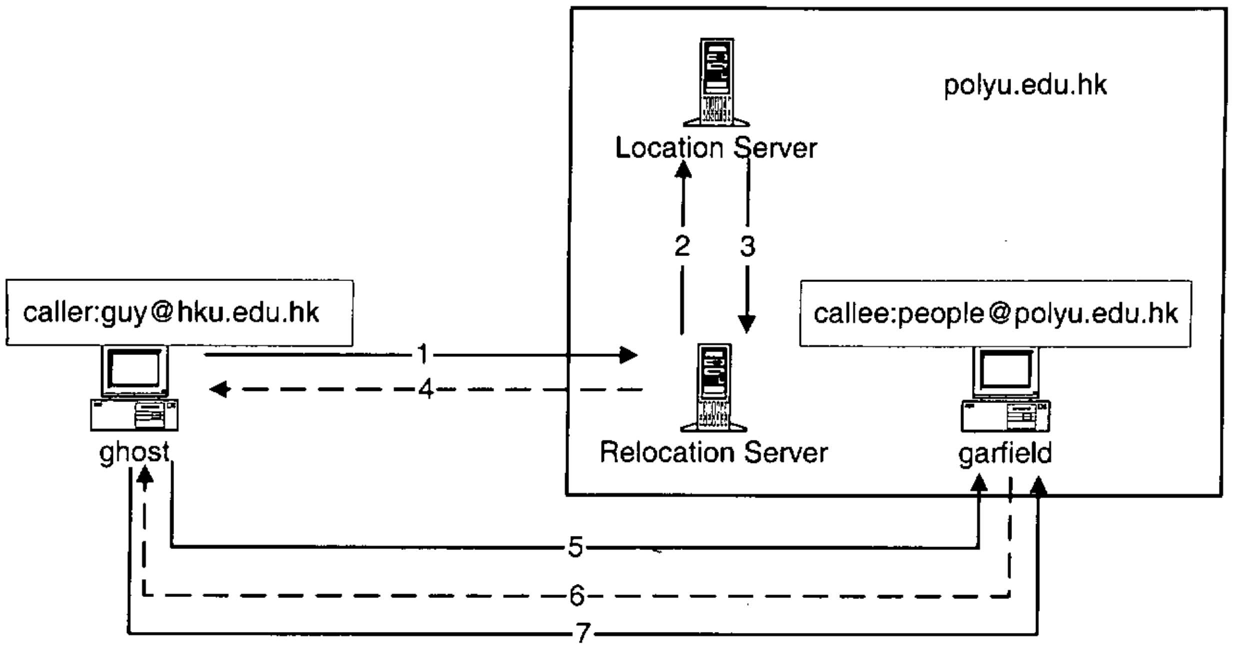


Figure A- 3 Through Relocation Server

APPENDIX B

DSP Array Board Driver Design

B.1 Design of HI08 interface driver

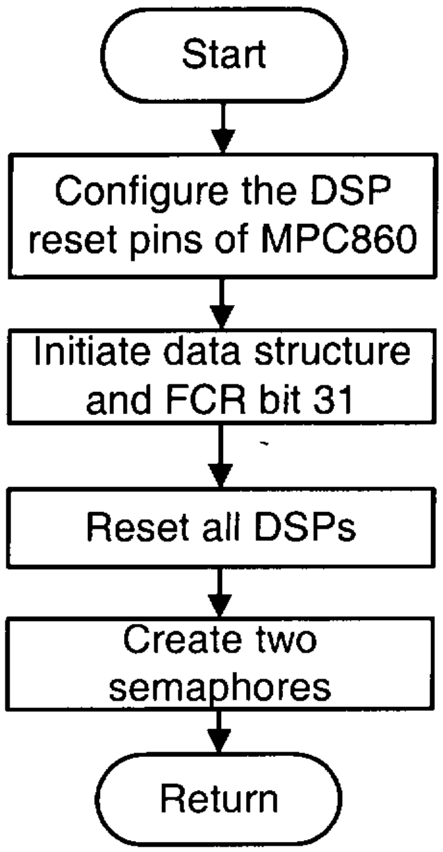

Start
Configure the DSP reset pins of MPC860
Initiate data structure and FCR bit 31
Reset all DSPs
Create two semaphores
Return

Figure B- 1 Flowchart of Hinit

1. de_init & H03Init

de_init() invokes the function H03Init().

H03Init() configures the DSP Reset pins of MPC860 and resets all DSPs. Two semaphores are created to ensure that only one task is using the driver at a time.

2. de_open & H03Open

de_open() invokes the function H03Open().

H03Open() downloads DSP programs to the specified DSPs via direct interface. The last word (3 Bytes) of the DSP program is modified to be the total checksum (the 2 most significant bytes) and the DSP number (least significant byte).

After the DSP program is downloaded and started running successfully, a Success Indicator (Check Sum of all data) will be received by MPC860 from the DSPs. If a Success indicator is received, a "On" status is marked in the array DSPStatus and unmasked the interrupt handlers of DSP_INT & DMA_INT, the task for DSP-Request, and TxRx-Queue are initialized and started.

3. IDMA Transfer Flow

After the DSP program is successfully downloaded, interrupt handlers of IRQ is initialized. DSP_INT is the interrupt from the data hub to MPC860 when DSPs request data transfer from MPC860. DSP_DMA is the interrupt from the data hub to MPC860 when IDMA data transfer is finished.

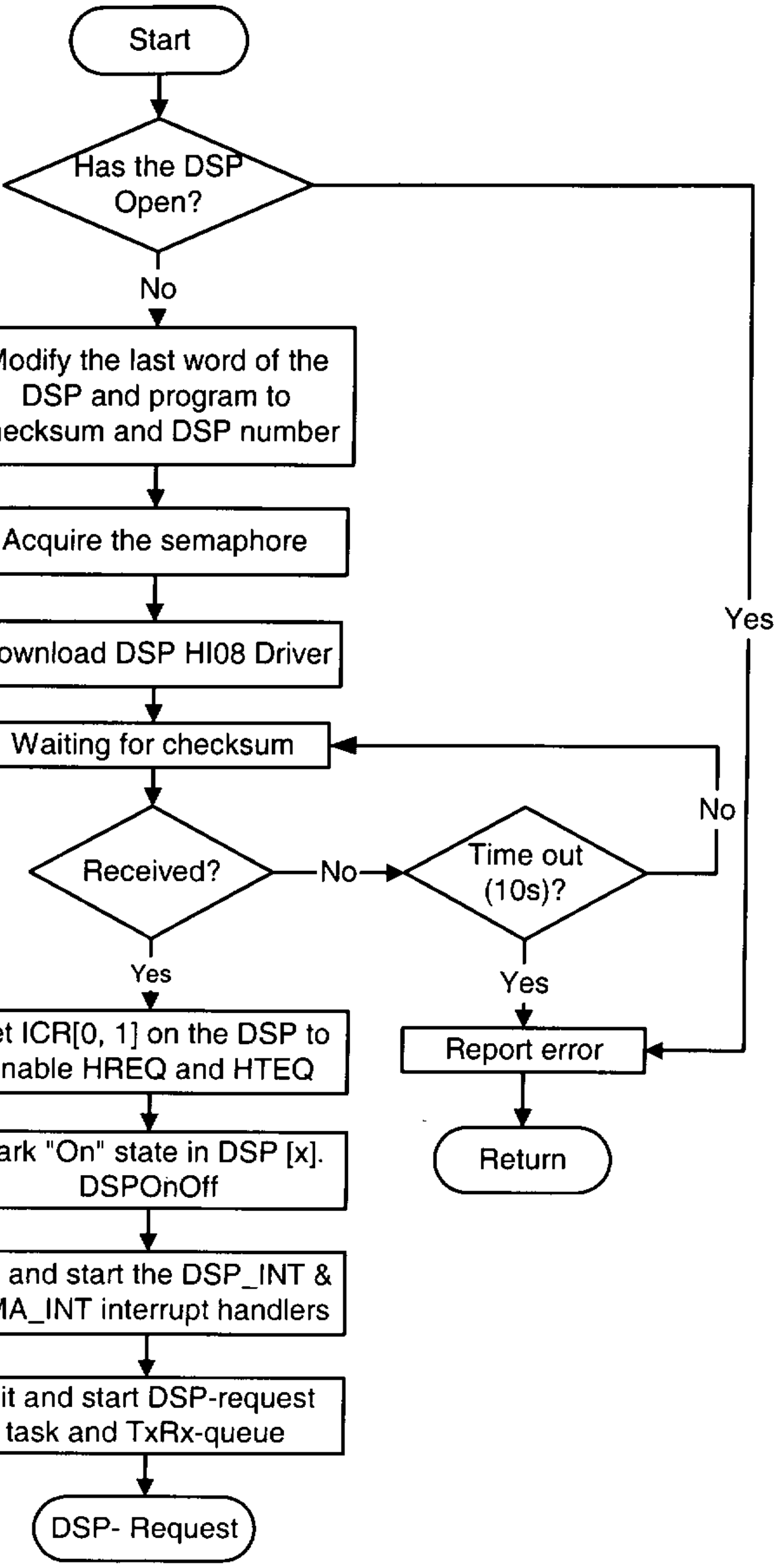

Start
Has the DSP Open?
No
Yes
Modify the last word of the DSP and program to checksum and DSP number
Acquire the semaphore
Download DSP HI08 Driver
Waiting for checksum
Received?
No
Time out (10s)?
No
Yes
Yes
Set ICR[0, 1] on the DSP to enable HREQ and HTEQ
Report error
Return
Mark "On" state in DSP [x]. DSPOnOff
Init and start the DSP_INT & DMA_INT interrupt handlers
Init and start DSP-request task and TxRx-queue
DSP- Request

Figure B- 2 Flowchart of Hopen

If an interrupt is received, MPC860 first checks whether a DSP_INT or DMA_INT is received from DSPs by reading the DSPV and DMAV bit in the FPGA Interrupt Pending Register (FPR).

If DSPV is set, interrupt handler of DSP_INT is called. Otherwise, if DMAV is set, interrupt handler of DMA_INT is called.

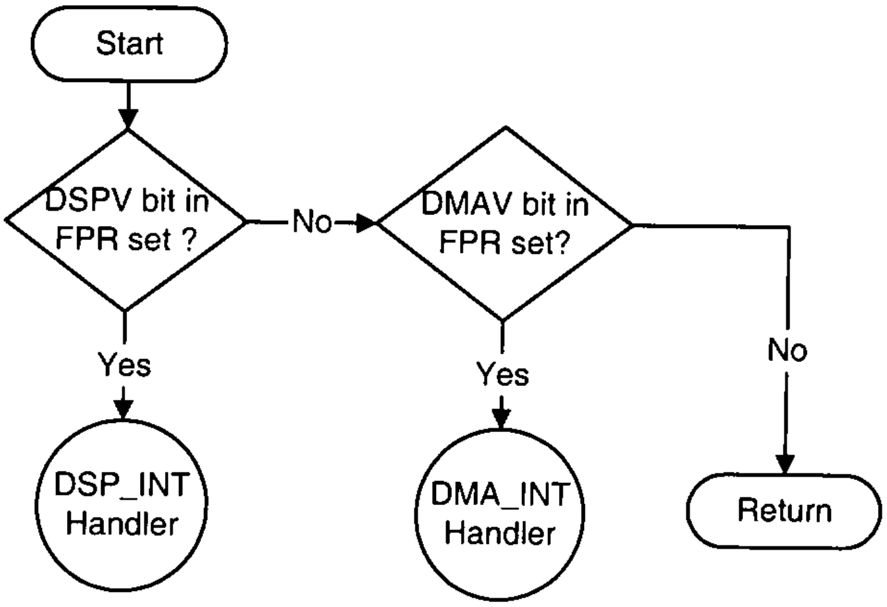


Figure B- 3 Flowchart of Interrupt Handler

4. Interrupt handler of DSP_INT

MPC860 first checks which DSPs have been requested for the data transfer by reading the FPGA Rx Pending Interrupt Register (FRXIPR).

If bits in FRXIPR are set, the corresponding DSP number and the direction (Rx) are sent as a pSOS queue message to the TxRx-request queue. The corresponding bits of FPGA Rx Interrupt Mask Register (FRXIMR) are masked to avoid the same DSP continuously interrupting the MPC860. Lastly, DSP_INT is cleared in the SIU Interrupt

Pending Register (SIPEND) of MPC860.

Following is the format of the messages sent to TxRx-request Queue, each message consists of 4 bytes:

	Nil	Nil	Nil	DSP Numbe
BYTE	0	1	2	3

Figure B- 4 Format of messages sent to TxRx-Queue

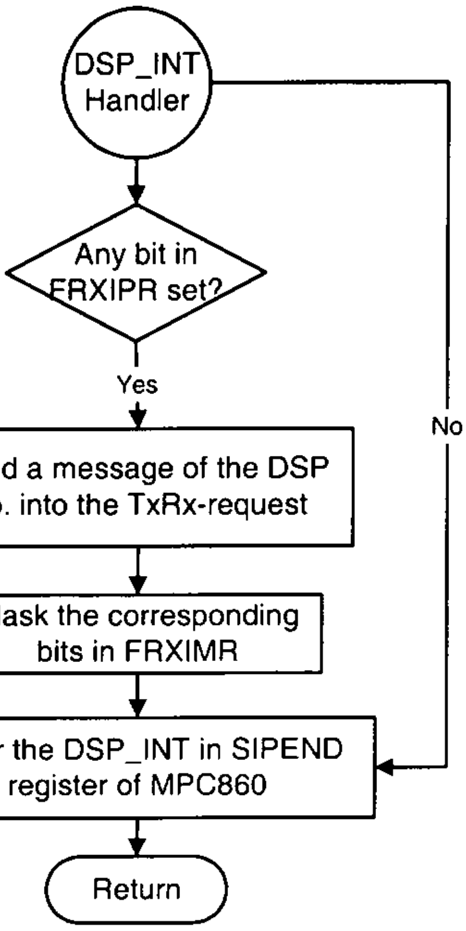

DSP_INT Handler
Any bit in FRXIPR set?
Yes
No
Send a message of the DSP no. into the TxRx-request
Mask the corresponding bits in FRXIMR
Clear the DSP_INT in SIPEND register of MPC860
Return

Figure B- 5 Flowchart of DSP_INT interrupt handler

5. Interrupt handler of DMA_INT

MPC860 first checks which DSPs finished the data transfer from the IDMA channel of MPC860 and its direction by reading the DSPn, and DIR bits in the FPGA Status Register (FSR). The corresponding DSPs number is unmasked in the FPGA Rx Interrupt Mask Register (FRXIMR). Lastly DMA_INT is cleared in the SIU Interrupt Pending Register (SIPEND) of MPC860.

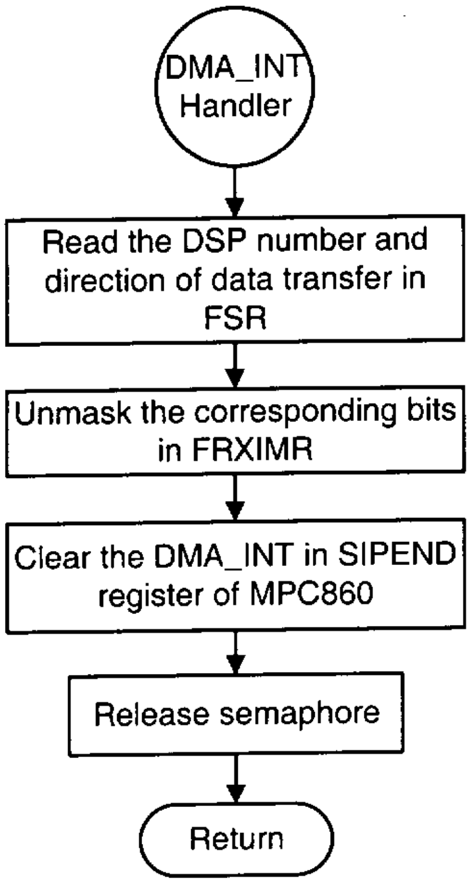

DMA_INT Handler
Read the DSP number and direction of data transfer in FSR
Unmask the corresponding bits in FRXIMR
Clear the DMA_INT in SIPEND register of MPC860
Release semaphore
Return

Figure B- 6 Flowchart of DMA_INT Interrupt Handler

6. DSP-Request Task

Receive header from DSPs Directly

When there is an entry in the TxRx-Queue, MPC860 starts data transfer between the MPC860 and DSPs.

MPC860 first gets the DSP number from the queue. Then, MPC860 gets the ISR (Interface Status Register) flags bit 4(HF3) and bit 3 (HF2) to indicate the direction of the data transfer. These two flags have a special format to show the task request type. MPC860 gets the header from DSP via direct interface.

	HF3	HF2
Data	0	0
Rx	0	1
Tx	1	0
Reserve	1	1

1 byte	2 bytes
Header	Byte Count

Figure B- 7 Request Type and Structure of the DSP header

Data Transfer between MPC860 and DSPs via data hub interface requested by DSPs.

MPC860 starts the data transfer to DSPs via data hub interface after the header is received or sent.

MPC860 checks whether the DMA channel are free by reading the BUSY and DONE bit in FPGA status register (FSR). If the channel is free, MPC860 initializes the DMA channel. Next, MPC860 write the FPGA control register (FCR) to specify the byte count, DSP number and transfer direction. Lastly, Go bit in FCR is set to enable the DMA transfer. i.e. If the byte count is not divisible by 12, it is increased to be a multiple of 12.

After downloading program, then set ICR (Interface Control Register) Bit 0

(RREQ) and Bit 1 (TREQ).

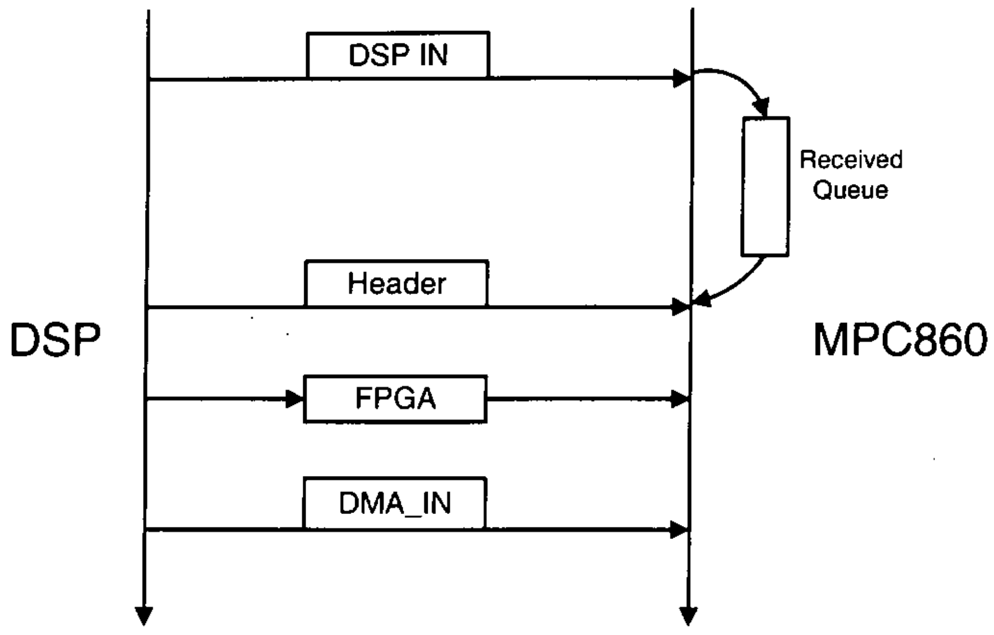


Figure B- 8 Flow of the IDMA Data Transfer from DSP to MPC860

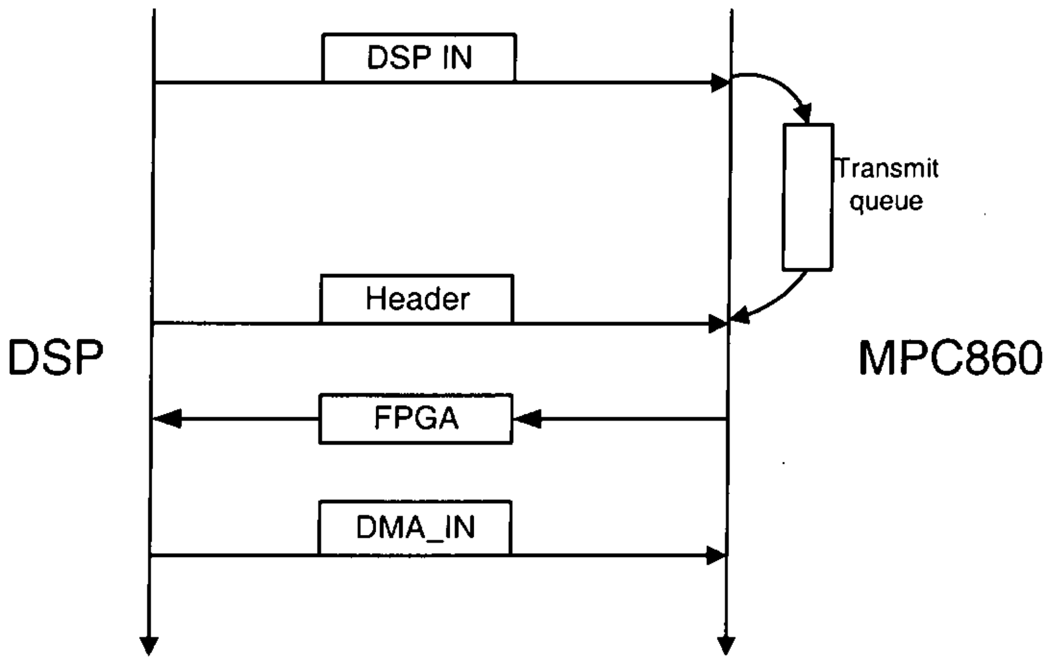


Figure B- 9 Flow of the IDMA Data Transfer from MPC860 to DSP

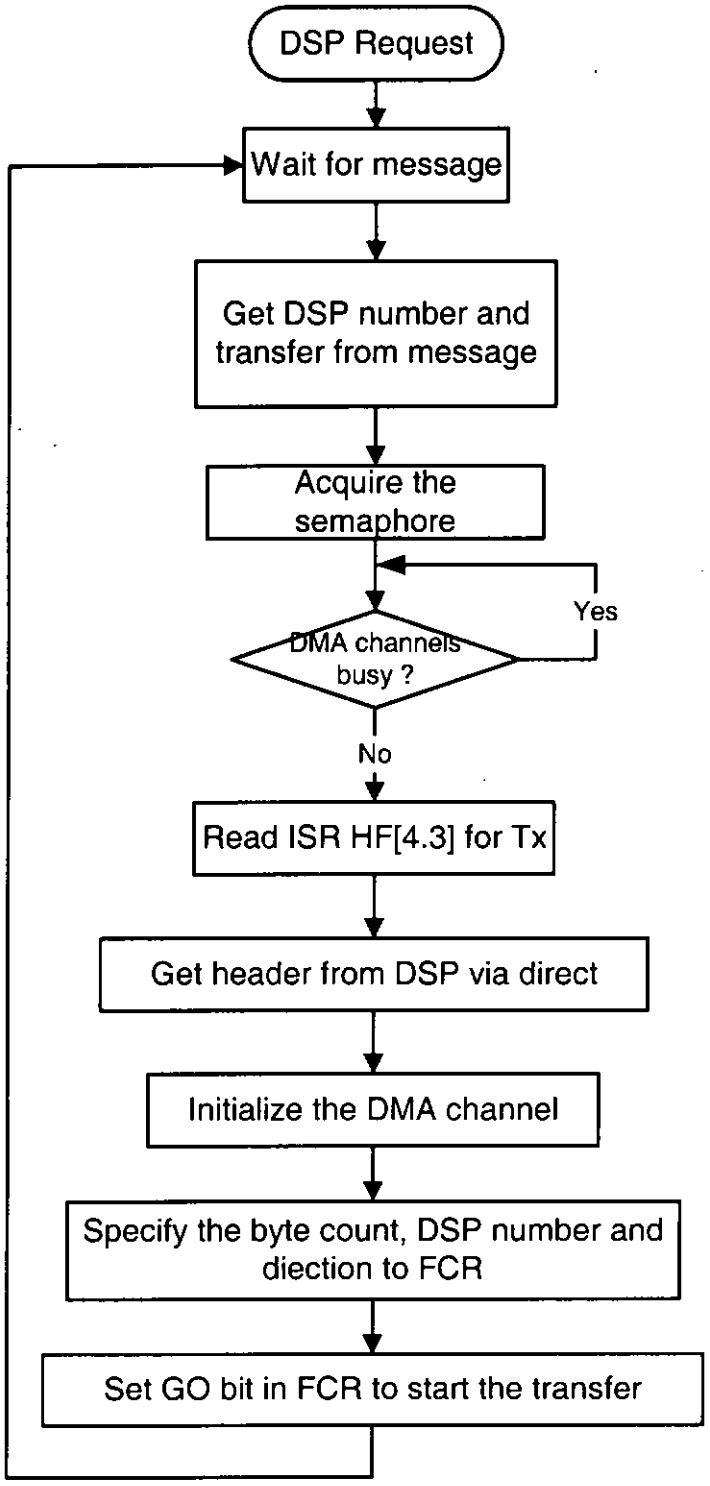

DSP Request
Wait for message
Get DSP number and transfer from message
Acquire the semaphore
DMA channels busy ?
Yes
No
Read ISR HF[4.3] for Tx
Get header from DSP via direct
Initialize the DMA channel
Specify the byte count, DSP number and diection to FCR
Set GO bit in FCR to start the transfer

Figure B- 10 Flowchart of data transfer between MPC860 and DSPs via data hub interface

7. de_close & H03Close

de_close() invokes the function H03Close(). H03Close() reset all DSPs and mask the DSP Interrupt.

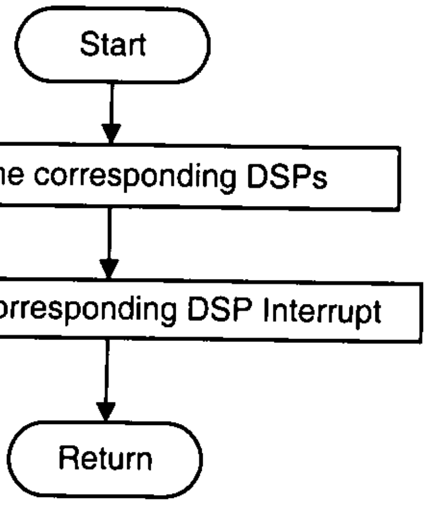

Start
Read the corresponding DSPs
Mask the corresponding DSP Interrupt
Return

Figure B- 11 Flowchart of Hclose

8. de_read & H03Read

de_read() invokes the function H03Read().

H03Read() calls the MPC860 to read, either from the registers of DSPs directly or from the Rx-FIFO. If DSP mode is used, MPC860 will read the register of the DSP directly. If FIFO mode is used, MPC860 will read data from the FIFO directly.

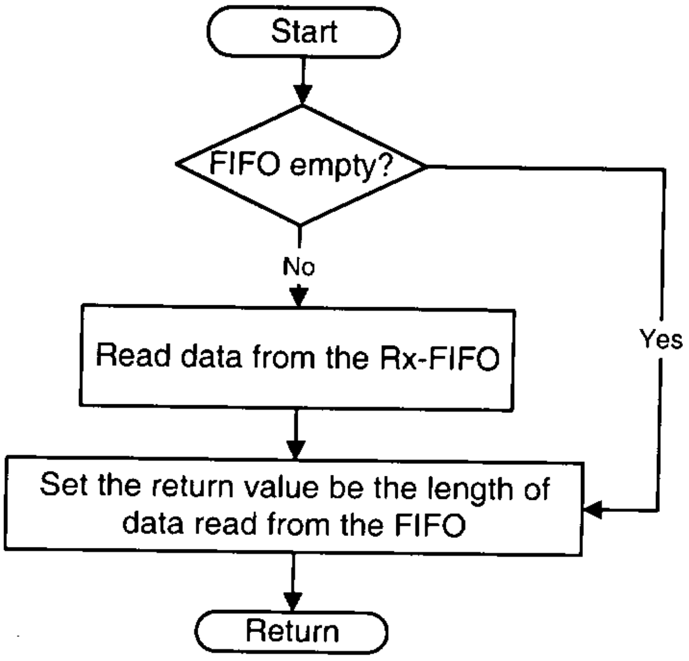

Start
FIFO empty?
No
Yes
Read data from the Rx-FIFO
Set the return value be the length of data read from the FIFO
Return

Figure B- 12 Flowchart of FIFO Read

FIFO Read

MPC860 reads the Rx-FIFO of the specified DSP. After data is read from Rx-FIFO, the length of data is returned to the user since it may not has enough data in the FIFO.

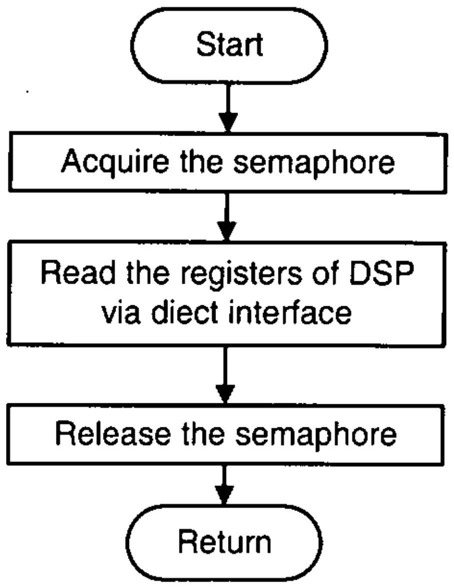


Figure B- 13 Flowchart of DSP Read

DSP direct READ

MPC860 read the registers of DSP directly. There are a total of 7 DSP registers:

- Interface Control Register (ICR)
- Command Vector Register (CVR)
- Interface Status Register (ISR)
- Interrupt Vector Register (IVR)
- Received/Transmit Bytes (RXH, RXM, RXL, TXH, TXM, TXL)
- A graceful IRQ stop should be called before DSP Read by de_cntrl(). A IRQ

start should be called after DSP Read by de_cntrl().

9. de_write & H03Write

de_write() invokes the function H03Write().

H03Write() calls the MPC860 to write, either the registers of DSPs directly or data to the Tx-FIFO. If DSP mode is used, MPC860 will write the register of the DSP directly. If FIFO mode is used, MPC860 will write data to the FIFO directly.

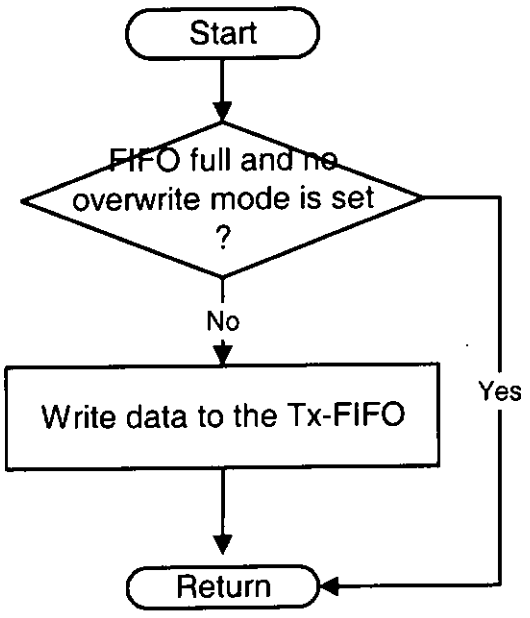

Start
FIFO full and no overwrite mode is set ?
No
Yes
Write data to the Tx-FIFO
Return

Figure B- 14 Flowchart of FIFO Write

FIFO Write

MPC860 writes the Tx-FIFO of the specified DSP. If the FIFO is full and the overwrite mode is set, data is overwritten to the Tx-FIFO. Otherwise, extra data is neglected.

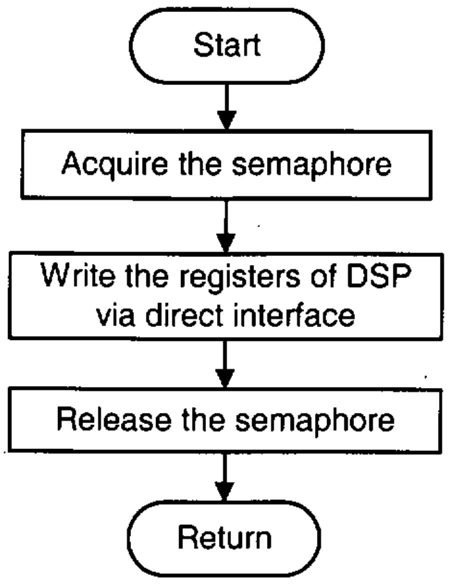


Figure B- 15 Flowchart of DSP Write

DSP direct Write

MPC860 writes to the registers of DSP directly. There are 7 DSP registers:

- Interface Control Register (ICR)
- Command Vector Register (CVR)
- Interface Status Register (ISR)
- Interrupt Vector Register (IVR)
- Received/Transmit Bytes (RXH, RXM, RXL, TXH, TXM, TXL)

10. de_cntrl and H03Cntrl

de_cntrl() invokes the function H03Cntrl().

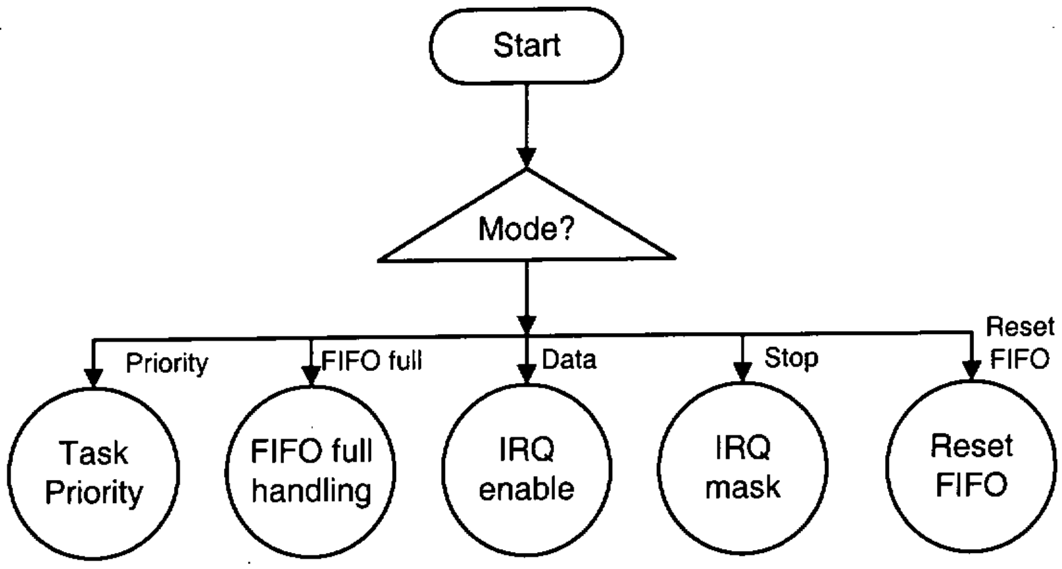


Figure B- 16 Flowchart of FIFO Reset

B.2 HI08 Interface Driver System Calls

de_init

Configure DSPs reset pins of MPC860 & reset all DSPs

```
#include <psos.h>
unsigned long de_init(
    unsigned long dev,          /* major/minor device number */
    void *iopb,                 /* I/O parameter block */
    void *retval,               /* return value */
    void **data_area            /* device data area */
)
```

Description

de_init() configures the DSPs reset pins of MPC860 and resets all DSPs. A semaphore per data hub is created to ensure only one task is using the driver. Also, de_init() initiates the internal data structure and reset FCR (FPGA Control Register) DSPn(31) bit for both FPGAs.

Argument

dev	Specify the device number of HI08 Interface driver
iopb	No Input parameters
retval	No return value
data_area	No longer used

Error Codes

Hex	Mnemonic	Description
0x101	ERR_IODN	Illegal device (major) number
0x102	ERR_NODR	No driver provided
0x103	ERR_IOOP	Illegal I/O function number

de_open

Download DSP programs & start interrupt handler for DSP

```
#include <psos.h>
unsigned long de_open(
     unsigned long dev,           /* major/minor device number */
     void *iopb,                  /* I/O parameter block */
     void *retval,                /* return value */
}
```

Description

de_open() downloads a DSP program to the specified DSPs and starts the interrupt handler for DSPs. DSP automatically runs after download. The task priority of the interrupt handler is set as 180 and it could be changed in de_cntrl.

Argument

dev	Specify the device number of HI08 Interface driver
iopb	Points to I/O parameter block of HI08 Interface driver

```
typedef struct {

    unsigned byte DSPNum[];       /* DSP Number (0-63)*/
    unsigned byte DSPProg[];      /* DSP Program Number */
    unsigned long NumEntry;       /* Number of entries   */
}H03Oiopb;
```

e.g. to open DSP 1,12, 45, 56 with DSP program 1,2,3,1

```
DSPNum[0]=1;          DSPProg[0]=1;

DSPNum[1]=12;         DSPProg[1]=2;

DSPNum[2]=45;         DSPProg[2]=3;

DSPNum[3]=56;         DSPProg[3]=1;

EntryNum=4;
```

retval returns pointer to an array which describes each DSPs status.

0 - OK. DSP opens successfully,

1 - Device not present,

2 - Device already opened

Error Codes

Hex	Mnemonic	Description
0x101	ERR_IODN	Illegal device (major) number
0x102	ERR_NODR	No driver provided
0x103	ERR_IOOP	Illegal I/O function number
0x104	ERR_OPEN	Device is already opened

de_close

Reset the DSP and mask the interrupt for DSP

```
#include <psos.h>
unsigned long de_close(
     unsigned long dev,        /* major/minor device number */
     void *iopb,               /* I/O parameter block */
     void *retval,             /*return value */
}
```

Description

de_close() resets the specified DSPs and masks the interrupt for DSP.

Argument

dev	Specify the device number of HI08 Interface driver
iopb	Points to I/O parameter block of HI08 interface driver

```
typedef struct {
      unsigned byte DSPNum[];          /* DSP Number (0-63)*/
      unsigned byte DSPProg[];         /* DSP Program Number */
      unsigned long NumEntry;          /* Number of entries   */
}H03Oiopb;
```

e.g. to close DSP 1,12, 45, 56 with DSP program 1,2,3,1

```
DSPNum[0]=1;         DSPProg[0]=1;
DSPNum[1]=12;        DSPProg[1]=2;
DSPNum[2]=45;        DSPProg[2]=3;
DSPNum[3]=56;        DSPProg[3]=1;
EntryNum=4;
```

retval	pass an array to indicate which DSP is closed

Error Codes

Hex	Mnemonic	Description
0x101	ERR_IODN	Illegal device (major) number
0x102	ERR_NODR	No driver provided
0x103	ERR_IOOP	Illegal I/O function number

de_read

get data from Rx-FIFO / read the registers of DSP directly

```
#include <psos.h>
unsigned long de_read(
    unsigned long dev,      /* major/minor device number */
    void *iopb,             /* I/O parameter block */
    void *retval,           /* return value */
)
```

Description

de_read() calls the MPC860 to read, either the registers of DSPs directly or data from the Rx-FIFO. I/O parameter *mode* is used to set which service is provided. If DSP mode is used, MPC860 will read the register of the DSP directly. If FIFO mode is used, MPC860 will read data from the FIFO.

If DSP mode is used, DSP Number and name of the register also need to be set in *DSPNum* and *RegName*. Transfer byte count does not needed to be set. It is automatically set to 1 byte for ICR, IVR, ISR, and CVR and 3 bytes for Data (ICR, CVR ISR, IVR, RXH, RXM, RXL).

If FIFO mode is used, DSP Number, byte count of data and destination address also need to be set in *DSPNum, length*, and *destAddr*.

Argument

dev	Specify the device number of HI08 Interface driver
iopb	Points to I/O parameter block of HI08 Interface driver

```
typedef struct {

    unsigned long DSPNum;      /* DSP Number (0-63)*/
    unsigned long Mode;        /* DSP -0 , FIFO - 1 */
    unsigned long Length;      /* FIFO Transfer byte count*/
/*multiple of 12 byte for FIFO counter*/
    void *      DestAddr;      /* Destination Address */
    unsigned long RegName;     /* Name of the DSP Register */
                               /* ICR, CVR, ISR, IVR, RXH, RXM,
RXL*/
}H03RWiopb;
```

retval if FIFO is used, the return value is the length of data read from the FIFO

Error Codes

Hex	Mnemonic	Description
0x101	ERR_IODN	Illegal device (major) number
0x102	ERR_NODR	No driver provided
0x103	ERR_IOOP	Illegal I/O function number

de_write

put data to Tx-FIFO / write the registers of DSP directly

```
#include <psos.h>
unsigned long de_write(
    unsigned long dev,    /* major/minor device number */
    void *iopb,            /* I/O parameter block */
    void *retval,          /* return value */
)
```

Description

de_write() calls the MPC860 to write the registers of DSPs directly or put data to the Tx-FIFO. I/O parameter *mode* is used to set which service is provided. If DSP mode is used, MPC860 will write the register of the DSP directly. If FIFO mode is used, MPC860 will put data to the Tx-FIFO.

If DSP mode is used, DSP Number and name of the register also need to be set in *DSPNum* and *RegName*. Transfer byte count does not needed to be set. It is automatically set to 1 byte for ICR, IVR, ISR, and CVR and 3 bytes for Data (ICR, CVR ISR, IVR, RXH, RXM, RXL).

If FIFO mode is used, DSP Number, byte count of data and destination address also need to be set in *DSPNum, length*, and *destAddr*.

Argument

dev	Specify the device number of HI08 Interface driver
iopb	Points to I/O parameter block of HI08 Interface driver

```
typedef struct {

    unsigned long DSPNum;      /* DSP Number (0-63)*/
    unsigned long Mode;        /* DSP -0 , FIFO - 1 */
    unsigned long Length;      /* FIFO Transfer byte count */
    /* multiple 12 bytes for FIFO, not for DSP*/
    void *SourceAddr;          /* Source Address */
    unsigned long RegName;     /* Name of the DSP Register */
                     /* ICR, CVR, ISR, IVR, TXH, TXM, TXL*/
}H03RWiopb;
```

retval If FIFO mode is used, return value is the length of data written to the FIFO.

Error Codes

Hex	Mnemonic	Description
0x101	ERR_IODN	Illegal device (major) number
0x102	ERR_NODR	No driver provided
0x103	ERR_IOOP	Illegal I/O function number

de_cntrl Request special services

```
#include <psos.h>
unsigned long de_cntrl(
   unsigned long dev,        /* major/minor device number */
   void *iopb,               /* I/O parameter block */
   void *retval,             /* return value */
```

Description

de_cntrl() provides the following special services in the driver. Different parameter provides different job to be performed:

1. FIFO reset

   Parameter *Mode* is set as *FIFO*. Reset and clear the DSP Tx/Rx FIFO. DSP Number needs to be set in parameter *DSPNum*.

2. interrupt handler mask

   Parameter Mode is set as *mask*. Mask the IRQ interrupt handler of MPC860.

3. interrupt handler enable

   Parameter *Mode* is set as *Enable*. Enable the IRQ interrupt handler of MPC860.

4. FIFO full handling

   Parameter *full* is set to *Overwrite* if old data is overwritten or is set to *Reject* if new data is rejected. Parameter Mode is set to FIFO *Full*. Set whether overwrite the old data in FIFO or reject new data input if the FIFO is full.

5. HI08 Task priority setting

   Parameter *Mode* is set as *Priority*. Set the priority of IRQ interrupt handler. Priority needs to be set in parameter *Priority*, and its default value is 180.

6. serving priority (not implemented in this stage).

   set the serving priority of the DSP interrupts.

Argument

dev	Specify the device number of HI08 Interface driver
iopb	Points to I/O parameter block of HI08 Interface driver

```
typedef struct {
    unsigned long DSPNum;      /* DSP Number (0-63)*/
    unsigned long Mode;       /* FIFO reset-0,Mask-1,Enable-2/
                               /* FIFO Full-3,priority-4 */
    unsigned long Full;       /* Overwrite-0 , Reject - 1 */
    unsigned long Priority;    /* default set as 180 */
}HCNiopb;
```

retval	No return value

Error Codes

Hex	Mnemonic	Description
0x101	ERR_IODN	Illegal device (major) number
0x102	ERR_NODR	No driver provided
0x103	ERR_IOOP	Illegal I/O function number