

Clock Synchronization in Satellite, Terrestrial and IP Set-top Box for Digital Television

THESIS

Submitted in partial fulfilment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

by

MONIKA JAIN

Under the Supervision of

Prof. P.C. Jain



**BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE
PILANI (RAJASTHAN) INDIA
2010**

**BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE
PILANI (RAJASTHAN)**

CERTIFICATE

This is to certify that the thesis entitled “**Clock Synchronization in Satellite, Terrestrial and IP Set-top Box for Digital Television**” submitted by Mrs. Monika Jain, ID. No. 2006PHXF427 for award of Ph.D. Degree of the Institute, embodies original work done by her under my supervision.

PROF.PREM CHAND JAIN
HEAD & Professor- School of Electronics.
CDAC- Noida(U.P)

Date: 29 March 2010

ACKNOWLEDGEMENTS

This thesis could have been completed by grace of GOD and with the support, help and encouragement of many people that have been by my side during this experience.

I am immensely thankful to Prof. L. K. Maheshwari, Vice-Chancellor, BITS, Pilani for providing me this opportunity to pursue the off-campus PhD of the Institute. I express my gratitude to Prof. Ravi Prakash, Dean, Research and Consultancy Division (RCD), BITS, Pilani for his constant official support, encouragement and making the organization of my research work through the past few years easy.

I thank Dr. Hemanth Jadav, Mr. Dinesh Kumar, Ms. Monica Sharma, Mr. Sharad Shrivastava, Mr. Gunjan Soni, Mr. Amit Kumar and Ms. Sunita Bansal, nucleus members of RCD, BITS, Pilani, without whose cooperation and guidance it would not have been possible for me to pursue such goal oriented research during each of the past few semesters.

I also express my gratitude to the office staff of RCD whose secretarial assistance helped me in submitting the various evaluation documents in time and give pre-submission seminar smoothly.

I thank my Doctoral Advisory Committee (DAC) members, Dr. Anu Gupta and Dr. S.K. sahoo, who spared their valuable time to go through my draft thesis and were audience to my pre-submission seminar in order to provide several valuable suggestions that immensely helped in improving the quality of my PhD thesis report.

I wish to express my deep sense of gratitude and sincere thanks to my thesis supervisor, Dr. P.C. Jain, Professor & Head, School of Electronics, CDAC, Noida, for having hosted me in his group, for his able guidance, encouragement and suggestions in my thesis. It has been a privilege for me to work under his guidance. I am immensely indebted to Dr. Jain for the enthusiasm he transmitted, for his competence, as well as for the richness of his guidelines and his invaluable suggestions. The time he spent throughout my research period with me in discussing and reviewing my work was extremely precious. I am highly grateful to Dr. Jain for his consent to become my supervisor.

Gratitude is also accorded to STMicroelectronics Pvt. Ltd. Gr. Noida for providing all the necessary support to complete the research work. I am indebted to Mr. Vivek Sharma, Director, STMicroelectronics and Mr. Vivek Khaneja, Department Manager, HVD R&D IP Development, STMicroelectronics, Gr. Noida, to enhance knowledge in Set-top box domain, for Joint Authorship of IP with ST and approving the student sponsorship for my Ph.D work.

I would be immensely pleased to thank Mr. John Mark Benamount, Architect in STMicroelectronics, Bristol, U.K. and Mr. Vincent Capony, Architect in STMicroelectronics in Grenoble, France for being the guiding force during the work for helping me to get familiarized with various standards and drivers of the set-top box. His invaluable suggestions and continuous check on the progress of the project proved to be indispensable. I would like to express my sincere gratitude towards Mr. Sharad Jain, Section Manager, STMicroelectronics, for his invaluable guidance, interaction and untiring efforts throughout the research work. This has helped me from time to time at various stages of the research work, and showing me the way out whenever I got stuck.

I am extremely thankful to Dr. Ashok K. Chauhan, Founder President Ritnand Balved Education Foundation (RBEF) , Maj. Gen. K. Jai Singh, Vice-Chancellor, Amity University, UttarPradesh, Noida for encouraging me for the Research work. I cannot forget the immense help and support provided by Dr. Balvinder Shukla, Pro-VC, Amity University and Director-General-ASET, Noida for giving me directions which ultimately proved to be very important and for inspiring me to complete the work. Her indefatigable efforts were a big motivation to perform efficiently.

I am grateful to Prof. Dinesh Chandra, HOD - ECE at JSSATE, Mr. K.M Soni, HOD at ASET, Noida for their ineffaceable contribution and regular help. I am also indebted to my friends and relatives that have shared with me this experience from the beginning: Mr. K.P. Singh , Mr. Akshat Jain, Mr. Ankit Jain, Mr. Avdesh Kumar, Dr. Arun Singh , Ms. Anjula, Ms. Dimpy, Ms Kshama Jain who have contributed in many ways to complete my research work.

Special thanks are due to library staff at JSSATE and at AMITY University for making timely availability of various books, on-line journals. The support extended by librarian of IIT, New Delhi for literature survey is also gratefully acknowledged.

My family deserves a special gratitude for the patience and the attention that I enjoyed: it was a precious assistance throughout this period. Finally a very special expression of appreciation is extended to my parents Mr. Vinod Kumar Jain and Mrs. Sheela Jain without their encouragement, patience, and understanding this endeavor would not have been possible. I would like to record my special affection to my son and my daughter and very special thanks to my husband Sharad Jain, and my sister Ms. Sonia Jain whose constant persuasion and moral support has been a source of inspiration to me.

[Monika Jain]

TITLE OF THE THESIS: “Clock Synchronization in Satellite, Terrestrial and IP Set-top Box for Digital Television ”

ABSTRACT:

With the increasing trend and demand of digital television in today’s digital world, set-top box (STB) is becoming more and more popular due to broadcast quality pictures and lot more flexibility. STB follows the MPEG-2 digital-compression standard, which is the basis of both the Digital Video Broadcast standard (DVB) and the Advanced Television Systems Committee (ATSC) standard. MPEG-2 allows the delivery of any information that can be digitized, including audio, video and text. MPEG-2 stream can be easily displayed on High Definition television sets (HDTV) in analog NTSC, PAL or SECAM formats with help of set-top box. Thus MPEG streams in a real-time software manner are gaining more and more importance. Clock synchronization is always an important part of a MPEG system. It is even desirable to extract the timing information where the decoder obtains its input from a controllable digital medium such as a CD-ROM. However it is not always necessary in such applications. But clock synchronization becomes very critical in designs where the compressed input to the decoder arrives from an uncontrollable real-time channel such as satellite transmission, cable transmission, terrestrial transmission, IP transmission.

The Clock- Synchronization has two concepts : **Clock Recovery**, and **A/V sync (Audio-Video Sync)**. Clock recovery is a method to recover the encoder clock in order to build a local clock running at the same frequency (+/-offset). This module also provides local STC (reference clock) for video & audio driver use. Due to accurate clock recovery video and audio streams are played at the same pace as they are processed at encoder side. Both clocks (encoder & decoder) are synchronized. Video and audio bit buffers stay correctly filled hence no under- or overflow.

However, AV sync is a method to synchronize audio and video mutually e.g. the sound heard by the viewer matches the pictures seen (e.g. character speaking, door banging closed, ...etc.)

Correct clock synchronization at Set-Top Box (STB) receiver side is mandatory for smooth operation of MPEG decoder and correct user experience because loss of synchronization between the received and the transmitted clocks can lead to degradation in system performance like color loss, jerky video, audio dropouts etc. There are several factors, which lead to degradation in clock recovery process. Some of these are like frequency drift between the transmitter and the receiver clock, network induced jitter, inner interleaving and de-interleaving blocks in the transmitter and receiver side, packetization jitter etc. Many times worse channel conditions (such as $\frac{1}{4}$ guard interval, channel fading, low carrier-to-noise ratio etc.), the tuner front-end takes much more time to detect and demodulate the signal and this can delay the arrival of Program Clock Reference (PCR) packets. The jitter in DVB Broadcasting is high and it vary dominantly depending upon transmission media. It has been analyzed that jitter is less in satellite transmission, high value in terrestrial and very high in IP transmission. .

Our thesis is addressing the solution for problem of clock recovery and AV sync both. The core contribution is to present suitable clock recovery modules to be implemented in digital set-top box for three different transmission mediums such as DVB-S, DVB-T and DVB-IP. Three different schemes for AV synchronization and some enhancements in existing approaches has also been proposed.

The first contribution of this thesis is to achieve clock recovery for direct satellite broadcast or Direct-to-Home (DTH) television to receive digital TV signal. In this we proposed an economical and very simple, real time software approach to solve the clock synchronization problem. A basic moving window averaging filter approach has been

described. Our methodology is evaluated by both analysis and extensive simulation experiments in a satellite broadcasting environment. Computer simulations verify the design approach illustrated.

The second contribution of our research work is presenting a low cost, low complexity and efficient clock recovery module to solve the color loss problem in Digital Set-top Box used for terrestrial television. In Digital Video Broadcasting for terrestrial environment (DVB-T) there is large jitter due to multipath reflections and other stochastic losses. In digital set-top box most of the CPU bandwidth is consumed in audio & video data decoding so clock synchronization algorithms get very less CPU time to execute. Existing Set-Top Boxes experiences very large jitter in terrestrial environment due to multi-path reflections. Our main aim is addressing the effects of jitter on MPEG-2 Transport streams for DVB-T. “Moving window weighing FIR filter” approach has been used in our module. Our enhanced FIR filter approach solves the color loss problem due to clock de-synchronization. It aims to be stable with jittery Program Clock Reference (PCR) and achieves synchronization quickly. Its performance has been compared with Linear Regression based algorithm and shows distinct advantages. This methodology is evaluated by both analysis and extensive simulation experiments for satellite and terrestrial broadcasting environment. Simulations results also verify the design approach. Proposed module was also implemented on a set-top box SOC (system-on-chip) and it demonstrated high performance.

The third core contribution is to present a new technique for Clock Recovery in MPEG-2 Transport Streams in DVB-IP environment. With the increasing penetration of packet-switching network technologies that provide asynchronous network services, and the simultaneous necessity of accommodating synchronization-sensitive data types (i.e., voice, audio and video), the problem of keeping end-systems mutually synchronized is becoming critical and more complex than in traditional circuit-switching networks. This complexity is

due to the fact that stochastic packet transmission delay jitter may lead to significant variation in packet arrival time. Eventually this makes clock synchronization much more difficult. The amount of jitter in this environment is very high compared to DVB-S and DVB-T. Researchers in the similar field suggest use of linear Regression algorithm for Clock synchronization over packet switched networks. However, the stability of such a filter in Real Time set-top box application posed a problem. A newly devised approach of “Continuous adaptive feedback loop used with the Linear Regression Algorithm” provides accurate clock synchronization even when subjected to jitter of half a second. This robustness is needed for IP environment because of the increasing network traffic. The algorithm was simulated and found to require low computation time, high jitter attenuation and fast response as needed for the IP network. It is auto-adaptive for both low and high jitter environments by analyzing the jitter present in the stream. The algorithm was implemented for Real time application in IPTV set-top box and tested to be stable in the RTOS environment.

The fourth main contribution is presenting certain enhancements for AV Sync (Audio Video synchronization) in digital STB (Set-Top Box). Audio Video synchronization has always proved difficult to implement in STB. The main reason of AV Sync problem is difference in processing path of audio and video. Simultaneous synchronization of audio with HD (High Definition) and SD (Standard Definition) output impose very tight constraints. Researchers in the similar field have suggested various models for Correct AV Sync implementation. However, none of them addresses the AV Sync problem of dual video (HD & SD simultaneously) with audio. Set-top box (STB) has not been considered as possible application for their implementation so such models are not suitable for Real Time Systems. Detailed analysis of various STB applications scenarios and their tight constraints has been carried out. The current AV Sync implementations of STB have been analyzed and

certain enhancements in current software implementations have been proposed. In this thesis we have also proposed few novel synchronization models that offer better synchronization accuracy. These models have been implemented in Real time application in set top box. Testing of proposed implementation in the RTOS environment meets STB AV sync constraints.

TABLE OF CONTENTS

	Page No
Acknowledgements	i
Abstract	iii
List of Figures	xv
List of Tables	xix
List of Symbols and Abbreviations	xx
Chapter 1 Introduction	
1.1 Background of the Clock Recovery Problem	1
1.2 Background of the A/V sync Problem	5
1.3 Objectives of the Thesis	6
1.4 Methodology adopted for the Research Work	7
1.5 Scope of the Thesis	9
1.6 Organization of the Thesis	11
Chapter 2 Literature Review	
2.1 Literature review of Clock recovery for Satellite and Terrestrial Broadcasting	14
2.2 Literature review of Clock recovery for IPSTB Broadcasting	18
2.3 Literature review of Audio Video Synchronization	30
2.4 Research gap identified in existing clock recovery algorithms	31
2.5 Research gap identified in existing AV sync Implementations	33
Chapter 3 MPEG Standard	
3.1 Introduction	35
3.2 MPEG2 Standard Overview	37
3.3 Part of MPEG2 Standard	39

TABLE OF CONTENTS (Contd...)

3.4 MPEG2 System Layer	41
3.4.1 Elementary Stream (ES)	43
3.4.2 Packetized Elementary Stream (ES)	43
3.5 MPEG2 Multiplexing Layer	47
3.5.1 MPEG Transport Stream	49
Chapter 4 Set-Top Box System	
4.1 Introduction	59
4.2 Set-Top Box components	60
4.2.1 The Network Interface	60
4.2.2 The Decoder	61
4.2.3 The Buffer	61
4.2.4 Synchronization hardware	61
4.3 Set-Top Box function	62
4.4 Functioning of STX7109 HDTV Set-Top Box decoder chip	64
Chapter 5 Clock recovery for Satellite TV	
5.1 Introduction to Clock synchronization	69
5.1.1 Voltage controlled Oscillator based clock recovery module	73
5.1.2 FS based clock recovery module	74
5.2 Analysis of Jitter in DVB-S Environment	74
5.3 Moving window Basic Averaging algorithm FOR DVB-S Broadcasting	75
5.3.1 Filter parameters	76
5.3.1.1 PCR_DRIFT_THRESHOLD	77
5.3.1.2 MIN_SAMPLE_NUM	77

TABLE OF CONTENTS (Contd...)

5.3.1.3 MAX_WINDOW_SIZE	78
5.3.2 Error calculation and applying corrections	78
5.4 Performance of Moving window Basic Averaging algorithm	81
5.5 Conclusion	83
Chapter 6 Clock recovery for Terrestrial TV	
6.1 Analysis of Jitter in DVB-Terrestrial Environment	84
6.2 Limitation of DVB Terrestrial Broadcasting	86
6.3 Moving window Weighting FIR filter algorithm for DVB-T Broadcasting	88
6.3.1 Low Pass Filter	89
6.3.2 Moving window Weighting FIR filter algorithm	89
6.3.3 Applying Corrections	91
6.4 Closed loop analysis of proposed algorithm	92
6.5 Performance of proposed algorithm for Terrestrial Broadcasting	96
6.6 Conclusion	99
Chapter 7 IP Streaming: Detailed Analysis in IP Environment TV	
7.1 IP Network Interface	101
7.1.1 Networking Technologies	101
7.2 IP Protocols suite	102
7.2.1 Unicast and Multicast	102
7.2.2 IP Protocols Byte and Bit Representation	103
7.2.3 Transport Protocols	104
7.3 Protocols stacks	108

TABLE OF CONTENTS (Contd...)

7.3.1 Stack Representation	108
7.3.2 Data Encapsulation	108
7.3.3 Protocol Stack Implementation	109
7.4 AV Content Transport format	112
7.4.1 MPEG Transport Stream based format	113
7.5 IP streaming receiver software	115
7.5.1 Server Push or Client Pull	115
7.5.2 MPEG TS based Receiver Software Stack	116
7.5.3 RTP Elementary Stream based Receiver Software Stack	116
7.6 IP streaming receiver clock synchronization	118
7.6.1 Clock Synchronization for TS based Transport	119
Chapter 8 Clock recovery for IP TV	
8.1 Analysis of Jitter and color loss in IP environment	121
8.2 New proposed Algorithm for clock recovery in IP Set-top box	125
8.2.1 Proposed algorithm	126
8.3 Algorithm enhancements	129
8.3.1 Piece-wise Linear Adaption	130
8.3.2 Over-lapped Piece-wise Linear Adaption	130
8.3.3 Continuous Adaption technique	130
8.3.3.1 Exponential Weighting	131
8.3.3.2 X Co-ordinate Shift	132
8.3.3.3 Parameter Scaling	132
8.4 Proposed feedback loop	133
8.4.1 System response	135

TABLE OF CONTENTS (Contd...)

8.4.1.1 Over-Damping	136
8.5 Matlab simulation of Proposed algorithm for IP environment	137
8.6 Performance analysis of new proposed LR algorithm in real time IP environment	141
8.6.1 System Matlab simulation with real time data	149
8.6.1.1 Matlab Simulation of Real time data for 0 ms jitter	149
8.6.1.2 Matlab Simulation of Real time data for 20 ms jitter	151
8.6.1.3 Matlab Simulation of Real time data for 100 ms jitter	154
8.6.2 Results of Real time IP streams for varying jitter	156
8.6.2.1 Results of Real time IP streams for 0 ms jitter	157
8.6.2.2 Results of Real time IP streams for 20 ms jitter	157
8.6.2.3 Results of Real time IP streams for 50 ms jitter	159
8.6.2.4 Results of Real time IP streams for 80 ms jitter	159
8.6.2.5 Results of Real time IP streams for 100 ms jitter	159
8.7 Performance Comparison of new proposed LR algorithm with old LR algorithm	160
8.8 Conclusion	162

Chapter 9 Audio Video Synchronization

9.1 Introduction to Audio Video synchronization (AV sync)	163
9.2 Audio Video synchronization definition	166
9.3 Proposed enhancement in existing schemes	170
9.3.1 Improving timing accuracy	170
9.3.1.1 Timing improvements by local clock tick quantization	170

TABLE OF CONTENTS (Contd...)

9.3.1.2 Timing improvements by correct interrupt handling	171
9.3.1.3 Timing improvements by software scheduling Optimization	171
9.3.2 Improving quantization accuracy	171
9.3.2.1 PTS Quantization	171
9.3.2.2 Arithmetic Quantization	172
9.3.2.3 Audio Quantization	172
9.3.2.4 Video Quantization	172
9.4 Proposed AV sync design variation	173
9.4.1 Minimize Control Complexity	175
9.4.2 Minimize High Quality Error	176
9.4.3 Minimize Memory Requirements	178
9.5 Test setup for AV sync problem	179
9.6 Conclusion	180
Chapter 10 Results and Conclusions	
10.1 Summary of results	181
10.2 Conclusions	186
10.3 Specific Contributions	187
10.4 Further Scope of Work	189
References	191
Appendices	
I. Application of Set-Top Box decoder chip	A-1
II. Clock recovery algorithm simulation in matlab	A-7

III. Signaling and management protocols for IP streams	A-14
III. Signaling and management protocols for IP streams	A-23
List of Publications	A-1
Brief Biography of the Candidate	A-1
Brief Biography of the Supervisor	A-1

LIST OF FIGURES

Figure No	Title	Page No
1.1	Methodology Adopted for Present Work	9
2.1	Block diagram of a PLL used in MPEG decoder	20
2.2	LLR based Clock Recovery Method	24
3.1	Evolution of MPEG	35
3.2	Summary of MPEG Compression Capability	39
3.3	Formation of MPEG stream	42
3.4	PES Packet Schematic	43
3.5	Detailed structure of PES Packet	45
3.6	Functional Block diagram of Standard MPEG2	49
3.7	MPEG Transport Stream	50
3.8	Detailed Syntax of Transport stream	52
3.9	Three types of TS Packets	53
3.10	Example of PSI Tables	57
3.11	PSI Table Structures	57
4.1	Schematic of Digital Set-Top Box	63
4.2	7109 block diagram	66
5.1	PCR mechanism	70
5.2	PCR mechanism in Transport Stream	71
5.3	Timing Diagram in a Typical Set-top Box	72
5.4	VCXO based clock recovery	74

Figure No	Title	Page No
5.5	Generalized Design of PLL based Clock Recovery	76
5.6	Concept of Moving Window	79
5.7	Behavior of algorithm as simulated in Matlab for Satellite data with P=50, S=10, M=80	82
5.8	Input PCR Count vs Decoder Frequency graph demonstrate behavior of Proposed Algorithm in Satellite Environment	83
6.1	Hardware Setup	85
6.2	PCR samples Arrival time in Terrestrial Environment	87
6.3	Frequency Error Samples in Terrestrial Environment	87
6.4	Feedback Loop expressed as Z transform	94
6.5	Behavior of simulated algorithm in MATALAB for terrestrial data with P=1000, S=50, M=150.	96
6.6	Behavior under G.I. = 1/8 , 64 QAM, FEC=3/4 terrestrial environment(jitter = 0.4 ms)	98
6.7	Behaviour under G.I. = 1/4 , 64 QAM, FEC= 7/8 terrestrial environment(jitter = 1.2 ms)	99
7.1	IP Network Stack Representation	109
7.2	IP Data Encapsulation	110
7.3	Protocol Stack Implementation	111
7.4	MPEG TS based Receiver Software Stack	117
7.5	RTP ES based Receiver Software Stack	118
7.6	PCR vs STC	120
8.1	Jitter distribution	122
8.2	Jitter and Latency in IP environment	124

Figure No	Title	Page No
8.3	Principle of Enhanced LLR algorithm for Clock Recovery in IP environment	127
8.4	Clock Recovery module using Enhanced Linear Regression	135
8.5	Quantized Frequencies in Response To 20ms Jitter	136
8.6	Linear Comparison of Different order of Jitter distribution Model	138
8.7	Performance of new LR algorithm in Matlab environment (Pareto-2 for jitter 18 ms with FS=3 OD= 4.83)	139
8.8	Performance of new LR algorithm in Matlab environment (Pareto-2 for maximum delay of 35 ms with FS=3, OD= 9.38)	140
8.9	Performance of new LR algorithm in Matlab environment (Pareto-2 for jitter 52 ms with FS=3 OD= 18.73)	140
8.10	Performance of new LR algorithm in Matlab environment (Pareto-2 for jitter 70 ms with FS=5 OD= 18.73)	141
8.11	IP-STB clock recovery test setup	142
8.12	Plot of quantized decoder frequency for real time data having 0 ms jitter	149
8.13	Plot of quantized decoder frequency in matlab for real time data having maximum jitter=20 ms	153
8.14	Plot of quantized decoder frequency in matlab for real data having maximum jitter=100ms	156
8.15	Plot of PCR count and quantized decoder frequency for 0 ms jitter in real time IP stream	157
8.16	Plot of PCR count and quantized decoder frequency for 20 ms jitter in real time IP stream	158
8.17	Plot of PCR count and quantized decoder frequency for 50 ms jitter in real time IP stream	158
8.18	Plot of PCR count and quantized decoder frequency for 80 ms jitter in real time IP environment	159

Figure No	Title	Page No
8.19	Plot of Decoder Frequency vs time elapsed in a VBR stream 2-nd order pareto 100 ms jittery environment (FS =9).	160
8.20	Performance comparisons of old LR algorithm with new enhanced LR algorithm for VBR (max 33.7 Mbps) 2-nd order pareto jitter distribution.	161
9.1	Standard MPEG system: Full clock recovery is mandatory with PCR, Lip Sync with PTS/STC	164
9.2	One program in Full screen+PIP (Picture in Picture) Full clock recovery+ Lip Sync with PTS/STC for full screen program, PIP is either free running or video synced.	164
9.3	Video only (Mosaic): clock recovery is mandatory , but Lip sync is not needed	164
9.4	Audio only: Full clock recovery is mandatory , but Lip sync is not needed	164
9.5	DVR(recorded system) : clock recovery is mandatory , but Lip sync is not needed	164
9.6	Audio errors with respect to slower video rate	167
9.7	General Display Timing	173
9.8	Minimize control complexity	176
9.9	Minimize FR Error	177
9.10	Minimizing memory requirement	178
9.11	Audio Video synchronization testing setup	180
A 1.1	Low cost satellite HD Set- Top Box	A-2
A 1.2	Low cost dual satellite & terrestrial HD Set- Top Box with HDD & DVD	A-3
A 1.3	Low cost cable HD Set- Top Box with return channel	A-4
A 1.4	Low cost HD IPTV Set- Top Box	A-6

LIST OF TABLES

Table No	Description	Page No
3.1	PES Scrambling control	45
3.2	PSI Table	54
3.3	Example of PAT for Three Program Multiplex	55
3.4	Example PMT for Program 2	55
3.5	Example CAT for Program 2	56
6.1	Effect of Terrestrial Parameters on Jitter	97
8.1	Test1- SportsVBR3.8 - NoShunra	144
8.2	Test2- AllVBR38 – No Shunra	145
8.3	Test3- SportsCBR20 – No Shunra	146
8.4	Test4-SportsCBR40-NoShunra	147
8.5	Real Time PCR- STC data set (0 ms jitter)	150
8.6	Real Time PCR- STC data set (20 ms max jitter)	151
8.7	Real Time PCR- STC data set (100 ms max jitter)	154
8.8	Performance Comparison of OLD LLR and enhanced LLR	162
9.0	STB customer's Audio Video Sync requirement	165

LIST OF SYMBOLS AND ABBREVIATIONS

Symbol/Abbreviation	Description
AAC	Advanced Audio Coding
AAL	ATM Adaptation Layer
AEPS	Average Error Per Sample
ARMA	Auto Regressive Moving Average
ATM	Asynchronous Transfer Mode
ATSC	Advanced Televisions Systems Committee
CAT	Conditional Access Table
CBR	Constant Bit Rate
CRC	Cyclic Redundancy Check
CVBS/YC	Color, Video, Blank, Sync/Y-separate luminance (Y), Chrominance(C)
DCXO	Digitally Compensated Crystal Oscillator
DDR	Double Data Dynamic RAM
DHCP	Dynamic Host Configuration Protocol
DOCSIS	Data Over Cable Service Interface Specification
DSL	Digital Subscriber Line
DSM-CC	Digital Storage Media Command & control
DTS	Decode Time Stamp

LIST OF SYMBOLS AND ABBREVIATIONS

Symbol/Abbreviation	Description
DVB-CI	Digital Video Broadcasting-Common Interface
DVB-S	Digital Video Broadcasting - Satellite
DVB-T	Digital Video Broadcasting -Terrestrial
DVB-IP	Digital Video Broadcasting – Internet Protocol
DVI	Digital Video Interface
EIT	Event Information Table
EMM	Event Management Module
EMI	External Memory Interface
EPG	Electronic Program Guide
ESCR	Elementary Stream Clock reference
ES	Elementary Stream
FEC	Forward Error Correction
GCF	Gradual Correction Factor
HD	High Definition
HDD	Hard Disk Drive
HDMI	High Definition Memory Interface
HTTP	Hyper Text Transfer Protocol
IGMP	Internet Group Management Protocol
JTS	Jitter Time Stamp

LIST OF SYMBOLS AND ABBREVIATIONS

Symbol/Abbreviation	Description
LLR	Least Square Linear Regression
LMI	Local Memory Interface
MAFE	Modem Analog Front End Interface
MPLS	Multiprotocol Label Switching
MII	Media Independent Interface
NIT	Network Information Table
NTSC	National Advanced Televisions Systems Committee
PAL	Phase Alternate Line
PAT	Program Association Table
PCR	Program Clock Reference
PES	Packetised Elementary Stream
PID	Packet Identifier
PLL	Phase Locked Loop
PMT	Program Map Table
PSIP	Program Specific Information Protocol
PSTN	Public Switched Telephone Network
PTS	Presentation Time Stamp

LIST OF SYMBOLS AND ABBREVIATIONS

Symbol/Abbreviation	Description
RMII	Reduced Media Independent Interface
SATA	Serial Advanced Technology Attachment
SD	Standard Definition
SDRAM	Synchronous Dynamic RAM
SECAM	Sequential Color with Memory
SOC	System On Chip
S/PDIF	Sony-Philips Digital Interconnect Format
STB	Set Top Box
TCP	Transmission Control Protocol
TS	Transport Stream
VBR	Variable Bit rate
VCXO	Voltage Controlled Crystal Oscillator

LIST OF SYMBOLS AND ABBREVIATIONS

Symbol/Abbreviation	Description
a, b	Exponential Weighting Parameters
α , β	Damping Weights

α	Delay in current decode frame from previous FR frame
β	Audio delay w.r.t FR video
FS	Filter Strength
L	PCR_ diff.
M	Max. Window Size
N	No. of data points
P	PCR drift threshold
S	Min. sample number.
S_a	Scaling parameters
σ_y	Standard Deviation
λ	Maximum Slow Video Delay w.r.t. audio
γ	Maximum Audio Delay w.r.t. slow video
T_F	Time Interval for faster frame rate
T_S	Time Interval for slow frame rate
T_M	Maximum Specified AV sync. Delay
T_E	AV sync. Error window specification

Chapter 1: Introduction

1.1 BACKGROUND OF THE CLOCK RECOVERY PROBLEM

In the late 19th century the research to represent images with electrical signals began. In 1897 the cathode ray tube was invented, which still is the most widely used technique in TV sets and computer monitors to display the image. The possibility of transmitting audio-visual information became possible only with arrival of television in the early thirties.

The first television broadcast took place both in Berlin and Paris in 1935 and the first public television service was started in New York in 1939. In the forties, television services started in more and more countries in Europe, but each country developed its own standard. It was not until 1952 that a single standard was proposed and progressively adopted for use in Europe.

Apart from gradually improving quality of sender and receiver equipment, three major innovations have characterized for the development of television in the fifties: The introduction of color television began in the mid-fifties, high definition television in the late seventies, and digital television in the nineties.

One major problem with analogue television is its high demand of bandwidth. Thanks to the digital representation of images, advanced image coding, and data compression techniques, this helped to reduce the bandwidth significantly. Typically, about six digital TV channels can fit into the bandwidth of a single analogue TV channel. One major advantage of digital television over analogue, apart from the bandwidth reduction, is the possibility of interaction between the TV receiver and the viewer. Due to many more other advantages, demand for new audiovisual services and applications, such as Digital TV (Marsch, 1999), Digital Versatile Disc (Taylor, 1997) and videoconference continuously increased in the recent years. The introduction of these

services, and in particular those that have a distributed nature like videoconference and DTV, has been facilitated by three key factors: availability of coding/compression techniques of audiovisual data, high-capacity transport networks, and high-speed access technologies to network facilities.

Today digital television is delivered over dedicated broadcast networks (Fischer, 2004) such as satellite, cable, and terrestrial transmission. The most widely used video coding standard used in these networks is MPEG-2 (Motion Picture Expert Group-2) (MPEG2-System). It is part of the DVB (Digital Video Broadcasting) standard for broadcasting of digital television, which is most widely used standards in Europe. This is also used in storage of digital video for example on DVD. To enable some sort of interactivity, the networks have to provide support for an information flow from the receiver to the viewer. Therefore, there is large interest in providing new, interactive TV services over data communications networks, like IP networks. In order to provide interactive TV services over data communications networks, a lot of work has been done during the nineties. Especially, ATM (Asynchronous Transfer Mode) networks have been studied in this respect. An overview of the issues of asynchronous transfer of video over packet switched networks is given in (Karlsson, 1996). Since the Internet has grown and developed enormously in the last few years, it was expected that in a near future more services, like high quality digital television, will be offered beyond the usual data transmission that the Internet was first designed for. An Internet provider can provide both broadband connections to the Internet i.e. digital television and IP-telephony on the same cable. The transmission of digital television over IP-based network provides opportunities of interactive services for the viewers, e.g. VoD (video on demand) (Milenkovic, 1998) (where the viewer decides when to watch a certain movie or TV program).

There are some problems with real time transmission of audio-visual information over IP based networks because these types of networks were not designed for such applications. Traditionally IP-based networks behave as classical packet switched networks, providing no guarantee regarding delivery of the information on a "network level". When the network is heavily loaded, i.e. congested, some data may be lost or significantly delayed during the transmission. Audio-visual data are generally vulnerable to data loss because of the coding techniques used in real time, for example the most commonly used subsets of MPEG-2, generate bit streams with limited resilience to packet losses. Another major problem is that end-to-end delay is variable, which depends on the load of the network. In order to deliver MPEG-2 audio and video streams in real time with high quality, these delay variations have to be reduced at the receiving end, otherwise the decoder will not operate correctly.

Thus, most of the multimedia applications need time-sensitive information in voice, audio, and moving pictures. Mostly applications require strict synchronization properties. One can divide real-time streaming applications into different categories, depending on the service it provides and its tolerance to delay:

Information Retrieval Services: These types of services include video-on-demand, where the viewer decides when to watch a specific TV program or movie. Usually these services are not very delay sensitive. The viewer can accept to wait maybe a second from the moment that he/she presses "play" and the video sequence is displayed. These services are usually only suited for unicast.

Communicative Services: These types of service include videoconferencing and video telephony. Communicative services are sensitive to delay and response time. For videoconferencing, the end-to-end delay should not be more than 150 ms, see (Wolf 1997). Actually different authors suggest different delay limits. (The one suggested by

Wolf can be regarded as a quite stringent requirement.) These services can be either of type unicast or multicast.

Distributive Services: These types of services include broadcasting/multicasting of e.g. real time audio, video and text to Television. Distributive services are also delay sensitive. An example of this is a TV program where viewers can call in live to the program and take part in e.g. a quiz show or other competitions. Movie channels are less delay sensitive. However it should be noted that excessive buffering at the receiver might introduce a too long channel change time.

Regarding synchronization issues, an important distinction is that not all the implementations of MPEG-2 services require strict synchronization e.g. VoD (video on demand), RealPlayer (RealPlayer, 2009), MPEGplayer (MPEG TV, 2009) or ARTeMeD (Noro, 1997) and frame-based implementations do not require very strict synchronization.

However, in digital TV, the receiving system i.e. STB decoder must be able to reproduce the information with strict timing as when it was produced by the sender. All the concerns related to timing are encompassed by the term synchronization. Any synchronization error causes serious disruption of the quality perceived by the viewer, not only loss of information and excessive delay, but also loss of inter-stream synchronization between audio and video streams. Another issue in Digital TV (DTV) is due to the fact that set-top box decoder should provide accurate colour sub-carrier frequency synchronization as decoder provides a output of colour visual signal compatible with TV set (e.g., PAL and NTSC). A need is felt for a clock synchronization system that overcomes the clock synchronization problems in the presence of jitter in digital video broadcasting typically DVB-S, DVB-T and DVB-IP.

1.2 BACKGROUND OF THE A/V SYNC PROBLEM

With the introduction of advanced digital delivery systems for audio and video, there is an increased awareness of the timing relationship between audio and video. Owing to advanced data compression technologies such as Dolby Digital (AC-3) for audio and MPEG-2 for video, sound is clearer and pictures are sharper. Technologies such as Digital Television (DTV), DVD, Direct Broadcast Satellite (DBS), and Digital Cable use above compression techniques to deliver extremely high quality to consumers. However, performance can degrade significantly due to occurrence of audio/video synchronization problems.

Reports on (Linear acoustic, 2004) have indicated that most film editors are able to detect A/V Sync errors as short as $\pm \frac{1}{2}$ film frames. As film is projected at 24fps in the US and 25fps in Europe, this equates to approximately ± 20 msec. It is claimed that some editors can detect even smaller errors, but this might be more accurately attributed to their familiarity with the material being viewed. Dolby Laboratories has specified that any Dolby Digital decoder must be within the range of +5msec audio leading video to -15 msec audio lagging video. This is because human perception of A/V Sync is weighted more in one direction than the other.

The fact is that human brain is accustomed to hear things slightly after seeing them happen because light travels much faster than sound. We are all used to seeing this proven, although as it is such a common situation many times we do not notice. For example, a basketball hitting the court in a large sports venue would appear relatively correct to the first few rows, but the further back a viewer gets, the more the sound lags behind the sight of the ball hitting the floor. The further back you get, the more the sound lags, but it still seems OK.

Now, imagine if the A/V timing was reversed. You are watching a basketball game, and the sound of the ball hitting the court arrives before the ball looks like it makes contact. This would be a very unnatural sight and would seem incorrect even if you were in the first few rows where there was just a small amount of A/V Sync error (lip-sync error). The point is that the error is in the “wrong” direction. To summarize, human perception is much more forgiving for sound lagging behind sight as this is what we are used to seeing in everyday occurrences, so it’s better for the audio to be slightly late rather than early.

According to (ITU-R, 1998), the thresholds of timing delectability are about +45 ms to -125ms, and the thresholds of acceptability are about +90ms to -185ms. However, this range is probably far too wide for truly acceptable performance, and much tighter tolerances need to be obeyed. The ATSC recommends (ATSC, 2003) that the sound program never leads the video program by more than 15ms and does not lag the video program by more than 45ms. In today’s consumer electronics market, STB customers even ask to have a maximum delay of half a video frame (hence ± 20 ms or ± 17 ms). In addition to these tight requirements of end user, case of simultaneous synchronization of SD and HD video with audio becomes more critical. In this evolving situation, we have taken-up the problem of AV sync for our research work.

1.3 OBJECTIVES OF THE THESIS

In view of above, the main objectives of the research work are

- i. Design a clock recovery mechanism to adjust the locally generated clocks with the encoder clock for DVB-S (DVB- Satellite) broadcasting in digital satellite set-top box (STB).

- ii. Analysis of jitter in terrestrial environment and developing clock recovery algorithm for DVB-T (DVB-Terrestrial) in digital terrestrial set-top box.
- iii. Detailed analysis of IP environment and identify limitations imposed by IP broadcasting for suitable clock recovery algorithm in IP digital set-top box.
- iv. Detailed analysis of complete IP data flow. Design a new efficient clock recovery algorithm that can handle jitter up to several hundred mili seconds in IP environment.
- v. Design a new novel audio-video synchronization (AV Sync) module for digital set-top box based on drawbacks in existing audio-video synchronization implementation.

1.4 METHODOLOGY ADOPTED FOR THE RESEARCH WORK

The methodology adopted for the research work is shown in Figure 1.1.

Following tasks were completed to achieve the objectives of the study.

Phase 1: Extensive literature survey has been done. This involves in-depth study of existing research work done for achieving clock recovery mechanism and audio-video synchronization (lip-synchronization) in set-top box for satellite, terrestrial, and IP environment as shown in Figure 1.1.

Phase 2: Detailed study of MPEG-2 (Systems) standards, Set-top box subsystem, ST40 toolset.

Phase 3: A new algorithm named “Basic Averaging algorithm” has been proposed. It has been simulated and tested in satellite environment for set-top box application.

Phase 4: Another algorithm “Moving Window Weighing Filtering algorithm” has been proposed. It has been simulated and tested for satellite and terrestrial environment for set-top box application.

Phase 5: Study of IP set-top box functionality, detailed analysis of IP environment and issues in clock recovery for IP Set-top box has been done.

Phase 6: Proposing a new algorithm for IP environment. It has been simulated and tested for IP-set-top box. Comparative study of performance of newly developed algorithm with existing algorithm has also been done.

Phase7: Certain improvements have been proposed in existing audio-video synchronization design. Three new design schemes have also been proposed for dual video (HD & SD simultaneously) for AV Sync in digital set-top box.

Phase 9: Results have been compiled with the help of the output obtained throughout the research work in this thesis.

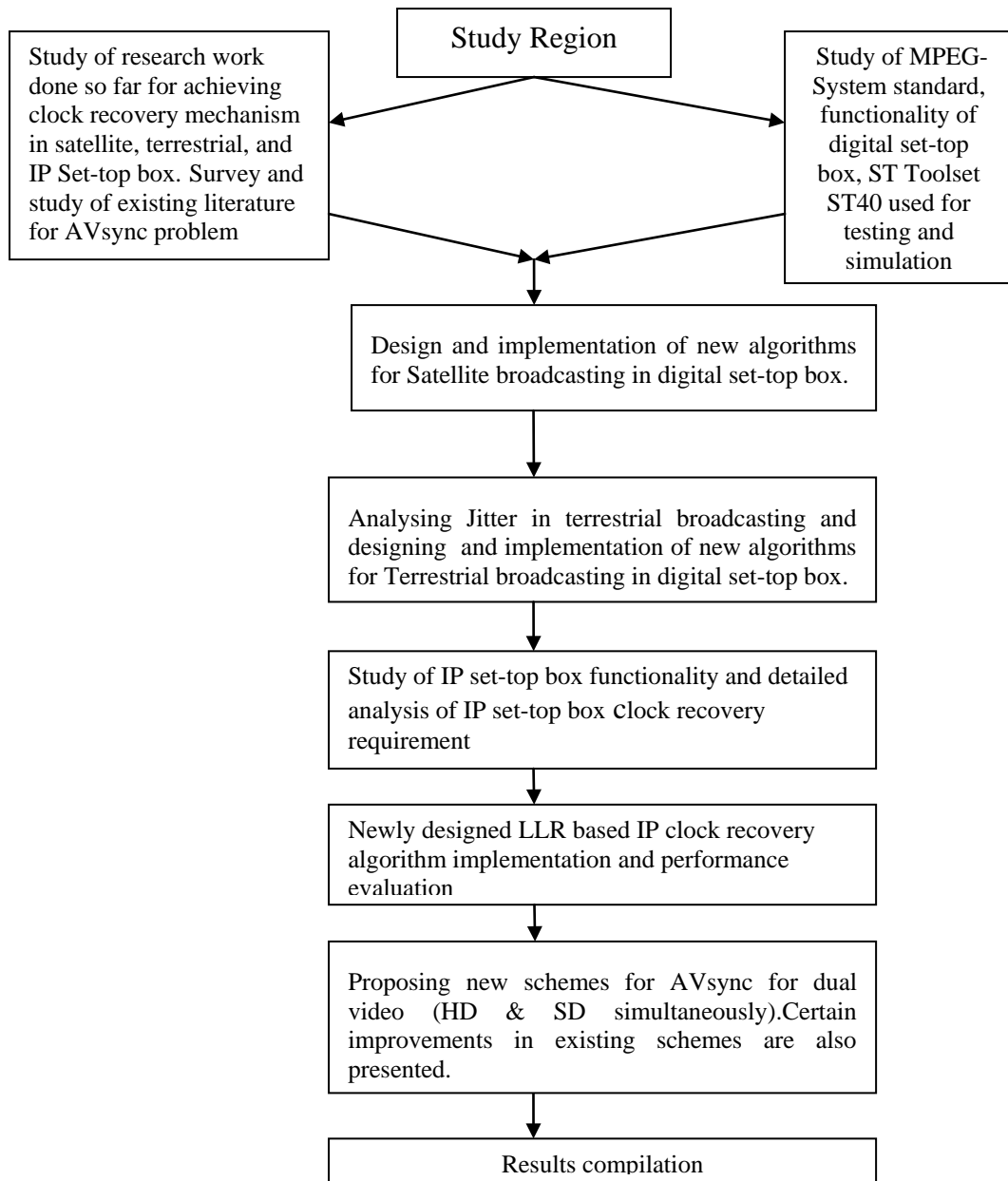


Figure 1.1 Methodology Adopted for the Present Work

1.5 SCOPE OF THE THESIS

The scope of thesis is mainly focused on achieving clock recovery for low jittery satellite broadcasting and moderate to high jittery Terrestrial broadcasting for digital set-top box. Major scope is to achieve clock recovery in STB for very high jittery IP broadcasting. A/V sync problem in digital set-top box for dual video (SD, HD) STB is another significant scope of our work.

We aimed for developing and implementing efficient clock recovery mechanism for satellite and terrestrial set-top boxes. In digital set-top boxes, decoder-clock has to be strictly synchronized with encoder by timing information (Program clock reference i.e. PCR timestamps embedded in transport stream). MPEG data is broadcasted over various transmissions medium. Satellite and Terrestrial are jittery medium where PCR packets get delayed and we get large +/- errors proportional to the delay and very frequent glitches. It leads to incorrect synchronization and color loss. The clock recovery problem in satellite and terrestrial environment has been analyzed. Existing algorithms applied in this area have certain limitations. Sometimes color loss has been observed in very high jittery environments. Our proposed approaches provide an efficient, economical, and a simple real time software solutions meeting all constraints posed by set-top boxes for satellite and terrestrial environment.

Further major work has been done for IP set-top box. The amount of jitter in IP environment is drastically high (may be shoot up to hundred of ms) compared to DVB-S or DVB-T. The detailed study has been done towards development of a Least-square Linear Regression (LLR) synchronization algorithm. The LLR algorithm is nearly two times faster than a PLL of equivalent synchronization accuracy. With continuous adaption enhancement in LLR, we minimize the end-to-end delay and hence the loss caused by synchronization errors. The optimal performance of LLR enables the deployment of methodology adopted in the present thesis in digital TV applications based on the MPEG-2 standard over a wider range of jitter inducing networks, where the conventional second-order PLLs generally fail. We have designed a new synchronization algorithm to perform efficient synchronization of two dispersed clocks for IPTV environment.

Another challenging scope of the thesis is in area of AV sync error. Audio-video Synchronization (AV sync) is very important to consumer and the display industry since

newer technologies have created a noticeable delay between the processing of video signals and the processing of audio signals. Lip sync correction algorithms take into account processing delays, so that both signals can be synchronized and presented to the viewer together. Correct A/V sync greatly improves the entertainment for the viewer. In a typical set-top box A/V sync (lip-sync) has always proved to be difficult. The main reason of AV Sync problem is difference in processing path of audio and video. Simultaneous synchronization of audio with HD (High Definition) and SD (Standard Definition) output impose even much tighter constraints. In this thesis detailed analysis of various Set-top box applications scenarios and their limitations has been explored. The current AV Sync implementations of Set-top box have been analyzed and certain enhancements in existing approaches have been suggested. We have also proposed some new schemes for AV sync for digital STB in this thesis.

1.6 ORGANIZATION OF THE THESIS

The research work is presented in ten chapters as follows:

Chapter – 1: In this chapter, rationale and structure of the thesis is presented. Need of development of clock recovery algorithms and audio-video synchronization in set-top box application is highlighted. This chapter also states the objectives of the research followed by the methodology adopted and organization of the thesis.

Chapter – 2: In this chapter, literature survey on various issues and techniques involved in achieving clock recovery for Satellite, Terrestrial and IP broadcasting has been presented. Difficulties and shortcomings of already existing approaches of clock recovery for set-top box application has been identified and understood. A review of literature on

existing audio-video synchronization (lip-sync) problem is presented and research gaps are highlighted.

Chapter – 3: In this chapter, popular MPEG standard is being explained. In-depth understanding of this standard is very much essential for software implementations of clock recovery algorithms. Major technical details of MPEG-System have been described in detail in this chapter.

Chapter – 4: In this chapter, state-of-the-art of set-top box subsystem has been discussed. As we may like to develop clock recovery algorithms for set-top box applications, so understanding the functionality of set-top box and identifying constraints posed by STB was immensely required. Clock recovery is being handled by decoder chip in STB system, SoC (system-on-chip) of STB decoder chip has also been explained in this chapter.

Chapter – 5: In this chapter, a new designed algorithm “Moving Window Basic Averaging” clock recovery algorithm for Satellite Set-top box has been discussed. The methodology adopted for clock synchronization in set-top box is described. The algorithm has been simulated and tested in real time environment.

Chapter – 6: In this chapter, nature of jitter in terrestrial broadcasting has been analyzed. Another suitable algorithm “Moving Window Weighing Filter” for terrestrial Set-top box has been proposed. The results of newly developed algorithms in terrestrial environment are being presented.

Chapter – 7: An overview of various protocols and how MPEG-2 is to be transmitted over IP-based networks is presented.

Chapter – 8: This chapter will firstly describe the problems that occur when real time audio-visual information is streamed over IP networks. After that newly designed “Continuous Adaption Enhancement in LLR algorithm” for IP-STB has been discussed. The performance of proposed algorithm is simulated and tested in real environment. Superiority over other existing algorithms has been illustrated.

Chapter – 9: In this chapter the problem of AV synchronization (lip-synchronization) has been analyzed. Certain enhancements in existing approaches are suggested. Three different proposals for dual video (HD & SD simultaneously) with audio synchronization has been presented. The results are used to select optimal scenario for implementation.

Chapter–10: This chapter presents the summary of results, conclusions, and recommendations of the future research work. Further scope of work and specific contributions in the thesis are also presented.

Chapter 2: Literature Review

In this chapter, survey of literature regarding clock synchronization for digital set-top box is presented. A review of literature on major issues involved in various issues and techniques involved in achieving clock recovery for Satellite broadcasting, Terrestrial broadcasting and IP broadcasting is discussed. Mainly following major issues are presented in this chapter:

- Review of Clock Recovery algorithm for Satellite broadcasting in set-top box.
- Review of Clock Recovery algorithms for Terrestrial broadcasting in set-top box.
- Review of Clock Recovery algorithms for IP set-top box.
- Review of existing Audio-Video synchronization techniques for set-top box

2.1 LITERATURE SURVEY OF CLOCK RECOVERY FOR SATELLITE AND TERRESTRIAL BROADCASTING

The growing demand for digital satellite & terrestrial broadcasting services has identified the need of suitable clock recovery algorithm for Satellite and Terrestrial environment. Clock synchronization plays a key role in correct working of the STB system for real time audio and video transmission. MPEG encoder is a variable bit rate device. To transport a variable bit rate stream through a constant rate channel or medium, buffers are required both at the encoder and the decoder. Since buffers are placed at the encoder and the decoder sides, these buffers have to be managed so that they neither overflow nor underflow. Buffer management in turn requires clock synchronization between the encoder system clock and the decoder system clock.

In DVB-Satellite (DVB-S) and DVB-Terrestrial (DVB-T) communication, uncertain arrival of PCR packets at the receiver end causes serious problems in synchronization of receiver system clock with that of the transmitter. The conventional methods available to overcome this aspect cannot reduce all the problems, specifically in worst case channel conditions of terrestrial set-top box. Problems such as color loss, jittery video, and audio dropouts still persist. Apart from this, existing methods have high computational overhead and high cost.

Audio and video have separate frame size definitions and sampling rates and have entirely separate encoding strategies. To synchronize audio and video streams, it is convenient to have a common clock between the encoder system and the decoder system. This common clock between the encoder and the decoder systems can easily provide the time reference needed for inter-stream synchronization. This common clock is achieved by the process of clock recovery. Clock recovery between transmitter and receiver in existing set-top boxes is achieved by configuring clock recovery hardware registers i.e. software approach in STB decoder chip for adjustment of decoder clock.

In a purely hardware based decoder system, building a synchronization subsystem is not difficult. Large body of literatures (Edward and David, 1988; Harry and Van, 1971; Holborow, 1994) are available on the subject of frequency and phase locked loops.

Various hardware implementation modules for clock adjustment is achieved by either DCXO chip (Liming, 2008), or by VCXO chip (Watanabe *et al.*, 2006), or by on-chip VCXO/DCXO (Voltage controlled Crystal Oscillator/Digitally-Compensated Crystal Oscillator) module (Lin J., 2005; Mujica *et al.*, 2003; Lee and Bulzacchelli, 1992).

A real time software solution for resynchronizing filtered MPEG2 transport stream has been proposed by Bin and Klara (2002). Author discussed the scenario of streaming of HD video in MPEG-2 Transport Layer syntax with software streaming/processing and hardware decoding. Several approaches to solve synchronization problem has been suggested. In one approach, padding with NULL packets is done in MPEG stream. In other approach some of the filtering intelligence is exported to the end hosts, for example, instead of inserting NULL packets, inserting only a special packet saying that the next N packets should be NULL packets. Note that this padding is important to maintain correct timing, especially if the client is using some standard hardware decoding board. This way, the bandwidth is indeed saved, but this introduces non-standard protocol outside MPEG2. Solution proposed (Bin and Klara, 2002) has some drawbacks. It can only handle bit rate adaptation operation. We only try to fix the header of each frame to its original position on the line, which means the changed frame should not occupy more bits larger than distance between the current frame header and next header. This property does not always hold, since some filtering operations like information embedding and watermarking may increase the frame size in bits. The saved bits are padded with NULL packets to maintain the original constant bit rate and the starting point of each frame and this ironically runs to counter our initial goal of bit rate reduction for some operations like low pass filtering and color frame dropping. The resulting stream contains the same number of packets as the original one. The only difference is that the number of bits representing each frame has been shrunk; yet this saving is spent immediately by padding null packets at the end of each frame. Filtering intelligence approach will introduce problem associated with non-standardized solutions such as difficulties in software maintenance and upgrading, so it can only be considered as secondary choice not as a major solution.

Kaiser (1993) has given a technique to minimize the effects of jitter in the clock recovery process is by counting the time difference between successive timestamps in the packet stream. Although the jitter introduced by the network may be computed on per packet-basis in this scheme, it requires constant spacing between timestamps in the packet stream, an assumption that may not hold in MPEG-2 Transport Streams.

Antonio *et al.*(2001) has proposed efficient non-data-aided carrier and clock recovery for satellite DVB at very low signal-to—noise ratios. Analysis of clock and carrier (frequency/phase) synchronization algorithm intended for use digital video broadcasting systems is presented. However clock recovery needed for audio and video streams in MPEG decoder in set-top box has not been explored.

In this research work, lot more literature is reviewed but we found that there is not much research work done regarding the synchronization problems of MPEG over satellite and terrestrial set-top box. A large body of literature (Monika *et.al.*, June 2009; Shen *et.al.*,2004; Osaki, 2002; Weillan and Akyildiz, 2001; Tryfonas and Verma,1999; Ramamoorthy,1997; Rangan,1996; and Andreotti *et.al.*,1995) suggested clock recovery for ATM and packet switched networks but the drawback in these systems is that the computation involved for clock recovery requires a complex algorithm and high cost apparatus. The complexity of algorithm is an issue in set-top box application. This issue is due to the fact that real time audio and video processing consumes most of the processing space of set-top box decoder chip, so algorithm for clock recovery must be simple so that CPU of decoder chip is not loaded. Another drawback of high cost is well known fact in consumer market of set-top box.

It has been realized that jitter in Satellite and Terrestrial environments is much lesser as compared to IP environment hence a light weighted, cost effective algorithm can be deployed for handling jitter in such a transmission. To date, to our knowledge, no existing clock recovery scheme is meeting constraints of STB for satellite and terrestrial transmission.

2.2 LITERATURE SURVEY OF CLOCK RECOVERY FOR IP STB BROADCASTING

Several researchers (Markopoulo *et al.*,2003; Gardner *et al.*,2003; Liang *et al.*, 2001; Sanneck *et al.*,2001; Cole and Rosenbluth,2001; Vleeschauwer *et al.*,2000; Kostas *et al.*,1998; Moon *et al.*,1998 ; Bolot *et al.*,1995) have worked on VoIP but it's a real challenge to achieve clock synchronization while real-time audio and real-time video both are broadcasted on IP networks and this real time data has to be displayed on digital television.

Due to stochastic nature of IP network, the clock recovery for MPEG stream in IP-STB is a big issue. IP networks were originally designed to optimize bandwidth utilization by resource sharing, but they are unable to optimally meet the needs of real time traffic that requires timely clock synchronization. This limitation is due to the fact that stochastic packet transmission delay jitter may lead to significant variation in packet arrival time. According to DVB-IP (2005), section 7.2.1.1, the amount of jitter on an IP network is typically 20 milliseconds (ms). However, maximum jitter may be more than this specification to the tune of 100 ms, eventually this makes clock synchronization more difficult. In IP environment, there is no limit on how late a packet may arrive. Theoretically this delay could be infinitely long so we have to discard packets reaching

after a certain time otherwise all the subsequent packets get delayed. This maximum delay should include 99% of the packets while 1% of the packets may be discarded. Color loss is the other problem that has serious impact on the picture quality of IP Digital Set-Top Box. As per (MPEG-2 System) to prevent color loss, local frequency overshoot should be less than 1350 Hz. The TV monitor connected to set-top box has to locate a color burst signal in order to decode the color signal. According to Robin and Poulin (2000), the frequency of the color burst must be accurate to 50 ppm. If the decoder clock has a large error, then the color burst signal will not be found and this will result in color loss. When the color burst is successfully found, a PLL (Phase Locked Loop) in the monitor is used to track subsequent changes in the color burst frequency. After sampling the small color burst signal at the start of each line, the PLL then generates the color reference for the duration of each line. If the STB frequency changes too rapidly, then a 'color shift' is observed as the PLL momentarily loses track. Although this is a known problem, at present no specification for the maximum acceptable rate of frequency change was found in any standard. This maximum drift rate was approximated under the assumption that typical time constant for a PLL in TV is 15 ms and set top box drift rate and tolerance should be 1000 times faster than this time constant. Thus drift rate come out to be 67 KHz per second. Poor accuracy of clock synchronization results in audio distortion or video flickering. Long response time results in the buffer overloading, causing loss and excessive delays. The MPEG-2 standard does not formally constraint the network jitter: instead, it requires that all TS decoders must be capable to absorb a network jitter of 4 ms, unfortunately, this threshold is quite often exceeded in actual packet-switching networks, causing several malfunctions of the MPEG-2 systems decoder (Gringeri *et al.*, 1998). Thus there is a strong need to find a robust technique of clock recovery in emerging IPTV environment that can handle all the above limitations.

The standard technique of synchronization between the transmitted and received clock, as discussed in the MPEG2 standard uses a discrete time Phase locked Loop (PLL). A typical phase-locked loop (PLL), used in the MPEG decoder to synchronize the clock of the decoder to the STC of the encoder (Best, 1993), is shown in Figure 2.1. It works as follows: Initially, the PLL waits for the first PCR to arrive. When the first PCR arrives it is loaded to the PCR counter. Now the PLL starts to operate in a close loop fashion. Each time as a PCR arrives it is compared to the current value in the PCR counter. The difference gives an error term e . This error term is sent to a low pass filter (LPF). The output of this stage is a control signal " f " which controls the frequency of the voltage-controlled oscillator (VCXO) whose output provides the system clock frequency of the decoder. The output of the VCXO is sent to the PCR counter. The nominal frequency of the VCXO is approximately 27 MHz. After a while the error term e converges to zero which means that the PLL has been locked to the incoming time base.

The requirements on stability and frequency accuracy of the recovered STC clock depend on the application. In our DTV applications, the output from the decoder will be fed to an analogue TV set. The colour sub-carrier and all synchronization pulses will be derived from this STC clock. In our case the STC must have sufficient accuracy and stability so that a TV set can synchronize correctly to the video signal.

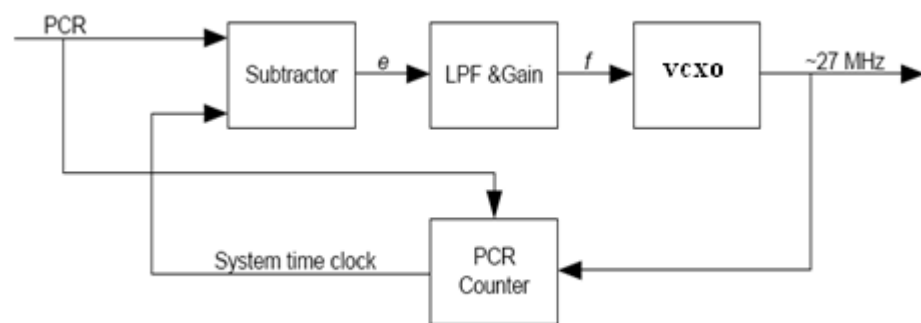


Figure 2.1 Block diagram of a PLL used in MPEG decoder

The clock synchronization of the receiver and the transmitter clock with the PLL requires a loop filter for jitter absorption. The loop filter is usually a Proportional-Integrative (PI) (Gardner, 1979) filter of extremely narrow bandwidth (typically, less than 0.1Hz), which eliminates most of the high-frequency components of the jitter. Unfortunately, the response time of the filter is dependent on the bandwidth of the filter, a narrow filter bandwidth has the inconvenience of a very slow convergence of the local signal to the reference signal (typically, in the order of minutes). As the jitter to be removed increases, the only possible strategy with PLLs is to further reduce the bandwidth: in this way, the response time becomes even longer (Noro, 2000).

In (Andreotti *et.al.*,1995) ordinary PLLs are used to dejitter MPEG-2 stream, which uses the PCR timestamps in the MPEG-2 Systems layer. In this simulations it is assumed that the MPEG-2 stream is delivered over a network with small delay variations, and therefore uses a peak-to-peak jitter (delay variation) amplitude of up to 1 ms. Thus author simulates his algorithm using jitter models (like Gaussian distributions), however amplitudes of delay variations in these regions are regarded as very low in IP-based networks. This scheme is therefore not suited for IP networks.

Monika *et.al.*,(Dec 2008) has proposed an approach based on PLL design. Moving Window Averaging Filter for Clock Recovery in Set-Top Box has been presented. Approach is low cost and simple but it is performing well only for satellite broadcasting where jitter is very less.

Monika *et.al.*, Jan 2009; Monika *et.al.*, Oct 2009; has presented another approach based on PLL design. Rather simple averaging, some more enhancements like weighing is also applying for handling more jitter. Author has proposed an enhanced FIR

Filter based Module for Clock Synchronization in MPEG2 Transport Stream is efficient and low cost. Its performance is also evaluated (Monika *et.al.*, March 2009) and to be found stable. Algorithm is able to handle jitters of Terrestrial broadcasting but it fails in highly jittery environment of IP.

In summary, conventional PLL synchronization algorithms fail in recovering an accurate clock signal in the presence of high amounts of network jitter, that can be to the order of dozens of milliseconds on Internet links. We can conclude that a PLL like design for highly jittery environment such as IP STB applications has to face several issues:

- The large jitter experienced on IP networks complicates the filtering, the relative clock drift is a slowly moving value hidden behind a large white noise.
- Efficient filters will have a slower response time, letting the relative clock frequencies drift further apart.
- Digital filters are based on the assumption that the input variable has a fixed sampling rate however for MPEG transport streams there is no fixed frequency requirements for PCR time stamps, other than sending PCR at least every 100 ms.

Thus PLL designs are not suitable for IP networks due to above mentioned reasons. A number of possible alternatives to PLLs are proposed in the literature (Moon *et.al.*, 1999; Roppel, 1995; Cristian, 1989) principally derived from the statistical signal processing field.

One such technique (Lau and Fleischer, 1992) proposes a new implementation that suggests that packet switched networks include the synchronization of the transmitter clock and the receiver clock using timestamps. The transmitter sends a series of explicit

time references as time stamps in sequence of packets and the time stamps are used by receiver to synchronize its clock with the transmitter, since no common network clock is used, the receiver rely on locking its clock to the arrival of time stamps. This technique is analogues to the common method of periodic insertion of synchronizing pattern into bit stream at the transmitter whereby receiver is adapted to detect these synchronizing patterns and use them to generate a reference clock signal for PLL at the receiver. Techniques have been developed for clock synchronization using a linear modeling of error between transmitter clock and receiver clock. Using a linear regression analysis, the frequency offset between transmitter clock and receiver clock for a given time period or time instance is estimated or predicted and receiver clock then is adjusted by this estimated error. However, major disadvantage in use of linear regression analysis as an estimation technique is that the large number of consecutive clock samples generates accurate timing signals and is needed to accurately estimate the model coefficients. However, the storage capacity for storing a large number of time series samples and the associated calculations would be prohibitive.

Another such technique is based on the Least Square linear Regression (LLR) (Noro, 2000) principle. This principle will take N samples of (STC,PCR) data points as shown in Figure 2.2 and generate the best fitting line such as the sum of the squares of the distance between each (STC,PCR) data point and the line is minimal (hence, “least-square” name). At the time of arrival of the PCR packet, N consecutive samples of the transmitted clock (PCR) and the local clock (STC) values are being collected. These (STC,PCR) data points are represented on a bi-dimensional space and fitted with a straight line. The **slope** of the straight line represents the **correction factor** to be used for

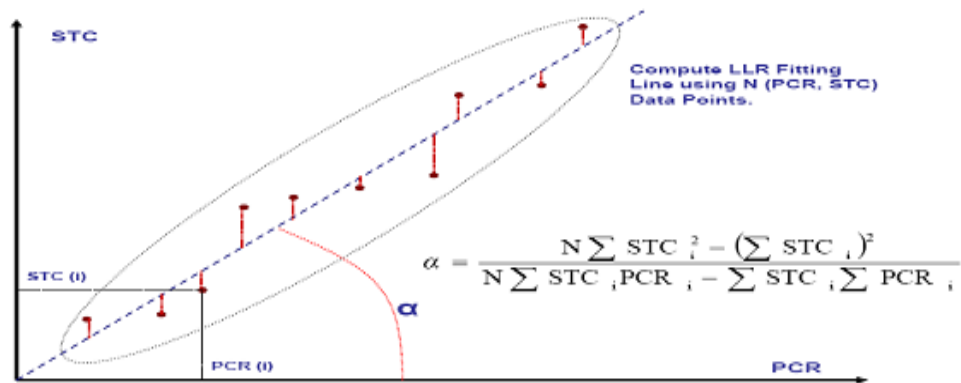


Figure 2.2 LLR based Clock Recovery Method

synchronizing the frequency of the receiver clock with the frequency of the transmitter clock.

The LLR offers three significant advantages over other algorithms:

- It shows superior performance of jitter removal.
- It is not affected by the variable PCR sampling interval and more robust than PLL based designs in the presence of jitter.
- It provides a better response time than PLL based designs.

In spite of above advantages, the main issue of LLR with respect to PLL based designs is the relatively heavy computational load on the CPU (Central Processing Unit). LLR uses several multiplications and divisions, often using floating point arithmetic, due to the large amplitude values so it is better suited to systems with high performance CPU. We can summarize that although this technique and others based on least squares linear regression analysis generally perform considerably more efficient than conventional

second-order PLLs, these techniques have drawbacks of requiring large number of samples (thus a large storage requirement) in order to generate accurate timing signals.

Several other techniques such as dejittering buffer and time stamp approach have been proposed in literature for systems with larger jitter. In first technique a **de-jittering buffer** is used at the receiver to absorb the network introduced jitter. A disadvantage of this approach is that it requires a prior knowledge of the maximum jitter in the network to avoid an overflow or the underflow of the de-jittering buffer. Also this approach wastes memory by using two separate buffers– one system decoder buffer and the other the de-jittering buffer. Major issue of suitable clock recovery in set-top box is due to the fact that in real time signal processing audio/video drivers consume most of processing space. If clock recovery involves more computation time, audio/video decoding suffers. Hence, latency i.e. time taken by clock recovery algorithm to apply correction in the decoder clock, acts as a bottleneck for the CPU to provide adequate time for audio-video decoder.

Rangan *et.al.*(1996) proposed the idea of providing a constant de-jittering space in MPEG2 system decoder by combining the two buffers. These techniques use de-jittering buffer to achieve clock recovery. This introduces more latency to this process.

Singh *et.al.*(1994) proposed a scheme based on first one, often denoted adaptive buffer, monitors the fullness of the input buffer and determines the playout rate according to some low pass algorithm. If the buffer fills faster, local clock speed is increased and vice versa. The logic is beneficial in preventing any A/V glitches. However, if the local clock frequency is changed too frequently or above the maximum frequency, due to overshoot limited by the TV monitor, color loss is obvious.

Parekh (1997) monitor buffer level to maintain the A/V buffers towards 50% occupancy effectively. But the jitter attenuation in this algorithm is not very high. The residual jitter amplitude violates the (MPEG-2 RTI) specifications. Moreover, this scheme is also possible to be used only in CBR video traffic i.e. if the stream runs at a constant bit rate. If the stream run at variable bit rate the buffer occupancy method would fail drastically.

Shen *et.al.*(2004) proposed a new buffer measurement based packet network clock recovery algorithm for fast and accurate synchronization. This algorithm has the ability to filter out buffer level fluctuation efficiently and removes the negative contribution of delay jitter in clock recovery. Simulations are performed for jitter distribution based on a geometric distribution model developed by Bolot (Bolot, 1993). However this jitter model practically does not exist in today's real time IP network. More real time jitter distribution models like Pareto (Weisstein-Pareto, 1999), Weibull (Weisstein-Weibull, 1999) needs to be used for testing algorithm's capability to handle real time jittery environment in IP transmission.

Based on second approach, Weillan and Akyildiz (2001), introduces a new source rate recovery scheme, called the jitter time-stamp (JTS) approach. Each packet from the source carries this time stamp information. Here in PCR-unaware approach, the sender does not check whether PCR values are contained within a transport packet and may therefore introduce significant jitter to PCR values during the encapsulation, which in turn may affect the perceived quality of the video signal. The scheme is providing synchronization at asynchronous transfer mode (ATM) adaption layer (AAL). However it is not able to meet constraints of STB as possible application.

Tryfonas and Verma, 1999 have suggested a design methodology in which a constant amount of de-jittering space is provided in the system decoder buffer by subtracting an offset from the incoming PCR values. The jitter estimator performs restamping on all the incoming packets containing clock values is used in conjunction with standard PLL. The approach minimizes the effect of the jitter on clock recovery by using a jitter estimator to calculate the jitter on a per packet basis and re-stamp the incoming packets based on the estimated jitter. This scheme can be used to correct both the source induced and the network induced jitter. The estimation of jitter is the real challenge in this technique. Further, jitter introduced due to the set-top box front-end is not taken into account by this algorithm. Probably approach used was designed to work in ATM networks and was therefore not designed to handle very high jitter amplitudes in IP-STB.

Ramamoorthy, 1997 uses timestamp information for correction to the decoder clock. This method is feasible for both VBR (Variable bit rate) and CBR (Constant bit rate) streams. However, there is a possibility of A/V glitches if efficient jitter reduction is not achieved due to buffer underflow or overflow. The simulation technique discussed here provides clock synchronization using sub-sampling. The algorithm was not tested by the author under actual set-top box environment constraints. Author has identified the set-top box as the possible application but not practically tested in this environment. The system constraints for the RTOS (Real-time OS) include CPU time usage and memory usage. Comparing with above algorithms, it was found that complexity caused a bottleneck for implementation in RTOS environment under the constraints discussed above. The simulation technique discussed by (Ramamoorthy, 1997) lacks sufficient testing in real-time DVB-IPTV environment.

For handling both CBR and VBR streams, Varma(1996) discusses that if maximum jitter is known for the VBR stream, then analysis like CBR stream may be carried out. But the stochastic nature of jitter in IPTV suggests fallacy in this argument.

Bjorn (2000) deals efficiently with the problem of handling delay variations of MPEG-2 audio-visual streams delivered over IP-based networks. Here, the focus is also on high quality digital television applications. A scheme to handle delay variations (jitter) has been designed and evaluated by simulations. The results have been compared to the expected requirements of an MPEG-2 decoder and an ordinary consumer TV set. A simple channel model has been used to simulate the IP-based network, where the jitter process is uniformly distributed with a peak-to-peak delay variation of 100 ms. From simulations it has been shown that it is possible to design a dejittering scheme capable of filtering 100 ms of peak-to-peak IP-packet delay variation, producing a residual jitter amplitude in the order of a microsecond. Such a low jitter amplitude is obviously well below the MPEG-2 RTI specification of $\pm 25 \mu\text{s}$. The scheme also matches the performance requirements that can be expected of a consumer TV set. It has also been shown that it is possible to combine an extreme low-pass filtering with a sufficiently small additional delay added by the dejittering scheme. But proposed dejittering scheme has only been tested in simulations and some aspects have therefore been neglected, which could cause problems in a real implementation. If the scheme is to be implemented in a real system some further investigations need to be made, especially concerning issues around real time support of common operating systems.

Mathur and Saha (2007) proposed scalable integer based error estimation technique for clock recovery in packet switching networks. The proposed scheme requires only integer level precision as compared to conventional floating point precision. The

system shows stable clock recovery, however, the response time for this algorithm was too high for high jitter streams.

The Broadcom patent (Fisher *et. al.*,2007) discusses the hardware module 7140 used to achieve clock recovery. No algorithm which uses PCR timestamps to modify the system clock is explained or claimed.

Aweya *et.al.*,(2007) uses a differential clock recovery algorithm for packet switched network in general. It suggests good recovery in IP, MPLS and Ethernet. We use RTP protocol for STB. For our application of IP-STB, the LLR algorithm is different and much robust from above techniques.

Perkins and Lookabaugh (1998) simulations for reduction of timing jitter in audio-video transport streams indicate that for a total network jitter of 2 msec peak-to-peak, the output steady-state peak-to-peak jitter can be reduced below a threshold amount. However in IP-STB jitter is much high.

Baker (2003) invention provides a method of measuring MPEG PCR jitter, frequency offset and drift rate with a selectable, constant measurement bandwidth over non-uniform PCR arrival times and a variable PCR rate. Siu *et.al.*,(1999) also claim a apparatus for measuring program clock reference (PCR) jitter in a data stream, such as an MPEG-2 transport stream. The PCR jitter value may be displayed on a display unit, preferably in a histogram. Jitter is measured effectively, however, in both inventions, handling of jitter in IP network is not claimed.

Osaki (2002) invention claims an apparatus for reducing a jitter of a program clock reference of an MPEG signal transmitted over ATM (Asynchronous Transfer Mode) system. In our STB application, due to major constraint of CPU bandwidth in Real

time operating system, a separate apparatus for reducing jitter can not be an appropriate choice.

2.3 LITERATURE SURVEY OF AUDIO-VIDEO SYNCHRONIZATION

A/V sync issues within the TV plant are not new to digital television, they are perhaps more noticeable. Different amounts of delay in the signal processing in both the audio and video channels might occur independently from each other, which require the signals from the two channels to be re-aligned. Lip-sync error occurs when the sound is heard earlier than the movement of lips is seen. Lip-sync error has been observed and discussed intensively in the literature (Cooper, 2008) and is the most common type of A/V sync error. Very tight AV sync requirements have been demanded by various key customers in STB applications. Therefore the present section focuses on survey of literature for A/V sync.

Younkin and Corriveau,(2008) tries to determine the absolute detection threshold of lip sync error. This specific work is aimed at lip-sync detection in context of a single speaker. Much tighter constraints posed in application of handling dual video (HD & SD) synchronization with audio, are not handled here.

Chanwoo *et. al.*,2006; Kim *et. al.*,2005; Kim and Seo,2006; devised algorithm quite suitable for consumer cellular phone with limited computational resources. Proposed algorithm is suitable for Video-Telephony applications but this can not handle constraints posed by digital TV.

Bhattacharya (2006), explores four system-level design options for PTS-based AV-synchronization mechanisms where the AV playback chain spans more than one

processor. The first of these extends a single clock domain over multiple processors through hardware support. The second realizes PTS translation across clock domains using explicit clock modeling. The third bundles AV effects processing into an extended logical renderer with fixed delay and forces logical presentation at the decoder output. The fourth scheme constrains the streaming playback chain onto a single processor to eliminate cross-processor synchronization issues. However, out of above four, no particular design paradigm fits into overall system design of SoC of set-top box decoder chip.

Chen *et.al.*(1995) has proposed a novel audio/video synchronization model. Based on it, a powerful multimedia authoring system was implemented. This system provides a friendly and functionally complete environment for users to conduct their A/V editing operations. In our application, rather editing, dual video sync with audio is much more important.

Yuong and Ouhyoung(1993) have proposed an A/V Synchronization scheme based on the idea of aligning the physical location of A/V data storage. However, the above method suffers from inaccurate estimation of media loading time in Real Time Operating System (RTOS) environment in STB.

2.4 RESEARCH GAPS IDENTIFIED IN EXISTING CLOCK RECOVERY ALGORITHMS

In providing the smooth audio-video viewing using Satellite, Terrestrial and IP Set-Top Box, the efficient handling of jitter in above environments is very much essential and critical. Detailed analysis of the existing algorithms implementations for clock

recovery and Audio-video synchronization in various environments has been carried out.

In view of the literature survey above, following research gaps are identified:

- A light weight, low cost, fast algorithm is needed for low jittery Satellite and medium jittery Terrestrial environment which can handle all constraints of real time STB without consuming much CPU bandwidth.
- The PLL based designs are not suitable for IP clock recovery as large jitter experienced in IP networks. Efficient filters will have a slower response time, results in the relative clock frequencies drift further apart. Digital filters are based on the assumption that the input variable has a fixed sampling rate, which is not true for MPEG TS streams.
- Although LLR based clock synchronization method has several advantages and provides better response time compared to PLL based design but due to relatively heavy computational load placed on the CPU (Central Processing Unit), it is better suited to systems with high performance CPU.
- De-jittering buffer technique suggested by various researchers introduces more latency to the clock recovery process. This is not acceptable for Real time STB where time needed by audio-video decoder for smooth operation is very much critical.

In view of the above, it would be desirable to provide a technique for synchronizing a receiver clock with a transmitter clock that overcomes the above described inadequacies and shortcomings. The present research work aims at applying PLL based design for Satellite and Terrestrial environment where jitter is less. For IP

applications certain practical problems in existing implementation of the LLR algorithm have been identified. In most of existing approaches additional de-jittering buffer is needed to store the sample set of incoming PCR, STC values which is being eliminated in this thesis. The major contribution in this thesis is to apply enhancements in old LLR algorithm implementation so that LLR performance in IP application can be improved significantly.

2.5 RESEARCH GAPS IDENTIFIED IN EXISTING AV SYNC IMPLEMENTATIONS

Advance data compression technologies such as Dolby Digital (AC3) for audio and MPEG-2 for video provide viewers high quality multimedia experience with the help of STB. Lip-sync problem can be observed if AV synchronization specifications are not properly met. Simultaneous synchronization of audio with HD (High Definition) and SD (Standard Definition) output impose very tight constraints. In view of the literature survey above, following main research gaps are identified.

- Various models for correct AV Sync implementation has been suggested by researchers in the similar field. However, none of them addresses the AV Sync problem of dual video (HD & SD simultaneously) with audio.
- Research work done so far have not considered set-top box as possible application for their implementation so such models are not suitable for Real Time set top box application.

In this chapter, the various techniques and methodologies for clock recovery and AV synchronization have been explored rigorously. Most of the existing algorithms for

clock synchronization in Satellite, Terrestrial and IP network have been discussed. Various AV sync approaches suggested by various researchers in the similar fields have been discussed. This chapter has provided a solid background and good summary of work done in this field.

Chapter 3: MPEG Standard

The list of systems use MPEG standard is extensive and continuously growing. Some of them are digital TV (cable, satellite and terrestrial broadcast), Video on Demand (VOD), Digital Versatile Disc (DVD), personal computing, MPEG test and measurement, interactive TV, etc. In this chapter popular MPEG standard has been presented.

3.1 INTRODUCTION

MPEG (Motion Pictures Expert Group) is an encoding and compression system which defines a series of standards for compression of moving picture information. The MPEG was established in January 1988 with the mandate to develop standards for coded representation of moving pictures, audio, and their combination. Figure 3.1, shows how MPEG systems have evolved over time. This diagram shows that MPEG-1 standard developed in 1991 offered VHS quality at 1.2 Mbps, primarily used to record audio – video in CD ROMs. This standard was updated in 1995 and became MPEG-2 which was used for satellite, terrestrial, and cable digital television along with DVD distribution. The MPEG specification then evolved into MPEG-4 in 1999 to permit multimedia distribution

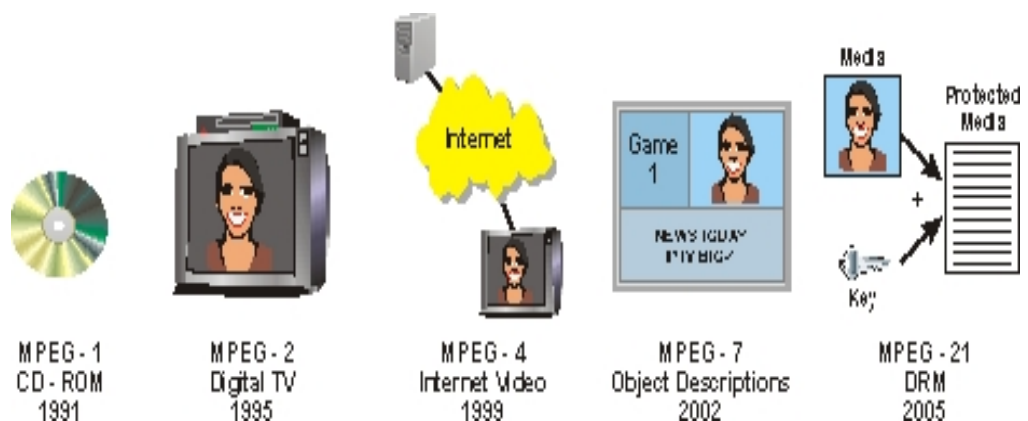


Figure 3.1 Evolution of MPEG

through the Internet. The work continues with MPEG-7 for object based multimedia and MPEG-21 for digital rights management.

Different types of MPEG Standards are mentioned below:

- MPEG-1, A standard for storage and retrieval of moving pictures and audio on storage media e.g. CDROM. It is for medium bandwidth (upto 1.5 Mbits/sec)
- MPEG-2, A standard for digital television. It is for higher bandwidth. It can deal with wider range of frame sizes (including HDTV)
- MPEG-3, A standard was developed for HDTV application with dimensions up to 1920 x 1080 x 30 Hz, however MPEG-2 and MPEG-2 syntax worked very well for HDTV rate video so MPEG-3 was discarded.
- MPEG-4, a standard for multimedia applications. It is for very low bandwidth (64Kbits/sec). It is optimized for videophones.
- MPEG-7, a content representation standard for information search.
- MPEG-21, a standard for Digital Rights Management.

MPEG-1, MPEG-2 and MPEG-4 have been standardized, whereas MPEG-7 and MPEG-21 are currently being developed. The MPEG-2 standard has been extended by various groups including:

- MPEG-21, a standard for Digital Rights Management.
- Digital Video Broadcasting (European)
- The UK Digital TV Group
- U.S. Advanced Televisions Systems Committee (ATSC)
- Digital Audio Visual Council (DAVIC)

- Digital Versatile Disk (DVD)

The synchronization aspects that are found at the application level are mostly related to audio-visual services. In this area, the applications based on the Moving Picture Expert Group standards MPEG-2 play a major role (MPEG2,13818-1; MPEG2,13818-2; MPEG2,13818-3). The family of MPEG-2 standards defines the rules for coding, storing, multiplexing and transmitting digital audio and video signals. In the case of streamed audio and video, MPEG-2 is intentionally made independent from the network technology that is employed to provide the communication facility. The synchronization schemes defined for other audiovisual applications, such as MPEG-1, MPEG-4, H.261- and H.263-based videoconference systems (MPEG1 11172-1; MPEG1 11172-2; MPEG1 11172-3; H.261,1993; H.263,1998) have in general similar features than those defined for MPEG-2. In this chapter we restrict our attention to the case of MPEG-2.

3.2 MPEG2 STANDARD OVERVIEW

MPEG-2 is widely used as the format of digital television signals that are broadcasted by terrestrial (over-the-air), cable, and direct broadcast satellite TV systems. As such, TV stations, TV receivers, DVD players, and other equipment are often designed to this standard. MPEG-2 was the second of several standards developed by the MPEG and is an international standard ISO/IEC 13818. Parts 1 and 2 of MPEG-2 were developed in a joint collaborative team with ITU-T, and they have a respective catalog number in the ITU-T Recommendation Series.

The MPEG-2 standard is divided into two main layers:

- Compression layer (includes audio and video streams)

- Systems layer (including timing information to synchronize video and audio as well as multiplexing mechanisms)

Thus MPEG-2 extends the basic MPEG system to provide compression support for TV quality transmission of digital video. To understand why video compression is so important, one has to consider the vast bandwidth required to transmit uncompressed digital TV pictures. Phase Alternate Line (PAL) is the analogue color TV transmission standard used in the Europe, and throughout many parts of the world. An uncompressed digital PAL TV picture requires a massive 216 Mbps data rate, far beyond the capacity of most radio frequency links. The U.S. uses an analogue TV system based on NTSC standard. This system provides less precise color information, and a different frame rate. An uncompressed digital NTSC TV signal requires slightly less data rate at 168 Mbps. The situation becomes much more acute, when one realizes the high definition TV (HDTV) which is around the corner. Now-a-days a High Definition TV picture requires data rate exceeding 1 Gbps (1000 Mbps).

MPEG-2 provides a way to compress this digital video signal to a manageable bit rate. The compression capability of MPEG-2 video compression is shown in the Figure 3.2. Since MPEG-2 standard exhibits good compression using standard algorithms, it has become the standard for digital TV. It has the following features:

- MPEG-2 Video compression is backwards compatible with MPEG-1
- Full-screen interlaced and/or progressive video (for TV and Computer displays)
- Enhanced audio coding (high quality, mono, stereo, and other audio features)
- Transport multiplexing (combining different MPEG streams in a single transmission stream)



Figure 3.2 Summary of MPEG Compression Capability

3.3 PARTS OF MPEG2 STANDARD

MPEG-2 is a standard currently in 9 parts. The first three parts of MPEG-2 have reached International Standard status; other parts are at different levels of completion. One has been withdrawn.

- ISO/IEC 13818-1:2000 Information technology -- Generic coding of moving pictures and associated audio information: Systems (available in English only)
Part 1: Systems specifies the system coding layer of the MPEG-2. It defines a multiplexed structure for combining audio and video data and means of representation the timing information needed to replay synchronized sequences in real time of MPEG-2. This is specified in two forms: the Program Stream and the Transport Stream. Each is optimized for a different set of applications.
- ISO/IEC 13818-2:2000 Information technology -- Generic coding of moving pictures and associated audio information: **Part 2: Video** (available in English only) Part 2 specifies set of rules for encoding and compression of video.

- ISO/IEC 13818-3:1998 Information technology -- Generic coding of moving pictures and associated audio information -- **Part 3: Audio** (available in English only) Part 3 specifies set of rules for encoding and compression of audio.
- ISO/IEC 13818-4:1998 Information technology -- Generic coding of moving pictures and associated audio information -- **Part 4: Conformance testing** (available in English only)
- ISO/IEC TR 13818-5:1997 Information technology -- Generic coding of moving pictures and associated audio information -- **Part 5: Software simulation** (available in English only)
- ISO/IEC 13818-6:1998 Information technology -- Generic coding of moving pictures and associated audio information -- **Part 6: Extensions for DSM-CC** (Digital Storage Media Command and Control). It is the specification of a set of protocols which provides the control functions and operations specific to managing MPEG-1 and MPEG-2 bitstreams.
- ISO/IEC 13818-7:1997 Information technology -- Generic coding of moving pictures and associated audio information -- **Part 7: Advanced Audio Coding (AAC)** (available in English only)
- **Part 8** of MPEG-2 was originally planned for coding of video when input samples are 10 bits. Work on this part was *discontinued* when it became apparent that there was insufficient interest from industry for such a standard.
- ISO/IEC 13818-9:1996 Information technology -- Generic coding of moving pictures and associated audio information -- **Part 9: Extension for real time interface for systems decoders** (available in English only)

- ISO/IEC 13818-10:1999 Information technology -- Generic coding of moving pictures and associated audio information -- **Part 10: Conformance extensions for Digital Storage Media Command and Control (DSM-CC)** (available in English only)

MPEG2-System is formally known as ISO/IEC 13818-1 and as ITU-T Rec. H.222.0. The video and audio data is encoded as described in ITU-T Rec. H.262|ISO/IEC 13818-2 and ISO/IEC 13818-3. MPEG-2 System layer (Part-1) contains timing and synchronization information to allow the MPEG player to multiplex the audio and video portions of the media file so that they are synchronized during playback.

Because our main objective in this thesis is to achieve clock synchronization for digital set-top box hence MPEG-2 System layer has been understood and described in further section in detail.

3.4 MPEG2 SYSTEM LAYER

It's a communication layer encapsulating compressed video, audio and data streams in packets. It multiplexes elements of a single program: video, audio, program related data.

This layer deals with multiplexing of multiple programs. It synchronizes all elements of a program and provides flexibility by allowing dynamic mix of content.

A program perhaps most easily thought of as a television program, or a Digital Versatile Disk (DVD) track contains a combination of elementary streams (typically one for video, one or more for audio, control data, subtitles, etc).

To understand how the component parts of the bit stream are multiplexed, we need to first look at each component part. The most basic component is known as an

Elementary Stream (ES). The Figure 3.3 shows that multiple types of signals are digitized and converted into ES suitable for the MPEG packetizers i.e. a MPEG channel that includes video, audio, and user data for a television message. It also shows that each media source is packetized and sent to a multiplexer that combines the channels into a single **Transport/Program stream**. The multiplexer also combines **Program Specific Information (PSI)** that describes the content and format of the media channels. The multiplexer uses a clock to time stamp the MPEG information to allow it to be separated and recreated in the correct time sequence at receiver side. In next subsections all components of MPEG- System layer is described.

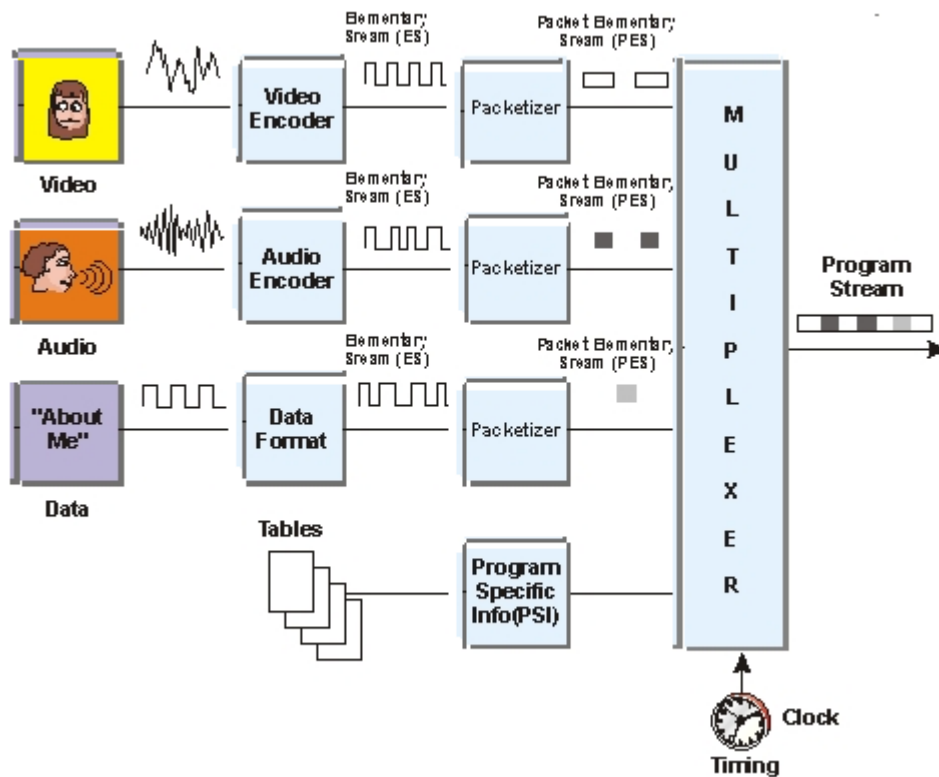


Figure 3.3 Formation of MPEG Stream

3.4.1 Elementary Stream (ES)

ES is an MPEG audio, video and data encoder output. ES contains a single type of (usually compressed) signal. There are various forms of ES including:

- Digital Control Data
- Digital Audio (sampled and compressed)
- Digital Video (sampled and compressed)
- Digital Data (synchronous, or asynchronous)

For video and audio, the data is organized into access units, each representing a fundamental unit of encoding. For example, in video, an access unit will usually be a complete encoded video frame.

3.4.2 Packetized Elementary Stream (PES)

Each endless Elementary Stream (ES) is input to an MPEG-2 packetizer which divides the ES into streams of convenient size packets (64 Kbytes). This stream is known as **Packetized Elementary Stream (PES)**. A PES packet may be a fixed (or variable) sized block, with up to 65536 bytes per block which includes a 6 byte protocol header also. A PES is usually organized to contain an integral number of ES access units. PES packet schematic is shown in Figure 3.4.

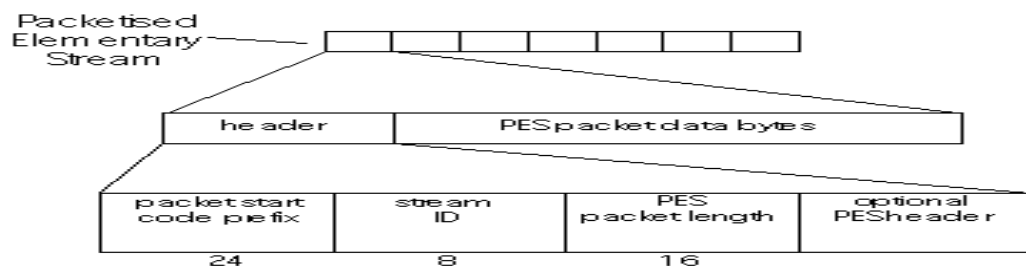


Figure 3.4 PES Packet Schematic

It is shown in Figure 3.4, that PES header starts with a 3 byte start code, followed by a one byte stream ID and a 2 byte length field. Various bits of PES header are discussed below:

- (i) The **packet_start_code_prefix** is a 24-bit code. Together with the `stream_id` it constitutes a packet start code that identifies the beginning of a packet. The `packet_start_code_prefix` is the bit string '0000 0000 0000 0000 0000 0001' (0x000001).

The following well-known stream IDs are defined in the MPEG standard:

1. 110x xxxx - MPEG-2 audio stream number x xxxx.
2. 1110 yyyy - MPEG-2 video stream number yyyy.
3. 1111 0010 - MPEG-2 DSM-CC control packets.

(ii) The next field contains the **PES Indicators**. These provide additional information about the stream to assist the decoder at the receiver. These indicators are defined as: `PES_Scrambling_Control`, `PES_Priority`, `Data_Alignment_Indicators`, `Copyright Information`, `Original_or_Copy`, `Presentation Time Stamp (PTS)`, `Decode Time Stamp(DTS)`, `Elementary Stream Clock Reference (ESCR)`, `Elementary Stream Rate`, `Trick Mode`, `CRC`, `PES Extension Information`

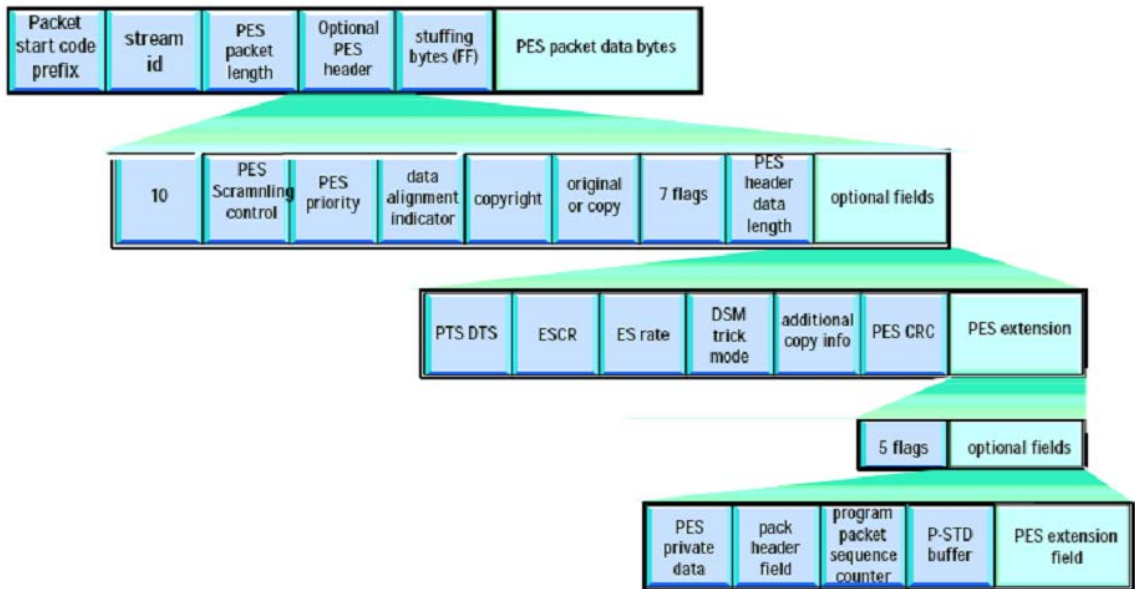


Figure 3.5 Detailed structure of PES Packet

PES_Scrambling_Control - Defines whether scrambling is used, and the chosen scrambling method. The 2 bit PES Scrambling Control field indicates the scrambling mode of the PES packet payload. When scrambling is performed at the PES level, the PES packet header shall not be scrambled.

Table 3.1 PES Scrambling Control

Value	Description
00	Not scrambled
01	User defined
10	User defined
11	User defined

- **PES_Priority** – It is 1-bit field . A “1” indicates a higher priority of the current PES payload. A multiplier can use PES_Priority bit to prioritize its data within an ES. This field shall not be changed by TS mechanism.
- **Data_alignment_indicator** - if set to 1 indicates that the PES packet header is immediately followed by the video or audio start code
- **Copyright information** – If it is =1, indicates that the payload is copyright protected.
- **Original_or_copy** - If it is =1, indicates this is the original ES. If it =0, i.e. contents of associated PES payload is a copy.

A one byte flag field completes the PES header. This defines the following optional fields, which if present, are inserted before the start of the PES payload.

Time Stamps

- **Presentation Time Stamp (PTS)** and **Decode Time Stamp (DTS)** – indicate exact time to decode or present audio or video i.e. these time stamps are used to synchronize a set of elementary streams and control the rate at which they are replayed by the receiver.
- **Elementary Stream Clock Reference (ESCR)** – It provides elementary stream clock reference.
- **Elementary Stream rate** - If it is =1, ESCR base and extension field are present in PES packet header.
- **Trick Mode** – It is 3-bit field. It indicates the video/audio is not the normal ES.
- **Copyright Information** - set to 1 to indicate a copyright protected ES.
- **CRC** – The Cyclic Redundancy Check to verify the correctness of data

- **PES Extension Flag** –A ‘1’-bit flag. If it is ‘1’, indicates that an extension field exist in PES packet header. It may be used to support MPEG-1 streams. The PES packet payload includes the ES data. The information in the PES header is, in general, independent of the transmission method used.

Two options are possible for inserting PES data into the TS packet payload:

1. The simplest option, from both the encoder and decoder viewpoints, is to send only one PES (or a part of single PES) in a TS packet. This allows the TS packet header to indicate the start of the PES, but since a PES packet may have an arbitrary length, also requires the remainder of the TS packet to be padded, ensuring correct alignment of the next PES to the start of a TS packet. In MPEG-2 the padding value is the hexadecimal byte 0xFF.
2. In general a given PES packet spans several TS packets so that the majority of TS packets contain continuation data in their payloads. When a PES packet starts, the `payload_unit_start_indicator` bit is set to ‘1’ which means the first byte of the TS payload contains the first byte of the PES packet header. Only one PES packet can start in any single TS packet. The TS header also contains the PID so that the receiver can accept or reject PES packets at a high level without burdening the receiver with too much processing. This has an impact on short PES packets.

3.5 MPEG- 2 MULTIPLEXING

The MPEG-2 standard allows two forms of multiplexing:

- **MPEG Program Stream (PS)** A group of tightly coupled PES packets referenced to the same time base. Such streams are suited for transmission in a

relatively error-free environment and enable easy software processing of the received data. The PS is widely used in digital video storage devices, and also where the video is reliably transmitted over a network (e.g. video-clip download).

Thus PS has following characteristics:

- combine one or more PES with a common time base into a single stream
- designed for error free environments
- packets of variable length
- works well on a single program with variable bit-rate in recording environment (used in DVD)

- **MPEG Transport Stream (TS)** Each PES packet is broken into fixed-sized transport packets forming a general purpose way of combining one or more streams. TS format is designed to carry digital video and audio over possibly lossy media, such as broadcasting, examples of which include ATSC and DVB. This is suited for transmission in which there may be potential packet loss or corruption by noise, or/and where there is a need to send more than one program at a time. Digital Video Broadcast (DVB) uses the MPEG-2 Transport Stream over a wide variety of under-lying networks. Thus TS has following characteristics:

- combine one or more PES with one or more independent time bases into a single stream (sometimes called multiplex)
 - First byte of the PES packet must be the first byte of the Transport packet
 - Each transport packet must contain data from only one PES packet
- designed for environments where errors are likely to occur

- packets are 188 bytes in length
- works well on multiple programs in a fixed bit-rate transmission environment

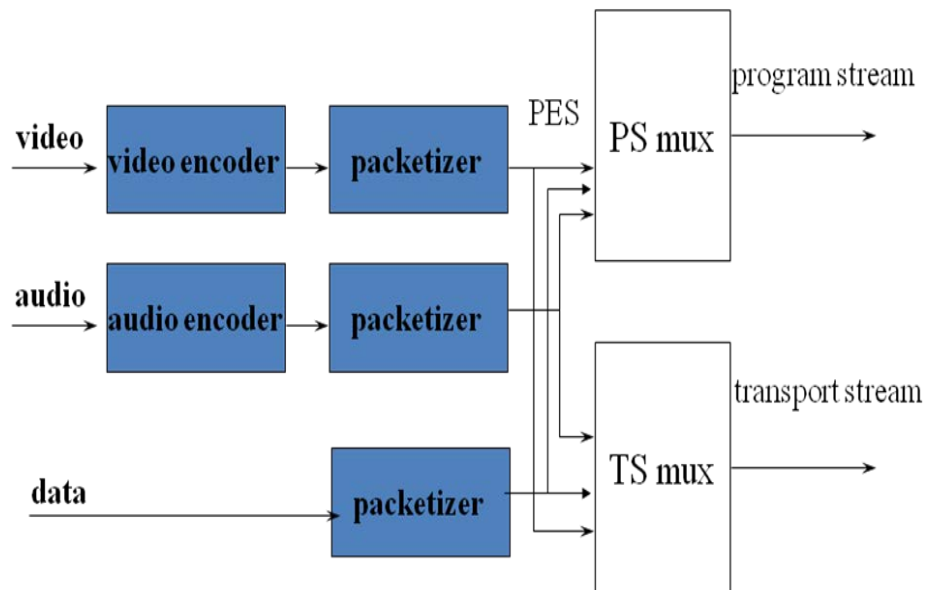


Figure 3.6 Functional Block diagram of Standard MPEG2

Note: Since both the Program Stream and Transport Stream multiplex a set of PES inputs, interoperability between the two formats may be achieved at the PES level. Since in Set-top box, TS format is used hence TS syntax has been provided in detail in section 3.4

3.5.1 MPEG Transport Stream

The Transport Stream is a stream definition which is tailored for communicating or storing one or more programs of coded data according to ITU-T Rec. H.262|ISO/IEC 13818-2 and ISO/IEC 13818-3 and other data in environments in which significant errors may occur. Such errors may be manifested as bit value errors or loss of packets. The format of the transport stream is described using the Figure 3.7. This Figure shows two elementary streams sent in the same MPEG-2 transport stream. Each packet is associated

with a PES through the setting of the PID value in the packet header (the values of 64 and 51 in the figure). The audio packets have been assigned PID 64, and the video packets PID 51 (these are arbitrary, but different values). As is usual, there are more video packets than audio packets, but it can be observed that two types of packets are not evenly spaced in time. The MPEG-TS is not a time division multiplex, packets with any PID may be inserted into the TS at any time by the TS multiplexer. If no packets are available at the multiplexer, it inserts null packets (denoted by a PID value of 0x1FFF) to retain the specified TS bit rate. The multiplexer also does not synchronize the two PESs, indeed the encoding and decoding delay for each PES may be different. A separate process is therefore required to synchronize the two streams (see below).



Figure 3.7 MPEG Transport Stream

The syntax of transport stream is shown in Figure 3.8. It consists of a sequence of fixed sized transport packet of 188 bytes. Each packet comprises 184 byte payload and 4 byte header. One bit in this 4 byte header is 13 bit Packet Identifier (PID) which plays a key role in the operation of the TS.

The header (4 Byte) has the following fields:

- The header starts with a well-known **Synchronization Byte (8 bits)**. This has the bit pattern 0x47 (0100 0111).
- A set of three flag bits are used to indicate how the payload should be processed.
 1. The first flag indicates a **transport error**.

2. The second flag indicates the start of a payload
(**payload_unit_start_indicator**)
 3. The third flag indicates **transport priority bit**.
- The flags are followed by a **13 bit Packet Identifier (PID)**. This is used to uniquely identify the stream to which the packet belongs (e.g. PES packets corresponding to an ES) generated by the multiplexer. The PID allows the receiver to differentiate the stream to which each received packet belongs. Some PID values are predefined and are used to indicate various streams of control information. A packet with an unknown PID, or one with a PID which is not required by the receiver, is silently discarded. The particular PID value of 0x1FFF is reserved to indicate that the packet is a null packet (and is to be ignored by the receiver).
 - The **two scrambling control bits** are used by conditional access procedures to encrypt the payload of some TS packets.
 - Two adaptation field control bits which may take four values:
 - a) 01 – no adaptation field, payload only
 - b) 10 – adaptation field only, no payload
 - c) 11 – adaptation field followed by payload
 - d) 00 - RESERVED for future use
 - there is a half byte **Continuity_counter** (4 bits) : increments with each TS packet with same PID :
 - a) Wraps to 0 after max value
 - b) Not incremented when adaptation_field_control is 00 or 01
 - c) Used to find packet loss

There may be Optional Transport Packet Adaption Field. The presence of an adaptation field is indicated by the adaption field control bits in a transport stream packet. If present, the adaption field directly follows the 4 byte packet header before any user payload data. It may contain a variety of data used for timing and control.

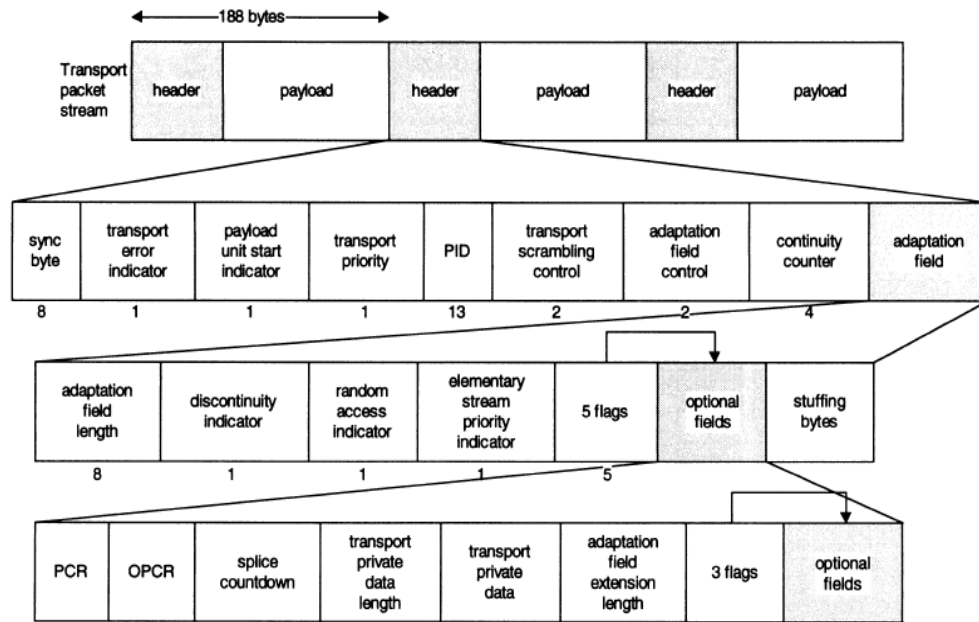


Figure 3.8 Detailed Syntax of Transport stream

- One important item in most adaption packets is the **Program Clock Reference (PCR)** field. This carries the Encoder clock time stamps.
- Another important item is **splice_countdown** field. This field is used to indicate the end of a series of ES access units. It allows the MPEG-2 TS multiplexer to determine appropriate places in a stream where the video may be spliced to another video source without introducing undesirable disruption to the video replayed by the receiver. Since MPEG-2 video uses inter-frame coding a seamless switch-over between sources can only occur on an I-frame boundary (indicated by a splice count of 0). This feature may, for instance be used to insert a news flash in a scheduled TV transmission.

- One other bit of interest here is the **transport_private_data_flag** which is set to 1 when the adaptation field contains private data bytes.
- Another is the **transport_private_data_length field** which specifies how many private data bytes will follow the field. Private data is not allowed to increase the adaptation field beyond the TS payload size of 184 bytes.

Different payload types in a TS packet are shown in Figure 3.9.

sync '0x47'	tei (1)	pusi (1)	tp (1)	PID (13)	tsc (2)	afc '01'	cc (4)	Payload data_bytes (184 bytes)	
sync '0x47'	tei (1)	pusi (1)	tp (1)	PID (13)	tsc (2)	afc '10'	cc (4)	Adapt. Field (AF bytes)	Decoder ignores these 184-AF bytes
sync '0x47'	tei (1)	pusi (1)	tp (1)	PID (13)	tsc (2)	afc '11'	cc (4)	Adapt. Field (AF bytes)	Payload data_bytes (184-AF bytes)

Figure 3.9 Three types of TS Packets

Transport stream has a concept of programs, which are groups of one or more PIDs that are related to each other. For instance, a transport stream used in digital television might contain three programs, to represent three television channels. Suppose each channel consists of one video stream, one or two audio streams, and any necessary metadata. A receiver wishing to tune to a particular "channel" merely has to decode the payload of the PID's associated with its program. It can discard the contents of all other PID's after the adaptation field, in the TS packet. The payload can be either PES or section field containing PSI (Program specific Information) tables. Section field contain information about program number, scrambling control, network parameters etc. In TS each transport packet is tagged with an appropriate Packet ID (PID) value indicating to which ES its payload belongs. There can be many ES comprising many different programs. Additional information required for decoder is called PSI and is present in every TS. The PSI tables in MPEG-2 consist of a description of the elementary streams

which need to be combined to build programs, and a description of the programs. Figure 3.10 and Figure 3.11 shows structure of PSI tables. Each PSI table is carried in a sequence of **PSI Sections**, which may be of variable length (but are usually small). Each section is protected by a CRC (checksum) to verify the integrity of the table being carried. The length of a section allows a decoder to identify the next section in a packet. A PSI section may also be used for down-loading data to a remote site. Tables are sent periodically by including them in the transmitted transport multiplex

Table 3.2 PSI Tables

Table Name	PID #	Description
Program Association Table (PAT)	0	Associate Program number with PMT
Program Map Table(PMT)	Assigned	Associate PID's with Programs
Network Information Table(NIT)	Assigned	Contains physical network parameters
Conditional Access Table(CAT)	1	Associate PID's with private stream

Program Association Table: The PAT lists the PIDs of tables describing each program. The PAT is sent with the well-known PID = 0. It provides the correspondence between a program_number and the PID value of the transport stream packets which carry the program definition. The program_number is the numeric label associated with a program.

Table 3.3 Example of PAT for Three Program Multiplex

Program	PMT PID	Meaning
2	32	PID 32 contains map for program 2
3	48	PID 48 contains map for program 3
4	64	PID 64 contains map for program 4

PAT provides correspondence between a Program number and the PMT PID that carries program definition.

- Program # is similar to Channel # in broadcast TV.
- PAT always assigned as PID 0

Program Map Table: The PMT provides the mappings between program numbers and the program elements that comprise them i.e. defines the set of PIDs associated with a programme, e.g. audio, video, auxiliary data and PCR etc.

Table 3.4 Example PMT for Program 2

PID	Stream Type
32	PMT
33	Video & PCR
36	Audio
42	Data

PMT provides correspondence between a Program number and the elementary streams that comprise it. Descriptors may be sent to provide more information about the program and/or program elements. PMT stream type describes the elementary stream e.g. MPEG-1 Video, MPEG-2 Video, MPEG-1 Audio, MPEG-2 Audio, Private Sections, PES Private Data, DSM-CC, User Private data (e.g., AC-3 Audio)etc.

Network Information Table: NIT is optional and its contents are private. If it is present, it is represented by Program 0 in PAT. It contains physical network parameters such as RF frequencies, Satellite Transponder Numbers, etc.

Conditional access Table: The CAT is sent with the well-known PID = 1 and must be present if a stream is scrambled. This table defines type of scrambling used and PID values of transport streams which contain the conditional access management (ECM) and entitlement management information (EMM)).

Table 3.5 Example CAT of Program 2

CA System	CA PID
1	201
2	202
3	203

CAT provides correspondence between CA systems and their Entitlement Management Message (EMM) streams. EMM's are system-wide private streams that specify authorization levels of specific decoders.

Private Tables: These are provided for transmission of private data. It can be used for sending non-MPEG data, such as stock quotes, downloadable software modules, etc.

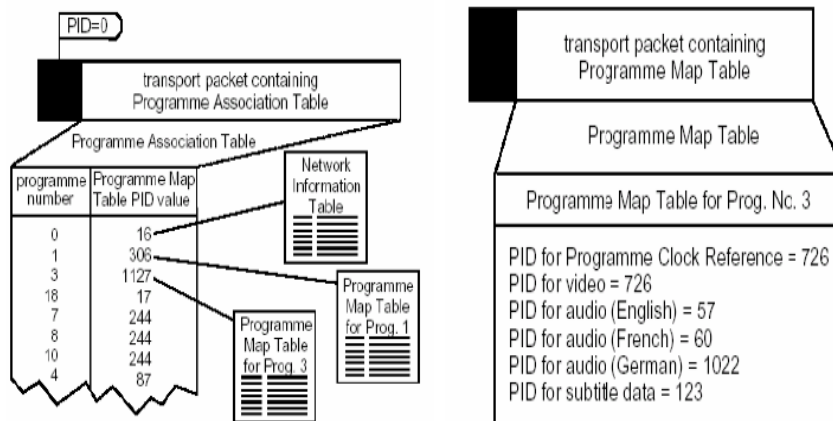


Figure 3.10 Example of PSI Tables

Event Information Table (EIT): Contains information about present, following and future events.

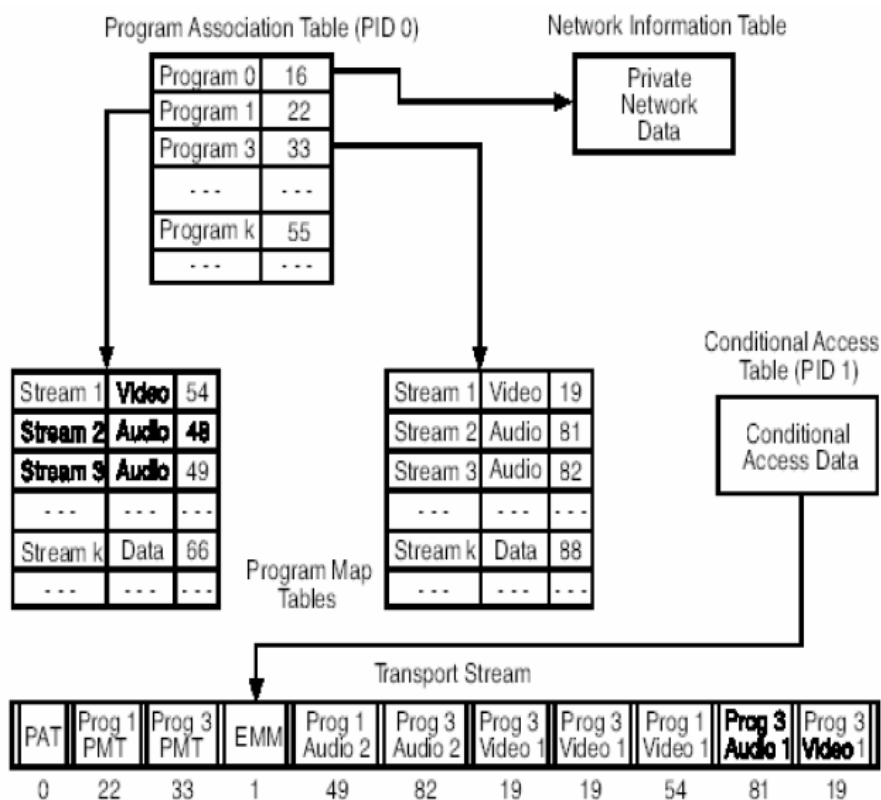


Figure 3.11 PSI Table Structures

To identify the required PID to de-multiplex a particular PES, the user searches for a description in a particular table, the Program Association Table (PAT). This lists all programs in the multiplex. Each program is associated with a set of PIDs (one for each PES) which correspond to a Program Map Table (PMT) carried as a separate PSI section. There is one PMT per program. PAT/PMT form a “Mini Program Guide” and certain information in those tables and in PSIP (Program specific information protocol) must be consistent.

As MPEG is basis in dealing with digital set-top box. Thus in this chapter, MPEG standard has been presented in detail. This chapter has provided a solid background to work on clock recovery. In next chapter, Set-top box system is presented.

Chapter 4: Set-Top Box Subsystem

The research work in this thesis is mainly aimed at clock recovery and clock synchronization for various broadcasting environments for digital set-top box, hence set-top box (STB) system and various components associated with it has been described in this chapter. Brief description and functioning of a latest STB decoder chip has been explained to understand the capability of STB.

4.1 INTRODUCTION

A set-top box is a device that connects to a television and an external source of RF signal, turning the signal into content which is displayed on the television screen. Today, it offers more than the basic definition of set-top box like interactive viewing, subtitles in various languages, cable modem etc. Set-top box is equivalent to a computer that process digital information. These typically have on-screen user interfaces that can be seen on the TV screen and interacted with through the use of an hand-held interactive keypad, which is little more than an advanced remote control. Set-Top Box also has facilities for upgrading software such as browsers and Electronic Program Guides (EPGs). Some have huge hard-drives and smart card slots to put your smart card into it for purchases and identifying yourself to your cable, satellite TV provider. Many set-top boxes are able to communicate in real time with devices such as camcorders, DVDs, CD players, music keyboards, video-conferencing, sending e-mail and cable telephony etc.

Set-top Boxes act as a gateway between television or PC or PC-TV and telephone, satellite or cable feed (incoming signal.) In terms of ITV, the STB receives encoded and/or compressed digital signals from the signal source (satellite, TV station, cable

network, etc.) and decodes (and/or decompresses) those signals, converting them into analog signals displayable on your television. Thus a set-top box, or STB, refers to any device inserted between the cable or satellite feed and the user's television set. These devices have the capability to select and display individual channels. Satellite television today follows the MPEG-2 digital-compression standard, which is the basis of both the Digital Video Broadcast standard (DVB) and the Advanced Television Systems Committee (ATSC) standard. Most U.S. cable providers have either adopted the MPEG-2 standard or are in the process of upgrading their services from analog to digital in order to do so. MPEG-2 allows the delivery of any information that can be digitized, including audio, video and text. MPEG-2 stream can be easily displayed on digital High Definition television sets, or in analog NTSC, PAL or SECAM formats with help of set-top box. Thus it is possible to get digital quality pictures and lots of flexibility due to powerful set-top box. The STB is a combination of hardware and software modules for audio-video decoding, tuning, encryption etc. as discussed in further sections.

4.2 SET-TOP BOX COMPONENTS

To provide interactive services, the Set-top Box, from the standpoint of its hardware, needs four important components: a network, an interface, a buffer, as well as decoder/synchronization hardware.

4.2.1 The Network Interface

Allows the user to receive data from the server and send data back to the server, in a manner that the server can understand it.

4.2.2 The Decoder

In order to save storage space, disk bandwidth, and network bandwidth, movies are usually encoded (compressed) before they are sent over the network. Thus, the end-users need a decoder to decode (decompress) the incoming stream's data before it's viewable.

4.2.3 The Buffer

Due to delay jitters in the network, the arrival time of a video stream cannot be determined exactly. In order to guarantee continuous consistent playback for the viewer (end-user/subscriber) the stream is often received one or even a few seconds before it's actually seen by the end-user. This way if there are fluctuations (even those measured in milliseconds) in the transport time of the video stream to that receiver, the viewer won't know the difference as their buffer has a bit of time to spare.

4.2.4 Synchronization Hardware

Digital TV signal consists of both video and audio streams. They must be synchronized with each other before being viewed. In simple words it is a kind of a computer that translates digital signals into a format that can be viewed on a television screen. Today digital TV usually requires a STB, which is used to decode and tune digital signals and converts them to a format that is understood by analog TV. In a STB, the tuner receives a RF signal from a cable, satellite or terrestrial network and tunes to a particular channel.

Set-top Boxes are being designed to be integrated with game consoles, DVDs, videophones, telephones and televisions. Ideally Set-top Boxes should have as much

interoperability as possible. Interoperability refers to the characteristics of a Set-top Box that allow it to operate on equipment made by different manufacturers.

4.3 SET-TOP BOX FUNCTION

Majority of the functions are as below-

- Decodes the incoming digital signal
- Verifies access rights and security levels
- Displays studio-quality pictures on TV set

The STB is comprised of front-end and back-end modules as shown in Figure 4.1.

The vertical line demarcates the front-end and back-end. Front-end comprises of a tuner, FEC and digital demodulator, Back-end comprises of transport stream (TS) demultiplexer, A/V decoder, digital video encoder, audio DAC and CPU along with on-site memory. Major blocks functions are as below-

- Tuner receives a RF signal from a cable/satellite or terrestrial network.
- The digital demodulator demodulates into the digital format, checks for errors & forwards it to the de-multiplexer.
- De-multiplexer extracts the audio, video and data from the digital stream and sends it to the appropriate decoding blocks.

Thus front-end is responsible of translating RF signal into a digitally demodulated error corrected TS (Transport Stream).

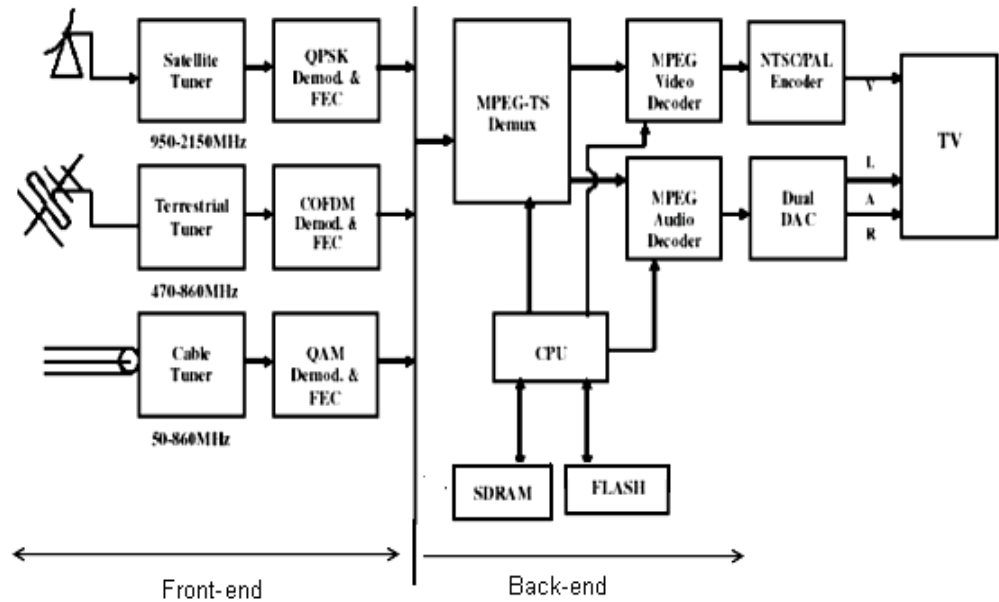


Figure 4.1 Schematic of Digital Set-Top Box

The demodulation techniques used for different broadcast media are:

- QPSK for satellite transmission
- OFDM for Terrestrial transmission
- QAM for Cable transmission

Digital demodulator along with tuner is called Network Interface Module (NIM).

NIM is controlled using I²C bus to select required channel, convert into intermediate frequency and fed to demodulator. After demodulation, the back-end module demultiplexes, descrambles and processes the raw TS data coming from the front-end to be used by audio-video decoders. TS demultiplexer extracts audio, video and clock and sends to A/V decoder and smart card. Smart card determines access rights to various digital services. A/V decoder decompresses audio, video data which are fed to DAC and video encoder to display on TV receiver. Thus digital A/V signal is converted to analog format for SD viewing or to suitable digital stream for HD viewing, depending on the SoC used and features supported by it.

4.4 FUNCTIONING OF STx7109 HDTV SET-TOP BOX DECODER CHIP

The STx7109 is designed around the well-proven Omega2 (STBus) interconnect and presents a new generation of HD video decoders for VC1 Microsoft® video (including Windows Media® video 9, WMV-9) as well as H.264 (MPEG-4 part 10) and MPEG-2 high-definition (HD) decoding. This SoC(system-on-chip) (STB-7109) is a full back-end processor for digital terrestrial, satellite, cable, DSL (Digital Subscriber Line) and IP client high-definition set-top boxes. It also includes all processing for DVD applications. Different applications by using this chip are presented in appendix-I.

The STx7109 integrates a Win CE compatible 266 MHz ST40-202 processor core that features a 32-bit superscalar RISC CPU and IEEE-754 compliant floating point unit (FPU). The ST40-202 includes 2-way set-associative caches, 16 Kbytes of instruction cache and 32 Kbytes of data cache, as well as core support peripherals such as a real-time clock and interrupt controller.

The STx7109 demultiplexes decrypts and decodes HD or SD (Standard Definition) video streams with associated multi channel audio. Video is output to two independently formatted displays: a full resolution display intended for a TV monitor, and a down sampled display intended for a VCR or DVD-R. Connection to a TV or display panel can be analog through the DACs, or digital through a copy protected DVI/HDMI (Digital Video Interface/High Definition Memory Interface). Composite outputs are provided for connection to the VCR with macro vision protection. Audio is output with optional PCM mixing to an S/PDIF interface, PCM interface, or through integrated stereo audio DACs. Digitized analog programs can also be input to the STx7109 for reformatting and display. The STx7109 includes a graphics rendering and display

capability with a 2D graphics accelerator, two graphics planes and a cursor plane. A dual display compositor provides mixing of graphics and video with independent composition for each of the TV and VCR/ SD TV outputs. The STx7109 includes a stream merger to allow three different transport streams from different sources to be merged and processed concurrently.

Applications include DVR time-shifted viewing of a terrestrial program, while acquiring an EPG/data stream from a satellite or cable front end. The STx7109 embeds a 266 MHz ST40-202 CPU for applications and device control. A dual DDR1 SDRAM memory interface is used for higher performance, to allow the video decoder the required memory bandwidth for HD VC1/H.264 and sufficient bandwidth for the CPU and the rest of the system. A second memory bus is also provided for flash memory, storing resident software, and for connection of peripherals. This bus also has a high speed synchronous mode that can be used to exchange data between two STx7109 devices. This can be used to connect a second STx7109 as a co-decoder for a dual TV STB application.

A hard-disk drive (HDD) can be connected either to the SATA interface, or as an expansion drive through the USB 2.0 port. The USB or Ethernet interfaces can also be used to connect to a DOCSIS 2.0 CM gateways for interactive cable applications.

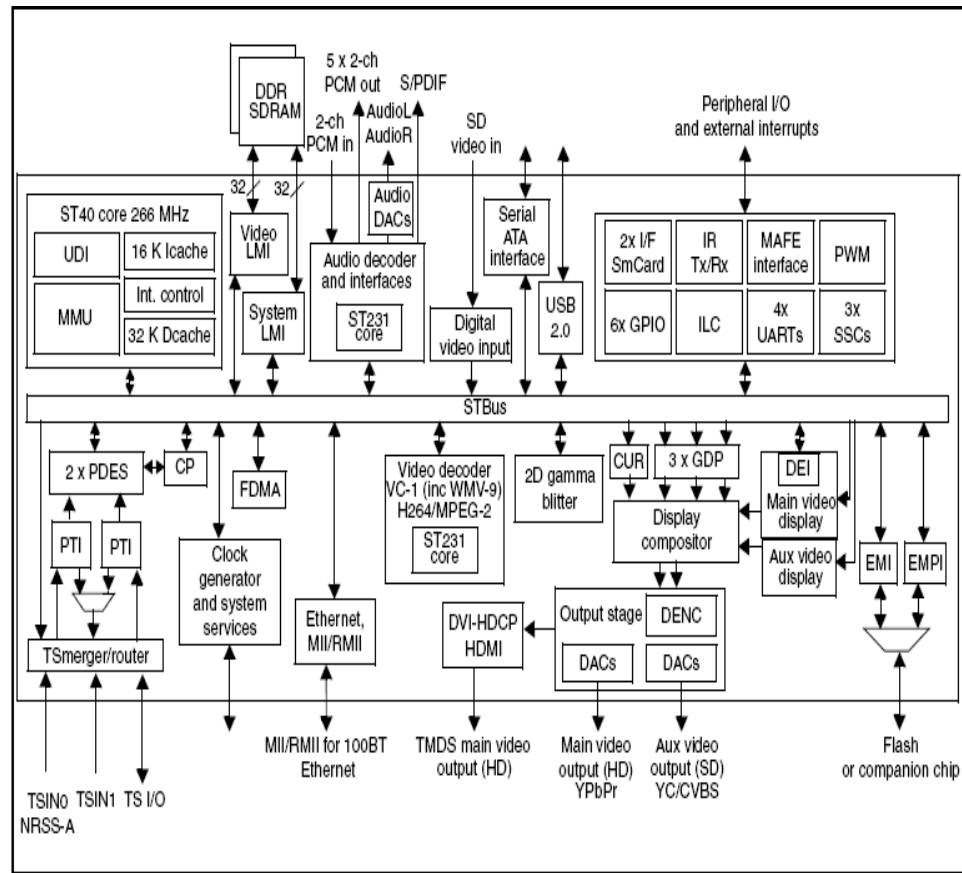


Figure 4.2 7109 block diagram

Set-top box (STB) transport streams are received and processed by the TS subsystem. DVD streams are received through the serial ATA (SATA) interface, processed by the decryption cell and then demultiplexed. The resulting PES streams and section data are stored in memory buffers in DDR1 SDRAM attached to the local memory interface (LMI).

A flexible DMA controller (FDMA) performs PES parsing and starts code detection and routes elementary streams to audio and video bit buffers in DDR1 SDRAM. A video decoder decodes HD or SD video streams. Audio decoding and PCM mixing is performed by the audio decoder and output through S/PDIF and PCM interfaces. Audio can also be output through an integrated 24-bit stereo DACs system. After video

decoding, two independently formatted video displays (main and auxiliary) can be generated, and each mixed independently with graphics to create main and auxiliary display compositions. The main display composition (HD or SD) can be supplied as analog RGB or YPbPr outputs and, through a copy protected DVI/HDMI, as digital outputs. The auxiliary display composition (SD only) can be output as YPbPr analog component or composite video on a separate interface for connection to a VCR. A digital video input interface allows the STx7109 to receive SD non compressed digital video and to output this through the main and auxiliary displays in place of decoded video. A separate PCM input allows any associated audio to be received, mixed and output in place of decoded audio. The graphics subsystem comprises a separate 2D blitter, two graphics planes, a cursor plane and dual display compositor. Graphics buffers are created, stored in and displayed from buffers in DDR1 SDRAM.

The STx7109 embeds an ST40-202 CPU core for applications and data processing, and device control. The CPU boots from flash/SFlash™ on the external memory interface (EMI) and can execute in place, or transfer the main executable to the DDR1 SDRAM and execute from there. CPU data is held in DDR1 SDRAM where cacheable and non-cacheable regions can be programmed. The 16-bit EMI is also used for connecting to external peripherals. System performance is enhanced with the multi-channel FDMA, which can be used for 2D block and stream data transfers with minimal CPU intervention. DVR applications are supported using a hard-disk drive (HDD) connected either to the serial ATA interface, or to the USB 2.0 interface. IP-TV and home networking applications are also supported by the integration of an Ethernet controller with integrated MAC (Media access control). The STx7109 also integrates a range of

peripherals, system services, and a clock generator module with embedded VCXO (voltage controlled oscillator), to significantly reduce external component cost.

In this chapter, basic concepts of Set-top box have been presented. In section 4.5 functioning of a typical low cost set-top box decoder chip has been explained. Various blocks of Set-top box decoder SoC, and how a set-top box is able to perform important functions for digital TV has been explored. This chapter has provided a background for understanding and analyzing clock recovery problem in various broadcasting environments for set-top box.

Chapter 5: Clock Recovery for Satellite TV

In this chapter, first concept of clock synchronization has been explained, and then in next section jitter in satellite environment has been analyzed. Further a newly designed Clock Recovery algorithm for Satellite broadcasting for STB has been presented. In the last section simulation results and performance in real environment has been illustrated.

5.1 INTRODUCTION TO CLOCK SYNCHRONIZATION

Clock synchronization is an essential feature of modern digital network and digital system. Synchronization is of fundamental importance at both the application level (voice, audio, video and graphics), and at transmission level (packet, cell, symbol and bit).

In a Set-top box subsystem, synchronization of decoder clock with the encoder-side clock is one of the most important parts of MPEG standard since it is absolutely critical to extract the timing information accurately in order to play the audio/video smoothly at the decoder end. A typical block diagram of the clock synchronization principle in set-top box is shown in Figure 5.1. As audio and video have different sampling rates and different frame sizes, it is convenient to have a common single clock at the encoder providing the timing information of audio as well as of the video. The clock information is then used by the decoder side as a reference to retrieve the stream timing information to decode the audio/video data accurately and then to synchronize audio with the video. At the encoder, the clock reference is encoded in the form of timestamps termed as **Program Clock Reference (PCR)**. The decoder side uses clock information coming in the stream as a reference (PCR) to retrieve the stream-timing

information to decode the audio-video data accurately. The timestamps from the local decoder clock are termed as **System Time Clock (STC)**. The encoder contains a 27 MHz master oscillator running at worst case tolerance (MPEG-System) of ± 810 Hz. The encoder samples the 33bit counter periodically to provide PCR base value. Encoder also provides the 9-bit remainder value as a PCR extension. These two values (PCR Base Value and 9 bit extension) are then multiplexed with the audio-video data and transmitted in the transport stream. PCR time stamps are required to occur at maximum 100ms intervals. DVB recommends PCR insertion rate at 40ms. VCXO runs the counter to derive the System Time Clock as shown in Figure 5.1.

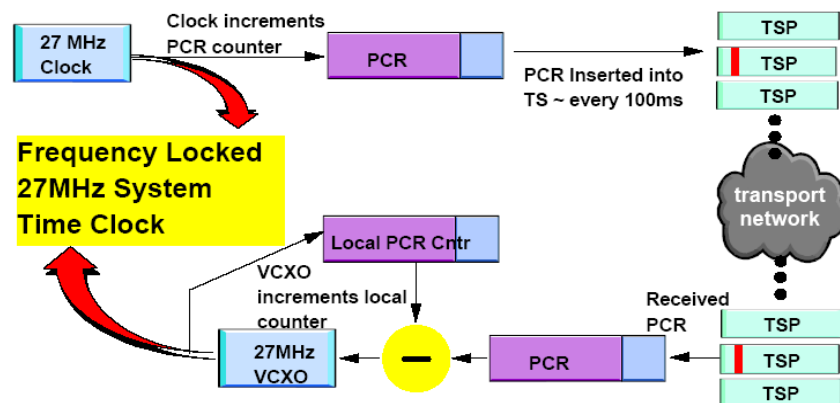


Figure 5.1 PCR mechanism

Decoder system is driven by a separate independent 27 MHz clock providing local clock time stamps also called System Time Clock (STC). The STC first divides the 27MHz by 300, giving a 90 kHz clock that is counted by a 33-bit counter giving the base STC value. The remainder is taken as a 9-bit value, used as STC extension. The 33-bit counter at decoder side wraps around to zero after a period of ~ 95443 seconds (26.51 hours). Since the encoder clock can run anywhere in between $27\text{MHz} \pm 810\text{Hz}$, decoder does not know the exact frequency of the encoder side clock. It only knows the minimum and maximum values of the encoder clock. So there is a need to find out the exact

frequency of the encoder side to run the decoder system at the correct frequency for any transport stream. Now let us consider the worst case. Suppose an encoder side, clock is running at 27MHz-810Hz while the decoder side is running at 27 MHz + 5.4 KHz. Total error in the system is $810+5.4 \text{ KHz} = 6210 \text{ Hz}$ which we need to correct. Such large error will result in video sync loss or in the worst case audio/video data loss. Data loss may occur because if we are not playing the data at the correct rate, it will lead to accumulation of the data if we are slow compared to the encoder clock and eventually data buffer overflows resulting in data loss. On the other hand, if we are running fast, we may dissipate the data all too soon resulting in no more data to play. In both these cases, final result is bad audio with glitches, no video on the TV sometimes and/or color loss of the video. Therefore, it is very essential to recover the encoder side clock at the decoder end for smooth viewing on TV. This is known as clock recovery.

As shown in Figure 5.2, if the encoder transmits N PCR values, in a given absolute time period T, then the local receiver must receive 'N' PCR values in the same absolute time period 'T'. The PCR time stamp is the encoder's relative measure of the time when a PCR value was transmitted. Similarly, the STC time stamp is the local receiver's relative measure of the time when a PCR value was received.

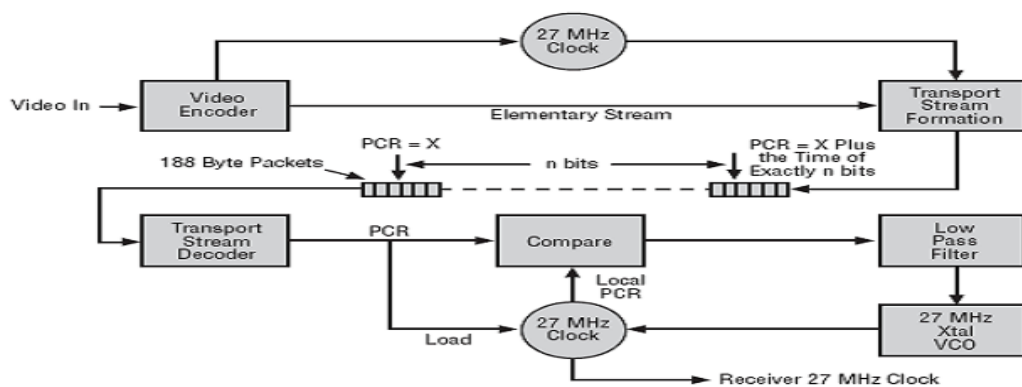


Figure 5.2 PCR mechanism in Transport Stream

After N PCR received by the decoder, the difference between the time elapsed as measured by the encoder, and the time elapsed as measured by the local receiver is the ‘clock difference’. This discrepancy can be used to correct the local receiver clock.

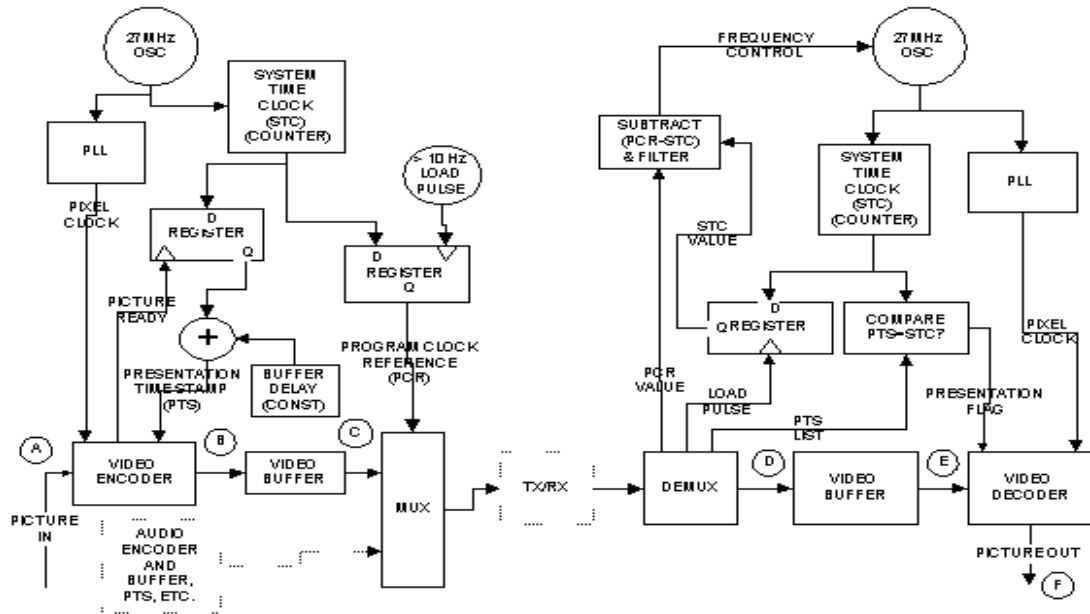


Figure 5.3 Timing Diagram in a Typical Set-top Box

A typical timing block diagram of Set-top box subsystem is shown in Figure 5.3. It has two parts- Encoder and Decoder. The MPEG audio video encoded and output data is formatted into elementary Stream (ES). At the encoder side, an endless ES is broken into number of packets of convenient size (64 Kbyte) called as packetized elementary stream (PES). Header is also added in this PES that contains PTS (Presentation Time Stamp) and DTS (Decoding time stamp). Time stamps in each PES ensure lip-synchronization between the video and audio. The PES is then further subdivided into transport stream (TS) 188 byte packet size that contains 4 byte Header and 184 Byte Payload. Thus PES packets are being broken into short fixed-size packets and multiple programs encoded with different clocks can be carried out. This is possible because the stream has program clock references (PCR's) mechanism which allows transmission of

multiple clocks, one of which is selected and regenerated at the decoder. At decoder side, TS is demultiplexed by demultiplexer block and converted into PES. The PCR information is extracted from TS and fed to frequency control block for encoder and decoder clock synchronization. The PES parser in demultiplexer block converts PES into ES. This ES is then fed to audio and video-decoder. PTS and DTS are extracted from PES and used by audio and video decoder drivers for their synchronization.

Clock recovery is a process of synchronizing set-top box decoder system clock with encoder system clock. It is a mechanism to adjust the locally generated clock with the encoder side clock referenced in the Program Clock Reference (PCR) located in the adaptation field of the incoming transport stream. In a typical Set-Top Box decoder, System Time Clock is derived from an adjustable 27MHz clock source (with a tolerance of 5.4 KHz). Decoder latches local STC counter value as soon as PCR packet arrives at decoder. Output frequency is adjusted at the decoder side accordingly. The multiplexed bit stream from the encoder is made available to set-top box decoder system through a broadcasting channel (satellite, cable, terrestrial or IP) or storage medium (CD etc). Thus specific contribution of our research work is to develop clock-recovery algorithms at decoder side for extracting the transmitted PCRs in transport stream and accordingly adjustment of the decoder clock.

In a set-top box typically there are two ways of frequency adjustment at decoder side.

5.1.2 VOLTAGE CONTROLLED OSCILLATOR (VCXO) BASED CLOCK RECOVERY MODULE

In a VCXO type device as shown in figure 5.4, there is a single clock source (voltage controlled oscillator) used for both audio and video. A single 27 MHz clock is

modified, using a PWM pulse train, and as a consequence all other clocks derived from this clock are modified. Frequency change affected by varying the Mark/Space ratio linear over a sub-range of available PWM Mark values, from 40 to 216. Over this range a change in frequency is of approximately 61 ticks per second per PWM Mark change

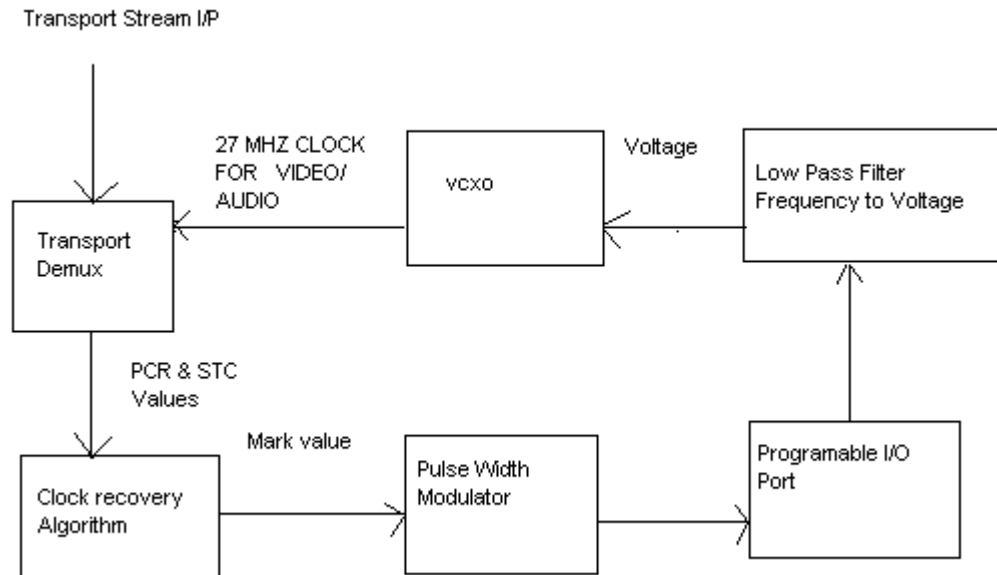


Figure 5.4 VCXO based clock recovery

5.1.2 FS BASED CLOCK RECOVERY MODULE

In Frequency Synthesizer (FS) based system, a single fixed frequency crystal acts as the source to multiple independently controllable clocks by programming FS registers. Frequency generated by the frequency synthesizer is controlled by programming various FS registers.

5.2 ANALYSIS OF JITTER IN DVB-S ENVIRONMENT

For detailed analysis of satellite environment, in STB decoder, Event Manager (EVT is used to receive events from one software driver to other software driver) and

Programmable Transport Interface (PTI for demultiplexing the transport stream and extract the PCR) software modules are used. These modules help in providing PCR & STC to clock recovery module. When PCR packet reaches the PTI, PCR event occurs and then PTI call-back function will be invoked and inform the clock recovery module. In one of the parameters of PTI call-back function, we get the PCR value and corresponding STC value. Five hundred samples of raw PCR value (PCR packet value received by the backend), raw STC value (STC value corresponding to the PCR packet received) are collected. The difference between two successive PCR packet values (PCRdiff) and the difference between two successive STC values (STCdiff). Mismatch (deltas) between corresponding PCR differences and STC differences (STCdiff – PCRdiff) is calculated. This (STCdiff - PCRdiff) gives us the jitter value. We also noted that PCRdiff and STCdiff values are in milliseconds along with the system time when the PCR packet is gathered by the PTI.

In case of satellite transmissions, the STC follows the PCR almost always and is in sync. This indicates that jitter effect is not very high in satellite transmission.

5.3 MOVING WINDOW BASIC AVERAGING ALGORITHM FOR DVB-S BROADCASTING

A classical PLL based clock recovery approach has been explained in Figure 5.6. In this clock recovery system error signal PCR (Program Clock Reference)-STC (System Time Clock)) is filtered to eliminate the effects of the network jitter, resulting in a correction value to be applied to the local clock generator. The newly corrected clock will be the base for the subsequent STC value read upon the arrival of the next PCR time stamp.

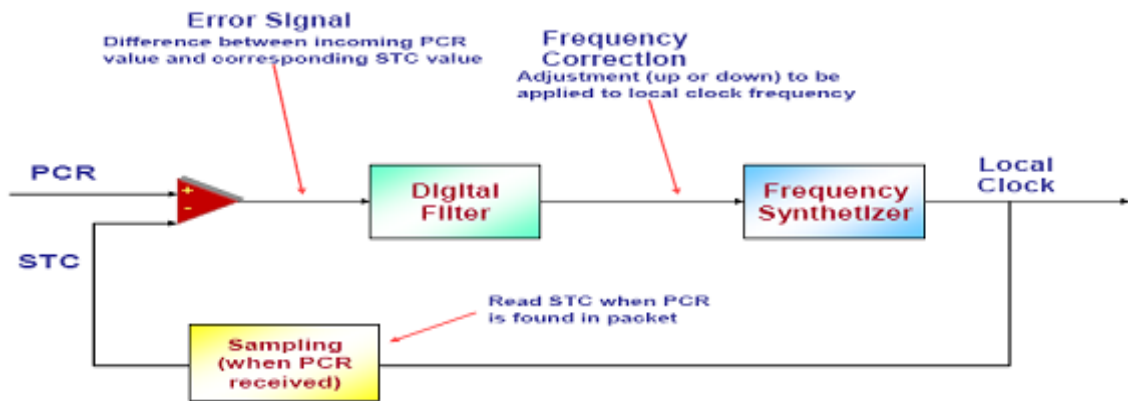


Figure 5.5 Generalized Design of PLL based Clock Recovery

The algorithm developed in this chapter for low jittery environment is based on PLL based design. The first step involves finding the frequency error between the head-end and the local-end by sampling the PCR and STC by the SD (Standard Definition) clock. The tick error is calculated by taking the difference of successive STC and PCR values.

$$\text{Tick error} = (S1 - S0) - (P1 - P0) \quad (1)$$

Where $S_0, S_1...$ denotes successive STC values and $P_0, P_1...$ denotes successive PCR values. This tick error is then converted in to frequency error using

$$\text{Frequency Error} = [27000000 / (P1 - P0)] * \text{Tick Error} \quad (2)$$

For two successive PCR samples, error is being calculated and fed to our newly designed Moving Window Basic Averaging filter. This section is divided into two parts. In first part we will explain the three most important parameters of this algorithm and later, how these parameters are being used in the algorithm.

5.3.1 Filter Parameters

Here we will discuss three main parameters as per below-

5.3.1.1 PCR_DRIFT_THRESHOLD (P)

It defines the maximum number of ticks in an error sample beyond which the corresponding frequency error sample will be ignored. This maximum limit depends upon the transmission jitter in the environment. If maximum frequency error is $5.4\text{KHz} + 810\text{Hz} = 6210\text{ Hz}$. Assuming a PCR rate of 40 ms, maximum tick error should be $6210/25 \sim 250$. And all error samples above 250 ticks will be ignored. Thus PCR_DRIFT_THRESHOLD characterizes a Low Pass Filter in which all the frequency error samples above a particular frequency defined by PCR_DRIFT_THRESHOLD are ignored. In applications like change of a channel in TV and 33-bit counter wraparound case, a very large jitter error may occur in the system. For example, taking 33 bit counter wraparound, from eq.(1) difference between two consecutive PCR values will be too large. (max. 33 bit value - ~minimum 33 bit value) resulting in a very large frequency error. However, actually it is not an error and the problem occurs because of wraparound. Similarly, when we change a TV channel, PCR data may change entirely since timing information of two different channels is different. There may be situation where we are subtracting the last PCR value of the last channel with the first PCR value of the current channel again results in a very large error sample. We need to ignore such spurious error samples. After passing the frequency error samples through a low pass filter, error samples are accumulated until a pre-defined number of samples are reached as explained further.

5.3.1.2 MIN_SAMPLE_NUM (S)

MIN_SAMPLE_NUM is the minimum number of error samples after which frequency corrections can be applied. MIN_SAMPLE_NUM is just to fasten the clock recovery so that as soon as we have some minimum number of error samples defined by

MIN_SAMPLE_NUM , we will apply a correction to the clock. The value should be taken in such a manner so that correction is neither applied very late nor too early because we need to take an average of error samples to cancel out the jitter error. In satellite environment we can start applying correction very early since there is less or no jitter and even by averaging very few error samples we can calculate the actual error. So MIN_SAMPLE_NUM is set to a small value of 10 or 20.

5.3.1.3 MAX_WINDOW_SIZE (M).

MAX_WINDOW_SIZE is the maximum number of error samples to be considered in a window to calculate the actual error and the corresponding correction value. After receiving this maximum number of error samples and on the arrival of next PCR value, we will remove the oldest sample from the sample window and include the latest PCR sample in its place. Thus in a way, our error sample window will start moving. Thus there will always be MAX_WINDOW_SIZE number of error samples in the window. In case of satellite this can be set to a small value(100 or 150) as there is very less jitter and error in the decoder clock can be simply corrected by accumulating the error of a very few number of samples.

5.3.2 Error Calculation and Applying Correction

First of all, the error samples are passed to low pass filter. All error samples beyond PCR_DRIFT_THRESHOLD are ignored. Now from second sample onwards error is being calculated for every two successive PCR samples. But correction is applied only when MIN_SAMPLE_NUM of samples have been received. To calculate the correction to be applied, an average of all error samples (less than or equal to MAX_WINDOW_SIZE) need to be taken. To keep window size fixed most recent sample is included in window and oldest sample of window is being excluded from

window as shown in Figure 5.6. This is how a window has moved hence named **Moving Window and Averaging Filter**.

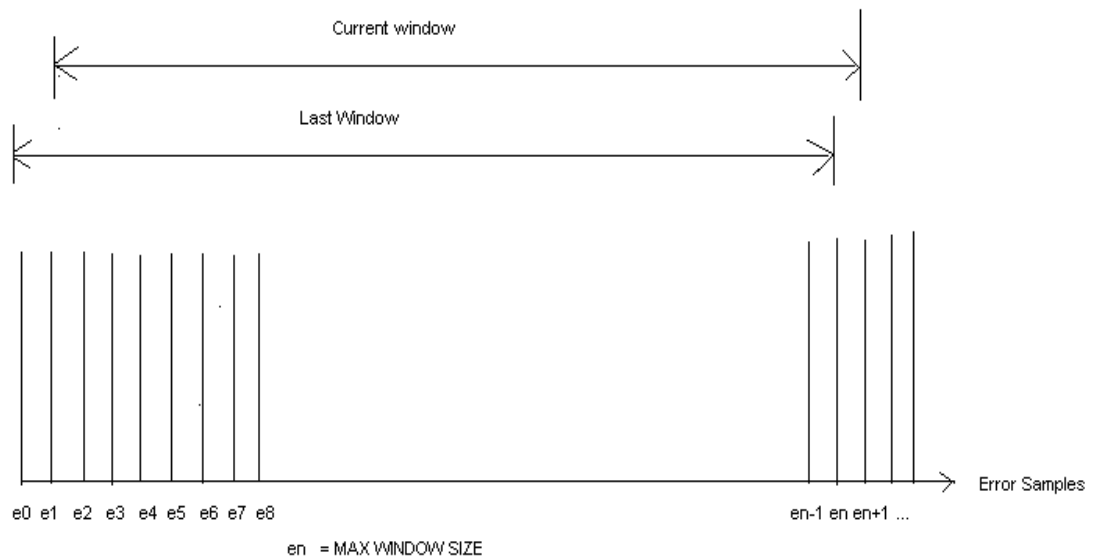


Figure 5.6 Concept of Moving Window

After calculating error, correction need to be applied in an appropriate manner. Here one important thing to consider is the settling time of the decoder clock. If we take an average of a very large number of error samples, the error calculated will be more accurate but it will take a large time for the decoder clock to settle to the encoder clock value. So larger the value of `MAX_WINDOW_SIZE` larger is the settling time but more is the accuracy. Similarly a small window size is also inappropriate as it can destabilize the sum of whole window and STC clock will not be stable. In a small window, if we receive a very large error sample it may happen that the negative counterpart of this error sample is not included in the window because of its small size. So after averaging, jitter error will not reduce to zero resulting in a correction being applied to the frequency when it is not required. Now, when we receive the counterpart of such an error sample later on, it will shift the sum of the window hugely in the opposite direction resulting in another

inaccurate correction being applied to the frequency. Thus in a nutshell, we will have an unstable frequency at the decoder. Keeping in view above facts, we will be calculating the actual error by applying Basic Average filter, and then a correction is applied to the running STC clock. But the correction corresponding to the actual error passed by the filter in a single step can make the clock recovery unstable. If PWM correction is applied too fast, clock recovery became unstable because of the averaging of current error sample with previous errors, for example, if there is a initial error of 2000 Hz then

$$\text{Average_error} = (2000+2000+2000+0)/4$$

On the third sample, Average_error of 2000 Hz is corrected in one go then fourth error sample (0 Hz) is arrived. At the arrival of fourth sample there is no error in the system, but due to the averaging of all error samples, error now comes out to be 1500 Hz. Because of this fact clock recovery applied further correction in opposite direction and filter ends up in toggling between low PWM mark and the high PWM mark. Solution to this problem was to apply PWM correction gradually so that toggling of the filter can be really slowed down. An Average_error should not be corrected in one step, it should be applied gradually. Average_error is divided by a multiple number of samples (Store_count) in the window and a constant multiple called GCF(GRADUAL_CORRECTION_FACTOR), which gives Average_error Per Sample (AEPS).

$$\text{AEPS} = \text{Average_error} / (\text{GCF} * \text{Store_count})$$

Above Average_error per sample gives the correction to be applied to the local decoder clock as –

$$\text{New_freq} = \text{Old_freq} + \text{AEPS}$$

For satellite environment, normally GCF has been chosen as 50. Frequency change effected by varying the Mark/Space ratio is linear only over a sub-range of available PWM Mark values, from 40 to 216. By one PWM Mark change, approximately 61 ticks per second change is observed in frequency.

Correction (in terms of PWM mark value)

$$= (\text{AEPS} + \text{Outstanding error}) / \text{PWM_TICK_MARK} ,$$

where PWM_TICK_MARK= 61

AEPS could be too small to be corrected for one sample so it is carried forward for next PWM correction. This carry forward error is called ‘Outstanding Error’. AEPS comes out to be a small value and Outstanding Error increases with every PCR received. Whenever the sum of AEPS and Out Standing Error is higher, than one step of VCXO (appx 61 ticks per second), correction is made and outstanding error is adjusted. After applying this Correction, the filter behaved in a stable manner. In jitter less environment it is locked to correct frequency and it also takes care of any residual error which is too less to correct (means less than one PWM step). In the simulation on the Satellite, we can see that the filter is stabilized to +/- 1 PWM step around the lock frequency and then toggles in such a way so that the long term average approaches encoder frequency.

5.4 PERFORMANCE OF NEW DESIGNED ALGORITHM

The algorithm was simulated on MATLAB for testing the response to the satellite environments. Appendix- II provides the necessary information about MATLAB code. The PCR data is extracted from live satellite streams and corresponding STC data is captured from the decoder clock. The plot of decoder frequency change with PCR count after running the simulation for satellite environment is shown in Figure 5.7.

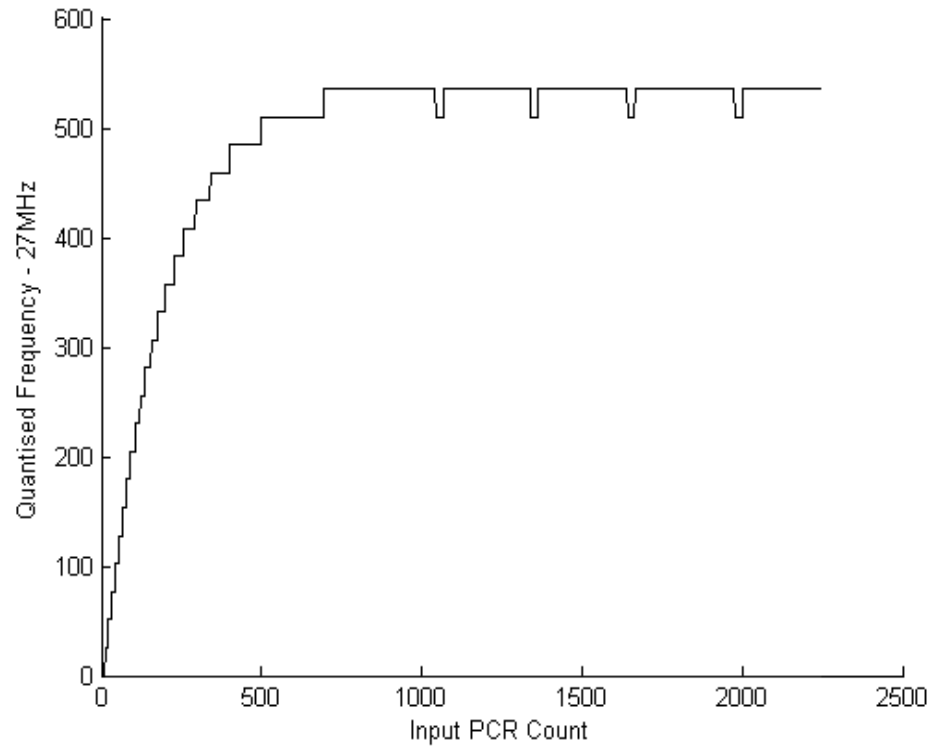


Figure 5.7 Behaviour of algorithm as simulated in Matalab for Satellite data with $P=300$, $S=10$, $M = 50$

Test results using real time set-top box in satellite environment and is shown in Figure 5.8. In satellite transmission, error in the encoder side clock is tracked very accurately. Assuming PCR arrival rate of 40 ms the decoder side clock is synchronized with the encoder side clock in $40 * 500 = 20$ seconds. This has been achieved with parameters $MIN_SAMPLE_NUM = 10$, $MAX_WINDOW_SIZE = 50$ and $PCR_DRIFT_THRESHOLD = 300$.

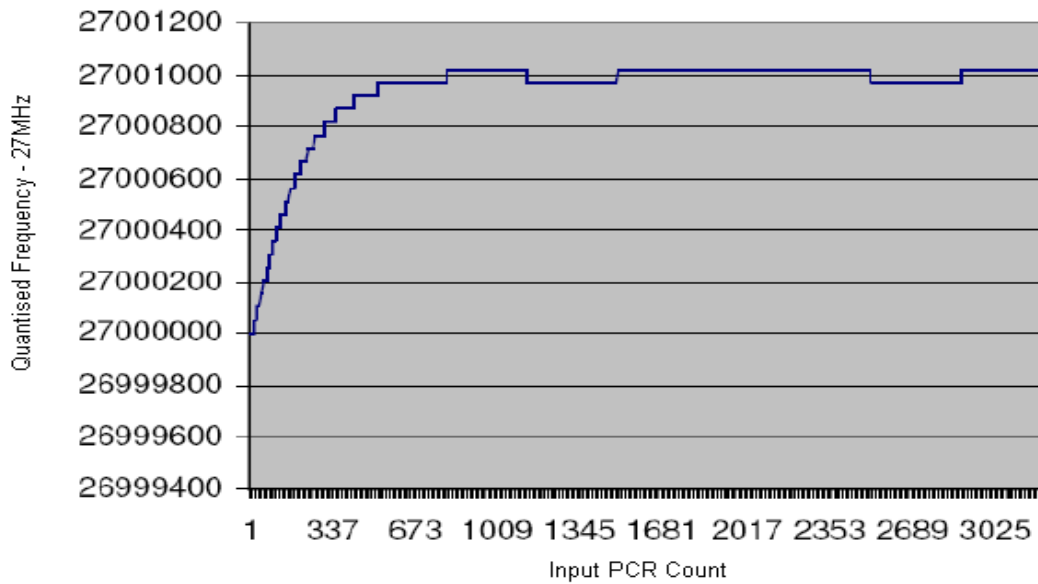


Figure 5.8 Input PCR Count vs Decoder Frequency graph demonstrate behaviour of Proposed Algorithm in Satellite Environment with $P=300$, $S=10$, $M=50$

5.5 CONCLUSION

Thus our designed algorithm shows good response as it is meeting all constraints of real time STB and achieving clock recovery efficiently. A light weighted, simple, low cost and stable software approach to recover the encoder clock in a satellite transmission environment has been proposed in this chapter.

Chapter 6: Clock Recovery for Terrestrial TV

Digital Video Broadcasting (DVB) transmission over terrestrial medium introduces high jitter in the stream (max jitter upto the order of 1ms). In DVB-T, loss of synchronization between the received and the transmitted clocks can lead to degradation in system performance like color loss, jerky video, audio drop outs etc. There are several factors, which lead to degradation in clock recovery process. In this chapter, analysis of various factors that constitutes jitter in receiver clock during acquisition of PCR packets in terrestrial environment has been done. It has been tested and simulated with already existing algorithms, which are not able to handle jittery environment of terrestrial broadcasting. Limitations of Moving Window Basic Averaging algorithm for terrestrial environment have been removed by modifying the algorithm. A new Moving Window Weighting filter algorithm, which offers low computational complexity, low cost and stable clock recovery has been presented for terrestrial transmission in set-top boxes.

6.1 ANALYSIS OF JITTER IN DVB-TERRESTRIAL

Jitter is a term that refers to the variance in the arrival rate of packets from the same data flow, and abnormal jitter values can negatively impact real time applications (Davis, 2008). For terrestrial environment hardware setup shown in Figure 6.1 has been used. It has been observed in case of terrestrial transmission that sometimes the STC does not follow the PCR properly i.e. jitter exist.

Jitter may arise due to the sources namely-

- There may exist a frequency drift between the transmitter and the receiver clock.

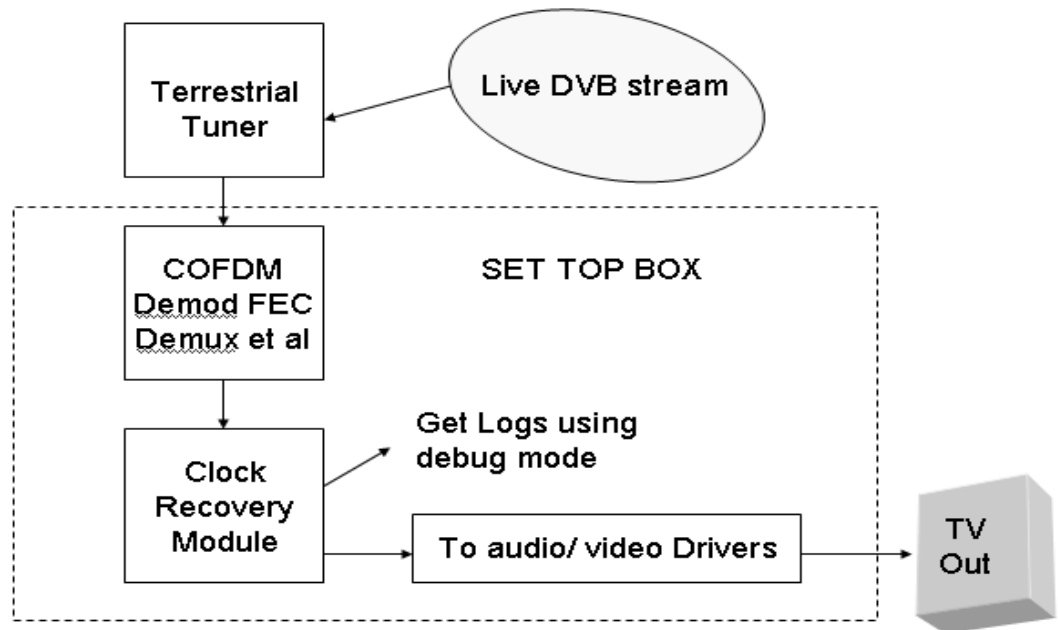


Figure 6.1 Hardware Setup

- The communication channel introduces a significant amount of jitter called the network jitter. In terrestrial environment, the maximum network jitter of 500 μ s is being analyzed. The high jitter is due to multi-path reflections of transmitted waves by building, trees etc. before it is finally received.
- At the set-top box front-end, more jitter is introduced because of the de-interleaving and other variable delays before providing data to the audio/video drivers. The inner interleaving and de-interleaving blocks in the transmitter and receiver side, particularly in Digital Video Broadcast-terrestrial (DVB-T) transmission, introduce a significant amount of jitter (ETS 300744, 1999).
- Packetization at source may result in displacing the timestamp values in the stream. This is called packetization jitter.
- In worst case channel conditions (such as $\frac{1}{4}$ guard interval, channel fading, low carrier-to-noise ratio etc.), arrival of Program Clock Reference (PCR) packets

being delayed as the STB tuner front-end takes too much time to detect and demodulate the signal.

High amount of jitter causes color loss and audio/video data loss. Data loss may also occur if the decoder clock is slower than the encoder clock. In this case, the data input to the display buffer memory is more than the output, which would lead to accumulation of the data in it, eventually causing buffer overflow. Similarly, if the decoder clock is faster than the encoder clock and eventually the buffer empties, the result is no more data to display. In both the cases, improper clock synchronization results in bad audio with glitches, no video display, and some times video color loss.

6.2 LIMITATIONS FOR DVB TERRESTRIAL BROADCASTING

Lot many terrestrial streams have been analyzed to identify the limitations in terrestrial streams. This analysis has helped us to propose a suitable algorithm for clock recovery for terrestrial set-top box.

Analysis-1: In Figure 6.2, PCR sample arrival time for terrestrial conditions is shown. As per DVB standards, PCR insertion-rate is 40ms, however, maximum allowable PCR time stamping is 100 ms. This time interval varies a lot in the terrestrial channels. It has been observed that a few PCR samples in some terrestrial streams were stamped even at duration of less than 5 ms as shown in the inset of Figure 6.2. **This varying rate needs to be considered for designing the algorithm to prevent any false correction in frequency of the decoder clock.**

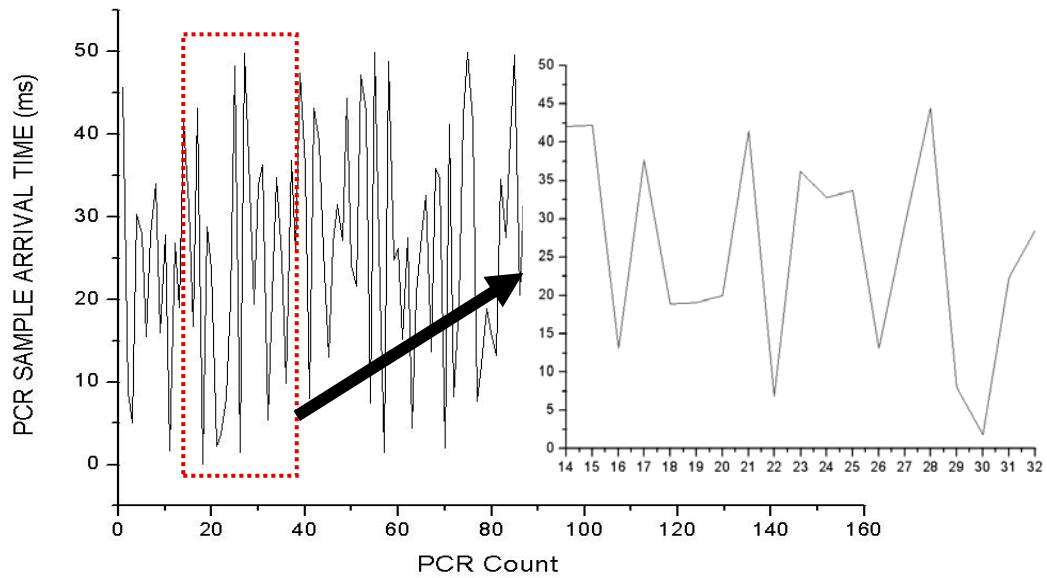


Figure 6.2 PCR samples Arrival time in Terrestrial Environment

Analysis-2: The frequency error with respect to PCR count is being shown in Figure 6.3. In a typical terrestrial environment these error samples complement each other and all the error samples may be averaged. By averaging, the positive and negative error cancel out and only the actual drift between the decoder and encoder clock remains. As shown in Figure 6.3 inset, the complement of error samples is **not necessarily in its**

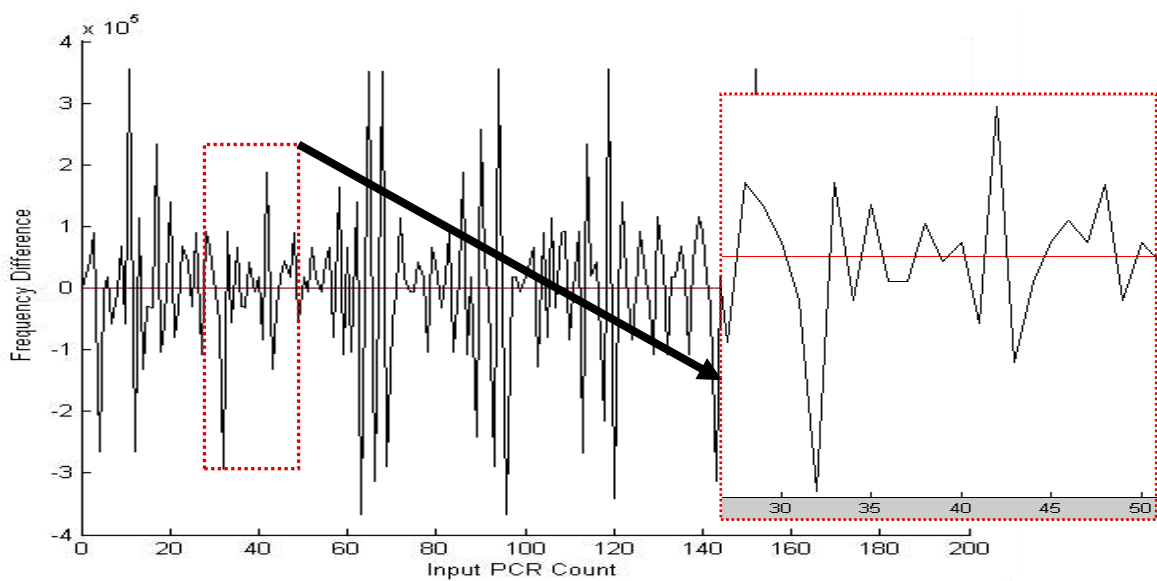


Figure 6.3 Frequency Error Samples in Terrestrial Environment

vicinity but may arrive after some time (a large spike in the downward direction is complemented after about 10 samples), so **we need to consider some minimum previous values while averaging and find out the actual correction in the decoder clock**. Above analysis are the starting point of developing new algorithm suitable for terrestrial environment. The clock frequency on the decoder side in the final stable state should be at 27MHz with a tolerance of 30ppm. It is observed that if the jitter error is not effectively removed, this may result in a faulty correction applied to the decoder clock. This correction can cause the decoder clock to operate above the tolerance limits, which may cause color loss in the television sets.

6.3 MOVING WINDOW WEIGHTING FIR FILTER ALGORITHM FOR DVB-T BROADCASTING

When a new PCR value arrives at the decoder, the previous value is subtracted from the present one to get PCR_Diff. The STC timestamp is similarly processed to get STC_Diff. Neglecting any jitter for the moment, the error difference, called the Tick_Diff, is then calculated as in (1)

$$Tick_Diff = (PCR_diff) - (STC_Diff) \quad (1)$$

The true error estimate is the error per cycle, or percentage error is found out-

$$Percentage_err = \frac{Tick_Diff}{PCR_Diff} \quad (2)$$

The frequency error is then calculated as in (3)

$$Freq_err = Percentage_err \times Encoder_frequency \quad (3)$$

The Freq_err is calculated using the assumption that there is no jitter in the stream and the encoder is transmitting at the frequency exact 27MHz. Hence, Encoder_frequency = 27MHz. This error samples should ideally be zero if there is no jitter and the decoder and encoder clocks are at same frequency. The non-zero frequency error samples are then sent to a Low Pass Filter for rejecting the samples, which arrive too fast.

6.3.1 LOW PASS FILTER

The error samples need to be checked to be valid or invalid. This is done using a simple low pass filter to reject all the values above a parameter termed as PCR_DRIFT_THRESHOLD (P). All the error samples beyond P are ignored. There are two cases when error sample can be very large. PCR samples are further checked if its arrival time is lesser than 20 ms to reject all the samples coming too fast. The PCR and STC data arriving before 20 ms is ignored otherwise it would have caused a false correction on the decoder clock. In this manner, bad PCR and STC data is handled effectively. This valid data is stored into a Moving window buffer for average calculations.

6.3.2 MOVING WINDOW WEIGHTING FIR FILTER ALGORITHM

The sample values are stored into a window of size defined by a parameter MAX_WINDOW_SIZE (M). This parameter is of principal importance as it defines the amount of previous samples to be used for efficient averaging. It also determines the settling time for this algorithm. If the filter is applied over a large error sample size, the error calculated is more accurate but it takes a larger time for clock synchronization.

However, a small window size is inappropriate as a single large sample can destabilize the sum of whole window and STC clock will not be stable. So, optimized value of M is found out for different jitter situations.

Another consideration while designing is that the error is being calculated from second sample onwards but correction is not applied at the same time. We have designed our algorithm in such a manner that correction in the frequency of the decoder clock should be applied only when minimum numbers of samples are collected, defined by the parameter MIN_SAMPLE_NUM (S). This approach is adopted to prevent the initial spikes due to average over a small sample set.

The strategy and modification over here as compared to our earlier developed algorithm as discussed in chapter-5, for calculation of average error is to assign a **weight** to each of the frequency error samples depending on some parameters. The Finite Impulse Response (FIR) filter is made using the taps equal to the size of the window. The weights are proportional to the position of the element in the moving window in a pyramidal manner (the central tap being the highest and the edge taps being lowest). The output of the filter is the average error calculated as in (4)

$$Average_err = \frac{\sum_{MAX_WINDOW_SIZE} Freq_error * weigh}{\sum_{MAX_WINDOW_SIZE} weigh} \quad (4)$$

A triangular filter is chosen compared to the exponential filters or any other filters used by researchers in the similar field. The reason for this is to develop a simpler filter to be implemented in an RTOS environment. The maximum weight in the centre of the window is designed under the assumption that the compliment of error samples in the

centre exist in the window, however the samples at the edges may not have their compliments in it.

6.3.3 APPLYING CORRECTION

After calculating the Average_error by maintaining the moving window and applying the FIR filter, this correction should be applied to the decoder clock. But this correction should not be applied in a single step as it can force the decoder clock outside tolerance limits. It must be applied in a gradual manner by damping this correction value. If no damping is done, the frequency change may be so much that this sharp change in frequency may not be sensed by the Television monitor's PLL(Phased Lock Loop) and may cause colour shift of the video signal. To achieve above, Average_error was divided by the number of samples in the window (Store_count) and a constant damping factor called Gradual Correction Factor (GCF).

This gives Average Error Per Sample (AEPS) as shown in equation (5)

$$AEPS = \frac{Average_err}{GCF \times Store_count} \quad (5)$$

The value of the damping factor was chosen such as to optimize the settling time with respect to the stability of the decoder. This average error per sample gives the correction to be applied to the local decoder clock as in (6)

$$New_freq = Old_freq + AEPS \quad (6)$$

The parameters for the moving window were found out by analyzing the amount of jitter present in the stream. The STC difference between consecutive STC samples was found out after arrival of two consecutive PCRs. The standard deviation of the STC differences

gives the approximate jitter. Extensive simulations were conducted to find out the value of the filter parameters M, S and P in jitter conditions varying from 0.1 ms to 1.2ms. A lookup table was made regarding the parameters to be adapted for the Clock Recovery module according to the amount of jitter in the stream. According to the jitter in the stream, the algorithm tested in the set top box auto-adapts the parameters required.

6.4 CLOSED LOOP ANALYSIS OF PROPOSED ALGORITHM

In this section, the main clock recovery feedback loop is described as a filter that transforms the input PCR and STC time stamps into an output frequency correction value. The filter transfer function will be expressed as a Z transforms. Z transform analysis assumes that the input samples are periodic. In clock recovery, the error difference samples E_n are not periodic. However, if the error difference is divided by the PCR difference P_n to obtain a percentage error measurement, and assuming small PCR timing error at the encoder, we can obtain samples with close approximation of periodicity. Therefore, for rest of this analysis input samples will be assumed as periodic data.

Let us consider an absolute time interval T, between two successive PCR time stamps (P_n and P_{n-1}). The head-end clock at the encoder, running at frequency H, will measure this as-

$$(P_n - P_{n-1}) = H \cdot T = L \quad (7)$$

A similar equation relates the STC time stamps at the decoder -

$$(S_n - S_{n-1}) = F_n \cdot (T + j_n) \quad (8)$$

where \mathbf{F}_n is the composite decoder frequency, and \mathbf{j}_n is the jitter due to the variability in the delivery time of the PCR packet. \mathbf{F}_n has two components, the programmed output frequency of the FS, and crystal frequency driving the FS. These are related as follows -

$$\mathbf{F}_n = (\mathbf{H} + \mathbf{C}_{n-1}).(\mathbf{X} + \mathbf{X}_n) / \mathbf{X} \quad (9)$$

where the programmed output frequency of the FS is represented as the encoder frequency (\mathbf{H}) plus a previous programmed correction (\mathbf{C}_{n-1}), and the crystal frequency is represented as a nominal value (\mathbf{X}) plus the drift error (\mathbf{X}_n). We are trying to correct crystal drift error here. Assuming both (\mathbf{C}_n/\mathbf{H}) and ($\mathbf{X}_n / \mathbf{X}$) are small, the following approximation can be made -

$$\mathbf{F}_n \cong \mathbf{H} + \mathbf{C}_{n-1} + \mathbf{H} \cdot \mathbf{X}_n / \mathbf{X} \quad (10)$$

The difference error (\mathbf{E}_n) is evaluated as -

$$\mathbf{E}_n = (\mathbf{P}_n - \mathbf{P}_{n-1}) - (\mathbf{S}_n - \mathbf{S}_{n-1}) \quad (11)$$

Substituting values from (7) and (8) into (11) gives -

$$\mathbf{E}_n = \mathbf{H} \cdot \mathbf{T} - \mathbf{F}_n \cdot (\mathbf{T} + \mathbf{j}_n) = (\mathbf{H} - \mathbf{F}_n) \cdot \mathbf{T} - \mathbf{F}_n \cdot \mathbf{j}_n \quad (12)$$

Substituting for the first \mathbf{F}_n from (10) gives -

$$\mathbf{E}_n = (\mathbf{H} - (\mathbf{H} + \mathbf{C}_{n-1} + \mathbf{H} \cdot \mathbf{X}_n / \mathbf{X})) \cdot \mathbf{T} - \mathbf{F}_n \cdot \mathbf{j}_n = -(\mathbf{C}_{n-1} \cdot \mathbf{T}) - (\mathbf{H} \cdot \mathbf{T} \cdot \mathbf{X}_n / \mathbf{X}) - (\mathbf{F}_n \cdot \mathbf{j}_n) \quad (13)$$

This can be re-expressed as-

$$\mathbf{E}_n = -\text{Programmed Correction} - \text{DriftError} - \text{RandomError} \quad (14)$$

The random error is a noise term that averages to zero, so it can be eliminated from this part of the analysis. The system stabilizes when the programmed correction is equal to (minus) the drift error. If we express the drift as -

$$\mathbf{D}_n = \mathbf{H} \cdot \mathbf{X}_n / \mathbf{X} \quad (15)$$

Then neglecting the noise term, equation (13) becomes-

$$E_n = -(C_{n-1} \cdot T) - (D_n \cdot T) = -T(C_{n-1} + D_n)$$

(16) This equation is represented at the left hand side of the system diagram shown in Figure 6.4.

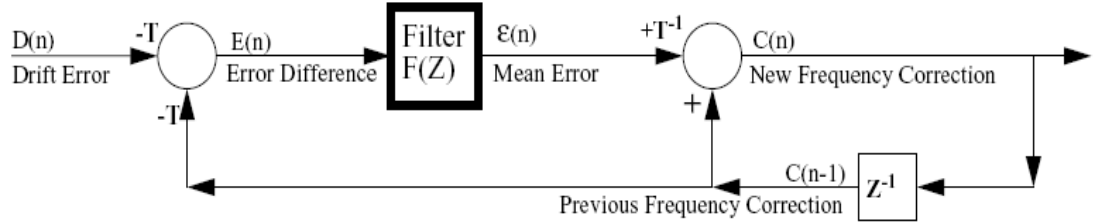


Figure 6.4 Feedback Loop expressed as Z transform

The Z-transform representation of the feedback loop is shown in above Figure 6.4.

The error difference is filtered to form the mean error ϵ_n . By dividing the mean error by the time period (T), we get a new increment to the frequency correction. This is added to the previous estimate of the frequency correction to form the new estimate of the frequency correction. When the system stabilizes -

$$C_{n-1} = -D_n, E_n = \epsilon_n = 0, C_n = C_{n-1},$$

For Figure 6.4, the following time series equations are given with their Z transform equivalents -

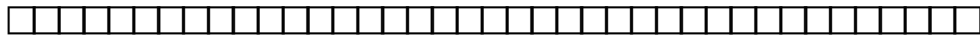
$$E_n = -(T \cdot D_n) - (T \cdot C_{n-1}) \Leftrightarrow E(Z) = -(T \cdot D(Z)) - (T \cdot Z^{-1} \cdot C(Z)) \quad (17)$$

$$\epsilon_n = F(E_n) \Leftrightarrow \epsilon(Z) = F(Z) \cdot E(Z) \quad (18)$$

$$C_n = \left(\frac{\epsilon_n}{T}\right) + C_{n-1} \Leftrightarrow C(Z) \Leftrightarrow \frac{\epsilon(Z)}{T} + Z^{-1}C(Z) \quad (19)$$

(17), (18), (19) can be used to eliminate E(Z) and $\epsilon(Z)$. The resulting equation can be re-arranged to the following form-

$$\frac{C(Z)}{D(Z)} = \frac{-F(Z)}{1 - Z^{-1} + Z^{-1} \cdot F(Z)} = \frac{-Z}{(Z-1) / F(Z) + 1} \quad (20)$$



The right hand side of (20) is the overall transfer function of the feedback loop.

Coefficients of Z^{-n} in the numerator represent coefficients of a moving average (FIR)

filter. Coefficients of Z^{-n} in the denominator represent coefficients of an auto-regressive

filter (IIR). The simplest system is when there is no in-loop filter. In this case $F(Z) = 1$,

and (20) reduces to
$$\frac{C(Z)}{D(Z)} = -1$$

i.e. frequency correction $C(Z)$ is set to minus drift error $D(Z)$. For example, if the local crystal is 1% fast at 27.27 MHz, the frequency synthesizer will be programmed to be 1% slow at 26.73 MHz.

The most simple in-loop filter averages the current input and the previous input. This is represented by

$$F(Z) = \frac{(1 + Z^{-1})}{2}$$

In this case (20) reduces to

$$\frac{C(Z)}{D(Z)} = \frac{-\left(\frac{1}{2} + \frac{1}{2} Z^{-1}\right)}{1 - \frac{1}{2} Z^{-1} + \frac{1}{2} Z^{-2}} \quad (21)$$

This equation has coefficients of Z^{-n} in both the numerator and the denominator.

Therefore the total transfer function of the feedback loop forms an ARMA filter. In

general, any attempt to use in-loop filtering will produce a total system filter with the

complexity of an ARMA filter.

6.5 PERFORMANCE OF PROPOSED ALGORITHM FOR TERRESTRIAL BROADCASTING

The algorithm was simulated on MATLAB for testing the response to the terrestrial environments. The PCR data is extracted from live terrestrial streams and corresponding STC data is captured from the decoder clock. This dataset is then applied to the input to the MATLAB script. Since the STC data is captured when the decoder clock is running constantly at 27 MHz, this data is not the same as the STC value if the decoder clock frequency is changed by the clock recovery module. The MATLAB script calculates the simulated actual STC and then applies the algorithm discussed above to find out the correction at the decoder end. The terrestrial plot using Guard Interval 1/8, 64 QAM, FFT 8K, FEC 3/4 (the parameters are explained later) is shown in Figure 6.5. The clock settles in $40 \times 2000 / 1000 = 80$ sec (for Jitter = 0.4 ms). The behavior of the algorithm is stable with the parameters of $P=10,000$, $S=50$ and $M=150$. Note that the variation in the frequency is well within the tolerance limits of 30ppm.

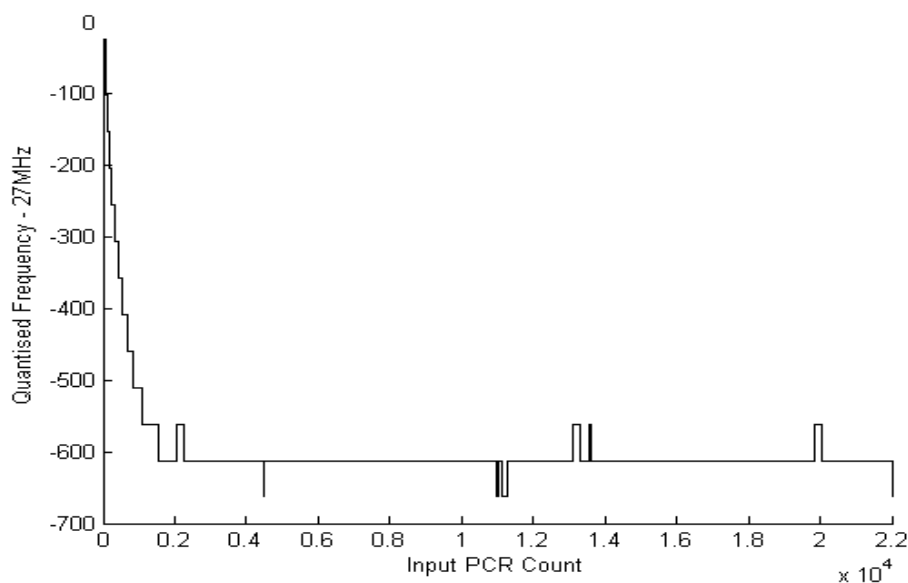



Figure 6.5 Behaviour of simulated algorithm in MATALAB for terrestrial data with $P=10000$, $S=50$, $M=150$.

The set-up for testing of the algorithm in terrestrial environment has already been shown in Figure 6.1. The live DVB stream is fed to DVB-T front-end which generates Transport Stream packets for the set top box drivers. The Clock Recovery module provides the timestamps to the video and audio drivers, which provide a decoded audio-video signal to view on television for testing of color loss or audio-video data loss. Using this test setup the amount of jitter is measured for various streams using different values of parameters like Guard Interval, FFT, FEC, and modulation technique is described in the Table 6.1.

Table 6.1 Effect of Terrestrial Parameters on Jitter

Guard Interval	FFT	FEC	Modulation
1/32	2K	1/2	16 QAM
1/16	2K	3/4	16 QAM
1/8	8K	3/4	64 QAM
1/4	8K	7/8	64 QAM


 Increasing
order of
jitter

The jitter varied from 0.1 ms to 1.2ms under these test conditions. The behaviour for 0.4 ms jitter, using the conditions Guard Interval (G.I.) = 1/8, FEC = 3/4 and modulation technique = 64 QAM is shown in Figure 6.6. The clock stabilises in about 80 sec. The variation in the clock frequency after stabilising is 1 step only and it shows good stability over a long time. The plot confirms with the MATLAB simulations too. The stability of the module follows from these results. The algorithm has adapted to M=150, S = 50 and P = 10000 under conditions with Guard interval = 1/8, Modulation=64-QAM, FEC=3/4.

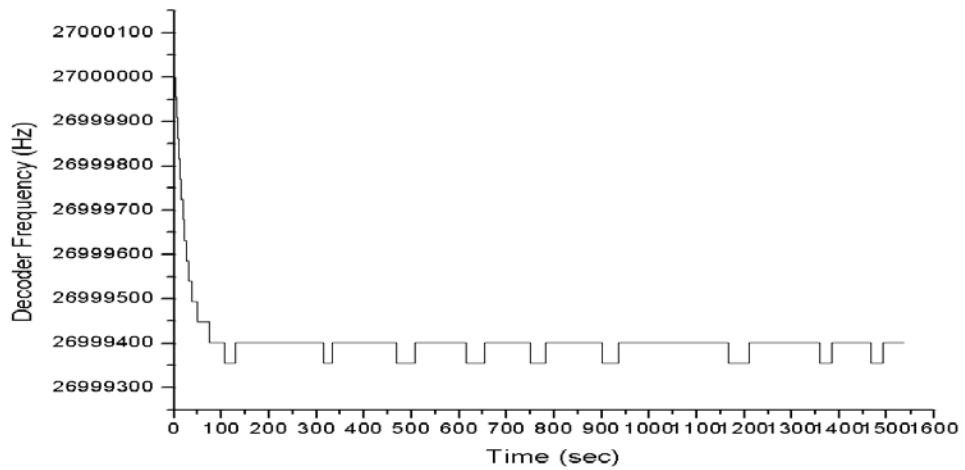


Figure 6.6 Behaviour under G.I. = 1/8 , 64 QAM, FEC=3/4 terrestrial environment(jitter = 0.4 ms)

The worst case jitter (1.2ms) was found using 1/4 Guard Interval, FEC 7/8, and 64QAM. The algorithm was tested with different values of Guard Interval and FEC. The behaviour of our module under this stressed jittery condition is shown in Figure 6.7. The clock settles in about 150 sec with reasonable stability. The jump in frequency of the module is only of 2 steps after settling. Furthermore, no data buffer overflow or underflow was observed and the television viewing went smoothly. It may seem that the settling time of the decoder clock is large, however, it is not so. If the correction to the decoder side is applied in a single step, the settling time may reduce, but the sharp frequency change in the decoder clock may cause colour loss. The Algorithm has adapted to M = 300, S = 100 and P= 30000 for the worst-case terrestrial conditions with Guard interval = 1/4, 64 -QAM, FEC=7/8.

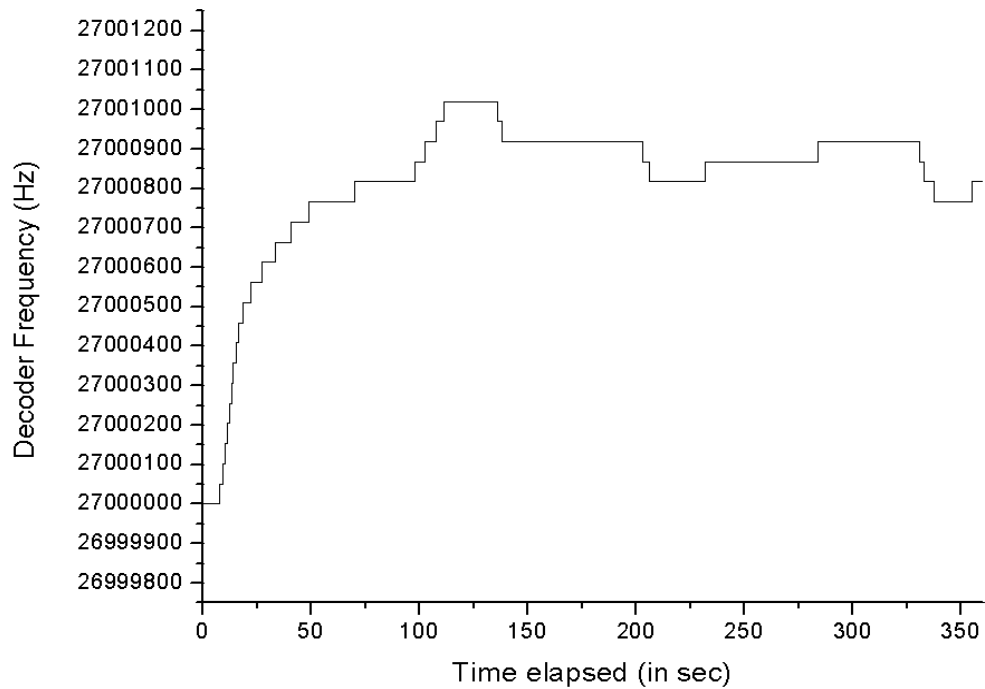


Figure 6.7 Behaviour under G.I. = 1/4 , 64 QAM, FEC= 7/8 terrestrial environment(jitter = 1.2 ms)

6.6 CONCLUSION

In this chapter a newly designed efficient, cost- effective, simple and smart algorithm named Moving Window Weighting Filter algorithm for terrestrial transmission for digital set-top box has been presented. The performance of new proposed algorithm has been simulated in MATLAB. Results are illustrated for real time environment in STBs. With our proposed algorithm, decoder clock is synchronized with encoder clock in 80 seconds for 0.4 ms jitter and for 1.2 ms jitter, decoder clock is synchronized with encoder clock in 150 seconds so proposed weighting algorithm shows cost effective and simple clock recovery without significant overshoots, in low as well as in high jitter environment.

Chapter 7: IP Streaming: Detailed Analysis in IP Environment

Proper clock synchronization is very critical for ensuring good performance of IPTV. Analysis and understanding of various component associated with IP environment is essential for developing clock recovery algorithms for IPTV. Fundamental concepts of IP networking technologies, TCP/IP protocol, various AV contents transport formats, IP protocol stack, and different IP streaming receiver software have been explored in this chapter. However, signaling and management protocols related to above have been presented in Appendix-III.

IP streaming is a generic term for transmitting digital audio and video contents over an IP (Internet Protocol) network. Inside the home, AV contents can be transferred from two consumer appliances (e.g. two set-top boxes) via a network connection, using IP streaming over an in-home network. Similarly, an Internet Service Provider (ISP) can provide digital AV contents, such as live TV programs or video on demand, to its customers via a broadband Internet connection, just like cable or satellite operators. Such a service is often called IPTV or even Telco-TV (because it is mostly offered by telecom operators). **IPTV (Internet Protocol Television)** is a system where a Digital Television service is delivered using Internet Protocol over a network infrastructure, which may include delivery by a broadband connection. A general definition of IPTV is that television content, instead of being delivered through traditional broadcast and cable formats, is received by the viewer through the technologies used for computer networks. The set-top box receiving and decoding AV contents over an IP network is called an IP set-top box or IP STB.

When AV content is streamed over an IP network rather than broadcasted from a satellite, terrestrial, or over cable, there are some additional constraints put on the receiving device. These constraints are:

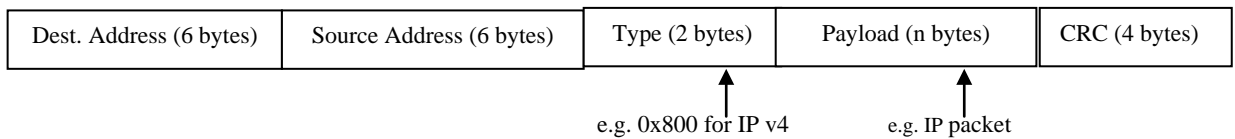
- **Format:** Many IPTV systems use a familiar MPEG Transport Stream, but some use entirely different formats.
- **Network jitter:** Due to the **asynchronous** nature of IP networks, network induced jitter values are much larger than what can be found on broadcast networks: several milliseconds and sometimes higher (up to 500 ms). The high jitter value requires extra buffering to compensate and also complicates the receiving device local clock synchronization with the sender's clock. In fact, clock recovery in IP network is really challenging and critical.
- **Lost packets:** IP packets can be discarded due to bit errors on a noisy transmission line (e.g. ADSL). No FEC scheme is available for the moment, although organization like DVB, are working on such a mechanism. IP packets generally contain more AV contents than MPEG TS packets, so the loss of an entire IP packet is more difficult to compensate for.
- **Out of sequence packets:** Although very unlikely on a home network or carefully engineered telecom providers networks, IP packets could arrive at destination in a different order than when they were sent out.

7.1 IP NETWORK INTERFACES

7.1.1 Networking Technologies

Ethernet Networks: IP packets can be transported over different kind of networks, encapsulations and formats. One common feature of all the different networks used for IP

Streaming, either in-home or from an outside provider, is the use of the familiar ubiquitous Ethernet frame format for transporting data.



The Ethernet frame format is used not only over regular wired Ethernet, but also over ADSL or cable modems broadband access networks, wireless LAN (IEEE802.11 Wi-Fi), power line networks (Home Plug), in-home coaxial cable (MoCA (Multimedia Over Coax alliance), Coaxsys, HPNA(Home Phone Network Alliance)), etc. Most IP streaming systems use the Ethernet frame format for transporting IP packets, regardless of the actual physical network. The “Destination Address” field will be either the IP STB unique Ethernet address (unicast frame) in the case of dedicated contents like Video on Demand (VOD), or an Ethernet multicast address (multicast frame) when streaming broadcast channels to the IP-STB. The “Type” field indicates the type of payload transported inside the Ethernet frame. A Type value of 2048 (0800 hexadecimal) indicates that the payload is an IP packet (also called IP datagram). The payload size is variable with a minimum of 60 bytes and a maximum of 1500 bytes (also called Maximum Transmission Unit or MTU). Typically for IP streaming, AV contents payload will be fragmented so that each fragment can be fit into a 1500-byte IP packet that can, in turn, fit into a single Ethernet frame. A 32-bit CRC is appended to the Ethernet frame to check for transmission errors.

7.2 IP PROTOCOLS SUITE

7.2.1 Unicast and Multicast

Each device connected to an IP network (also called IP host) is identified by a unique IP address. For internet protocol version 4 (IPv4), an IP address is 32 bits wide, often

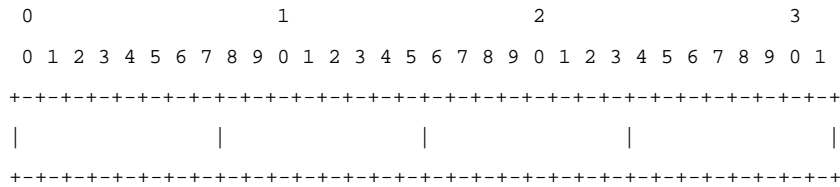
displayed using a so-called “byte decimal” notation, such as: 192.168.1.100, 127.0.0.1, 255.255.255.255, or 164.129.119.51, etc. when sending an IP packet (sometimes called IP datagram) to an IP host, the packet’s destination address must be set to the host unique IP address. Such a unique IP address is called a unicast address.

There is also a mechanism to send an IP packet to more than one host by using a “special” address called multicast address (or group address) as the destination address. The IP address range 224.0.0.0 to 239.255.255.255 is reserved to multicast addresses and cannot be assigned to an individual IP host device. An IP host can receive IP packets sent to a multicast (or group) address by setting its network interface and IP stack to accept packets sent to this group address.

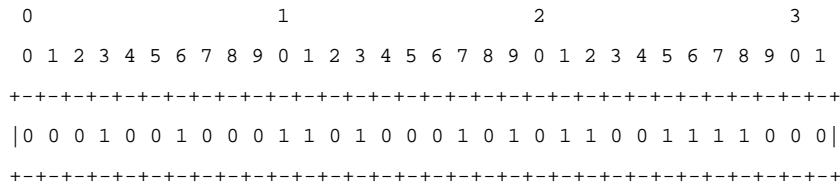
In video IP streaming systems, a “live” broadcast TV channel will typically be sent to a multicast address, where several receiver devices will be able to receive the same content. On the other hand, a Video on Demand (VOD) server will send content to the unicast IP address of the intended destination device.

7.2.2 IP Protocols Byte and Bit Representation

By convention, multi-byte data on IP networks are transported with the most significant byte first (big endian), as per IP protocol described (RFC 791). All the standards (RFC) in the IP protocol suite use the same data representation, grouped by 32-bit words, with the most significant bit having the number 0 always on the left side and the least significant bit being the bit 31 on the right side:



For instance, the 32-bit value x12345678 is represented as follows:



When stored in memory, after reception from a network interface, the same data will be stored as follows (often called “network byte order”):

0x12	Address N
0x34	Address N+1
0x56	Address N+2
0x78	Address N+3

So, when parsing IP protocol data on a small endian CPU, all 16-bit and 32-bit words must be converted from network byte order to host byte order, using macros that are provided with the protocol stack. Else, without this “network to host” conversion, the above example would be erroneously interpreted as 0x78563412 instead of 0x12345678. Note that on Ethernet networks (Ethernet IEEE 802.3), each byte is sent with the least significant bit first. So, in the above 32-bit word example, the first bit transmitted will be bit 7, followed by bit 6, and so on with bit 24 (most significant bit of the last byte) being transmitted last.

7.2.3 Transport Protocols

IP packets carry data using a transport layer. The two basic transport protocols used in IP networks are the Transmission Control Protocol (TCP) and the Universal

Datagram Protocol (UDP) (TCP/IP, 2001). There are some additional signaling and management transport protocols used in IP streaming like DHCP (Dynamic Host Configuration Protocol), IGMP (Internet Group Management Protocol) and RTSP (Real Time Streaming Protocol), but all these additional protocols run either on top of either TCP or UDP transport services.

TCP: Transmission Control Protocol: TCP is mostly used for reliable file transfers. It establishes a connection between two end points and uses an acknowledge mechanism for each packet. If a packet is not received by the destination host, it will be retransmitted by the sender (guaranteed delivery). TCP protocol also includes other functions such as flow control, packet re-ordering, etc. Because TCP will attempt retransmission of all lost packets, it is not necessarily well suited to “real-time” AV contents where it is more important to ensure a timely delivery of the AV payload with minimal delay, even at the cost of having to conceal eventual errors due to missing packets. TCP is, by definition, a “one-to-one” transmission protocol; so it can be used only for unicast data transfer and cannot be used to send data to a multicast address. So, it is not suitable to “live” broadcast TV channel.

TCP also provides an additional addressing mechanism within each host called port numbers. In addition to the host being identified by a unique IP address, each TCP port number (from 0 to 65535) identifies a unique end point inside the host. This mechanism allows for several simultaneous TCP transfers inside the same host, using different TCP destination port numbers to deliver each packet to the right target end point. A TCP packet (called “TCP segment”) is sent from a unicast IP source address and a source port number to a destination port number at a unicast IP address. TCP is described in RFC 793.

UDP: Universal Datagram Protocol: UDP is a “fire-and-forget” transport protocol. There is no acknowledging mechanism between source and destination. Lost packets must be handled by the upper level application that is using UDP transport. UDP uses source and destination port numbers (from 0 to 65535) to identify more than one end point inside a host. Unlike TCP, a UDP packet (called “UDP datagram”) can be sent to a multicast address (more than one destination host) as well a unicast address (unique destination host). For that reason live broadcast TV channels will often be streamed over IP networks using UDP based transport protocols.

Since UDP does not guarantee packet delivery at the destination end point, applications using UDP transport will have to implement adequate mechanisms to handle lost packets. When streaming AV contents over an IP network, it is generally preferable to implement error concealment mechanisms to handle lost or missing packets rather than attempting a lost packet retransmission. UDP is described in RFC 768.

RTP: Real-Time Transport Protocol: RTP is a transport protocol that generally uses UDP transport: RTP packets are carried into UDP datagrams (TCP/IP, 2001). The UDP destination port must always be an even number. RTP has been designed for the transport of “real-time” content: voice (Instant Messaging or telephony over IP), video, etc.

IP header (20 bytes)	UDP header (8 bytes)	RTP header (12 +n bytes)	AV payload (n bytes)
----------------------	----------------------	--------------------------	----------------------

The RTP header size is generally 12 bytes, plus optional extensions. RTP headers provide following features that are well suited to the transport of AV payload:

- Payload type identifier: MPEG-2, H.263
- Sequence numbers: to identify lost or out-of-order packets
- Time stamp: to allow synchronization between sender and receiver

- Source identifier: a 32-bit number mostly used for multi-party audio/video conferences.
- Marker bit (M): generally used to mark a frame boundary.

RTP is described in RFC 3550.

HTTP: Hyper Text Transport Protocol: HTTP is the transport protocol used by a web browser to download a web page from a web server (or to upload user's data). It is using TCP as underlying transport protocol. Since IP STB and other networked consumer appliances are by definition connected to an IP network, using a web browser for the user interface is an obvious solution. All web browsers include an HTTP client that will send commands (called methods), over TCP segments, to an HTTP server. Main HTTP methods include:

- GET: to download a web page (and binary files eventually) from a web server.
- POST: to upload data (such as a form filled by the user) to the server

HTTP can also be used as a transport protocol for AV contents for example: DLNA (Digital Living Network Alliance) (DLNA, 2004). HTTP version 1.1 is described in RFC 2616.

Thus in above section various transport protocols have been described. In addition to these AV transport protocols, IP streaming systems need signaling and management protocols to set up and monitor the AV payload transport. Those are explored in appendix-III.

7.3 PROTOCOL STACKS

7.3.1 Stack Representation

As explained in the previous sections, IP networks require the use of several protocols for transmitting data from end to end. These protocols are logically organized as different layers piled on top of each other in a stack (hence the name “protocol stack”), where each protocol layer uses the services provided by the layer underneath and, in turn, provides its services to the layer above. For instance:

- RTP uses UDP services
- UDP uses IP services
- IP uses Ethernet services

At the “bottom” of the stack are the network interfaces (like Ethernet) while the application software is shown at the top of the stack:

7.3.2 Data Encapsulation

When observing data circulating on an IP network, the data structure reflects the different protocol layers by wrapping each layer data inside another data structure, in a ‘Russian doll’ fashion. For instance:

- A web page (HTML file) is encapsulated into an HTTP method
- An HTTP method is encapsulated into TCP segments
- A TCP segment is encapsulated into an IP datagrams
- An IP datagram is encapsulated into an Ethernet frame

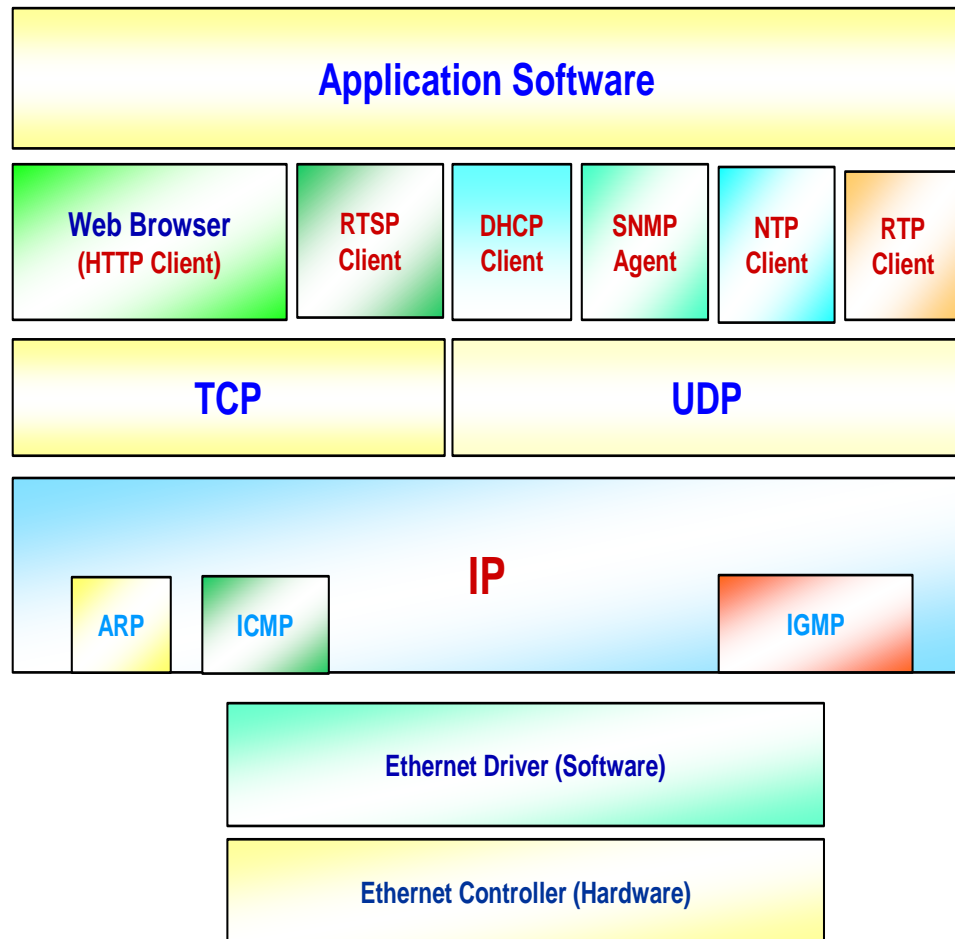


Figure 7.1 IP Network Stack Representation

7.3.3 Protocol Stack Implementation

Most operating systems today include a TCP/IP protocol stack (Linux, Windows, VxWorks,...). All these protocol stacks implement similar functionalities (IP, TCP and UDP protocols) and provide similar interfaces to the other software components. A protocol stack provides a standard interface to attach one or more network interface drivers, such as an Ethernet driver. The exact interface details vary from one OS to the other, but there are generally functions to set the network interface up or down, modify some settings, send data and receive data from the network.

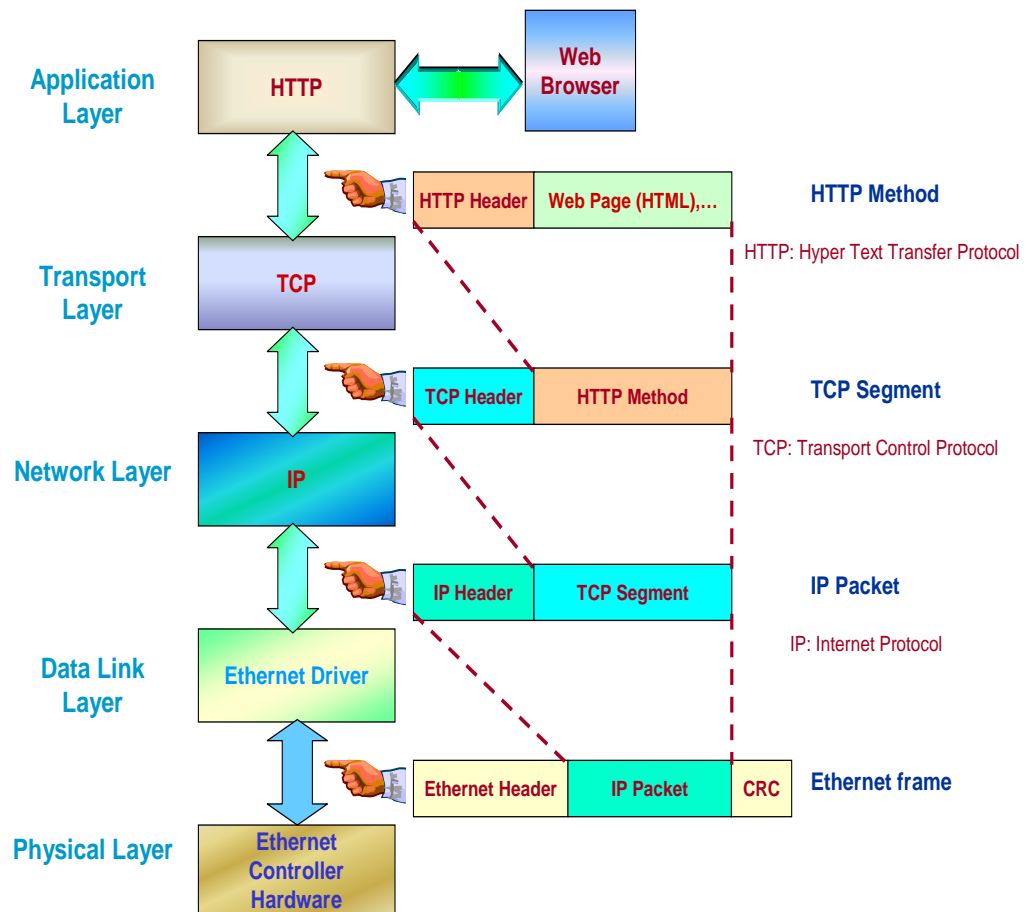


Figure 7.2 IP Data Encapsulation

At the “top”, most TCP/IP stacks provide a standard interface called socket. Socket implementations are often derived from the original Unix sockets and are very similar from one OS to another. The two most widely used types of sockets are:

- Stream sockets that use TCP transport
- Datagram sockets that use UDP transport

All the TCP/IP stack components, up to the socket interface, as well as the network interface driver are generally running in kernel mode. Application software using socket interfaces are running in user mode.

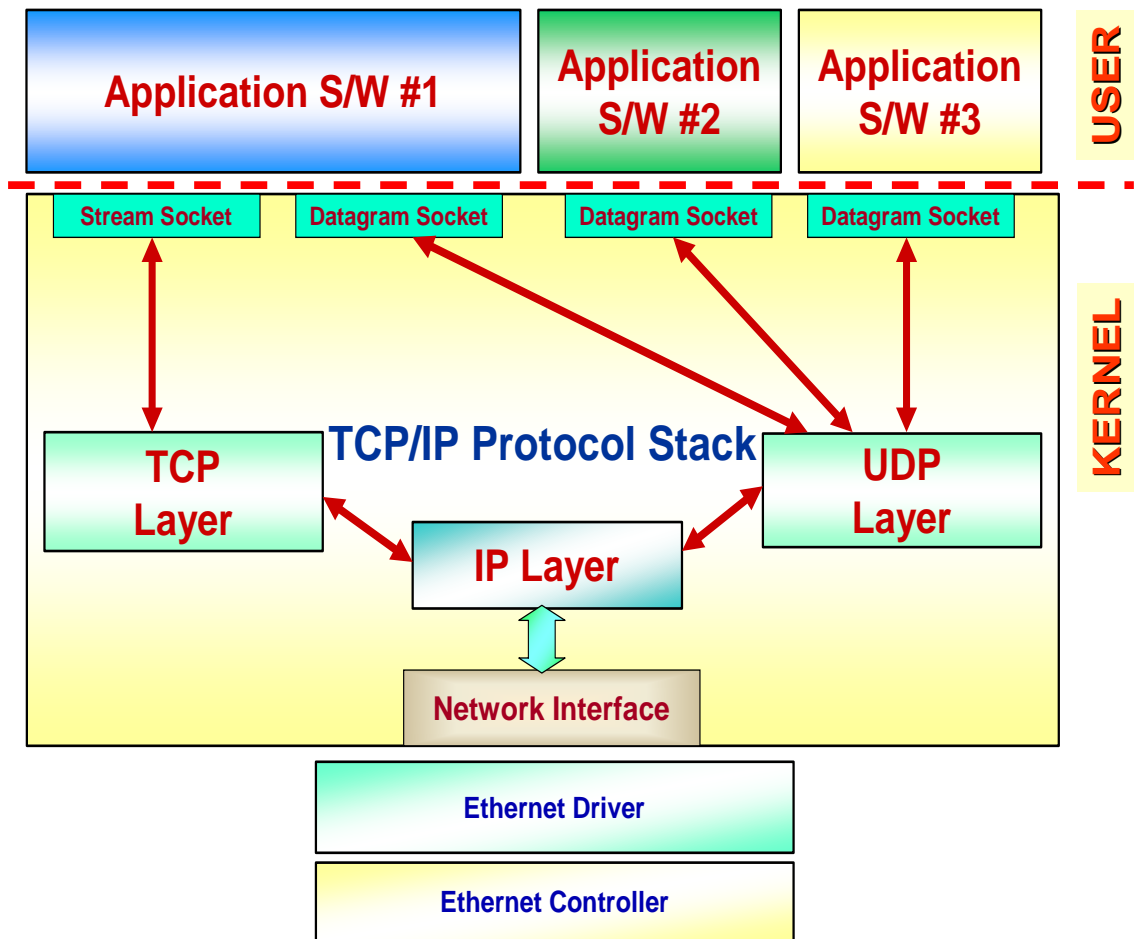


Figure 7.3 Protocol Stack Implementation

Note: An IP network connected device can have more than one network interface. The above schematic shows a single Ethernet-like network interface. Any software task running in user space can open one or more sockets of either stream or datagram type. The socket can be associated with a specific UDP or TCP port number, for instance using the `bind()` function under Linux, in order to receive the traffic with this specific destination port value. This is generally the case for a server application. If no port number is assigned to the socket, the protocol stack will assign a UDP or TCP port number when initiating a connection (TCP protocol) or sending a datagram (UDP). This

is generally the case for a client application. Socket interfaces provide read and write functions to send and receive data over the network.

7.4 AV CONTENT TRANSPORT FORMATS

There are several formats used to transport AV content over an IP network. The following sections will deal some of the most frequently used formats. AV content transport formats can be divided into several categories:

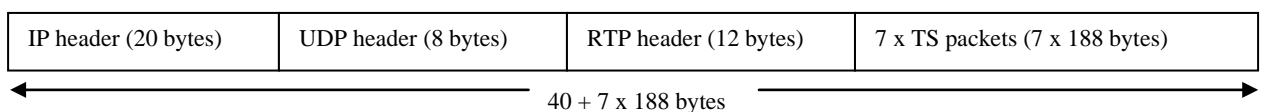
- **MPEG Transport Stream based Formats:** These formats use the MPEG Transport Stream format used in broadcast systems (satellite, cable, terrestrial) as a first packetization layer; TS packets are then aggregated into RTP/UDP datagrams.
- **Native AV Content over RTP Formats:** When the RTP protocol was originally designed, it was intended for each AV component (video, audio) to be transmitted over a dedicated RTP stream, according to payload formats defined for each audio or video codec. This is what most PC based media players do (QuickTime, Real,...), a control protocol like RTSP is used to set up separate RTP streams for each audio and video component. Thus, each video or audio stream is transported over separate RTP streams, each according to the RTP packetization format defined for this audio or video content.
- **HTTP based Transport Formats:** On an in-home network, where the distance between the server and the receiver is short (resulting in a short transmission time) and where there is a single receiver (no multicast), it is possible to use this transport protocols and formats that retransmit eventual lost packets, for a better

picture and sound quality. Thus, this format is mostly used on in-home networks (like DLNA), the AV payload is transferred just like a data file downloaded from a web site, using an HTTP GET method. The Digital Living Network Alliance is actually recommending such a media transport system in its guidelines. (DLNA, 2004)

As mentioned above, MPEG Transport Stream based Formats is used for our application, so only this format is discussed in detail further.

7.4.1 MPEG Transport Stream based Formats

DVB IP Transport Format and Encapsulation: DVB has created a technical group, TM-IPI (Technical Module, IP Infrastructure) in charge of defining standards for IP streaming on operator networks as well as inside the subscriber’s home. The first version of the DVB-IP standard has been published by ETSI in 2005: ‘Transport of MPEG-2 Based DVB Services over IP Based Networks’ (DVB IP –TS 102034 v1.1.1). DVB-IP defines a format using MPEG-2 transport stream encapsulated in RTP over UDP, using the IETF standard encapsulation method defined in RFC 2250. The main requirement is that a RTP packet must contain an integral number of MPEG TS packets. Since the maximum IP packet size that can fit into a single Ethernet frame (MTU) is 1500 bytes, and given the size of IP, UDP and RTP headers, there are generally 7 MPEG TS packets in each IP packet.



The RTP header includes a time stamp that must be “derived from the sender’s 90KHz clock reference” (PCR) and is “representing the target transmission time for the

first byte of the packet.” (RFC 2250) The time stamp is intended for jitter correction and clock recovery purposes. The RTP header also includes a sequence number that can be used to detect lost packets as well as out of sequence packets. According to DVB (DVB IP –TS 102 034 v1.1.1) this format can be used to encapsulate any MPEG-2 Transport Stream, whether containing single PTS (SPTS) or multiple programs (MPTS), with constant bit rate (CBR) or variable bit rate (VBR). Transport streams containing multiple Program Clock References (PCRs) are, by definition, constant bit rate streams. In practice, in most IPTV systems, the data link from the servers to the IP STB has a limited bandwidth (e.g. ADSL), so a single program transport stream or a partial transport stream will be used.

Direct MPEG TS over UDP Format and Encapsulation: Even though DVB-IP specifies the use of an RTP header for transporting MPEG-2 transport stream over IP, many actual IPTV deployments, especially in Europe, do not use RTP, but a direct UDP encapsulation instead. There is a proposal in DVB TM-IPI to include this “de facto” format standard into the next revision of the DVB-IP standard.

The main reason for not using RTP is that the timing information provided by the time stamp are already present in the MPEG TS (PCR) and that lost packets can be detected when parsing the transport stream.



In this format, an integral number of MPEG TS packets are transmitted in a UDP datagram. As for RTP encapsulation, the “magic number” is seven MPEG TS packets in each IP packet, resulting into a 1344-byte IP packet in each Ethernet frame.

7.5 IP STREAMING RECEIVER SOFTWARE

7.5.1 Server Push or Client Pull

Server Push: When IPTV operators are sending live TV channels to several IP-STBs at the same time, this is called a “Server Push” model. Each and every receiving IP STB has to play the AV content at the same rate it is being sent (pushed) by the server. This also implies that the IP STB has to synchronize its local clock to the IPTV server via a clock recovery method.

A “Server Push” system

- Allows streaming from one server to several client receivers (multicast).
- Impose that each client synchronizes its local clock to the server’s clock.

Client Pull: In a “Client Pull” system, the receiver’s local clock is the reference clock for AV content playout, so there’s no need for clock synchronization to the server’s clock. By definition, a “Client Pull” system is a one-to-one system: it is not adapted to “live” TV channels streaming to multiple receivers. A “Client Pull” system

- Is always streaming contents to a single client receiver.
- Does not force the receiver to synchronize its local clock to the server’s clock.

When streaming AV content to a single receiver, a “Server Push” model can also be used, but it is also conceivable to have the receiver (client) pacing the content delivery rate from the server (“Client Pull” model). The receiver would instruct the server via the network to “send more data” whenever the receiver is ready to accept it. A “Client Pull” method is workable in practice if the whole network path between the server and the receiver has a relatively short round trip communication delay and low error rate (because

any packet loss would result into the receiver asking the server for a resend). This would typically be the case on the same in-home network. All “Server Push” systems use UDP based transport formats (with or without RTP), which allows for IP multicast streaming. “Client Pull” systems could use UDP/RTP transport formats, with another protocol (such as RTSP) for flow control interaction with the server, but in practice, TCP based transport is used, since TCP has a built-in flow control mechanism. DLNA specified HTTP based transport is an example of a “Client Pull” system.

7.5.2 MPEG TS based Receiver Software Stack

When using an MPEG TS based transport format, such as DVB-IP, the client receiver software will be able to use the existing demultiplexing resources provided by the DEMUX hardware and the other Software drivers. TS packets are received via the appropriate UDP socket interface (with or without RTP header) and copied in system memory, Then the DMA(Direct Memory Access) is programmed to inject these TS packets into the DEMUX sub-system, using SWTS input.

Since the Transport Stream contains PCR and PTS time stamps, they can be used to synchronize the audio and video streams and for the head-end clock recovery. For the audio and video drivers, this injection from memory via SWTS is essentially indistinguishable for other SWTS injection, for instance, when playing a stream back from a disk drive.

7.5.3 RTP Elementary Stream based Receiver Software Stack

When using an ES over transport formats, such as those defined in ISMA, the client receiver software does demultiplexing of audio and video, since the different audio and video streams will arrive on separate RTP streams, each on different sockets.

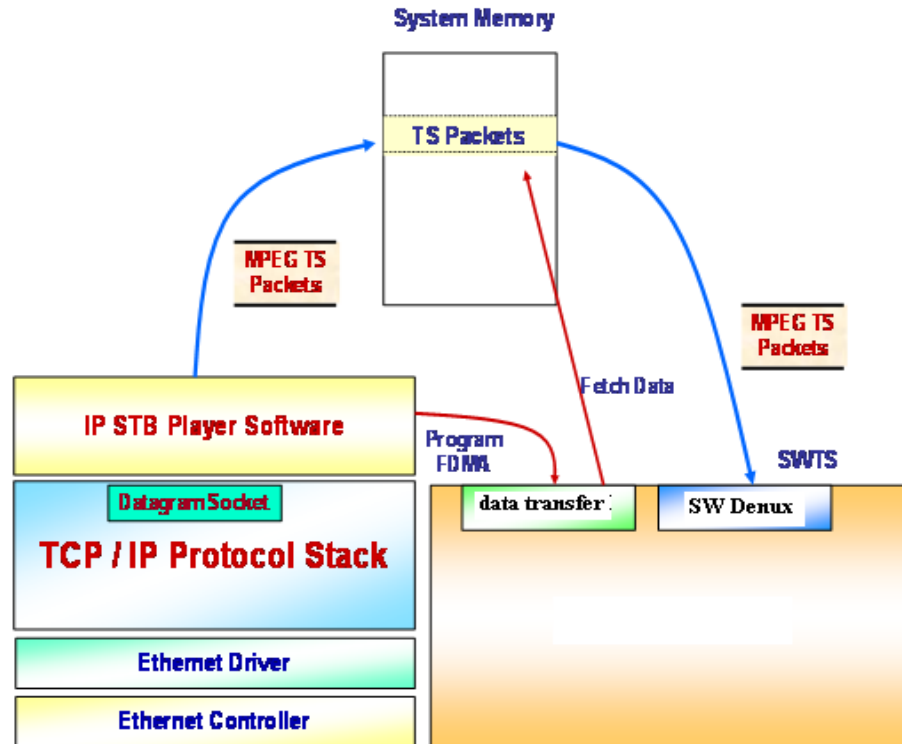


Figure 7.4 MPEG TS based Receiver Software Stack

The main issue is that the RTP payloads are in the form of Elementary Streams and neither contain any start codes nor PTS time stamps. As for now, Software drivers require PES (Packetized Elementary Stream) packets as input. So, the RTP client receiver software will have to artificially reconstruct a PES packet out of the RTP payload, before submitting it to the audio or video driver. A PES header must be reconstructed, with a specific start code for each RTP stream and a 33-bit PTS field that will be derived from the 32-bit RTP time stamp. The resulting PES packets can then be fed to AUDIO or VIDEO drivers for PES parsing and further decoding.

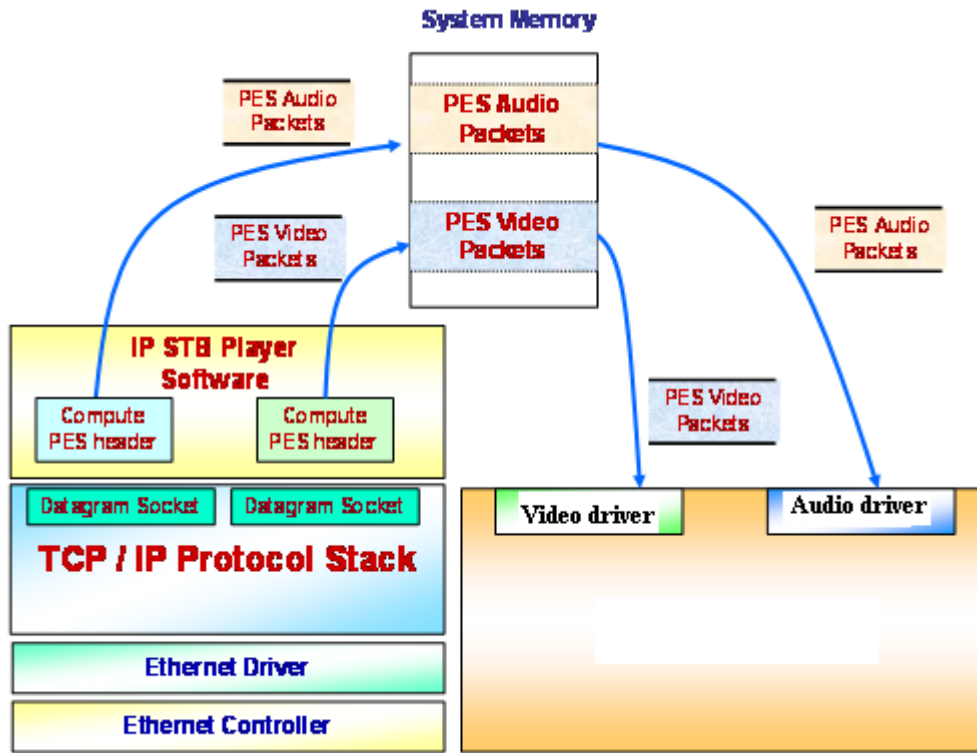


Figure 7.5 RTP ES based Receiver Software Stack

7.6 IP STREAMING RECEIVER CLOCK SYNCHRONIZATION

In “Server Push” systems using UDP transport (with or without RTP), the IP streaming receiver system must synchronize its local clock to the server’s clock: if the local clock is even slightly slower than the server’s clock, the receiver will play the audio and video payload at a slower rate than it is being sent out by the server. Inevitably, the receiver’s buffers will inevitably overflow at some time. Similarly, if the local clock is faster than the server’s clock, the receiver will play the audio and video at a faster rate than what the server is sending out. In that case, the receiver’s buffers will be empty (underflow). There are several techniques that can be used in such a situation: either skip or repeat a frame in the case of overflow or underflow in order to accommodate the slight

difference in the server's and receiver's clock frequencies. Such a "skip & repeat" may produce visible and audible artifacts in the video and audio. Another technique will aim at synchronizing the receiver's clock with the server's clock, using the timing information received from the server. This clock synchronization is being analyzed (section 7.6.1) and is found difficult due to the large jitter found in IP networks.

7.6.1 Clock Synchronization for TS based Transport

In MPEG transport stream, the PCR time stamps are present after every 100 ms. Each time a PCR time stamp is received from the server, the corresponding local System Time Clock (STC) is read. If the receiver clock is perfectly synchronized with the server's clock, the PCR time stamps should increase at exactly the same rate as the STC time stamps. When plotted on a graph in Figure 7.6, the (PCR, STC) pairs should result into a straight line with a slope of 1. Except, that due to the IP network jitter, the STC value will vary and the points will look more like a "cloud of points". However, the long term trend of this "cloud" should average to a straight line. A clock recovery algorithm is aimed at computing the slope of the line out of the accumulated (PCR, STC) pairs measured over time. If the computed slope is less than 1, the STC values grow at a slower rate than the PCR values, indicating that the local clock frequency is lower than the server clock frequency and should be adjusted upwards. If the slope is greater than 1, the STC values grow at a faster rate than the PCR values, indicating that the local clock frequency is higher than the server clock frequency and should be adjusted downwards. A clock recovery system is essentially a feedback loop control system: the computed slope value (frequency error) is used to correct the local clock frequency (frequency synthesizer) which results in a new STC rate of increase and so on.

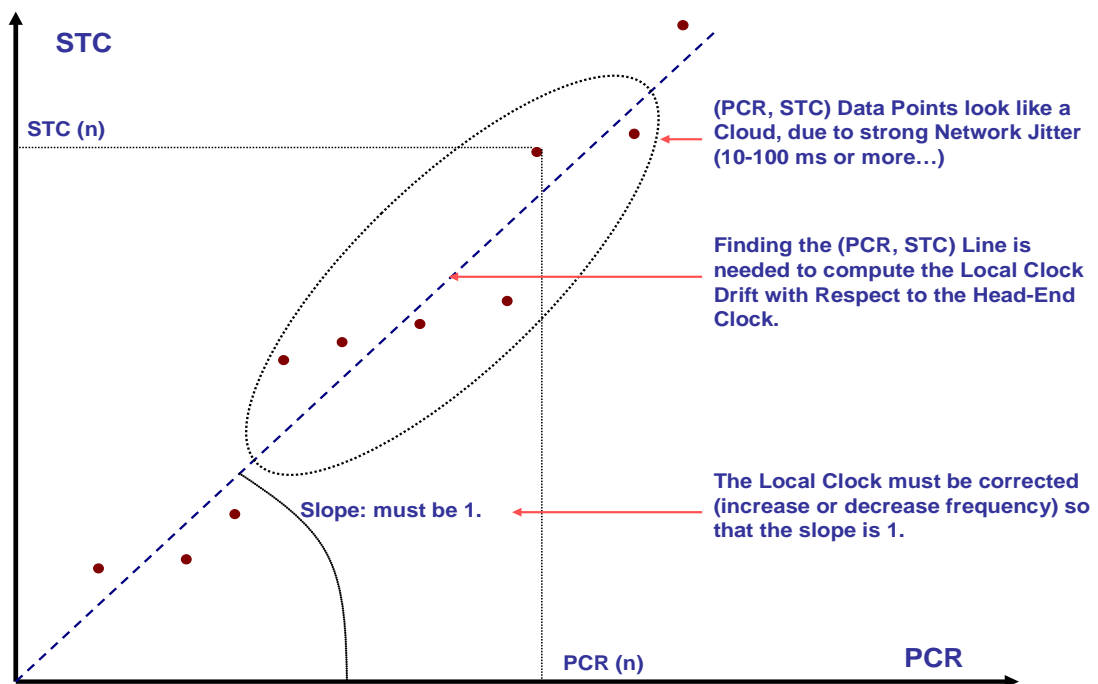


Figure 7.6 PCR vs STC

A basic approach to implement such a clock recovery system is based on PLL design that, by contrast, provides satisfactory clock recovery only when the network jitter is minimized and thus it is appropriate only for conventional network services like those provided by circuit-switching networks. A more suitable approach for packet switching networks is LLR based design. This approach is also not suitable for IP-STB as already explained in chapter-2. Therefore, a strong need is felt for suitable clock recovery algorithm for IP set-top box.

In this chapter, detailed study of IP environment for developing clock recovery algorithm has been presented. The present chapter has described the main technical issues associated with the application of streaming AV contents over a telecom operator's IP network to an IP-STB. In-depth study of IP environment and analysis of various issues helped us to develop a robust and stable clock recovery algorithm for IP-STB. Our newly developed algorithm would be presented in next chapter.

Chapter 8: Clock Recovery for IPTV

In this chapter, analysis of Jitter and color loss in IP environment has been done. Limitations of existing linear regression techniques have been discussed. New continuous adaption enhancements in linear regression algorithm for IP environment have been proposed. Simulation results and performance in real environment has been illustrated.

8.1 ANALYSIS OF JITTER AND COLOUR LOSS IN IP ENVIRONMENT

IP networks are asynchronous by nature. Each packet may have a different transit time between source and destination. Even consecutive packets of the same size from the same server to the same receiver will have a variable transit time, depending on the rest of the traffic on the network. Such a variation of the average packet transit time between server and receiver is called network induced jitter. Jitter on IP networks is often in the order of tens of milliseconds, sometimes several hundreds of milliseconds.

In a highly jittery IP environment, possible sources of error are:

- Transmission delay
- Encoder processing delay
- Receiver processing delay

Above delays can be combined into two components 'latency' and 'jitter'. Mathematically the latency is the mean delay, and the jitter is the standard deviation of the delay. The standard deviation is a measure of how widely values are dispersed from the average value (the mean). It is the 'jitter' component that causes the problem for clock recovery. When data is transmitted over an IP network, the data packets are collected and placed in the correct order before entering the clock recovery process.

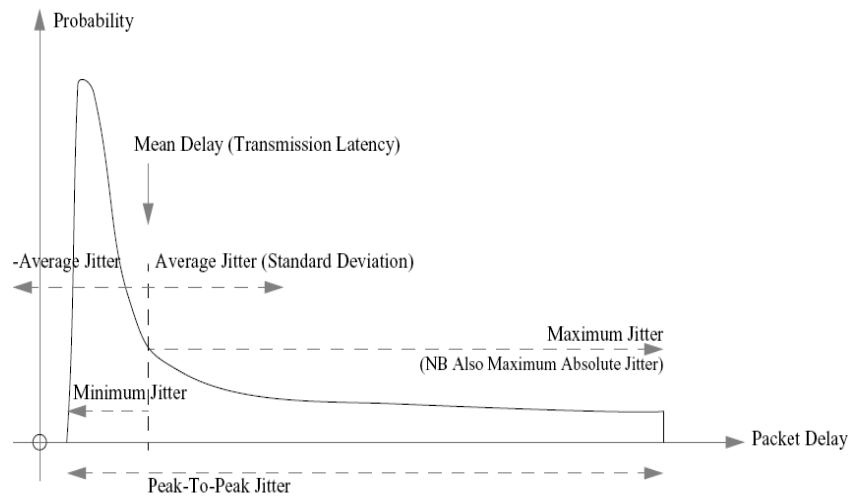


Figure 8.1 Jitter distribution

This re-ordering process introduces a large asymmetry into the jitter distribution. Some key characteristics of jitter distributions are shown in above Figure 8.1. The transmission delay is composed of a few long delays, and a large number of short delays. The average of these values is the mean delay, or the average transmission latency. The decoder can not measure the mean delay, the decoder can only detect if a packet arrives earlier or later than average. Although each measurement is subject to jitter, if the time period T is increased, then the jitter becomes a smaller component of the clock difference. Therefore for a given jitter error, we have to average over a sufficiently large time period T to reduce the error in the calculation to an acceptable level. However, the receiver can not afford to wait a long time to calculate the clock frequency. So another method has to be found. This method is ‘filtering’, an estimate of the clock difference is made after one or more new PCR/STC pairs have been received, and the estimates are ‘averaged’ to reduce the amount of ‘jitter’. Initially this estimate will have a large amount of error, but this will reduce as more and more PCR/STC values are added to the average. The resultant ‘averaged’ jitter will reach an acceptable amount as the time approaches T . This was the approach used previously and have been discussed in chapter 5 & 6. Our old

algorithm (chapter 5 & 6) performed well for use with satellite and terrestrial data because the amount of jitter on these networks was relatively small. However, there is a new requirement for clock recovery to work with data transmitted over highly jittery IP network.

According to the DVB-IP standard (DVB-IP, 2005), the amount of jitter expected on an IP network is typically about 20 milliseconds, it is assumed this is the standard deviation of the delay. However, as can be seen from Figure 8.1, the maximum delay is probably much larger. Values up to several hundred of milliseconds might be experienced under some circumstances. To avoid stalling the decoder because of late packets, a buffer is generally used to store a certain amount of received packets prior to sending them out to the demultiplexer and decoding chain: this is called the jitter buffer. As a rule of thumb, the jitter buffer should be large enough to store at least the amount of packets that correspond to the maximum jitter expected on the network. For instance, if the maximum expected jitter is 40ms peak-to-peak (recommended maximum jitter value by DVB-IP), the jitter buffer size should large enough to store at least 40 ms worth of AV payload packets. If the AV stream average bit rate is, say 4 Mbps, the jitter buffer size should be at least: $(4 \times 10^6 \times 40 \times 10^{-3}) / 8 = 20,000$ bytes

A second rule of thumb is to start playing out the AV payload packets when the jitter buffer is about half full as shown in Figure 8.2, to further allow the buffer content to vary around this mid-point, from “almost full” to “almost empty”. The downside of inserting a large buffer between the protocol stack and the AV processing (demux and decoding), is the added latency when switching the IP streaming receiver to a new AV stream.

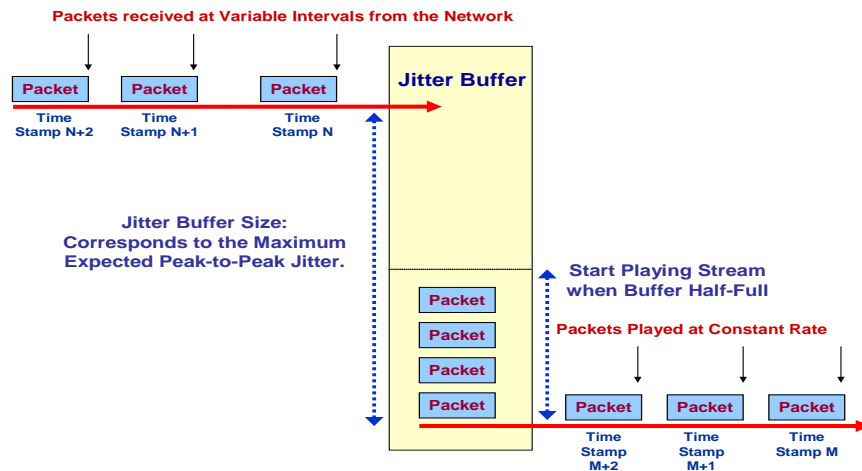


Figure 8.2 Jitter and Latency in IP environment

- The jitter buffer must first be flushed
- Then the buffer must be filled up again with the AV payload packets corresponding to the new AV stream
- When the jitter buffer is about half full, then the AV playout can resume with the new AV stream.

In the DVB-IP example (40ms peak-to-peak maximum jitter), this would add another 20 ms to the overall channel change time. If the jitter buffer is sized for 500 ms peak-to-peak maximum jitter, then the additional latency when changing channels will be 250 ms. As a general rule, limiting the jitter and the need for a large jitter buffer is crucial to improve the interactivity of an IP streaming system.

Some constraints are also posed by the output monitor as well. The TV receiver connected to the set top box (STB) has to locate a color burst signal in order to decode the color signal. According to Robin and Poulin (2000), the frequency of the color burst must

be accurate to 50 ppm that is +/- 1350 Hz at 27 MHz. In STB, if the decoder clock has a large error, then the color burst signal will not be found. The requirement of finding colour burst signal is important due to below explained reason-

When the color burst has been successfully found, a PLL in the receiver is used to track subsequent changes in the color burst frequency. After sampling the small color burst signal at the start of each line, the PLL generates the color reference for the duration of each line. If the STB frequency changes too rapidly, then a 'color shift' is observed as the PLL momentarily loses track. Although this is a known problem, at present no specification for the maximum acceptable rate of frequency change was found in any standard. This maximum drift rate was approximated under the assumption that typical time constant for a TV PLL is 15 ms and STB drift rate tolerance should be 1000 times faster than this time constant. Under these mentioned assumptions, we arrive at a maximum acceptable drift rate of about 67 KHz/second. If the local clock is adjusted every 100ms, this would limit the FS output correction to a maximum of 6.7 KHz.

Thus from above analysis, we conclude that to prevent color loss the local frequency overshoot should be less than 1350 Hz. (overshoot may occur, while applying correction on decoder clock to achieve clock recovery). Secondly we can infer that at present no figures exist for the maximum rate of change of frequency. However, larger step changes in frequency are likely to upset the PLL circuitry in a TV monitor and cause a visible 'color shift'.

8.2 NEW PROPOSED ALGORITHM FOR CLOCK RECOVERY IN IP SET-TOP BOX

LLR algorithm is the simplest and most commonly applied form of linear

regression (Weisstein- least squares, 2009) and provides a solution to the problem of finding the best fitting straight line through a set of points. Regression is a mathematical procedure for finding the best-fitting curve to a given set of points by minimizing the sum of the squares of the offsets of the points from the curve. The old application of Least squares Linear Regression (LLR) algorithm for achieving clock recovery in IP environment monitors Program Clock Reference (PCR) arrival and generation time. This is done by finding the PCR and corresponding System Time Clock (STC) error values in consecutive samples. One of the problems of the LLR algorithm was the requirement for an additional dejittering buffer to store the sample set of incoming PCR and STC values. Moreover, the algorithm uses floating point precision which may increase the time for computation and hence increase RTOS system instability for application on STB. Furthermore, the frequency-drift rate of the algorithm was indeterministic, too. If the drift rate increases more than 0.1 Hz, color loss could be a potential problem.

8.2.1 Proposed Algorithm

A continuous feedback loop has been designed to solve above mentioned problems in the LLR algorithm. In our newly devised enhanced LLR algorithm, the X value is the PCR difference (called PCR_diff), and the Y value is a simulated pseudo buffer level (rather monitoring of STC) which varies according to the sum of current errors between the PCR and STC values. The reason for this design change is explained below. Many existing set top boxes suffer from high incidence of dropped or skipped frames in DVB-IPTV environment. Let's consider input buffer at the decoder end, if the decoder frequency is too low, this buffer will overflow. The decoder can crudely correct the buffer level by skipping a frame. Similarly, the decoder frequency that is too high may lead the decoder to pause as no frames are available to decode. On correction of the

local frequency, the buffer level should stabilize and should no longer overflow or underflow. However, the buffer may now be operating at nearly full, or nearly empty. The natural variability of the compressed input data rate may well still push the input buffer into an overflow or underflow state.

We proposed solution to above problem by change in clock recovery algorithm, such that errors in buffer level are corrected, rather than errors in frequency. For the purpose of clock recovery, it is not necessary to know the real input buffer level. A simulated ‘pseudo buffer’ can be created as follows:

$$Buf_lvl = Buf_lvl + Err_diff \quad (1)$$

Here, Err_diff is difference of encoder and decoder clock as calculated by subtracting the PCR and STC errors. If the encoder frequency is faster than the decoder frequency, then Err_diff will be positive, and the buffer level will be increased. The true buffer level is not monitored but pseudo buffer level, a function of timestamp errors is monitored. This simulated buffer level is fed to LR algorithm.

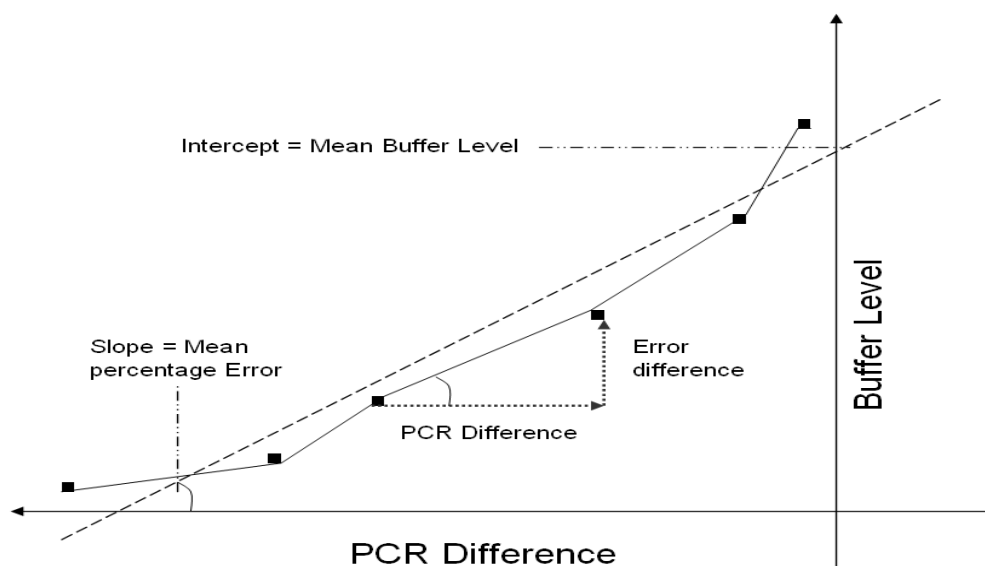


Figure 8.3 Principle of Enhanced LLR algorithm for Clock Recovery in IP environment

In the enhanced LR algorithm as shown in Figure 8.3, the X-value is PCR error called PCR_diff, and the Y value is the simulated buffer level. If a triangle is drawn using two adjacent points, as shown in Figure 8.3, then the horizontal edge represents the PCR difference, the vertical edge represents the Err_diff, and the gradient is the incremental percentage error. The slope of the fitted line is the mean percentage error in decoder frequency. If the Y-axis is drawn through the current data point, then the Y intercept is an estimate of the mean buffer level. The LR algorithm calculates the intercept and slope. The algorithm minimizes the errors by minimizing the sum of least squares of the sample points. The formulae for obtaining the slope and intercept (Weisstein- least squares, 2009) from N data points are-

$$Slope = \frac{\left[N \left[\sum_{i=1}^N x_i y_i \right] - \left[\sum_{i=1}^N x_i \right] \left[\sum_{i=1}^N y_i \right] \right]}{\left[N \left[\sum_{i=1}^N x_i^2 \right] - \left[\sum_{i=1}^N x_i \right]^2 \right]} \quad (2)$$

$$Intercept(K) = \frac{\left[\left[\sum_{i=1}^N y_i \right] - \left[m \cdot \left[\sum_{i=1}^N x_i \right] \right] \right]}{N} \quad (3)$$

The above values for slope and intercept are estimates. To see how this relates to the current application, following assumptions are made

- Number of data points are large (represented by N)
- PCR_diff is approximately constant equal to L
- Mean of Err_diff is zero and standard deviation is σ_y
- The encoder and decoder frequencies are close that is the unfiltered percentage error (σ_y/L) is small, that is the slope (m) is very small compared to the intercept.

Based on equation (2) & (3) and applying above assumption, Slope error and Intercept error are derived in appendix IV at section B4.1.1.2 and at section B4.1.1.1 and rewritten as below-

$$Slope_Error \approx \frac{\sigma_y}{L} \times \sqrt{\frac{48}{N^5}} \quad (4)$$

$$Intercept_Error \approx \frac{\sigma_y}{\sqrt{N}} \quad (5)$$

As N increases, the error in the pseudo buffer level estimate (intercept) reduces by a factor of $1/N^{0.5}$, this is the same as would be obtained by strict averaging. However, the percentage error (slope) reduces by a factor of $1/N^{2.5}$, which is 5 times quicker than averaging.

8.3 ALGORITHM ENHANCEMENTS

Clock recovery is continually receiving new data, and therefore N is continuously increasing. If left unchecked eventually the buffer will overflow. Also, as time progresses the relationship between the clocks may change. For instance due to program change or temperature drift etc. Therefore, the newer data is more likely to predict the correct frequency correction than the older data. Three approaches are considered for handling these problems -

- Piece-wise linear adaption
- Overlapped piece-wise linear adaption
- Continuous adaption

Each of these approaches will now be considered in more detail.

8.3.1 Piece-wise Linear Adaption

With the piece-wise linear method, the continuous stream is split into pieces, and LR is performed on each piece. The size of each piece is important. This could be measured by using the number of PCR values or the number of encoder clock ticks. The more data points in each piece, the more noise reduction is achieved. However, more data points also increase the latency before a calculation can be made. Furthermore, as the size of each piece increases, the rate at which frequency corrections are applied is reduced. Also when the corrections are applied they occur in large steps. These large steps could upset the stability of the algorithm, and also create other adverse effects such as color loss etc. Clearly some extra processing would be required to cope with these unwelcome side-effects.

8.3.2 Over-lapped Piece-wise Linear Adaption

In order to reduce some of the artifacts of the above technique, the pieces of the LR could be overlapped. That is, a new LR analysis could be started every n data points, where $n < N$. Each LR still uses N data points. The smaller the value of n , the smaller the steps sizes would be on each frequency update. The down-side of reducing n is the amount of extra processing required. Also circular buffers would be required to hold the previous $(n + N)$ data points.

8.3.3 Continuous Adaption Technique

Neither of the piece-wise methods successfully addresses the issue of weighting the newer data more highly than the older data. With both the piece-wise methods the last N data points are weighted equally (weight one), and older points are not used (weight zero). The proposed continuous adaption method **smoothly reduces** the weighting of data points as they age. Furthermore, with the continuous method, a new frequency correction

can be produced every data point (similar to the overlapped case with $n = 1$), and all this is achieved without the extra overhead of maintaining circular buffers. This continuous adaption method has the following 3 refinements: Exponential weighting, X co-ordinate shift and Parameter Scaling. These will now be discussed in more detail.

8.3.3.1 Exponential Weighting

In the simple piece-wise algorithm, a data point is used once, and then never again. This is equivalent to having a weight = 1 when it is used, and then a weight zero when it is not used.

Now consider a data point with the following series of weights a, ab, ab^2, ab^3, ab^4 , where $a + b = 1$. In the continuous scheme, the first time a data point is used it has a weight a , each subsequent time the data point is used the weight is reduced by a factor b . This achieves the goal of **smoothly reducing** the importance of a data point as it ages. Furthermore, the sum of this infinite set of weights is also 1, so it has the same overall contribution as if it was used once only. Another feature of using the above weighting schedule is that it eliminates the requirement to store all the previous data points. Consider how this operates for the summation of Y values -

$$\sum_{i=0}^N ab^{N-i} x_i = ax_N + b \sum_{i=0}^{N-1} ab^{N-1-i} x_i \quad (6)$$

This may be relabeled as

$$WeightedSumX_n = ax_n + b.WeightSumX_{n-1} \quad (7)$$

The previous weighted sum of **X** values is reduced by the factor **b**, and added to the new data value weighted by the initial factor **a**. All the other sums in equations (2) and (3) are similarly updated. Therefore no circular buffers are required. In the piece-wise algorithms, the noise reduction was improved by increasing the value of **N**. In the continuous algorithm the noise reduction is increased by reducing the value of the parameter **a**. For convenience, 'a' is expressed as a power of 2 as follows-

$$a = \frac{1}{2}^{\text{Filter_strength}} \quad (8)$$

If Filter Strength (FS) is a positive integer, then **a** is an exact power of 2 and fast algorithms are available for performing the summation updates. If the input jitter is doubled then the value of **a** needs to be halved to achieve the same output jitter value. The slope and intercept error is now calculated in appendix-IV at section B4.1.2.2 and at section B4.1.2.1 as written below -

$$\text{Weighted_Slope_err} \approx \frac{\sigma_y}{L} \sqrt{\left(\frac{e}{8}\right) a^5} \quad (9)$$

$$\text{Weighted_Intercept_err} \approx \sigma_y \sqrt{\frac{a}{2}} \quad (10)$$

With exponential weights, $1/a$ seems to play a similar role to that of the number of samples (**N**) in equation (4) and (5).

8.3.3.2 X Co-ordinate Shift

Potentially we are now dealing with an infinite data stream. The PCR clock tick value (**X** value) could become very large. We have to keep the summation expressed in equations (2) and (3) finite. Also, the PCR_Diff values are being passed to the linear regression as the **X** values. These may become very large to handle. To prevent this, **X** coordinate is shifted to make the newest sample at **X=0** while the older sample lie in the 2nd quadrant. Thus another refinement is achieved by **X** coordinate shift.

8.3.3.3 Parameter Scaling

Even after the co-ordinate shift, there is a possibility that under some circumstances the summations could overflow, Therefore, the input values were downscaled by 'a' on entrance to the LR routine, and the output intercept was up-scaled by $1/a$. As the slope is a ratio of **X** and **Y** values, it does not require up-scaling. The

output of this feedback loop is used to provide the error correction as shown in Figure 8.4.

8.4 PROPOSED CONTROL FEEDBACK LOOP

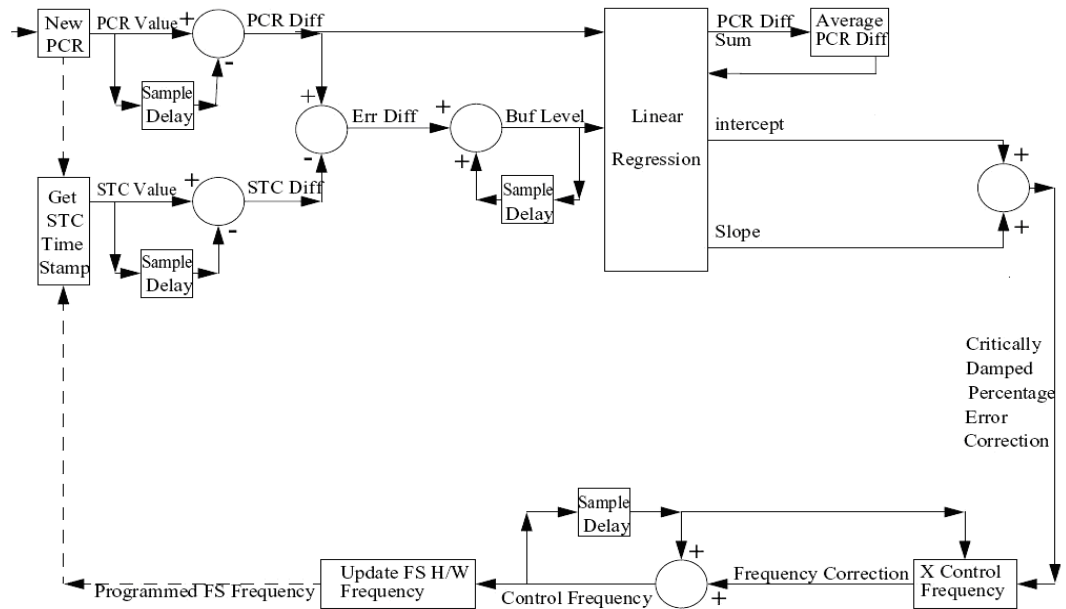


Figure 8.4 Clock Recovery module using Enhanced Linear Regression

Figure 8.4 shows a schematic of our proposed control feedback loop. It shows how LR algorithm interfaces to the rest of the clock recovery algorithm. As LR algorithm filtering the data, the feedback loop itself also offers some filtering capability. It is the filter response of the whole feedback loop, including the in-loop filter that predicts the behavior of the clock recovery algorithm. A thorough understanding of how the in-loop filter interacts with the feedback loop is critical to achieving a stable solution for clock recovery.

In a feedback loop errors can build up very easily, it is important to make every effort to keep the errors to a minimum. Therefore the algorithm using LR maintains an estimate of the local clock frequency at a higher precision than can be achieved by the FS

hardware. This value is referred to as the control frequency. Once a new estimate for the control frequency is available it is converted to FS parameters for programming the hardware. The quantized output frequency of the hardware is never directly used in the feedback loop. However the quantized frequency does indirectly govern the next STC measurement.

As discussed in Section 8.2.1, the inputs to the Linear Regression algorithm are the PCR difference, and the pseudo buffer level, and the outputs are the slope and intercept. Another output is the weighted sum of PCR differences (7), this is required to scale the intercept value. This will be discussed in detail below.

The slope is an estimate of the mean percentage frequency error, and indicates by how much the local clock needs correcting to make the local frequency equal to the encoder frequency. The intercept is an estimate of the mean pseudo buffer level and indicates how much the local frequency needs to be altered in order to return the pseudo buffer level to its ideal operating level. This is a design goal of the newly developed algorithm.

If the buffer has stabilized at its correct operating value, then the frequency must also have stabilized at the correct correction value. In fact, either the slope or intercept can be used with the appropriate damping factor to correct the FS frequency. However, using both estimates provides more flexibility in tailoring the transient response of the system, For example in achieving a fast settling time.

The slope is a ratio of the frequency difference divided by the PCR difference. The intercept is the mean pseudo buffer level and has the units of frequency. The intercept is normalized by dividing by the mean PCR difference so that the two estimates

have the same units and can be combined. The mean PCR difference is proportional to the weighted sum of PCR differences calculated in the LR routine, as below-

$$MeanPCRdiff = WeightedSumX \cdot ScaleUp \cdot \frac{a}{b} = -\frac{WeightedSumX}{b} \quad (11)$$

In above equation (11), value of the scaling parameter = $1/a$ is substituted (as mentioned in Section 8.3.3.3). Weighted sum needs to be stored to high accuracy to prevent unstable behavior. As the sum is still in units of percentage error, it is multiplied by the previous estimate of the control frequency to generate a frequency correction value. The frequency correction value is then added to the previous control frequency to generate the new control frequency. Once the control frequency has been updated it can be used to change the programmed output frequency of the Frequency Synthesizer. The programmed output frequency is quantized (roughly 50 Hz step size), and is therefore not the same as the control frequency. The quantized frequency is then used to measure the next STC time stamp. This closes the feedback loop.

8.4.1 System Response

The control feedback loop has been analyzed and a system response produced. When the outputs of the LR algorithm are combined, a damping weight of α is applied to the normalized intercept, and a damping weight of β is applied to the slope to achieve the fast settling time without any oscillatory behavior. The damping parameters α and β must have a range of values for which the system is stable, and provide a flexibility point for trying to meet the required design goals. For a critically damped system α and β were derived in appendix-IV at section D4.1 . The value of α and β are rewritten as below –

$$\alpha = (2\sqrt{3} - 3)S_a^2 \quad \text{and} \quad \beta = bS_a \quad (12)$$

where S_a is a scaling parameter and is equal to $a/6$. \mathbf{a} and \mathbf{b} are the exponential weighting parameters discussed in Section 8.3.3.1. The scaling parameter demonstrates that while α varies approximately linearly with \mathbf{a} , β varies as the square of \mathbf{a} .

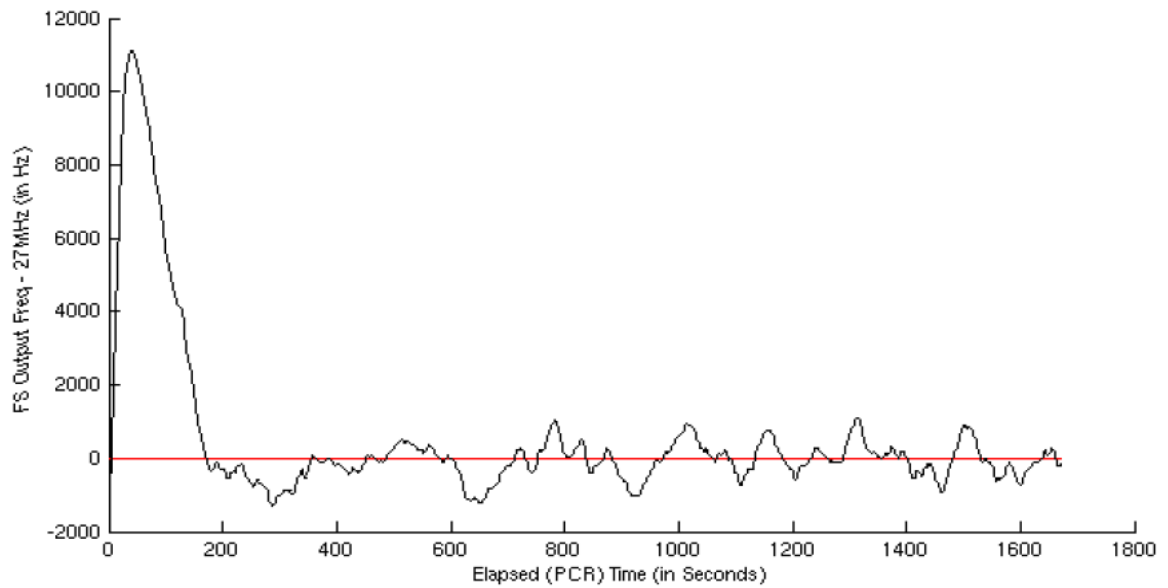


Figure 8.5 Quantized Frequencies in Response to 20ms Jitter

8.4.1.1 Over-Damping

The frequency response shown in Figure 8.5 shows a maximum excursion of about 11 KHz from the nominal frequency (27 MHz). This is far too large and does not meet the specification for the color burst of 1350Hz (See Section 8.1).

It was analyzed that to reduce the frequency overshoot either α or β can be reduced. However, this will increase the settling time. Hence, an ‘over-damping’ factor (**OD**) was introduced into the scheme to reduce the overshoot to an acceptable level. This is used to reduce the values of α and β . OD effectively introduces more filtering into the feedback loop. As discussed in section 8.4.1, the amount of filtering provided by α and β is controlled by a scaling parameter S_a . So this is the point at which the over-damping is introduced as follows -

$$S_a = \frac{(a/6)}{OD} \quad (13)$$

This results in β being over-damped linearly with OD, while α is over-damped as the square of OD. It may be possible to find a different combination of α and β that meets the over-shoot requirement and has a lower settling time. Another problem is the value OD is very much data dependant and has to be found empirically.

8.5 MATLAB SIMULATION OF PROPOSED ALGORITHM FOR IP ENVIRONMENT

Firstly algorithm has been simulated in MATLAB environment. Following different statistical jitter distributions (Weisstein- Pareto, 2009), (Weisstein-Weibull, 2009) have been studied.

- Pareto (Order 2)
- Pareto (Order 3)
- JMB (Order 1)
- JMB (Order 2)
- JMB (Order 3)
- Weibull (Order 1)
- Weibull (Order 2)
- Weibull (Order 2)

Where the order is represented by n in the following equations. The PDF's (Probability Distribution Function) for these jitter models are -

$$Pareto(x) = \frac{n}{(x+1)^{n+1}} \quad (14)$$

$$JMB(x) = \frac{2nx}{(x^2 + 1)^{(n+1)}} \quad (15)$$

$$Weibull(x) = nx^{(n-1)}e^{-x^n} \quad (16)$$

By suitable choice of parameter **n**, these distributions can all be made to approximate the network jitter curve shown in Figure 8.6

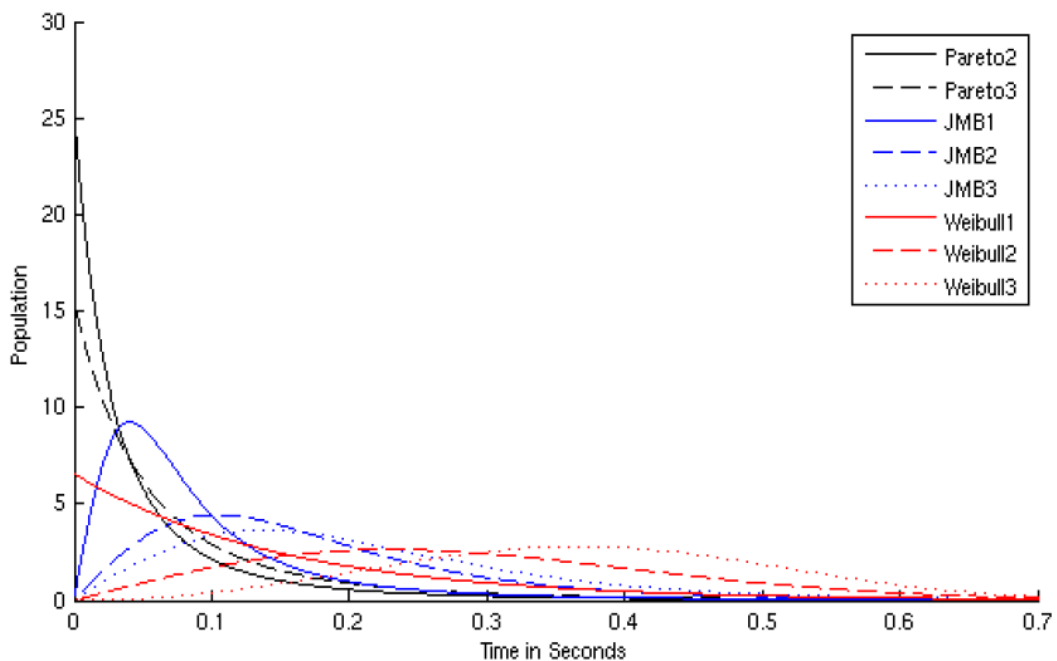


Figure 8.6 Linear Comparison of Different order of Jitter distribution Model

In our analysis, we found that Pareto (Order 2) Jitter distribution is very close to real time IP environment so same jitter distribution has been selected for testing in matlab simulation environment and real time testing. Although the jitter values are selected from a statistical distribution, the actual value chosen is determined by a uniform random number generator. The random generator can be seeded to produce repeatable test data. Figure 8.7, Figure 8.8, Figure 8.9 and Figure 8.10 shows the response of new LR

algorithm for Pareto-2 jitter distribution for maximum delays in PCR equals to 18 ms, 35ms, 52 ms and 70 ms. The decoder clock has been recovered in 20 seconds, 24 seconds, 35 seconds and 50 seconds respectively for above four scenarios.

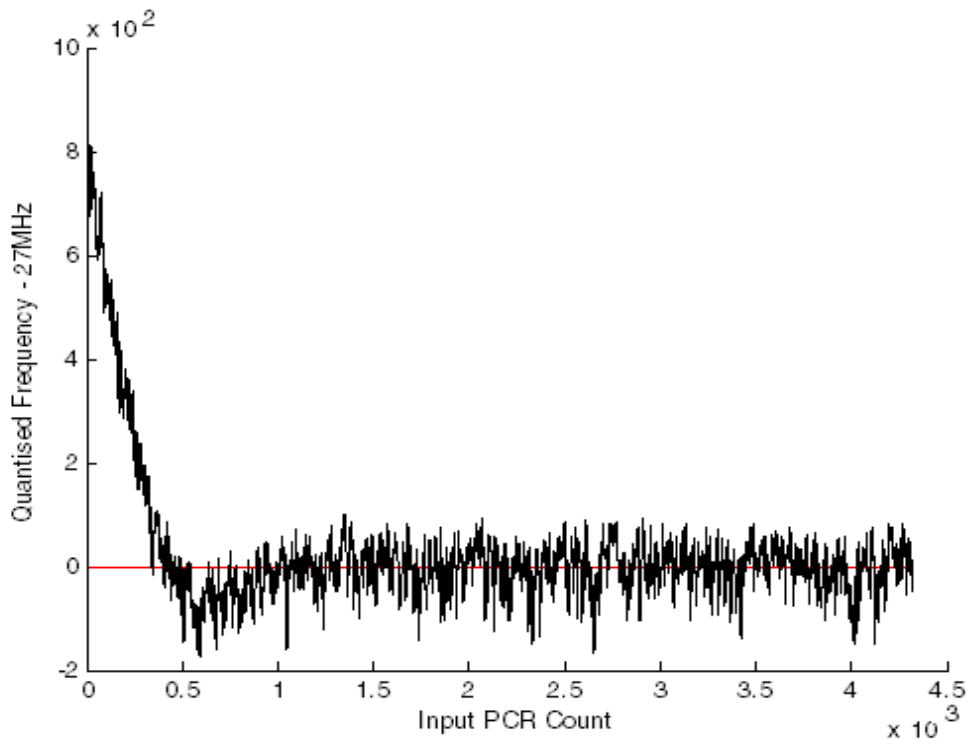


Figure 8.7 Performance of new LR algorithm in Matlab environment (Pareto-2 for jitter 18 ms with FS=3 OD= 4.83)

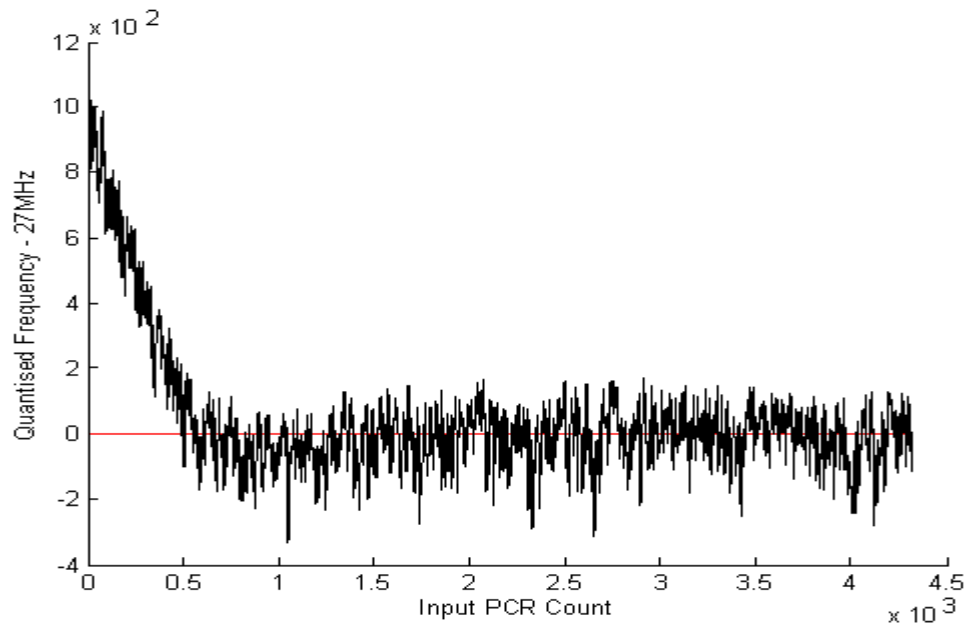


Figure 8.8 Performance of new LR algorithm in Matlab environment (Pareto-2 for maximum delay of 35 ms with FS=3, OD= 9.38)

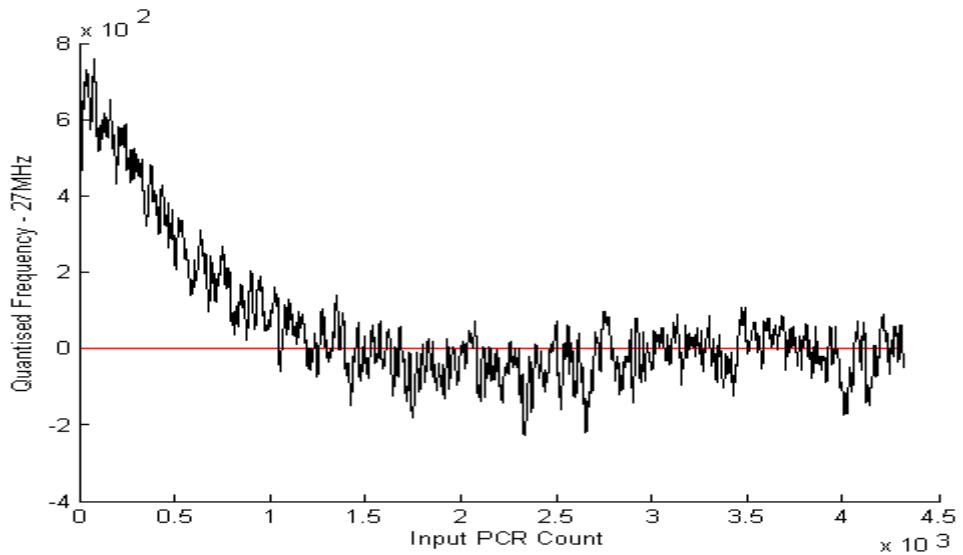


Figure 8.9 Performance of new LR algorithm in Matlab environment (Pareto-2 for jitter 52 ms with FS=3 OD= 18.73)

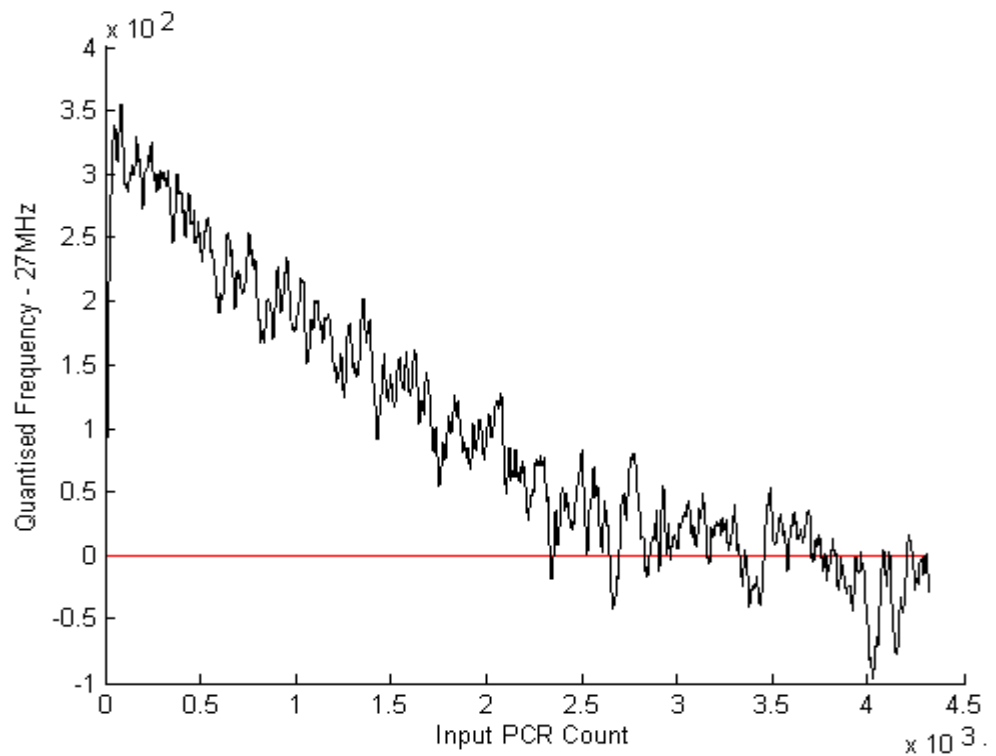


Figure 8.10 Performance of new LR algorithm in Matlab environment (Pareto-2 for jitter 70 ms with FS=5 OD= 18.73)

8.6 PERFORMANCE ANALYSIS OF NEW PROPOSED LR ALGORITHM IN REAL- TIME IP ENVIRONMENT

As discussed in section 8.5, new designed algorithm was checked in the Matlab simulation and had proved the competency. Now, this algorithm has been tested in real time environment. The algorithm was implemented for IP compliant STB RTOS environment. Test setup for this environment has been shown in Figure 8.11.

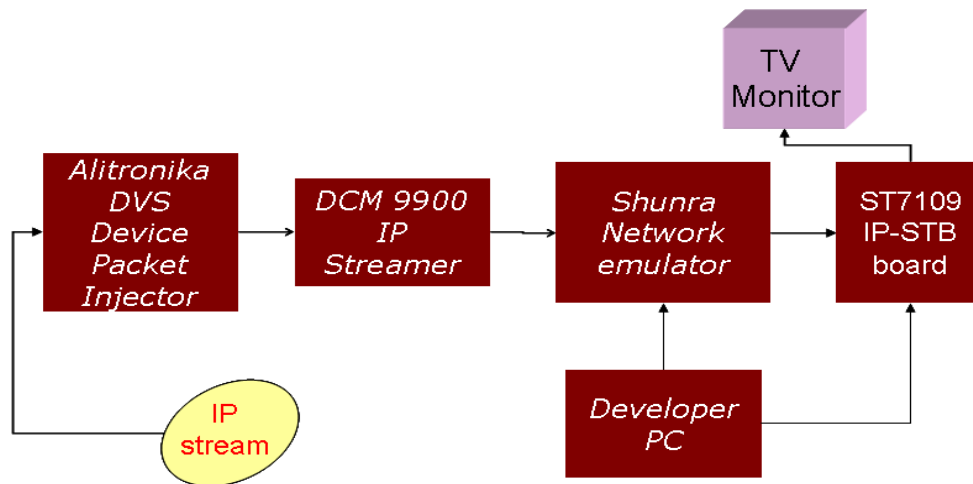


Figure 8.11 IP-STB clock recovery test setup

As per set-up shown above, the IP stream is sent to ‘Alitronika DVS Packet Injector Device’, which injects the MPEG stream to the DCM9900 IP streamer. IP streamer converts the MPEG stream to IP stream. The IP stream is then sent to ‘Shunra Emulator’, which introduces network jitter in the stream as required. The jittered IP stream is sent to the ST7109 SoC of IP-STB board, which uses the enhanced LLR algorithm for clock recovery. The STB is connected to TV receiver set to view any A-V frame skips or color loss. Pareto (Order-2) jitter distribution was used to simulate real-time IPTV environment. Several PCR-STC data-set was taken from live IP jittered VBR stream with artificially introduced jitter using ‘Shunra’ network emulator. Initial test result suggested that new algorithm couldn't stabilize the clock even in zero-jitter environment.

We analyzed whole system and could suspect if new IP data path itself has not introduced any jitter so we first calculated amount of average jitter in incoming stream. To calculate jitter in any stream, PCR & STC log have been taken without correcting decoder clock by our clock recovery algorithm. Certain analysis and tests(Test1-Test4) were conducted under various conditions with zero jitter introduced by shrunra emulator

to calculate average value of jitter in ms as shown in Table 8.1 to Table 8.4. We followed the below mentioned procedure for calculating average jitter-

By subtracting consecutive samples of PCR, PCRdiff in terms of tick error is calculated. For decoder clock of 27 MHz, time delay between two consecutive PCR samples can be calculated as-

$$t(PCR) = \frac{PCR_diff}{27 \times 10^6} = \frac{PCR_diff}{27 \times 10^3} \times 10^{-3} \quad (17)$$

$$t(PCR)(ms) = \frac{PCR_diff}{27 \times 10^3} \quad (18)$$

t(PCR)(ms) will give indication that in a specific stream if PCR's are encoded as per MPEG standard uniformly at approximately 40 ms. Similarly, by subtracting consecutive samples of STC, STCdiff in terms of tick error is calculated. For decoder clock of 27 MHz, time delay between two consecutive STC samples can be calculated as-

$$t(STC) = \frac{STC_diff}{27 \times 10^6} = \frac{STC_diff}{27 \times 10^3} \times 10^{-3} \quad (19)$$

$$t(STC)(ms) = \frac{STC_diff}{27 \times 10^3} \quad (20)$$

t(STC)(ms) will give an indication at amount of jitter presented in a specific stream. In next step mean value and deviation from mean is calculated. Further, average jitter (Standard deviation) using formula STDEV (STDEV (STC_Diff0:STC_Diffn)) has been calculated, where n is number of samples taken for Jitter analysis.

Test1- DCM 9900 IP streamer output was selected for VBR encoding stream at rate of 3.8 Mbps (max.) A sports channel (AV) is selected from the entire MUX and injected into

the loop. In this test there was no Shunra Emulator i.e. no network jitter was introduced.

Table 8.1 is shown for this test.

Table 8.1: Test1- SportsVBR3.8 - NoShunra

PCR	STC	t(PCR) in ms= PCRdiff /27000	t(STC) in ms= STCdiff /27000	Mean	Deviation from mean	Average Jitter (ms)
525643478825	22536739566	-----	-----	35.24309	-----	23.62686
525644412463	22536786855	34.57919	1.751444		33.49164556	
525645388829	22538148289	36.1617	50.42348		-16.18039148	
525646320331	22539571238	34.50007	52.70181		-17.45872481	
525647291359	22539628614	35.964	2.125037		33.11805296	
525648218588	22541021051	34.34181	51.57174		-16.32865074	
525649175726	22542447548	35.44956	52.83322		-17.59013222	
525650126457	22542506883	35.21226	2.197593		33.04549741	
525651076118	22543930043	35.17263	52.70963		-17.46653963	
525652016165	22545345549	34.81656	52.42615		-17.18305815	
525652963691	22545391812	35.09356	1.713444		33.52964556	
525653925103	22546787941	35.60785	51.70848		-16.46539148	
525654879039	22548200158	35.33096	52.30433		-17.06124333	
525655813745	22548246913	34.61874	1.731667		33.51142333	
525656770884	22549684703	35.44959	53.25148		-18.00839148	
525657732297	22551076484	35.60789	51.54744		-16.30435444	
525658688370	22551123722	35.41011	1.749556		33.49353444	
525659679692	22552538099	36.71563	52.38433		-17.14124333	
525660601580	22553930536	34.144	51.57174		-16.32865074	
525661562992	22553987939	35.60785	2.126037		33.11705296	
525662514790	22555390630	35.25178	51.95152		-16.70842852	
525663468726	22556809934	35.33096	52.56681		-17.32372481	
525664394885	22556867961	34.30219	2.149148		33.09394185	
525665361640	22558307689	35.80574	53.32326		-18.08016926	
525666301687	22559695326	34.81656	51.39396		-16.15087296	
525667258825	22559754589	35.44956	2.194926		33.04816407	
525668198874	22561162517	34.81663	52.14548		-16.90239148	
525669154946	22562553758	35.41007	51.52744		-16.28435444	
525670103538	22562611965	35.13304	2.155815		33.08727519	
525671062814	22563984251	35.52874	50.82541		-15.58231741	
525672006067	22565434261	34.9353	53.70407	-18.46098407		
525672978162	22565494305	36.00352	2.223852	33.01923815		
525673925687	22566885017	35.09352	51.50785	-16.26476185		
525674871076	22568295678	35.01441	52.2467	-17.0036137		

Observation: Even with zero Shunra jitter we have a net jitter of ~20ms. We were suspecting parasitical jitter being setup in the datapath. We tried another test, by keeping in mind that this parasitical jitter could be compensated by ensuring an output bitrate comparable with the system capacity (20~30 ms).

Test2- IP streamer O/P selected was VBR encoding stream at increased rate of 33.7 Mbps (max.) The entire MUX was injected into the loop. In this test also, Shunra Emulator is out-of-loop i.e. no network jitter was introduced. Table 8.2 is shown for this test.

Table 8.2: Test2- AllVBR38 – No Shunra

PCR	STC	t(PCR) in ms= PCRdiff /27000	t(STC) in ms= STCdiff /27000	Mean	Deviation from mean	Average Jitter (ms)
636907070028	7545270888	-----	-----	28.66768	-----	6.153074
636907715242	7546060088	23.89681	29.22963		-0.56194963	
636908362595	7546574582	23.97604	19.05533		9.612346667	
636909372080	7547624778	37.38833	38.89615		-10.22846815	
636910022636	7548260477	24.09467	23.54441		5.123272593	
636910960549	7549171926	34.73752	33.75737		-5.08969037	
636911626060	7549823710	24.64856	24.14015		4.527531852	
636912250979	7550642096	23.14515	30.31059		-1.642912593	
636913261531	7551523270	37.42785	32.63607		-3.968394074	
636913904611	7552165987	23.81778	23.80433		4.863346667	
636914837181	7553049739	34.53963	32.73156		-4.063875556	
636915494148	7553764301	24.33211	26.46526		2.202420741	
636916151112	7554402071	24.332	23.62111		5.046568889	
636917164869	7555385459	37.54656	36.42178		-7.754097778	
636917788722	7556038792	23.10567	24.19752		4.470161481	
636918742658	7556952539	35.33096	33.84248		-5.174801481	
636919380395	7557617825	23.61989	24.64022		4.027457778	
636920023473	7558302218	23.8177	25.34789		3.319791111	
636921029753	7559232647	37.26963	34.46033		-5.792653333	
636921676036	7559910412	23.93641	25.10241		3.565272593	
636922632109	7560836775	35.41011	34.30974		-5.642060741	
636923266643	7561504669	23.50126	24.73681		3.930865185	
636923917199	7562189627	24.09467	25.36881		3.298865185	
636924564550	7562834708	23.97596	23.89189		4.775791111	
636925568694	7563820960	37.19052	36.52785		-7.860171852	
636926514083	7564787489	35.01441	35.79737		-7.12969037	
636927159299	7565440664	23.89689	24.19167		4.476013333	
636927822674	7566063133	24.56944	23.05441		5.613272593	
636928448662	7566688791	23.18474	23.17252		5.495161481	
636929466692	7567682335	37.70481	36.79793		-8.130245926	
636930100156	7568373662	23.46163	25.6047		3.062976296	
636931047681	7569263682	35.09352	32.9637		-4.296023704	
636931692898	7569953420	23.89693	25.54585	3.121828148		
636932338113	7570605937	23.89685	24.1673	4.500383704		

Observation: When the variable bitrate figures around/exceeds the system capacity, jitter observed is much less here. But some parasitical jitter is still induced during IP

packetization at DCM 9900 O/P. Another test were performed for CBR encoding stream rather VBR stream.

Test3- IP streamer's O/P selected was CBR encoding stream at lesser rate of 20 Mbps (max.) The stream of a sports channel is selected from the entire MUX and injected into the loop. In this test also, Shunra Emulator is out-of-loop .Table 8.3 is shown for this test.

Table 8.3: Test3- SportsCBR20 – No Shunra

PCR	STC	t(PCR) in ms= PCRdiff /27000	t(STC) in ms= STCdiff /27000	Mean	Deviation from mean	Jitter (ms)
525643484763	46467067203	-----	-----	35.2407	-----	3.328957
525644422808	46467909584	34.74241	31.1993		3.969953704	
525645403489	46468963793	36.32152	39.04478		-10.37709778	
525646329351	46469822616	34.29119	31.80826		-3.140579259	
525647295824	46470865556	35.7953	38.62741		-9.959727407	
525648231838	46471897871	34.66719	38.23389		-9.566208889	
525649184093	46472755236	35.2687	31.75426		-3.086579259	
525650136352	46473792628	35.26885	38.42193		-9.754245926	
525651088609	46474649100	35.26878	31.72119		-3.053505185	
525652026652	46475683326	34.74233	38.30467		-9.636986667	
525652978910	46476607270	35.26881	34.22015		-5.552468148	
525653931167	46477450335	35.26878	31.22463		-2.55694963	
525654885455	46478497628	35.344	38.78863		-10.12094963	
525655823499	46479341058	34.74237	31.23815		-2.570468148	
525656775757	46480388010	35.26881	38.776		-10.10832	
525657740196	46481238424	35.71996	31.49681		-2.829134815	
525658694486	46482286396	35.34407	38.81378		-10.14609778	
525659687351	46483343132	36.77278	39.13837		-10.47069037	
525660611183	46484193164	34.216	31.48267		-2.814986667	
525661577653	46485234730	35.79519	38.57652		-9.908838519	
525662529909	46486091318	35.26874	31.72548		-3.057801481	
525663482168	46487129551	35.26885	38.45307		-9.785394074	
525664405998	46487979778	34.21593	31.48989		-2.822208889	
525665372470	46489020796	35.79526	38.55622		-9.888542222	
525666310514	46489874132	34.74237	31.60504		-2.937357037	
525667262769	46490909970	35.2687	38.36437		-9.69669037	
525668202845	46491762750	34.81763	31.58444		-2.916764444	
525669167286	46492805364	35.72004	38.61533		-9.947653333	
525670107359	46493661440	34.81752	31.70652		-3.038838519	
525671071799	46494585178	35.72	34.21252		-5.544838519	
525672011874	46495620015	34.81759	38.3273		-9.659616296	
525672990528	46496487023	36.24644	32.11141	-3.443727407		
525673930601	46497518969	34.81752	38.22022	-9.552542222		
525674880829	46498378825	35.19363	31.84652	-3.178838519		

Observation: This was another confirmatory test in which even if the input bitrate is not upto the mark, maintaining a constant output bit rate equaling system capacity (using padding) compensates for the jitter due to IP packetization.

Test4 - DCM 9900 IP streamer O/P selected was CBR encoding stream at higher rate of 40 Mbps (max.) The stream of a sports channel (AV) is selected from the entire MUX and injected into the loop. Table 8.4 is shown for this test.

Table 8.4: Test4-SportsCBR40-NoShunra

PCR	STC	t(PCR) in ms= PCRDiff /27000	t(STC) in ms= STCDiff /27000	Mean	Deviation from mean	Jitter (ms)
525643480171	68216799887	-----	-----	35.24076	-----	1.661782
525644418216	68217770087	34.74241	35.93333		-0.692573333	
525645391791	68218707517	36.05833	34.71963		-6.05194963	
525646322729	68219646095	34.47919	34.76215		-6.094468148	
525647296306	68220641662	36.05841	36.87285		-8.205171852	
525648220138	68221552813	34.216	33.74633		-5.078653333	
525649179501	68222535200	35.53196	36.3847		-7.717023704	
525650131760	68223509727	35.26885	36.09359		-7.425912593	
525651084016	68224429515	35.26874	34.06622		-5.398542222	
525652022060	68225340648	34.74237	33.74567		-5.077986667	
525652967211	68226285682	35.00559	35.00126		-6.333579259	
525653926575	68227279308	35.532	36.80096		-8.133282963	
525654885940	68228238855	35.53204	35.53878		-6.871097778	
525655816876	68229138348	34.47911	33.31456		-4.646875556	
525656776240	68230127610	35.532	36.63933		-7.971653333	
525657735605	68231171679	35.53204	38.66922		-10.00154222	
525658694971	68232040105	35.53207	32.16393		-3.496245926	
525659682759	68232985299	36.58474	35.00719		-6.339505185	
525660606591	68233895986	34.216	33.72915		-5.061468148	
525661565953	68234906968	35.53193	37.44378		-8.776097778	
525662518211	68235838476	35.26881	34.5003		-5.832616296	
525663470470	68236826142	35.26885	36.58022		-7.912542222	
525664401406	68237726205	34.47911	33.33567		-4.667986667	
525665367878	68238711879	35.79526	36.50644		-7.838764444	
525666305921	68239633363	34.74233	34.12904		-5.461357037	
525667265283	68240617138	35.53193	36.43611		-7.768431111	
525668203329	68241520331	34.74244	33.45159		-4.783912593	
525669162694	68242519114	35.53204	36.99196		-8.324282963	
525670107844	68243429014	35.00556	33.7		-5.03232	
525671067207	68244382973	35.53196	35.33181		-6.664134815	
525672012358	68245324559	35.00559	34.87356		-6.205875556	
525672985936	68246335384	36.05844	37.43796		-8.770282963	
525673931086	68247286300	35.00556	35.21911	-6.551431111		
525674876237	68248190474	35.00559	33.48793	-4.820245926		

In this test also, Shurna Emulator is out-of-loop i.e. no network jitter was introduced.

Observation: Above test confirms that by increasing the output bit rate, IP packetization jitter could be minimized further.

After above observation (Table 8.1-Table 8.4), jitter was observed even without shurna emulator. Therefore, above conducted tests (Test-1 to Test-4) helped us to find out the causes of instability of new proposed algorithm. After analyzing whole system, we come-up with some interesting observations regarding whole IP data path. Although, issues in IP data path is not scope of our work, even a brief idea is mentioned below-

In the data path for IP streaming from Network to Transport Stream Merger, IP packets are intercepted by the Linux TCP/IP Stack which is a part of the kernel. There is a *skb pointer (socket buffer pointer) to each IP packets (1.5k max). A maximum of 7 TS packets are present in each IP packets. NET (Network driver) registered callback function “netsb_nethook()” is called by the Linux stack for each received IP packets. NET does the reordering, striping of headers and enqueue in an enqueue list. When injection granularity occurs reached a pre-defined value, NET driver was injecting the packet using Flexible Direct Memory Access to an intermediate buffer managed by application. Data flow from buffer to SWTS (Software Writable Transport Stream) was managed by a time scheduled polling task in PRM (Play Record Manager) application. Therefore, as explained above, this whole IP data path was not optimized. Therefore, some optimization was needed and has been done on application side. The application of optimization work is not under the scope of our thesis but was very critical and important for our testing. In view of above, after optimization in IP playback overall system, no parasitical jitter was observed and testing of new algorithm could started.

8.6.1 Matlab simulation with Real time data

Matlab simulation environment is used to calculate the value of FS & OD. It was analyzed that low value of FS causes instability. To handle large jitter, value of FS must be high. However, too high FS value resulted in loss of numerical precision. So, a suitable value depending on the jitter was chosen as a compromise. Thus, to determine the optimum value of Filter Strength (FS) and OD, data set of PCR and STC has been captured without correction in decoder clock. Because, with enabled clock recovery, value of STC is being continuously changed (as applying correction on decoder clock) , so that time STC values can not give true measure of jitter. Therefore, we captured PCR-STC data with optimized IP playback path in real time and calculated jitter as shown in Table 8.5 – Table 8.7. MATLAB simulation of our algorithm for these varying jittery streams is shown in Figure 8.12 –Figure 8.14.

8.6.1.1 Matlab Simulation of Real time data for 0 ms jitter

In this simulation, PCR- STC data set were captured in real time environment with zero jitter introduced by shunra emulator. PCR-STC data stream for 0 ms jitter is shown in Table 8.5-.

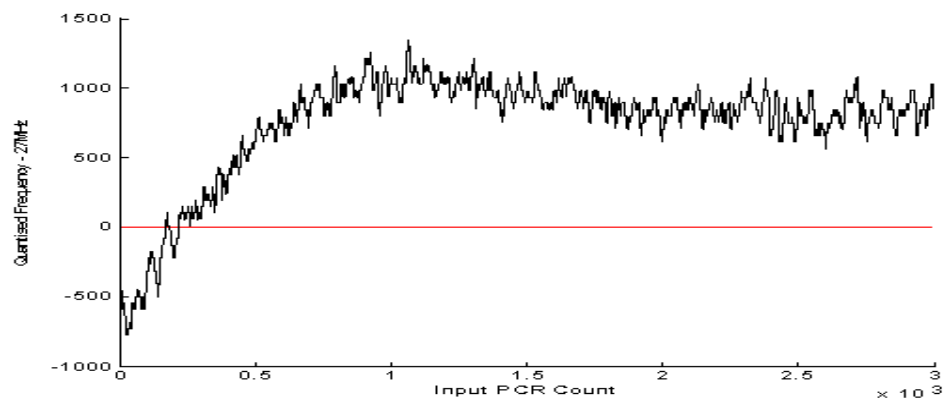


Figure 8.12: Plot of quantized decoder frequency for real time data having 0 ms jitter

Table 8.5: Real Time PCR- STC data set (0 ms jitter)

PCR	STC	t(PCR) in ms= PCRdiff /27000	t(STC) in ms= STCdiff /27000	Mean	Deviation from mean	Jitter (ms)
1081188	40053091022	-----	-----	3.999876419	-----	0.080957
2162376	40054162567	4.0044	3.968685		0.031191234	
3241026	40055270222	3.995	4.102426		-0.102549507	
4322214	40056334482	4.0044	3.941704		0.058172715	
5400864	40057414413	3.995	3.999744		0.000131975	
6482052	40058467772	4.0044	3.90133		0.09854679	
7560702	40059581505	3.995	4.124937		-0.125060618	
8641890	40060652826	4.0044	3.967856		0.032020864	
9720540	40061728009	3.995	3.982159		0.01771716	
10801728	40062823788	4.0044	4.058441		-0.058564322	
11880378	40063889774	3.995	3.948096		0.051780123	
12961566	40064956786	4.0044	3.951896		0.047980123	
14040216	40066068971	3.995	4.119204		-0.119327285	
15121404	40067135544	4.0044	3.95027		0.049606049	
16200054	40068231121	3.995	4.057693		-0.057816173	
17281242	40069269854	4.0044	3.847159		0.15271716	
18362430	40070381832	4.0044	4.118437		-0.118560618	
19441080	40071445159	3.995	3.938248		0.061628271	
20522268	40072512690	4.0044	3.953819		0.046057901	
21600918	40073628875	3.995	4.134019		-0.134142099	
22682106	40074691131	4.0044	3.934281		0.065594938	
23760756	40075755637	3.995	3.942615		0.057261604	
24841944	40076826062	4.0044	3.964537		0.035339382	
25920594	40077935702	3.995	4.109778		-0.109901359	
27001782	40079011128	4.0044	3.983059		0.01681716	
28080432	40080068790	3.995	3.917267		0.082609752	
29161620	40081181925	4.0044	4.122722		-0.122845803	
30240270	40082252399	3.995	3.964719		0.035157901	
31321458	40083313979	4.0044	3.931778		0.068098641	
32400108	40084426119	3.995	4.119037		-0.119160618	
33481296	40085492045	4.0044	3.947874		0.052002345	
34562484	40086557345	4.0044	3.945556		0.054320864	
35641134	40087643530	3.995	4.022907	-0.023030988		
36722322	40088738051	4.0044	3.968685	-0.053905062		

Jitter analysis of stream in Table 8.5, suggest that average jitter was 0.080957.

Above, PCR- STC data set that was captured in real time environment has been simulated

in matlab by choosing best suitable value of FS=3 and OD =15. Response of new algorithm in MATAB simulation environment is shown in Figure 8.12. From above graph it is clear that decoder clock is stabilizes at around $0.6 \times 10^3 = 600$ PCR samples i.e. $600 \times 40\text{ms} = 24$ seconds for 0 ms jitter.

8.6.1.2 Matlab Simulation of Real time data for 40 ms jitter

In this simulation, PCR- STC data set were captured in real time environment with maximum jitter of 40 ms introduced by shrunra emulator. PCR-STC data stream for 40 ms jitter is shown in Table 8.6-.

Table 8.6: Real Time PCR- STC data set (40 ms max jitter)

PCR	STC	t(PCR) in ms= PCRdiff /27000	t(STC) in ms= STCdiff /27000	Mean 39.99909	Deviation from mean	Average Jitter (ms)
147960324	7132848299					4.551915
149041512	7133888636	39.95	40.36926		-	
150120162	7134953738	40.044	38.531		1.468093405	
151201350	7136123904	39.95	39.44822		0.550871182	
152280000	7137405124	40.044	43.33948		-	
153361188	7138229562	39.95	47.45259		3.340388077	
154442376	7139651913	40.044	30.53474		-	
155521026	7140486974	40.044	52.67967		7.453499188	
156602214	7141498576	39.95	30.92819		9.464352664	
157680864	7142612677	40.044	37.46674		-	
158762052	7143676860	39.95	41.263		9.070908219	
159840702	7144750144	40.044	39.41419		2.532352664	
160921890	7145782611	39.95	39.75126		-	
					1.263906595	
					0.584908219	
					0.247834145	

162000540	7146867785	40.044	38.23952		1.759574886	
163081728	7147961987	39.95	40.19163		0.192536225	
164160378	7149080531	40.044	40.526		0.526906595	
165241566	7150145102	39.95	41.42756		1.428462151	
166320216	7151285703	40.044	39.42856		0.570537849	
167401404	7152299530	39.95	42.24448		2.245388077	
168480054	7153617576	40.044	37.54915		2.449945256	
169561242	7154718898	39.95	48.81652		8.817425114	
170642430	7155544093	40.044	40.7897		0.790610299	
171721080	7156581190	40.044	30.56278		9.436315627	
172802268	7157693749	39.95	38.411		1.588093405	
173880918	7158842571	40.044	41.20589		1.206795484	
174962106	7159829954	39.95	42.54896		2.549869558	
176040756	7161075792	40.044	36.56974		3.429352664	
177121944	7161978774	39.95	46.14215		6.143054744	
178200594	7163179286	40.044	33.44378		6.555315627	
179281782	7164211096	39.95	44.46341		4.464314003	
180360432	7165388841	40.044	38.21519		1.783908219	
181441620	7166355157	39.95	43.62019		3.621091781	
182520270	7167464461	40.044	35.78948		4.209611923	
183601458	7168656387	39.95	41.08533		1.086239929	
184680108	7169629271	40.044	44.14541		4.146314003	
185761296	7170897415	39.95	36.03274		3.966352664	
186842484	7171798529	40.044	46.9683		6.969202892	
187921134	7172888468	40.044	33.37459		6.624500812	
189002322	7173901215	39.95	40.36811		0.369017706	

190080972	7175045002	40.044	37.50915		2.489945256	
191162160	7176160453	39.95	42.36248		-	2.363388077
192240810	7177121898	40.044	41.313		-	1.313906595
193321998	7178589051	39.95	35.60907			4.390019331
194400648	7179399269	40.044	54.339			-14.3399066
195481836	7180416459	39.95	30.00807			9.991019331

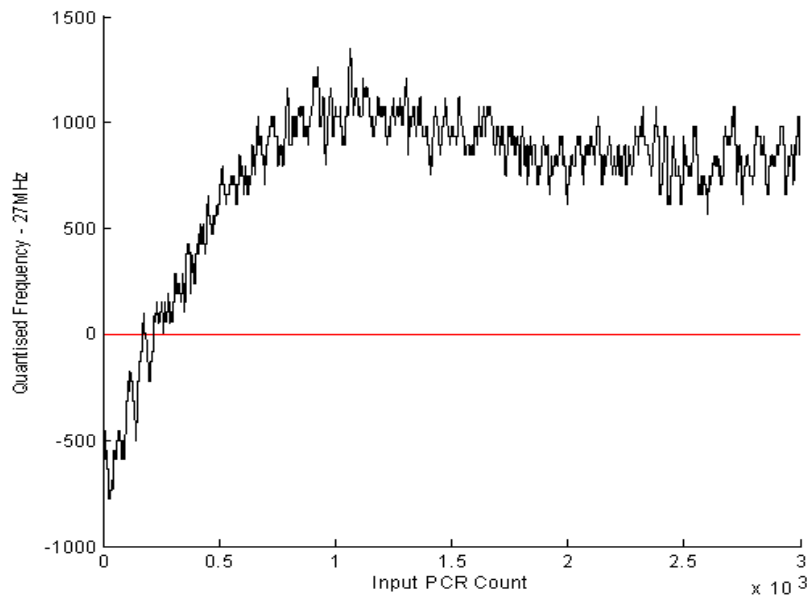


Figure 8.13: Plot of quantized decoder frequency in matlab for real time data having maximum jitter=40 ms

Jitter analysis of stream in Table 8.6, suggest that average jitter in this case was 4.551915. Above, PCR- STC data set for 40 ms jitter has been simulated in matlab by selecting best suitable value of FS=4 and OD =10. Response of new algorithm in matlab is shown in Figure 8.13. From above graph it is clear that decoder clock is stabilizes at around $0.8 \times 10^3 = 800$ PCR samples i.e. $800 \times 40\text{ms} = 32$ sec.

8.6.1.3 Matlab Simulation of Real time data for 100 ms jitter

In this simulation, PCR- STC data set were captured in real time environment with maximum jitter of 100 ms introduced by shunra emulator. PCR-STC data stream for 100 ms jitter is shown in Table 8.7-.

Table 8.7: Real Time PCR- STC data set (100 ms max jitter)

PCR	STC	t(PCR) in ms= PCRdiff /27000	t(STC) in ms= STCdiff /27000	Mean 43.65455096	Deviation from mean	Average Jitter (ms) 19.32083
704160486	17955298	-----	-----		4.055773177	
705241674	17957269	39.95	39.59878		21.7529954	
706320324	17962692	40.044	21.90156		-	
707401512	17965597	39.95	60.25204		11.37321762	
708480162	17970469	40.044	32.28133		-	
709561350	17973501	39.95	54.12437		10.46981942	
710640000	17975710	40.044	33.69578		9.958773177	
711721188	17982795	40.044	24.53852		19.11603244	
712802376	17984636	39.95	78.72007		-	
713881026	17987312	40.044	20.466		35.06552312	
714962214	17991032	40.044	78.72007		23.18855096	
716040864	17993093	39.95	29.73107		13.92347688	
717122052	18000294	40.044	41.32737		2.327180585	
718200702	18003548	39.95	22.90019		20.75436577	
719281890	18008654	40.044	80.01815		-	
722520378	18015703	39.95	36.14574		36.36359719	
723601566	18018548	40.044	56.73444		7.508810215	
724680216	18022955	119.944	78.33141		-	
		40.044	31.60593		13.07989349	
					-	
					34.67685645	
					12.04862503	
					-	
					5.309819415	

725761404	18029629	39.95	48.96437		-	30.50518979
726840054	18032514	40.044	74.15974			11.6056991
729002430	18037807	39.95	32.04885		-	15.15767127
730081080	18040362	80.088	58.81222			15.25721762
731162268	18044588	39.95	28.39733		-	3.301597193
732240918	18047618	40.044	46.95615			9.991143548
733322106	18052784	39.95	33.66341		-	13.74278238
734400756	18059136	40.044	57.39733		-	26.92007867
735481944	18061815	39.95	70.57463			13.88158799
736560594	18065685	40.044	29.77296			0.657662066
737641782	18067616	39.95	42.99689			22.19421762
738720432	18071310	40.044	21.46033			2.61036577
739801620	18077621	39.95	41.04419		-	26.45852312
740880270	18080045	40.044	70.11307			16.71177318
743040108	18086502	39.95	26.94278		-	28.08315275
744121296	18091540	79.994	71.7377		-	12.32926386
745202484	18097644	40.044	55.98381		-	24.16352312
747362322	18102212	40.044	67.81807		-	7.102671267
748440972	18104049	79.994	50.75722			23.24232873
750600810	18112238	39.95	20.41222		-	47.33563423
751681998	18114217	79.994	90.99019			21.67088429
752760648	18116672	40.044	21.98367			16.37114355
753841836	18119349	39.95	27.28341			13.91266207
754920486	18122830	40.044	29.74189			4.97736577
756001674	18127801	39.95	38.67719		-	11.58367127
757080324	18138370	40.044	55.23822		=	73.76878238

758161512	18140395	39.95	117.4233		4.055773177	
-----------	----------	-------	----------	--	-------------	--

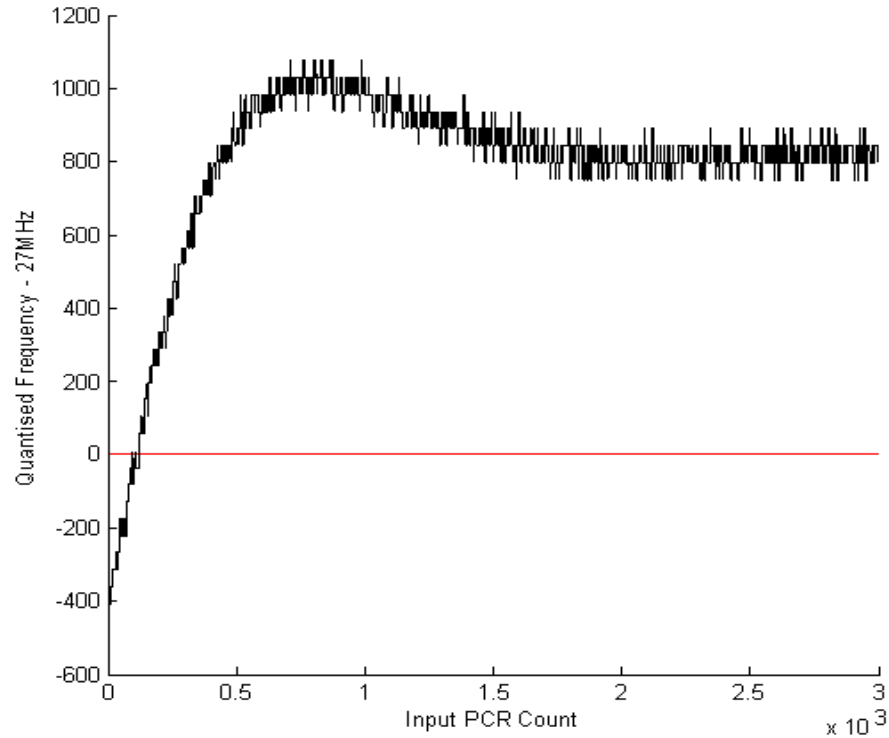


Figure 8.14: Plot of quantized decoder frequency in matlab for real data having maximum jitter=100ms

Jitter analysis of stream in above Table 8.7, suggest that average jitter in this case was 19.32083. Above, PCR- STC data set for 100 ms jitter has been simulated in matlab by selecting best suitable value of FS=6 and OD =8. Response of new algorithm in matlab is shown in Figure 8.14. From above graph in Figure 8.14 it is clear that decoder clock is stabilizes at around $1.2 \times 10^3 = 1200$ PCR samples i.e. $1200 \times 40\text{ms} = 48$ sec.

8.6.2 Results of Real time IP streams for varying jitter

As per set-up shown in Figure 8.11, the jittered IP stream is sent to the ST7109 SoC of IP-STB board. The Stream used was VBR stream with maximum speed of 33.7 Mbps. The jitter introduced using ‘Shunra’ was second order Pareto distribution with varying jitter. Our proposed enhanced LLR algorithm for clock recovery is being

implemented in ST7109 decoder chip. Response of clock recovery algorithm for varying jitters is shown in Figure 8.15 - Figure 8.17.

8.6.2.1 Results of Real time IP streams for 0 ms jitter

Graphical plot shown in Figure 8.15 is the response of decoder frequency with FS=4 and OD = 6 in 0 ms jitter in real time IP environment on IP Set-Top Box. In Figure 8.15, PCR counts and quantized decoder frequency is plotted. Our clock recovery algorithm is able to recover decoder clock in 26 seconds ($650 \times 40\text{ms} = 26\text{sec}$).

8.6.2.2 Results of Real time IP streams for 20 ms jitter

Graphical plot shown in Fig 8.16 is the response of decoder frequency with FS=4 and OD=10 in 20 ms jitter in real time IP environment. In Figure 8.16, PCR counts and quantized decoder frequency is plotted. Our clock recovery algorithm is able to recover decoder clock in 34 seconds ($850 \times 40\text{ms} = 34\text{ sec}$).

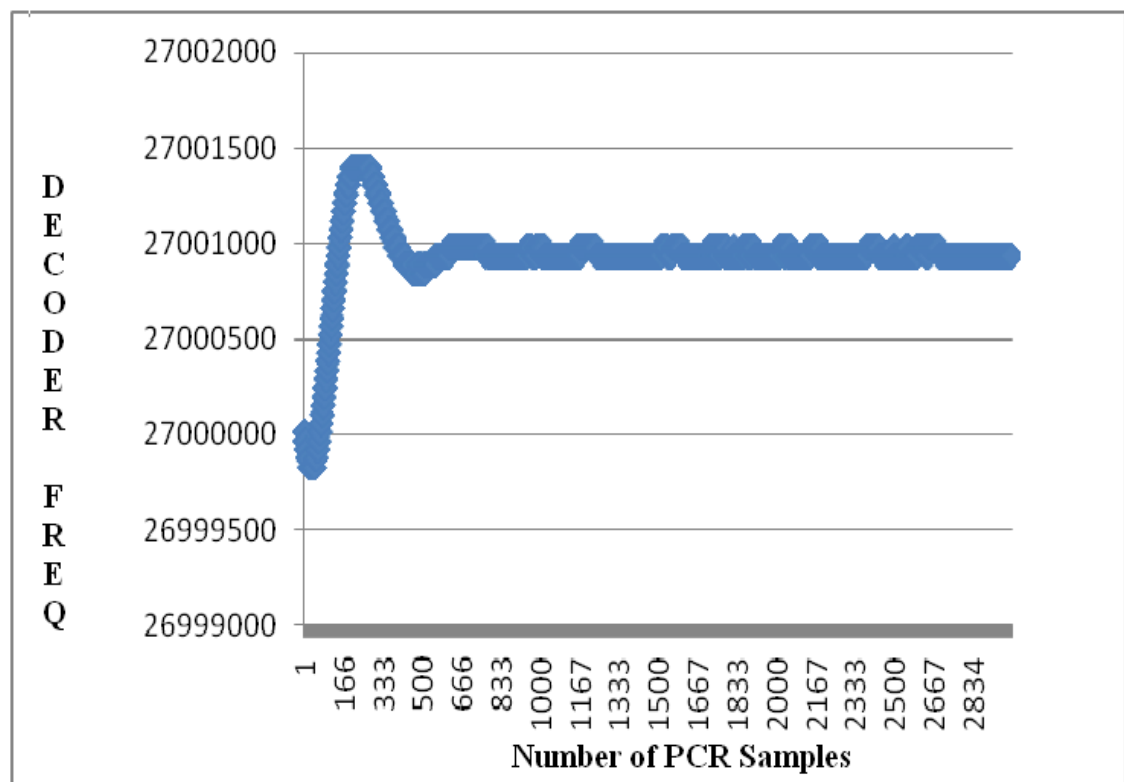


Figure 8.15 Plot of PCR count and quantized decoder frequency for 0 ms jitter in real time IP stream

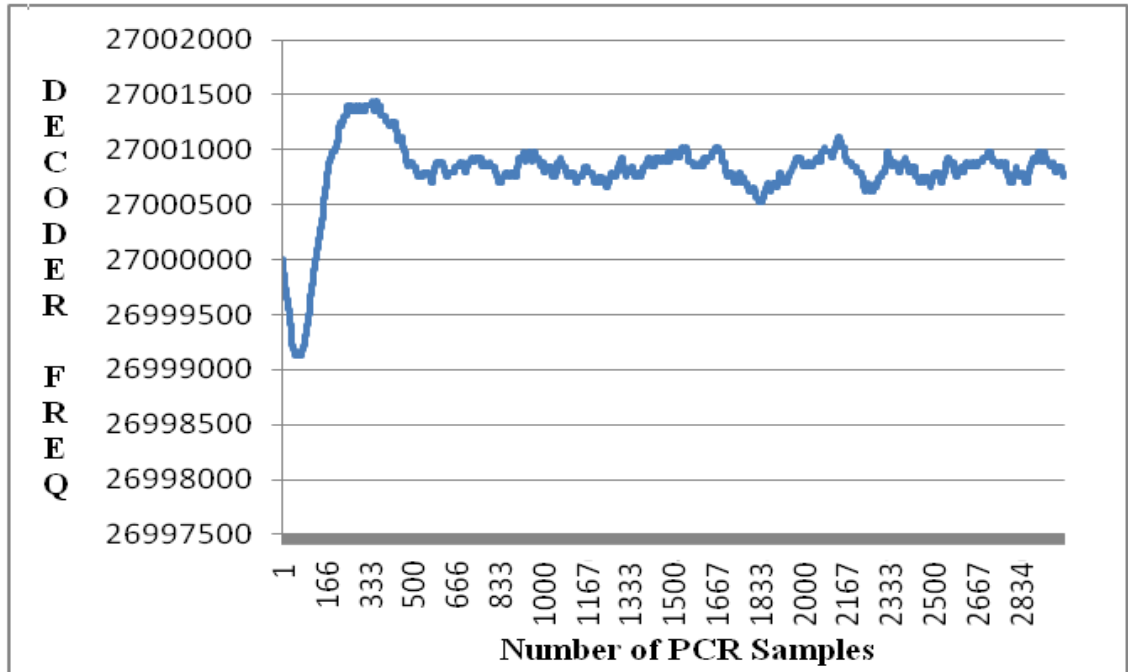


Figure 8.16 Plot of PCR count and quantized decoder frequency for 20 ms jitter in real time IP stream

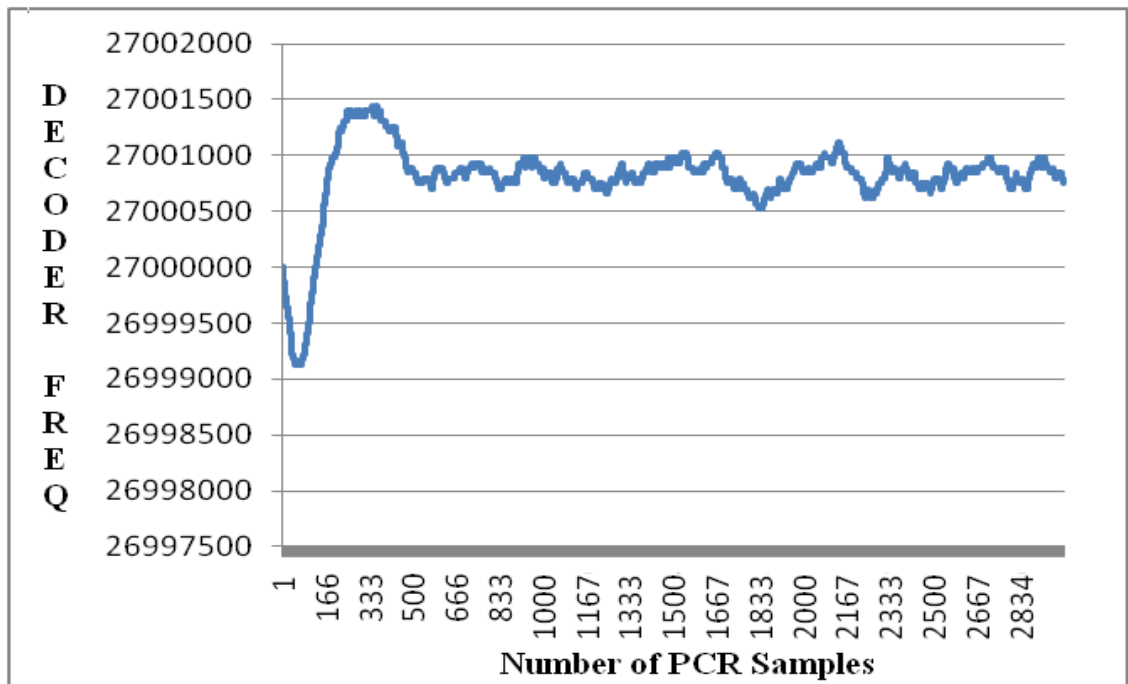


Figure 8.17 Plot of PCR count and quantized decoder frequency for 50 ms jitter in real time IP stream

8.6.2.3 Results of Real time IP streams for 50 ms jitter

Graphical plot shown in Figure 8.17 is the response of decoder frequency with FS = 6 and OD= 10 in 50 ms jitter in real time IP environment. In above Figure 8.17, PCR counts and quantized decoder frequency is plotted. Our clock recovery algorithm is able to recover decoder clock in 38 seconds ($950 \times 40\text{ms} = 38 \text{ sec}$).

8.6.2.4 Results of Real time IP streams for 80 ms jitter

Graphical plot shown in Figure 8.18 is the response of decoder frequency with FS = 8 and OD=10 in 80 ms jitter in real time IP environment. In above Figure 8.18, PCR counts and quantized decoder frequency is plotted. Our clock recovery algorithm is able to recover decoder clock in 44 seconds. ($1100 \times 40\text{ms} = 44 \text{ sec}$).

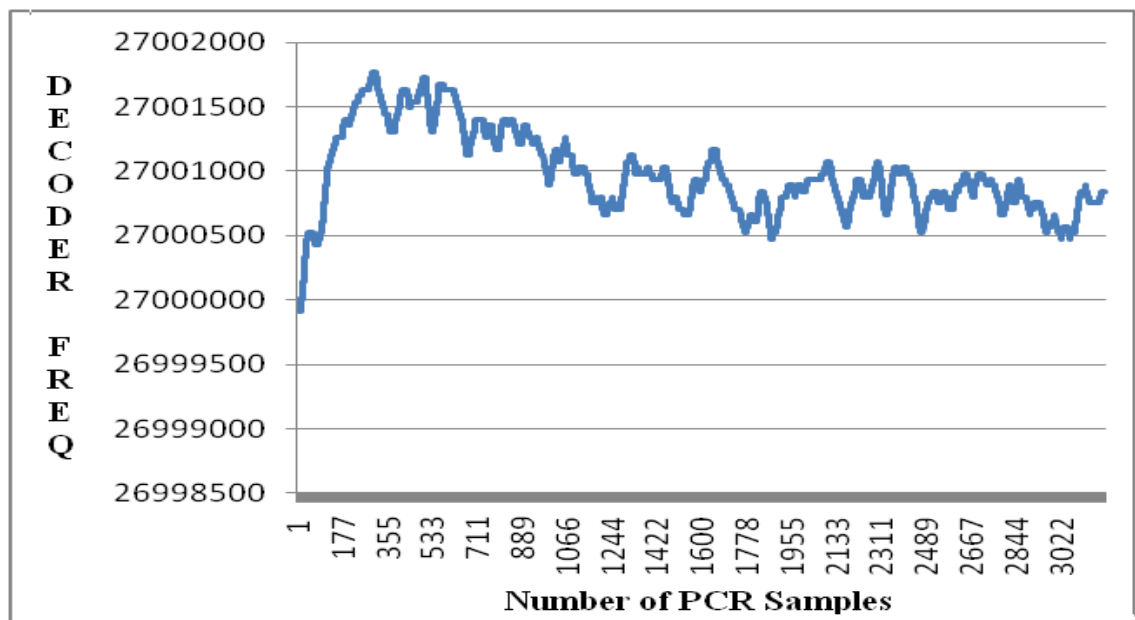


Figure 8.18 Plot of PCR count and quantized decoder frequency for 80 ms jitter in real time IP environment

8.6.2.5 Results of Real time IP streams for 100 ms jitter

Graphical plot shown in Figure 8.19 is the response of decoder frequency with FS=9 and OD=8 in 100 ms jitter in real time IP environment.

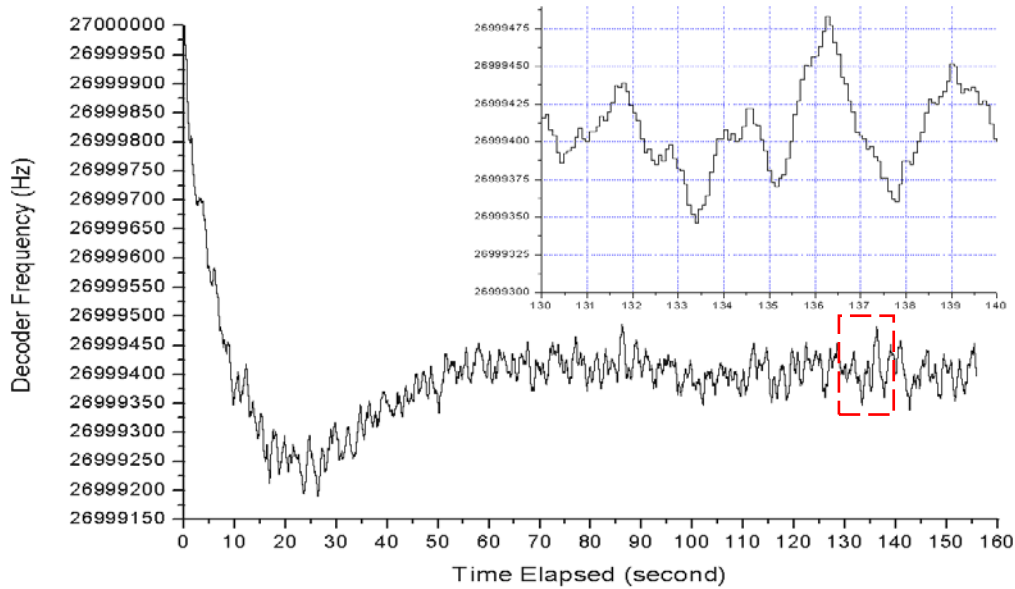


Figure 8.19 Plot of Decoder Frequency vs time elapsed in a VBR stream 2-nd order pareto 100 ms jittery environment (FS =9).

In above Figure 8.19, PCR counts and time elapsed is plotted. Our clock recovery algorithm is able to recover decoder clock in 50 seconds. However, due to a large amount of jitter, the clock frequency varies around 100 Hz. The maximum jump is shown in inset. Clearly, the frequency drift rate is about 100Hz per second, which is lesser than the maximum drift rate of 6.7 KHz per second as discussed in section 8.1.

8.7 PERFORMANCE COMPARISON OF NEW PROPOSED LR ALGORITHM WITH OLD LR ALGORITHM

Further, we compared our algorithm with old method as shown in Figure 8.20. The new algorithm was substituted by old LR algorithm described. Figure 8.20 compares the performance of both the algorithms when subjected to a jitter of 100 ms. A circular

buffer was maintained instead of continuous adaptation feedback loop.

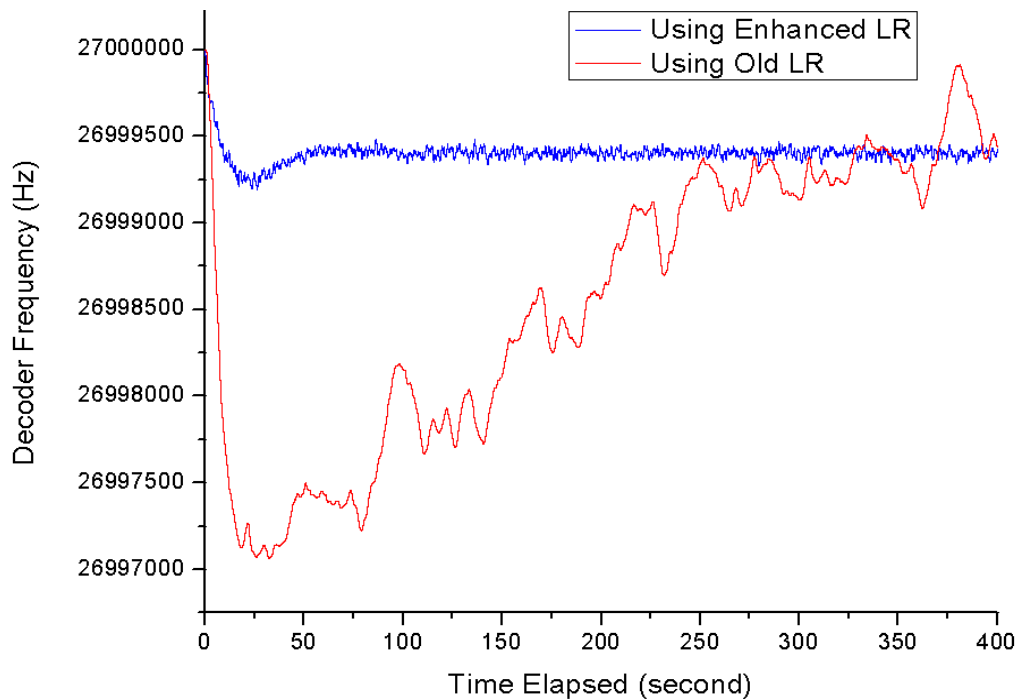


Figure 8.20 Performance comparisons of old LR algorithm with new enhanced LR algorithm for VBR (max 33.7 Mbps) 2-nd order pareto jitter distribution.

Color loss is observed when viewed on TV. Some frame-skips were also observed as the decoder is unable to track the frequency of the encoder clock. This was probably due to more time taken by this algorithm to maintain the buffer and apply correction to the local clock. This limitation has also overcome by choosing suitable value of FS. The plot clearly suggests that original LLR algorithm is not stable in RTOS constraints. However, our algorithm gives an estimate of the higher jitter attenuation and faster response time. Table 8.8 shows the relative settling times of LMS, LLR and enhanced LLR algorithms. FS values corresponding to the broadcast stream for the enhanced algorithm are also shown. Table 8.8 shows the performance comparisons of these algorithms when subjected to low jitter environment to high jitter environment. Results clearly suggest that while LMS & LLR fails to sustain in high jittery environment, with

our designed Enhanced LLR implementation no frame skip or color loss has been observed.

TABLE 8.8 Performance Comparison of OLD LLR and enhanced LLR

<u>Broadcast medium</u>	<u>Peak Jitter (ms)</u>	<u>Settling Time (sec)</u>			<u>FS</u>
		<u>LMS</u>	<u>LLR[3]</u>	<u>New LLR</u>	
<u>DVB-S</u>	<u>0.1</u>	<u>24</u>	<u>18</u>	<u>15</u>	<u>2</u>
<u>DVB-T</u>	<u>0.5</u>	<u>60</u>	<u>30</u>	<u>25</u>	<u>4</u>
<u>DVB-IPTV</u>	<u>100</u>	<u>300</u>	<u>250</u>	<u>50</u>	<u>7</u>

8.8 CONCLUSION

In this chapter large jitter in IP environment has been analyzed. Difficulties in older algorithms are explored and new algorithm has been presented. Superiority of newly developed algorithm over other has been illustrated with various results and graph. The clock stabilizes in around 50 seconds only even with 100 ms jitter, displaying high jitter attenuation and fast response Furthermore, no color loss or audio-video frame skips were observed under these jittery conditions. The stability of the algorithm in RTOS environment follows from the fact that audio-video decoding went smoothly using the algorithm.

Chapter 9: Audio-Video Synchronization

In this chapter, detailed analysis of Audio Video Synchronization (AV Sync) module has been done. Audio-Video Synchronization has always been proved difficult to implement in STB. This chapter is motivated by a review of existing implementation of Audio video sync software in Set-top box, explores the problem and provides some possible solutions. A simple implementation that in principle offers better synchronization accuracy than can currently be achieved is a key objective.

9.1 INTRODUCTION TO AUDIO VIDEO SYNCHRONIZATION (AV SYNC)

As explored in earlier chapters, Clock Synchronization is an important part of the MPEG standard and ensures smooth operation of the MPEG decoder. It is important to mention here that clock synchronization have two concepts Clock recovery and AV sync-

- Clock recovery at STB decoder-end to recover the encoder clock so that audio and video streams are played at the same speed at which they are processed at encoder side. It also ensures that audio and video bit buffers (BB) should not suffer from underflow or overflow. This aspect have been addressed in chapter 5 to chapter 8.
- AV sync is synchronization mutually between audio and video. Accurate AVsync (Lip-Sync) is mandatory for correct user experience as TV viewer expects to hear and see matching lips movement in the picture. In this chapter, issue of AV sync problem for dual video STB and possible solutions has been presented.

The clock recovery and lip Sync mechanism are needed individually or simultaneously depending on STB use cases as shown in Figure 9.1 to Figure 9.5.

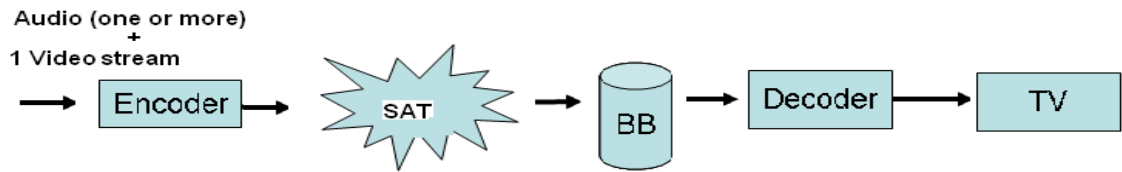


Figure 9.1 Standard MPEG system: Full clock recovery is mandatory with PCR, Lip Sync with PTS/STC is also needed

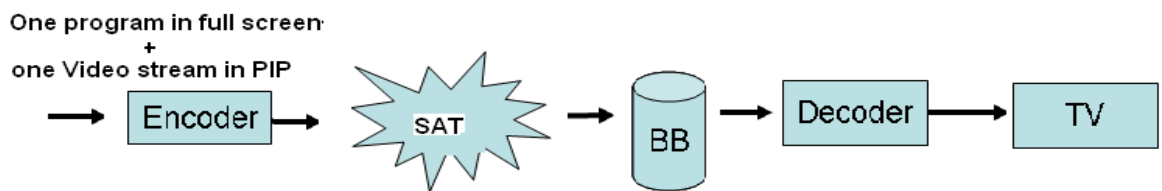


Figure 9.2 One program in Full screen+ PIP (Picture In Picture): Full clock recovery + Lip sync with PTS/STC for full screen program, PIP is either free running or video synced

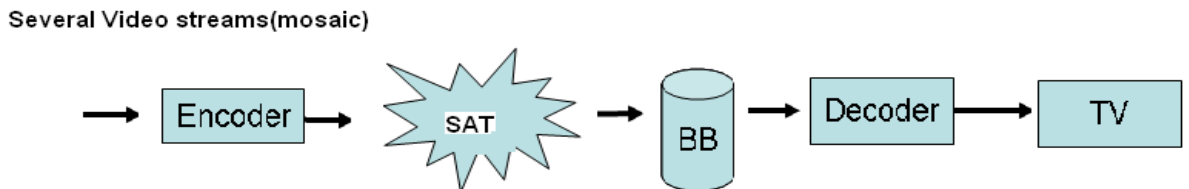


Figure 9.3 Video only (Mosaic): clock recovery is mandatory, but Lip Sync is not needed

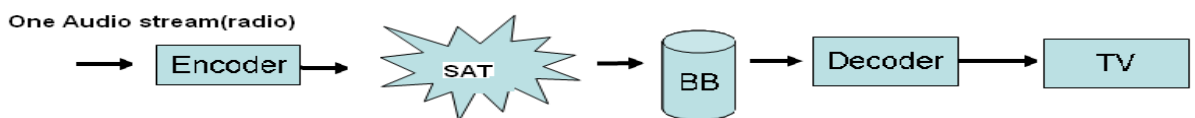


Figure 9.4 Audio only: Full clock recovery is mandatory, but Lip Sync is not needed

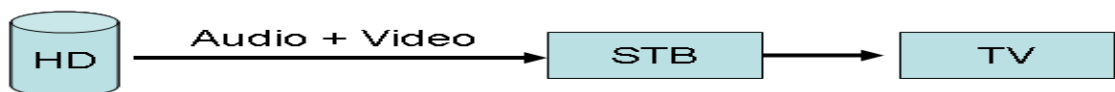


Figure 9.5 DVR (recorded system): clock recovery is not needed, but Lip-Sync is mandatory

Table 9.0 STB Customer’s Audio Video Sync Requirements

STB customer	Audio delays in milli seconds with respect to video
Premier	-110 to +35
Echo Star	-20 to +40
DirecTV(DHMC)	-10 to +30
DirecTV(others)	-20 to +20

The main reason of AV Sync problem is difference in processing path of audio and video. Simultaneous synchronization of an HD and SD output poses even tighter constraints. Various models for correct AV Sync implementation has been suggested by researchers in the similar field. However, none of them addresses the AV Sync problem of dual video (HD & SD simultaneously) with audio. STB has not been considered as possible application for their implementation so such models are not suitable for Real Time set-top box application. In this chapter detailed analysis of various STB applications scenarios and their tight constraints has been done. The current AV Sync implementations of STB have been analyzed and certain enhancements in current software implementations have been proposed. Few novel synchronization models that offer better synchronization accuracy has also been discussed. Audio-Video Synchronization is an important part of MPEG (Moving Picture Expert Group) standard (MPEG-System). It ensures the smooth operation of MPEG decoder. The current methods of software implementation of audio and video impose serious problems on audio and video synchronization. In this chapter certain software enhancements in existing approaches have been proposed.

9.2 AV SYNCHRONIZATION DEFINITION

There are several factors that affect the AV synchronization. The fundamental mechanism of AV sync is to present the audio video content when STC (System time Clock) matches the PTS of a specific audio sample or a specific video frame. Typically software polling is used for determining the current STC.

PCR (Program Clock Reference) is the snapshot of encoder clock and it is found in header of TS (Transport Stream). TS packets contain different audio and video PES (Packetized Elementary Stream) packet identified by Packet ID (PID). PES is made from ES (Elementary Stream) and PES header contains PTS (Presentation Time Stamps). PESs are grouped (e.g., two audio streams+ one video stream) to make a program. To assist the decoder, TS provides PCRs on one of the PID of the program (Video PES). PTS is encoder STC time stamp of the picture to be displayed (or audio frame played).

The PTS references this master clock and specifies when the picture must be displayed on the screen or audio frame to be played on the speakers. STC (System Time Clock) is a hardware clock of decoder that is reconstructed from PCR. Typically software polling is used for determining the current STC. Required action on reaching the required or desired STC is handled by software, which is subject to execution variation (due to other threads executing in the system).

In video the PTS of a frame is not correlated to the vertical synchronization signal, therefore, even if everything else is totally accurate there could be a worst-case presentation delay equivalent to video frame duration. For 29.97Hz NTSC this delay is 33.3667 ms, for 25Hz PAL this delay is 40ms and for 30Hz (High Definition) it is 33.3333 ms. Recent STB chips (STi 7109) have both, a 30 Hz High Definition (HD) and 29.97 Hz Standard Definition (SD) video output. There is normally one HD decoder and

each SD frame is created by down-sampling the HD frame. The audio now has to synchronize simultaneously with the two different output frame rates.

Audio errors with respect to slower video rate are shown in Figure 9.6. The audio is perfectly synchronized with the fast video. Initially the audio is also (coincidentally) in sync. with the slower video. However as time progresses, the audio becomes earlier and earlier wrt. the slower video, until the slower stream drops a frame. Now, the audio is later than the slow video. The audio then becomes less late, until eventually it will be back in synchronization, and the cycle will start again i.e. audio now has to ‘stay close enough’ to two different video frames at all points in time.

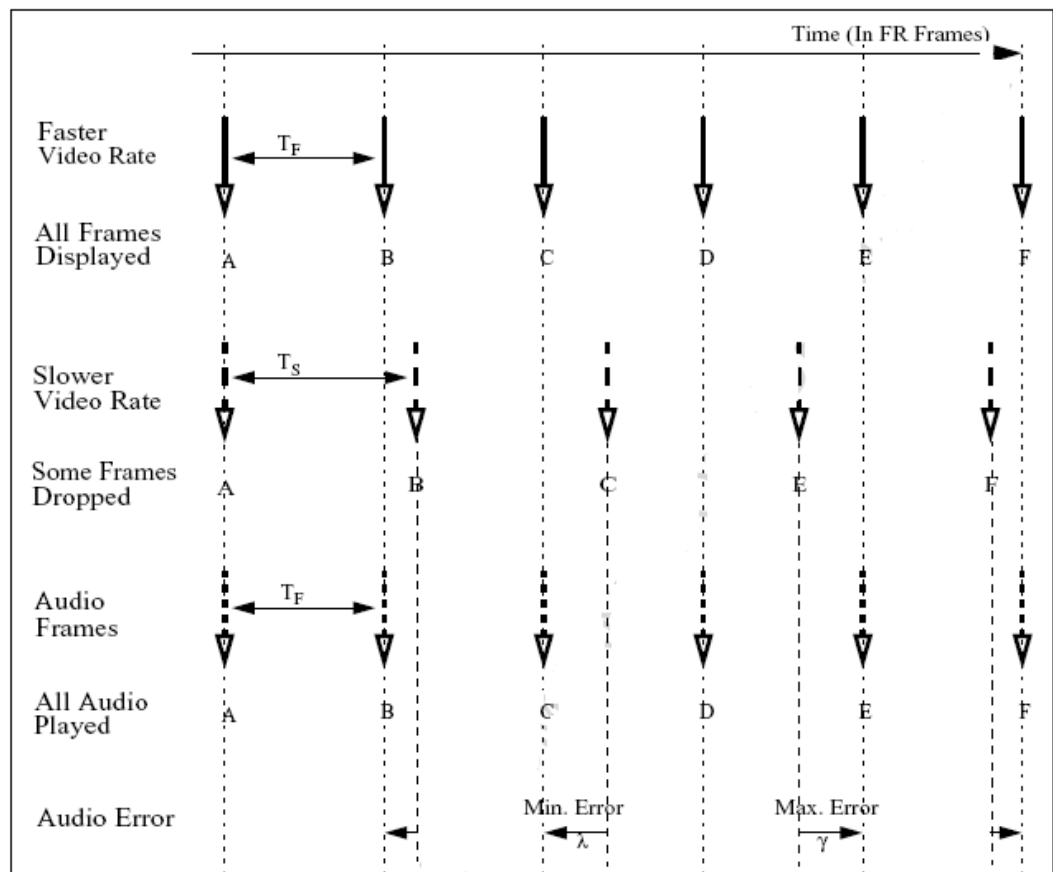


Figure 9.6 Audio errors with respect to slower video rate

Let's consider in Figure 9.6 at a point in time where the fast video and slow video both display frames simultaneously, this occurs for frame A at the left hand edge of the diagram. As time progresses, the frames displayed at the slower rate are displayed later and later with respect to the faster stream. There comes a point when the slow stream drops a frame in order to keep in step. In the diagram the slow stream drops frame **D**.

Consider, the audio in Figure 9.6, for simplicity, it is assumed that the audio frames are clocked at the same rate and have the same duration as the fast video. The audio is perfectly synchronized with the time a frame of fast video appears on the monitor. Initially the audio is also (coincidentally) in synchronization with the slower video. However as time progresses, the audio becomes earlier with respect to slower video until the slower stream drops a frame. Now the audio is later than the slower video. The audio then becomes less late, until eventually it will be back in synchronization, and the cycle will start again, audio now has to 'stay close enough' to two different video frames at all points in time. With respect to frames of slow video, the audio frames are played earlier up to frame **C**. At this point the slow video is λ ms later than the audio. This is the maximum video delay allowed, and a frame must be dropped. Frame **D** of the slow video is dropped, and frame **E** is next to be displayed. Now the audio is γ ms later than the slow video. This is the maximum audio delay allowed, and the audio delay will now become shorter and shorter. Expressions for evaluating λ and γ can be calculated. Let T_F & T_S are time interval for faster and slower frame rate, T_E specifies AV Sync error window, T_M is max. specified AV Sync delay, γ is max. audio delay with respect to slow video and λ is max. slow video delay with respect to the audio. Consider the point just after a frame drops; this is when the audio delay is maximum with respect to slow video. This delay is γ .

To confirm to the AV Sync specification

$$\gamma < T_M \quad (1)$$

Consider the point just before a frame drop, this is when the slow video delay is maximum with respect to audio. This delay is λ . For AV Sync specification we require-

$$\lambda < (T_E - T_M) \quad (2)$$

Let ε be the margin by which both (1) and (2) are satisfied. Then we have the two equalities-

$$\gamma + \varepsilon = T_M \quad (3)$$

$$\lambda + \varepsilon = T_E - T_M \quad (4)$$

Eliminating ε from (3) and (4) gives-

$$T_M - \gamma = T_E - T_M - \lambda \quad (5)$$

From Fig. 6 we also have

$$2T_F = \lambda + T_S + \lambda + \gamma \quad (6)$$

Equation (5) and (6) are simultaneous equations in λ and γ , solving them gives-

$$\gamma = T_F + T_M - \frac{(T_E + T_S)}{2} \quad (7)$$

$$\lambda = T_F - T_M + \frac{(T_E - T_S)}{2} \quad (8)$$

The safety margin by which the specification is satisfied is ε ms or a window of opportunity 2ε wide. From (3) we get the safety margin-

$$\varepsilon = T_M - \gamma \quad (9)$$

Substituting for γ from (7)

$$\varepsilon = \frac{(T_E + T_S)}{2} - T_F \quad (10)$$

For synchronization to be within specification we require ε to be greater than zero. From (10) we get the following restriction for AV Sync error window specification T_E -

$$T_E > 2T_F - T_S \quad (11)$$

As per DTV (DirecTV) specification in Table 9.0, We have

$$T_F = 33.33 \text{ ms}, T_S = 33.34 \text{ ms}, T_E = 40 \text{ ms and } T_M = 30 \text{ ms}$$

By substituting above values in equations (7), (8), (10) and (11) we get $\varepsilon = 3.35 \text{ ms}$, $\lambda = 6.65 \text{ ms}$, $\gamma = 26.65 \text{ ms}$, $T_E > 33.30 \text{ ms}$. As T_E is 40 ms i.e. > 33.3 , DTV specification can be satisfied if audio is synchronized to the video with an accuracy of 3.35 ms , i.e. audio must be placed in a window of opportunity that is 6.7 ms wide. This accuracy is not currently achievable in existing STB AV Sync applications so some design optimization in existing application software design implementations have been proposed in next section.

9.3 PROPOSED ENHANCEMENTS IN EXISTING SCHEMES

Some design optimizations to improve AV sync accuracies in existing STB have been proposed and described as below-

9.3.1 Improving timing accuracy

9.3.1.1 Timing improvements by local clock tick quantization

The typical RTOS (Real Time Operating System) such as OS21 uses 64 micro second clock ticks. These are thought to be sufficiently accurate. However, in Linux these clock ticks are 1 ms , which is not sufficiently accurate. Linux api `gettimeofday ()` should be used that returns time to a precision of 48KHz .

9.3.1.2 Timing improvements by correct interrupt handling

The audio and video software needs to ensure that the local clock is inspected closer to detection of event in the software. This reduces the possibility of code being preempted before the local time and the STC are captured.

9.3.1.3 Timing improvements by software scheduling optimization

The AV Sync sub-system software should be written as a suite of real-time functions. It competes with other time-critical sub-systems for time on the processor, so actual time that a piece of software runs can not be guaranteed. It is possible to alter the priorities of tasks in the scheduling queue. The critical AVSync functions should be given a high priority, and should not be preempted. The OS21 scheduler should not have any quantization issues. However, the Linux scheduler only inspects the list of queued processes every millisecond. This is a major risk. To minimize this effect, it is recommended that the Linux scheduler is called as infrequently as possible. This requires the AV Sync code for audio and video to be written to have the minimum number of threads, callbacks, and semaphores.

9.3.2 Improving quantization accuracy

9.3.2.1 PTS Quantization

In existing audio and video implementation PTS time stamp is quantized to the nearest milliseconds, however, it is available from the stream with 90 kHz accuracy (11.11 micro seconds). This can lead to significant improvement in AV Sync calculations.

9.3.2.2 Arithmetic Quantization

Normally lots of calculations in software for AV sync implementation truncated to the nearest millisecond. All software should use the same accuracy as the PTS i.e. 90 kHz accuracy. Rounding is preferred to truncation, although if truncation is required to improve speed performance, this should only introduce errors of about 11 μ s.

9.3.2.3 Audio Quantization

In audio software audio is skipped on a frame basis. This introduces timing errors of up to 32 ms. The audio software needs to skip samples on the output (as well as frames on input). If the audio can be delayed or advanced by a single sample at the output, then it can be adjusted with an accuracy equal to the sample rate, e.g. at 48kHz sample rate, the audio timing quantization will be 20.83 micro sec. Consequently audio quantization should not be a problem.

9.3.2.4 Video Quantization

Video frames can be either skipped or dropped. Although the finer quantization of video fields exists, it is not practical to skip or drop just one field as this creates interlacing artifacts. A top and bottom field must be displayed as a pair. Therefore video quantization is carried over one frame. At a frame rate of 25 Hz this will introduce a timing error of up to 40 ms, this is the major source of timing error for AVSync. Hence, adjustment for AV Sync error cannot be done by skipping or dropping video frames to meet AV Sync requirement. The audio software must perform the fine (sample-rate based) adjustment.

9.4 PROPOSED AV SYNC DESIGN VARIATIONS

This section suggests a number of AVSync design implementations and contains a more in-depth analysis of the display timings. In all implementations, the timing relationship between the decoded frames, the audio frames and the SR VTG (Video Time Generator) always remains the same. It is only the placement of the FR (Fast Rate) VTG that changes. This complicates the implementation as in this scenario VTG of the FR video drives the synchronization.

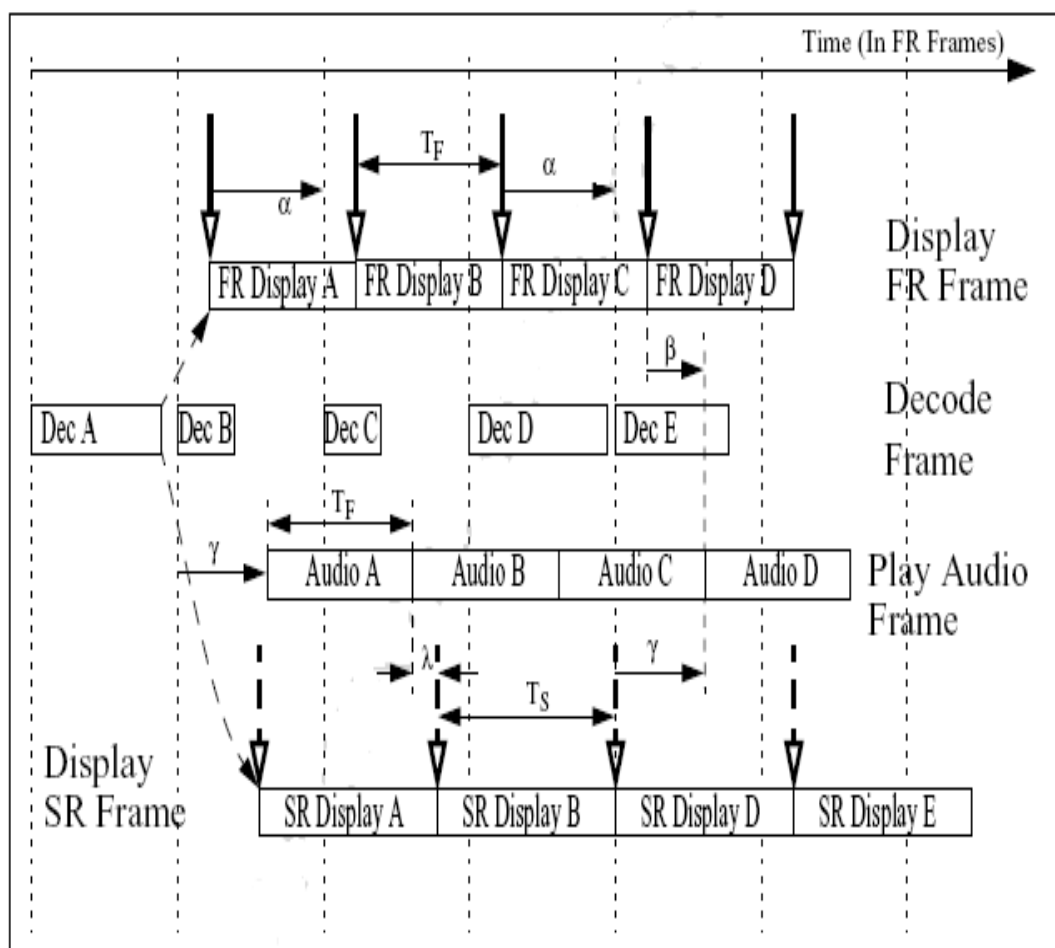


Figure 9.7 General Display Timing

General display timings are shown in Figure 9.7, the video decoder will take a variable amount of time to process a frame depending on picture content. This is represented by a

variable size decode block. Sometimes a decoded picture would be ready for SR (Slow Rate) display slightly early. However, for synchronization purposes the worst case decode time must be allowed for, and the worst case decode time is a whole FR frame interval. SR frame B has the longest delay before being displayed. This frame has to be buffered for two whole FR frames intervals. This is one more FR frame buffer than the case for a single video stream. It is easier to understand the synchronization relative to the decode frame intervals. These are shown as vertical dotted lines. In all the following options the audio will be delayed by γ ms with respect to start of the previous frame decode interval. However, the decode interval is in turn delayed by α ms from the previous FR VTG signal. The value for α can vary in the range $0 \leq \alpha \leq T_F$.

A number of special case values for α will be considered below. The audio will be delayed with respect to FR video by β ms. The choice of α determines β according to the following relationship (from frame C)

$$\alpha + \gamma = T_F + \beta \quad (12)$$

Substituting for γ from (7), gives -

$$\alpha - \beta = \frac{(T_E + T_S)}{2} - T_M \quad (13)$$

For the FR video to satisfy the AVSync specification we must have

$$(T_M - T_E) < \beta < T_M \quad (14)$$

Substituting for β from (13), and re-arranging, gives the corresponding limits on α -

$$\frac{(T_S - T_E)}{2} < \alpha < \frac{(T_S + T_E)}{2} \quad (15)$$

However if we require the whole range of α values to be possible, $0 \leq \alpha \leq T_F$. This places the following limits on the error window (T_E).

$$T_E > T_S \quad (16)$$

$$(T_E > 2.T_F - T_S) \quad (17)$$

By definition, $T_F < T_S$ so this simplifies to the single condition $T_E > T_S$. The most demanding specification in Table 9.1 is for DirecTV suggests: $T_S = 33.34$ ms, $T_E = 40$ ms. Therefore all the specifications in Table 9.1 are satisfied.

Now three main variations to above synchronization strategy are possible. The variants will be considered by how early or late an FR video frame is displayed compared to each SR video frame, starting with the variation where the FR frame is delayed the most-

- FR frame never ends early than any SR frame. (Minimize control complexity)
- FR frame perfectly synchronized with the audio frame. (Maximize FR Quality)
- FR frame never starts later than any SR frame. (Minimize memory requirements)

9.4.1 Minimize Control Complexity

In this model, as shown in Figure 9.8 the FR stream is delayed by a whole frame such that the FR frame is always the last to be displayed i.e. the end of the FR frame display is synchronized with the end of the latest SR frame display ($\alpha = 0$). Consequently, when the FR display is finished with a decoder buffer, the SR display will also finish with the same decoder buffer. Furthermore, the re-use of a decoder buffer can be synchronized with the FR VTG, as there is a one-to-one relationship between the number of decoded frames and the number of FR frames displayed. The SR video is

displayed as soon as possible after a decode interval ends, and the audio is delayed by the optimum amount γ .

As shown in Figure 9.8, the decode delay α is zero, so a decoder can be triggered directly by the FR VTG and no extra timing control is required. From (13) we get the following expression for the audio delay with respect to FR Video β

$$\beta = T_M - (T_E + T_S) / 2 \quad (18)$$

By substituting, $T_S = 33.34$ ms, $T_E = 40$ ms, $T_M = 30$ ms in (18), we get $\beta = - 6.68$ ms.

The audio delay with respect to FR video is negative, implying the audio is in advance of the video. Comparing this to the results for the SR video in Section 9.2, this is slightly worse than the maximum audio advance for the SR video ($\lambda = 6.65$ ms).

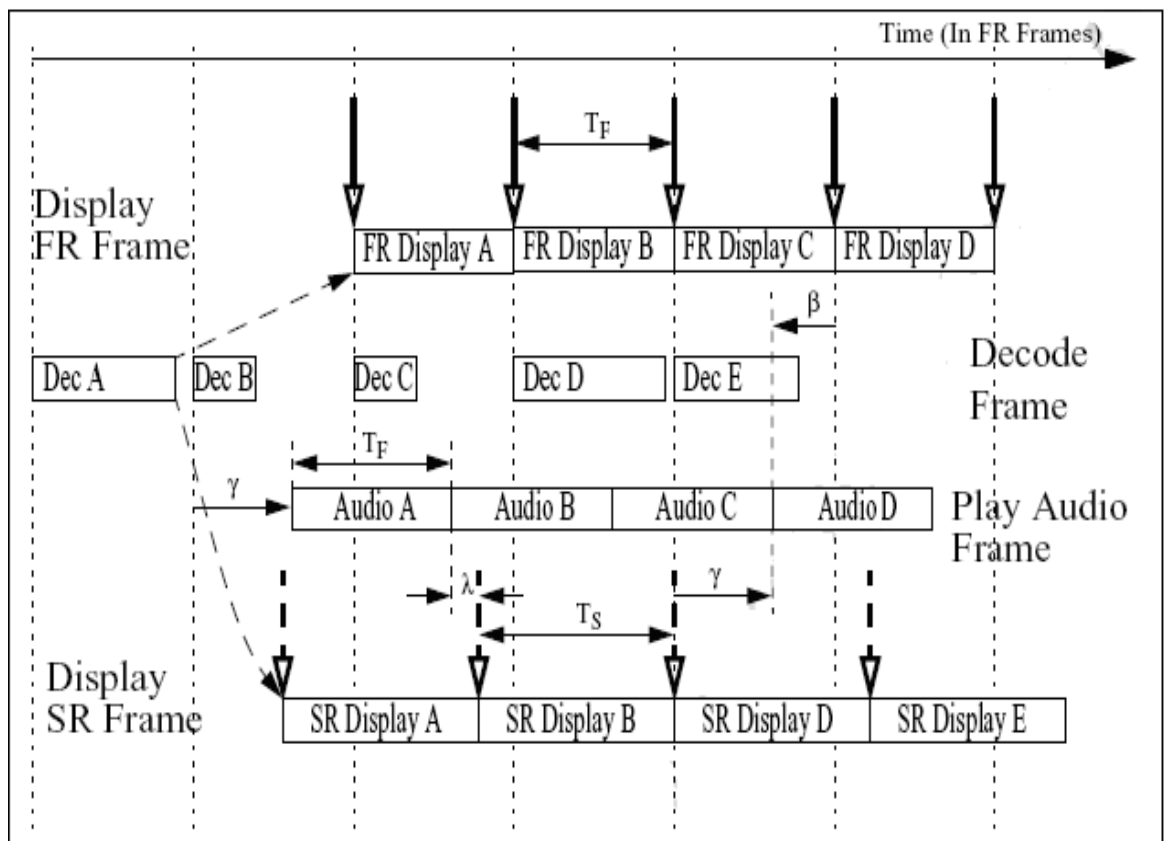


Figure 9.8 Minimize control complexity

9.4.2 Minimize High Quality Error

This option is used to preserve the high quality nature of the faster rate (FR) video stream. As shown in Figure 9.9 all the AVSync error is designed to occur between the audio and the slower rate (SR) video stream. Figure 9.9 suggest that value of α is chosen so that the audio delay with respect to FR video β is zero.

From (13), we get the following expression for the FR video delay with respect to decode interval α

$$\alpha = \frac{(T_E + T_S)}{2} - T_M \quad (19)$$

By substituting, $T_S = 33.34$ ms, $T_E = 40$ ms, $T_M = 30$ ms in (19), we get $\alpha = - 6.68$ ms i.e. the FR video should be synchronized in 6.68 ms after the last decode interval.

The software implementation for this scheme is also same as explained in minimize control complexity. FR frame will be displayed on FR VTG signal, however, the decoded buffer pointer must now be arranged such that the FR display uses the most recently filled decoded buffer.

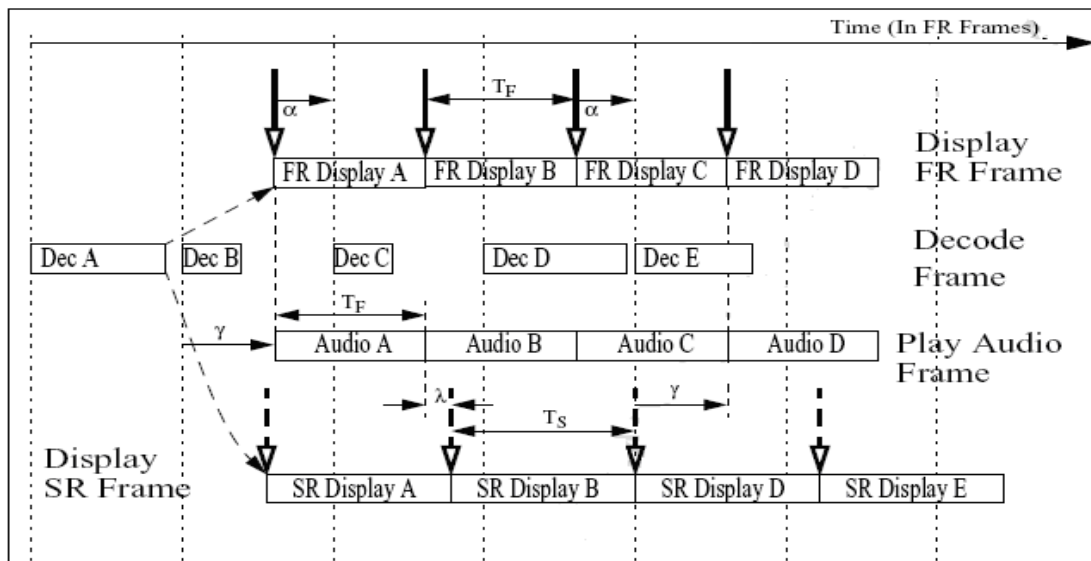


Figure 9.9 Minimize FR Error

9.4.3 Minimize Memory Requirements

This option is used to release the decoder buffers as soon as possible. The worst case will still require an extra decoder buffer, but statistically this will be required less often.

As shown in Figure 9.10, the FR display is started at the first decode interval ($\alpha = T_F$).

From (13), we get the following expression for the audio delay with respect to FR video β

$$\beta = T_F + T_M - (T_E + T_S) / 2 \quad (20)$$

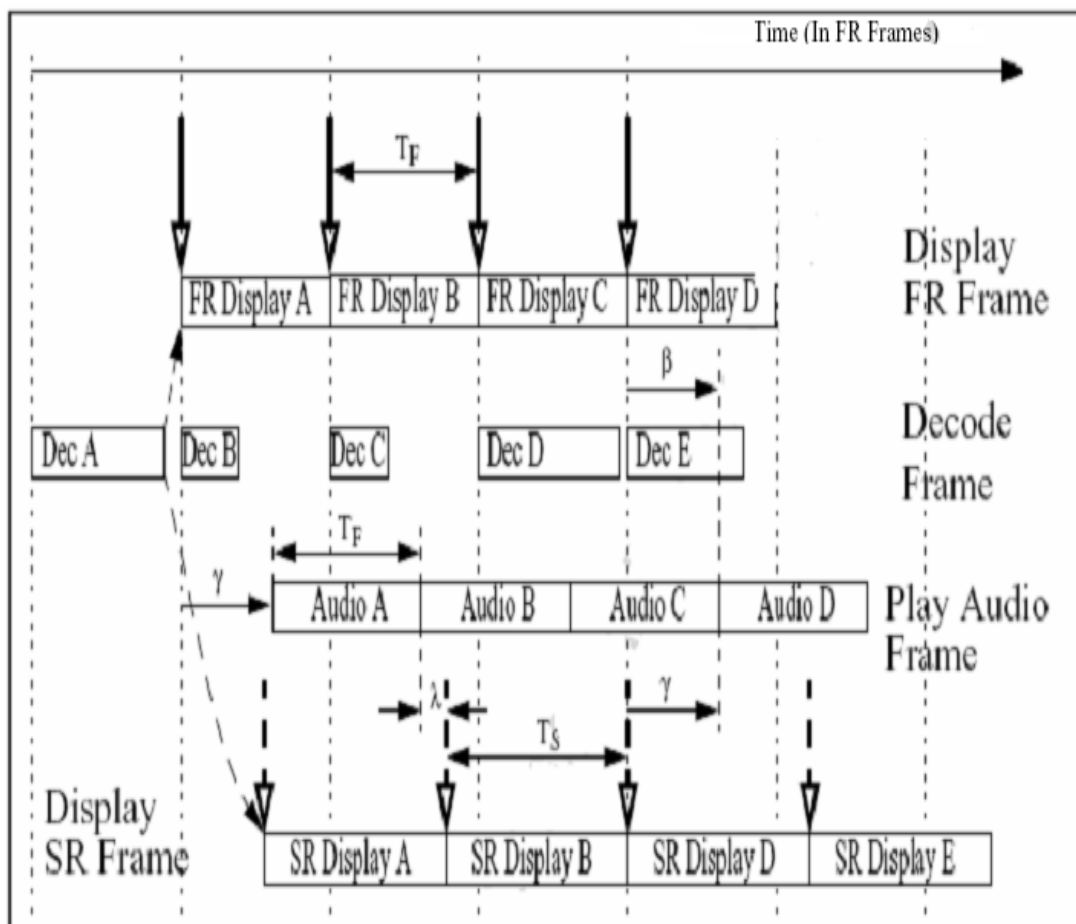


Figure 9.10 Minimizing memory requirement

This is the same result as derived for maximum audio delay with respect to SR video γ in (7). Thus, $\beta = \gamma = 26.65$ ms, i.e. audio plays 26.65 ms later than FR video. The software implementation for this scheme is also same as explained in minimize control complexity model, except that decoded buffer pointer must now be arranged such that the FR display uses the decoder buffer that has just been filled. But this would not minimize the memory requirement; we need to add more control complexity. The method to improve memory usage exploits the fact that FR and SR display overlap in time i.e. both will start displaying before either has finished.(except at frame drop).

Any of the above three schemes can be easily implemented in STB as per customer requirement (in terms of memory, complexity, quality of FR video) by altering the audio and video software implementation accordingly.

9.5 TEST SET-UP FOR AV SYNC PROBLEM

The Enhancements suggested in this thesis is being tested as per test setup shown in Figure 9.11. The live DVB stream has been played using packet injector and terrestrial modulator and fed to DVB-T front-end. The transport stream TS from demodulator and tuner block is sent to demux block. Demux block de-multiplexes the transport stream and convert it into the PES. The clock recovery mechanism implemented in STB provides the timestamps to the video and audio software drivers. Audio and Video software drivers may be redesigned according to recommendations suggested in section 5.4.

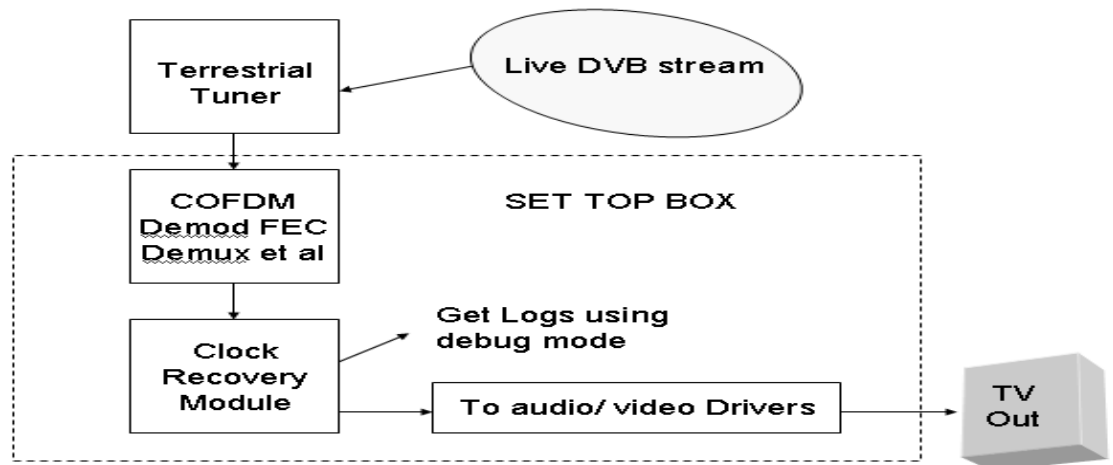


Figure 9.11 Audio Video synchronization testing setup

Performance evaluation of a prototype STB suggest that with proposed enhancements AV sync has been achieved in 20 ms. This shows significant improvement as compared to old conventional method that takes nearly 28 ms.

9.6 CONCLUSION

For future STB products, a much tighter AV Sync specifications are required. Suggestions proposed in this paper shows significant improvements when compared with existing implementations. Advance data compression technologies such as Dolby Digital (AC3) for audio and MPEG-2 for video provide viewers high quality multimedia experience with the help of STB. Lip sync problem can be observed if AV synchronization specifications specified are not properly met. Hence, an efficient solution to meet these AVSync requirements has been proposed. The enhancements and AV synchronization schemes suggested in this chapter have shown significant improvements in AV Sync timings.

Chapter 10: Results and Conclusions

In digital TV, the receiving system i.e. STB decoder must be able to recover clock which was embedded as PCRs in transport stream at encoder side. Real time transmission of audio, video and data may take place through broadcasting medium such as satellite, terrestrial and IP. Apart from well known difficulties in clock recovery at decoder, another issue in Digital TV (tight DTV) is due to the fact that set-top box decoder has to provide very accurate color sub-carrier frequency synchronization as decoder need to provide a output of color visual signal compatible with TV set (e.g., PAL and NTSC). Therefore, a need is felt for a clock recovery system that overcomes the clock synchronization problems in the presence of jitter in digital video broadcasting typically DVB-S, DVB-T and DVB-IP.

Another aspect of clock synchronization is audio-video synchronization (AV sync). AV sync has always proved difficult to implement in STB. The main reason of AV Sync problem is difference in processing path of audio and video. Simultaneous synchronization of audio with HD (High Definition) and SD (Standard Definition) output impose even more constraints.

In this chapter, summary of results and conclusions of the complete research work are presented. The salient features of the work are highlighted. Scope for further work is also presented.

10.1 SUMMARY OF RESULTS

In this thesis, suitable clock recovery algorithm based on PLL design for satellite and Terrestrial set-top box has been presented. Continuous adaption based linear regression algorithm for IP network also has been proposed. Detailed review of existing

AV sync implementation for digital Set-Top box has been done and certain enhancement in existing AV sync implementations have been proposed.

Test results using real time **set-top box in satellite environment** is shown in chapter-5. In satellite transmission, error in the encoder side clock is tracked very accurately. This has been achieved with selection of filter parameters such as $\text{MinSampleNum} = 10$, $\text{MaxWindowSize} = 50$ and $\text{PCRDriftThresh} = 300$, decoder clock is stabilizes in 20 seconds. Our designed algorithm shows good response as it is meeting all constraints of real time STB and achieving clock recovery efficiently. Less recovery time, very simple, and very much cost effective are unique features of our proposed algorithm.

Digital Video Broadcasting (DVB) transmission over terrestrial medium introduces high jitter in the stream (max jitter of the order of 1ms). In DVB-T, loss of synchronization between the received and the transmitted clocks can lead to degradation in system performance like color loss, jerky video, audio dropouts etc. It has been tested and simulated with already existing algorithms, which are not able to handle jittery environment of terrestrial broadcasting. Lot many terrestrial streams have been analyzed to identify the limitations in terrestrial streams. This analysis has helped us to propose a suitable moving window weighting algorithm for clock recovery for terrestrial set-top box. For 0.4 ms jitter (Guard Interval (G.I.) = $1/8$, FEC = $3/4$ and modulation technique = 64 QAM) algorithm has adapted to $M=150$, $S = 50$ and $P = 10000$. The clock stabilises in about 80 sec. The variation in the clock frequency after stabilising is 1 step only and it shows good stability over a long time. The plot confirms with the MATLAB simulations too. For worst case jitter 1.2ms (Guard Interval (GI)= $1/4$, FEC = $7/8$, and modulation technique = 64QAM) the algorithm has adapted to $M = 300$, $S = 100$ and $P= 30000$ for the worst-case terrestrial conditions. The clock settles in about 150 sec with reasonable

stability. Furthermore, no data buffer overflow or underflow was observed and the television viewing went smoothly. It may seem that the settling time of the decoder clock is large, however, it is not so. If the correction to the decoder side is applied in a single step, the settling time may reduce, but the sharp frequency change in the decoder clock may cause colour loss.

IP networks are asynchronous by nature. Even consecutive packets of the same size from the same server to the same receiver will have a variable transit time, depending on the rest of the traffic on the network. Such a variation of the average packet transit time between server and receiver is called network induced jitter. Jitter on IP networks is often of the order of tens of milliseconds, sometimes several hundreds of milliseconds. Moreover, in STB application, to prevent color loss the local frequency overshoot should be less than 1350 Hz. Secondly, larger step changes in frequency are likely to upset the PLL circuitry in a TV monitor and cause a visible 'colour shift'. Therefore, the system stability, jitter attenuation, latency introduced by the algorithm, maximum frequency overshoot, and drift rate of the decoder clock are critical parameters in clock recovery module for IP –STB.

In weighting filter, ordinary PLLs are used to dejitter MPEG-2 stream, which uses the PCR timestamps in the MPEG-2 Systems layer. PLL based approaches work fine for clock recovery in satellite and Terrestrial environment where jitter was comparatively less as in IP environment. In PLL based simulations it is assumed that the MPEG-2 stream is delivered over a network with small delay variations, and therefore uses a peak-to-peak jitter (delay variation) up to 1 ms, however, magnitude of delay variations in these regions are regarded as very low in IP-based network where these delay variations can be of the order of few hundred of milli seconds. PLL based schemes are therefore not suited for IP networks.

Linear regression (LR) based approaches suggested by Noro (2000) have been explored and modified so that it can be utilized in IP environment. It was found that old LR algorithm uses a buffer to store the various data points, introducing latency in the decoding process. It reduces the feasibility of using this algorithm in RTOS constraints. To eliminate the need of storing PCR data for LR algorithm, the enhancement in existing approach is achieved by applying exponential weighting parameters.

Different statistical distributions of jitter were considered to simulate real-time IPTV environment. The algorithm was first simulated in MATLAB simulation environment to analyze its stability. The simulations were also used to find the optimum value of Filter Strength (FS) for various Jitter values. It was analyzed that low value of FS causes instability. To handle large jitter, value of FS must be high. However, too high FS value resulted in loss of numerical precision. So, a suitable value depending on the jitter was chosen as a compromise.

IP environment thus selected for testing. In our analysis, we found that Pareto-2 (Weisstein - Pareto, 2009), jitter distribution was found to be close to real time IP environment so same jitter distribution has been selected for testing in Matlab simulation environment and real time testing. Although the jitter values are selected from a statistical distribution, the actual value chosen is determined by a uniform random number generator. The random generator can be seeded to produce repeatable test data. It has been verified that for Pareto-2 jitter distribution with maximum delays in PCR equals to 18 ms, 35ms, 52 ms and 70 ms, decoder clock has been recovered in 20 seconds, 24 seconds, 35 seconds and 50 seconds respectively using new LR algorithm based on continuous adaption. Algorithm parameter value such as FS (Filter Strength) have been determined in matlab simulation environment by just changing the FS parameter while running and selecting the most suitable value for specific jitter.

In next step PCR-STC data was taken from live IP VBR stream with artificially introduced jitter using 'Shunra' network emulator. To determine the optimum value of FS and OD, data set of PCR and STC has been captured without correction in decoder clock. Because, with enabled clock recovery, value of STC is being continuously changed (as applying correction on decoder clock), so that time STC values can not give true measure of jitter.

Initial analysis suggested that IP playback path was itself generating parasitical jitter. After certain optimization in IP playback overall system, no parasitical jitter was observed and testing of new algorithm could start.

PCR- STC data with optimized IP playback path in real time has been captured for various jitter values. For 0, 40, and 100 ms jitter values, the decoder clock stabilizes in 24, 32, and 48 seconds respectively in MATLAB simulation.

Now newly developed algorithm has been tested on STB. For maximum jitter values of 0, 20, 50, 80, and 100 ms, clock recovery algorithm is able to recover decoder clock in 26, 34, 38, 44, and 50 seconds respectively.

Audio-Video synchronization has always proved difficult to implement in STB. Various models for Correct AV Sync implementation has been suggested by researchers. However, none of them addresses the AV Sync problem of dual video (HD & SD simultaneously) with audio. Moreover, STB has not been considered as possible application for their implementation so such models are not suitable for Real Time set-top box application. There was strong requirement for detailed analysis of various STB applications scenarios, their tight constraints and achieving optimization in AV synchronization.

The current AV Sync implementations of STB have been analyzed and certain new implementations have been proposed under the scope of research work presented in

this thesis. Certain enhancements in existing approaches have been proposed. We have also proposed some new schemes for AV sync for digital STB in this thesis. The enhancements suggested in this paper are incorporated in a prototype STB system. Audio and Video software drivers have been redesigned according to the recommendations suggested. Results suggest that proposed enhanced solution provide significant improvement as compared to conventional method. Enhanced solution provides AV sync implementation in the range of 20 ms (audio delay w.r.t. Video =20 ms), while in conventional method AV sync implementation is in the range of 28 ms.

10.2 CONCLUSIONS

In this thesis, new, low cost, and low complex software approach for clock recovery in DVB transmission has been adopted. Firstly, constraints due to RTOS environment in set-top box application have been identified. Analysis of jitter for DVB transmission in satellite, terrestrial and IP environment was utmost important. Further, constraints offered by TV receiver limitations for existing clock recovery algorithm has been understood.

In this research work, we discussed the clock recovery requirement for typical set-top box system. One light weighted, low cost algorithm for DVB-S has been proposed. Proposed module is based on implementation of basic averaging, moving window and low pass filtering. Experimental results show that our clock recovery algorithm is working perfectly in the satellite environment.

Problem of clock synchronization in the jittery terrestrial environment has been addressed in this research work. Proposed module does not involve complex computations compared to the suggested algorithms by researchers in the similar field, rather it is based on implementation of low pass filtering, moving window, applying

weight and averaging of error samples for our enhanced FIR filter. The module also benefits in its simplicity, economy and less CPU time usage. This approach has handled all the limitations and complications of terrestrial broadcasting networks in a smart and efficient manner. Results are illustrated for real time environment in STBs. With our proposed algorithm, decoder clock is synchronized with encoder clock in 16.13 seconds for 250 microseconds jitter, however, LR algorithm takes 20.17 seconds for the same. In another case, with our proposed algorithm, decoder clock is synchronized with encoder clock in 21.93 seconds for 500 microseconds jitter; however, LR algorithm takes 29.17 seconds for the same. The new algorithm tracks the encoder clock accurately and no audio-video data packets loss is observed. Even with a jitter of 1.2 ms, no color loss is observed using this algorithm when viewed on Television and the decoder clock settles in about 100 seconds. Therefore, new proposed weighing algorithm shows fast response in low as well as high jitter environment as compared to LR algorithms. In addition, the proposed weighing algorithm does not suffer from significant overshoots, which are present in LR algorithms. Weighing algorithm will in turn, eliminate the need for additional damping procedures to overcome these overshoots. The simulation results and the actual testing both proved the novelty in the algorithm (stable in nature) and require very less resources for an RTOS environment.

10.3 SPECIFIC CONTRIBUTIONS

Our research work is mainly focused on achieving clock recovery for low jittery satellite broadcasting and moderate to high jittery Terrestrial broadcasting for digital set-top box. Major contribution is to achieve clock recovery for very high jittery IP broadcasting in STB. A/V sync problem in digital set-top box for dual video STB is another significant scope of our work.

Specific contributions of the research work are:

1. A new “Moving Window Basic Averaging filter” for clock recovery in digital set-top box for Satellite broadcasting has been presented.
2. Terrestrial medium is much more jittery as compared to satellite. PCR packets get delayed and we get large +/- errors proportional to the delay and very frequent glitches. It leads to incorrect synchronization and color loss. The clock recovery problem in terrestrial environment has been analyzed. Limitations of existing basic averaging algorithms applied in this area have been identified. Sometimes color loss has been observed in very high jittery environments. Our proposed Moving window weighting filter approaches provide an efficient, real time software solutions which offer low computational complexity, low cost and stable clock recovery meeting all constraints posed by set-top boxes for terrestrial environment.
3. The major achievement of our work was to develop efficient algorithm for IP environment. We have designed a new synchronization algorithm to perform efficient synchronization of two dispersed clocks for IPTV environment. The amount of jitter in IP environment is drastically high compared to DVB-S or DVB-T. The detailed study has been done for understanding the limitations of existing work available in field of packet switching network. With Continuous Adaptation Enhancements in Linear Regression Algorithm, we minimize the end-to-end delay and hence the loss caused by synchronization errors. The optimal performance of our new implementation of LLR enables the deployment of methodology adopted in the present thesis in digital TV applications based on the MPEG-2 standard over a wider range of jitter inducing networks, where the conventional second-order PLLs generally fail.

4. Another specific contribution of the thesis is in area of AV sync error. Audio-video Synchronization (AV sync) is very important to consumer and the display industry since newer technologies have created a noticeable delay between the processing of video signals and the processing of audio signals. Lip sync correction algorithms take into account processing delays, so that both signals can be synchronized and presented to the viewer together. Correct A/V sync greatly improves the entertainment for the viewer. In a typical set-top box A/V sync (lip-sync) has always proved to be difficult. Simultaneous synchronization of audio with HD (High Definition) and SD (Standard Definition) output impose tighter constraints. The current AV Sync implementations of Set-top box have been analyzed and certain enhancements in existing approaches have been proposed. We have also proposed some new schemes for AV sync for digital STB in this thesis.

10.4 FURTHER SCOPE OF WORK

The clock synchronization algorithm described in the present work that has been tested for satellite, terrestrial and IP digital set-top boxes can also be adapted to other multimedia applications. However, depending upon exact application of clock recovery module, certain modifications or enhancements might be needed.

The transmission medium, whether Internet, satellite or terrestrial, is subject to variable delays. However, currently, the most challenging medium is the Internet. We have proposed a solution for 100 ms jitter in IP network. Recently, maximum values for jitter of 700 ms have been reported which is much higher as compared to the typical value of 20ms according to the DVB-IPI . This standard specifies only the standard deviation of the network jitter. The analysis done in this thesis, suggests that a more important

parameter is the maximum network delay. This directly impacts on the amount of buffering required and color loss. When stream of max jitter of 700ms need to be handled, a large amount of incoming data being not 'well behaved' would be a serious problem. Because, a packet of 700 ms peak jitter is arriving with large delay. Therefore a PCR packet with this much long delay may well hold up lot many other later PCR packets. Hence, a PCR with a long delay would commonly be followed by a burst of later PCR packets that were queued behind it. Also, large number of unusual spikes can occur in the PCR tick values due to large transmission errors, in this scenario.

Thus, identifying the anomalous PCR/STC data to skip and handling good PCR/STC data for clock recovery without affecting other issues of STB application might be really challenging. Moreover, for handling such large jitter, IP playback path in IP-STB may need further optimization.

For A/V sync, the proposed new designs has been explored in prototype STB system, however, real time implementation of our approach in Dual video STB products could be further extended. We have also presented some suggestions in existing schemes in this thesis, which shows significant improvements when compared with existing implementations. However, some more modifications might be needed, for future STB products in which, a much tighter AV Sync specifications are being expected.

References

- (Andreotti *et.al.*, 1995) Andreotti Giovanni Fausto, Giampaola Michieletto, Luigi Mori and Alberta Profum, “Clock Recovery and Reconstruction of PAL Pictures for MPEG coded Streams Transported Over ATM Networks”, *IEEE Trans on Circuits and Systems for Video Technology*, December 1995; 5(6): 508-514.
- (Antonio *et.al.*, 2001) Antonio A. D’Amico, Aldo N. D’Andrea and Ruggero Reggiannini, “Efficient non-data-aided carrier and clock recovery for satellite DVB at Very Low signal-to-noise ratios”, *IEEE on Selected areas in communications*, December 2001; 19(12): 2320-2330.
- (ATSC, 2003) ATSC Implementation Subcommittee Finding: Relative Timing of Sound and Vision for Broadcast Operations, Doc. IS-191, June, 2003.
- (Aweya *et. al.*, 2007) James Aweya, Michel Ovevette, Delfin Y. Montuno, Kent E. Felske, “Differential clock recovery in Packet networks”, Patent WO2007051282, May 2007.
- (Baker, 2003) Daniel G Baker, “MPEG PCR jitter, frequency offset and drift rate measurements”, US Patent 6650719, Nov 18, 2003.
- (Best, 1993) R. Best, “Phase-Locked Loops: theory, design and applications”, ISBN 0-07-911386-9, McGraw-Hill,1993.
- (Bhattacharya, 2006) P. Bhattacharya, “System level PTS-based AV synchronization strategies for playback chains spanning multiple processor cores”, *Proc. IEEE Conf. on Consumer Electronics*, LasVegas, Nevada, 2006; 351-352.
- (Bin and Klara 2002) Yu Bin, Nahrstedt Klara, “A Realtime software solution for resynchronizing filtered MPEG2 transport stream”, *Fourth IEEE International Symposium on Multimedia Software Engineering (ISMSE’02)*, Newport Beach, USA, 2002; 296-303.
- (Bjorn, 2000) Bjorn Kaxe, “Synchronisation of MPEG-2 based digital TV services over IP networks”, Master thesis project performed at Telia research AB, 2000.
- (Bolot,1993) J.C. Bolot, “Characterizing end-to-end packet delay and loss in Internet”, *Journal of high speed Networks*, December 1993; 2: 305-323.

- (Bolot *et. al.*,1995) J.C. Bolot, H. Crepin, and A. V. Garcia, “Analysis of audio packet loss in the Internet,” *Proc. NOSSDAV*, Durham, NH, 1995; 154–165.
- (Chanwoo *et.al.*,2006) Chanwoo Kim, Kwang deok Seo, Wonyong Sung, Soon heung Jung, “Efficient audio/video synchronization method for video telephony system”, *Proc. IEEE Conf. on Consumer Electronics*, Las Vegas, Nevada, 2006; 137-138.
- (Chen *et. al.*,1995) Herng-Yow Chen, Nien-Bao Liu, Chee-Wen Shiah, Ja-Ling Wu, Wen-Chin Chen, Ming Ouhyoung, “A Novel Audio/video Synchronization Model and its Application in Multimedia Authoring System”, *IEEE Trans. on Consumer Electronics* ,Feb 1995; 41(1): 176-177.
- (Cole and Rosenbluth, 2001) R.G. Cole,J. Rosenbluth, “Voice over IP performance monitoring,” *Journal of Computer Communications Review*, April 2001; 4(3): 19-24.
- (Cooper, 2008) J.C. Cooper, “A Short Tutorial on Lip Sync Errors, the Sources and Solutions”, by Pixel Instruments Corp, 2008. [online] Available: <http://www.pixelinstruments.tv/articles.html>
- (Cristian,1989) F.Cristian,“Probabilistic clock synchronization”, *Distributed Computing*, 1989; 3: 146-158.
- (Davis, 2008) Kevin Davis, “latency and Jitter adapted from “Sources of Latency”, June 23, 2008; [online] Available: http://www.networkperformancedaily.com/2008/06/latency_and_jitter_1.html
- (DLNA, 2004) Digital Living Network Alliance Home Networked Device Interoperability Guidelines, version 1.0, June 2004, [online] Available: <http://www.dlna.org/guidelines/>
- (DVB-IP, 2005) Transport of MPEG-2 Based DVB Services over IP Based Network, DVB Bluebook, A086r3, draft TS 102 034. Available : <http://www.dvb.org>.
- (Edward and David 1988) Edward A. Lee and David G. Messerschmitt, “Digital Communication”, Kluwar Academic Publishers, 1988, Chapter 13.
- (Ethernet IEEE 802.3) “Ethernet IEEE 802.3”, 2002, [online] Available: <http://standards.ieee.org/getieee802/802.3.html>

- (ETS 300744, 1999) ETS Document 300744, “Digital broadcasting system for television, sound and data services framing structure, channel Coding and modulation for digital terrestrial television,1999.
- (Fischer, 2004) Walter Fischer, “Digital television: a practical guide for engineers”, Springer, 2004.
- (Fisher *et.al.*, 2007) Fisher Jeffrey, Mamidwar Rajesh S, Schoner Brian, “PCR clock recovery in an IP network”, US Patent 2007286241 (A1), Dec 2007
- (Gardner *et al.*, 2003) Gardner M.T, Frost V.S., Petr D.W.(2003),Using Optimization to achieve efficient quality of service in voice over IP networks in Performance Computing and Communication conference Cat. No.03CH37463 , 475-480.
- (Gardner,1979) F. M. Gardner, “Phaselock Techniques”, John Wiley Sons Inc., New York, 1979.
- (Gringeri *et al.*, 1998) S. Gringeri, B. Khasnabish, A. Lewis, K. Shuaib, R. Egorov, and B. Basch, “Transmission of MPEG-2 Video Streams over ATM”, *IEEE Trans. on Multimedia*, May 1998; 5: 58-71.
- (Harry and Van 1971) Harry L. Van Trees, “*Detection, Estimation, and Modulation Theory - Part II : Nonlinear Modulation Theory*”, John Wiley and Sons, Inc.,1971, Chapter 3.
- (Holborow, 1994) C. E. Holborow, “Simulation of Phase-Locked Loop for processing jittered PCRs”, *ISO/IEC JTC1/SC29/WG11, MPEG94/071*,1994.
- (H.261,1993) “Line Transmission of non-Telephone Signals Video Codec for Audiovisual Services ”, *ITU-T, Recommendation H.261.*, Mar. 1993.
- (H.263,1998) “Video coding for low bit rate communication”, *ITU-T, Recommendation H.263.*, Feb. 1998.
- (ISMA, 2004) ISMA: Internet Streaming Media Alliance Implementation Specification, Ver 1.0, June 3, 2004, [online]: <http://www.isma.tv/specreq.nsf/SpecRequest>
- (ITU-R,1998) ITU-R BT.1359-1,“Relative Timing of Sound and Vision for Broadcasting” , Jan 1,1998.
- (Kaiser, 1993) Kaiser Y., “Synchronization and dejittering of a TV decoder in ATM networks”, *Proc. of Packet Video Workshop*, 1993; 1-3.

- (Kim *et al.*, 2005) C. Kim, S. Park and K.d.Seo, "An efficient audio/video synchronization method for video telephony," *KISS Korea Computer Congress*, July 2005.
- (Kim and Seo, 2006) C. Kim and K.d. Seo, "An apparatus for synchronizing audio/video for hand-held devices," *Korea Patent Application*, P04-046697, 2006.
- (Kostas *et al.*, 1998) T. Kostas, M. Borella, I. Sidhu, G. Schuster, and J. Grabiec, "Real-time voice over packet-switched networks," *IEEE Trans. On Networks*, 1998; 12: 18–27,
- (Lau and Fleischer, 1992) R. C. Lau & P.E Fleischer, "Synchronous Techniques for timing recovery in BISDN", *Proc. IEEE GLOBECOM*, 1992; 814-820.
- (Lee and Bulzacchelli, 1992) T. H. Lee & J. F. Bulzacchelli, "A 155-MHz Clock Recovery Delay and Phase-Locked Loop", 39th IEEE Intern. Solid State Circuit Conference, *ISSCC92*, San Francisco, CA, USA , Feb 1992; 160-161.
- (Liang *et al.*, 2001) Y. Liang, N. Färber, and B. Girod, "Adaptive playout scheduling using time scale modification in packet voice communications," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP)*, Salt Lake City, UT, May 2001; 3: 1445–1448.
- (Liming, 2008), Liming xiu "A novel DCXO module for clock Synchronization Module in MPEG2 Transport System", *IEEE Transaction on Broadcasting*, 2008; 55(8): 2226-2237.
- (Lin J., 2005) Lin J., "A Low-Phase-Noise 0.004 ppm/Step DCXO with Guaranteed Monotonicity in the 90 nm CMOS process", *IEEE Transactions of solid state circuit*, 2005; 40(12): 2726-2734.
- (Linear acoustic, 2004) "Audio and Video Synchronization: Defining the Problem and Implementing Solutions by Linear Acoustic Inc. Rev. 1", 2004; [online]: available on www.LinearAcaoustic.com
- (LR tut) [online] Available: <http://easycalculation.com/statistics/learn-regression.php#> (last accessed on august 2009)
- (Marsch, 1999) D. Marsch, "Digital video TV comes into focus", *EDN Magazine*, July 1999; 6: 40-51.

- (Markopoulou *et.al.*, 2003) Markopoulou A.P.,Tobagi F.A,Karam M.J, “Assessing the quality of voice communications over Internet backbones”, *IEEE/ACM Trans. on Networking*, 2003; 11(5): 747-760.
- (Mathur and Saha, 2007) M. Mathur, K. Saha, “Scalable Integer based Frequency error estimation technique for clock recovery in Packet switched Networks”, *Proc. of Internet and Multimedia Systems and Applications*, Hawaii, USA, 2007.
- (MatLab-ver 7) MatLab: Signal Processing Toolbox,V7 , *The MathWorks Inc*,
- (Milenkovic, 1998) M. Milenkovic, “Delivering Interactive Services via a Digital TV Infrastructure”, *IEEE Trans. On Multimedia*, 1998; 12: 34-43.
- (Monika *et.al.*,December 2008) Jain Monika, Jain P.C., Jain Ankit, Jain Sharad, “Moving Window Averaging Filter for Clock Recovery in Set-Top Box”, *Proc. National Conference on wireless and optical communication (WOC 2008)*, Punjab Enginering College, Chandigarh, December 18-19, 2008;1-5.
- (Monika *et.al.*, Jan 2009) Jain Monika, Jain P.C, Jain Ankit, Jain Sharad, “Enhanced FIR Filter based Module for Clock Synchronization in MPEG2 Transport Stream”, *Proc. Intern. Conference on Advances in Computing, Communication & Control ICAC3’09 (2009 ACM 978-1-60558-351-8)*, FRCRCE Mumbai, India, January 23–24, 2009; 312-316.
- (Monika *et.al.*, March 14, 2009) Jain Monika, Jain P.C., Jain Ankit, Jain Sharad, “Performance Evaluation of Enhanced FIR Filter based module for Clock Synchronization in MPEG-2 Transport stream”, *Intern. Conference on Multimedia, Signal Processing and Communication Technologies IMPACT ’09 (2009 IEEE 978-1-4244-3604-0/09)*, Aligarh Muslim University, Aligarh, India, March 14 - 16, 2009; 262-265.
- (Monika *et.al.*, March 16, 2009) Jain Monika, Beaumont J.M, Jain P.C., Jain Sharad, “Novel Approach for Audio Video Synchronization in Digital Set-Top Box”, *Proc. IASTED Intern. Conference on Advances in Computer Science and Engineering (ACSE 2009)*, Phuket, Thailand, March 16-18, 2009;171-176.
- (Monika *et. al.*, June 2009) Jain Monika, Beaumont J. M., Capony Vincent, Jain P. C., Jain Sharad, and Jain Ankit, “Continuous Adaptation Enhancement in Linear

Regression Algorithm for Clock Recovery in IPTV Environment”, *IEEE Transactions on Broadcasting*, June 2009; 55(2): 485-490.

(Monika *et.al.*, oct 2009) Jain Monika, Jain P.C., Jain Sharad., Jain Ankit, “Efficient Low cost clock recovery module for Terrestrial Set-top Boxes”, *JDCTA: International Journal of Digital content Technology and its Applications*, Oct 2009 (Accepted-in press)

(Moon *et.al.*,1999) S. B. Moon, P. Skelly, and D. Towsley, “Estimation and Removal of Clock Skew from Network Delay Measurements”, *Proc.of Infocom'99*, New York, March 1999.

(Moon *et.al.*,1998) S. Moon, J. Kurose, and D. Towsley, “Packet audio playout delay adjustment: performance bounds and algorithms,” *ACM/Springer Multimedia Syst.*, Jan. 1998; 6: 17–28.

(MPEG2- System) “Generic coding of moving pictures and associated information: Systems CH-1211”, *ISO/IEC 13818-1*, Geneva:, 2006.

(MPEG TV, 2009) MPEG TV homepage, [online] Available: <http://www.mpegTV.com/>. (Last accessed, August 2009)

(MPEG2 RTI) “Coding of Moving Pictures and Associated Audio, Real Time Interface Specification”, *ISO/IEC CD 13818-9*, July 1996.

(MPEG1,11172-1) “Coding of moving pictures and associated audio for digital storage media at up about 1.5 Mbps, Part 1: Systems”, *ISO/IEC FIS 11172-1*, May 1993,

(MPEG1,11172-2) “Coding of moving pictures and associated audio for digital storage media at up about 1.5 Mbps, Part 2: Video”, *ISO/IEC FIS 11172-2*, May 1993,

(MPEG1,11172-3) “Coding of moving pictures and associated audio for digital storage media at up about 1.5 Mbps - Part 3: Audio”, *ISO/IEC, FIS 11172-3*, May 1993.

(MPEG2,13818-1) “Information technology - Generic coding of moving pictures and associated audio information, Part 1: Systems”, *ISO/IEC DIS 13818-1*, Dec 2000

(MPEG2,13818-2) “Information technology - Generic coding of moving pictures and associated audio information, Part 2: Video”, *ISO/IEC DIS 13818-2*, Dec 2000.

(MPEG2,13818-3) “Information technology - Generic coding of moving pictures and associated audio information, Part 3: Audio”, *ISO/IEC DIS 13818-3*, 1998.

- (MPEG4,14496) “MPEG-4 Overview, Version 2”, *ISO/IEC FDIS 14496*, Oct. 1999.
- (Mujica *et.al.*, 2003) F.A Mujica, U.Dasgupta, M. Ali, “Digital timing recovery for communication systems”, *IEEE Global Telecommunications Conference, GLOBECOM '03*, Dec. 2003; 4: 2130- 2135.
- (Noro, 1997) R. Noro, J.P. Hubaux, R. Meuli, R. Laurini, and R. Patthey, “Real-Time Telediagnosis of Radiological Images through an Asynchronous Transfer Mode network: The ARTeMeD Project”, *Journal of Digital Imaging*, August. 1997; 10(1); 116-127.
- (Noro, 2000) Noro Raffaele, “Synchronization over packet switching networks: theory and applications”, Ph.D.dissertation, Dept. of Communications, Ecole Polytechnique Federale de Lausanne (EPFL), Switzerland, 2000.
- (Osaki, 2002) Osaki Bunri, “Method and apparatus for reducing jitter of a program clock reference in a transport stream of MPEG over ATM and MPEG decoder”,US Patent 6377588, April 23, 2002
- (Parekh,1997) Parekh S.P., “Jitter Estimation and clock Recovery with Dejitterization for CBR MPEG-2 Video over ATM Networks”, *Proc of AVSPN*, Aberdeen,UK, 1997; 103-107.
- (Perkins and Lookabaugh, 1998) Michael G. Perkins, L Thomas Lookabaugh, “Reduction of timing jitter in audio-video transport streams”, US Patent 5828414, Oct 27, 1998
- (Peters,1985) J. Peters, “Television 50 years”, European Broadcasting Union 1985. [online] Available: http://www.dvb.org/dvb_articles/dvb_tv-history.pdf
- (Rangan *et.al.*,1996) Rangan P., Kumar S.S., Rajan S, “Continuity and synchronization in MPEG”, *IEEE Journal on Selected Areas in Communications*, 1996; 14(1): 52-60.
- (Ramamoorthy, 1997) Ramamoorthy Victor, “Clock synchronization in software MPEG2 Decoder”, *Proc. SPIE* , 1997; 3021:194 -210.
- (RealPlayer, 2009) RealNetworks, [online] Available: <http://www.real.com> (Last accessed on August 2009)
- (RFC 768) RFC 768 - UDP, User Datagram Protocol, J. Postel ISI, August 28,1980; [online]. Available: <http://www.rfc-editor.org/rfc/rfc768.txt>

- (RFC 791) RFC 791- Internet Protocol, Information Sciences Institutes, University of southern California, Sept 1981; [online]. Available:
<http://www.rfc-editor.org/rfc/rfc791.txt>
- (RFC 793) RFC 793- TCP, Transmission Control Protocol, Information Sciences Institutes, University of southern California, Sept 1981; [online]. Available:
<http://www.rfc-editor.org/rfc/rfc793.txt>
- (RFC 1305) RFC 1305-NTP, Network Time Protocol ver-3, David L. mills, March 1992; [online]. Available : <http://www.rfc-editor.org/rfc/rfc1305.txt>
- (RFC 2030) RFC 2030- SNTP, Simple Network Time Protocol ver-4, David L. mills, Oct 1996; [online]. Available : <http://www.rfc-editor.org/rfc/rfc2030.txt>
- (RFC 2131) RFC- DHCP, Dynamic Host Configuration Protocol, R. Droms, Mar1997; [online]. Available: <http://www.rfc-editor.org/rfc/rfc2131.txt>
- (RFC 2250) RFC 2250- RTP , RTP Payload Format for MPEG1/MPEG2 Video by IETF, Jan 1998; [online]. Available: <http://www.rfc-editor.org/rfc/rfc2250.txt>
- (RFC 2326) RFC 2326- RTSP, Real time Streaming Protocol, H. Schulzrinne, A. Rao, April 1998; [online]. Available : <http://www.rfc-editor.org/rfc/rfc2326.txt>
- (RFC 2616) RFC 2616- HTTP, Hyper Text transfer Protocol, June 1999; [online]. Available: <http://www.rfc-editor.org/rfc/rfc2616.txt>
- (RFC 3376) RFC 3376- IGMP, Internet Group Management Protocol, Ver 3, B.Cain, S.Deering, I. Kouvelas, B. Fenner, A. Thyagarajan, Oct 2002; [online]. Available: <http://www.rfc-editor.org/rfc/rfc3376.txt>
- (RFC 3411) RFC 3411- SNMP, Simple Network Management protocol, D. Harrington, Presuhn, Dec 2002;[online]. Available: <http://www.rfceditor.org/rfc/rfc3411.txt>
- (RFC 3550) RFC 3550- RTP, A Transport Protocol for Real-Time Applications, H.Schulzrinne, S. Casner, R. Frederick, V. Jacobson, July 2003; [online]. Available: <http://www.rfc-editor.org/rfc/rfc3550.txt>
- (Robin and Poulin, 2000) Robin Michael, Poulin Michel, Digital Television Fundamentals, ISBN:0071355812, 2000.
- (Roppel,1995) C. Roppel, "Estimating cell transfer delay in ATM networks using in-service monitoring methods", *Proc. of Globecom'95*, Singapore, Nov.1995.

- (Sanneck *et al.*, 2001) H. Sanneck, L. Le, and A. Wolisz, “Intra-flow loss recovery and control for VoIP,” *Proc. ACM Multimedia*, 2001; 441–454.
- (Shen *et al.*, 2004) Shen G., Nizam M.H.M, Liu E., Giu L., Xu. X, “Fast and accurate clock recovery in packet switched networks”, *Intern. Conf. on Networking and Communication (INCC. 2004.1366585)*, Pakistan, Lahore, 2004; 95-98.
- (Singh *et al.*,1994) R.P. Singh, Sang Hoon Lee, Chong Kwoon ,“Jitter and Clock recovery for Periodic Traffic in Broadband Packet Networks”, *IEEE Trans. on Communications*, May 1994; 42(5): 2189-2196.
- (Siu *et. al.*,1999) Siu, Michael Yiu-kwan, Dyck, Trevor S. R. ,Block, Simon Alexander Ong, Chong T., “Method and apparatus for PCR jitter measurement in an MPEG2 transport stream using sliding window”, US Patent 5883924 , March 16, 1999.
- (STB-7109) Low-cost HDTV set-top box decoder for H.264 and Microsoft WMA9 [online].Available:
<http://www.st.com/stonline/products/literature/bd/11660/sti7109.pdf> (last accessed on March, 2009)
- (Taylor,1997) J. Taylor, DVD Demystified, McGraw-Hill, New York, 1997.
- (TCP/IP, 2001) TCP/IP Tutorial and Technical Overview, IBM, August 2001; [online].Available : http://www.redbooks.ibm.com/redbooks/pdfs/gg_243376.pdf
- (Tryfonas and Verma, 1999) Tryfonas Christos, Varma Anujan, “A Restamping approach to clock recovery in MPEG-2 system Layer”, *Proc. IEEE International Conf. on Communications (ICC.1999.765542)*, Vancouver, Canada, June 6-10,1999; 1285-1290.
- (Varma, 1996) Subir Varma, “MPEG-2 over ATM: System Design Issues”, *IEEE Proceedings of COMPCON*, Santa Clara, USA, 1996; 26–31.
- (Vleeschauwer *et al.*, 2000) V. Vleeschauwer, J. Janssen, G. Petit, and F. Poppe, “Quality bounds for packetized voice transport”, Alcatel, Tech. Rep.,1st Quarter, 2000.
- (Watanabe *et al.*, 2006) Watanabe M., Umeki M., Okazak M., “High Performance VCXO with 622.08 MHz Fundamental Quartz Crystal Resonator”, *International Frequency Control Symposium and Exposition*, 2006; 54-57.

- (Weillan and Akyildiz, 2001) Weillan Su., Akyildiz I.F., “The jitter time-stamp approach for clock recovery of real time variable bit rate traffic”, *IEEE/ACM Transactions on Networking*, 2001; 9(6): 746-754.
- (Weisstein- ArithmeticSeries, 2009) Weisstein, Eric W., “ArithmeticSeries”, MathWorld [online].Available: [http://mathworld.wolfram.com/ ArithmeticSeries.html](http://mathworld.wolfram.com/ArithmeticSeries.html) (last accessed on Jan 2009)
- (Weisstein- GeometricSeries, 2009) Weisstein, Eric W., “GeometricSeries”, MathWorld [online].Available: [http://mathworld.wolfram.com/ GeometricSeries.html](http://mathworld.wolfram.com/GeometricSeries.html) (last accessed on Jan 2009)
- (Weisstein- least squares, 2009) Weisstein, Eric W., “Least Squares Fitting”, MathWorld [online].Available: [http://mathworld.wolfram.com/ LeastSquaresFitting.html](http://mathworld.wolfram.com/LeastSquaresFitting.html) (last accessed on July 2009)
- (Weisstein- Pareto, 2009) Weisstein, Eric W., “Pareto Distribution”, MathWorld [online].Available: <http://mathworld.wolfram.com/ParetoDistribution.html> (last accessed on July 2009)
- (Weisstein-Weibull, 2009) Weisstein, Eric W., “Weibull Distribution”, MathWorld [online].Available: <http://mathworld.wolfram.com/WeibullDistribution.html> (last accessed on July 2009)
- (Weisstein- Z-Transform, 2009) Weisstein, Eric W., “Z-Transform”, MathWorld [online]. Available: [http://mathworld.wolfram.com/ Z-Transform.html](http://mathworld.wolfram.com/Z-Transform.html) (last accessed on Jan 2009)
- (Wolf, 1997) C. Wolf, C. Griwodz, R. Steinmetz, “Multimedia Communication”, *Proceedings of the IEEE*, December 1997; 85(12): 1915-1933.
- (Younkin and Corriveau, 2008) A.C. Younkin, P.J. Corriveau, “Determining the Amount of Audio-Video Synchronization Errors Perceptible to the Average End-User”, *IEEE Trans. on Broadcasting*, 2008: 54(3): 623-627.
- (Yuong and Ouhyoung 1993) Yuong-Wei Lei , Ming Ouhyoung, “A New architecture For A TV Graphics Animation Module”, *IEEE Trans. on Consumer Electronics*, 1993; 39(4): 797-800.

Appendix I: Application of Set-Top box Decoder chip

In today's market lots of set-top box decoder chips are available in the market. This appendix provides an overview of a popular, low-cost interactive HDTV set-top box decoder chip STx7109 and various applications of this Set-top box decoder chip.

A1.1 LOW COST SATELLITE HD SET-TOP BOX

DTH is very popular in rural areas and Farm Houses. Free-to-Air (FTA) STB is not aligned with a particular service provider. DTH is growing at rapid pace .Expected 50 million worldwide DTH subscriber to be added by 2009. DTH needs 18" parabolic dish antenna with Low noise block converter (LNBC) LNBC converts from Ku-Band to L-Band.

In Front-end, tuner receives a digital signal from a satellite network and tunes to a particular channel in the frequency range of 950-2250MHz. The signal is digitally demodulated using QPSK demodulator to obtain digital data . Forward Error Correction (FEC) decoder will check errors and send TS to Back-end.

Back-end TS demultiplexer extracts audio, video and clock and sends to A/V decoder and smart card. Smart card determines access rights to various digital services. A/V decoder decompresses audio, video data which are displayed on TV after DAC and Video encoder.

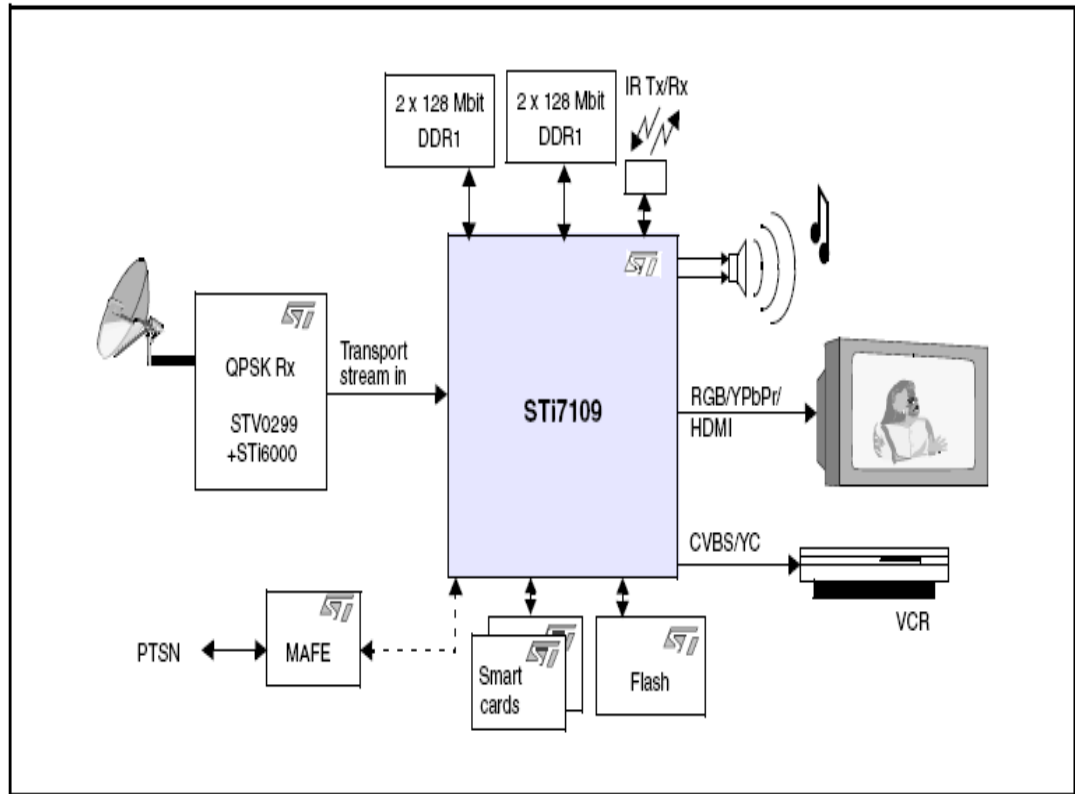


Figure A1.1 Low cost Satellite HD Set- Top Box

STi 7109 can be used as Back end chip for Satellite High Definition Set Top Box using STV0299 Front-end chip which integrates tuner, QPSK/8-PSKdemodulator, and Viterbi code & Reed Soloman code as FEC.

A1.2 LOW COST DUAL SATELLITE & TERRESTRIAL HD STB

Terrestrial platform provides regional & national contents. Household uses both satellite and terrestrial platform. Terrestrial platform is used by 60% users as secondary TV set. Digital Terrestrial set-top box operates in UHF range from 470-860MHz. It uses OFDM modulation to avoid multipath fading. Yagi antenna is being needed to capture the signal. Front-end needs RF tuner, OFDM chip with FEC decoder. Tuner receives RF signal, perform

channel selection, and converts to IF 36MHz .OFDM demodulation performed by applying 2K or 8K FFT to IF signal. If some carriers damaged due to multipath fading, lost bits recovered by FEC decoder. FEC decoder and Back-end are same as in digital satellite STB.

STi 7109 can be used as Low cost Dual Satellite & Terrestrial HD Set- Top Box with HDD & DVD as shown in Figure A1.2. STV0299 front-end chip which integrates tuner, QPSK/8-PSK demodulator, and Viterbi code & Reed Soloman code as FEC is used for satellite decoder while STV0370 front-end chip is used for terrestrial reception. So one out of these two inputs is decoded and displayed while other input can be decoded and recorded in HDD (Hard-disk drive) simultaneously so that it can be played later.

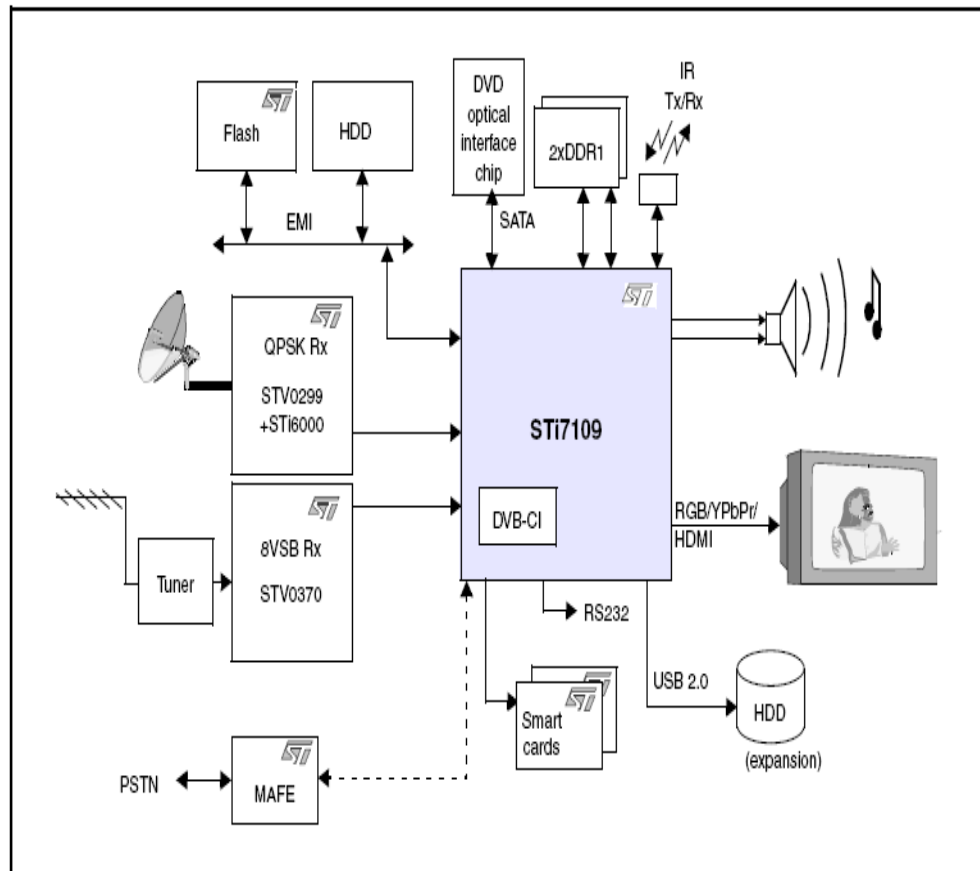


Figure A1.2 Low cost Dual Satellite & Terrestrial HD Set- Top Box with HDD & DVD

A1.3 LOW COST CABLE HD STB WITH RETURN CHANNEL

Fast growing Asian market has an estimated 150 million analog cable subscribers . In India, around 10 million customers use cable network.

Digital Cable STB operates in the VHF and UHF range from 50-860MHz. It uses QAM modulation since transmission on cable is hardly subjected to disturbances 64- or 256- front-end needs RF tuner, QAM chip with FEC (R-S code) decoder. Tuner tunes one of the video channels in 50-860 MHz and select channel on IF 36MHz. QAM demodulates IF signal into digital TS . Bit errors are corrected by FEC decoder. Symbol rate is from 0.87 to 11.7 MBaud. Back-end is same as digital satellite STB.

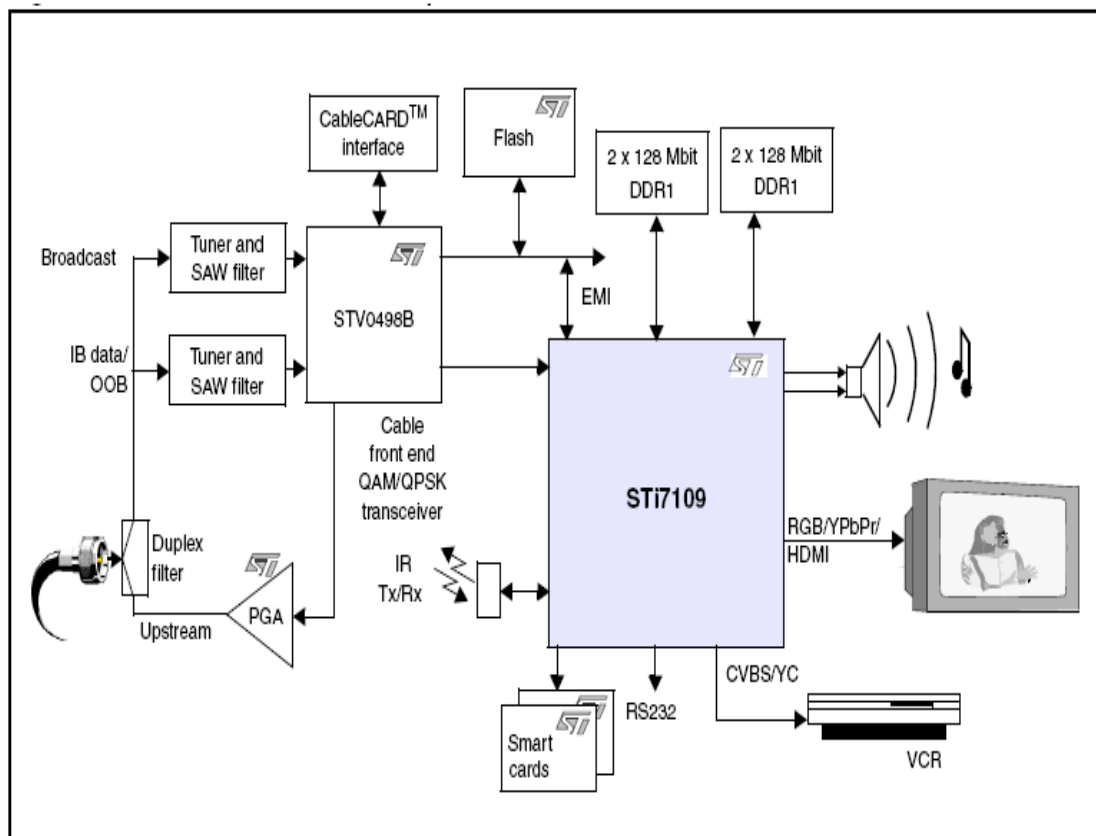


Figure A1.3 Low cost Cable HD Set- Top Box with return channel

QAM permits 6 or 8 bits instead of 2 bits/symbol in QPSK used in satellite STB. With 7 or 8MHz BW, bit rate achieved could be 30Mb/s. Subscriber needs a digital cable connection from a cable operator. STV0498B cable frontend QAM transceiver is used for cable reception along with STi 7109 back end as shown in above Figure A1.3.

A1.4. LOW COST HD IPTV SET- TOP BOX WITH HDD

Video over IP, allows TV viewers to see their favorite channels from say USA, Japan, Korea etc. IPTV will bring every channel in every Nation to every viewer. Increasing worldwide broadband penetration and efficient video compression H.264 enable Telecom services to deliver IP-video to customers. IP-STB consists of MPEG A/V decoder alongwith Ethernet controller. Ethernet controller implement MAC (Media Access Control) and PHY (Physical layer) portion of the CSMA/CD protocol at 10 and 100Mb/s. It integrates IEEE802.3 PHY for twisted pair Ethernet Applications.

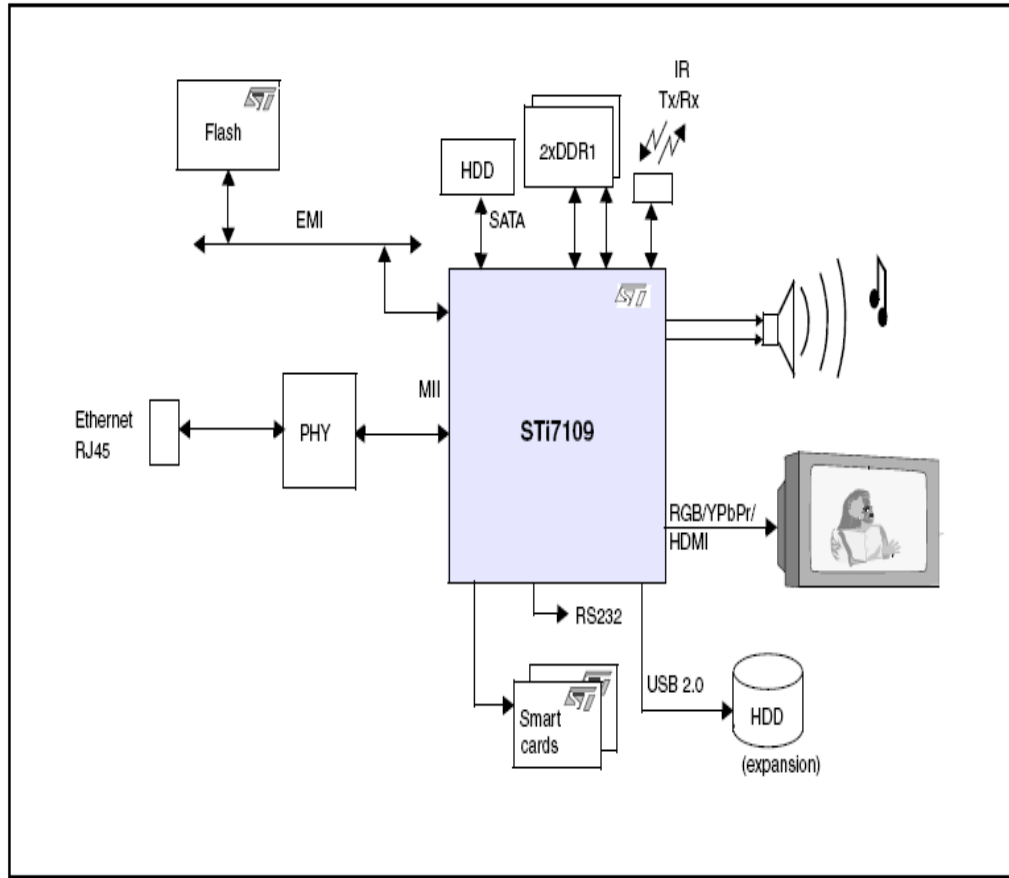


Figure A1.4 Low cost HD IPTV Set- Top Box with HDD

STB7109 chip is an A/V decoder for H.264 and MPEG-2 decoding. It includes Ethernet controller also. For IP-TV applications, the integrated 100BT Ethernet controller and MII/RMII interface can be used for generic Ethernet delivery, as shown in above Figure A1.4.

Thus above appendix demonstrated various capability of latest STD decoder chip STB7109. By choosing the suitable frontend, we can realize any (Satellite, Terrestrial, Cable or IP) broadcast reception. Because of dual decode feature, recording of one decoded stream in hard-disk is possible simultaneously when playing of other decode is in progress.

Appendix-II: Clock Recovery Algorithms Simulation in MATLAB

This appendix provides an overview of the clock recovery simulation process in the MATLAB simulation environment. Our developed algorithms for clock recovery has been firstly implemented in MATLAB to find out the optimum values of various filter parameters. The newly designed clock recovery module's functionality is simulated and tested using MATLAB.m scripts and its functions before applying correction and diagnostics to the local decoder clock for clock recovery in real time environment.

The MATLAB script when run, takes a set of STC-PCR data stored in a file, processes them and synchronizes the decoder clock with the encoder clock. The programming of the FS registers of the decoder clock is also simulated in the MATLAB environment only. The output values are stored in a file for taking logs. The test also plots graphs between PCR count, quantized frequency, and PCR difference.

A2.1 ABOUT MATLAB.M FILES

MATLAB is a powerful programming language as well as an interactive computational environment. Files that contain code in the MATLAB language are called M-files. These M-files can be created using a text editor, and then these files can be used as any other MATLAB function or command. There are two kinds of M-files-

- Scripts, which do not accept input arguments or return output arguments. They operate on data in the workspace.

- Functions, which can accept input arguments and return output arguments. Internal variables are local to the function. If you duplicate function names, MATLAB executes the one that occurs first in the search path.

Scripts can operate on existing data in the workspace, or they can create new data on which to operate. Although scripts do not return output arguments, any variables that they create remain in the workspace, to be used in subsequent computations. In addition, scripts can produce graphical output. Functions are M-files that can accept input arguments and return output arguments. The name of the M-file and of the function should be the same.

A2.2 TESTING THE SCRIPT

To include the PCR-STC data set, PCR-STC values (in free running mode) are stored in to a data file and make sure the file present is in the path given by the variable: `real_dir` which is present in `init_clk_rec_f.m` file. The next step would be to initialize the variable: `filename` with the name of our file. Please note that for testing depending upon the environment whether Satellite or terrestrial, we need to change the parameters like `MAX_WINDOW_SIZE`, `MIN_SAMPLE_THRESHOLD` etc. All the parameters are initialized in `init_clk_rec_f` file.

For running the test, we need to make sure that the working directory is the directory in which all the `.m` files are present and execute the command `main_f` from the MATLAB command prompt. After the simulation is completed, the script asks for input parameters for plotting 2-D graphs involving variables PCR count, quantized frequency and PCR Difference. For example, we need to provide 1, 3 at command prompt for a plot between PCR count and Frequency Difference plot.

A2.3 CODE FLOW

After the execution of the command `main_f`, the file `main_f` is run by the MATLAB in which the main function resides.

main calls the following functions in the order-

- `init_clk_rec_f`: Initializes all the global constants and variables and returns back to main.
- `read_file_f`: reads the entire PCR-STC data set and stores it into the memory.
- `clock_recovery_f`: Inputs the entire data set and performs clock correction and diagnostics.
- `plot_results_f` : Plots the output data according to the input parameters provided by the user

A2.3.1 `init_clk_rec_f`

The constants and variables used for clock recovery are initialized in the file. It contains several user definable variables like `MIN_SAMPLE_THRESHOLD` **and** `MAX_WINDOW_SIZE`, based on which the Weighing Moving Average(WMA) algorithm is implemented .

The global constants declared in this file are:

- `NP_NUM_K` Number of Input data fields
- `PCR_K` PCR clock value, Input data field
- `STC_K` STC clock value, Input data field
- `OUT_NUM_K` Number of Output data fields

- CNT_K Result Count, Output data field
- DIF_K PCR Difference, Output data field
- FRQ_K Freq Difference, Output data field
- WORK_DIR Directory where the .m files are present
- PLOT_DIR Directory of Plotting Functions
- ENC_FREQ Nominal encoder frequency
- XTAL_FRQ Nominal local crystal frequency
- PRECISION_FACTOR used to get the correction up to 2 decimal points
- MIN_SAMPLE_THRESHOLD Number of samples to receive before PWM correction
- MAX_WINDOW_SIZE Maximum number samples in moving average window
- GRADUAL_CORRECTION_FACTOR used for damping the correction value
- PCR_MAX_SAMPLE_ERROR maximum error allowed in tick difference
- SDC_STEP_SIZE step size in clock recovery algorithm*precision factor
- SDC_STEP_VAL step size increments in clock recovery algorithm

Important global variables include:

- file_name the name of the file containing the PCR-STC dataset
- qnt_freq current STC clock frequency
- results_row Current row of output results
- skip_good_pcr Flag set if skipping good PCR/STC data
- pcr_cnt stores the current value of pcr being processed
- window maintains the moving window buffer
- store_count contains the number of elements stored in window

- `real_dir` specifies the location of the directory where the input dir is present

To change the behavior of the Clock recovery module, only this file needs to be edited with the changes in the user variable values.

A2.3.2 read_file_f

This function reads the input data present in the variable filename, located in the directory given by `real_dir`. This file performs a single I/O operation and loads the entire data into memory, thus reducing simulation time.

A2.3.3 clock_recovery_f

The function uses all the data loaded into the memory and performs simulation of all PCR-STC data. The function basically runs a loop feeding PCR-STC data set one by one to the function **sync_management_f** and increments counters for looping purposes. It checks for a `skip_good_pcr` flag and accordingly makes a matrix containing the results, which is used to plot the graph and take logs.

sync_management_f This function takes a PCR-STC data row as input and performs clock correction and diagnostics on a single value. The function stores these values as current PCR value and the current crystal timestamp. The crystal timestamp is processed to generate the simulated STC value by calling the function **gen_simul_stc_f**.

Gen_simul_stc_f checks for clock wrap and calculates STC value using the formula-

$$(xtal_diff * current_decoder_frequency)/(encoder_frequency)$$

After getting the simulated STC value and current PCR value, `PCR_difference` and `STC_difference` are calculated using the function **sub_uint_32**. Tick difference is then found out by subtracting pcr and stc values and converting the unsigned numbers to signed

numbers. The function **u32_to_s32_f** achieves the task of converting unsigned 32 bit numbers to signed numbers. The error conditions are checked using the function **detect_wierd_clock_f** which checks if the pcr_diff or tick_diff is non zero. It also checks if the tick difference is within limits of MAX_PCR_THRESHOLD. The erroneous value is passed on as err_cnt. If err_cnt is not set, the function **control_valid_pcr_f** is called. After execution of check synchronization, the current values of PCR, STC and frequency are saved as previous for future calculations of PCR, STC and tick error.

get_simul_stc_f The function is called from sync_management and it is used to generate simulated STC value from timestamp provided by the free running clock value and programmed output frequency of Frequency Synthesizer. It checks for clock wrap-around and then scales crystal difference to get STC difference using the formula

$$(xtal_diff * current_decoder_frequency)/(encoder_frequency)$$

The new STC value is found out then and wrap around is again checked before converting to unsigned number using the function **uint32**.

control_valid_pcr_f The function inputs the valid PCR, STC and tick error values and uses them to find the frequency difference using the formula

$$freq_diff = (tick_diff * ENC_FREQ)/(pcr_diff)$$

Moving window buffer is then maintained by calling the function **moving_wind_f**.

This function increments the storecount as the data arrives and returns to control_valid_pcr_f. If the moving window size becomes greater than MIN_SAMPLE_THRESHOLD, weighing filter is called which applies weighs to the window and updates a variable weightedsum. The weights applied start from zero and

increment till the middle element of the window is reached. Afterwards, they reduce to zero at the last element. This weighted sum is used to calculate average error and average error per sample (AEPS) greater than 51 Hz. There is no update in the decoder frequency and correction remainder is incremented till the error reaches by 51 Hz. The results `pcr_cnt`, `pcr_diff` and `freq_diff` are saved in the results matrix. After the execution of `clock_recovery_f`, the code returns to `main_f` and the output data matrix is saved and written into a file named 'out.dat'. Further, the output data matrix is fed to `plot_results_f` present in the plotting directory (which is given by the variable `PLOT_DIR`).

A2.3.4 plot_results_f

This function inputs the output matrix and plots the 2-D graphs of the variables required. It asks the user while running the script which plot is required by the user. The plot key displayed is "1-CNT, 2-DIF, 3-BUF, 4-FRQ, 5-IPE". Based on the user input, the function converts the string values of the matrix to integer values using the MATLAB command 'str2num' and feeds them to **plot_graph_f**. This function uses the plot command to generate a graph with the values that are passed to it.

This appendix addresses simulation work details of newly developed algorithms for clock recovery. Initial simulated results shows that newly developed algorithm tracks the encoder clock accurately and no audio-video data packets loss is observed.

Appendix-III: Signaling and Management Protocols for IP Streams

When AV content is streamed over an IP network rather than broadcasted on a MPEG multiplex beamed from a satellite, over cable or over the air, there are some additional challenges. In view of this, IP Network technologies and A/V transport protocols were addressed in chapter-7 . In addition to AV transport protocols, IP streaming systems need other protocols to set up and monitor the AV payload transport. These are various “Signaling & Management Protocols” discussed in this appendix.

A3.1 DHCP: DYNAMIC HOST CONFIGURATION PROTOCOL

An IP host uses DHCP when booting up to acquire an IP address and other operating parameters from a DHCP server on the network. The DHCP client will broadcast, upon boot up, a DHCP DISCOVER message. DHCP server will reply to this request by a DHCP OFFER message, offering the host to lease an IP address for a certain time period (set by the server). The host will accept the offer by sending a DHCP REQUEST message and the server will finalize the transaction by sending a DHCP ACK message. Upon the lease time expiration, the DHCP client will try to renew the lease by sending a DHCP REQUEST message and the server will accept the renewal by sending DHCP ACK.

DHCP protocol includes numerous options that can be used by an IPTV operator to set up an IP STB. The IP STB can also use some DHCP vendor specific options in its DHCP DISCOVER message to identify its type and model number, software revision, etc...

allowing the DHCP server to return box specific options in its response. DHCP is described in RFC 2131.

A3.2 IGMP: INTERNET GROUP MANAGEMENT PROTOCOL

Each broadcast TV channel by an IPTV provider is carried over a dedicated multicast IP group (identified by its multicast IP address). The IPTV provider edge router (located behind the DSLAM for TV-over-ADSL), will forward multicast IP packets only to STBs that explicitly requested them, using the Internet Group Management Protocol (IGMP) (RFC 3376). To start receiving a TV channel identified by a multicast IP address, an IP STB must send an IGMP “Membership Report” message with the desired multicast IP address. This IGMP command received by the edge router will send the requested multicast IP packets to the IP STB. Periodically (every 125 seconds), the edge router will send a “Membership Query” message to the multicast group IP address; the IP STB must reply with an IGMP “Membership Report” message to confirm that it still “tuned” to the TV channel associated with this multicast IP group and wants to keep receiving these packets. When the user is switching to another broadcast TV channel, the IP STB must send an IGMP “Leave Group” message before sending an IGMP “Membership Report” for the new desired TV channel. The IGMP “Leave Group” command when received by the edge router will make it stop transmitting multicast IP packets corresponding to the previously watched TV channel.

This is especially crucial for applications where the data link to the IP STB is of limited capacity, such as ADSL. State-of-the-art DSLAM products now often include multicast IP routing functionalities in order to effectively support IPTV over ADSL networks. IGMP version 3 is described in RFC 3376.

A3.3 RTSP: REAL TIME STREAMING PROTOCOL

Video on Demand (VoD) programs are transmitted to an IP-STB using unicast IP packets, sent to the IP STB's own unique IP address. VoD requires different signaling protocols than the ones used for watching broadcast TV: the user wants to control the video play out and use trick modes (play, pause, fast-forward, rewind, etc...). The protocol used for such control tasks is RTSP ("the web's remote control"). RTSP allows an IP STB to fetch information about a program from a video server, to request the server to start streaming an A/V content file (using RTP or another transport protocol such as HTTP), pause, and do trick modes. Since RTSP protocol uses URLs, pointers to video stream files can be easily embedded into web pages presented to the user.

The first command (called "method" in RTSP jargon) generally used on a new video file is DESCRIBE. The video server will then return a description file, generally formatted using the Session Description Protocol (SDP) format. Other RTSP methods (SETUP, PLAY, PAUSE, TEARDOWN) allow the IP STB to control the A/V content streaming from the server. RTSP is described in RFC 2326.

A3.4 DVB-STP: DVB SERVICE DISCOVERY & SELECTION PROTOCOL

DVB-STP is defined in the DVB-IP standard TS 102 034, published in 2005 by the ETSI: Transport of MPEG-2 Based DVB Services over IP Based Networks (DVB IP –TS 102 034 v1.1.1). Its main usage is to allow an IP STB to retrieve information on the different services available on the network (live TV channels, VOD content,...) and the methods to

access them. A server sends DVB-STP messages over a multicast IP address using UDP transport. By default, the multicast IP address 224.0.2.14 and the UDP destination port 3937 are used. DVBSTP messages include a 12-byte header (with optional extensions) and a content payload using the XML format.

A3.5 SNMP: SIMPLE NETWORK MANAGEMENT PROTOCOL

IPTV service providers often use SNMP to manage IP STBs during IPTV network operation. A SNMP agent is running on every IP STB, listening for incoming SNMP requests from a network management station (also called SNMP manager). The SNMP agent has access to a certain number of the IP STB operating parameters, some parameters with read/write access, others with read-only access.

In SNMP standard, a group of such parameters is called a Management Information Base (MIB). Each individual variable (or MIB object) within a MIB can be read from or written to (if allowed) over the network by the SNMP manager. Commands sent by the SNMP manager to the IP STB's SNMP agent include:

GET: read the specified MIB object value

GETNEXT: read the next MIB object value

SET: set the specified MIB object value

In addition, an IP STB can initiate the sending of an SNMP message (called TRAP) to the SNMP manager. Some standard MIBs are often implemented (System MIB, Interface MIB, IP MIB, Ethernet MIB, etc...); however, there are no standard MIB requirements for IP STB. Most of the times, each IPSTB manufacturer implements a vendor-specific MIB, in

addition to the standard MIBs. Most IP STB deployments use SNMPv2. Since SNMP gives read and sometimes write access to large number of IP STB parameters, it is not considered as a potential security risk.

SNMPv3 is a more secure network management solution. SNMPv3 includes access control and authentication of the SNMP manager by the agent. It is based on key exchange and digital certificates. The SNMPv3 agent will respond only to authorized and authenticated SNMP management stations. SNMPv3 message exchanges can also be encrypted to prevent any snooping. SNMP version 3 is described in RFC3411.

A3.6 NTP: NETWORK TIME MANAGEMENT & SNTP: SIMPLE NETWORK TIME PROTOCOL

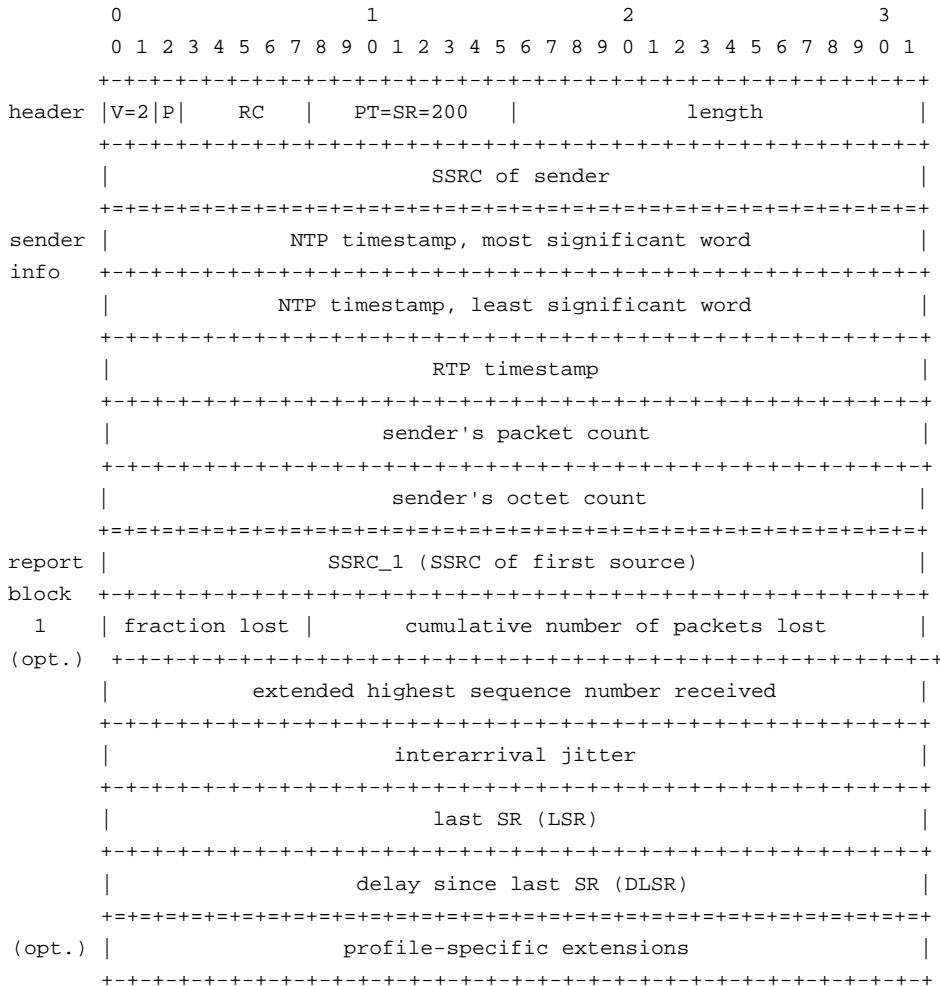
IPTV systems use NTP or SNTP to provide accurate and consistent time stamps to all IP STBs and video servers. NTP uses sophisticated algorithms to ensure high precision timing accuracy. SNTP is a simplified version of NTP using the same data format. The NTP time stamps are also used by RTCP and RTSP protocols. When queried, a NTP or SNTP server will reply with a message carrying a 64-bit time stamp (wall clock). These time stamps can be used by the NTP/SNTP client on the IP STB to synchronize the local “wall clock”.

A3.7 RTCP: Real Time Control Protocol

RTCP is a control protocol defined in the same manner as RFC in RTP (RFC 3550) in order to provide information to both sender(s) and receiver(s) involved in a RTP streaming session. RTCP packets are always transmitted to the next odd UDP port number immediately following the even port number of the associated RTP stream (RFC 3550). RTCP specification defines several RTCP packet types to carry a variety of control information.

- SR: Sender report, for transmission and reception statistics from participants that are active senders
- RR: Receiver report, for reception statistics from participants that are not active
- SDES: Source description items
- BYE: Indicates end of participation
- APP: Application-specific functions

Sending of RTCP packets is optional, especially for receivers. In IPTV systems with one server sending RTP packets to hundreds or thousands of receivers, RTCP traffic from receivers would be impossible. Sender's reports from the (unique) server, on the other hand, are possible and actually are of particular interest for media timing synchronization. A large part of RTCP specification is actually dedicated to computing the frequency and size of RTCP reports, so that the total RTCP traffic never takes more than 5% of the overall RTP stream bandwidth (RFC 3550) . The most widely used RTCP message for most systems (e.g. one sender, multiple receivers) is the Sender Report (SR). SR packets should not take more than 25% of the total bandwidth allocated to RTCP traffic, itself limited to no more than 5% of the total RTP stream bandwidth. SR message format (RFC 3550) is as follows -



PT (Payload Type) must be 200 for a Sender report.

The “sender info” part contains -

- A 64-bit NTP time stamp (see earlier section) indicating the sender’s “wall clock” time when this sender’s report was sent.
- The RTP time stamp value corresponds to the same time as the NTP timestamp (above), but in the same unit and with the same random offset as the RTP timestamps in RTP data packets.
- Sender’s packet and byte counters.

The NTP and corresponding RTP time stamps can be used by the receiver to synchronize different RTP streams (like audio and video), sent from the same server using the same wall clock reference in RTCP SR messages. The reception report block(s) are optional.

An IP Set-Top Box is capable of decoding and displaying digital audio and video received over a high-speed IP network. In this appendix, various signaling and management protocol needed to handle various aspects of real time live audio/video transmission on IP networks have been presented.

Appendix-IV: Derivation and analysis of Clock Recovery Algorithms

This appendix consists of four sections. First section deals with FIR analysis of our developed light weighted algorithm for DVB-S and DVB-T. In second section of Linear Regression Analysis, formulae for the intercept error and slope errors have been derived. In third section, transfer function of linear regression algorithm has been derived and overall system transfer function is evaluated. Further, in last section, differential analysis of the whole system when LR is used as in-loop filter is described.

A4: FIR ANALYSIS

This section details results of a MATLAB simulation of the transfer function of our developed algorithm for DVB-S and DVB-T. In that algorithm, 2 in-loop FIR filters and a damping weight of 0.3.

The total system transfer function for closed feedback loop used for clockrecovery was derived in chapter-6 and it was shown by equation (6). Recalling that equation, total system transfer function is expressed as below in equation (1)-

$$\frac{C(Z)}{D(Z)} = \frac{-F(Z)}{1 - Z^{-1} + Z^{-1}.F(Z)} = \frac{-Z}{(Z-1) / F(Z) + 1} \quad (1)$$

A general FIR filter that filters the current input and the previous (n-1) inputs, has a transfer function given by -

$$F(Z) = a_0 + a_1 z^{-1} + a_2 z^{-2} + \dots + a_{n-1} z^{-(n-1)} \quad (2)$$

Where a_i are the weights for the FIR filter taps .When $F(Z)$ from (2) is substituted in (1)

we get the following total system transfer function -

$$\frac{C(Z)}{D(Z)} = \frac{-(a_0 + a_1 z^{-1} + a_2 z^{-2} + \dots + a_{n-1} z^{-(n-1)})}{1 + (a_0 - 1)z^{-1} + \{a_1 z^{-2} + a_2 z^{-3} + \dots + a_{n-1} z^{-n}\}} \quad (3)$$

In our algorithm there are two in-loop FIR filters, a pyramid weighted 30 tap filter followed by a 8 tap averaging filter. These two filters can be combined into one 37 tap FIR filter. If this filter is normalized and a damping factor of 0.3 applied to the output, then the filter weights are as shown in Figure 6.5, largest central weight is 0.017.

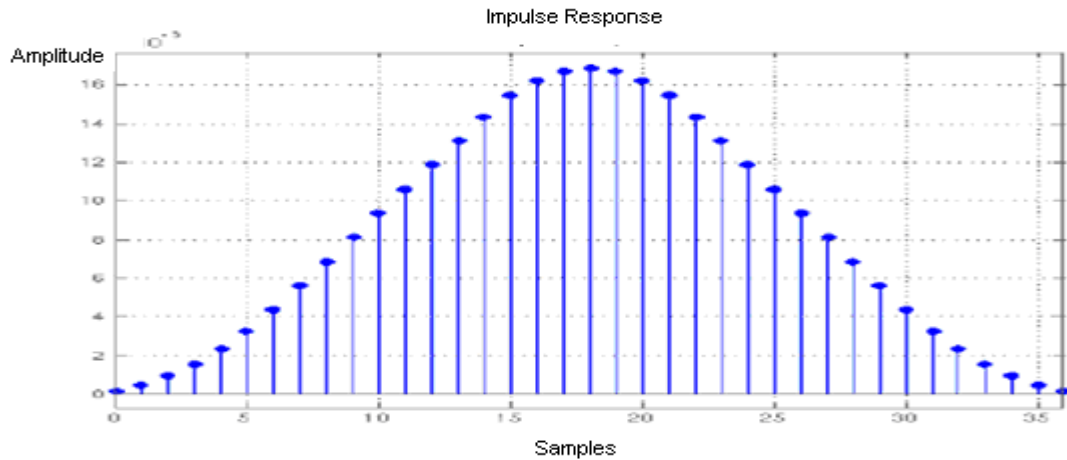


Figure A4.1 FIR damped filter Weights

This response corresponds to (2) multiplied by the damping factor. The equivalent response for equation (3), the total system impulse response, is shown in Figure A4.2. It oscillates with increasing amplitude. That is, the total system is unstable.

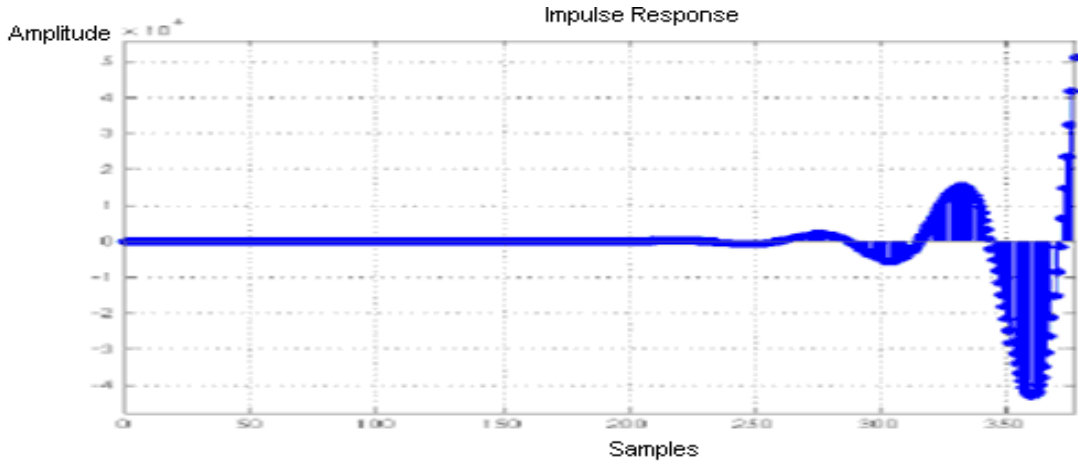


Figure A4.2 Total system impulse response with FIR inloop filter

The corresponding Pole-Zero plot for total system is shown in Figure A4.3. On the far right 2 poles can be seen which are outside the unit circle which gives another indication of an unstable system.

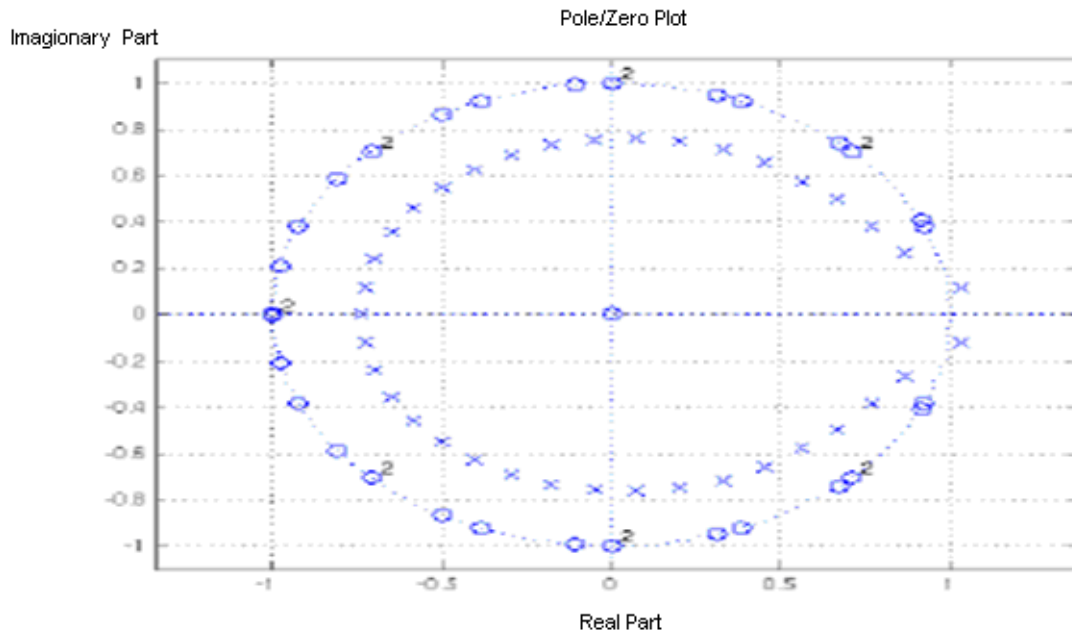


Figure A4.3 Pole-Zero plot for total system with FIR in-loop filter (X- Pole, O = Zero)

B4: LINEAR- REGRESSION ANALYSIS

B4.1 Error Analysis

This section derives formulae for the uncertainty in the intercept and slope estimates. Consider three independent variables **A**, **B** and **C**, each has an error associated with it. Let the standard deviation of these errors be σ_A , σ_B and σ_C respectively. Let the weighted sum $\mathbf{S} = \mathbf{aA} + \mathbf{bB} + \mathbf{cC}$, where **a**, **b**, and **c**, are weights. Then the variance of sum (**S**), is a weighted sum of the variances of **A**, **B** and **C** . Where the new weights are shown in the equation below-

$$\sigma_S^2 = \frac{(a^2 \cdot \sigma_A^2 + b^2 \cdot \sigma_B^2 + c^2 \cdot \sigma_C^2)}{(|a| + |b| + |c|)} \quad (4)$$

B4.1.1 Finite Series Of Samples

The following standard results for summing arithmetic series as per Weisstein-ArithmeticSeries, 2009 will be needed-

$$\sum_{i=1}^N i = \frac{N(N+1)}{2} \quad (5)$$

$$\sum_{i=1}^N i^2 = \frac{N(N+1)(2N+1)}{6} \quad (6)$$

B4.1.1.1 Intercept Error

The intercept estimated over N samples is given by-

$$\text{Intercept(K)} = \frac{\left[\left[\sum_{i=1}^N y_i \right] - \left[\mathbf{M} \cdot \left[\sum_{i=1}^N x_i \right] \right] \right]}{N} \quad (7)$$

Assuming there is no error in x_i and the slope \mathbf{M} is small, then most of the error will be due to the y_i values. Let the standard deviation of each error in y_i be σ_i . Furthermore assume that all the y_i are independent. Then we can use the result derived in (4) to form the following equation for the variance of the intercept estimate-

$$\sigma_K^2 = \frac{\left[\sum_{i=1}^N \left(\frac{1}{N} \right)^2 \cdot \sigma_i^2 \right]}{\left[\sum_{i=1}^N \left| \frac{1}{N} \right| \right]} = \frac{\left[\frac{1}{N^2} \sum_{i=1}^N \sigma_i^2 \right]}{(1)} \quad (8)$$

If the errors σ_i are all drawn from the same distribution with standard deviation σ_y then this simplifies to-

$$\sigma_K^2 = \frac{1}{N^2} (N\sigma_y^2) = \frac{\sigma_y^2}{N} \quad (9)$$

The intercept error (standard deviation) is then given by-

$$\sigma_K = \frac{\sigma_y}{\sqrt{N}} \quad (10)$$

B4.1.1.2 Slope Error

The slope estimated over N samples is given by-

$$\text{slope}(M) = \frac{\left(N \cdot \left(\sum_{i=1}^N x_i y_i \right) - \left(\sum_{i=1}^N x_i \right) \left(\sum_{i=1}^N y_i \right) \right)}{\left(N \cdot \left(\sum_{i=1}^N x_i^2 \right) - \left(\sum_{i=1}^N x_i \right)^2 \right)} \quad (11)$$

This analysis uses the approximation that the x_i values are periodic. These are the PCR difference values, and for a fixed time interval \mathbf{T} , they are all equal to \mathbf{L} (from equation (7) of chapter-6). Therefore the x_i values form the series (\mathbf{L} , $2\mathbf{L}$, $3\mathbf{L}$, $4\mathbf{L}$, ...).

Equation (11) now becomes -

$$M = \frac{\left[N \cdot \left[\sum_{i=1}^N iLy_i \right] - \left[\sum_{i=1}^N iL \right] \left[\sum_{i=1}^N (y_i) \right] \right]}{\left[N \cdot \left[\sum_{i=1}^N (iL)^2 \right] - \left[\sum_{i=1}^N (iL) \right]^2 \right]} \quad (12)$$

This simplifies to-

$$M = \frac{\left[N \cdot \left[\sum_{i=1}^N iy_i \right] - \left[\sum_{i=1}^N i \right] \left[\sum_{i=1}^N (y_i) \right] \right]}{\left[N \cdot \left[\sum_{i=1}^N i^2 \right] - \left[\sum_{i=1}^N i \right]^2 \right]} L \quad (13)$$

Substituting for the series sums from (5) and (6) gives -

$$M = \frac{\left[N \cdot \left[\sum_{i=1}^N iy_i \right] - \frac{N(N+1)}{2} \left[\sum_{i=1}^N (y_i) \right] \right]}{\left[N \frac{N(N+1)(2N+1)}{6} - \left[\frac{N(N+1)}{2} \right]^2 \right]} L \quad (14)$$

Re-arranging gives -

$$M = \left(\frac{12}{LN(N^2-1)} \right) \sum_{i=1}^N \left(i - \frac{(N+1)}{2} \right) y_i \quad (15)$$

This is now in the form of a sum of independent variables y_i and using the result in (4), we

can form the following equation for the variance of the slope -

$$\sigma_M^2 = \frac{\left(\frac{12}{LN(N^2-1)} \right)^2 \left[\sum_{i=1}^N \left(i - \frac{(N+1)}{2} \right)^2 \sigma_i^2 \right]}{\sum_{i=1}^N \left(i - \frac{(N+1)}{2} \right)^2} \quad (16)$$

Multiplying out the inner terms and re-arranging to remove the modulus operand gives-

$$\sigma_M^2 = \frac{\left(\frac{12\sigma_y}{LN(N^2-1)}\right)^2 \left(\sum_{i=1}^N i^2 - (N+1)i - \left(\frac{(N+1)^2}{2}\right)\right)}{\left[\sum_{i=1}^{N/2} \frac{(N+1)}{2} - i\right] + \left[\sum_{i=N/2+1}^N i - \frac{(N+1)}{2}\right]} \quad (17)$$

Forming independent sums, and re-arranging gives-

$$\sigma_M^2 = \frac{\left(\frac{12\sigma_y}{LN(N^2-1)}\right)^2 \left(\frac{N(N+1)^2}{4} + \left[\sum_{i=1}^N i^2\right] - (N+1)\sum_{i=1}^N i\right)}{\left[\sum_{i=1}^N i\right] - 2\left[\sum_{i=1}^{N/2} i\right]} \quad (18)$$

Substituting for the series sums from (5) and (6) gives-

$$\sigma_M^2 = \frac{\left(\frac{12\sigma_y}{LN(N^2-1)}\right)^2 \left(\frac{N(N+1)^2}{4} + \frac{N(N+1)(2N+1)}{6} - (N+1)\frac{N(N+1)}{2}\right)}{\left[\frac{N(N+1)}{2}\right] - 2\left[\frac{N/2(N/2+1)}{2}\right]} \quad (19)$$

Re-arranging gives-

$$\sigma_M^2 = \left(\frac{\sigma_y}{L}\right)^2 \left(\frac{12}{N(N^2-1)}\right)^2 \left(\frac{N(N^2-1)}{12}\right) \Bigg/ \left(\frac{N^2}{4}\right) = \left(\frac{\sigma_y}{L}\right)^2 \left(\frac{48}{N^3(N^2-1)}\right) \quad (20)$$

Therefore for a sufficiently large number of samples (**N**), the following approximation holds-

$$\sigma_M \approx \left(\frac{\sigma_y}{L}\right) \sqrt{\frac{48}{N^5}} \approx \left(\frac{\sigma}{L}\right) \cdot \frac{6.928}{N^{2.5}} \quad (21)$$

B4.1.2 Infinite Series Of Samples

This section repeats the analysis of section B4.1.1 but using exponential weights on a

continuous stream of samples. The following standard results from Weisstein-GeometricSeries, 2009 for summing geometric series are required -

$$\sum_{i=1}^{\infty} r^{(i-1)} = \frac{1}{1-r} \quad (22)$$

$$\sum_{i=1}^{\infty} i.r^{(i-1)} = \frac{1}{(1-r)^2} \quad (23)$$

$$\sum_{i=1}^{\infty} i^2 .r^{(i-1)} = \frac{(1+r)}{(1-r)^3} \quad (24)$$

First we derive some preliminary results based on the approximation that the samples arrive periodically with time interval \mathbf{T} , and increment \mathbf{L} . As it was discussed in chapter-8: section 8.3.3.1 how a continuous stream could be handled by using a set of exponentially distributed weights (\mathbf{a} , \mathbf{ab} , \mathbf{ab}^2 , \mathbf{ab}^3 , \mathbf{ab}^4 , ...), and that if $\mathbf{a} + \mathbf{b} = 1$, then the weights are normalised. That is, their sum is 1.

For infinite series, this simplifies the result in (4) to the following -

$$\sigma_K^2 = \sum_{i=1}^{\infty} (W_i \sigma_i)^2 \quad (25)$$

Taking the equations for the intercept and slope estimate from (7) and (11), they can be re-expressed for an infinite series as -

$$\text{Intercept}(\mathbf{K}) = \bar{y} - M \bar{x} \quad (26)$$

$$\text{Slope}(\mathbf{M}) = \frac{\overline{xy} - \bar{x}.\bar{y}}{x^2 - (\bar{x})^2} \quad (27)$$

Recall from chapter8: section 8.3.3.2 that the current sample sits at the origin, and the older samples are at negative \mathbf{x} locations. So that the series x_i is (... $-4\mathbf{L}$, $-3\mathbf{L}$, $-2\mathbf{L}$, $-\mathbf{L}$, 0), with respective weightings (... \mathbf{ab}^4 , \mathbf{ab}^3 , \mathbf{ab}^2 , \mathbf{ab} , \mathbf{a}).

Therefore the mean \mathbf{x} value can be expressed as -

$$\bar{x} = \sum_{i=0}^{\infty} w_i x_i = \sum_{i=0}^{\infty} (ab^i)(-iL) = -abL \sum_{i=1}^{\infty} ib^{(i-1)} \quad (28)$$

Using the result from (23), this gives -

$$\bar{x} = abL \left(\frac{1}{(1-b)^2} \right) = \frac{-bL}{a} \quad (29)$$

The mean x^2 value is given by -

$$\overline{x^2} = \sum_{i=0}^{\infty} w_i x_i^2 = \sum_{i=0}^{\infty} (ab^i)(-iL)^2 = abL^2 \sum_{i=1}^{\infty} i^2 b^{(i-1)} \quad (30)$$

Using the result from (24), this gives -

$$\overline{x^2} = -abL^2 \left(\frac{(1+b)}{(1-b)^3} \right) = \frac{b(1+b)L^2}{a^2} \quad (31)$$

The mean \mathbf{y} value can be expressed as -

$$\bar{y} = \sum_{i=0}^{\infty} w_i y_i = \sum_{i=0}^{\infty} (ab^i)(y_i) = a \sum_{i=0}^{\infty} b^i y_i \quad (32)$$

The mean \mathbf{xy} value can be expressed as:-

$$\overline{xy} = \sum_{i=0}^{\infty} w_i x_i y_i = \sum_{i=0}^{\infty} (ab^i)(-iL)(y_i) = -abL \sum_{i=1}^{\infty} ib^{(i-1)} y_i \quad (33)$$

Having now finished the preliminaries, lets evaluate the intercept and slope errors for an exponentially weighted continuous stream.

B4.1.2.1 Intercept Error

Substituting the results for the mean \mathbf{x} and mean \mathbf{y} values from (29) and (32) in the intercept equation (28) yields -

$$K = \left[a \sum_{i=0}^{\infty} b^i y_i \right] - M \cdot \left(\frac{-bL}{a} \right) \quad (34)$$

Assuming the slope \mathbf{M} is small, then most of the error will be due to the y_i values and the variance equation (25) for infinite series can be used to obtain the intercept variance in the form shown below -

$$\sigma_K^2 = \frac{\left[\sum_{i=0}^{\infty} (ab^i)^2 \cdot \sigma_i^2 \right]}{\left[\sum_{i=0}^{\infty} |ab^i| \right]} = (a\sigma_y)^2 \frac{\left[\sum_{i=1}^{\infty} (b^2)^{(i-1)} \right]}{(1)} \quad (35)$$

Using the result in (22), this gives -

$$\sigma_K^2 = (a\sigma_y^2) \left(\frac{1}{1-b^2} \right) \quad (36)$$

Therefore for a sufficiently small value for filter coefficient \mathbf{a} , the following approximation holds-

$$\sigma_K = \sigma_y \sqrt{\frac{a}{1+b}} \approx \sigma_y \sqrt{\frac{a}{1+1}} \approx \sigma_y \cdot 0.7071 \sqrt{a} \quad (37)$$

B4.1.2.2 Slope Error

The corresponding slope error will now be derived. In the slope equation for infinite series (27), substituting for the mean values of \mathbf{x} , \mathbf{x}^2 , \mathbf{y} and \mathbf{xy} from equations (29), (31), (32) and (33) gives -

$$M = \frac{\left[-abL \sum_{i=1}^{\infty} ib^{(i-1)} y_i \right] - \left(\frac{-bL}{a} \right) \left(a \sum_{i=0}^{\infty} b^i y_i \right)}{\left(\frac{b(1+b)L^2}{a^2} \right) - \left(\frac{-bL}{a} \right)^2} \quad (38)$$

Multiplying out and re-arranging gives -

$$M = \frac{a^3}{L} \left(\frac{1}{a} \left[\sum_{i=0}^{\infty} b^i y_i \right] - \sum_{i=1}^{\infty} i b^{(i-1)} y_i \right) = \frac{a^3}{bL} \left(\frac{b}{a} \left[\sum_{i=0}^{\infty} b^i y_i \right] - \left[\sum_{i=0}^{\infty} i b^i y_i \right] \right) \quad (39)$$

Re-arranging gives -

$$M = \frac{a^3}{bL} \sum_{i=0}^{\infty} \left(\frac{b}{a} - i \right) b^i y_i \quad (40)$$

This is now in the form of a sum of independent variables y_i , and using the result in (4), we

can form the following equation for the variance of the slope -

$$\sigma_M^2 = \frac{\left(\frac{a^3}{bL} \right)^2 \left[\sum_{i=0}^{\infty} \left(\frac{b}{a} - i \right)^2 b^{2i} \sigma_i^2 \right]}{\sum_{i=0}^{\infty} \left(\frac{b}{a} - i \right) b^i} \quad (41)$$

Re-arranging to eliminate the modulus operand gives -

$$\sigma_M^2 = \frac{\left(\frac{a^3 \sigma_y}{bL} \right)^2 \left[\left(\frac{b}{a} \right)^2 + b^2 \sum_{i=1}^{\infty} \left(\frac{b}{a} - i \right)^2 b^{2(i-1)} \right]}{\left[\left(\frac{b}{a} \right) + b \sum_{i=1}^{\frac{b}{a}} \left(\frac{b}{a} - i \right) b^{i-1} + b \left[\sum_{i=\frac{b}{a}+1}^{\infty} \left(i - \frac{b}{a} \right) b^{i-1} \right] \right]} \quad (42)$$

Multiplying out the inner term, and rearranging gives -

$$\sigma_M^2 = \frac{\left(\frac{a^3 \sigma_y}{bL} \right)^2 \left[\frac{1}{a^2} + \left[\sum_{i=1}^{\infty} \left(\frac{b}{a} \right)^2 b^{2(i-1)} - \left(\frac{2bi}{a} \right) b^{2(i-1)} + i^2 b^{2(i-1)} \right] \right]}{b \left[\frac{1}{a} + \left(\sum_{i=1}^{\infty} \left(\frac{b}{a} - i \right) b^{i-1} \right) + 2 \left(\sum_{i=\frac{b}{a}+1}^{\infty} \left(i - \frac{b}{a} \right) b^{i-1} \right) \right]} \quad (43)$$

Forming independent sums and re-arranging gives -

$$\sigma_M^2 = \frac{\left(\frac{a^3 \sigma_y}{L}\right)^2 \left[\frac{1}{a^2} + \left(\frac{b}{a}\right)^2 \left[\sum_{i=1}^{\infty} b^{2(i-1)} \right] - \frac{2b}{a} \left[\sum_{i=1}^{\infty} i b^{2(i-1)} \right] + \left[\sum_{i=1}^{\infty} i^2 b^{2(i-1)} \right] \right]}{b \left[\frac{1}{a} + \frac{b}{a} \sum_{i=1}^{\infty} b^{i-1} + \left(2b^{\frac{b}{a}} - 1\right) \sum_{i=1}^{\infty} i b^{i-1} \right]} \quad (44)$$

Substituting with the results for infinite series in (22), (23) and (24) gives-

$$\sigma_M^2 = \frac{\left(\frac{a^3 \sigma_y}{L}\right)^2 \left[\frac{1}{a^2} + \left(\frac{b}{a}\right)^2 \left[\frac{1}{1-b^2} \right] - \frac{2b}{a} \left[\frac{1}{(1-b^2)^2} \right] + \left[\frac{(1+b^2)}{(1-b^2)^3} \right] \right]}{b \left[\frac{1}{a} + \frac{b}{a} \left[\frac{1}{1-b} \right] + \left(2b^{\frac{b}{a}} - 1\right) \left[\frac{1}{(1-b)^2} \right] \right]} \quad (45)$$

Multiplying out and cancelling terms gives -

$$\sigma_M^2 = \frac{\left(\frac{a^3 \sigma_y}{L}\right)^2 \left(\frac{2a^3}{(1+b)^3}\right)}{\frac{a^2}{2b^{\frac{b}{a}}}} = \left(\frac{\sigma_y}{L}\right)^2 \left(\frac{a^5}{b^{\frac{1}{a}}(1+b)^3}\right) \quad (46)$$

Therefore for a sufficiently small value for filter coefficient **a**, the following approximation holds -

$$\sigma_M = \left(\frac{\sigma_y}{L}\right) \sqrt{\frac{a^5}{b^{\frac{1}{a}}(1+b)^3}} \approx \left(\frac{\sigma_y}{L}\right) \cdot \sqrt{\frac{ea^5}{(1+1)^3}} \approx \frac{\sigma_y}{L} \cdot 0.5829a^{2.5} \quad (47)$$

Compare the equations for intercept and slope errors for finite series in (10) and (21), with those for infinite series in equations (37) and (47) . With exponential weights, 1/**a** seems to play a similar role to that of the number of samples (**N**) in the finite stream version.

C4: FILTER TRANSFER FUNCTION

This section of appendix derives a transfer function for the Linear Regression algorithm. In order to do this the linear regression formula for a continuous series needs to be recast into a recursive form.

C4.1.1 Recursive Mean

First a recursive form for the mean y value (μ) will be derived. Referring back to equation (32) we have-

$$\mu_0 = a \sum_{i=0}^{\infty} b^i y_i = ay_0 + a \sum_{i=1}^{\infty} b^i y_i = ay_0 + b.a \sum_{i=1}^{\infty} b^{(i-1)} y_i \quad (48)$$

By noticing that the summation is now the previous mean y value, we have the following recursive formula for the mean y value-

$$\mu_0 = ay_0 + b\mu_1 \quad (49)$$

Recalling that $\mathbf{a} + \mathbf{b} = 1$, equation (49) is the simple IIR filter shown in Figure C4.1.

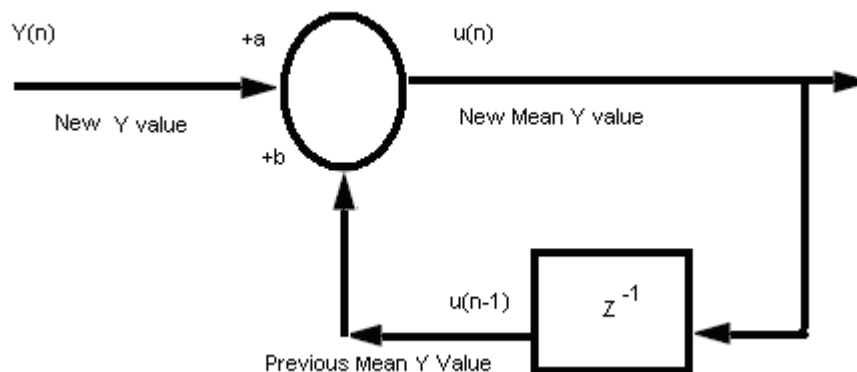


Figure C4.1 IIR Filter for Mean value

C4.1.2 Recursive Slope

A recursive relation for the linear regression slope will now be derived. From equation (40), the formula for the slope can be re-expressed as-

$$\frac{bL}{a^3} \cdot M_0 = \sum_{i=0}^{\infty} \left(\frac{b}{a} - i \right) b^i y_i = \frac{b}{a} \cdot y_0 + \sum_{i=1}^{\infty} \left(\frac{b}{a} - i \right) b^i y_i = \frac{b}{a} \cdot y_0 + b \sum_{i=1}^{\infty} \left(\frac{b}{a} - i \right) b^{(i-1)} y_i \quad (50)$$

$$\frac{bL}{a^3} \cdot M_0 = \frac{b}{a} \cdot y_0 - b \sum_{i=1}^{\infty} b^{(i-1)} y_i + b \sum_{i=1}^{\infty} \left(\frac{b}{a} - i + 1 \right) b^{(i-1)} y_i \quad (51)$$

Noticing that the last summation is related to the previous slope value, and substituting for the first summation from (48) gives -

$$\frac{bL}{a^3} \cdot M_0 = \frac{b}{a} \cdot y_0 - \left(\frac{\mu_0}{a} - y_0 \right) + b \cdot \left(\frac{bL}{a^3} \cdot M_1 \right) \quad (52)$$

Re-arranging gives the following recursive formula for the slope-

$$M_0 = b \cdot M_1 + \frac{a^2}{bL} (y_0 - \mu_0) \quad (53)$$

C4.1.3 Recursive Intercept

A recursive relation for the linear regression intercept will now be derived. From equations (26), (34) and (48), we can obtain the intercept from the slope and μ as follows -

$$K_0 = \left(\frac{bL}{a} \right) \cdot M_0 + \mu_0 \quad (54)$$

Substituting for M_0 from (53) and μ_0 from (49) gives -

$$K_0 = \left(\frac{bL}{a} \right) \left(bM_1 + \frac{a^2}{bT} (y_0 - \mu_0) \right) + (a.y_0 + b.\mu_1) \quad (55)$$

If this is re-arranged in the form -

$$K_0 = b \left(\left(\frac{bL}{a} \right) M_1 + \mu_1 \right) + a(2.y_0 - .\mu_0) \quad (56)$$

It will be noticed that the first term contains the previous intercept value, and substituting from (54) gives -

$$K_0 = b.K_1 + a(2.y_0 - .\mu_0) \quad (57)$$

C4.1.4 Transfer Function For Linear Regression

In Figure 6.4, there is a block for a general filter F(Z). In this section this will be replaced with a recursive linear regression filter. The transfer function for this filter LR(Z) will be derived. So far, the linear regression has been presented in terms of **X** and **Y** inputs. As discussed in section 8.2.1, the **X** value is the PCR tick value, and the **Y** value is the unfiltered pseudo buffer level. As shown in Figure 8.4, the Linear Regression filter produces 2 outputs (slope and intercept) that are combined to form a frequency correction. This relationship, along with all the other relevant time series are expressed below with their Z transform equivalents. They are also shown diagrammatically in Figure C4.2.

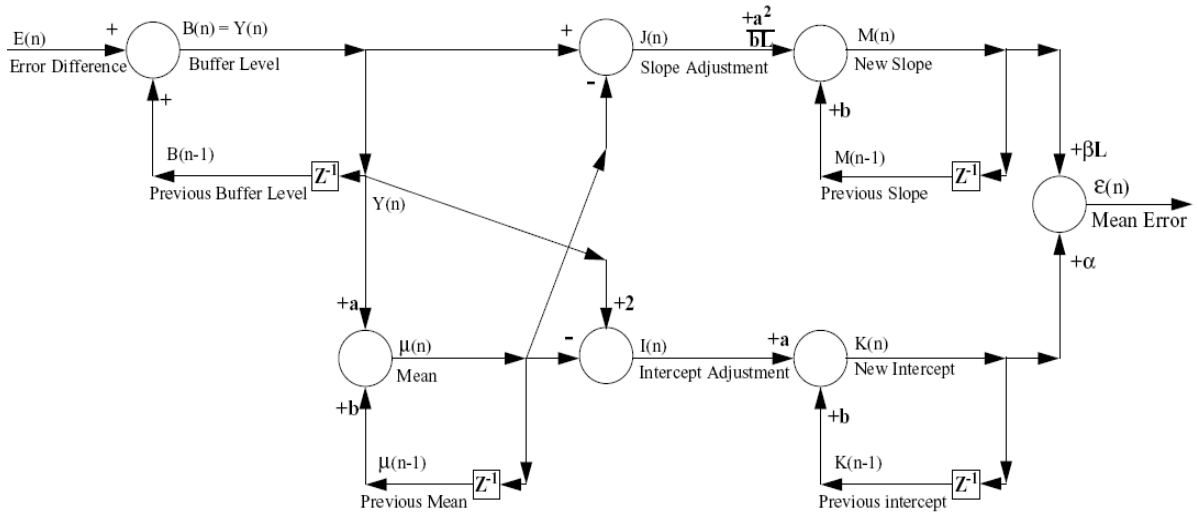


Figure C4.2 Z Transform for Linear Regression

Update the buffer from input difference errors (as E_n derived in chapter-6 in equation (9))-

$$B_n = B_{n-1} + E_n \Leftrightarrow B(Z) = Z^{-1} \cdot B(Z) + E(Z) \quad (58)$$

Update the mean buffer level (Y value) (derived in (49)) -

$$U_n = b \cdot U_{n-1} + a \cdot B_n \Leftrightarrow U(Z) = b \cdot Z^{-1} \cdot U(Z) + a \cdot B(Z) \quad (59)$$

Form the slope adjustment (as required in (53))-

$$J_n = B_n - U_n \Leftrightarrow J(Z) = B(Z) - U(Z) \quad (60)$$

Form the intercept adjustment (as required in (57))-

$$I_n = 2B_n - U_n \Leftrightarrow I(Z) = 2B(Z) - U(Z) \quad (61)$$

Update the slope adjustment (derived in (53))-

$$M_n = b \cdot M_{n-1} + \left(\frac{a^2}{bL} \right) \cdot J_n \Leftrightarrow M(Z) = b \cdot Z^{-1} \cdot M(Z) + \left(\frac{a^2}{bL} \right) \cdot J(Z) \quad (62)$$

Update the intercept adjustment (derived in (57))-

$$K_n = b \cdot K_{n-1} + a \cdot I_n \Leftrightarrow K(Z) = b \cdot Z^{-1} \cdot K(Z) + a \cdot I(Z) \quad (63)$$

As discussed in section 8.4 in chapter-8, the slope has to be scaled by the mean PCR difference to have the same units as the intercept. This analysis uses a fixed PCR difference of \mathbf{L} (from equation (7) of chapter-6). Weight and add 2 error estimates -

$$\varepsilon_n = \alpha.K_n + \beta L.M_n \Leftrightarrow \varepsilon(Z) = \alpha.K(Z) + \beta L.M(Z) \quad (64)$$

Substituting from (62) and (63) for $\mathbf{M}(\mathbf{Z})$ and $\mathbf{K}(\mathbf{Z})$ in (64) gives-

$$\varepsilon(Z) = \alpha \left(\frac{al(Z)}{1-bZ^{-1}} \right) + \beta L \left[\frac{\left(\frac{a^2}{bL} \right) J(Z)}{1-bZ^{-1}} \right] = \frac{a[\alpha bl(Z) + \beta aJ(Z)]}{b(1-bZ^{-1})} \quad (65)$$

Substituting from (60) and (61) for $\mathbf{I}(\mathbf{Z})$ and $\mathbf{J}(\mathbf{Z})$ in (65) gives-

$$\varepsilon(Z) = \frac{a[\alpha b(2B(Z) - U(Z)) + \beta a(B(Z) - U(Z))]}{b(1-bZ^{-1})} \quad (66)$$

Re-arranging gives -

$$\varepsilon(Z) = \frac{a[(2\alpha\beta + \beta a)B(Z) - (\alpha\beta + \beta a)U(Z)]}{b(1-bZ^{-1})} \quad (67)$$

Substituting from (59) for $\mathbf{U}(\mathbf{Z})$ in (67) gives -

$$\varepsilon(Z) = \frac{a \left[(2\alpha\beta + \beta a)B(Z) - (ab + \beta a) \left(\frac{aB(Z)}{1-bZ^{-1}} \right) \right]}{b(1-bZ^{-1})} \quad (68)$$

Re-arranging gives -

$$\varepsilon(Z) = \frac{a[(\alpha + \alpha\beta + \beta a) - (2\alpha\beta + \beta a)Z^{-1}]B(Z)}{b(1-bZ^{-1})^2} \quad (69)$$

Substituting from (58) for $\mathbf{B}(\mathbf{Z})$ in (59) gives -

$$\varepsilon(Z) = \frac{a[(\alpha + \alpha\beta + \beta a) - (2\alpha\beta + \beta a)Z^{-1}]}{b(1 - bZ^{-1})^2} \left(\frac{E(Z)}{1 - Z^{-1}} \right) \quad (70)$$

Finally, the above equation can be re-arranged to form the following Linear Regression transfer function-

$$LR(Z) = \frac{\varepsilon(Z)}{E(Z)} = \frac{aZ^2[(\alpha + \alpha\beta + \beta a)Z - (2\alpha\beta + \beta a)]}{(Z - 1)(Z - b)^2} = \frac{aZ^2[\phi.Z - \varphi]}{(Z - 1)(Z - b)^2} \quad (71)$$

Where $\varphi = (\alpha + \alpha\beta + \beta a)$, and $\psi = (2\alpha\beta + \beta a)$

C4.1.5 System Transfer Function

This subsection will analyse the stability of the total system when Linear Regression is used for the in-loop filter.

In the system transfer function (1), by substituting for $\mathbf{F(Z)}$ with $\mathbf{LR(Z)}$ from (71) gives -

$$\frac{C(Z)}{D(Z)} = \frac{-Z}{(Z - 1) \left(\frac{aZ^2[\phi.Z - \varphi]}{(Z - 1)(Z - b)^2} \right) + 1} \quad (72)$$

Re-arranging gives -

$$\frac{C(Z)}{D(Z)} = \frac{-a\phi.Z^4 + a\varphi.Z^3}{Z^4 - (2b + 2 - a\phi)Z^3 + (b^2 + 4b + 1 - a\varphi)Z^2 - (2b^2 + 2b)Z + b^2} \quad (73)$$

This equation is of little use for studying the stability of the system as α and β are varied.

D4: DIFFERENTIAL ANALYSIS

This section described the system as a differential equation with respect to time. If the frequency correction changes slowly with respect to time, then the time series equations, (58) to (64) can be approximated to differential equations as will now be demonstrated.

Consider equation (59) at the start of the time interval the mean value is U_{n-1} , and at the end of time interval T the mean has changed to U_n . If the mean changes smoothly, then the change of mean over the time interval $[0, T]$ can be approximated by the following straight line-

$$U(t) = U_{n-1} + \frac{at}{T}(B_n - U_{n-1}) \quad (74)$$

Differentiating we get -

$$\frac{dU}{dt} = \dot{U} = A(B - U) \quad (75)$$

Where $A = a/T$.

Rearranging gives the following differential equation -

$$\frac{\dot{U}}{A} + U = B \quad (76)$$

In a similar manner, the following differential equations can be obtained. From (63) for the intercept and from (61) for the intercept adjustment -

$$K_n = b.K_{n-1} + a.l_n \Leftrightarrow \frac{\dot{K}}{A} + K = 2B - U \quad (77)$$

From (62) for the slope, and from (60) for the slope adjustment -

$$M_n = b.M_{n-1} + \left(\frac{a^2}{bL}\right).J_n \Leftrightarrow \frac{\dot{M}}{A} + M = \frac{a}{bL}(B - U) \quad (78)$$

We require more equations to complete the feedback loop. For the frequency correction, from equation (19) of chapter-6 and from (64) for the error estimate -

$$C_n = C_{n-1} + \left(\frac{\varepsilon_n}{T}\right) \Leftrightarrow \frac{\dot{C}}{A} = \frac{A}{a^2}(\alpha K + \beta LM) \quad (79)$$

From (58) for the buffer level, and from equation (17) of chapter-6 for the input sum -

$$B_n = B_{n-1} + E_n \Leftrightarrow \dot{B} = -D - C \quad (80)$$

Assuming that the drift frequency error (**D**) varies very slowly compared to the frequency correction(**C**), then differentiating (80) gives -

$$\ddot{B} = -\dot{C} \quad (81)$$

All the differenatial equations will now be combined to obtain a differential equation only in

C. Differentiating (79) gives -

$$\frac{\ddot{C}}{A} = \frac{A}{a^2}(\alpha\dot{K} + \beta LM) \quad (82)$$

Summing (82) divided by **A**, and (79) gives -

$$\frac{\ddot{C}}{A^2} + \frac{\dot{C}}{A} = \frac{1}{a^2}(\alpha\dot{K} + \beta LM) + \frac{A}{a^2}(\alpha K + \beta LM) = \frac{A\alpha}{a^2}\left(\frac{\dot{K}}{A} + K\right) + \frac{A\beta L}{a^2}\left(\frac{\dot{M}}{A} + M\right) \quad (83)$$

Substituting from (77) for terms in **K**, and from (78) for terms in **M** gives -

$$\frac{\ddot{C}}{A^2} + \frac{\dot{C}}{A} = \frac{A\alpha}{a^2}(2B - U) + \frac{A\beta L}{a^2}\left(\frac{a}{bL}(B - U)\right) = \frac{A}{a^2 b}[(2\alpha b + \beta a)B - (\alpha b + \beta a)U] \quad (84)$$

Differentiating (84) gives -

$$\frac{\ddot{C}}{A^3} + \frac{\dot{C}}{A^2} = \frac{A}{a^2 b}[(2\alpha b + \beta a)\dot{B} - (\alpha b + \beta a)\dot{U}] \quad (85)$$

Summing (85) divided by **A**, and (84) gives -

$$\frac{\ddot{C}}{A^3} + 2\frac{\dot{C}}{A^2} + \frac{C}{A} = \left[\frac{[(2\alpha b + \beta a)\dot{B} - (\alpha b + \beta a)\dot{U}]}{a^2 b} \right] + \left[\frac{[A(2\alpha b + \beta a)B - (\alpha b + \beta a)U]}{a^2 b} \right] \quad (86)$$

Collecting terms in **B** and **U** gives -

$$\frac{\ddot{C}}{A^3} + 2\frac{\dot{C}}{A^2} + \frac{C}{A} = \frac{A}{a^2 b} \left[(2\alpha b + \beta a) \left(\frac{\dot{B}}{A} + B \right) - (\alpha b + \beta a) \left(\frac{\dot{U}}{A} + U \right) \right] \quad (87)$$

Substituting from (75) for terms in **U** give -

$$\frac{\ddot{C}}{A^3} + 2\frac{\dot{C}}{A^2} + \frac{C}{A} = \frac{A}{a^2b} \left[(2\alpha b + \beta a) \left(\frac{\dot{B}}{A} + B \right) - (\alpha b + \beta a)(B) \right] \quad (88)$$

Re-arranging and differentiating gives -

$$\frac{\ddot{C}}{A^4} + 2\frac{\dot{C}}{A^3} + \frac{C}{A^2} = \frac{(2\alpha b + \beta a)}{a^2b} \left(\frac{\ddot{B}}{A} \right) + \frac{\alpha}{a^2} (\dot{B}) \quad (89)$$

In above equation \ddot{C} has been used. As MS word cannot handle 4 diacritical marks, hence substituting from (80) and (81) for terms in **B** give -

$$\frac{\ddot{C}}{A^4} + 2\frac{\dot{C}}{A^3} + \frac{C}{A^2} = \frac{(2\alpha b + \beta a)}{a^2b} \left(\frac{-\dot{C}}{A} \right) + \frac{\alpha}{a^2} (-D - C) \quad (90)$$

Re-arranging gives the following 4th order differential equation for the frequency correction (**C**) in terms of the slowly changing drift frequency error (**D**)-

$$\frac{\ddot{C}}{A^4} + 2\frac{\dot{C}}{A^3} + \frac{C}{A^2} + \frac{(2\alpha b + \beta a)}{a^2b} \left(\frac{\dot{C}}{A} \right) + \left(\frac{\alpha}{a^2} \right) C = - \left(\frac{\alpha}{a^2} \right) D \quad (91)$$

This equation has a general solution of the form -

$$C = K_p e^{-pAt} + K_q e^{-qAt} + K_r e^{-rAt} + K_s e^{-sAt} - D \quad (92)$$

This solution is only valid if the roots **p, q, r, s** are all different. There are relationships between the roots and the coefficients of (92) as follows -

$$2 = p + q + r + s \quad (93)$$

$$1 = pq + pr + ps + qr + qs + rs \quad (94)$$

$$\frac{2\alpha b + \beta a}{a^2b} = pqr + pqs + prs + qrs \quad (95)$$

$$\frac{\alpha}{a^2} = p q r s \quad (96)$$

Let's investigate some properties of these roots. For stability we require all the roots to be positive so all exponential terms in (92) decay at time progresses, and **C** approaches the value **-D**. A large root will decay very quickly, a small root will decay slowly. In order to get a fast settling time, we require large roots. However, by inspecting (93) it will be seen that all roots can not be made simultaneously large. Therefore, the fastest settling time will occur when 2 or more of the smallest roots are equal. This condition is called critical damping. This will now be discussed below.

D4.1 Critical Damping Coefficient

Critical damping occurs when the maximum number of smallest roots are equal. (So solution (92) is not appropriate.) If all roots are equal to **p**, then from (93) and (94) we have -

$$2 = 4 p \quad (97)$$

$$1 = 6 p^2 \quad (98)$$

These 2 relationships can **not** be satisfied simultaneously, so 4 equal roots is **not** a solution.

If 2 pairs of equal roots (**p** and **q**) are a solution -

$$2 = 2 p + 2 q \quad (99)$$

$$1 = p^2 + 4 p q + q^2 \quad (100)$$

These 2 relationships have the following solutions, either [**p** = 0, **q** = 1], or [**p** = 1, **q** = 0].

These two solutions do **not** have positive root values, and some terms will **not** decay.

Therefore 2 pairs of equal roots is **not** a solution. If a single root (**p**), and a triple root (**q**) is a solution-

$$2 = p + 3q \quad (101)$$

$$1 = 3(pq + q^2) \quad (102)$$

These 2 relationships have the following solution -

$$p = \frac{(1 \pm \sqrt{3})}{2} \quad (103)$$

$$q = \frac{(3 \mp \sqrt{3})}{6} \quad (104)$$

Remembering that all roots must be positive, the following values are the solution -

$$p = \frac{(1 + \sqrt{3})}{2} \approx 1.366 \quad (105)$$

$$q = \frac{(3 - \sqrt{3})}{6} \approx 0.2113 \quad (106)$$

The triple root (\mathbf{q}) is the smallest, so we have a critically damped triple root solution-

$$C = K_p e^{-pAt} + (K_q + K_r(At) + K_s(At)^2) e^{-qAt} - D \quad (107)$$

The values of α and β that generate this solution will now be determined. The triple root solution of (96) is-

$$\frac{\alpha}{a^2} = pq^3 \quad (108)$$

Substituting from (105) and (106) for \mathbf{p} and \mathbf{q} gives -

$$\alpha = a^2 \left(\frac{1 + \sqrt{3}}{2} \right) \left(\frac{3 - \sqrt{3}}{6} \right)^3 = (2\sqrt{3} - 3) S_a^2 \quad (109)$$

Where S_a is the scaling parameter $\mathbf{a/6}$. The triple root solution of (95) is -

$$\frac{2\alpha b + \beta a}{a^2 b} = 3pq^2 + q^3 \quad (110)$$

Re-arranging gives the following expression for β -

$$\beta = ab(3pq^2 + q^3) - \left(\frac{2b}{a} \right) \alpha \quad (111)$$

Substituting for α from (109), and re-arranging gives -

$$\beta = ab(3pq^2 + q^3) - \left(\frac{2b}{a}\right)(a^2 pq^3) = abq^2(3p + q - 2pq) \quad (112)$$

Substituting from (105) and (106) for **p** and **q** gives -

$$\beta = ab \left(\frac{3-\sqrt{3}}{6}\right)^2 \left[3 \left(\frac{1+\sqrt{3}}{2}\right) + \left(\frac{3-\sqrt{3}}{6}\right) - 2 \left(\frac{1+\sqrt{3}}{2}\right) \left(\frac{3-\sqrt{3}}{6}\right) \right] = bS_a \quad (113)$$

Therefore to achieve critical damping, the intercept contribution α (109) scales as the square of the filter strength (**a**), while the slope contribution β (113) scales approximately linearly with filter strength (assuming **a** is small).

In above appendix, first section shows that when FIR filter is used as in-loop filter then whole system becomes unstable for high jittery environment of IP-STB. In further section, transfer function for LR algorithm is derived and then overall system transfer function is calculated. It is being found that equation of overall system transfer function is of little use for studying the stability of the system. Hence in further section system is analyzed as a differential equation with respect to time. For stable system, the condition of critical damping is derived for our newly devised continuous adaption enhanced LR algorithm for IP-STB.

List of Publications/Presentations by Author

- [1] Jain Monika, Beaumont J.M., Capony V., Jain P. C., Jain S., and Jain A. Continuous Adaptation Enhancement in Linear Regression Algorithm for Clock Recovery in IPTV Environment, **IEEE Tran. On Broadcasting**, June 2009, 55(2); 485-490.
- [2] Jain Monika, Jain P.C., Jain S., Jain A. Novel clock recovery module for MPEG stream in Terrestrial Set-top Boxes, **JDCTA: International Journal of Digital Content Technology and its Applications**, Oct 2009, 3(4); 85-90.
- [3] Jain Monika, Jain P. C., Jain Sharad “Technical Review of Various Clock Recovery Methods For IP Set-Top Box” JTES 164” in **Journal Delving-Journal of Technology and Engineering Sciences (JTES)**, ISSN no.: 0975-5829, (In Press)
- [4] Jain Monika, Jain P.C., Jain A., Jain S. Enhanced FIR Filter based Module for Clock Synchronization in MPEG2 Transport Stream, Proc. International Conference on Advances in Computing, Communication & Control ICAC3’09 (2009 **ACM**. 978-1-60558-351-8), FRCRCE Mumbai, India, January 23–24, 2009; 312-316.
- [5] Jain Monika, Jain P.C., Jain A., Jain S.. Performance Evaluation of Enhanced FIR Filter based module for Clock Synchronization in MPEG-2 transport stream, International Conference on Multimedia, Signal Processing and Communication Technologies IMPACT ’09 (2009 **IEEE** 978-1-4244-3604-0/09), Aligarh Muslim University, Aligarh, India, March 14 - 16, 2009; 262-265.
- [6] Jain Monika, Beaumont J.M., Jain P.C., Jain Sharad. Novel Approach for Audio Video Synchronization in Digital Set-Top Box, Proc. **IASTED** International Conference on Advances in Computer Science and Engineering ACSE ‘09, **Phuket, Thailand**, March 16-18, 2009; 171-176.
- [7] Jain Monika, Jain P.C., Jain A., Jain S. Moving Window Averaging Filter for Clock Recovery in Set-Top Box, Proc. National Conference on Wireless and Optical Communication WOC ’08, PEC, Chandigarh, December 18-19, 2008;1-5.

Brief Biography of the Candidate

Ms. Monika Jain did her M.Sc (Electronic & Instrumentation) from IIT Roorkee (formerly known as University of Roorkee) in 1997 and M.Tech (Instrumentation & Control) from Devi Ahilya University, Indore in 1998. Presently, she is working as Asst. Professor in Dept. of ECE in Amity School of Engg. & Technology of Amity University at Noida since Oct 2007. She is presently pursuing Ph.D from BITS, Pilani since Jan 2007. She has been working on critical industrial problem of Clock Synchronization for Set top box domain in real time systems .

She is having more than 10 years of teaching experience. Earlier, she has worked at JSS Academy of Technical Education affiliated to U.P. Technical University, Noida for 9 years. She has taught various courses at B.Tech level. She has been coordinating Final year students Projects, Industrial Training Seminar and In-House Training programs. She has designed syllabus for Theory subjects, set-up various Labs. She is life member of IETE. She has been awarded cash prize for first rank in branch in IIT Roorkee, gold Medal for highest marks during graduation and prize for rank at district level in High School. She is having several papers in her account in International Journal and in reputed conference Proceedings.

Brief Biography of the Supervisor

Dr. P. C. Jain acquired BE and ME (Electronics) from BITS, Pilani in 1968 and 1972 respectively and MS and D.Sc. (Applied Sciences) from University of Louvain , Belgium in 1977 and 1979 respectively. He worked at CEERI, Pilani for 30 years on speech and video compression and guided various BE and ME students for their thesis work. In 1989 he was invited to Philips, Germany to design VLSI chip for 2-dimensional discrete cosine transform (DCT) processor to be used in video compression. He has been to USA two times once for presenting a paper in IEEE conf. on consumer electronics and later on to University of California, Santa Barbara to work on MPEG-4 and H.263 for very low bit rate video coding. He has visited Singapore and Hong Kong in various exhibitions and presented a paper at Singapore in Broadcast-Asia in 2007.

Since 1994 he was responsible for Integrated Receiver Decoder (IRD) and NICAM receiver projects in CEERI. Since 1998 he was working as General Manager (R&D) in HFCL. He has developed digital satellite set-top box for DTH application, digital cable, and terrestrial set-top boxes with CONAX conditional access for pay-TV. Since 2003 he was working as Section Manager in STMicroelectronics Pvt. Ltd. to customize the ST products for Indian OEMs. Since 2007 he is working as Professor and Head of School of Electronics at CDAC, Noida.

He has 97 papers in his account in national and International Journals. He is life Fellow of IETE and BES India. He has bagged IETE Prof. S.N. Mitra memorial award and IETE K.S. Krishnan memorial award in 1996 and 1997 respectively. He has been nominated as a Council member of Gerson Lehrman Group Technology, Media & Telecom. Council in 2005, Man of year-1998 by American Biographical Institute, USA, and member of International Who's Who Historical Society for his overall contribution to the society.