Headed Gateway System Integration & Full Validation

Major Project Report

Submitted in partial fulfillment of the requirements

for the degree of

MASTER OF TECHNOLOGY

IN

ELECTRONICS & COMMUNICATION ENGINEERING

(VLSI Design)

By

Jaimin N. Panchal
(12MECV19)

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

INSTITUTE OF TECHNOLOGY

NIRMA UNIVERSITY

AHMEDABAD-382 481

MAY 2014

Headed Gateway System Integration & Full Validation

Major Project Report

Submitted in partial fulfillment of the requirements

for the degree of

MASTER OF TECHNOLOGY

IN

ELECTRONICS & COMMUNICATION ENGINEERING

(VLSI Design)

By

Jaimin N. Panchal

(12MECV19)

Under the Internal Guidance of

Dr. Nagendra Gajjar

and

External Guidance of

Mr. Vaibhav Pathak

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

INSTITUTE OF TECHNOLOGY, NIRMA UNIVERSITY

AHMEDABAD-382 481

Declaration

This is to certify that

- The thesis comprises of my orginal work towards the degree of Master of Technology in VLSI Design at Nirma University and has not been submitted elsewhere for a degree.
- Due acknowledgement has been made in the text to all other material used.

- **Jaimin N. Panchal** [12MECV19]

Certificate

This is to certify that the Major Project entitled **"Headed Gateway System Integration & Full Validation"** submitted by **Mr. Jaimin N. Panchal (12MECV19)**, towards the partial fulfillment of the requirements for the degree of **"Master of Technology" in "VLSI-Design"** of **Nirma University of Science & Technology; Ahmedabad** is the record of work carried out by him under our supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for the examination. The results embodied in this major project, to the best of our knowledge, haven't been submitted to any other university or institution for award of any degree or diploma.

External Guide:

_____

Mr. Vaibhav Pathak
Senior Project Manager
DCG Deparment
STMicroelectronics India Pvt. Ltd.

Internal Guide:

_____

Dr. Nagendra Gajjar
Sr. Associate Professor
M.Tech (Embedded System)
Nirma University

PG Co-ordinator:

_____

Dr. N. M. Devashyaree
VLSI Design

HOD:

_____

Dr. P. N. Tekwani
Professor, EC

Director:

_____

Dr. K. Kotecha
Director, IT-NU

Date: _____

Place: Ahmedabad

Acknowledgements

No work can be completed by an individual effort. The success of this project largely depends on intensity, drive and technical competence of many individuals, who contributed to make this project a reality even in face of in surmountable obstacles.

"Words are a dress of thoughts, appreciating and acknowledging those who are responsible for the success of a project".

First and foremost, sincere thanks to **Dr. N. M. Devashrayee**, P.G. Coordinator of VLSI design,Institute of technology,Nirma University,Ahmedabad.I enjoyed his vast knowledge and thank him a lot for giving valuable support for project work.

I would also like to thank **Dr. K. R. Kotecha**, Director & **Dr. P. N. Tekwani**, Head of Department, Institute of Technology, Nirma University, Ahmedabad for providing me an opportunity to get an internship at STMicroelectronics India Private Limited, Greater Noida.

I would also like to thank my internal guide **Dr. Nagendra Gajjar** of Nirma University, Ahmedabad and all the faculty member of Electronics & Communication Department for their effort of constant co-operation which have been significant factor in the accomplishment of my industrial training.

It is my proud privilege and pleasure to bring my indebtedness and warm gratitude to **Mr.Vaibhav Pathak**, *Senior Project Manager*, STMicroelectronics, Greater Noida, for his support during my project work.

I express my sincere gratitude and thanks to **Mr. Manish Kumar Sharma** and **Mr. Saurabh Kumar Awasthi**, who has always been a guide during my work. I have warm regards for my team mates in ST Microelectronics, who have obliged me with their time to time guidance.

Finally, my thanks to everyone who has in some way or other helped me in completing this project successfully. I should not fail to mention my parents who have always been a source of inspiration. I am grateful to my friends for their valuable support and help.

- **Jaimin N. Panchal** [12MECV19]

Abstract

"Headed Gateway System Integration & Full Validation"

Electronic products have long become an integral part of the modern lifestyle. Demand and supply of such products with smaller size, lower power consumption and better performance are ever increasing. The evolution of home entertainment and computing mirrors can increasingly digital and networked lifestyle of the new millennium. A new kind of "wireless video" is currently entering consumers' homes – Digital Television. From digital TV (DTV) to Digital Video Recorder (DVR), the living room is going digital.

I-STB means Interactive Set-top Box. It gives user the flexibility of interaction with two worlds (TS based Linear TV and Internet-based Operator Services). This term has a breadth from kind of Modem (DOCSIS currently) to variety of Back-End. It provides user with variety of options for services it can address (depending on region/market).

I-STB is composed of Back End and Front End. Both are self-sufficient and independent enough. There are scenarios where the synchornization of both entities are required to take some decisisve action as per Quality Standards. To satisfy such sceanrios, focus has been given to make the system robust enough such that ongoing activities in the system are not disturbed due to faulty behaviour of one activity.
For example :
1. During Linear TV, if there is some fault in Modem, there should not be any impact on Linear TV sceanrio. User is able to view the channel without major glitch.
2. During Software upgrade, if there is issue in new image, System should be able to revert to previous good image. Scenarios may vary as per customer requirements.

Power is managed as per standard requirements. As per Quality of Service expectations of customer, BE or FE may need to go into Low Power state. Various ongoing services in I-STB need to be stopped accordingly, and when System comes out of Low power, it may resume from previous ongoing services. From Battery Management aspect, Architectural activities are going on and not finalized yet.

Linear TV is a television service where the viewer has to watch a scheduled TV program at the particular time it's offered and on the particular channel. It therefore pertains to broadcast TV programs.

• Use-Cases
Target for I-STB and derivatives is to provide a complete Home Connected Multimedia Solution with one Server and various Clients connected in the home. Server has the capability to get Video streams via RF (cable, terrestrial, etc) and transmit them to connected IP clients.

Digital video recorders are also changing the way television programs advertise products. Watching pre-recorded programs allows users to fast-forward through commercials, and some technology allows users to remove commercials entirely. This feature has been controversial for the last decade, with major television networks and movie studios claiming it violates copyright and should be banned.

Company Profile

ORGANISATION PROFILE:

STMicroelectronics is the world's fifth largest semiconductor company with net revenues of US$8.51 billion in 2009. Offering one the industry's broadcast product portfolios, ST serves customers across the spectrum of electronics applications with innovative semiconductor solutions by leveraging its vast array of technologies, design expertise and combination of intellectual property portfolio, strategic partnerships and manufacturing strength. STMicroelectronics was created in 1987 by the merger of SGS Microelectronic of Italy and Thomson Semiconductors of France with the aim of becoming a world leader in the sub-micron area. The new company pursued on aggressive growth strategy, investing Heavily in R&D, forging strategic alliances with blue-chip customers and academia, building up an integrated presence in major economic regions, and honing one of the world's most efficient manufacturing operations.

According to the latest industry data, ST is the world's fifth largest semiconductor company with market leadership in many fields. For example, ST is the leading producer of application-specific analog chips and power conversion devices. It is also the #1 supplier of semiconductors for the Industrial market and for set-top-box applications, and occupies leading positions in fields as varied as discrete devices, camera modules for mobile phones and automotive integrated circuits.

CORPORATE RESPONSIBILITY:

STMicroelectronics was one of the first global industrial companies to recognize the importance of environmental responsibility and, over the past 15 years, the Company's sites have received more than 100 awards for excellence in all areas of Corporate Responsibility, from quality to corporate governance, social issues and environmental protection. The Company's corporate responsibility policy is detailed in its Principles for Sustainable Excellence.

PRODUCT PORTFOLIO:

ST offers one of the world's broadest product ranges, with over 3,000 main types of products. The carefully balanced portfolio includes both application-specific products containing a large proprietary IP content and multi-segment products that range from discrete devices to high-performance microcontrollers. St pioneered and continues to refine the use of platform-based design methodologies for complex ICs in demanding applications such as set-top-boxes, secure smart cards And mobile multimedia, which minimizes development time and cost. The balanced portfolio approach allows ST to address the needs of all microelectronics users, from global strategic customers for whom ST is the partner of choice for major System-on-chip (SoC) projects to local enterprises that need fully-supported general-purpose devices.

MANUFACTURING MACHINE:

To provide its customers with an independent, secure a cost-effective manufacturing machine, ST operates a worldwide network of front-end (wafer fabrication) and back-end (assembly and test and packaging) plants. ST's principal wafer tabs are presently located in Agrate Brianza and Catania (Italy), Crolles, rousset and Tours (France), and Singapore. The wafer tabs are complemented by world-class assembly-and-test facilities located in China, Malaysia, Malta, Morocco and Singapore.

RESEARH AND DEVELOPMENT:

Since its creation, ST has maintained an unwavering commitment to R&D and in 2009 it spent US$2.37 billion I n R&D, which is approximately 28% of its revenue, and includes the R&D activities related to ST Ericsson, as consolidated by St. Among the industry's most innovative companies ST draws on a rich pool of chip fabrication technologies, including advanced CMOS (Complementary Metal Oxide Semiconductor), mixed-signal, analog and power processes, and is a partner in the International Semiconductor Development Alliance (ISDA) for the development of next-generation CMOS technologies.

THE KNOWLEDGE NETWORK:

ST has developed a worldwide network k of strategic alliances, including product development with key customers, technology development with customers and other semiconductor manufacturers, and equipment-and Cad-development alliances with major suppliers. These industrial partnerships are complemented by a wide range of research programs conducted with leading universities and research institutes around the world. By augmenting its rich portfolio of proprietary technologies and core competencies with complementary expertise from a variety of carefully chosen strategic partners, ST has developed an unsurpassed capability to offer leading-edge solutions to consumers in all segments of the electronics industry.

Many of ST's research and development programs are managed by its AST (Advanced System Technology) organization, whose mission is to develop the strategic system knowledge that will be required within 3-5 years by St's product divisions. Among AST's significant recent achievements are innovative technologies for digital consumer, networking, mobile security, and on-chip interconnect.

SUSTAINABLE EXCELLENCE:

ST's technical, marketing, and manufacturing strengths are matched and further enhanced by au unswerving commitment to Sustainable excellence that has earned prestigious awards around the world. Since 1991, the Company's sites have received more than 70 awards for excellence in all areas of Corporate Responsibility, from quality to corporate governance, social issues a d environmental protection.

ST's commitment to environmental responsibility has resulted in substantial reductions over the years in the consumption of energy, water, paper, and hazardous chemicals, increased recycling of waste products and a significant cut in greenhouse-gas emissions. St has constantly pushed the boundaries of excellence in Corporate Responsibility, achieving outstanding performance in key areas such as occupational health and safety- including the certification of 16 manufacturing sites and 4 non-manufacturing sites to the international standard OHSAS 18001; the application of low-power technology to its wad e product range; an bridging the digital divide through the Digital Unify Program, led by the STMicroelectronics Foundation.

FACTS AND FIGURES:

The group totals approximately 50,000 employees, 16 advanced research and development units, 39 design and application centers, 17 main manufacturing sites and 78 sales offices in 36 countries. Corporate Headquarters, as well as the head quarters for Europe and for Engineering Markets, are in Geneva. The Company's U.S. Headquarters are in Carrollton (Texas); those for Asia-Pacific are based in Singapore and Japanese operations are headquarters in Tokyo. The recently- established "Greater China" region, which includes Hong Kong, China and Taiwan, is head quartered in Shanghai.

The Company now has around 900 million outstanding shares, 72.4% of which are publicly traded on the various stock exchanges. The balance of the shares is held by STMicroelectronics Holding II B.V., a company whose shareholders are CassiaDeposite Prestiti and Finmeccanica of Italy, and Areva of France.

APPLICATIONS:

The application ground of this company deals under the following heads:

- Automotive

- Communication, Consumer and Commercial
- Computer & peripherals

Software Requirement Specification

1. INTRODUCTION

1.1 PURPOSE: The purpose of this SRS is to present a detailed description of the solutions provided by ST DVR and MEDIAPLAYER. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which the system must operate.

1.2 SCOPE:

ST delivers its own solution of Digital Video Recorder and Media Player and already has presence across the globe. The scope of this project is to do automation, benchmarking and analysis of ST's digital Video recorder and Media player solution.

1.3 DEFINITIONS AND ABBREVIATIONS:

Set-top-box: a small box sits on the standard Television which internally receives input signal coming from cable, satellite of terrestrial, and converts its content into specific form and display it onto the TV screen.

Tuner: A module or device which converts low-amplitude radio-frequency signals into a form suitable for processing by further modules or equipment. Here suitable format is error-corrected (Transport Stream) format.

TS Demux: Demultiplexes the Audio, Video and Data packets accordingly and forward them to respective decoders. The main challenging part of the Set-top-box is at TS Demux.

AV Decoder: Decodes the Audio and/or Video packets, following diagram shows the MPEG AV Decoder.

Digital Video Encoder: Encode the elementary stream data, extract actual video information and convert into proper display format like NTSC/PAL format.

Audio DAC: Same as Digital Video decoder this converts digital audio (came from digital medium) into analog and passes it to speaker into TV.

CPU with on site memory: CPU handles all the tasks with the use of SDRAM as primary memory and FLASH as secondary memory.

DVR: A digital video recorder (DVR) or Personal Video Recorder (PVR) is a device that records video in a digital format to a disk drive or other memory medium within a device.

DVR/MP: Digital Video Recorder/Media Player

1.4 OVERVIEW:

The next section, Overall Description of this document gives an overview of the functionality of the product. It also describes the informal requirements of the product. The third section, Requirement Specification of this document, is primarily written for the developer and describes in technical terms the details of the functionality of the product.

2. OVERALL DESCRIPTION

ST delivers its own solution of DVR & MP and already has presence across the globe. The scope of this project is to do automation, benchmarking and analysis of the ST's DVR & MP solution.

The key features to be targeted are:
Multiple record and playback
Time shift and catch live
Forward and backward trick modes

2.1 PRODUCT PRESPECTIVE
VCRs enjoyed a long run as the number device for recording TV. Today, digital video recorders have changed the way we watch and record our favorite TV shows and are better than VCRs in many ways. The set-top-boxes have brought a revolution in the history of television.

`2.1.1 System Interface`

To use this, the user should have a set-top-box with ST chip.

`2.1.2 Hardware Interface:`
- Mother board with set-top-box chip
- EXTERNAL data storage (SATA hard disk/ USB hard disk/ USB pen drive)
- Micro connect
- Packet injector
- Television set (SD/HD)
- UART
- MTX-100 (MPEG RECORDER AND PLAYER) with standard keyboard and mouse

`2.1.3 Communication Interfaces:`

UART cable, ETHERNET cable, HDMI cable, Audio/Video cable

2.1.4 Operations:

The key features to be targeted are:

• Multiple record and playback

• Time shift and catch live

• Forward and backward trickmodes

2.2 PRODUCT FUNCTIONS:

Set-top-boxes are very popular these days; ST provides its own set of solutions to these boxes. The DVR team is aiming at providing features like:

• Multiple record and playback

• Timeshift and catch live

• Forward and backward trickmodes

Functions of MediaPlayer are:

• Support various container formats

• Trickmodes

• Pause and seek

2.3 CONSTRAINTS

• The user has the access to ST specific operating systems.

• The user has the ST chip boards.

2.4 ASSUMTIONS AND DEPENDENCIES

• The application will work only with ST chips.

• The application will only work with ST specific operating systems.

• The user is assumed to have knowledge about the SET-TOP-BOX.

3. SPECIFIC REQUIREMENTS:

3.1 External Interface:

To have the various solutions of ST DVR and MEDIAPLAYER, one should have access to the ST boards. The input can be any program from satellite, or through the packet injector. To check the various operations, as output we need to have a television set.

3.2 Functional Requirements:

The various functions that can be performed by the DVR:

- Trickmodes
- Multiple records and playback
- Timeshift and catch-live

The various functions that can be performed by the Media Player:

- Trickmodes
- Support different container formats.
- Pause and seek operations.

3.3 Software System Attributes:

- `Reliability:` The solutions are reliable enough to support on various set top boxes.
- `Consistency:` The solutions are consistent with the growing market requirements.
- `Portability:` The solutions can be supported by operating systems that can support the hardware configuration.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

When we think about the history of television, there are a handful of events that stand out as extremely important. The invention of the Black-and-white TV set and the first broadcasts of television signals in 1939 and 1940 were obviously important. Then there is an advent of color TV and its huge popularity starting in the 1950s. Theirs is the rise of cable television and cable channels like HBO and CNN in the 1970s. In this same list must certainly go the development and popularization of the VCR starting in the 1970s and 1980s. The VCR (Video Cassette recorder) marks one of the most important events in the history of the TV because, for the first time, it gives people control of what they could watch on their TV sets. Prior to the VCR, there was no such thing as a video store.

When the VCR was first introduced to the public, the television industry reacted with panic. Then came a device that would let people record programs, watch them when they felt like as opposed to when the programming staff decided they should, and (scariest of all) skip through the commercials.

But the television industry survived despite the widespread popularity of VCRs. Now the dreaded VCR is in its death throes and a more modern innovation ahs come along that makes recording television programs even easier. The Digital Video Recorder, or DVR. The concept behind Digital Video Recorders originated in 1985, and Honeywell Inc. obtained a patent for the device in 1988.

The Digital Video Recorder (or DVR) is one of the most miraculous technological inventions of the 21st century. Digital Video Recording is a method of recording TV and Video digitally, by means of compressing the video signal using a Video Encoder, such as MPEG-2 (Motion Picture Experts Group-2), A DVR is essentially made up of two elements: the device that stores the hard disk drive and power supply,, and some type of Electronic Programming Guide (EPG) that allows the user to program recordings.

1.1 Introduction to Digital Video Recorder(DVR)

A digital video recorder (DVR) or personal video recorder (PVR) is a device that records video in a digital format to a disk drive, USB key drive, SD memory card or other memory medium within a device.

Since the video images are stored digitally, the image quality will not degrade overtime. As would a VHS tape when recorded over multiple times. There are main four types of DVR's:

- Standalone DVR's
- Computer DVR's
- Portable DVR's
- Set Top Boxes (STB) with DVR capability.

1.1.1 Stand alone DVRs

Stand alone DVRs have been made popular by brands such as TiVo which can be bought from most electronic stores and hooked up to most TV sets. These DVR's offer large storage capacities, usually coming in 30 or 60 GB's as well as fully functional TV viewing guides so that users can easily and effectively record their favorite TV show. Standalone DVR's usually cost a onetime fee for the purchase of the device and then charges an additional fee for each month of service which includes listing guide and features understanding your viewing choices and automatically recording shows for you that you might like.

1.1.2 Computer DVRs

Computer DVR's are one of the newer ways many people are recording and watching their favorite TV shows. Most people watch broadcast TV, cable TV or satellite TV on their computer LCD monitors with the help of a tuner card. Computer DVR's use the internet to find TV listings, the tuner card can pipe in the video into the computer and it is easily stored on the computer's hard drive.

1.1.3 Portable DVRs

Portable DVR's are growing in popularity and are sometimes referred to as portable media devices. Not only can they transfer and store videos, but music, photos and other types of media. Some portable DVR's usually consist of just a small function screen, USB port and hard drive. Others include an LCD screen to watch video directly on the unit.

1.1.4 Set Top Boxes with DVR capability

Cable and satellite TV companies also offer many of their customer DVR's. These DVR's come with large storage capacity for recording or storing TV shows and movies at home to each at later rime. These DVRs are usually built into set top boxes, so one box fulfills all your cable or satellite TV needs. Most cable and satellite companies allow their subscribers to rent these DVR's directly from them usually for small monthly fee. There is no additional membership for TV listing services.

1.2 Why DVR?

VCRs enjoyed a long run as the number one device for recording TV. Today, Digital video recorders have changed the way we watch and record our favorite TV shows and are better than VCRs in many ways.

• When using a VCR, you could set a program to record and then watch it at a later time. You could also watch one program while recording another. Digital video recorders with one TV tuner built-in allow users to record one show and watch a previously recorded program at the same time. DVRs with two TV tuners (the most common type today) allow you to record one program while watching another, or record two shows at the same time, all while watching a third previously recorded program. Very cool!! DVRs also give you the ability to start watching the beginning of a show while the show continues to record, something a VCR could never do.

• VCRs record to tapes while Digital Video Recorders record to a built-in hard drive. It's great not to have to use tapes; everything is recorded right in the DVR box.

• Digital Video Recorders record digital signals, while VCRs are analog machines. This means that DVRs can record digital cable and satellite channels, and more

• Importantly the ever increasing lineup of HDTV channels. While VCRs can be set up to record digital and Hi-def channels, the playback is downgraded to analog quality. Not so with a DVR, whatever quality you record in, HD, digital or analog, the playback quality is the same.

• The biggest difference that makes a DVR better than a VCR, however, is a DVR's ability to Pause and Rewind Live TV. Pause your favorite TV show when the phone rings, and resume where you let off at a later time, or rewind=d some missed dialog in a movie or a great play in the big game. With a VCR you must record a program first before you can pause, rewind of Fast-Forward.

• VCR images wear out quickly and degraded over time. They are often damaged as tapes can stretch or even tear; leaving images that simply cannot be trusted and used effectively as evidence in court. However, digitally recorded images can be stored, transferred and transmitted over networks and phone lines with no loss of image were recorded. Encrypted Watermarks and Thumbprints offer a level of security that your video image is always authentic and can be used as evidence in court.

• One more benefit of DVDs is the unprecedented control over playback. With a VCR, you have to wait for a program to finish recording before you can start watching it. Since there's no tape to rewind, digital recording doesn't have this limitation. A program that started recording 10 minutes ago can be viewed at any time, even while it's still recording.

DVR's are better than VCR's for these primary reasons:

• No more tapes to change every day.
• Better picture quality on playback.
• Instantly search through recorded images by time and date, without spending hours in front of a VCR trying to find the event you are looking for.
• Remote viewing and recording makes managing your business easy from any off-site location.
• Extended storage capabilities without long-term maintenance.

Chapter 2

Set Top Box

2.1 Introduction

A **set-top box (STB)** is a device that connects to an external signal source and Decodes that signal into content that can be presented on a display unit such as a TV.

Set Top Box or STB has become an integral part of TV viewing in many Parts of the world. We commonly see this sleek looking device sitting on side of TVs. Though this device looks slim and simple but it is one of the most complexes Embedded systems today. STBs are increasing their feature set day by day. Few Of the common features in current generation STBs are time shift mode viewing,Recording, Internet based viewing, video on demand, Full High definition video output etc.

STB is very complex embedded system; it consists of 30+ hardware blocks And similar number of software drivers. STB has lot of computing power distributed across main processor and various co-processors. In few of top end STBs if we add Operating frequencies of all co-processors then it would be in range of 3-4 GHz.

Modern day set-top boxes generally are digital devices that communicate us-In computer language. In the past when the set-top box functions were built in to another device, such as a TV, it might have been referred to as a device with a built-in. Now-a-days the phrase built-in has been superseded by the phrase "integrated". Now a TV with set-top box functionality built into it is more often called an "Integrated TV". If it's a digital TV, it would be known as an "Integrated Digital TV" (iDTV). Do note that just because a TV has set-top box functions built in to it, that doesn't mean it's a digital TV. In that case it's just an analog TV with set-top box functions built into it.

A set-top box is a computerized device that processes digital information. Set-top boxes (STB) come in many forms and can have a variety of functions. Digital Media Adapters,

Digital Media Receivers, Windows Media Extender and most video game consoles are also examples of set-top boxes. Currently the type of TV set-top box most widely used is one which receives encoded/compressed digital signals from the signal source (perhaps your cable or telecom TV provider's head end) and decodes/decompresses those signals, converting them into analog signals that your analog (SDTV) television can understand. The STB accepts commands from the user (often via the use of remote devices such as a remote control) and transmits these commands back to the network operator through some sort of return path.

Most set-top boxes deployed today have return path capability for two-way communication.

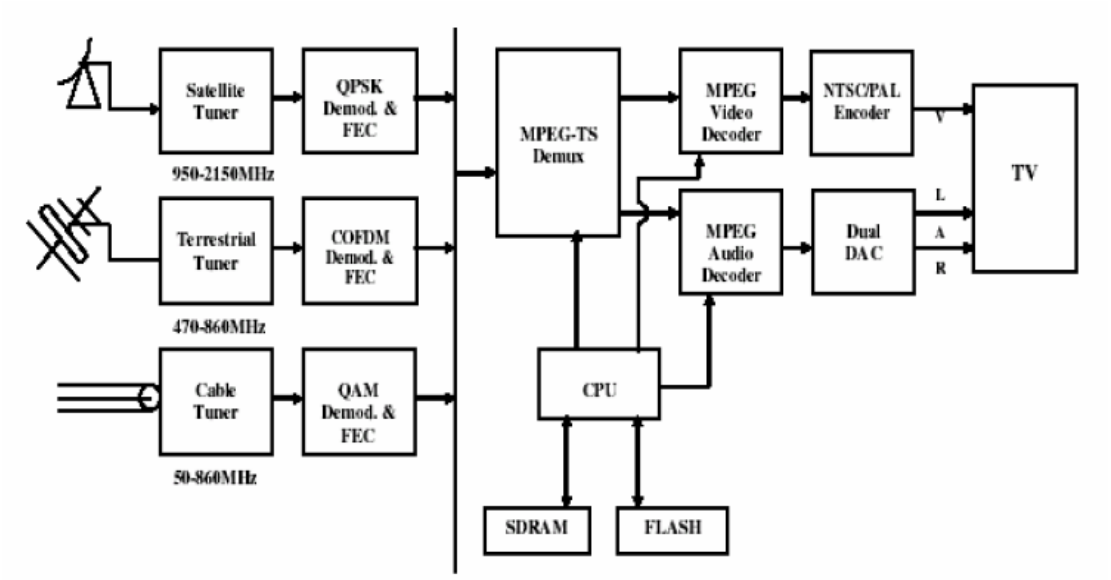

FIGURE 2.1: Block Diagram of STB

STBs can make it possible to receive and display TV signals, connect to net- works, play games via a game console, surf the Internet, and interact with Interactive Program Guides (IPGs), virtual channels, electronic storefronts, walled gardens, send e-mail, and videoconference. Many STBs are able to communicate in real time with devices such as camcorders, DVD and CD players, portable media devices and music keyboards. Some have huge hard-drives and smart card slots to put your smart card into for purchases and identification.

Generally put, to provide interactive services, the set-top box might need some or all of the below

• A network that offers the potential for interactivity.

• The network interface - This connects the STB to a network which makes it possible to communicate with the servers.

• A tuner is electronics that 'catch' the incoming signal.

• The decoder - In order to save storage space, disk bandwidth, and network bandwidth, programming is usually encoded (compressed) before being sent over the network to the STB. Thus, the end-user (subscriber) needs a decoder to decode (uncompressed among other things) the incoming stream's data before it can be viewable on the TV. This is part of what a modem does. The decoding process may be known as (or include) Demodulation (Heavy Lifting.) It could include Demultiplexing. Also see Codec. H.264 (MPEG-4)compression technology utilizes up to 40 percent less network bandwidth than the MPEG-2 compression used in most systems to date.

• The buffer - Due to delay jitters in the network, the exact arrival time of a video stream often cannot be determined. In order to guarantee continuous and consistent playback for the viewer, the video and/or data stream(s) may be received one or even a few seconds before it's actually seen by the end-user. This way if there are actuations in the transport time of the streams to that receiver (aka set-top box, decoder), the viewer won't know the difference as their buyer has a bit of time to spare.

• Synchronization software/hardware Video and audio streams must be synchronized with each other before viewing. Other streams may be added including those related to enhancements (such as metadata.)

• Middleware
• Platform
• Applications
• Any additional software and/or hardware.
• A return path (back channel).

2.2 STB Hardware Architecture

A typical STB would look similar to one shown in following image. This is picture of standard definition (SD) satellite based STB being used at my home. Number of components used are fairly less compared to complexity of this system. This credit goes to the main STB decoder chip which integrates a lot of hardware components required into a single chip.STB Blocks:
• Power Supply
• Smart Card Slot
• RAM

- STB Decoder
- Flash
- DVB-CI Slot
- Satellite Front End
- RF Modulator

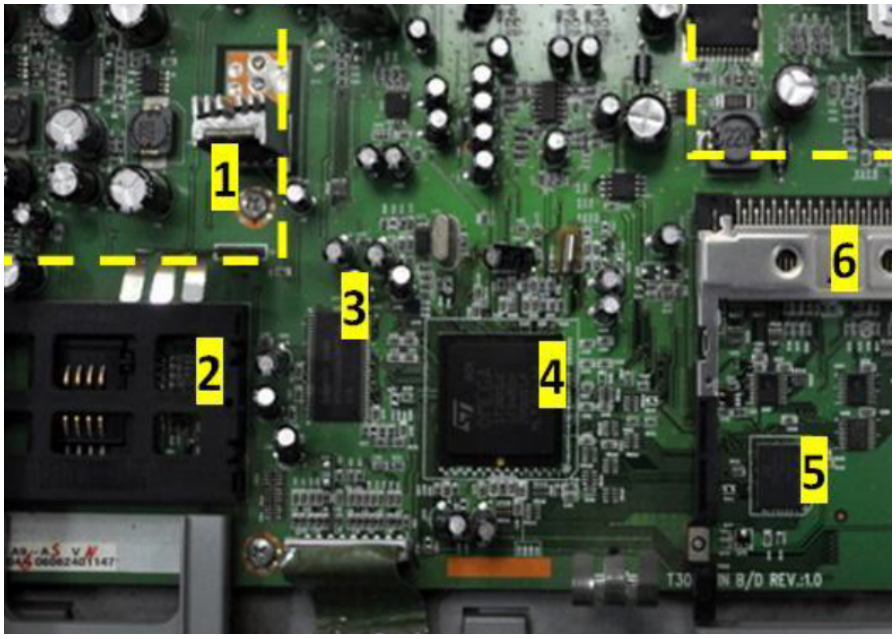

FIGURE 2.2: Inside View of STB

The above picture shows STB circuit board and major components on the board. A more logical relationship between various components is shown in following block diagram.

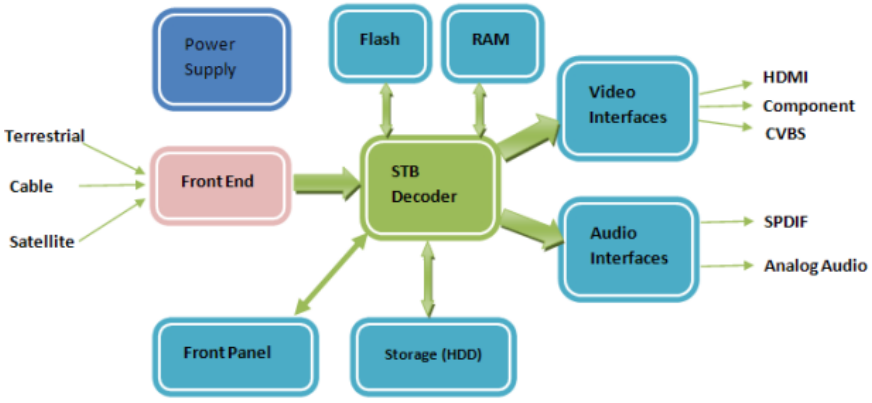

FIGURE 2.3: STB Hardware Blocks

STB Decoder: This is heart of the whole system. In current generation STBsmost of the features required by STB system are integrated in STB decoder chips. This level of integration is called System on Chip (SoC). STB SoCs contain a large number of block

ranging from de-multiplexer to decoders and peripherals like USB, SATA etc. We will discuss STB SoC in more details in next post Inside Set Top Box Part 2.

Front End: Front End part of STB is responsible for receiving the broad- casted signal, demodulating the signal and outputting digital data output for STB decoder chip. Depending upon broadcasting environment terrestrial or satellite or cable front end will be used. Front end unit consist of 3 main blocks tuner to tune correct frequency, demodulator to demodulate as per standard and forward error correction (FEC) unit for data recovery.

Power Supply: This is the main power source for board. This unit generates different voltage required by various components on board. Input to this unit can be main line AC (220/110) or DC 12V via standalone power adapters.

Flash: This is used to store boot loader, main application and other use specific nonvolatile data. Different STBs uses different sizes of ash ranging from 8MB to 64MB.

RAM: RAM is used to store all intermediate data (such as decoded video/audio buyers) and application variables. In many cases main application is also copied to RAM and is executed from RAM to speed up the operation (as RAM is faster compared to Flash). RAM size ranges from 32MB in standard definition STBs to 256MB in some top end Full HD STBs.

Video Interfaces: STB decoder chip outputs video data in analog or digital format. To make these signals compatible with external devices, special cir- circuitry like filter and physical connector are required. Current generation STBs provide many video output formats such as CVBS, S Video, and Component video HDMI.

Audio Interfaces: STB decoder chip outputs audio data in analog as well as digital format. In some cases high quality DACs are used to convert digital data into analog format. Digital data is also transmitted in digital format using SPDIF standard.

Storage: Few STBs also work as digital video recorders. To aid storage of programs some storage device (HDD) is added via any of the interfaces (SATA, eSATA, ATAPI or USB) provided by decoder chip.

Front Panel: This is STBs interface to external world. Front panels are dif- ferent for different boxes. But most of them provide IR input/output, Status LEDs, 7 segments or LCD and few switches to configure set top box. These features are controlled by parallel IOs of main decoder chip. In some cases a dedicated microcontroller is added to front end to reduce processing load for main chip and also to reduce the number of wires going from front panel to main PCB.

DVB-CI Slot: This slot is provided to support various conditional accesses schemes. Conditional Access providers provide compatible DVB-CI cards to be used with STB. The DVB-

CI card decrypts the channels encrypted by Conditional Access provider as per user's subscription policies.

Smart Card Slot: This slot is provided to use smart card for Conditional Access implementation. Unique subscriber ID is stored on each card. Smart card is also used in decrypting the channels.

RF Modulator: This is used to modulate Audio and video into RF signal. This is mainly to be used with older TVs which have only RF input and no composite (CVBS) input.

STB Decoder SoC: STB Decoder is one of the most complex systems on chip (SoC). There is generally one main processor and lot of co-processor do- in dedicated processing. A typical decoder SoC will have following blocks.

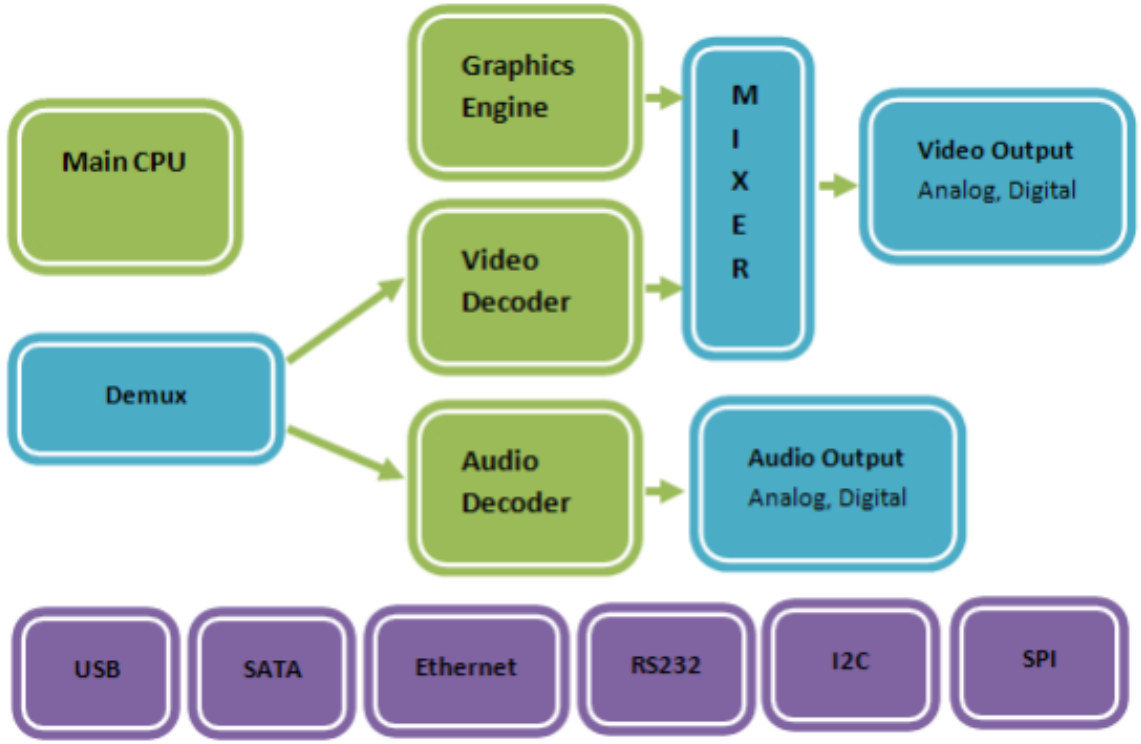

FIGURE 2.4: Set Top Box Decoder blocks

Main CPU: This is the main CPU executing the STB application. It is a general purpose CPU with lot of development tools available. Its speed range from 200MHz on standard definition devices to near 1GHz on High Definition devices. The main CPU is generally based on industry standard core such Super H or ARM to improve ease of tools and reusable stack availability.

Demux: Demux is a dedicated co-processor to de-multiplex the digital trans- port stream into audio, video and other data. Demux checks the input stream for errors and protocol compliance and filters the required data into desired buyers (Audio, video)

Video Decoder: This co-processor is responsible for converting compressed video (MPEG) data into basic video format. Current generation decoders have programmable video decoders, so video decoders can support a variety of formats such as MPEG2, H264, VC1 etc.

Graphics Engine: This co-processor is dedicated to graphics acceleration. Its main task is to draw pictures and menus for user interface (UI). This unit is becoming more powerful these days with introduction of 3D menus.

Mixer: This block is responsible for mixing the video output and graphics output and producing a signal single image. This is also responsible for ordering of video and graphics plane and transparency settings. As shown is image there are two planes one video in background on video plane and other is rectangular window showing program information on graphics plane. These two planes are mixed by mixer to generate a single image.

FIGURE 2.5: Set Top Box Mixer operation

Video Output: Final result after mixing video decoder and graphics accel- erator outputs is provided to video output block. This block is responsible for outputting data as per required standards (PAL, NTSC, SECAM, HDMI). This block generates output in analog format using DACs. This block also generates output in digital format using HDMI convertors.

Audio Decoder: Audio decoder converts the compressed audio data into basic audio data. Audio decoders are also programmable these days. They can be programmed to support any audio standard. Audio decoder generally supports MPEG, AAC, Dolby formats.

Audio Output: Audio output is fed to audio output block. This block provides audio output in analog format using internal DACs and in digital format using SPDIF convertor.

CPU (Processor): Just as in a regular PC, the processor inside the CPU takes care of interactions of all the hardware peripherals and software modules inside a set-top box. It manages the Real Time Operating System (RTOS). The processor also takes care of memory devices and other forms of digital storage like hard disks or flash drives.

Digital Storage: Digital storage is another optional component inside a set-top box. It is needed for persistent storage of any kind of data including audio/video. It communicates with the processor and is controlled by STB software modules and storage drivers. In fact, it is a hard disk which acts as a digital storage media inside an STB.

CA Module: The conditional access (CA) module can be considered as the most important component of a set-top box. This module is virtually the lifeline of all the leading set-top box solution providers. This peripheral, called Integrated Conditional Access Module (ICAM), is placed before the demultiplexer to be used for descrambling the encrypted signal, and also provide a smartcard interface for various security features.

Return Path: This is an optional component. A return path is used by an STB to communicate back with the Head End and send data packets. The return path can be present in various forms like a PSTN line connection, a cable modem in case of cable STBs or an Ethernet jack in IP set-top box using ADSL broadband modems. For example, a return path becomes a necessary component if a user needs to purchase PPV (pay per view) events directly from the set-top box using credit cards.

Peripherals: Other than main decoders and CPU a lot of peripheral devices are supported by STB SoCs for providing various features.
USB: For record/playback on external storage
SATA: Used to connect HDD for providing digital video recording facility.
Ethernet: Input source for IP based STBs
UART: Debug port. Sometimes also used for software upgrade in field.
I2C: Used by main STB SoC to communicate with external peripheral devices such as front end, SCART controller, HDMI controller etc.
SPI: Used for connecting to non-volatile storage on serial ash devices.

Typical Set Up Architecture of Set-Top Box

ST Micro Connect The kernel image compiled on the PC is loaded in to the SoC memory using the STMicro connects. ST Micro connect is a host-target interface from STMicroelectronics. It connects to a target development board's JTAG connector and provides host software with the ability to start up the target board, download programs and debug them in the target. It is easy to install and use the ST Micro Connection Package provides software utilities and firmware, including Target Packs for certain ST evaluation boards.

FIGURE 2.6: ST Micro Connect

JTAG: The JTAG cable facilitates the transfer of the kernel image from the PC to the SoC.

UART: The UART transmits the serial output from the board to the PC; it is useful in getting the debug information.

FIGURE 2.7: JTAG

TV: Serves as the receiver for digital output in various formats like HDMI, AV, SPDIF.

Digital Television: • Digital TV (DTV) is becoming an emerging consumer electronics appliance.

• It is a new way of broadcasting and is the future of Television.

• Digital Television is the successor of analog TV. All broadcasting will be done in digital format.

• Around the globe, **Satellite**, **Cable** and **Terrestrial** operators are moving to digital environment.

SATELLITE SYSTEMS: Satellite based systems deliver programs and multimedia content from broadcasters, who use a number of geostationary satellites to relay their signals to customers back on Earth. Customers must be within the "footprint" of a given satellite in order to receive the transmission. The STB's designed for receiving broadcasts from satellite based systems were the first to be deployed. It's a system which would improve reception quality and allow them to bring in services, which could not be possible with conventional TV system, and to a large extent this has governed the choice of as soon as possible used in the set top designed for their system.

CABLE SYSTEMS: In cable systems, broadcasts are sent to the home via coaxial or optical fiber based cable. In the near future, x-DSL systems will also be able to deliver these services over normal twisted pair telephone wire. Cable-based systems are beginning to ramp up significantly in volume. The rapid deployment of satellite systems is seen as an obvious threat by the cable companies. They are meeting the challenge, and are also looking to the future to how their set top boxes may evolve into the primary means for accessing the internet. This will open up the market and allow them to compete in areas normally associated with the PC systems.

TERRESTRIAL SYSTEM: In a terrestrial system, digital broadcast signals are transmitted via ground-based transmitters in exactly the same way as analog television signals are transmitted. In fact, in the majority of cases exactly the same aerial can be used. Systems designed for terrestrial systems are limited in terms of the number of channels they can offer compared to both satellite and cable based systems. The modulation scheme required is more complex than that required for cable or satellite. In this system the data is spread over a number of frequency channels. A concatenated error correction system is used, and the use of "guard intervals" is also employed in order to ensure as robust a scheme as is practical.

• DTV will provide cinema quality pictures, CD quality sound and hundreds of new channels.

• A small box sits on top of a standard TV set is called Set-top Box (STB).

• STB is central to this migration from analog-to- digital broadcasting.

• STB will become a gateway to the digital information super highway.

For digital broadcasting, TV signal is:
1. Digitized.
2. Compressed and.
3. Digitally modulated.

Digitization:
• Digitization of TV signal is carried out using Pulse code Modulation (PCM)
• Digital signal consists of 1's and 0's which are not affected by interference and noise.
• Digital signals are easy to process by standard techniques.
• PCM process needs.
1. Sampling.
2. Quantization and.
3. Encoding.
• Sampling determines samples corresponding to instantaneous amplitude of the input signal at uniform intervals.
• Input signal is divided into number of levels.
• Quantization allocates levels to the amplitude of sample values.
• Each sample peak falls within some specific level.
• This value is translated into binary code using encoder.
• N = Log2 (Number of levels) where N is number of bits
• Total bit rate = Fs X N
• For example: for TV signal Fm=5MHz with N=8 bits
• Then bit rate= 13.5X8=108Mb/s (CCIR 601 standard)
• After digitization the BW is increased by 21 times.

The details of compression would be elaborately put forth in subsequent chapters. Before divulging in details, the basic operation of SET-TOP box would be placed to keep the driving force and the whole working environment intact in the mind of the readers. The important point to keep in mind is:
• **QPSK** demodulator is used in case of **Satellite** transmission.
• **OFDM** demodulator is used in case of **Terrestrial** transmission.
• **QAM** demodulator is used in case of **Cable** transmission.

Flow of Data: Let's move on to the next level, where we'll go through the flow of data between this hardware while we tune an STB to a particular channel. The tuner receives modulated digital transmission from the antenna and passes it on to the demodulator.

This demodulator takes into account the type of demodulation (like QPSK) and forward error correction to give out a transport stream, which is a digital stream of bytes known as data

packets. According to DVB standards, a transport stream is of the size of 188 bytes. It then goes into the Demux where the content is separated in audio/video packetized elementary stream (PES) and data packets known as sections, as per DVB standards.

PES is a mechanism to carry audio/video elementary streams in packet format inside an MPEG-2 transport stream. Sections are the data packets containing information regarding the audio/video content and other metadata. Once the Demux does its job, audio/video is sent to MPEG-2/MPEG-4 decoder which gives the output to the Report to display the video on television.

The data packets are sent to the processor used by the STB software to enable viewing. This data can be persistent or kept in RAM as per the needs and performance. Any kind of user request, be it tuning or a purchase, goes through the CPU. In between, the most important functionality of descrambling is performed by a descrambler embedded inside the ICAM part of the chipset. In some variants of STBs, a descrambler can be a part of a decoder, or it can exist independently.

The descrambler takes care of decrypting the encrypted transport stream using the control word technique. Generally, these descramblers and control word algorithms are closely-guarded secrets with CA solution providers to prevent hackers from decrypting the signal. These are developed in conjunction with chipset vendors in a much secured environment.

STB Software Architecture: STB Software is organized as layered architecture as shown in block diagram below. RTOS is generally STB company proprietary or some industry standard OS such as Linux. Software drivers are written for all hardware blocks and some software components. A typical STB has 30-40 different drivers.

Middleware is generally used to standardize the interfaces from drivers to application so that device independence can be provided. Final application is on top of middleware and it usually remains same for one service provider across different STBs. There are normally two applications loaded on each STB. One is the boot loader and other is main application. Boot loader is responsible for downloading main application Over the Air (Application broadcasted over air by service provider) and updating the main application. The main application is responsible for all the features/functionality which the end user sees. STB software application has very high complexity. Atypical STB application including drivers consists of 0.5-0.6 million lines of code.

So STB does a great job of hiding this advance level of complexity in its simple form factor. Its application is also designed by keeping various users in mind, so the complexity of software is completely hidden from end users perspective.

TABLE 2.1: Software Architecture of STB

Application
Middleware
Driver Layer
OS
Hardware

Software Architecture of Set Top Box

Drivers and OS: An operating system is the most important piece of software in a STB. An OS is a suitable of programs used to manage the resources in a STB. In particularity is the OS, which talks to the STB hardware and manage their functions such as scheduling real time tasks, managing limited memory resources, etc. A STB OS is arranged in layers with each layer adding new capability. At the heart of any STB OS is the "Kernel" layer, which is stored in ROM. Once the STB is powered up, the kernel will be loaded first and remains in memory until the STB is powered down again. Typically the kernel is responsible for managing memory resources, real time applications and high-speed data transmission. The kernel supports multi-threading and multi-tasking which allows a STB to execute different sections of a program and different programmers simultaneously. The STB also requires 'drivers' to control the various hardware devices. Every hardware component in the STB must have a driver. A driver is a program that translates commands from the TV viewer to a format that is recognizable by the hardware device.

Finally a STB OS needs to incorporate a set of Application Program Interfaces which are used by the programmers to write high-level applications for a specific API. An API is basically a set of building blocks used by software developers to write programs that are specific to a STB OS environment.

Middleware layer: Central to the new software architecture of a STB is a connection layer that acts as a communications bridge between the OS and the 'subscriber applica- tions' called 'Middleware'. Middleware is a relatively new term in the set top business. It represents the logical abstraction of the middle and upper layers of the communication software stack used in set top software and communication system. Middleware is used to isolate set top application programs from the details of the underlying hardware and network components. Thus set top applications can operate transparently across a network without having to be concerned with the underlying network protocols. This considerably reduces the complexity of content development because applications can be written to take advantage of a common API. The terms API (Application Programmers Interface) and middleware are sometimes interchangeably used. The API is the standard environment that an application program

expects to see. The API itself consists of a set of well-defined and specified functions accessed using a well-defined and specified called mechanism. Early generation of STBs had no APIs but only a very basic operating system. As costs have fallen and processing power has increased, more recent STBs have included APIs. In order to progress beyond ordinary broadcasting to the new emerging interactive services an API is essential.

Application Layer: All the applications that run in a STB can broadly be classified into two main categories Enhanced and Interactive. An Enhanced TV application is the one which is based on 'local interactivity' and which does not require a return path back to the service provider. As opposed to this an interactive application is based on 'two way interactivity'. Here the viewer issues a request for extra information to the service provider, which travels along a return path and the service provider sends the requested data back either via the return path itself or 'over the air'. A good example of this would be calling up a home shopping application via the TV screen.

Set-top boxes may be associated with these major categories

Broadcast TV Set-top Boxes: (a.k.a. Thin Boxes) - A more primitive set-top box with no back channel (return path.) These might come with interface ports, some memory and some processing power.

Enhanced TV Set-top Boxes: (May be known as: Smart TV Set-top Box, Thick Boxes) These have a back channel (return path), often through a phone line. These may be capable of Video on Demand, e-commerce, Internet browsing, e-mail communications, chat and more.

Advanced Set-top Boxes: (a.k.a. advanced digital Set-top boxes, Smart TV Set-top Box, Thick Boxes, All-in-one Set Top Box, Media Center) - A fully integrated set-top box. These have good processors, memory, middleware, software applications and optional hard-drives. They're often used with high- speed (broadband) connections. Features could include high-speed Internet access, Interactive TV, digital video and gaming. Instead of this, a "sidecar" might be used in tandem with the set top box and/or TV. Advanced set-top boxes are more likely to be integrated with DVRs and high-definition TV.

Sidecar: This type of set-top box provides an additional transport stream of data from the network operator to compliment the main stream. With Charter Communications, the BMC-8000 (Broadband Media Center) is/was a sidecar box that works in tandem with the Motorola DCT-2000. A fully integrated unit would not require a Sidecar.

Hybrid Digital Cable Box: A Hybrid Digital Cable Box is a specialized cable TV set-top box with high end functions. Motorola Broad bands DCP501 home theater system is/was

an example. It has/had a DVD player and high-end stereo output. This term may be antiquated.

Over-the-top Boxes: Electronic device manufacturers are providing DVD play- ers, video game consoles and TVs with built- in wireless connectivity. These devices piggy back on an existing wireless network and pull content from the Internet and deliver it to the TV set. Typically these devices need no ad- additional wires, hardware or advanced knowledge in how to operate. Content suited for TV can be delivered via the Internet. These OTT applications include Facebook and YouTube. Also see Internet-connected TV.

Chapter 3

GStreamer

3.1 GStreamer

GStreamer is a framework for creating streaming media applications. GStreamer's development framework makes it possible to write any type of streaming multimedia application. The GStreamer framework is designed to make it easy to write applications that handle audio or video or both. It isn't restricted to audio and video, and can process any kind of data flow. The pipeline design is made to have little overhead above what the applied filters induce. This makes GStreamer a good framework for designing even high-end audio applications which put high demands on latency.

One of the most obvious uses of GStreamer is using it to build a media player. GStreamer already includes components for building a media player that can support a very wide variety of formats, including MP3, Ogg/Vorbis, MPEG-1/2, AVI, Quicktime, mod, and more. GStreamer, however, is much more than just another media player. Its main advantages are that the pluggable components can be mixed and matched into arbitrary pipelines so that it's possible to write a full-fledged video or audio editing application.

The framework is based on plugins that will provide the various codec and other functionality. The plugins can be linked and arranged in a pipeline. This pipeline defines the flow of the data. Pipelines can also be edited with a GUI editor and saved as XML so that pipeline libraries can be made with a minimum of effort.

Specifically, **GStreamer provides:**

• an API for multimedia applications

- a plugin architecture
- a pipeline architecture
- a mechanism for media type handling/negotiation
- a mechanism for synchronization
- over 250 plug-ins providing more than 1000 elements
- a set of tools

GStreamer plug-ins could be classified into:

- protocols handling
- *Sources:* for audio and video (involves protocol plugins)
- *Formats:* parsers, formatters, muxers, demuxers, metadata, subtitles
- *Codecs:* coders and decoders
- *Filters:* converters, mixers, effects, ...
- *Sinks:* for audio and video (involves protocol plugins)

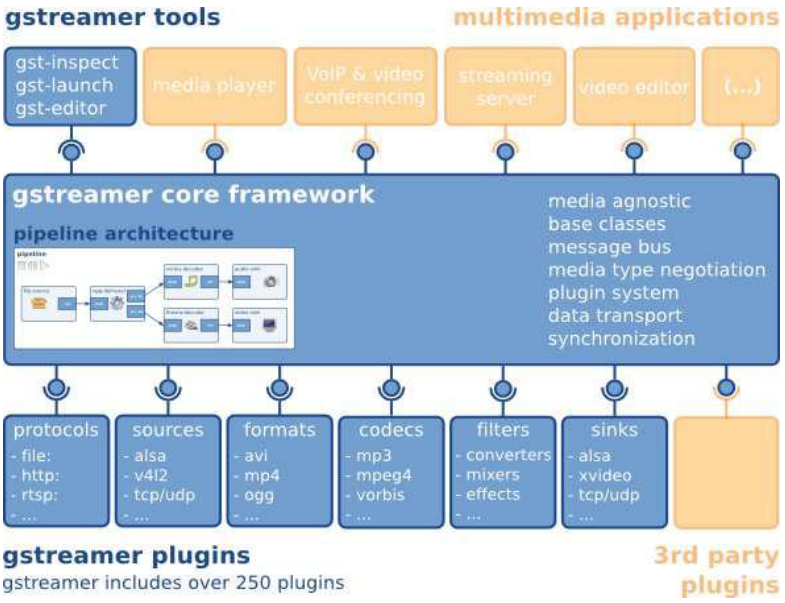

FIGURE 3.1: GStreamer

GStreamer is packaged into:

- *gstreamer:* the core package
- *gst-plugins-base:* an essential exemplary set of elements
- *gst-plugins-good:* a set of good-quality plug-ins under LGPL
- *gst-plugins-ugly:* a set of good-quality plug-ins that might pose distribution problems

- *gst-plugins-bad:* a set of plug-ins that need more quality
- *st-libav:* a set of plug-ins that wrap libav for decoding and encoding a few others package.

Chapter 4

Transport Stream

4.1 Transport Stream

Transport Stream is an audio, video and data communication transmission protocol that is specified in MPEG-2. TS allows multiplexing of digital video and audio, which means the data is combined in to a single synchronous transmission bit stream for transmission over a variety of standard mediums such as DSL, cable TV network. The TS is only designed to address only delivery. Storage application uses the program stream. Figure shows transport stream.

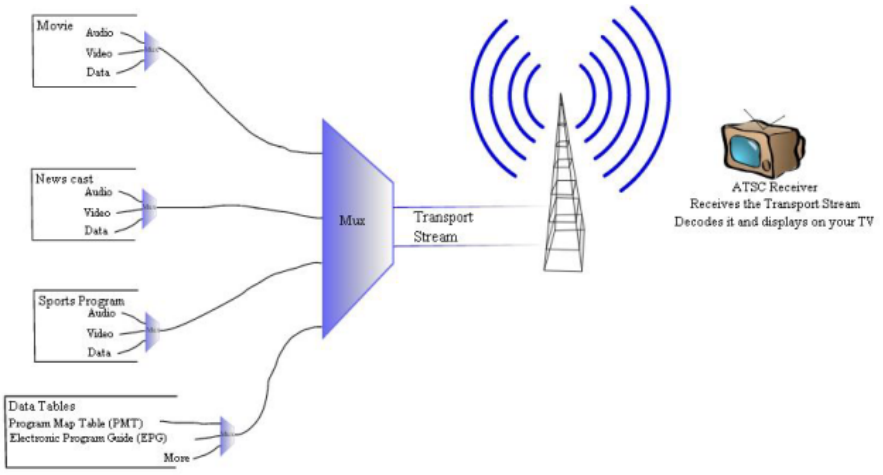

FIGURE 4.1: Transport Stream

4.2 The MPEG-2 standard allows two forms of multiplexing

4.2.1 MPEG Program Stream

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. This form of multiplexing is used for video playback and for some network applications.

4.2.2 MPEG Transport Stream

Each PES packet is broken into fixed-sized transport packets forming a general purpose way of combining one or more streams, possibly with independent time bases. 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. The Program Stream is widely used in digital video storage devices, and also where the video is reliably transmitted over a network (e.g. video-clip download). Digital Video Broadcast (DVB) uses the MPEG-2 Transport Stream over a wide variety of underlying networks. Since both the Program Stream and Transport Stream multiplex set of PES inputs, interoperability between the two formats may be achieved at the PES level.

4.2.2.1 MPEG Transport Streams

A transport stream consists of a sequence of fixed sized transport packet of 188 bytes. Each packet comprises 184 bytes of payload and a 4 byte header. One of the items in this 4 byte header is the 13 bit Packet Identifier (PID) which plays a key role in the operation of the Transport Stream.

The format of the transport stream is described using the Figure below (a later section describes the detailed format of the TS packet header). This Figure shows two elementary streams sent in the same MPEG-2 transport multiplex. 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 than audio packets, but you may also note that the 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, 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 (and usually is)slightly different. A separate process is required to synchronize the two streams. The audio and video packets in TS shown below. Figure shows audio and video packets in TS.

FIGURE 4.2: Audio and Video Packets in TS

4.2.2.2 Transmission of MPEG-TS

Although the MPEG TS may be directly used over a wide variety of media (as in DVB), it may also be used over a communication network. It is designed to be robust with short frames, each one being protected by a strong error correction mechanism. It's constructed to match the characteristics of the generic radio or cable channel and expects an uncorrected Bit Error Rate (BER) of better than 10-10. (The different variants of DVB each have their own outer coding and modulation methods designed for the particular environment.) The MPEG-2 Transport Stream is so called, to signify that it is the input to the Transport Layer in the ISO Open System Interconnection(OSI) seven-layer network reference model. It is not, in itself, a transport layer protocol and no mechanism is provided to ensure the reliable delivery of the transported data. MPEG-2 relies on underlying layers for such services.

4.2.2.3 Single and Multiple Program Transport Streams

A TS may correspond to a single TV program, or multimedia stream (e.g. with a video PES and an audio PES). This type of TS is normally called a Single Program Transport Stream (SPTS).An SPTS contains all the information required to reproduce the encoded TV channel or multimedia stream. It may contain only an audio and video PESs, but in practice there will be other types of PES as well. Each PES shares a common time base. In the DVB case, one or more SPTS streams are combined to form a Multiple Program Transport Stream (MPTS). This larger aggregate also contains all the control information (Program Specific Information (PSI)) required to co-ordinate the DVB system, and any other data which is to be sent.

Figure shows MPEG-2 TS layered architecture.

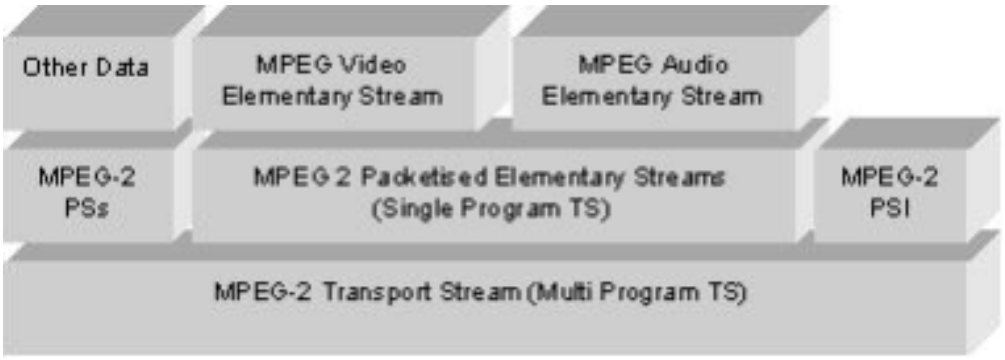

FIGURE 4.3: MPEG-2 TS Layered Architecture

Most transport streams consist of a number of related elementary streams (e.g. the video and audio of a TV program). The decoding of the elementary streams may need to be co-ordinated (synchronized) to ensure that the audio playback is in synchronism with the corresponding video frames. Each stream may be tightly synchronized(usually necessary for digital TV programs, or for digital radio programs), or not synchronized (in the case of programs offering downloading of software or games, as an example). To help synchronization time stamps are sent in the transport stream.

They are two types of time stamps:

1. The first type is usually called a reference time stamp. This time stamp is the indication of the current time. Reference time stamps are to be found in the PES syntax (ESCR), in the program syntax (SCR), and in the transport packet adaption Program Clock Reference (PCR) field.

2. The second type of time stamp is called Decoding Time Stamp (DTS) or Presentation Time Stamp (PTS). These time stamps are inserted close to the material to which they refer (normally in the exact moment where a video frame or an audio frame has to be decoded or presented to the user respectively. These rely on reference time stamps for operation.

4.2.2.4 Signaling Tables

For a user to receive a particular transport stream, the user must first determine the PID being used, and then filter packets which have a matching PID value. Because VBrick sends a Single Program Transport Stream, we have a special capability to automatically detect the PID and automatically configure the decoder to display the correct content. To help the user identify which PID corresponds to which program, a special set of streams, known as Signaling Tables, are transmitted with a description of each program carried within the MPEG-2 Transport Stream.

Signaling tables are sent separately to PES, and are not synchronized with the elementary streams (i.e. they are an independent control channel).

The tables (called Program Specific Information (PSI) 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. Each PSI table is carried in a sequence of PSI Sections, which may be of variable length (but are usually small, c.f. PES packets).

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. Figure shows packet consisting of PID, PAT, PMT.

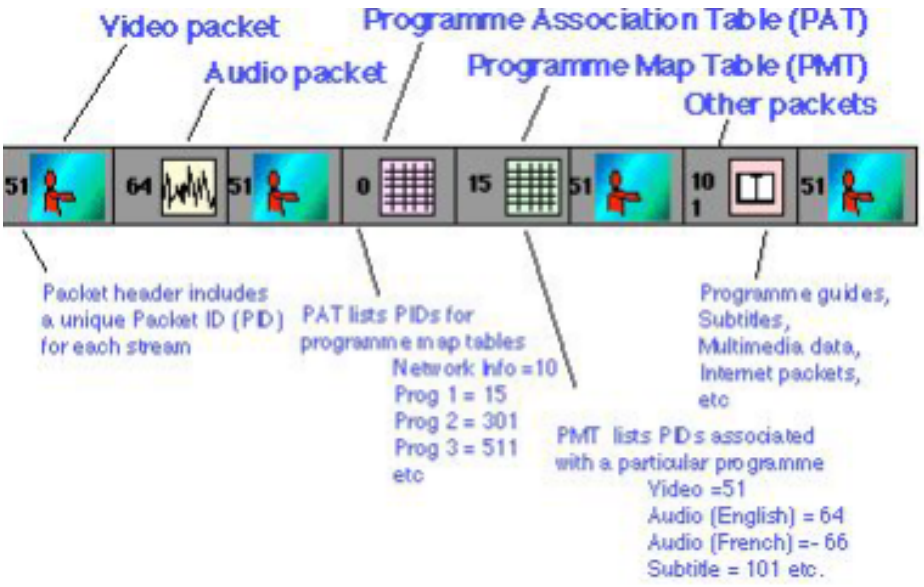

FIGURE 4.4: Packet Consisting of PID, PAT,PMT

4.2.2.5 MPEG-2 Signaling Tables

PAT: Program Association Table (lists the PIDs of tables describing each program).The PAT is sent with the well- scrambling used and PID values of transport streams which contain the conditional access management and entitlement information (EMM)). The PAT is sent with the well-known PID value of 0x001.

PMT: Program Map Table defines the set of PIDs associated with a program, e.g. audio, video, etc.

NIT: Network Information Table (PID=10, contains details of the bearer network used to transmit the MPEG multiplex, including the carrier frequency).

DSM-CC: Digital Storage Media Command and Control (messages to the receivers) Program Service Information (SI) provided by MPEG-2 and used by DVB. 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.

Format of a Transport Stream Packet: Each MPEG-2 TS packet carries 184 byte of payload data prefixed by a 4 byte (32 bit) header. A TS header structure is shown in below.

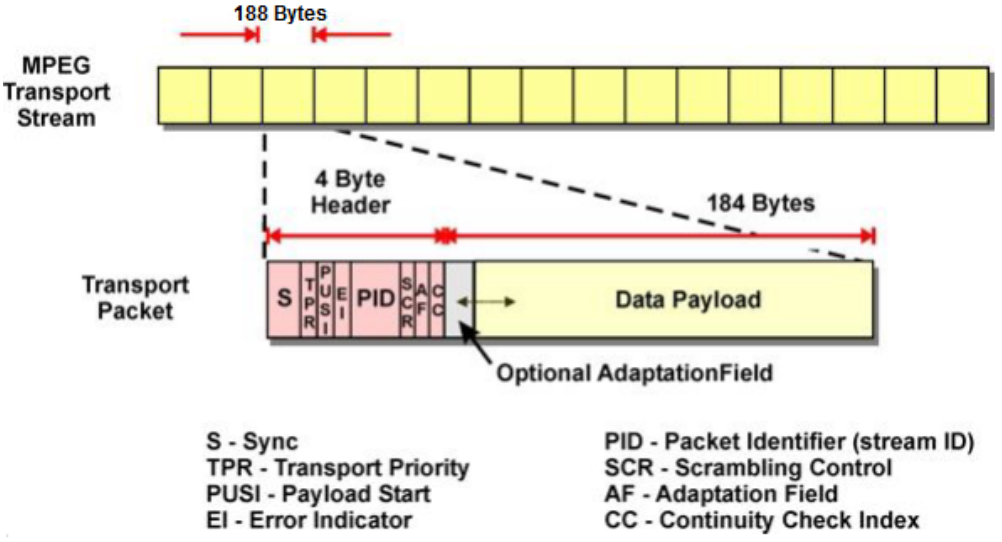

FIGURE 4.5: Transport Stream Header Structure

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 adaption field control bits which may take four values.

1. 01 – no adaptation field, payload only
2. 10 – adaptation field only, no payload
3. 11 – adaptation field followed by payload
4. 00 - reserved for future use

Finally there is a half byte Continuity Counter (4 bits). Two options are possible for inserting PES data into the TS packet payload:

The simplest option, from both the encoder and receiver 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.

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 is starting, however, 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. Figure shows PES header structure.

Option 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 B packet header, before any user payload data. It may contain a variety of data used for timing and control.

One important item in most adaption packets is the Program Clock Reference (PCR) field. 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.

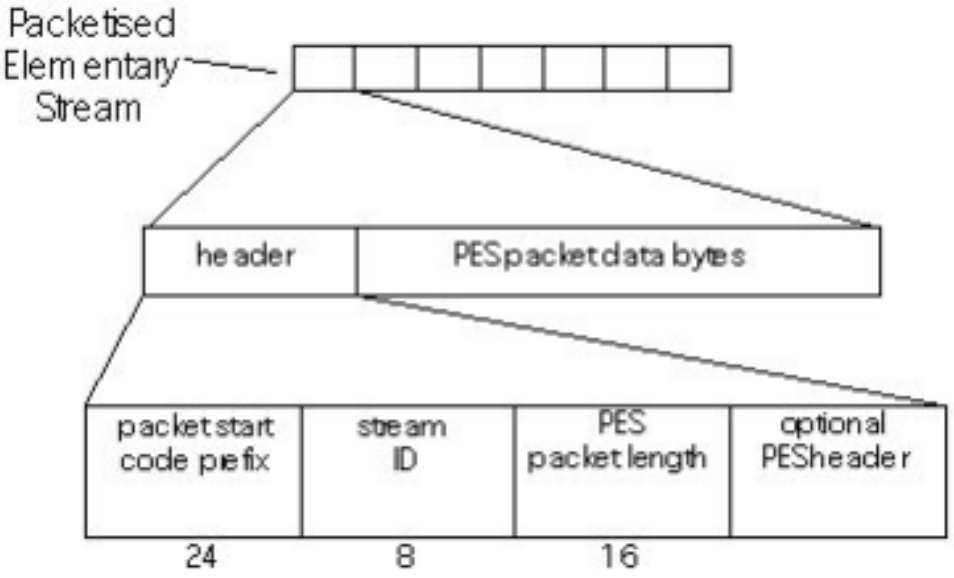

FIGURE 4.6: PES Header Structure

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. A typical representation of PAT, PMT, NIT, CAT is shown in Figure below.

Chapter 5

Digital Video Broadcasting

5.1 Digital Video Broadcasting

DVB, short for **Digital Video Broadcasting**, is a suite of internationally accepted, open standards for digital television. DVB standards are maintained by the DVB Project, an industry consortium with more than 300 members, and published by an *Joined Technical Committee (JTC)* of European Telecommunications Standards Institute (ETSI), European Committee for Electro technical Standardization (CENELEC) and European Broadcasting Union(EBU). The interaction of the DVB sub-standards is described in the DVB cookbook. Many aspects of DVB are patented, including elements of the MPEG video coding and audio coding.

However, the DVB is not itself a standards making body; it provides a forum for suppliers to agree specifications which are then passed to existing standards making bodies (ETSI, ISO) for ratification DVB passes so called "commercial modules" requirements to "technical modules "And not the other way around.

The core standards of DVB are:
• DVB-S (satellite)
• DVB-C (cable) and
• DVB-T (terrestrial)

All the core standards are based upon MPEG-2 **(DVB-MPEG)** for audio and video coding as well as the transport stream and are capable of high definition television (HDTV) as well.

The encoder processing in DVB-C, DVB-S and DVB-T is based on the same fundamental concept. These flavors differ mainly in the modulations used. The high frequency DVB-S

uses QPSK, DVB-c uses QAM (64-QAM in general) and DVB-T(in VHF and/or UHF band) uses COFDM.

New standard upcoming is **DVB-H** for mobile reception in cellular phone frequency bands.

Besides audio and video transmission, DVB also defines data connections (DVB-DATA) with return channels DVB-RC for several mediums (GSM, PSTN/ISDN etc.) and protocols (DVB-IP : Internet Protocol, DVB-NPI : network protocol independent).

DVB describes a lot of (network) interfaces, but most importantly the Common Interface (DVB-CI) for Conditional Access (DVB-CA) with the Common Scrambling Algorithm(DVB-CSA) required for (de-)scrambling pay TV.

5.1.1 DVB-S

DVB-S is the original Digital Video Broadcasting forward error coding and modulation standard for satellite television and dates from 1995. It is used via satellites serving every continent of the world. DVB-S is used in broadcast network feeds, as well as for direct broadcast satellite services. The transport stream delivered by DVB-S is mandated as MPEG-2.

In particular it describes the modulation and channel coding system for satellite digital multi programme Television(TV)/ High Definition Television(HDTV) services to be used for primary and secondary distribution in fixed satellite services(FSS) and Broadcast Satellite Service(BSS) bands.

DVB-S is intended to provide Direct-To-Home (DTH) services for consumer Integrated Decoders (IRD), as well as collective antenna systems and cable television head-end stations. DVB-S is suitable for use on different Satellite transponder bandwidths and is compatible with Moving Pictures Experts Group 2(MPEG 2) coded TV services. Flexibility defined within the specification enables the transmission capacity to be used for a variety of TV service configurations, including sound and data services.

Digital Video Broadcasting : Satellite- Second Generation (DVB-S2) is an enhanced specification to replace the DVB-S standard, developed in 2003 and ratified by ETSI (EN302307) in March 2005. DVB-S2 will probably be used for all future new European digital satellite multiplexes, and satellite receivers will be equipped to decode both DVB-S and DVB-S2. Today the main use for this new standard is the distribution of HDTV while the original standard was used mainly for SDTV services. The development of DVB-S2 coincided with the introduction of HDTV and H.264(MPEG-4) video codecs.

The system allows transmission of one or more MPEG-2 audio/video streams, using QPSK or 8PSK or MAPSK (M-ary amplitude and phase-shift keying) modulation with concatenated encoding. DVB-S2 is based on the DVB-S standard which is used for satellite broadcasting, and the DVB-DSNG standard, which is used by mobile units for sending external footage back to television stations.

Two new key features which were added to DVB-S are:
1. Changing encoding parameters in real time (VCM, Variable Coding and Modulation)
2. ACM (Adaptive Coding and Modulation) which optimizes the transmission parameters for various users.

5.1.2 DVB-T

DVB-T stands for Digital Video Broadcasting- Terrestrial and it is the DVB European consortium standard for the broadcast transmission of digital terrestrial television. This system transmits a compressed digital audio/video stream, using OFDM modulation with concatenated channel coding(i.e. COFDM). The adopted source coding methods are 2 and, more recently, H.264.

In June 2006, a study group named TM-T2 (Technical Module on Next Generation DVB-T) was established by the DVB group to develop an advanced modulation scheme that could be adopted by a second generation digital television standard, to be named DVB-T2.

According to the commercial requirements and call for technologies issued in April 2007, the first phase of DVB-T2 will be devoted to provide optimum reception for stationary (fixed) and portable receivers (i.e. units which can be nomadic, nut not fully mobile) using existing aerials, whereas a second and third phase will study methods to deliver high payloads (with new aerials) and the mobile reception issue. The novel system should provide a minimum 30 percentage increase in payload, under similar channel conditions already used for DVB-T. expected technologies will probably include:

• LDPC coding in compliance with the technique already adopted in the DVB-S2 satellite standard.
• MIMO and antenna diversity systems.
• More than 8k carriers (a requirement being to provide a 30 percentage increase in the size of single frequency networks);
• Flexible multiplexing;
• Variable coding and modulation.

5.1.3 DVB-C

DVB-C stands for Digital Video Broadcasting – Cable and it is the DVB European consortium standard for the broadcast transmission of digital television over cable. This system transmits an MPEG-2 family digital audio/video stream, using a QAM modulation with channel coding.

In 2007 a study mission of the DVB Technical Module produced a report identifying some possible technologies which considered as alternatives succeeding the existing DVB-C specification. The DVB-TM-C2 ad-hoc was requesting the submission of proposals for technologies which could be considered as candidates for a second generation DVB cable transmission system DVB-C2.

5.1.4 DVB-H

A more flexible and robust digital terrestrial system DVB-H has also recently been developed. The system is intended to be receivable on handheld receivers. DVB-H services will also use more efficient video compression systems such as MPEG-4 AVC (Advanced Video Coding). DVB-H is an extension of DVB-T with some backwards compatibility, i.e. it can share the same DVB-T multiplex.

Chapter 6

Digital Video Recorder

6.1 Digital Video Recorder

When the VCR was first introduced to the public, the television industry reacted with panic. Here was a device that would let people record programs, watch them when they felt like it as opposed to when the programming staff decided they should, and (scariest of all) skip through the commercials!

But the television industry survived despite the widespread popularity of VCRs. Now the dreaded VCR is in its death throes and a more modern innovation has come along that makes recording television programs even easier: the **Digital Video Recorder or DVR**.

A **Digital Video Recorder (DVR)** or **Personal video recorder (PVR)** is a device that records video in a digital format to a **disk drive, USB key drive, sd memory card or other memory** medium within a device. The television signal comes into the DVR's **built-in tuner** through antenna, cable or satellite, if the signal comes from antenna or cable, it goes into an **MPEG-2 Encoder**, which converts the data from analog to digital (**MPEG-2**, by the way, is the compression standard used to fit information onto a DVD).form the encoder, the signal is shipped off to two different places: first, to the **hard drive** for storage, and second, to an **MPEG-2 decoder**, which converts the signal back to analog and sends it to the television for viewing.

Some systems use dual tuners, allowing users to record different programs on different channels at the same time. On a few systems, you can even record two programs while watching a third pre-recorded show.

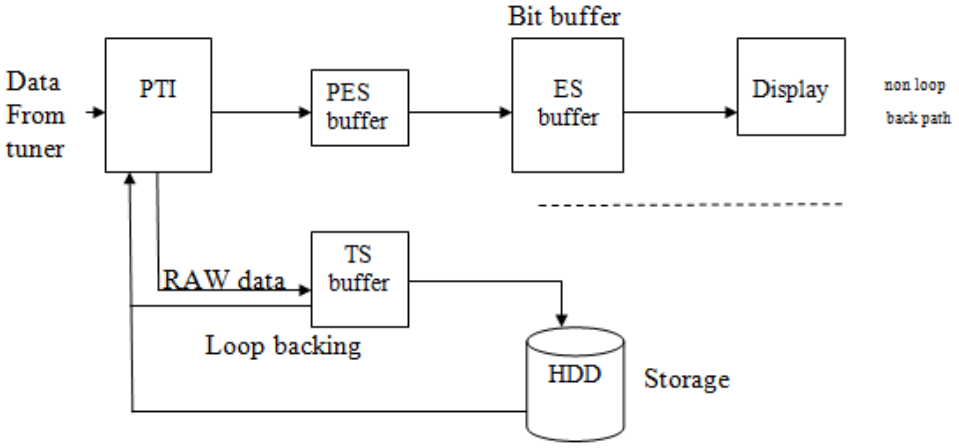

FIGURE 6.1: Block Diagram

6.2 DVR Operations

6.2.1 Recording:

We can record the live program and watch it at some later instance. We can pause the live and start recording the live. We can also record two programs at the same time as well as watch one live program and record another program.

6.2.2 Play Back

We can playback the Live, recorded. Also can play while recording is going on.

6.3 Features of DVR

Digital Video Recorder supports the features, such as, Live playback, recording and then its playback, Trickmodes etc. The various features supported are:

1 Live Playback

2 Playback from Local Storage

3 Record

4 Dual Record

5 Live and Record (same channel)

6 Live and 2 Records

7 Play and record

8 Play and 2 Records

9 Timeshift and record

The working of live playback is shown in Fig.

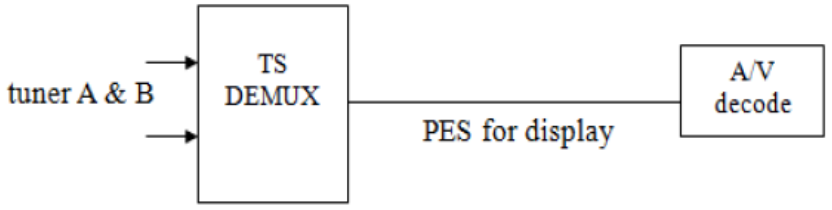

FIGURE 6.2: Working of Normal Live Playback

Normal live Playback:

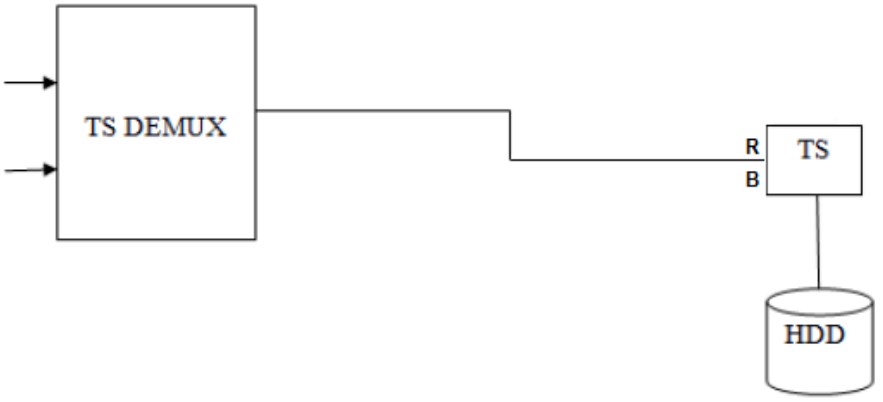

FIGURE 6.3: Rocord

6.4 Timeshift

Timeshift means program to be displayed on the TV with a delay as compared to LIVE. This operation is performed in three steps:

- Live pause
- Live Resume
- Catch Live

After resuming Live, timeshifted playback is shown; and after catching Live, we get the live playback but still from the hard disk.

Live Pause: when Live is paused, the program starts recorded in the hard disk. When Live is paused, the coming program starts recording in the hard disk.

Live Resume: When Live is resumed, playback starts from the hard disk recorded file, exactly from the location where we left and recording still going on in background.

Catch Live: If we want to catch Live, we can do this by doing some trickmodes. After catching live, we still playing our program from the hard disk.

Timeshift operation can be performed in combination with other applications like Timeshift and record, Timeshift and Live Playback.

6.5 Trickmodes

We call trickmode the playback of a video file at any speed different than the nominal speed. This includes slow or fast playback in forward or backward direction. Smooth trickmode is a trickmode where all frames are displayed. We speak about scan mode when only I frames are decoded and displayed. The audio is always muted during trickmodes.

Trickmode can be simply stated as " setting speed or changing the speed of the current playback". This is implemented in both DVR and MEDIAPLAYER. The speed can be set in both, the forward and backward, for a recorded playback.

Chapter 7

Testing

7.1 Testing

A primary purpose of **Testing** is to detect software failures so that defects may be discovered and corrected. Testing cannot establish that a product functions properly under all conditions but can only establish that it does not function properly under specific conditions. The scope of software testing often includes examination of code as well as execution of that code in various environments and conditions as well as examining the aspects of code: does it do what it is supposed to do and do what it needs to do. In the current culture of software development, a testing organization may be separate from the development team. There are various roles for testing team members. Information derived from software testing may be used to correct the process by which software is developed.

7.2 Different types of testing

7.2.1 Functional vs. Non-functional Testing

Functional testing refers to activities that verify a specific action or function of the code. These are usually found in the code requirements documentation, although some development methodologies work from use cases or user stories. Functional tests tend to answer the question of "can the user do this" or "does this particular feature work."

Non-functional testing refers to aspects of the software that may not be related to a specific function or user action, such as scalability or other performance, behavior under certain constraints, or security. Testing will determine the flake point, the point at which extremes of scalability or performance leads to unstable execution. Non-functional requirements tend

to be those that reflect the quality of the product, particularly in the context of the suitability perspective of its users.

7.2.2 Full Sub-system Testing

Full sub-system testing is done for each new release of SDK after low level code freeze to avoid regressions in compared to previous release. This is customer level testing and it is done so that customer face less issue at own end.

We can say this also unit testing. Our full sub-system testing is divided in different types:
- Mediaplayer
- DVR
- IP

7.2.3 Mediaplayer Testing

It is also categorized in diiferent types as:

HDD Testing:
In this testing, we apply different operations like seek, pause-resume, applying different speeds from -128x to 128x, display of subtitle, closed caption and teletext on the test vectors which are stored in a hard disk.

HTTP & RTSP Testing:
In this testing also, we apply the different operations like seek, pause-resume, applying different speeds from -128x to 128x but test vectors are stored at a server.

7.2.4 DVR Testing

Personal video recorder software requires changes not only for adding new features but also for solving customer issues that they may face. So to check all the basic functionality of the software is working properly, a test suite has been developed. On running the test, it would check for the chosen functionality and will return the status; PASS or FAIL and then ask for user comments.

The test cas are in accordance to features of system. These may include:
Test for dual recording: Dual recording for same or different program.
Test for record and playback in timeshift mode.
Test for playback with trickmodes:
Playback the recorded file and we can apply positive speed form 0.1x to 128x and negative speeds like form -0.1x to -128x.
Test for variations in trickmodes:
Playback the recorded file with changing speed from positive to negative, negative to negative, negative to positive or positive to positive.

Test for checking display of subtitle & teletext:

7.2.5 IPSTB Testing

IPSTB testing is done using a server which broadcasts the data. And different types of streamers are used for this purpose. Different type of testcases are written for IPSTB testing: Test for dual recording: Dual recording for same or different program. Test for record and playback in timeshift mode. Test for streaming through different types of protocols. Test for applying trickmodes on recorded file.

7.2.6 System Testing

This testing is done when a patch is applied for adding functionality or some modifications are done in existing stack for any improvement. This is done to avoid lower level regressions. This testing is done for every new release.

7.2.7 Sanity Testing

Sanity testing determines whether it is reasonable to proceed with further testing. It is done for every new release or for generating patch. For all these types of testing above mentioned some test cases are designed and new test cases are added acc. to our requirement to check testing of different functionality.

7.3 Python

Python is a widely used general-purpose, high-level programming language. Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts

in fewer lines of code that would be possible in languages such as C.
Like other dynamic languages, Python is used as a scripting language, but is also used in a wide-range of non-scripting contexts. Python interpreters are available for many operating systems.
Python is a multi-paradigm programming language:
object-oriented programming and structured programming are fully supported, and there are a number of language features which support for functional programming and aspect-oriented programming.

Most uses of python:

• Python is strongly typed.
• Python is used as a scripting language for web applications.
• Python has also been used in artificial intelligence tasks.
** *Most user friendly method to:*
• Scan a file
• Search for a given pattern
• Extract the information
• Considerable speed
• Easy to learn.
• Fast development time

Typical Uses of Python:

• Web and Internet Development
• Database Access
• Desktop GUIs
• Scientific and Numeric
• Education
• Network Programming
• Software Development
• Game and 3D Graphics
• Python is a superb language for teaching programming, both at the introductory level and for more advanced courses.

Chapter 8

Linux and Hardware Environment Setup

8.1 Linux Environment Setup

The hardware and software environment in which tests will be run.

8.1.1 ssh key generation

Secure Shell (SSH) is a cryptographic network protocol for secure data communication, remote shell services or command execution and other secure network services between two networked computers that connects, via a secure channel over an insecure network, a server and a client.ssh-keygen creates the public and private keys.ssh-copy-id copies the local-host's public key to the remote-host's authorized keys files.ssh-copy-id also assigns proper permission to the remote-host home, /ssh; and /ssh/unauthorized keys: Skip the below steps if you already have an access to git repository. Under user use following command:
ssh-keygen -t rsa -C
emailaddress : f /.ssh/id rsa

8.1.2 Installation of required packages

Repo is a tool that Google built on top of Git to manage the many Git repositories, do the uploads to revision control system, and automate parts of the Android development workflow. Repo is not meant to replace Git, rather to make it easier to work with Git

A tool on top of Git

Git is an open source revision control system designed to handle projects that are distributed over multiple repositories. In the context of Android, Git is used for local operations such as local branching, commits, diffs, and edits. Repo is used for across-network operations. For example, with a single Repo command it is possible to download files from multiple repositories into your local working directory. This option provides the user with freedom to install the components at desired location. But with this option, you should create a local working directory. Packages required to be installed are:

- ARMV7
- KPI drivers
- Build folder

For complete installation follow the steps:

Download source code from git repository: Git is a free and open source, distributed version control system designed to handle everything from small to very large projects with speed and efficiency. Git was initially designed and developed by Linux To rvalds for Linux kernel development. Every Git working directory is a full- edged repository with complete history and full revision tracking capabilities, not dependent on network access or a central remote repository.

GIT Properties:

- Git is fully distributed
- Almost everything is local
- Everything is very fast
- Every clone of a repository is a backup
- The main branch is master
- You can work online
- Every Git repository is a client and a server

Create a new repository:

To Create your own directory that contain your project

$ mkdir training.git

$ cd training.git

– Then ,initialize your new repository

$ git init

– Now git repository is created your directory (simply a new hidden directory .git was created).

– Next step is to create your project files and directories and commit them to be integrated to your GIT repository.

Clone an existing repository:

– Clone does mean that you will get the project history and source code.

– Git clone support many network protocols such as:

http://; git://;ssh://

$ git clone ssh://root@10.157.7.171/root/training.git

– " Git clone " command support also the locals paths

$ git clone /root/training.git

– Git automatically compresses transferred and stored data in order to save disk space and network bandwidth.

Modify source code and commit:

Generally when you are working with GIT you will follow these steps:

(a) Modify source code

(b) Test if your program is working fine

(c) Do a commit to save your modifications and record them in git database.

(d) Restart with the first step for another modification

Git Branching:

By default git store your commit in the master branch

To create a new branch:

$ git branch testing

The question now is how Git know what branch you're currently on ? Git keeps a special pointer called HEAD to the branch that you are currently connected.

$ git checkout testing

Switched to branch "testing"

After merging your work in master branch you can delete testing branch by using Git check that your work in the branch "testing" was well merged in "master". Otherwise, it warns you of it and forbids you to delete eliminate the branch (you would otherwise risk to lose all your work in this branch!). If you want to delete a branch even if it contains changes which you did not merge, use the option-D. Suppose you have to run some tests ,you decided to

work in the testing branch. This situation can be also seen by using a graphic tool of git such as gitk, giggle etc.

Rebasing:

– In Git there are two main ways to integrate changes from branch to another; the merge and rebase.

– If you diverged from master branch.

• You want to integrate the branches.

• $ git checkout testing

• $ git rebase master

– By doing a rebase, git go to the common ancestor of the two branches (getting the diff introduced by each commit, resetting the current branch to the same commits as the branch you are rebasing into and finally applying each change in turn).

Tracking branches:

– Checking out a local branch from a remote branch automatically creates what is called a tracking branch.

– Tracking branches are local branches that have a direct relationship to a remote branch. If you're on a tracking branch and type git push, Git automatically knows which remote repository and branch to push to.

– Also, running git pull while on one of these branches fetches all the remote references and then automatically merges in the corresponding remote branch.

– When you clone a repository, it generally automatically creates a master branch that tracks origin/master.

– That's why git push and git pull work out of the box with no other arguments...

• The simple case is the example you just saw, running

• git checkout -b [branch] [remotename]/[branch].

• $ git checkout track [local branch] origin/[branch].

– If the remote repository contains another branch, for example "origin / master", and then you wish to work with it.

– It is necessary to create a copy of this branch on your computer which is going "to follow" the changes in the remote repository. $ git checkout -b master local origin/master

– When you will make a pull from the branch "testing", the changes will be fetched to origin/testing in your "testing" premises.(this is also available for master branch).

$ git pull
Unpacking objects: 100
remote: Counting objects: 7, done.
remote: Compressing objects: 100remote: Total 6 (delta 2), reused 0 (delta 0)
From /home/ipreeti/training.git/
1ab6129..cef4d87 master - origin/master
Updating1ab6129..cef 4d87
Fastforward
file5.txt—1 +
file6.txt—1 +
2fileschanged, 2insertions(+),
createmode100644f ile5.txt
createmode100644f ile6.txt

– git pull will automatically fetch from origin and merge origin master into your local testing branch, without having specified it.

Useful features in GIT:
Git stash:
– Before changing branch, you must have committed all your changes.

Plainly a git status should show no file under modification.

– If you have changes that are not " committed " and you change branch, the modified files will stay as they were in the new branch (and it is not generally what you want!).

– To avoid having to make a commit in the middle of a current work, type:

$ git stash
• Your modified files will be saved. Now, git status should show any more no file (we say that your working directory is clean).
• You can then change branch, make your modifications, " commit ", then return on the branch where you were.
• To get back the changes which you had put aside in your branch,
type:
$ git stash pop
• Git under Codex:
– SHARK Validation project can be accessed on this urn
https://codex.cro.st.com/projects/sharkvalidation/

– To contribute to the git project, you have to generate a public key and export it to the codex remote repository.
This is done by following these steps:
ssh-keygen
– Generating public/private rsa key pair.
– Enter file in which to save the key (/home/jaimin/.ssh/id rsa):
– /home/jaimin/.ssh/id rsa already exists.
– Overwrite (y/n)?
– $ cat /home/jaimin/.ssh/id rsa.pub
– Then, copy the content of " /home/jaimin/.ssh/id rsa.pub " to the codex web site https://codex.cro.st.com/account/editsshkeys.php

– If someone wants to access the git validation project from his unix account in ST,VMWare machine or a Windows machine, he will have to generate the public keys in all his accounts and export them to Codex web site. Now you can clone for example sbag repository.

```
$ git clone gitolite@codex.cro.st.com:sharkvalidation/validation-valid-sbag.git
Cloning into validation-valid-sbag...
The authenticity of host 'codex.cro.st.com (10.18.11.100)' can't be
established.
RSA key fingerprint is 4b:0f:d1:d2:92:4b:84:eb:f5:77:a5:61:3d:b2:cd:d9.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'codex.cro.st.com,10.18.11.100' (RSA) to the list of
known hosts.
remote: Counting objects: 466, done.
remote: Compressing objects: 100% (209/209), done.
remote: Total 466 (delta 318), reused 372 (delta 250)
Receiving objects: 100% (466/466), 181.51 KiB, done.
Resolving deltas: 100% (318/318), done.
$ ls validation-valid-sbag/
config.in  includes  Makefile  sbag_appli  sbag_module  shark_build  test
version.txt
```

FIGURE 8.1: GIT Clone

Set IP Address To download the object:
File of the code this can run on the target platform from the host, and to observe the logs of the executing test on the integrator, we need to connect target board to the host, as it cannot be directly connected, we require a Micro connect to provide the interfacing equipment between host and target board. For that we need to assign IP address to the target board and Micro connect Go to build path/build/config.in Set the JEI, TARGET IP and Gateway IP.
• JEI = To download and boot kernel using Micro connect, provide the IP address of Micro connect.
• TARGETIP = IP address to be assigned to the target board after kernel is up and running.
• GWIP = IP address of Gateway.

8.1.3 Patches

A patch is a piece of software designed to fix problems with, or it's supporting data. This includes fixing security vulnerabilities and other bugs, and improving the usability or performance. Though meant to fix problems, poorly designed patches can sometimes introduce new problems (see software regressions). Patch is a UNIX program that updates text files according to instructions contained in a separate file, called a patch file. The patch file (also called a patch for short) is a text file that consists of a list of differences and is produced by running the related diff program with the original and updated file as arguments. Updating files with patch is often referred to as applying the patch or simply patching the files.

```
diff --git a/include/PLTF_build.xml b/include/PLTF_build.xml
index e5594e7..46c12fc 100644
--- a/include/PLTF_build.xml
+++ b/include/PLTF_build.xml
@@ -1,6 +1,6 @@
 <?xml version='1.0' encoding='UTF-8'?>
 <manifest>
-<project name="pltf/PLTF_build_ext" path="pltf/build" remote="codex" revision="refs/tags/PLTF_build_ext_ML_1.272.0">
+<project name="pltf/PLTF_build_ext" path="pltf/build" remote="codex" revision="refs/tags/PLTF_build_ext_ML_1.273.0">
     <annotation name="STM_SDK2_BUILD_VAR" value="BUILD_FRM" />
     <annotation name="STM_SDK2_BUILD_VAR_EXTEND" value="build/sdk2-build" />
     <annotation name="STM_GIT_BRANCH" value="master" />
diff --git a/include/PLTF_sanity_test.xml b/include/PLTF_sanity_test.xml
index 72dad78..678df92 100644
--- a/include/PLTF_sanity_test.xml
+++ b/include/PLTF_sanity_test.xml
@@ -1,6 +1,6 @@
 <?xml version='1.0' encoding='UTF-8'?>
 <manifest>
-<project name="pltf/PLTF_sanity_test" path="tests/sanity_test" remote="codex" revision="refs/tags/Sanity_ML_1.10.0">
+<project name="pltf/PLTF_sanity_test" path="tests/sanity_test" remote="codex" revision="refs/tags/Sanity_ML_1.11.0">
     <annotation name="STM_SDK2_BUILD_VAR" value="SANITY_TESTS" />
     <annotation name="STM_GIT_BRANCH" value="master" />
 </project>
diff --git a/include/VIBE_VIBE.xml b/include/VIBE_VIBE.xml
index 6c310a9..efda0dd 100644
--- a/include/VIBE_VIBE.xml
+++ b/include/VIBE_VIBE.xml
@@ -1,6 +1,6 @@
 <?xml version='1.0' encoding='UTF-8'?>
 <manifest>
-<project name="vibe/VIBE" path="vibe/VIBE" remote="codex" revision="refs/tags/VIBE_ML_5.4.0">
+<project name="vibe/VIBE" path="vibe/VIBE" remote="codex" revision="refs/tags/VIBE_ML_5.7.0">
     <annotation name="STM_SDK2_BUILD_VAR" value="STMFB" />
     <annotation name="STM_GIT_BRANCH" value="master" />
 </project>
```

FIGURE 8.2: Patch Example - 1

To apply patch follow the below command.

A patch is a structured file that consists of a list of differences between one set of files and another.

All code changes, additions, or deletions to Drupal core and contributed modules/themes between developers are done through patches.

Patches make development easier, because instead of supplying a replacement file, possibly consisting of thousands of lines of code, the patch includes only the exact changes that were made.In effect, a patch is a list of all the changes made to a file, which can be used to re-create those changes on another copy of that file.

Patch -dry-run -p1 -I "patchname" : This is used to dry run the patch file i.e. which will give the error details if there are some error in applying patch file.
Patch -p1 -I "patchname" : This is to apply patch
git apply index path/file.patch: This is used when you are using git
The -p option tells patch how many leading prefixes to strip. For patches created using git, -p1 is normally the right option, and is the default for git apply.

```
diff --git a/include/GWAPPS_GwDevControl.xml b/include/GWAPPS_GwDevControl.xml
index e1f6a04..48bcea7 100644
--- a/include/GWAPPS_GwDevControl.xml
+++ b/include/GWAPPS_GwDevControl.xml
@@ -1,6 +1,6 @@
 <?xml version='1.0' encoding='UTF-8'?>
 <manifest>
-<project name="gwapps/GW_DevControl" path="apps/gwdevcontrol" remote="codex" revision="refs/tags/GwDevControl_ML_1.5.0">
+<project name="gwapps/GW_DevControl" path="apps/gwdevcontrol" remote="codex" revision="refs/tags/GwDevControl_ML_1.6.0">
     <annotation name="STM_SDK2_BUILD_VAR" value="SDK2_SOURCE_GWDEVCONTROL" />
 </project>
 </manifest>
\ No newline at end of file
diff --git a/include/INFRA_comms.xml b/include/INFRA_comms.xml
index ce34ec1..62d2d5d 100644
--- a/include/INFRA_comms.xml
+++ b/include/INFRA_comms.xml
@@ -1,7 +1,7 @@
 <?xml version='1.0' encoding='UTF-8'?>
 <manifest>
-<project name="gwapps/INFRA_comms" path="infra/INFRA_comms" remote="codex" revision="refs/tags/INFRACOMMS_ML_1.16.0">
+<project name="gwapps/INFRA_comms" path="infra/INFRA_comms" remote="codex" revision="refs/tags/INFRACOMMS_ML_1.17.0">
     <annotation name="STM_SDK2_BUILD_VAR" value="SDK2_SOURCE_INFRACOMMS" />
     <annotation name="STM_GIT_BRANCH" value="master" />
 </project>
-</manifest>
+</manifest>
\ No newline at end of file
diff --git a/include/INFRA_kernel3.xml b/include/INFRA_kernel3.xml
index 5640734..152db83 100644
--- a/include/INFRA_kernel3.xml
+++ b/include/INFRA_kernel3.xml
@@ -1,6 +1,6 @@
 <?xml version='1.0' encoding='UTF-8'?>
 <manifest>
-<project name="infra/INFRA_kernel3" path="infra/kernel3" remote="codex" revision="refs/tags/KERNEL_12.0_3.4.37_stm24_0305.36.1">
+<project name="infra/INFRA_kernel3" path="infra/kernel3" remote="codex" revision="refs/tags/KERNEL_12.0_3.4.37_stm24_0305.36.2">
     <annotation name="STM_SDK2_BUILD_VAR" value="KERNEL" />
     <annotation name="STM_GIT_BRANCH" value="master" />
 </project>
```

FIGURE 8.3: Patch Example - 2

8.1.4 Build Process

The term build refers either to the process of converting source code files into standalone software artifact(s) that can be run on a computer, or the result of doing so. One of the most important steps of a software build is the compilation process where source code files are converted into executable code. In software version, the build number is often used as a versioning identifier. To Build all the modules, kernel, It is necessary to have Makefile.
(a) Compiler takes the source files and outputs object files.
(b) Linker takes the object files and creates an executable.
• Go to /build path/build/sdk2-build.b2112-d127_a9/ as a normal user and follow the below commands.
• make clean: To clean everything

• make modules: To build all sdk2 modules
• make module name: To build specific module
• make .clean module name: To clean specific module
• make .modules install module name: To install specific component
• make all: To build and install kernel, all sdk modules

8.1.5 Test execution

After build process, the executable files of the source code is generated, to test the STB drivers of the STB devices, we need to install this on to target and need to run the different test cases. The test cases are provided by the customer. For each devices and its function test cases are provided in the tool. To execute this test cycle we need to boot the target board. For this, The file config.in in build directory must be updated for JEI, Target IP and Gateway IP.

• make run

This will boot the board, i.e. VMLinux image will be loaded to the board memory. VM Linux is a executable file that contains Linux kernel in one of the object file formats supported by Linux.

Booting is the process in which your computer gets initialized. This process includes initializing all your hardware components in your computer and get them to work together and to load default operating system which will make your system operational.

When computer's power is switched on, control is transferred by hardware to the bootstrap procedure of BIOS in Rom. The bootstrap procedure carries out some hardware test to check whether the memory and other devices are functioning properly.

• telnet IP address of target -l root

Log in as a root user on the target platform. This should be done on different terminal as done to boot kernel

• ./framework_go.sh

Load Modules: To make the kernel aware to use the module. Mode probe insert the module and also the dependency of that mode. Mode probe loads the modules in to the kernel When we insert the device driver, To make it visible, we make a device nodes.

• Run /Application

This will do some initialization, load its database, load test suit and finally display test menu. Now the software set up is ready for the driver testing.

8.2 Hardware Set up

To set up the hardware to carry out the driver testing following are the required component details. Hardware Requirements

- STMC
- Target board
- Stream server
- JTAG
- UART
- HDMI cable for connection with TV
- Four Power Adapter (Each for Micro Connect and Board)

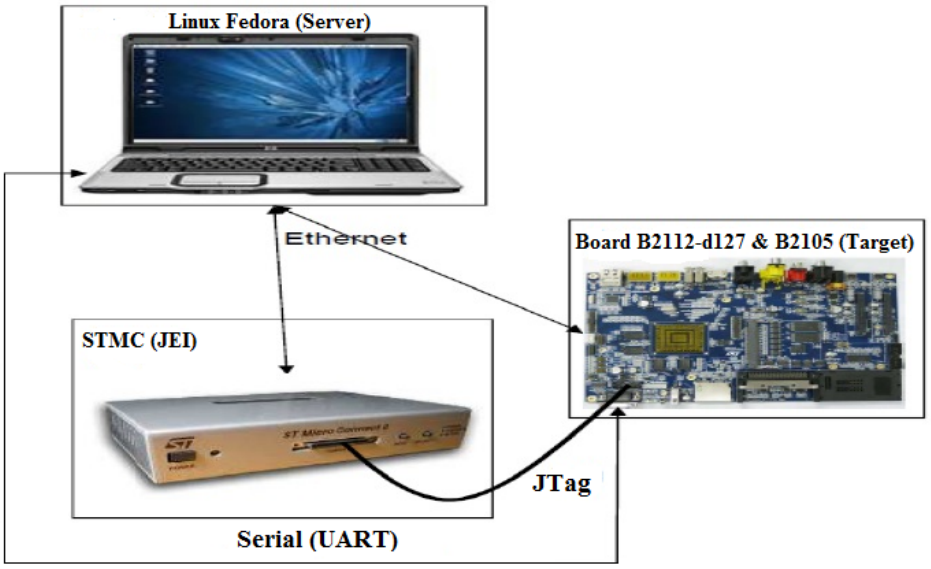

FIGURE 8.4: Hardware Arrangement

The different hardware components used are:

SoC:

The System on chip contain two ARM processors, one SH-4 (used for floating point calculations), one ARMv7

Micro Connect:

The kernel image compiled on the PC is loaded in to the SoC memory using the Micro connect. ST Micro connect is a host-target interface from STMicroelectronics. It connects to a target development board's JTAG connector and provides host software with the ability to start up the target board, download programs and debug them in the target. It is easy to

install and use the ST Micro Connection Package provides software utilities and firmware, including Target Packs for certain ST evaluation boards.

Stream Server:
The stream server contains all the required streams provided by the customer to be viewed on the receiver in digital format, during testing of audio video playback. Normally Tuner receives the stream via dish antenna. Here, to give streams to the board we connect stream modulator. Particular stream has its frequency, symbol rate, data rate, FEC. But we have to perform so many tests on different channel and symbol rate, so, we use the stream server. We have modulator card which is connected to the stream server. Modulator card is connected to the RF Tuner on the board via RF cable. We need the modulator card to convert the signal to the RF frequency to send it to the RF tuner card on the board via RF cable.

JTAG:
The JTAG cable facilitates the transfer of the kernel image from the PC to the SoC.

UART:
The UART transmits the serial output from the board to the PC; it is useful in getting the debug information.

TV:
Serves as the receiver for digital output in various formats like HDMI, AV, SPDIF

Chapter 9

Automation Through Testframework

9.1 Flow To Set An Environment of SDK2:

9.1.1 Starting the Test Framework:

9.1.1.1 Prerequisite:

Once SDK2 is installed on our system and the application compiled, we can launch the Test Framework. At this point, we should have installed the SDK2 and the STLinux files, launched the sdk2.sh script and generated the SDK2 executable.

We have two ways to start the Test Framework: 1. We can run a make file command from the same directory where we launched our SDK2 compilation 2. We can execute the main.py Python file located in the tests/Test Framework directory for a more flexible usage

9.1.1.2 Make file Command

After compiling the SDK2 stack, stay in the directory where we launched our make command. For example:

• sdk2/Wavefront/pltf/build/build/sdk2-build.b2120-h410_a9/

Running:

• $ make

Without any argument will display the list of available targets.

9.1.1.3 Framework Execution from the Command Line:

We can launch the Test Framework before or after the target has booted.
COMMAND LINE OPTIONS
• General syntax:
main.py [session options] root_file.xml[root_name]* [additional_path] * [targetIP]

The [session options] can be:

1. -initialboot
Reboots the board once before launching the tests

2. -initialbootonly
Reboots the board from the framework and exits

3. -listfiles
Provides the list of parsed files for this test

4. -skipnn
nn is an integer value, skips the first nn files

5. -startboard
When running on board framework execution, apply (Remote_Shell_Command) tags from (board_configuration) file

6. -stop-on-first-error For helping some debug cases, the framework exits on the first identified error, stops the streaming and provides the error code to the shell.

• [root_name] is the optional name we can encounter within a launch description of the root file. It helps launching only specific tests of a large program.
• [additional_path] is here for extending the PATH search of the python and XML files within the Framework.
• [targetIP] must be an ipv4 address.

9.1.1.4 Description:

The framework is made of data stored in XML format, and python code. General mindset is to maximize declarations and minimize code both on application side and in python interpreter side.

Declarations are made of:

1. Stream descriptions:

• This structure provides a unique file name, container, duration, codex for all tracks, resolutions, rates, stream ids.

• First parses directory tree and calls mediainfo for deep description of multimedia files.

• A second step is made of an xsl sheet transform that provides the compatible stream description XML required. We need a Mediainfo version installed in your Linux system for performing that import.

2. Stream locations:

• Aim is to make independent the way the file is accessed from the board and its properties. Role of Stream_locations.xml is to tell depending on the modes (local, http . . .), the Path in our system. Predefined given locations generally start with "mounted". To make this match on our board please ensure a "mounted" directory exists under /root of the board.

3. Application abstraction:

• Set of applications is providing an abstraction layer, and information for the framework such as startup options for the launching command and interactive commands. Remark: The XML description of interactive commands gives the names of the functions to be called during tests.

• The command line options are bringing a number of options but need some python specific code for making proper use of them. This work is done today within module_dvbtest.py and module_gstapps.py.

• The interactive commands are simply storing node names that user script shall use for accessing abstracted names from application. Node content contains the application command.

4. Test cases, or Calls:

• This level is application independent. We should not take care of which application is running the test case (except testing advanced features only available on one application).

Each test case defines:

• A file selection. This let choose between modes of accessing data from board, either locally (local keyword used) from any mounted device, or remotely (http keyword used). As usual with XML we can define our own definition, except those starting with "Streamer": Exclusively reserved for driving the packet injector.

• A file selector. This is an XPath expression based on the contents of the Stream descriptions. It provides a list of files and applies the test case on them.

• The name of the python module containing user script (¡use_case_script¿ node) and in the following launching order:

1. (¡pre_command¿) System call such as an fbset
2. ¡use_case_pre_launch¿ Python function accessed before program is launched [let prepare the context]
3. ¡use_case_execution¿ Python function performing the test itself
4. ¡use_case_post_launch¿ Python function after application exits, let perform real time verdicts with full statistics
5. ¡post_command¿ System call

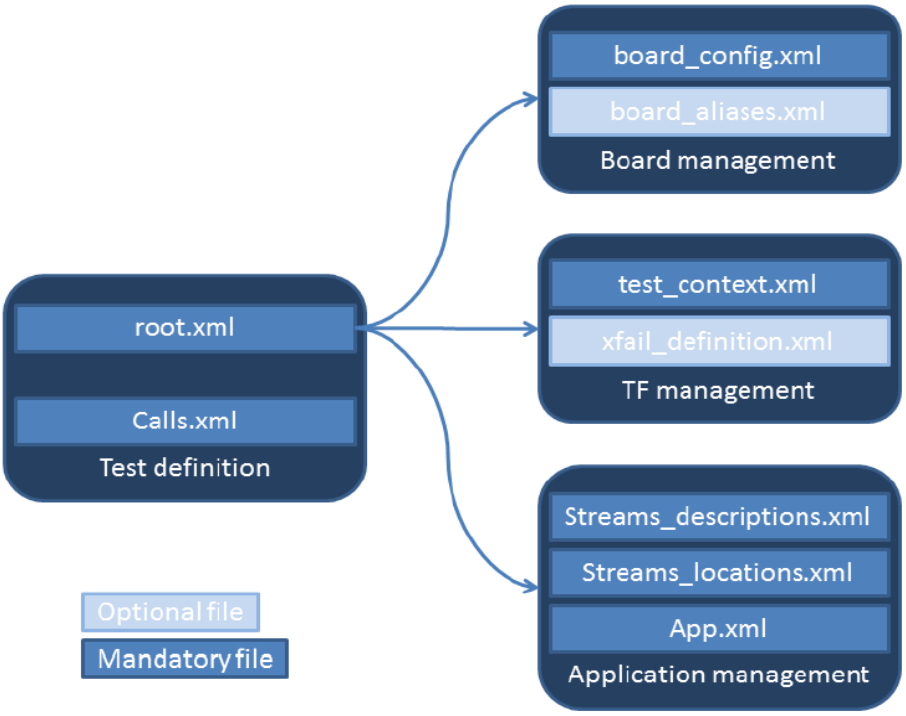

FIGURE 9.1: Arrangement of files in Test Framework

All python function receive as parameter (optionally) the Python dictionary of the stream properties to know duration, resolution, codex, and so on ... They return a value that is passed of failed, ruled by the framework to the reporting.

9.1.1.5 How to Execute Test Cases Uses Test Framework:

Step 1: Boot the Board
Step 2: View the UART logs via serial-relay
• Wait till the log shows login message by username
Step 3: Telnet the TARGETIP (IP of Board) as root
Step 4: Load the modules like media player, HDMI Manager etc.
Step 5: Mount the SCSI Disk in root folder inside opt directory.
Step 6: Change the directory to mainline manifest's test framework directory.
Step 7: Export some of the environmental variables in initial.sh file.
These variables are:
• SERVERIP : IP of host machine
• TARGETIP : IP of the board
• JEI : STMC (ST Micro Connect) IP
• GWIP : Gateway IP
• NFS-SERVER : Network File System IP
• STREAM_SERVER : Used for special cases of HTTP
Step 8: Source the above mentioned file
Step 9: Execute the script "main.py" along with required arguments to start the tests.
• Even we can give some options along with above command. If we give option "initialboot" then we have to follow from step 6.
Step 10: See the Reports in RESULT folder of that particular test framework.

9.1.1.6 Different Types Of Test Cases:

1. LIVE
2. DISPLAY_PIP_LIVE
3. DISPLAY_PIP_MIXED
4. DISPLAY_ASPECT_RATIO
5. DISPLAY_SUBTITLE
6. DVR_TIMESHIFT
7. DVR_TRICKMODE
8. PLAYBACK_CONTAINERS

9. PICTURES
10. IP_VOD_HTTP
11. IP_VOD_RTSP
12. IP_LIVE
13. IP_PIP

9.1.1.7 Flow to execute a particular test:

It can be done in two ways:
1. Manually by application.
2. By Test Framework

1. Manually By Application:
The Applications used to launch any of the test cases are:
• gst-apps
• dvbtest
Example: Running LIVE Test Case by gst-apps:
o All these are started after telnet and loading modules.
o Configure the dvb-channel-config.conf file according to the availability of channels.
o Executing the command:
• gst-apps - N0 dvb://[channel-name]
• This will lock the tuner according to the frequency of that particular channel and then start playing the channel on display via using all kernel modules.

```
root@b2105-h416_a9:~#
root@b2105-h416_a9:~# ./framework_go.sh
Loading module player2...
Module player2 successfully loaded
Loading module stlinuxtv...
Module stlinuxtv successfully loaded
root@b2105-h416_a9:~#
root@b2105-h416_a9:~#
root@b2105-h416_a9:~#
root@b2105-h416_a9:~# gst-apps dvb://robots
****************
GST-APPS v1.0.75
****************
Playing file dvb://robots
Trying to lock tuner...
Tuner locked
Received ASYNC DONE on bus - 1 video, 1 audio, 1 subtitle
Audio stream=0 codec=(null) language=(null)
Subtitle stream=0 codec=(null) language=und
```

FIGURE 9.2: Starting Live Use-case Manually

o Introducing ZAPPING in live manually:

- Zapping implies changing channel while one is already running.

- Executing the command:

gst-apps –N0 dvb://[channel-name], then give run time: commands:

• d: channel-down

• u: channel-up

```
root@b2105-h416_a9:~# gst-apps dvb://robots
*****************
GST-APPS v1.0.75
*****************
Playing file dvb://robots
Trying to lock tuner...
Tuner locked
Received ASYNC DONE on bus - 1 video, 1 audio, 1 subtitle
Audio stream=0 codec=(null) language=(null)
Subtitle stream=0 codec=(null) language=und
d
Trying to lock tuner...
Tuner locked
Playing CAB_MUX channel
Received ASYNC DONE on bus - 1 video, 1 audio, 1 subtitle
Audio stream=0 codec=(null) language=(null)
Subtitle stream=0 codec=(null) language=und
u
Trying to lock tuner...
Tuner locked
Playing robots channel
Received ASYNC DONE on bus - 1 video, 1 audio, 1 subtitle
Audio stream=0 codec=(null) language=(null)
Subtitle stream=0 codec=(null) language=und
u
Trying to lock tuner...
Tuner locked
Playing Cornell_MPEG2_HD-SD_H264_HD-SD_PAL_MPTS.trp channel
Received ASYNC DONE on bus - 1 video, 1 audio, 1 subtitle
Audio stream=0 codec=(null) language=(null)
Subtitle stream=0 codec=(null) language=und
d
Trying to lock tuner...
Tuner locked
Playing robots channel
Received ASYNC DONE on bus - 1 video, 1 audio, 1 subtitle
Audio stream=0 codec=(null) language=(null)
Subtitle stream=0 codec=(null) language=und
```

FIGURE 9.3: Zapping Live Use-case Manually

```
root@b2105-h416_a9:~# for ((i=0;i<10;i++)); do (sleep 5; echo 'd'; sleep 5;
echo 'u';sleep 5; echo 'd'; sleep 5; echo 'u'); done | gst-apps "dvb://robots1"
****************
GST-APPS v1.0.75
****************
Playing file dvb://robots1
Trying to lock tuner...
Tuner locked
Received ASYNC DONE on bus - 1 video, 1 audio, 1 subtitle
Audio stream=0 codec=(null) language=(null)
Subtitle stream=0 codec=(null) language=und
Trying to lock tuner...
Tuner locked
Playing CAB_MUX channel
Received ASYNC DONE on bus - 1 video, 1 audio, 1 subtitle
Audio stream=0 codec=(null) language=(null)
Subtitle stream=0 codec=(null) language=und
Trying to lock tuner...
Tuner locked
Playing robots1 channel
Received ASYNC DONE on bus - 1 video, 1 audio, 1 subtitle
Audio stream=0 codec=(null) language=(null)
Subtitle stream=0 codec=(null) language=und
Trying to lock tuner...
Tuner locked
Playing CAB_MUX channel
Received ASYNC DONE on bus - 1 video, 1 audio, 1 subtitle
Audio stream=0 codec=(null) language=(null)
Subtitle stream=0 codec=(null) language=und
```

FIGURE 9.4: Zapping Live Use-case Automatic

2. By Test Framework:

• There are some xml files used for executing each of the test cases. Each of the have different purpose.

• Few of them are:

Declarations are made of several xml files listed in the following paragraphs:

- Root
- Stream descriptions
- Stream locations
- Application abstraction
- Test calls
- Board configuration

• Test Result:

The Test Framework provides a test status based on the launched application execution.

The table below classes the priority from highest to lowest:

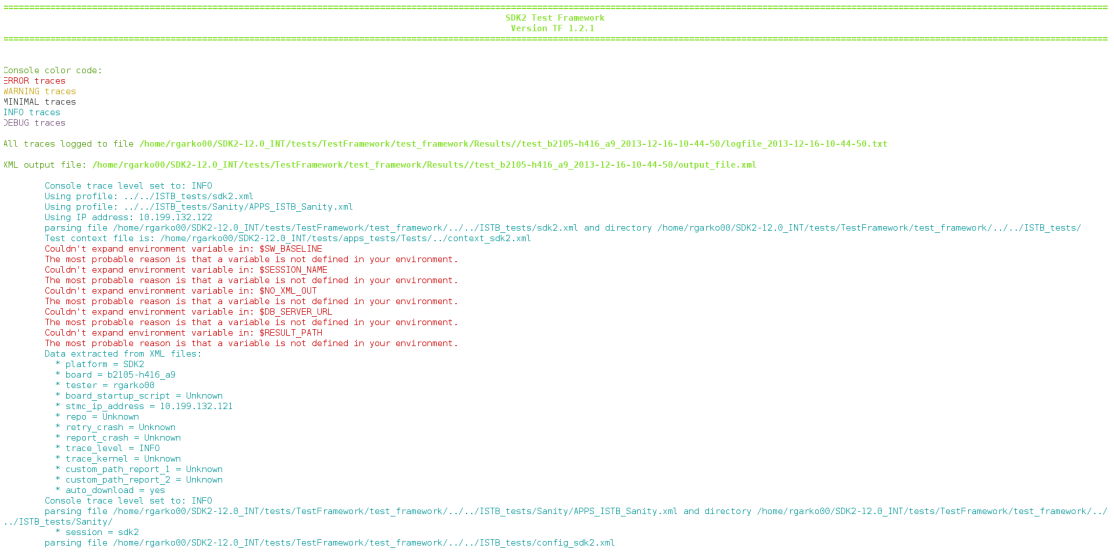

FIGURE 9.5: Verdict window

TEST REPORT

Session summary:

PASSED	0
XPASS	0
FAILED	1
XFAIL	0
TIMEOUT	0
CRASHED	1
MISSING	0
ABORTED	0
SKIPPED	0
Total launched	2
PASS RATE	0%

Session info:

Board:	b2105-h416_a9
Baseline:	Unknown
Tester:	rgarko00
Kernel:	3.4.37_stm24_0305-b2105-h416_a9
Audio firmware:	ACF_FWK v1.0 : audio_firmware-bd-internal-35.5.0-0
Video firmware:	Unknown
Linux version:	Linux version 3.4.37_stm24_0305-b2105-h416_a9 (rgarko00@localhost.localdomain) (gcc version 4.7.3 20130603 (STMicroelectronics/Linux Base 4.7.3-125) (GCC)) #1 SMP PREEMPT Tue Jan 14 12:15:31 IST 2014
Test started at:	2014-01-14 14:07:05
Test ended at:	2014-01-14 14:10:41
Root file:	Using profile: ../../ISTB_tests/sdk2.xml Using profile: ../../ISTB_tests/Sanity/APPS_ISTB_Sanity.xml

Detailed report:

Open all test cases
Close all items

Test	Result
TC_APPS_FUNC_ISTB_Tuner	CRASHED
TC_APPS_FUNC_STMF_PIP	FAILED

FIGURE 9.6: Failed Test Report

TABLE 9.1: Verdict Description

ABORTED	User termination of the test Execution
CRASHED	Execution leading to a KERNEL crash, system need to reboot
TIMEOUT	The application didn't succeed to exit, need to reboot
XFAIL	The test result is fail but it is a known issue and specified in a file
FAILED	Execution or verdict has failed but system is still considered as safe.
SKIPPED	Test listed in the test plan but not executed
PASSED	Execution and verdict (if exist) OK
MISSING	The file is missing
XPASS	The result was expected as False but it is passed

TEST REPORT

Session summary:

PASSED	7
XPASS	0
FAILED	0
XFAIL	0
TIMEOUT	0
CRASHED	0
MISSING	0
ABORTED	0
SKIPPED	0
Total launched	7
PASS RATE	100%

Session info:

Board:	b2120-h410_a9
Baseline:	Unknown
Tester:	rgarko00
Kernel:	3.4.58_stm24_0307-b2120-h410_a9+
Audio firmware:	Unknown
Video firmware:	Unknown
Linux version:	Linux version 3.4.58_stm24_0307-b2120-h410_a9+ (rgarko00@localhost.localdomain) (gcc version 4.7.3 20130603 (STMicroelectronics/Linux Base 4.7.3-129) (GCC)) #2 SMP PREEMPT Wed Feb 19 11:55:12 IST 2014
Test started at:	2014-02-19 14:00:31
Test ended at:	2014-02-19 14:12:53
Root file:	Using profile: ../../ISTB_tests/sdk2.xml Using profile: ../../ISTB_tests/Sanity/APPS_ISTB_Sanity.xml

Detailed report:

Open all test cases

Close all items

Test	Result
TC_APPS_FUNC_ISTB_Tuner0	PASSED
TC_APPS_FUNC_ISTB_Tuner1	PASSED
TC_APPS_FUNC_ISTB_Tuner2	PASSED
TC_APPS_FUNC_ISTB_Tuner3	PASSED
TC_APPS_FUNC_ISTB_Tuner4	PASSED
TC_APPS_FUNC_ISTB_Tuner5	PASSED
TC_APPS_FUNC_STMF_PIP	PASSED

FIGURE 9.7: Passed Test Report

EXECUTION FLOW OF TEST FRAMEWORK::

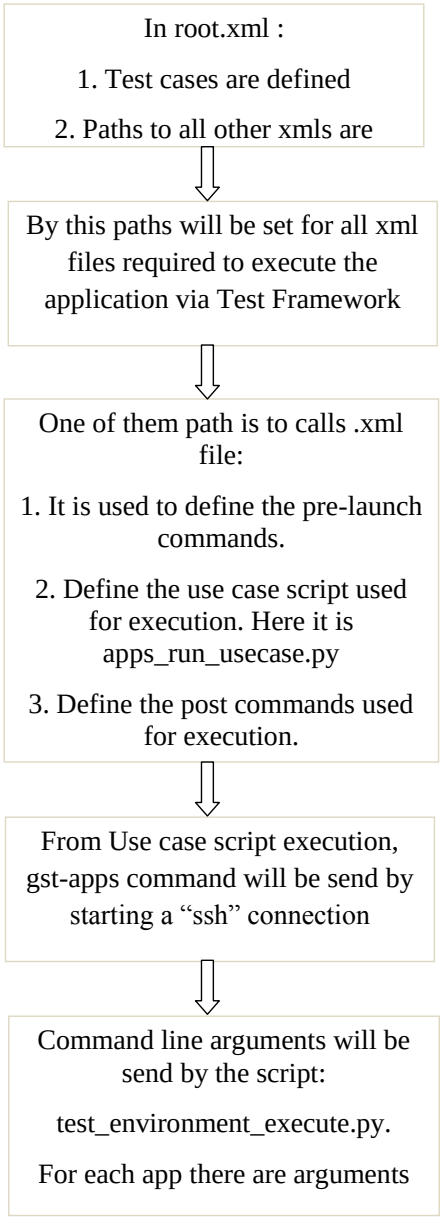

FIGURE 9.8: Execution Flow of Test Frame work

Chapter 10

Technology Used

10.1 Technology Used

10.1.1 Source Insight

It is a source code editor by Source Dynamics. Source Insight provides syntax highlighting, code navigation and customizable keyboard shortcuts. It bills itself not just as an editor but a tool to understand a large source code base, and for this reason is called "program editor and analyzer." It is agile and lightweight providing useful features such as relation, context, and symbol windows. It also can display reference trees, class inheritance diagrams, and call trees, as it builds an internal database of symbolic information as it self-parses the source. Its greatest benefit is to speedup code comprehension on an unfamiliar project.

10.1.2 Novel features

Source Insight provides all the features of the venerable scope in a GUI environment along with a program editor. These C features are extended to object oriented domain and made more robust by being tolerant of the 'typedefs' or 'pragmas' of embedded processor C extensions. Further innovative features such as 'ifdef support' and conditional parsing allows view of the code with inactive code visually and thus aiding speedy comprehension. As more code is added, Source Insight automatically keeps its database updated, displaying variables in different colors depending on if they are local, global, static, function arguments, or yet undefined.

10.1.3 Supported languages

Source Insight supports a wide variety of programming languages but primarily geared towards C/C++, C# and Java.

10.1.3.1 Features

Source Insight™ is a project-oriented program editor and code browser, with built-in analysis for C/C++, C#, and Java programs. Source Insight parses your source code and maintains its own database of symbolic information dynamically while you work, and presents useful contextual information to you automatically. Not only is Source Insight a great program editor, but it also can display reference trees, class inheritance diagrams, and call trees. Source Insight features the quickest navigation of source code and source information of any programming editor. Source Insight features quick and innovative access to source code and source information. Unlike many other editor products, Source Insight parses your source code and gives you useful information and analysis right away, while you edit.

- Always Up-To-Date Information
- Call Graphs and Class Tree Diagrams
- Context Sensitive Dynamic Type Resolution
- Symbol Windows For Each File
- Automatic Display of Declarations in the Context Window
- Syntax Formatting
- Context-Sensitive Smart Rename
- Mixed Language Editing
- Keyword Searches Like an Internet Search on Your Code Base
- Symbolic Auto-Completion
- Quick Access to All Symbols and Files
- Project Level Orientation
- Team Programming Support
- Finds References Quickly
- Hyper Source Links to Link Compiler Errors and Search Results
- Fast Project-Wide Search and Replace
- Project Window With Multiple Views
- Integrates with External Compilers and Tools
- Clip Window for Storing Multiple Clipboards and Boiler Plate Code
- Two-Stage Line Revision Marks and Selective Line Restoration
- Extensible Document Types and Languages

- Crash Recovery Offers Full-Time Protection
- Persistent Workspaces
- Customizable Menus and Keyboard
- Special Support for Remote Terminal Server Sessions
- Windows 7/Vista/2000/XP Support
- Outstanding Windows User Interface
- Full Featured Editor
- Drag and Drop Editing
- Real World Tested
- Speed and Convenience

10.1.4 VMWare

10.1.4.1 Core product design

VMware developed a range of products, most notable of which are their hypervisors. VMware became well known for their first type 2 hypervisor known as GSX. This product has since evolved into two hypervisor products lines, VMware's type 1 hypervisors running directly on hardware, along with their hosted type 2 hypervisors.

VMware software provides a completely virtualized set of hardware to the guest operating system. VMware software virtualizes the hardware for a video adapter, a network adapter, and hard disk adapters. The host provides pass-through drivers for guest USB, serial, and parallel devices. In this way, VMware virtual machines become highly portable between computers, because every host looks nearly identical to the guest. In practice, a system administrator can pause operations on a virtual machine guest, move or copy that guest to another physical computer, and there resume execution exactly at the point of suspension. Alternatively, for enterprise servers, a feature called vMotion allows the migration of operational guest virtual machines between similar but separate hardware hosts sharing the same storage (or, with vMotion Storage, separate storage can be used, too). Each of these transitions is completely transparent to any users on the virtual machine at the time it is being migrated.

VMware Workstation, Server, and ESX take a more optimized path to running target operating systems on the host than emulators (such as Bochs) which simulate the function of each CPU instruction on the target machine one-by-one, or dynamic recompilation which compiles blocks of machine-instructions the first time they execute, and then uses the translated

code directly when the code runs subsequently (Microsoft Virtual PC for Mac OS X takes this approach.) VMware software does not emulate an instruction set for different hardware not physically present. This significantly boosts performance, but can cause problems when moving virtual machine guests between hardware hosts using different instruction-sets (such as found in 64-bit Intel and AMD CPUs), or between hardware hosts with a differing number of CPUs. Software that is CPU agnostic can usually survive such a transition, unless it is agnostic by forking at startup, in which case, the software or the guest OS must be stopped before moving it, then restarted after the move.

VMware's products predate the virtualization extensions to the x86 instruction set, and do not require virtualization-enabled processors. On newer processors, the hypervisor is now designed to take advantage of the extensions. However, unlike many other hypervisors, VMware still supports older processors. In such cases, it uses the CPU to run code directly whenever possible (as, for example, when running user-mode and virtual 8086 mode code on x86). When direct execution cannot operate, such as with kernel-level and real-mode code, VMware products use Binary translation (BT) to re-write the code dynamically. The translated code gets stored in spare memory, typically at the end of the address space, which segmentation mechanisms can protect and make invisible. For these reasons, VMware operates dramatically faster than emulators, running at more than 80% of the speed that the virtual guest operating-system would run directly on the same hardware. In one study VMware claims a slowdown over native ranging from 0–6 percent for the VMware ESX Server.

VMware's approach avoids some of the difficulties of virtualization on x86-based platforms. Virtual machines may deal with offending instructions by replacing them, or by simply running kernel-code in user-mode. Replacing instructions runs the risk that the code may fail to find the expected content if it reads itself; one cannot protect code against reading while allowing normal execution, and replacing in-place becomes complicated. Running the code unmodified in user-mode will also fail, as most instructions which just read the machine-state do not cause an exception and will betray the real state of the program, and certain instructions silently change behavior in user-mode. One must always rewrite; performing a simulation of the current program counter in the original location when necessary and (notably) remapping hardware code breakpoints.

Although VMware virtual machines run in user-mode, VMware Workstation itself requires the installation of various drivers in the host operating-system, notably to dynamically switch the Global Descriptor Table (GDT) and the Interrupt Descriptor Table (IDT).

The VMware product line can also run different operating systems on a dual-boot system simultaneously by booting one partition natively while using the other as a guest within VMware Workstation.

10.1.4.2 Desktop software

• VMware Workstation (first product launched by VMware in 1999). This software suite allows users to run multiple instances of x86 or x86-64 -compatible operating systems on a single physical PC.

• VMware Fusion provides similar functionality for users of the Intel Mac platform, along with full compatibility with virtual machines created by other VMware products.

• VMware Player is for users without a license to use VMware Workstation or VMware Fusion. VMware offers this software as a freeware product for noncommercial personal use. While initially not able to create virtual machines, this limitation was removed in version 3.0.1.

10.1.5 JTAG

Joint Test Action Group (JTAG) is the common name for the IEEE 1149.1 **Standard Test Access Port and Boundary-Scan Architecture**. It was initially devised by electronic engineers for testing printed circuit boards using boundary scan and is still widely used for this application.

JTAG allows device programmer hardware to transfer data into internal non-volatile device memory (e.g. CPLDs). Some device programmers serve a double purpose for programming as well as debugging the device. In the case of FPGAs, volatile memory devices can also be programmed via the JTAG port, normally during development work. In addition, internal monitoring capabilities (temperature, voltage and current) may be accessible via the JTAG port.

JTAG programmers are also used to write software and data into flash memory. This is usually done using data bus access like the CPU would use, and is sometimes actually handled by a CPU, but in other cases memory chips have JTAG interfaces themselves. Some modern debug architectures, like ARM CoreSight and Nexus, provide internal and external

bus master access without needing to halt and take over a CPU. In the worst case, it is usually possible to drive external bus signals using the boundary scan facility.

As a practical matter, when developing an embedded system, emulating the instruction store is the fastest way to implement the "debug cycle" (edit, compile, download, test, and debug). This is because the in-circuit emulator simulating an instruction store can be updated very quickly from the development host via, say, USB. Using a serial UART port and bootloader to upload firmware to Flash makes this debug cycle quite slow and possibly expensive in terms of tools; installing firmware into Flash (or SRAM instead of Flash) via JTAG is an intermediate solution between these extremes.

Bibliography

[1] ST Internal Document.

[2] Set-top box software architectures for digital video broadcast and interactive services, Jaeger, R. ; BetaResearch, Germany.

[3] Inside the set-top box, Ciciora, W.S.

[4] Gateway, multi-service, home, network, IEEE1394, web, MPEG.

[5] Software Engineering P practitioner's approach, 4th edition, Roger S. Pressman.

[6] Software Engineering K.K. Aggarwal, Yogesh Singh.