MPEG2 -TS Multi-Program Channel Selector

Mohit Aggarwal
Dept.VLSI & Embedded System
M.tech in Ganpat University

Nayan Bhavsar
Dept.VLSI & Embedded System
M.tech in Ganpat University

Hardik Vala
Dept.VLSI & Embedded System
M.tech in Ganpat University

Punit Purohit
Dept.VLSI & Embedded System
M.tech in Ganpat University

***Abstract*-MPEG-2 is a video coding standard created by the Moving Picture Experts Group (MPEG) and finalized in 1994.Since 2005, it is the standard format used for satellite TV, digital cable TV, DVD movies, and HDTV. In this paper we have introduced "MPEG2-TS Based Multi-Program Channel Selector" which is designed to accept MPEG2TS based Multi- Program stream as in input and stream output a Single program MPEG2-TS stream over UDP/RTP protocol .The output program shall be selectable by user configuration through web interface provided by the processing module.**

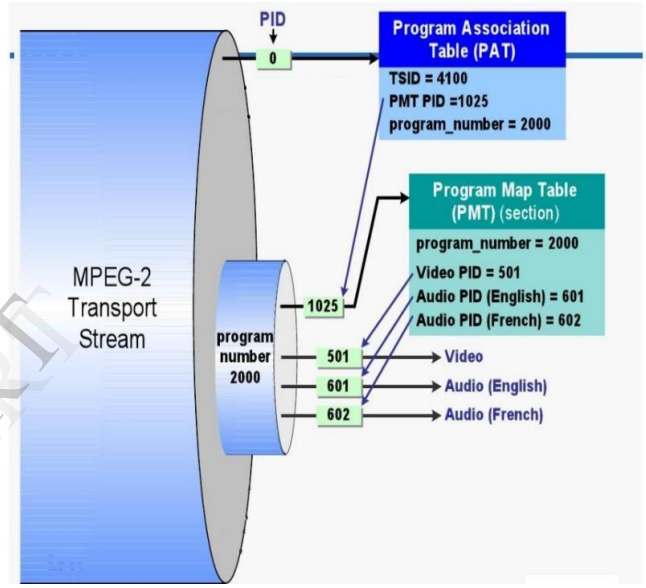
INTRODUCTION

In general, there are two types of video delivery: Video streaming and Video delivery by download. Video streaming is the real-time transmission of video to a client device that enables simultaneous delivery of video packets and playback of the video. In effect, video streaming splits the video into frames, transmits these frames in succession and enables the receiver to decode and playback the video to be delivered .Video streaming has a small buffer requirements since only small portion of the video is stored at the client at any point in time rather than as a file downloading where the entire video has to be send and stored at the client side before playback at the client. For these, in the system design we have introduced a buffer manager module which will manage the limited storage of frames and playbacks the frames without entire video to be downloaded. This paper is MPEG-2 TS streaming oriented focusing on Video on Demand.

MPEG2-TS TECHNOLOGY

MPEG transport stream (MPEG-TS, MTS or TS) is standard format for transmission and storage of audio,

Video, Program and System Information Protocol (PSIP) data. It is used in broadcast systems such as DVB, ATSC and IPTV. MPEG-2 Transport Streams are composed of 188 bytes TS Packets, each with a 4 byte header. Some TS packets contain an optional Adaptation Field whose size depends on flags set in the packet header and which may contain timing information, pad bytes, and other data, TS packet payloads may contain program information as well as Packetized Elementary Streams (PES), typically video and audio streams.



Drawing 1: MPEG2-TS Structure [1]

Transport packets have multiple interleaved elementary streams audio, video, data, PSI identified by packet id (PID) in packet header. Transport stream has a concept of programs. Each single program is described by a Program Map Table (PMT) which has a unique PID, and the elementary streams associated with that program have PIDs listed in the PMT. Each program consist of one or more elementary streams ,the receiver can decode the particular program channel by decoding the payload of selected elementary stream PID associated with program. A transport stream with more than one program is referred to as MPTS – Multi Program Transport Stream. A single program transport stream is referred to as SPTS - Single Program Transport Stream.

There are 4 PSI (Program Specific Information) tables: Program Association Table (PAT), Program Map Table (PMT), Conditional Access (CAT), and Network Information (NIT). PAT lists all PMT (programs) available in the transport stream. Each program has PID, TS packet containing PAT information always have PID 0x0000. PMT contain information about programs. The PMTs

provide information about the elementary stream which comprises the MPEG-2 program. Each elementary stream is labelled with a stream type value.

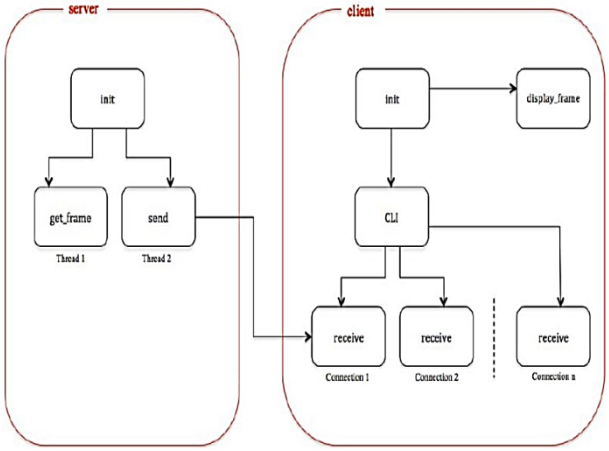
VIDEO STREAMING FRAME WORK

The basic framework consist of essentially three elements, namely Sender(Server),Receiver(Client) and Network (TCP / any other connection).Since communication between two ends is involved over a network , a concurrency model is to be applied. Multi-Threading is applied to resolve the issue and other times Mutex are used to co-ordinate between different events.

Both the server and client employ multi-threaded implementations to maintain a network connection with each other while processing the video stream. There is no particular reason for choosing threads over other concurrency models (e.g. event-driven or multi-process),but it should be noted that some type of parallel programming must be instituted to allow a server to both collect images from the video capture and stream it simultaneously. The ability of both applications to do tasks in parallel eliminates the issues of an element blocking the entire application.

Server Architecture: As previously stated, the server isa multi-threaded application. Threads are necessary for the server to be able to simultaneously maintain an Open CV window that displays the web camera input and send a Grey-scaled stream to its client. Therefore, the server employs two threads, each performing one of the aforementioned functions. The server will only allow a single client to connect and receive its video stream. Should the client exit or drop its connection with the server, the server would then accept the next connection in queue. In order for the server to accept any new connections, the same thread that is responsible for sending the video stream will reset the client socket and continue to listen until a new connection is formed has started, at which point the video stream would be sent to this new client.

Client Architecture: For the client to be able connect to multiple servers, process the video streams, and display each of the streams, concurrent programming is necessary. Similar to the server, each of the client's tasks requires its own thread in order to prevent the entire application from blocking. The command line interface (CLI) allows the user wants to add a new server stream to monitor, the IP and operating ports are the only parameters that are essential to establishing connection, specifying the width and height of the expected stream reduces complexity, so that video stream can be allocated accordingly. For flexibility, future implementation should also provide the ability to automatically detect the server's frame size automatically. The display thread manages all of the display windows that show the different video streams from all the servers it is connected to. The original design was to have a pair of threads that would be created for each server the client would connect to, but due the limitations in the Open CV, a single thread must initialization and updates all display windows.



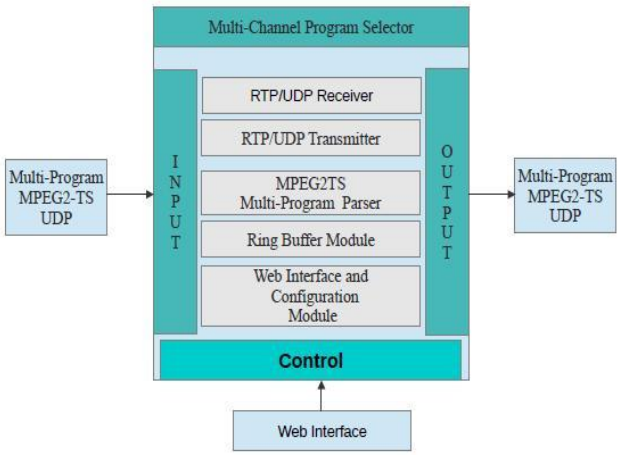
Drawing 2: Threaded Model [2]

For each connection to a server, the client creates a new networking thread, which manages and maintains a connection to a specified server. This thread is responsible for receiving the video stream from the server and updating the current frame for its corresponding server. If the connection to the server is dropped for whatever reason, the thread will reset the state of connection and attempt to reconnect to the server every three seconds.

SYSTEM DESIGN MODULE

The components setup diagram shows the detailed position of every module in the system:

Input to the system is Multi-Program MPEG2-TS Stream, Output to the system is Single Program MPEG2-TS stream, and Control/Parameters of the system are set by Web Interface, rest of the components form part of input, output, control and the core parsing.



Drawing 3: Component Setup

User Interface module: Acts as the only handle to the user to set and start the system. Input module provide the multi-program stream, output module provides the Parsed single TS stream.

MPPEG2-TS Multi-Program Parser: PAT Parser figures out the number of stream in the TS stream by parsing the PAT Packet and finding the number of available PMT Ids, Program List Generator get the number of program available in the MPEG2TS multi-program stream and writes it to the XML at specified address, PMT Parser records the details of the Individual Programs(video stream ID, audio stream ID)of respective PMT ID, Program Parser matches the Program Id of the elementary stream requested by the User Selection, Program Choice reader read the XML and send the selected program to the output module.

PAT DECODER FUNCTION

Step 1:Read 188 byte from MPEG2-TS file into Buffer(Buf),NoofProgram.
Step 2: Check (Sync byte=Buf[0]) = = 0x47.
Step 3: Check (PAT ID = Buf[1]&0x0f << 8 | Buf[2])==0x00
Step 4: check (Valid PAT=Buf[6]>>4) = = 0x0b.
Step 5: Section Length=(Buf[6]&0x0f << 8) | Buf[7].
Step 6:Payload Length = Section Length – 5.
Step 7: Make a PMT ID table in PMT[i]=Buf[15+i*4]&0x0f << 8 | Buf[16+i*4] i++.Payloadlength=Payloadlength – 4.
Step 8: Repeat 7 until Payload length = = 4 (CRC Length)
Step 9: NoOfProgram = i.

PMT DECODER FUNCTION

Step 1: Read 188 byte from MPEG2-TS file into Buffer(Buf),index=0,NoOfES=0.
Step 2: Check (Sync byte=Buf [0]) = = 0x47.
Step 3: Check (PMT ID = Buf[1]&0x0f << 8 | Buf[2])==PMT[i]
Step 4: Section Length=(Buf[6]&0x0f << 8) | Buf[7].
Step 5:Payload Length = Section Length – 9.
Step 6: Programinfo length=(Buf[15]&0x0f<<8) | Buf[16]
Step 7: index=13+ProgramInfo length + 4
Step 8: Make a Elementary Stream ID table inStream ID[index]=Buf[index]&0x0f << 8 | Buf[index+1],index=index+ESInfo length+4,Payload Length=Payload Length-5-ESinfolength. NOOfES++.
Step 9: Repeat 8 until payload length = = 4(CRC Length).
Step 10: Repeat from step 3 until PMT[I] = = NoOfProgram.

Program Selector:

Step 1: Read 188 byte from MPEG2-TS file into Buffer(Buf) .Step 2: Check Buf[4]=0x00 && Buf[5]=0x00 && Buf[6]=0x01
Step 3: Check headerPID==PMT[ProgramSelector].PES[i]
Step 4: Write buffer(buf) to output file.
Step 5: Repeat Step 3untilPMT[ProgramSelector].NoOfES

UDP/RTP TRANSMITTER/RECEIVER

TS streamer UDP/RTP module instance which streams the parsed program received from Program parser to the client machine and TS receive UDP/RTP module instance which receives the TS stream from the VLC player at the server side. Payload data which is Mpeg2-ts stream in this case is first enclosed by RTP header which in turn is enclosed by the UDP Header. The reason behind Introducing RTP is that it provide synchronization information at the receiver side which helps in case of out of order reception.

RING BUFFER

This module is used to store the data temporarily as the UDP transmission is asynchronous, this module stores data in a circular manner so in case of transmission delay the data is over written which a preferred handling in media buffering. This module is solely responsible for the smoothness of the playback at the client side. Ring Buffer is placed after every module for efficient data transfer, however a slight fault in tuning of the buffer can end up in sever data loss and eventually the video quality. Main tuning parameters are Buffer size, Number of buffers, and the delay.

WRITE BUFFER

Step 1: Create a circular linked list of N Buffer.
Step 2: Check write position = = N; write position = 0; write data.
Step 3: Check write position < read position || write position > read positionwrite data or position++.
Step 4: Check write position = = read position check overwrite = = 1; write data,write position++.

READ BUFFER

Step 1: Check read position = = N read position = 0;read data.
Step 2: Check write position < read position || write position > read positionread data or position++.
Step 3: Check write position = = read position do nothing.

ADVANCED ENCRYPTION STANDARD(AES) MODULE

This module perform two function: Encryption of the stream to be parsed on the transmission side and Decryption of the receive stream to be play on VLC player. In these module 188bytes of chunk of stream data is encrypted using 128 bit CBC mode of encryption function using EVP interface in openssl library with key and initialization vector producing 192bytes of cipher text generated by password. Same password is used to decrypt the data to be played on VLC. Thus it is providing an authentication to the video to be played on client side.

WEB INTERFACE MODULE

Apache Services handles the request on the server side, XML writer handles writing to the XML file, XML reader handle reading from the XML file. These XML file contain the Parameters such as IP address of the client, Number of programs in the stream and the selected channel number. First the server reads the client IP address which serves as handle for communication. Server writes the number of channel to the file and in return serve writes a channel of its choice to the XML file.

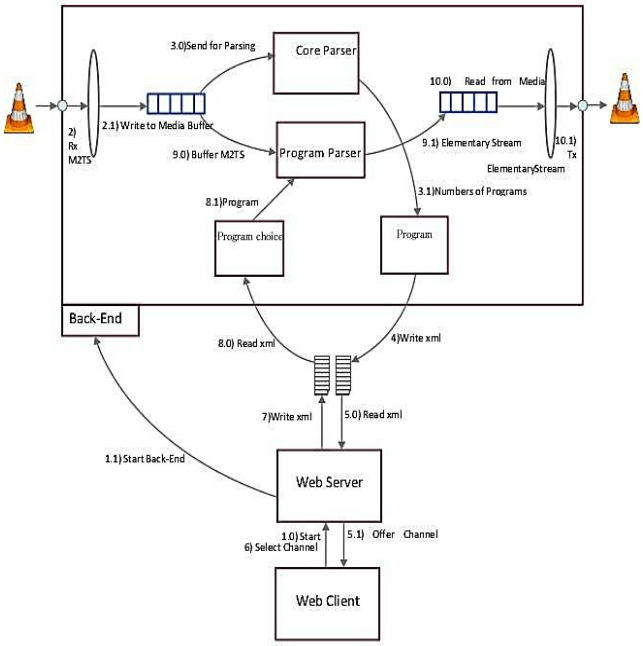
MODULE (COMPONENT) INTEGRATION

Step for the System Processing and showing System Synopsis in the given below diagram:

Step 1: UDP Receiver will receive the MPEG2 TS Multi - program stream from the VLC and write it into input media buffer 188 bytes at a time.

Step 2: Core Parser will read the input data from the buffer and pass it to PAT and PMT parser which will send necessarily information of Program ID to write XML.

Step3: The Program Choice Will check the program selected by the User (send through Web Servers on the XML).



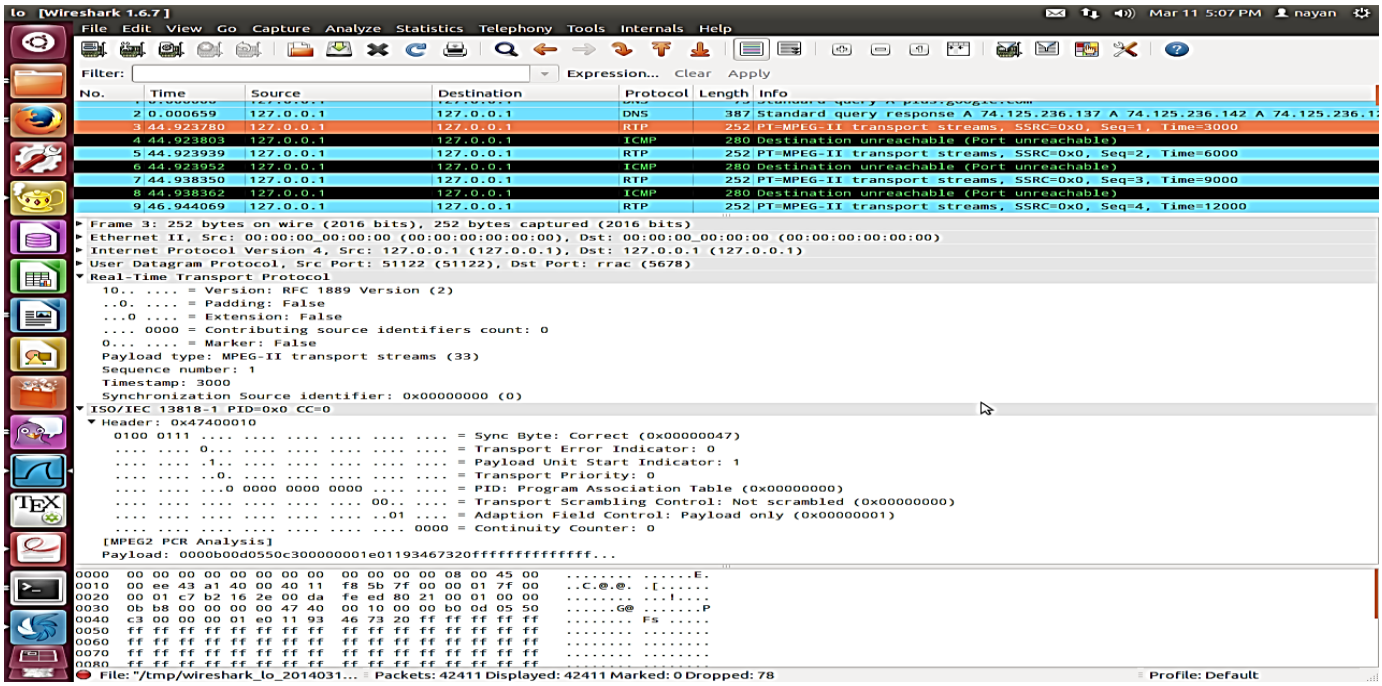
Step 4: Program Parser will be parsing the selected program with or without encryption on the output media buffer

Step 5: The UDP Transmitter will transmit the encrypted or without encryption data to the Client side VLC player.

Step 6: In encrypted stream the Decryption module will decrypt the Stream at the Client side and play it on the Client side VLC player (any Network player).

RESULTS

The whole system is divided into two parts namely the Client and the Server. Both this parts are formed of the same modules sets that are discussed above with just the exception of client not having the Parser parts(PAT Decoder, PMT Decoder, Program Selector) since that part is exclusively performed at the server relieving the client of the overhead of processing. After the Server streams the selected channel over the decided port to the client machine, the client listens to the receiving port and decrypts the streams in case it was encrypted or if the case is other way round it send the data directly (without decrypting) to the port which is used as source by the network player at client machine itself. Below is snap shot of the data received at the network player's source port. The data is observed to be perfectly in sequence and being identified as Mpeg2-TS stream carried RTP as is its payload.



CONCLUSION

An Introduction to the Mpeg2-TS technology is provided with a review on prevalent video streaming approaches. Considering the huge scope of this Technology, only a single well know area of application of this technology is tried to be replicated in this paper which is Video on Demand. The system discussed in this paper is a multi-threaded system with multiple modules Woking asynchronously to provide a video streaming framework. Asynchronous behaviour is possible due to the presence of Buffer manager module. AES encryption module gives a touch of propertiery software to the system. The core part of the system is formed by the Parser module which provides meaning to the system name which is mpeg2-ts multi-program channel selector. Channel selection is made by the Web interface module formed on PHP platform using XML files as means for parameter setting by the client. All the modules are independent of each other, yet they work closely with efficient tuning to deliver a seamless playback at the client side backboned by the RTP transmission and Reception. Other than this application the system can find use in various other scenarios with a little modification. One such scenario may be video surveillance of a large area powered by IP Cameras.

REFERENCES

[1]. IEEE BTS DL Seminar Rich Chernock CSO Triveni Digital, Downloaded on 15th November, 2013
[2]. Streaming Client and Server Model.PDF by Austin Alan Diec, Downloaded on 25th