CONSIDERATIONS FOR DIGITAL PROGRAM INSERTION OF MULTIPLE-VIDEO PROGRAMS

Aldo G. Cugnini
SpotOn[SM], an ACTV Company

Abstract

Applications are now being deployed that give multichannel video programming distributors the ability to deliver interactive and/or addressable (targeted) advertisements to homes equipped with digital set-top boxes. These applications, in concert with appropriately encoded bitstreams, provide the viewer of an enhanced program with an interactive experience. The initial use of this technology will be to bring the user enhanced advertisements, and to provide the advertiser with feedback on how viewers interacted with their advertisement. In order to enable this functionality, the enhanced advertisements must be inserted or "spliced" into the network programming in a seamless fashion at the headend. This can be accomplished by implementing the system described here.

INTRODUCTION

Digital Program Insertion (DPI) of Targeted Advertisements provides one means for accelerating the transition to digital cable by decreasing the complexity of local ad insertion equipment, and increasing revenue by providing additional spot sales opportunities. Targeted Ad Insertion can help subsidize the transition to fully Digital Cable, and can yield more bandwidth and MSO revenue by enabling a path to ultimately reclaim analog bandwidth. This technology will also further motivate the deployment of digital set top boxes as MSOs increasingly utilize non-customer based revenue.

An end-to-end system for inserting interactive ads into a network feed obtained from a satellite downlink will be described. Two different scenarios will be discussed, involving analog and digital source material. One scenario represents the situation where the downlink network is analog, or the feed must be decoded down to the analog level, e.g., to extract the analog cue-tone information needed by an ad server. The second scenario describes a system where the downlink signal format is digital, and DVS/253 cue information has been inserted into the stream. In this latter case, the video does not need to be decoded and re-encoded, a situation which greatly reduces complexity and cost.

In order to provide for seamless splicing, certain requirements must be satisfied in an MPEG bitstream. In addition to the proper handling of video frames and frame types, the set-top box video decoder buffer must be managed appropriately at the splice points. Several schemes for meeting this requirement will be discussed.

OVERVIEW OF DIGITAL PROGRAM INSERTION

Techniques for conventional single-program DPI are well known and will be summarized here. A typical DPI system is shown below in *Figure 1*. Because legacy equipment must be considered, a hybrid system is depicted that supports insertion into an analog network. A Traffic and Billing (T&B) System maintains the overall schedule of ads to be inserted. The ad server and ad splicer exchange timing and asset information concerning the scheduling of an ad insertion.

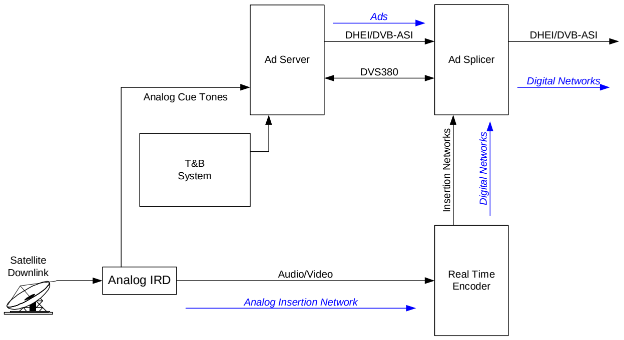

Figure 1: Digital Program Insertion.

In this scheme, analog cue tones are received from the insertion network and passed on to the ad server. Using DVS-380 protocol, the ad server and ad splicer establish a dialog to schedule and carry out a DPI event. Pre-compressed ads are delivered to the ad splicer at the appropriate time for insertion into the Insertion Network. A DVB/ASI or DHEI interface is then used to deliver the bitstream to the cable plant.

Several different technologies must be employed to realize this system, including bitstream multiplexing, bit-rate management, and cue message detection, as well as system intercommunication and event timing.

<u>INSERTION OF MULTIPLE VIDEO PROGRAMS</u>

In *Figure 2*, we see a new element, the SpotOn Controller, which is responsible for adding the necessary information to support Targeted Ad Insertion (TAI) in the set-top box (STB). Although the controller is shown as a separate element, its functionality could be integrated into the T&B System or the ad server; in some cases it could also be remotely located. Its purpose is to create a new data service, which will instruct the STB which of the various alternate programs (ads) should be viewed. These instructions are coded using a proprietary syntax called ACTV Command Language (ACL). This configuration can also be used to provide national distribution of targeted ads.

The STB program selection is enabled through the use of a small client resident in the STB, which decodes and acts upon the ACL commands. Using a user profile stored in the STB, the client switches the decoded video based on the ACL commands it receives.

There are two distinct “splices” which are necessary to implement a Targeted Ad Insertion: the headend ad insertion, where a multiplex of several ad programs is inserted into the program stream; and the STB program switch, where the STB switches between the different ads in the multiplex. Each of these splices must be done seamlessly, so that the viewer is unaware (if so desired) of the splice.

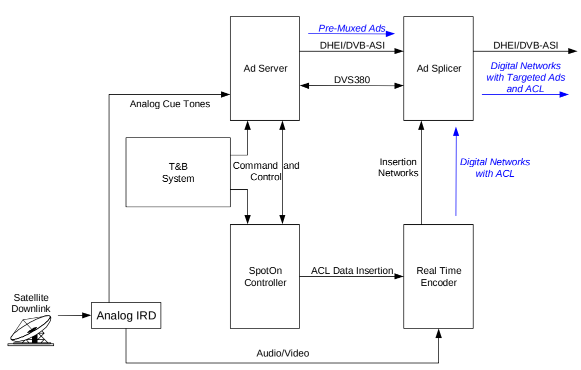

Figure 2: Targeted Ad DPI over Analog Insertion Network.

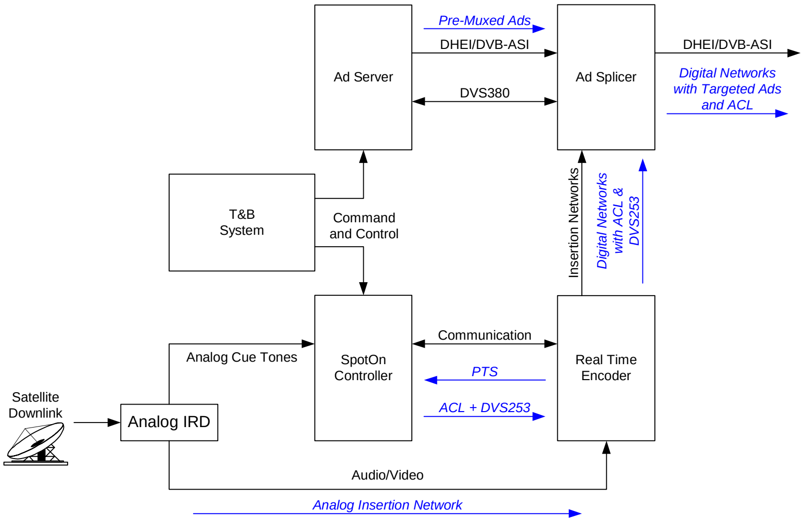

Figure 3. Insertion using DVS-253 Compliant Splicer.

The first-generation TAI system eases the real-time burden on the ad splicer by pre-multiplexing the "bundle" of ads before storing them on the ad server. Development of a real-time ad multiplex system will allow last-minute assembly of the ad multiplex.

INSERTION USING DVS-253

As ad splicers with DVS-253 capability become available, the system can evolve to support such devices, as shown in *Figure 3*. Since the insertion network is still analog here, there is no DVS-253 network messaging present. However, all the necessary information is available locally to synthesize these messages. The SpotOn controller can receive scheduling information and the analog cue tones upon which to base an insertion. After retrieving PTS information from the encoder, it can then assemble the appropriate DVS-253 message to hand off to the encoder, which then inserts the message into the stream.

DIGITAL INSERTION NETWORK

Ultimately, the evolution towards an almost fully-digital plant will enable the realization of the simple architecture shown in *Figure 4*. With the exception of locally produced analog programming, all video can ultimately be handled in compressed digital form. An IRT interfaces digital programming with embedded DVS-253 cue messaging directly to the ad splicer.

SPLICING CONSIDERATIONS

Bitstream Integrity

In order to provide for seamless splicing, certain constraints must be met in an MPEG bitstream, to assure that the STB video decoder is presented with an MPEG-compliant bitstream. The subject of proper switching of video frames and frame types is well known and will not be discussed at length here. For the purposes of our discussion, it is sufficient to assume that the streams must be "Closed GOP" and all splices must occur at GOP boundaries, so that the bitstream transition is from one intact "old" sequence to an intact "new" one.

In general, these conditions must be met at the input to the STB; hence, the input streams to the ad splicer can be

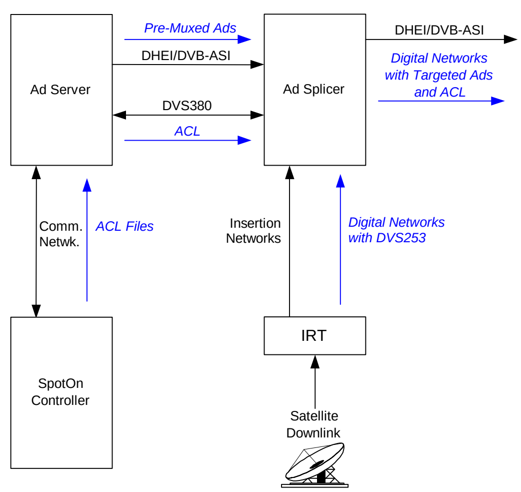

Figure 4: Fully-digital Program Insertion.

unconstrained, given a sophisticated splicer that can create these conditions at its output. However, certain new conditions must be present in the stored ad streams in order to facilitate splicing.

Multiplexing

Targeted Advertising is accomplished in existing digital STBs by commanding the Transport Demultiplexer to switch to a different video program (PID); see *Figure 6*. When the appropriate ACL command is received, the ACL Client instructs the Demultiplexer to switch to a different PID. The action of the client ensures that this switch is performed only at a GOP boundary.

In order to produce a seamless switch, the bitstream at the ad splicer must be assembled in such a way that any switch from one video (ad) program to any other results in an MPEG-compliant stream. (This includes a PID switch from the network video to one of the ad videos.) This requires all contemporaneous video sequences in the multiplex to start after and end before the transmission of any video from a previous or subsequent sequence, respectively. This "gap" is shown below in *Figure 5*.

This requirement must be met at any point deemed a "splice opportunity" at the STB. Note that, since a typical ad insertion will span several sequences, only the first and last sequences must meet this requirement. Thus, the required gap may be created at the time the video is encoded, and an entire multiplexed "package" can be stored on the ad server, greatly reducing the processing requirements of the ad splicer. Since this can be done in advance of airtime, and even in non-real-time, the process can be performed offline using a post-processing (software) algorithm after the video is encoded.

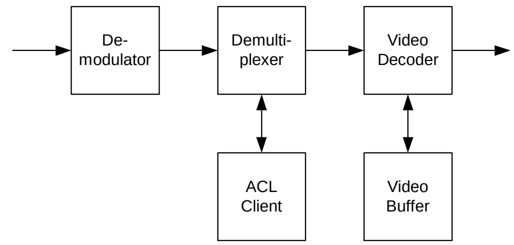

Figure 6: STB Decoder.

Alternatively, the individual ad videos can be encoded and stored on the ad server, so that a last-minute multiplexing can be performed. Of course, this requires more real-time processing in the ad splicer at airtime, but the added complexity may be desirable in order to support the added flexibility.

Bit-rate management

The video decoder buffer must be managed carefully at the splice points in order not to create an overflow or underflow condition. An MPEG video encoder ensures this by maintaining a predictive model called the Video Buffering Verifier (VBV). This model calculates how the decoder buffer will behave when presented with the MPEG bitstream. Because the action of the buffer can be completely predicted based on certain

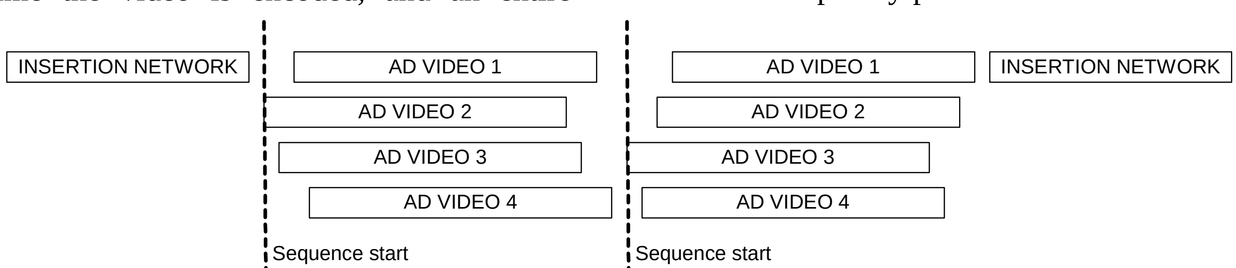

Figure 5: Bitstream Multiplex.

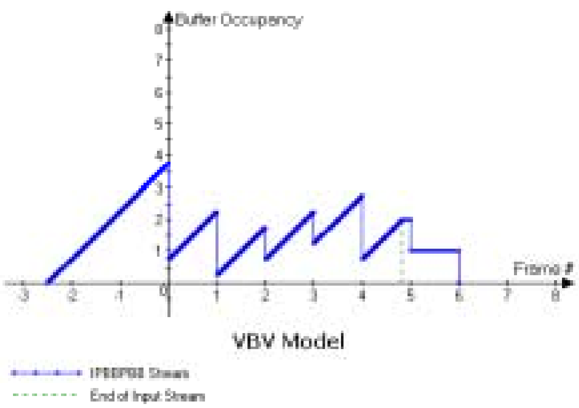

Figure 7. Video Buffer Trajectory.

bitstream parameters, and MPEG bitstream compliance requires that the VBV never overflow or underflow, proper operation of the decoder can be achieved by keeping an accurate VBV model in the encoder. See *Figure 7* above.

When presented with a Constant Bitrate (CBR) stream, the video buffer fills at a constant rate (linear slope of diagonal lines), and is emptied instantaneously by an amount equal to the size of each picture. The initial time that the buffer fills before the first picture is removed is the *vbv_delay* of that picture. It is important to note that if this sequence were followed immediately by another sequence, then the first bit of the new sequence would enter the buffer after the end of the first sequence. This would occur in the example at about time t = 4.8. In effect, this point becomes an available *splice point* in the old stream.

The result of blindly appending a new sequence is apparent in *Figure 8*. As an example, if we append the original sequence *to itself*, we see an undesired effect. Focusing on the region around the splice point, we can see that the buffer continues to fill at the video rate (as the size of each frame is unchanged from the previous example). However, the first frame of the new sequence must be removed at time t = 7. This requirement causes the buffer to fill for a time *less* than that originally specified for the first frame of the new sequence. In this example, we can see that the buffer fills for roughly 2.2 frames, whereas the original sequence called for a *vbv_delay* of 2.5 frames.

The consequence is that, upon the removal of the frame at time t = 8, the buffer underflows, i.e., not enough data has entered the buffer to ensure it is ready to be removed at the next access time. Since the new stream was encoded with the expectation of a specific VBV trajectory, this condition must not be violated at any downstream point. The *vbv_delay*, the bitrate, and the size of the new frames can only be modified if the resultant stream maintains VBV compliance.

It is important to note that the error will still exist at the end of the ad stream—it does not “flush” after one GOP, and we should not expect it to do so at any time in the future. The only way to correct the error is to re-establish the correct *vbv_delay* of a subsequent sequence.

One solution is to present the new sequence to the buffer at a time in advance of the decode time equal to the amount

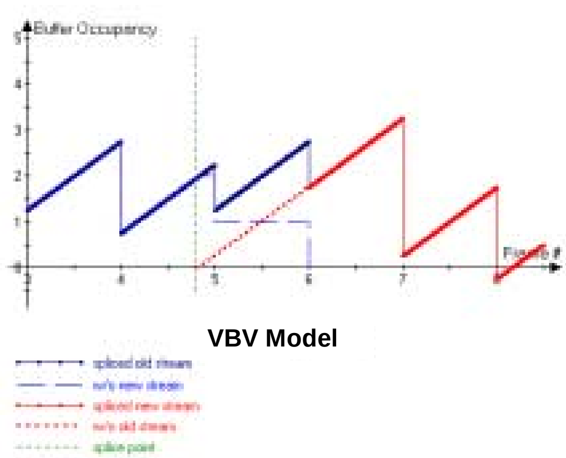

Figure 8. Buffer Occupancy at Splice Point (Bad Splice)

specified in the *vbv_delay* for the first frame of the new sequence. In other words, the last bit of the outgoing stream should remain in the buffer for a time equal to the *vbv_delay* parameter of the first frame of the new stream, minus the display time of the last frame of the outgoing stream.

One way to do this is to reduce the size of (i.e., re-code) the last picture(s) of the old stream. This is shown in *Figure 9* below. The size of the frame at time t = 4 was reduced from 2 to 1.5. This causes the old stream to end sooner, and allows the new stream to enter the buffer at the appropriate time, t = 4.5.

The specific solution depends upon the conditions at the splice point. If the outgoing stream terminates *before* the new stream should start, then null padding can be used to extend the life of the old stream in the buffer. If the outgoing stream would otherwise terminate *after* the new stream should start, then the last few frames of the old stream can be re-coded with fewer bits.

In this particular solution, all of the re-coding operations are accomplished in the ad splicer at airtime. Although the *vbv_delay* of the ad stream can be known in advance, the same parameter cannot be known of the network stream until it enters the ad splicer.

However, there are other solutions which distribute the complexity elsewhere. One such method is to force the *vbv_delay* of the first frame of the incoming network stream to a known specific value, so that the splicer can set up the conditions for a proper splice in advance of the new stream being available. This is the rationale for the *splice_decoding_delay* MPEG parameter, which is the specific value of *vbv_delay* needed to perform seamless splicing without post-modifying the bitstreams. This parameter essentially defines the point at which the old stream has finished entering the video buffer.

For an MP@ML stream at video rates up to 7.2 Mb/s, SMPTE 312M specifies a *splice_decoding_delay* of 250ms. Although this solution removes the constraints on the splicer, it places a large one on the bitstreams—one that can compromise the quality of the video. For this reason, it is not a preferred solution.

Another method is to use a large value of *vbv_delay* for the ad stream, coupled with relatively small picture sizes. This is shown below in *Figure 10*. Because the buffer is constrained in its excursion, any splicing errors will be less likely to result in over- or underflow. One advantage here is that this places no burden on the splicer; however, there is a large price paid in picture quality due to the limited picture sizes.

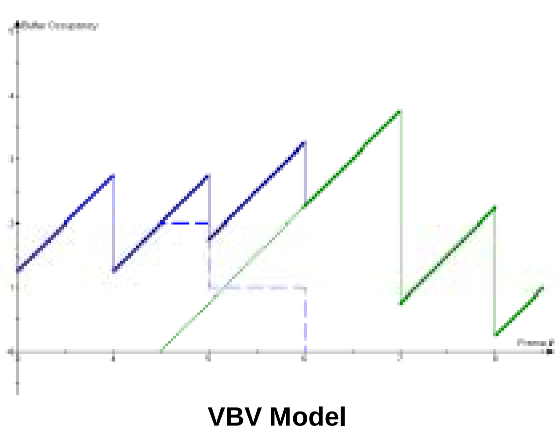

Figure 9. Buffer Occupancy at Splice Point (Good Splice).

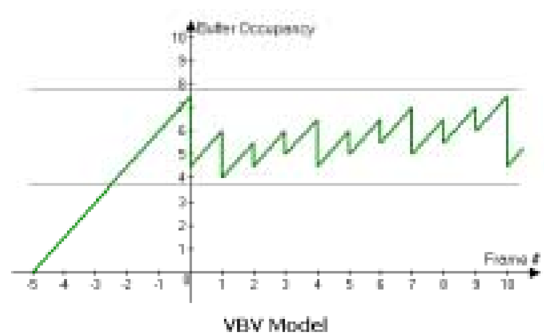

Figure 10. Effect of Large vbv_delay.

Due to the fact that the ad streams are pre-encoded in advance, other splicing solutions can be employed that maximize video quality, including dynamic combinations of the above and other proprietary techniques.

MULTIPLE VIDEO SPLICING

So far, we have focused our bitrate management discussion on that of single-video stream splicing into another single-video stream. We will now expand the discussion to that of a single-video program (or transport) stream splicing to and from a multiple-video stream. In order to properly perform these splices, we modify our recoding solution as follows. (Note that there are other solutions, as well.)

Single-video stream to multiple-video stream

1. The *vbv_delay* of each of the first pictures of the new stream is calculated. The longest *vbv_delay* sets the point at which the outgoing stream must terminate.
2. The end of the outgoing stream is shortened as needed by recoding the data into fewer bits.

Multiple-video stream to single-video stream

1. The *vbv_delay* of the first picture of the new stream is calculated. The *vbv_delay* sets the point at which the latest outgoing video component must terminate.
2. The end of each video component of the outgoing stream is shortened as needed by recoding the data into fewer bits.

Multiple-video stream to multiple-video stream

1. The *vbv_delay* of each of the first pictures of the new stream is calculated. The longest *vbv_delay* sets the point at which the outgoing stream must terminate.
2. The end of each video component of the outgoing stream is shortened as needed by recoding the data into fewer bits.

SUMMARY

An overview of digital program insertion of multiple video programs has been presented. Various solutions were described considering both legacy and emerging architectures, such as hybrid analog-digital systems and DVS-253 cue messaging. Bitstream considerations have been analyzed, including bitstream integrity, multiplexing, and bit-rate management. Splicing of multiple video programs can be accomplished by logical extension of single-program splicing techniques.

ACKNOWLEDGEMENTS

The author would like to thank his colleagues Mike Cristofalo, Frank Deo and Art Greenberg. for their useful discussions and contributions regarding this paper.

NOTICE

The reader's attention is called to the possibility that commercialization of the technologies described herein may require the use of inventions protected by patent and intellectual property rights.

REFERENCES

1) ISO/IEC 13818-1:1996(E), "Information technology - Generic coding of moving pictures and associated audio information: Systems," *MPEG-2*, April 4, 1996.

2) SMPTE 312M-1999, "Television—Splice Points for MPEG-2 Transport Streams," *SMPTE*, 1999.

3) SCTE 30 2001 (Formerly DVS 380), "Digital Program Insertion Splicing API," *SCTE*, 2001.

4) SCTE 35 2001 (formerly DVS 253), "Digital Program Insertion Cueing Message for Cable," *SCTE*, 2001.

5) Mukta Kar, Sam Narasimhan and Richard S. Prodan, "Local Commercial Insertion in the Digital Headend," *NCTA Technical Paper*, 2000.

6) D.T. Hoang, E. Linzer, J.S. Vitter, "Lexicographically Optimal Rate Control for Video Coding with MPEG Buffer Constraints," *Dept. of Computer Science, Duke University*, February 1996.