A Restamping Approach to Clock Recovery in MPEG-2 Systems Layer

Christos Tryfonas
Anujan Varma

UCSC-CRL-98-4
May 4, 1998

Board of Studies in Computer Engineering
University of California, Santa Cruz
Santa Cruz, CA 95064

ABSTRACT

This paper addresses the clock recovery problem while transporting MPEG-2 Systems Layer streams over packet-switched networks. The packet delay variation (jitter) introduced by the network affects the stability and thus the quality of the recovered clock. A decoder design methodology is described in which a jitter estimator that performs restamping on all the incoming packets containing clock values is used in conjunction with a standard phase-locked loop (PLL). A simple implementation of this methodology is described, where a new heuristic has been added to the standard PLL to eliminate the effects of the jitter. The methodology is evaluated by both analysis and extensive simulation experiments in a multi-hop ATM network using constant bit-rate MPEG-2 Transport Streams produced by hardware encoders with varying levels of cross traffic. The results show that the restamping approach outperforms standard dejittering methods, especially under heavy load conditions.

Keywords: MPEG-2, clock recovery, ATM networks, set-top box.

[0]This research is supported by the Advanced Research Projects Agency (ARPA) under Contract No. F19628-96-C-0038 and by the NSF Young Investigator Award No. MIP-9257103. The OPNET modeling tool used for simulations was donated to us by MIL-3, Inc.

1 Introduction

MPEG-2 is the emerging standard for audio and video compression. Being capable of exploiting both spatial and temporal redundancies, it achieves compression ratios up to 200:1, and can encode a video or audio signal to almost any level of quality. The MPEG-2 Systems Layer defines two ways to multiplex elementary audio, video or private streams to form a program: the *MPEG-2 Program Stream* and the *MPEG-2 Transport Stream* formats. The MPEG-2 Transport Stream is the approach suggested for transporting MPEG-2 over noisy environments, such as in a packet network. An MPEG-2 Transport Stream combines one or more programs into a single packetized stream with fixed-length packets. Using explicit timestamps (called Program Clock References or PCRs in MPEG-2 terminology) carried within the packets, MPEG-2 Transport Streams ensure synchronization and continuity, and provide ways to facilitate the clock recovery at the decoder end. Detailed descriptions of the MPEG-2 Systems Layer can be found in [15, 19].

Several issues need to be considered when transporting MPEG-2 encoded streams over packet-switched networks. These include the choice of the adaptation layer, method of encapsulation of MPEG-2 packets, choice of scheduling algorithms in the network for control of delay and jitter, and the design of the decoder. Concentrating on the decoder design, several approaches can be identified for recovering the system clock [9], depending on the accuracy and stability required by the application. In one category of applications, the reconstructed system clock is used directly to synthesize a chroma sub-carrier for the composite video signal. In this case, the chroma sub-carrier, the pixel clock and the picture rate are all directly derived from the system clock. The composite video sub-carrier must have at least sufficient accuracy and stability so that any normal television receiver's chroma sub-carrier PLL can lock to it, and the chroma signals which are demodulated using the recovered sub-carrier do not show any visible chrominance phase artifacts. In the case that the application has to meet NTSC, PAL or SECAM specifications, the requirements are even more stringent. For example, NTSC requires a sub-carrier accuracy of 3 ppm with a maximum long-term drift of 0.1 Hz/sec. In contrast, there is a second category of applications where the requirements on clock signal stability and accuracy can be relaxed significantly. For example, when picture and audio sample "slipping" is allowed, the system clock may not have stringent accuracy and stability requirements. In this case, the decoder need not use a PLL, but may operate from a free-running clock.

In this paper, we focus on the design of the system decoder and, in particular, the clock recovery problem for applications of the first type. This problem may arise while transporting MPEG-2 Transport Streams over packet-switched networks due to cell delay variation (jitter). The presence of jitter introduced by the underlying network or by the protocol layers below the MPEG-2 layer (such as adaptation layers) may distort the reconstructed clock at the MPEG-2 audio/video decoder. This, in turn, may degrade the quality when the synchronization signals for display of the video frames are generated from the recovered clock.

The jitter seen by an MPEG-2 packet stream at the receiving end may arise from three different sources: The first is the frequency drift between the transmitter and the receiver clocks, which is usually small compared to the other two components. The second component of jitter is due to the packetization at the source, which may displace timestamp values within the stream. Finally, the network may introduce a significant amount of jitter, owing to the variations in queueing delays in the network switches.

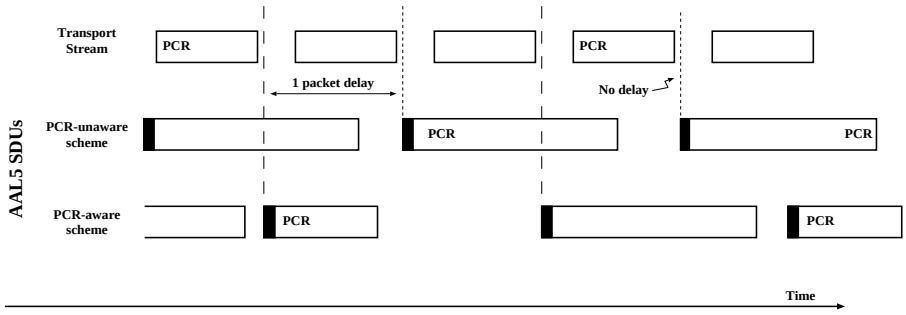

Figure 1.1: PCR packing schemes for AAL5 in ATM networks.

Packetization jitter is mainly caused by the packet encapsulation procedure. In the context of an ATM network, two distinct approaches have been proposed for encapsulation of MPEG-2 Transport Streams in ATM Adaptation Layer 5 (AAL5) packets [1]. In the *PCR-aware* approach, the packetization is done ensuring that when a Transport-Stream packet contains a PCR value it will be the last packet encapsulated in an AAL-5 packet. This reduces the jitter experienced by PCR values during packetization. In the *PCR-unaware* approach, the sender does not check whether PCR values are contained within a transport packet and may therefore introduce significant jitter to PCR values during the encapsulation, which in turn may affect the perceived quality of the video signal. The two approaches are illustrated in Figure 1.1.

Several approaches have been proposed for clock recovery from MPEG-2 streams in the presence of jitter. The traditional approaches use a PLL to recover the clock from the PCR timestamps transmitted within the stream. The presence of even a modest amount of jitter in this case can adversely affect the quality of the reconstructed clock. Several techniques have been proposed in the literature for improving the quality of the recovered clock. A common technique is to use a dejittering buffer at the receiver that absorbs the jitter introduced by the network. This makes the network transparent to the decoder phase-locked loop. A disadvantage of this approach is that it requires a priori knowledge of the maximum delay variation to avoid overflowing or underflowing the dejittering buffer. Several products are designed based on a maximum value of 10 ms for the jitter. Our simulations of MPEG-2 traffic over an ATM network showed that the jitter often exceeds 10 ms, severely degrading the recovered clock signal [19]. In addition, this approach wastes memory by using two separate buffers, the system decoder buffer and the dejittering buffer. Another approach to tolerate jitter at the receiver is to use special pre-filtering techniques to filter the delay variation before the PLL [7].

A third technique to minimize the effects of jitter in the clock recovery process is by counting the time difference between successive timestamps in the packet stream [10]. Although the jitter introduced by the network may be computed on a per packet-basis in this scheme, it requires constant spacing between timestamps in the packet stream, an assumption that may not hold in MPEG-2 Transport Streams. Finally, Akyildiz, et al. [1] proposed a simple method to deal with the packetization jitter of CBR MPEG-2 Transport Streams in an ATM network by subtracting a fixed offset from the received timestamps. This scheme, also called *Enhanced 2/2 scheme*, deals only with the packetization jitter, and is not designed to correct network-induced jitter.

All the above dejittering approaches attempt to maintain a constant buffer occupancy at the receiver and can therefore be applied to only constant bit-rate streams. In the case of a variable bit-rate stream, constant buffer occupancy is difficult to achieve without knowledge of the rate changes.

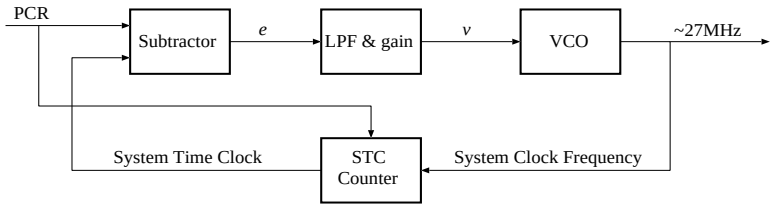

Figure 2.1: Block diagram of a PLL used in the MPEG-2 decoder.

These rate changes, in principle, can be determined from the PCR values in the stream using their piecewise linearity property [9]. However, changes in the transport rate cannot always be determined exactly from the PCR values. An interesting solution to this problem was proposed by Hodgins and Itakura [8], where a rate change indicator is sent within the stream. However, this scheme requires changes to the MPEG-2 standards. Alternative approaches for clock recovery in variable bit-rate streams include the use of a control system for frequency estimation and adjustment in order to provide constant average delay through the buffer [17].

This work is motivated by our observations from extensive simulations of MPEG-2 Transport Streams over ATM networks [19]: Although we found that the quality of the reconstructed clock was degraded even with moderate amounts of jitter, the jitter did not cause the MPEG-2 system decoder buffer to overflow or underflow. This suggests the possibility of combining the two buffers —the dejittering buffer and the system decoder buffer— and providing a constant amount of dejittering space in the system decoder buffer by subtracting an offset from incoming PCR values. The idea of providing a dejittering space in the MPEG-2 system decoder buffer was first proposed by Rangan, et al. [14]. Their scheme subtracts a constant offset from incoming timestamps to establish a dejittering space in the system decoder buffer. However, the stability and accuracy of the reconstructed clock are still affected by the jitter. We propose a simple algorithm to minimize the effects of jitter on clock recovery by using a jitter estimator to calculate the jitter on a per-packet basis and *restamping* incoming packets based on the estimated jitter. This avoids the need for a separate control system to estimate jitter. This scheme is general and can be used to correct both source-induced and network-induced jitter. Results from simulations of real MPEG-2 Transport Streams over ATM networks with varying levels of cross-traffic show that the quality of the recovered clock is substantially improved over other approaches.

The rest of this paper is organized as follows: Section 2 provides a general overview of the proposed decoder architecture and the clock recovery scheme. Section 3 describes the simulation experiments performed with MPEG-2 Transport Stream traces in an ATM network to validate the scheme. Section 4 contains an analysis of the stability, accuracy, and dynamics of the scheme. Finally, Section 5 concludes the paper with a summary of the lessons learned.

2 Restamping Algorithm

A typical clock recovery system found in an MPEG-2 decoder is shown in Figure 2.1. The PLL works as follows: Initially, the PLL waits for the reception of the first PCR value for use as the time base. This value is loaded in the local STC (System Time Clock) counter and the PLL starts operating in a closed-loop fashion. When a new PCR sample is received at the decoder, its value is compared with the value of the local STC. The difference gives an error term e. This error term is then sent to a low-pass filter (LPF) which is designed according to the specific application. The

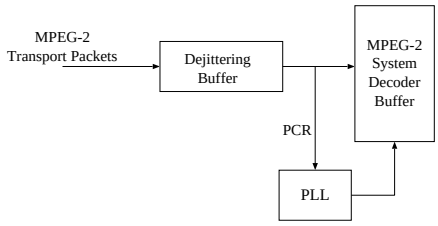

Figure 2.2: Architecture of an MPEG-2 decoder with a dejittering buffer.

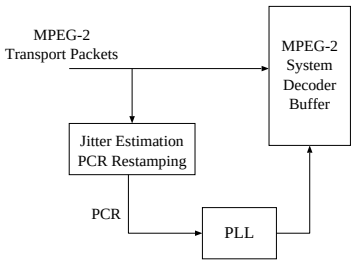

Figure 2.3: Architecture of an MPEG-2 decoder with a general restamping mechanism.

output of the LPF controls the instantaneous frequency of a voltage-controlled oscillator (VCO) whose output provides the decoder's system clock frequency. The VCO's central frequency is fixed at 27 MHz. Ideally, when the jitter is only due to the frequency difference between the encoder and decoder clocks, the error signal e will reflect this difference. In the presence of jitter from other sources, however, the error signal e will not reflect this actual frequency difference. This may affect the quality and accuracy of the recovered clock.

The classical approach to jitter compensation is to use a jitter compensation buffer, as shown in Figure 2.2. The jitter compensation buffer attempts to equalize the delay for each packet, so that the relative timing of packets at its output corresponds to that at the transmitter. This, however, requires a separate dejittering buffer with its own control system. In addition, its design requires knowledge of the maximum jitter, so that overflows and underflows can be avoided.

An alternative approach to dejittering buffer is to modify the PCR timestamp values in the incoming stream to compensate for the jitter. We refer to this approach as *restamping*. One method to perform restamping is by means of a jitter estimator, as shown in Figure 2.3, that estimates jitter on a packet-by-packet basis. In the ideal case, the jitter estimator is able to determine the exact value of the jitter in number of ticks of the encoder's clock and subtract it from the incoming PCR value. The resulting error term would then correspond to the actual phase difference due to frequency difference between encoder and decoder.

Although the architecture shown in Figure 2.3 with a separate jitter estimator control system can provide close-to-ideal results, its complexity may be unacceptably high because of the jitter estimation control system. It is also difficult to design a good jitter estimator for variable bit-rate streams. By modifying the decoder PLL, we can minimize the effects of jitter in a way equivalent to having a separate jitter estimation circuit. This provides the basic motivation for our scheme.

The basic idea behind our algorithm comes from the fact that the phase difference in the PLL arises from three sources: frequency difference between encoder and decoder, jitter due to network congestion, and packetization jitter at the adaptation layer. The first component is usually small compared to the second and third. Thus, if the magnitude of the resulting error term e crosses a pre-determined threshold $\mathcal{T}_f$, we can interpret it as being caused by reasons other than the frequency difference between the encoder and decoder clocks. In such an event, we can scale the error term e using a factor g_2, where $0 < g_2 < 1$. Therefore, the standard PLL architecture shown in Figure 2.1 is modified to the one shown in Figure 2.4.

The algorithm performs restamping of the incoming PCRs with different weights depending on the actual value of e as illustrated in Figure 2.5. This is equivalent to changing e. Formally, the

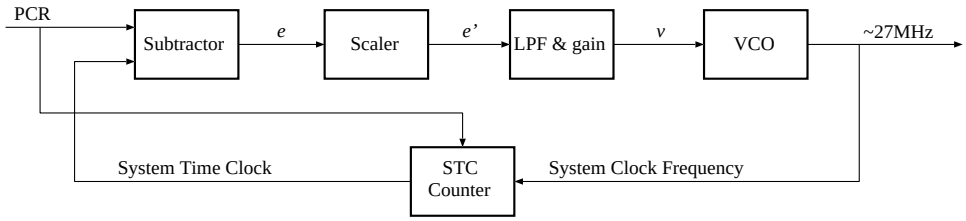

Figure 2.4: Block diagram of the enhanced PLL.

algorithm performs the following function on a PCR arrival in addition to performing the classical PLL function:

$$\begin{aligned} &\text{if } |e| < \mathcal{T}_f \\ &\text{then} \quad e' = g_1 \times e; \\ &\text{else} \quad e' = g_2 \times e; \end{aligned}$$

where T_f is a selectable threshold and g_1, g_2 are the *downpressure factors* used to scale e. In general, $1 \geq g_1 \gg g_2 > 0$ in order to minimize the unexpected jitter. The threshold $\mathcal{T}_f$ can be derived using the minimum frequency requirements for placing PCRs in an MPEG-2 Transport Stream and taking into consideration the worst-case settling time for an existing PLL, and is equal to the maximum phase difference during the settling time interval. Assuming the worst frequency change when the encoder and the decoder have the maximum allowable frequency difference, which is 60 ppm or 1620 Hz according to MPEG-2 standard [9], and t_s the settling time for this frequency difference, an approximate upper bound for $\mathcal{T}_f$ that assumes the maximum frequency difference over the loop acquisition time, is given by

$$\mathcal{T}_f \leq t_s \times 1620. \tag{2.1}$$

The settling time t_s can be calculated knowing the internal parameters of the PLL. Several enhancements could be made in the basic heuristic presented above. In a more general case, the threshold $\mathcal{T}_f$ can be varied dynamically as a function of the phase error during loop acquisition. This allows $\mathcal{T}_f$ to have smaller values in steady state than in the static case, The problem in this case is to estimate the current phase-error correctly when heavy jitter is present. A robust approach to estimating jitter was presented by Singh. et al. [17], where the phase error is averaged using a time-averaging algorithm in which the length of the averaging periods is not constant. Using a similar approach, the parameter $\mathcal{T}_f$ can be made variable for the loop acquisition time by using a simple function. If we denote by t_s the maximum settling time after a frequency change, t_m the maximum phase difference, and t_m at which this maximum phase difference occurs, then the following piecewise linear function could be used:

$$\mathcal{T}_f = \begin{cases} A + \frac{t}{t_m} e_m, & t \leq t_m; \\ (A + e_m) - \frac{e_m}{t_s - t_m}(t - t_m), & t_m < t \leq t_s; \\ A, & t > t_s; \end{cases} \tag{2.2}$$

where $0 \leq A$. Lower values for A make the system more immune to jitter whereas higher values make it more responsive. In any case, $\mathcal{T}_f$ should be at least equal to the running phase error at each time instant. The function above is illustrated in Figure 2.6. If the PLL of the decoder produces a

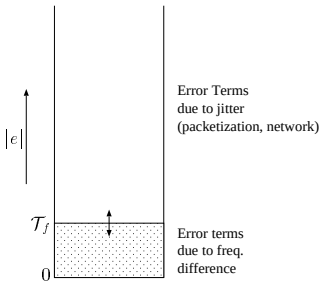

Figure 2.5: Zones used in the algorithm and their meaning.

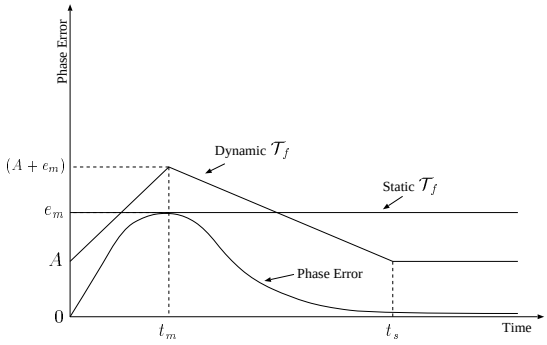

Figure 2.6: Illustration of static vs dynamic threshold $\mathcal{T}_f$.

non-zero phase error after a frequency change, then, for correct operation, this phase error should be the reference point and should be subtracted from the error terms before the computations, and added later on. Since $\mathcal{T}_f$ represents the phase error that is allowed due to frequency difference and e reflects the instantaneous phase difference, a final non-zero phase error e makes the computation biased and therefore the allowable phase error $\mathcal{T}_f$ should be counted above this non-zero phase error.

The two downpressure factors (g_1, g_2) can be varied, particularly during the loop acquisition time in a way similar to that of averaging $\mathcal{T}_f$ to facilitate the clock recovery process when only the first clock value suffered a very high jitter. In particular, g_2 should be close to g_1 when PLL starts acquiring the new frequency and equal to its final (low) value after t_s time. A linear function could be used for g_2 for the loop acquisition time interval. Another option is to use more than two zones to identify the various components of jitter. If the standard PLL is very immune to noise, then more zones give high flexibility to minimize the effects of jitter without sacrificing high responsiveness. In our experiments, the use of two zones seemed to be sufficient to obtain good behavior of the restamping method.

It should be noted that, the modification to the PLL in our algorithm primarily affects the amplitude of the signal e. There might be cases in which lower phase values result in higher error terms than higher phase values (e.g., when the two phase values are close and fall on opposite sides of $\mathcal{T}_f$). Even though the computed error term may result in a wrong initial decision in those cases, this will not persist and eventually the PLL will reach a stable state with the correct frequency.

The restamping approach has some disadvantages as well. As shown in Section 4, if the transport packet carrying the first PCR experiences maximum jitter and all the other incoming packets with PCR values have negligible jitter, then the locking time may be high depending on g_2 due to low gain. Besides, although the method proposed minimizes the effects of high jitter autocorrelation [2, 19], it does not eliminate the problem since it only compresses the error term.

In general, restamping algorithms based on heuristics similar to the one described earlier in this section estimate the frequency based on packets that have delays falling into a small zone called the *clocking delay zone* (Figure 2.7). We define this class of restamping algorithms as *clocking delay zone* ($\mathcal{CDZ}$) class. Restamping algorithms that belong to the $\mathcal{CDZ}$ class compact the incoming signal and use the PCR values within the clocking delay zone to drive the PLL. In the loop acquisition phase, the clocking delay zone may not be static but continuously changing if the phase difference is not taken into consideration. Similar error terms e may be produced by packets with different delays because of the increasing (or decreasing) phase difference initially. Although this results in

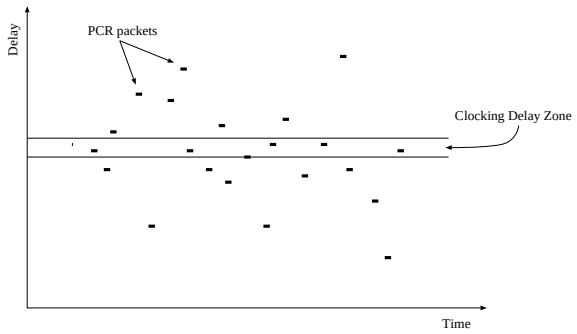

Figure 2.7: Illustration of the clocking delay zone concept.

the worst behavior for specific cases, it is still better than a standard PLL with gain equal to g_2. The clocking delay zone can be made more stable if the phase difference is taken into account. The phase difference can be subtracted from the resulting error term prior to the computations and then added back to the result. The last calculated error term can be used in order to give an estimate of the running phase difference used by the procedure described above.

The $\mathcal{CDZ}$ algorithms are most effective when the PLL is locked and high-amplitude noise is present in the delay of PCR values. In this case, the clocking delay zone is maintained close to the average delays found in the network. If the resulting error terms fall within the low-gain zone consistently, however, the clocking delay zone will drift slowly towards the direction of the new average delay in the network. With bimodal delay distributions, as in the case of the PCR-unaware scheme, the clocking delay zone may not stay on one side of the packet delay distribution depending on the ratio of numbers of packets containing PCR values between the two modes, and the choice of $\mathcal{T}_f$. If, however, the average delay is in between, then the clocking delay zone may be driven between the two modes resulting in a stable system since all the PCR values will fall into the low-gain zone. In our first experiment, the clocking delay zone remained in one mode of the packet delay distribution resulting in an almost perfect behavior similar to the Enhanced 2/2 scheme [1]. An interesting observation is that both the Enhanced 2/2 scheme with its variations and the various dejittering approaches make use of the clocking delay zone concept and fall indirectly into the $\mathcal{CDZ}$ class. In the Enhanced 2/2 scheme this zone is not stable since all odd (or even) numbered packets fall into it regardless of their delay. When the PLL becomes locked, packets belonging to this zone have similar delays with high probability.

Comparing the restamping approach to the dejittering approach, the latter results in a stable clocking delay zone when the dejittering buffer does not overflow or underflow. When the dejittering buffer overflows or underflows, however, the clocking delay zone moves to the new average delay instantly, affecting the quality of the recovered clock. The restamping algorithms move the clocking delay zone gradually to the new average delay producing a smoother recovered clock. In any case, the restamping algorithms can be combined with other schemes such as Enhanced 2/2 and dejittering to improve the quality of the recovered clock.

The clock recovery method described above does not identify any underflows that may occur in the system decoder buffer. This needs to be taken into account separately. We follow the same approach as the one proposed in [14] to impose a constant amount of dejittering space in the system decoder buffer. According to this approach, we delay using all the incoming PCR values by a time interval equal to the network jitter we want to absorb, which is equivalent to subtracting a constant

value (jitter converted to ticks of MPEG-2 clock) from all the incoming PCR values.

The restamping methods described do not count the jitter on a per packet basis, but only at the time instants when a new PCR sample is received. This makes them immune to packet losses, as well as attractive for use with VBR MPEG-2.

3 Simulation Results

An ATM network with varying levels of background traffic was used in order to test the algorithm. In our experiments with ATM networks, we assume that the adaptation layer is Adaptation Layer 5 (AAL5) which was initially proposed to carry data traffic over ATM networks. The results of our experiments indicate that the algorithm gives very good performance in most cases minimizing the effects of jitter from all sources.

3.1 Simulation Model

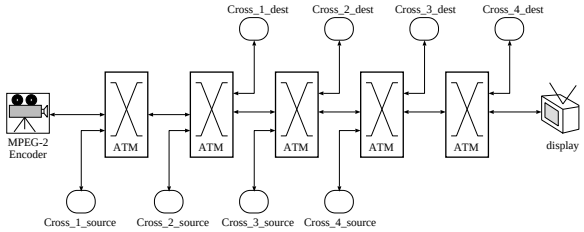

Figure 3.1: Network topology used in the simulations.

The network topology used is shown in Figure 3.1. It consists of five cascaded ATM switches. The switch nodes are non-blocking, output-buffered crossbar switches. The MPEG-2 Transport Stream is sent through all the cascaded switches to the display device at the other end. At each hop of the network, the end-to-end video stream shares the network link with cross traffic generated by a set of cell sources. All the cross-connections are between nodes that are connected to adjacent ATM switches. The propagation delay for each network link is set to 1 msec. To study the effect of scheduling policy in the switch on the end-to-end behavior of the video streams, we simulated both the FIFO scheduling policy and a fair-queueing scheduler that provides bandwidth guarantees to the end-to-end session. The actual fair-queueing algorithm simulated was *Frame-based Fair Queueing (FFQ)* [18]. The frame-size parameter in the FFQ algorithm was chosen as 3 ms in all the experiments.

The cross traffic sources generate ATM cells based on the ON-OFF traffic model (Figure 3.3). Both ON- and OFF-periods are exponentially distributed. Cells are sent during an ON-period at the peak rate of the link. The burstiness of the sources can be controlled by varying the mean length of the ON and OFF periods. In all simulations, we modified the number of cross-traffic connections through each link as well as their ON and OFF periods to vary the total load on each link.

The protocol stack of the simulation model at the MPEG-2 encoder end consists of the ATM layer, the adaptation layer, and the actual application layer from which the MPEG-2 transport packets are sent, as shown in Figure 3.2. At the adaptation layer, we simulated both the PCR-aware and PCR-unaware schemes.

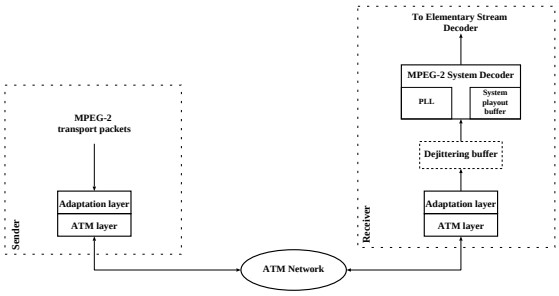

Figure 3.2: Protocol stack of the simulation model.

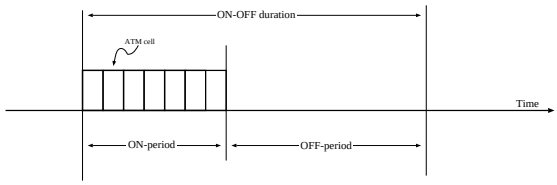

Figure 3.3: ON-OFF traffic model.

At the decoder end, the protocol stack consists of the ATM layer, the adaptation layer, an optional dejittering buffer and the MPEG-2 system decoder. The MPEG-2 system decoder includes the PLL used to recover the clock and the system playout buffer. The elementary decoders for each elementary stream present in the MPEG-2 Transport Stream are not incorporated in the model. All the simulations were performed using the OPNET simulation tool.

3.2 Description of Traces

The two traces we used are based on the CBR MPEG-2 Transport Stream format and were produced from hardware MPEG-2 system encoders.

The first trace A has a transport rate of 9.4 Mbps. It consists of one program that contains five elementary streams:

- One MPEG-2 video elementary stream.
- Two MPEG-1 audio elementary streams.
- Two more elementary streams that are used for other purposes such as teletext.

Two more PIDs are allocated for the Network Information Table (NIT) which acts as program zero, although they are not used. Trace A has a length of approximately 23 minutes. The MPEG-2 video stream is encoded from a PAL video signal with a frame rate of 25 Hz. The number of frames contained in the video stream is 34859. Another interesting characteristic of the trace is that the PES packets are of variable size and in the case of the MPEG-2 video elementary stream, each PES packet corresponds to exactly one frame. Clock information, in terms of PCRs, is sent through the MPEG-2 video elementary stream.

Since only one program is contained in the trace, there is no multiplexing involved among different programs. Thus, null transport packets needed to be placed in the trace in order to obtain a constant bit-rate. The number of null transport packets found in the trace was 1988032, accounting for a total bandwidth of 22.8%.

Although the transport rate of the stream was specified as 9.4 Mbps, the rate computed from the PCRs found in the trace was slightly different. The average transport rate over the entire length of the trace was found to be 9.399982236616 Mbps. Figure 3.4 shows the transport rate computed by the first 200 PCR values in the trace. Sending the stream at 9.4 Mbps would have introduced significant jitter at the decoder, increasing its locking time. To ensure stability of the decoder clock during the simulation time, a transport rate of 9.399994 Mbps was selected at the source after performing several experiments with different transport rates (see Figure 3.4). Since the actual trace

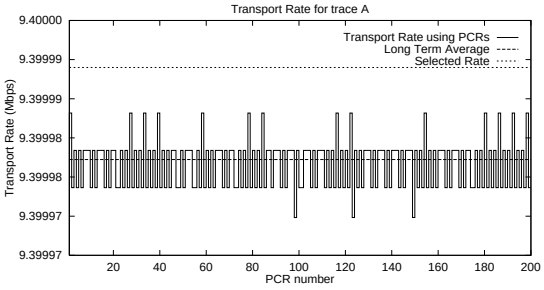

Figure 3.4: Transport Rate of trace A.

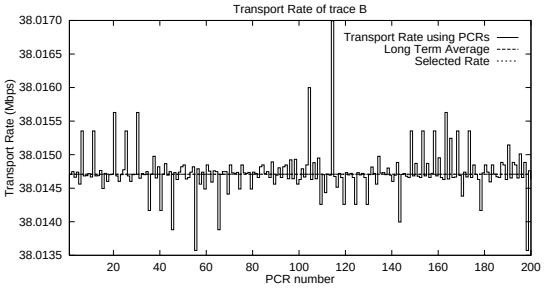

Figure 3.5: Transport Rate of trace B.

has a clock drift, the PLL does not lock at exactly 27 MHz but at (27 MHz + 1.6 ppm) as illustrated in Figure 3.6. This introduces a constant non-zero phase error even after the PLL is locked.
The second trace B is a high bit-rate trace that multiplexes 5 programs. Each program consists of:

- One MPEG-2 video elementary stream.
- Five MPEG-2 audio elementary streams.

Besides, information for the Network Information Table (NIT) has been placed in the stream as program zero. Since all the programs are similar, the first program was selected for the simulations. The length of the trace is 93.56 secs. The format of the video elementary stream is NTSC with frame rate of 29.97 Hz.

Since this trace was also produced by a hardware encoder, the transport rate computed from the PCRs in the trace is not constant, as is evident from Figure 3.5. An actual transport rate was not specified in this case. Thus, the long-term average (38.014707 Mbps) was used to transmit the stream to the network. The reason is that the long-term average is the same as the average transport rate during the simulation time interval for this case. Since the jitter introduced due to packetization in the PCR-unaware case is negligible and does not affect the lock time, and the frequency of the occurrences of PCR values is larger than in trace A, the time to lock on the 27 MHz frequency was small compared to the duration of the simulation. The total number of transport packets in the stream was 2364900 and the number of null transport packets needed for padding was 488844. Thus, the wasted bandwidth is approximately 20.6%.

The block diagram of the standard PLL used in the MPEG-2 decoder at the receiver is shown in Figure 2.1. We evaluated a number of LPF designs in order to select the most appropriate one for the simulations. All the LPFs considered were Butterworth LPFs with different orders and cutoff frequencies. The designed PLL must have a bandwidth that is much more narrow than the one of the demodulator sub-carrier reconstructor which is about 100 Hz [2]. Thus, different cutoff frequencies around 0.1 Hz were considered in the LPF selection process. The LPFs selected for the simulations was a second order digital Butterworth Low-Pass filter (LPF) with cutoff of 0.1 Hz and a sampling frequency of 30 Hz. This cutoff frequency was derived from both experimental results of [19] and the analysis done in [10].

Since the arrival times of the transport packets containing PCR values may not fall exactly at the sampling points of the filter, the actual PCR value is changed to the one that could have arrived at the next sampling tick at the decoder's frequency (which may not be the same as the encoder's one). This quantization process introduces a small error which, however, does not affect the convergence time and the steady-state behavior [2].

The loop acquisition time of the PLL affects the amount of additional buffering needed at the decoder. The phase difference is defined as the difference between the time at which an access

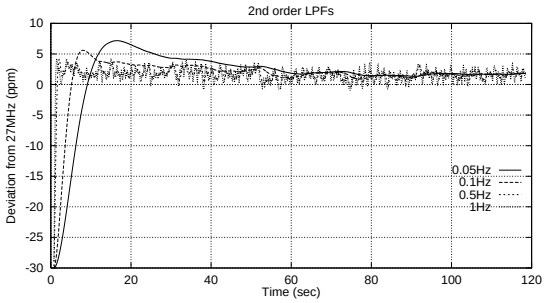

Figure 3.6: Decoder frequency using PCR-unaware and several second-order LPFs with different cutoff frequencies.

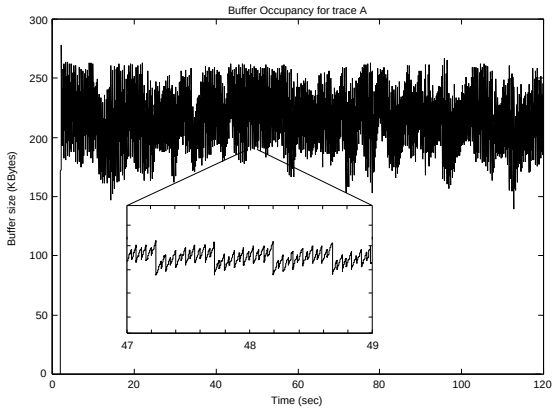

Figure 3.7: Buffer occupancy for trace A under PCR-unaware scheme using standard PLL.

unit is forwarded from the system decoder buffer in the absence of any network-imposed jitter, and that observed in the simulation experiment. In the majority of our experiments with trace A, the maximum buffer occupancy did not change and this maximum occurred just before the first access unit was forwarded to the elementary decoder. The reason is that at the time the first access unit needs to be forwarded, the phase difference is very small (at most 1000 ticks). During the time that corresponds to this phase difference (around 0.1 ms) at most one additional transport packet could be received. Even in that case, since the stream is padded with a large number of null transport packets, the total buffering may not change because a null transport packet is never placed in the system decoder buffer. The reason why the maximum occurs at the first access unit is because the first access unit from trace A to be forwarded is an I-frame, then a B-frame, and after that another I-frame, resulting in the only case in which two I-frames are forwarded in a frame sequence consisting of three frames. Since the maximum phase difference occurs some time after the first access unit is forwarded to the elementary decoder and this phase difference cannot exceed the first maximum, the maximum buffer occupancy during the experiments is not affected.

The sequences of I-, B- and P-frames in trace A are shown in the magnified view in Figure 3.7 of the first experiment, as well as the fact that each PES packet corresponds to exactly one frame for the video elementary stream. As can be observed, there is a fixed minimum buffer occupancy of 150 KBytes which has to do with the phase difference between the first PCR value and the PTS value of the first access unit, and is a characteristic of the actual trace. In this case, this difference gives us a cushion against underflows even with excessive jitter. This amount of buffering corresponds to almost 350 ms or 8.75 frames (sequence of I-, B- and P-frames). A similar behavior applies to trace B.

The voltage-controlled oscillator (VCO) of the PLL was designed to work according to the following formula:

$$\text{STC frequency} = 27\ \text{MHz} + \frac{810}{30000}v, \tag{3.1}$$

where v is the filtered difference between the current STC value and the incoming PCR value. The design of the VCO takes into account the maximum difference in ticks of a 27 MHz clock when the jitter is at its maximum allowed value. Since, according to MPEG-2 standard [9], the maximum

jitter expected is around 1 ms, this difference is around 30000 ticks. For this maximum difference, the STC frequency must operate within the limits defined by the standard [9].

The enhanced PLL architecture depicted in Figure 2.4 is used for the restamping method. Three schemes for doing the restamping were used throughout the experiments in addition to the standard and the Enhanced 2/2 scheme, whenever applicable. In the first scheme (restamping), the standard algorithm presented in Section 3 is used with $g_1 = 0.98$, $g_2 = 0.005$ and $\mathcal{T}_f = 3000$, all statically assigned. The first variation of the restamping method is when three zones are involved which may help in the loop acquisition in specific cases. In that case, the values assigned to the variables of the algorithm are $\mathcal{T}_{f1} = 3000$ and $\mathcal{T}_{f2} = 25000$ for the two thresholds and $g_1 = 0.98$, $g_2 = 0.5$ and $g_3 = 0.005$ for the three downpressure factors. The second variation incorporates two zones and introduces the notion of variable gain for the high zone during the loop acquisition time. The function used for g_2 is given by:

$$g_2 = 0.7 + ((0.005 - 0.7)/20) \times t, \tag{3.2}$$

where g_2 is decreasing linearly with respect to time t from 0.7 to 0.005 in a 20 seconds time interval. The dejittering approach uses buffering to absorb the jitter and assumes a priori knowledge of the exact rates of the traces, which makes it an idealized dejittering scheme.

Experiments 1 to 3 assume link capacities of 30 Mbps and use trace A whereas the last two experiments make use of OC-3 links (155 Mbps) and trace B.

3.3 Experiment 1

The goal of this experiment is to study how packetization jitter affects MPEG-2 performance. Since no cross-traffic is involved in this experiment, FIFO scheduling is adequate to compare the restamping approach with a standard PLL. The rate of the MPEG-2 source is approximately 10.5 Mbps including the overhead from the adaptation layer.

The delay experienced by transport packets containing PCRs for the PCR-unaware case is plotted in Figure 3.8. In this case, almost all of the jitter observed at the receiver is due to the packetization at the source, which is approximately 150 μsecs. This jitter affects the instantaneous phase difference (Figure 3.9) resulting in quality degradation of the recovered clock.

The recovered clock at the decoder is shown in Figure 3.10. The standard method suffers from the variation due to packetization jitter and gives the worst results whereas the Enhanced 2/2 scheme gives the best since it transforms the packetization jitter into ticks and adds it back to the PCRs of the odd-numbered transport packets. All the restamping methods fall between the two, giving good control over the packetization problem. In the case that the phase difference becomes zero after the loop acquisition (as is the case of second-order PLLs or higher when triggered by a frequency change), the performance of the restamping methods would have been very close to the optimal Enhanced 2/2 case. The same would have been the case if the constant phase difference that is present after the loop acquisition was subtracted from the resulting error term before the restamping computation takes place, and added back to the output of the restamping calculation. This is shown in Figure 3.11 in which $\mathcal{T}_f$ is set to 500 in order for the PLL to be more selective. The clocking delay zone is illustrated in Figure 3.12 and is derived from the upper zone of Figure 3.8.

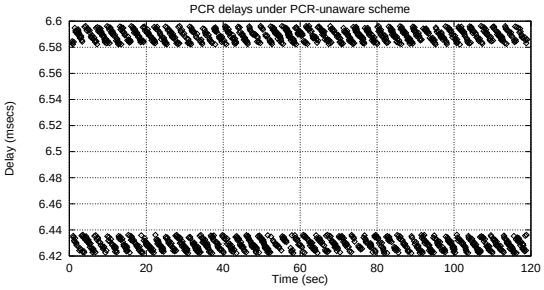

Figure 3.8: Delays experienced by MPEG transport packets containing PCRs under PCR-unaware scheme with no cross-traffic.

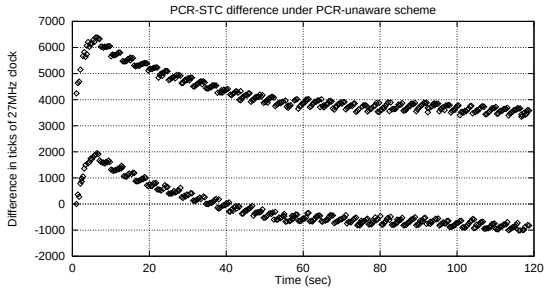

Figure 3.9: Phase difference due to packetization jitter with no cross-traffic.

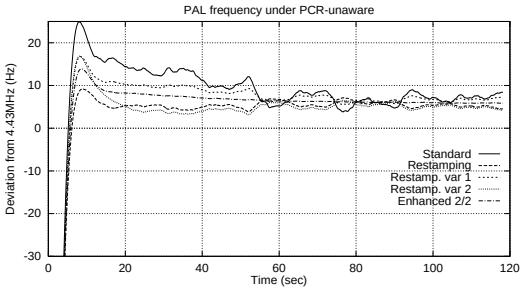

Figure 3.10: PAL color sub-carrier generation frequency under PCR-unaware scheme with no cross-traffic.

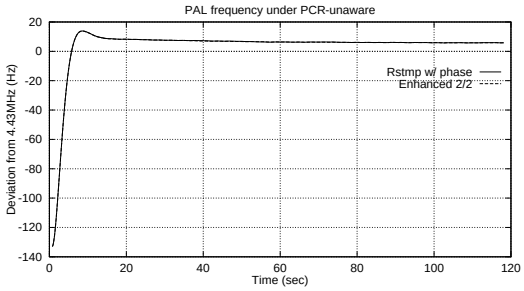

Figure 3.11: Comparison of PAL color sub-carrier generation frequencies with no cross-traffic under PCR-unaware scheme between restamping method that takes phase difference into consideration and Enhanced 2/2 scheme.

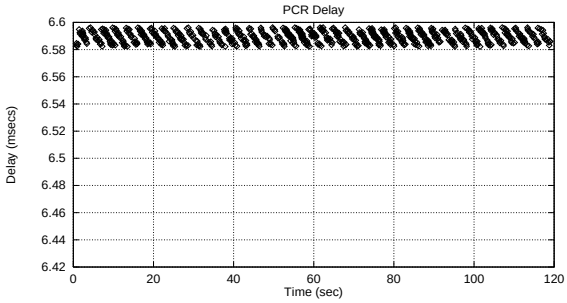

Figure 3.12: Delays experienced by MPEG transport packets containing PCRs that fall in the high-gain zone (clocking delay zone) of the restamping approach.

3.4 Experiment 2

This experiment was performed in order to test how the algorithm behaves under medium-load conditions. In this experiment, thirty ON-OFF sources from each cross-traffic node were multiplexed with the MPEG-2 stream at each network link. The overall load on any downstream output link of the ATM switches was increased to 70% resulting in 10 Mbps aggregate rate of the ON-OFF sources per hop, or 0.334 Mbps load per source.

The delays experienced by transport packets containing PCRs in the FIFO case are plotted in Figures 3.13 and 3.14, respectively, for the PCR-unaware and PCR-aware cases. The maximum delay is close to 8.2 msecs for the PCR-unaware case and approximately 8 ms for the PCR-aware case. Thus, the maximum jitter at the receiver for transport packets containing PCRs is 1.8 ms and 1.6 ms for the PCR-unaware and PCR-aware cases, respectively. In both cases, the delays are spread out for FIFO and the majority of the values fall between 6.4 and 7 ms. Even though the clock requirements, in terms of PAL frequency variation, are within the specifications (Figures 3.15 and 3.16), it is not the case for the clock drift since we must average it over a window of 80 seconds in order to meet the standard, as shown in Figure 3.17. The best of the restamping methods was the normal two-zone version which gave acceptable quality of the recovered clock, ensuring that the clock drift specifications are not violated even when the clock drift is averaged over small time intervals (Figure 3.18). The effect of jitter on the clock recovery with FIFO scheduling is more noticeable in the PCR-aware case (Figure 3.16). As described in Section 2 the clocking delay zone may not be stable initially, which is true in this experiment for the restamping approach as shown in Figure 3.19. Eventually, the clocking delay zone stabilizes around the average delay.

Use of FFQ scheduling discipline in the network switches yielded very good results in controlling both the network-induced jitter and the quality of the recovered clock, with the restamping method minimizing the packetization jitter (Figure 3.20). In the PCR-aware case, the results of the standard case were almost identical with those from the restamping method, since the latter never entered the second zone utilizing low gain (Figure 3.21).

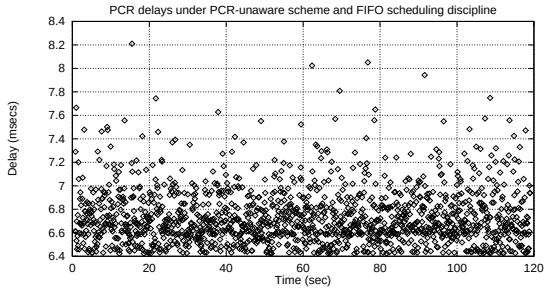

Figure 3.13: Delays experienced in MPEG transport packets containing PCRs with 70% load under FIFO and using PCR-unaware scheme.

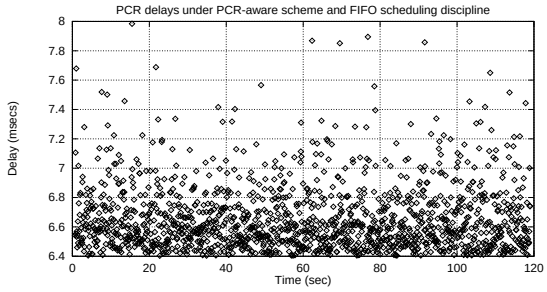

Figure 3.14: Delays experienced in MPEG transport packets containing PCRs with 70% load under FIFO and using PCR-aware scheme.

3.5 Experiment 3

The objective of this experiment is to study the performance of the restamping algorithm in a heavily loaded network. As in the previous experiment, thirty ON-OFF sources from each cross-traffic node were multiplexed with the end-to-end MPEG-2 stream. The overall load on each

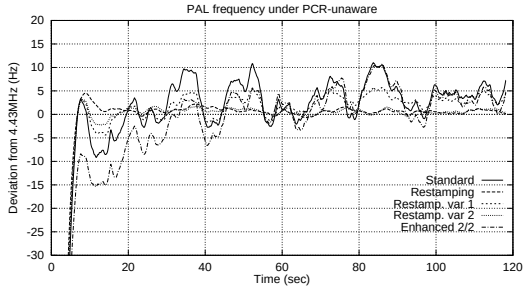

Figure 3.15: PAL color sub-carrier generation frequency with 70% load under FIFO and using PCR-unaware scheme.

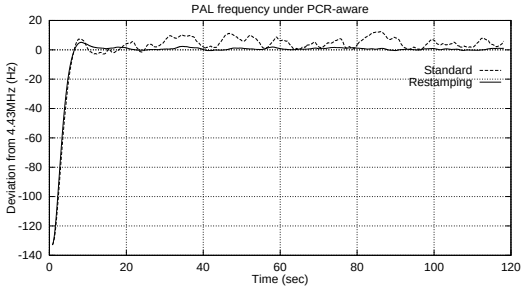

Figure 3.16: PAL color sub-carrier generation frequency with 70% load under FIFO and using PCR-aware scheme.

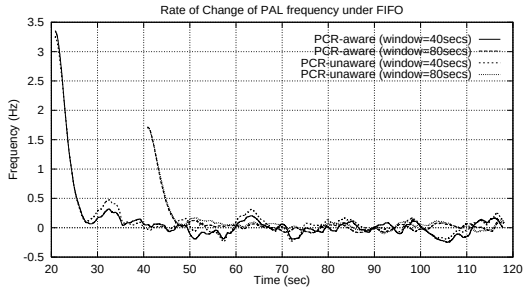

Figure 3.17: Rate of change of PAL color sub-carrier generation frequency under FIFO scheduling discipline.

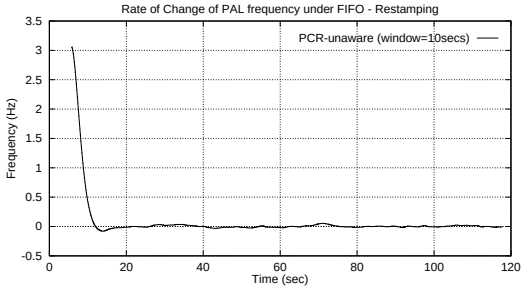

Figure 3.18: Rate of change of PAL color sub-carrier generation frequency using restamping under FIFO scheduling discipline and PCR-unaware scheme (averaged over a 10 sec window).

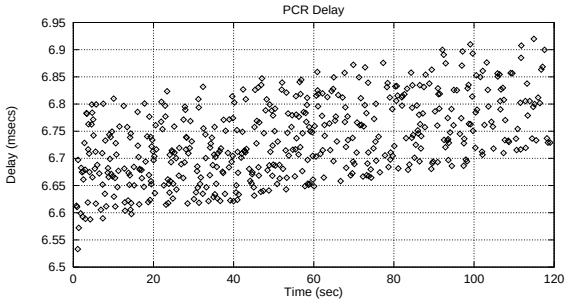

Figure 3.19: Delays experienced in MPEG transport packets containing PCRs that fall in the high-gain zone (clocking delay zone) of the restamping approach.

downstream output link of the ATM switches was increased to 95% yielding an aggregate rate of 18 Mbps for the ON-OFF cross-traffic sources at each hop. Besides the standard, the Enhanced 2/2 and the restamping approaches, a traditional dejittering approach is also tested. Two different sizes for the dejittering buffer are used, giving the flexibility to absorb 5 or 10 ms of jitter respectively, typical values used in actual products.

The maximum delay experienced by transport packets containing PCRs under FIFO scheduling

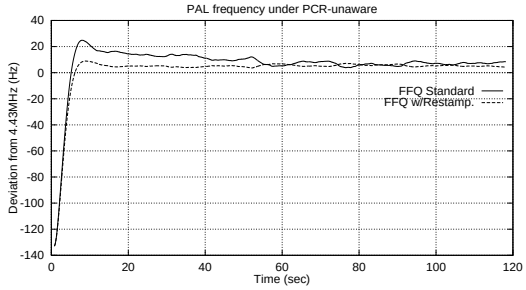

Figure 3.20: PAL color sub-carrier generation frequency with 70% load under FFQ and using PCR-unaware scheme.

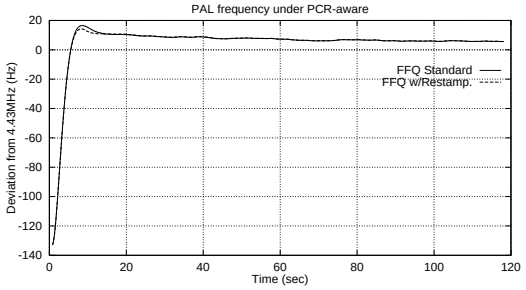

Figure 3.21: PAL color sub-carrier generation frequency with 70% load under FFQ and using PCR-aware scheme.

discipline is approximately 28 msecs for both PCR-aware and PCR-unaware schemes (PCR-unaware case is shown in Figure 3.22). The maximum jitter of the transport packets containing PCRs is 21.6 ms. In both cases (PCR-aware and PCR-unaware), the delays are spread out for FIFO and the majority of them fall between 6.4 and 15 ms. The packetization scheme in this case does not make any difference and the quality degradation is indistinguishable in both cases. The use of FIFO scheduling discipline results in extremely poor quality of the recovered clock in all but the restamping methods (Figure 3.23). The heavy jitter that is present in the FIFO case resulted in large phase differences (Figures 3.24 and 3.25) which are responsible for the poor quality of the recovered clock. The dejittering methods degrade the quality of the clock since the dejittering buffer under/overflows approximately 17 times even in the 10 ms case (Figure 3.26). This is because of the large amount of jitter experienced by the MPEG-2 transport packets in the network. The restamping methods exhibit good performance since they compress the incoming error terms resulting in a recovered clock with minor disturbances. Although the quality of the recovered clock with the proposed heuristic seems to be almost perfect, there is a slight discrepancy between the frequency of the acquired clock from its ideal value. This is because the enhanced PLL acquires the clock at a slow pace (depending on g_2) since the majority of error terms are high amplitude error terms. When the PLL becomes locked the clocking delay zone makes it immune to high-amplitude noise, yet responsive to small frequency changes. This behavior will be further demonstrated in Experiment 5. As illustrated in Figures 3.27 and 3.28, the high amplitude error terms are attenuated and fall into the ± 3000 ticks region. The high amplitude error terms will drive the enhanced PLL to the correct frequency. The locking time of the modified PLL is shorter than that of a standard PLL with low gain since, in the former, all the error terms that fall in the high-gain region will facilitate the loop acquisition process. It should be noted that, although the quality of the clock was unacceptable with the standard PLL, the MPEG-2 system buffer dynamics were almost unaffected during the experiment (Figure 3.29) and the maximum occurred at the same point as in the case with no cross-traffic. This behavior is consistent with that observed in [19].

Use of FFQ scheduling discipline in the switches yielded very good results in the quality of the recovered clock for both packetization schemes (Figures 3.30 and 3.31). Although the quality of the recovered clock in the standard method was acceptable, the restamping approach improved it, by reducing the packetization jitter (Figure 3.30).

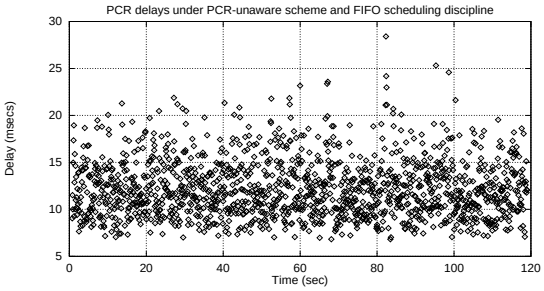

Figure 3.22: Delays experienced by MPEG transport packets containing PCRs with 95% load under FIFO using PCR-unaware scheme and without any dejittering.

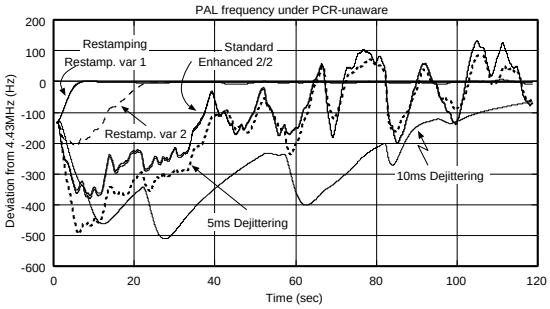

Figure 3.23: PAL color sub-carrier generation frequency with 95% load under FIFO and using PCR-unaware scheme.

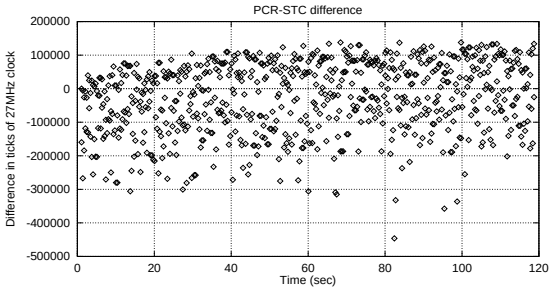

Figure 3.24: Phase Difference with 95% load under FIFO and using PCR-unaware scheme.

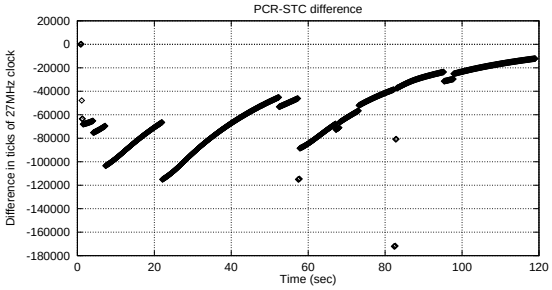

Figure 3.25: Phase Difference with 95% load under FIFO and PCR-unaware scheme using 10 ms dejittering.

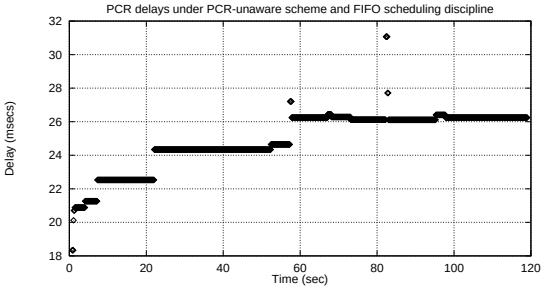

Figure 3.26: Delays experienced by MPEG transport packets containing PCRs with 95% load under FIFO using PCR-unaware scheme and with 10 ms dejittering.

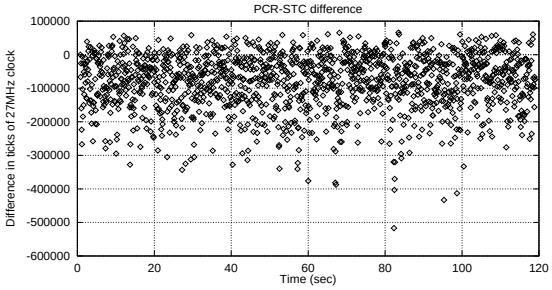

Figure 3.27: Phase Difference before restamping with 95% load under FIFO and using PCR-unaware scheme.

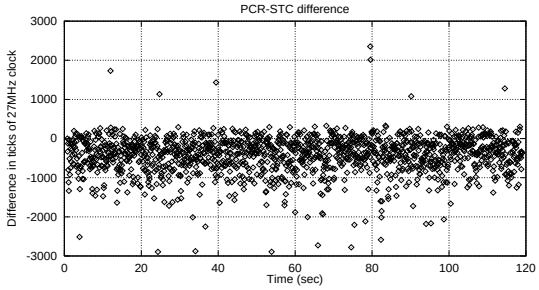

Figure 3.28: Phase Difference after restamping with 95% load under FIFO and using PCR-unaware scheme.

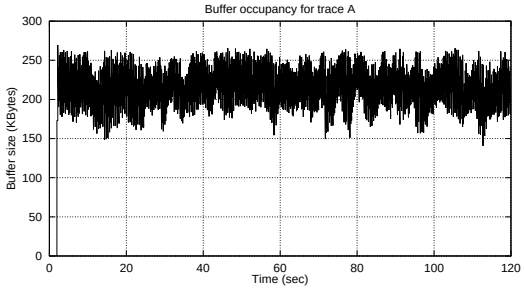

Figure 3.29: Buffer occupancy with 95% load under FIFO and PCR-unaware scheme using standard PLL.

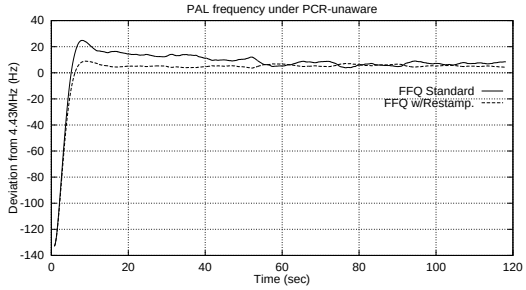

Figure 3.30: PAL color sub-carrier generation frequency with 95% load under FFQ and using PCR-unaware scheme.

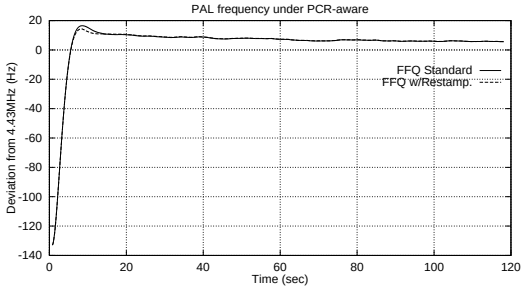

Figure 3.31: PAL color sub-carrier generation frequency with 95% load under FIFO and using PCR-aware scheme.

3.6 Experiment 4

This experiment was performed in order to test the performance of the restamping heuristic in a heavily loaded network with high-speed links. The experiment was done with trace B, which contains NTSC video as well as MPEG-2 audio elementary streams. The actual scenario is similar to that of Experiment 4 but with 150 cross-traffic sources. However, all the links were changed to 155 Mbps and the overall offered load was set to 90%. All the simulations were performed with the PCR-unaware scheme, since the packetization delay for this stream is negligible.

The choice of the scheduling algorithm used in the network switches has a significant influence on the jitter experienced by the video stream in this case. The cumulative distribution functions (CDFs) for transport packets containing PCR values obtained (Figure 3.32), indicate excessive jitter in the case of FIFO (maximum delay observed is approximately 87 msecs as shown in Figure 3.33). In contrast, the FFQ scheduling discipline offers good isolation and low levels of jitter (Figures 3.34 and 3.35). The quality of the reconstructed clock for the standard PLL is unacceptable with FIFO scheduling (Figure 3.36), but use of the FFQ algorithm for scheduling resulted in satisfactory quality (Figure 3.37). The restamping approach, on the other hand yielded a stable clock even with the large amount of jitter introduced by FIFO scheduling.

The maximum instantaneous phase difference was observed in the FIFO case, approximately 1400000 ticks (or almost two frames in NTSC format) as shown in Figure 3.38. The reason for the satisfactory performance of the restamping approach in this experiment comes from the fact that the low gain is effective in reducing the effect of large variations in delay (Figure 3.39). The modified PLL in the restamping approach may take a long time to lock, but is more immune to noise and is able to lock to the correct frequency in a smoother manner. As in the previous cases, depending on g_2, a large time interval may be needed before the clocking delay zone is stabilized which will drive the PLL to remain close to the correct frequency in the case of excessive jitter. This process may be extremely long if the system is designed to handle excessive jitter. However, when the restamping method is operating in the correct clocking delay zone, the acquisition time of the correct frequency is much faster. This was evident in the first experiment in which the amount of jitter was small.

Although the phase difference and its variation could affect the occupancy at the system decoder buffer, it did not occur in this experiment, verifying that the impact of excessive jitter is primarily on the clock recovery process rather than on the system decoder buffer. Even in the FIFO case, the system decoder buffer does not underflow (Figure 3.40) and its occupancy changes only slightly compared to the case when all the cross-traffic sources were turned off (Figure 3.41). This is consistent with the behavior in the previous experiments.

3.7 Experiment 5

This last experiment was performed in order to test the tracking performance of the algorithm when there are periods of heavy load. This experiment is similar to Experiment 4, with the difference that cross-traffic is present only for a time interval of 30 seconds (from 40 to 70 seconds). The idea behind the experiment is to have the PLL locked at the correct frequency before any load is applied. Therefore, the phase difference should be close to zero at the time the cross-traffic sources are turned on. FIFO scheduling discipline was used so that the delays are affected significantly by the presence of cross traffic.

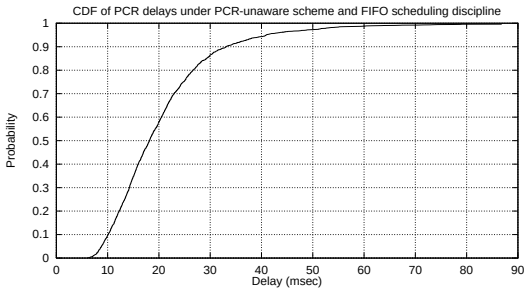

Figure 3.32: CDF for PCR delays with 90% load under FIFO and using PCR-unaware scheme.

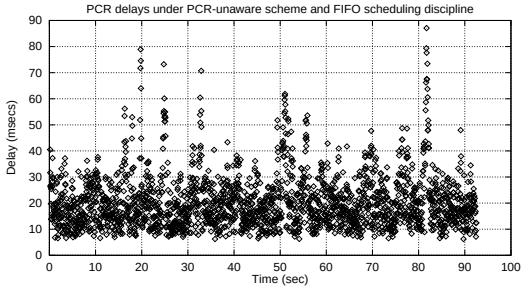

Figure 3.33: Delays experienced by MPEG transport packets containing PCRs with 90% load under FIFO and using PCR-unaware scheme.

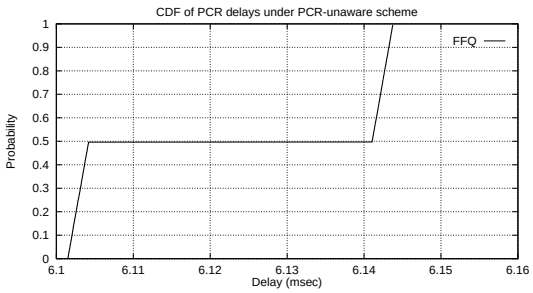

Figure 3.34: CDF for PCR delays with 90% load under FFQ and using PCR-unaware scheme.

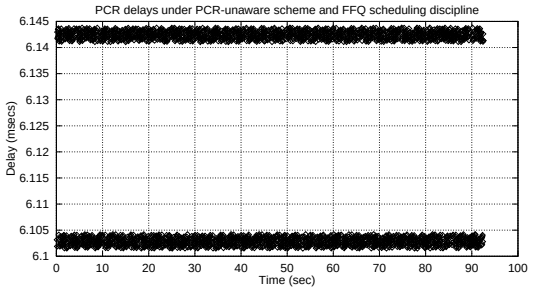

Figure 3.35: Delays experienced by MPEG transport packets containing PCRs with 90% load under FFQ and using PCR-unaware scheme.

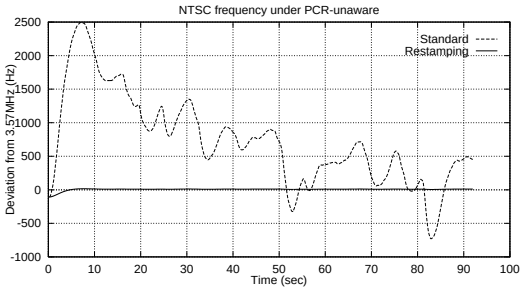

Figure 3.36: NTSC color sub-carrier generation frequency with 90% load under FIFO and using PCR-unaware scheme.

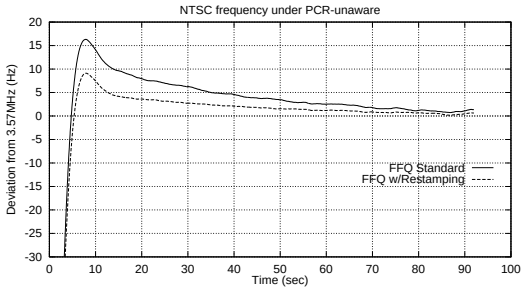

Figure 3.37: NTSC color sub-carrier generation frequency with 90% load under FFQ and using PCR-unaware scheme.

The transport packets carrying PCR values undergo excessive delays when cross-traffic is present, reaching a maximum value of 17.7 msecs (Figure 3.42). This results in severe quality degradation of the recovered clock with a standard PLL, since NTSC specifications could not be met (Figure 3.43). The same applies to the dejittering method since in that case the buffer under/overflows 4 times (Figure 3.44). The restamping algorithm makes the PLL operate in the high-gain zone before the cross-traffic becomes active. At the time when the cross-traffic is turned on, the algorithm enters the low-gain zone and, by compressing the signal, minimizes the effects of high-amplitude

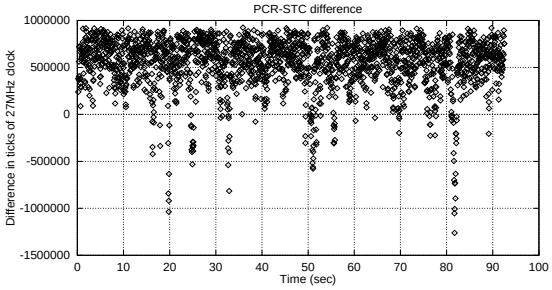

Figure 3.38: Phase Difference before restamping with 90% load under FIFO and using PCR-unaware scheme.

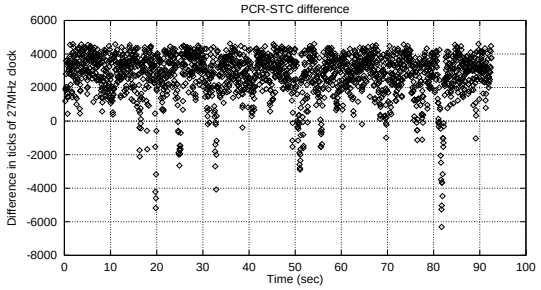

Figure 3.39: Phase Difference after restamping with 90% load under FIFO and using PCR-unaware scheme.

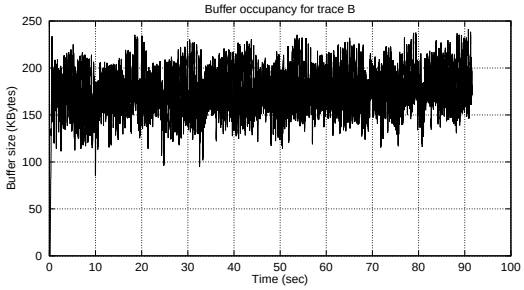

Figure 3.40: Buffer occupancy with 90% load under FIFO and PCR-unaware scheme using standard PLL.

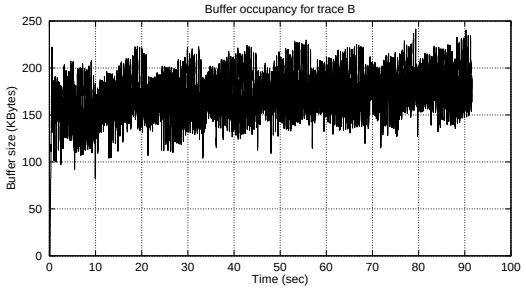

Figure 3.41: Buffer occupancy with no cross-traffic under FIFO and PCR-unaware scheme using standard PLL.

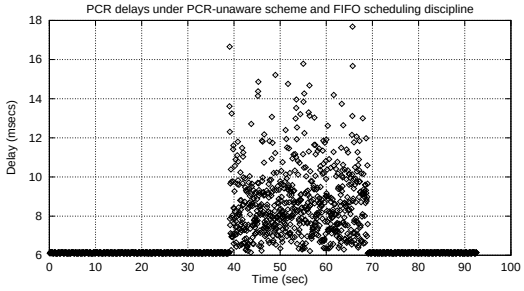

Figure 3.42: Delays experienced in MPEG transport packets containing PCRs with variable load under FIFO.

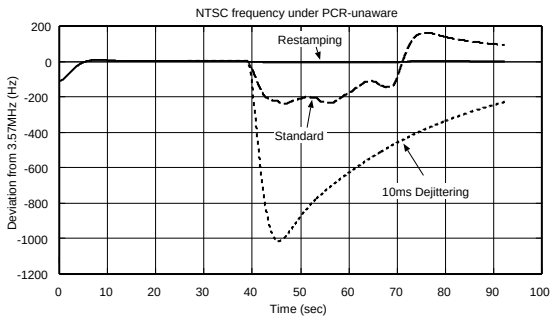

Figure 3.43: NTSC color sub-carrier generation frequency with variable load under PCR-unaware scheme and FIFO.

noise (Figures 3.45 and 3.46). Therefore, the quality of the reconstructed clock is only slightly affected and remains within the NTSC specifications (±10Hz), as depicted in Figure 3.47. Finally, when background load becomes zero again, the algorithm recalibrates itself to acquire the correct frequency by entering the high-gain zone again.

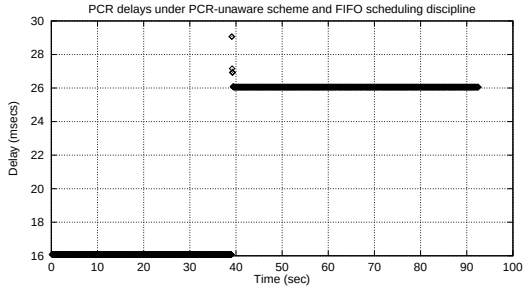

Figure 3.44: Delays experienced in MPEG transport packets containing PCRs with variable load under FIFO using 10 ms dejittering buffer.

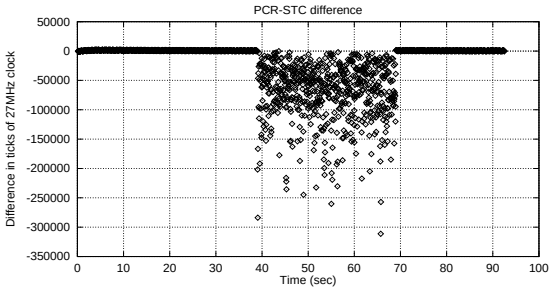

Figure 3.45: Phase Difference before restamping with variable load under PCR-unaware scheme and FIFO.

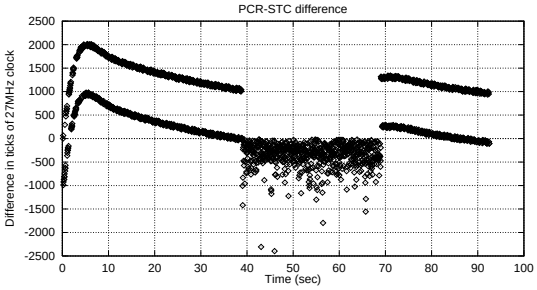

Figure 3.46: Phase Difference after restamping with variable load under PCR-unaware scheme and FIFO.

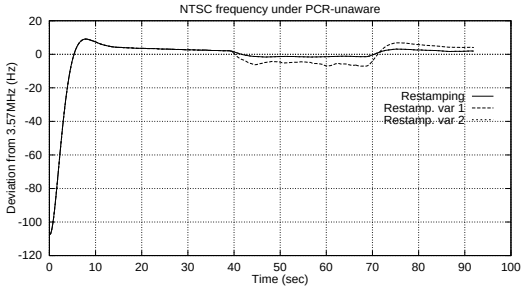

Figure 3.47: NTSC color sub-carrier generation frequency for variable restamping approaches with variable load under PCR-unaware scheme and FIFO.

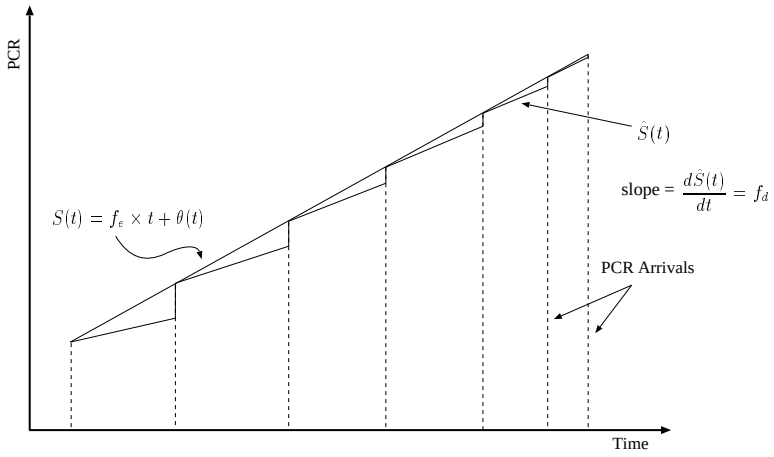

Figure 4.1: Actual PCR function and PCR function used in the analysis.

4 Analysis

In this section, we provide an analysis of the dynamics of the restamping approach. In order to analyze the behavior of the control loop, we need to derive its steady-state phase error, stability, tracking control and loop acquisition behavior. We follow an approach similar to that in [11] for traditional PLLs. The main difference in our analysis is the nature of the input signal. In our case, the input signal is a linear function as shown in Figure 4.1, whereas in the case of traditional PLLs, the input signal is considered to be a sinusoidal function. Before continuing with our analysis, we define some notations and assumptions used.

Although the PCRs arrive at discrete points in the time scale, we can assume that the incoming PCRs form a continuous-time function $S(t)$ that is updated at the instants when a new PCR value is received. We can model the incoming clock with the function

$$S(t) = f_e \times t + \theta(t), \tag{4.1}$$

where f_e is the frequency of the encoder sending the MPEG-2 stream and $\theta(t)$ is the incoming clock's phase relative to a designated time origin. As indicated in Figure 4.1 there is a small discrepancy when modeling the incoming clock function. The actual incoming clock function $\hat{S}(t)$ is a function with discontinuities at the time instants at which PCR values are received, and slope equal to f_d for each of its segments, with f_d being the running frequency of the decoder. For the sake of convenience, however, we use $S(t)$ in place of the actual PCR function $\hat{S}(t)$, since the interval between any two consecutive PCR arrivals is bounded by the MPEG-2 standard and equal to 0.1 second, which makes the two functions to be very close.

Analogously, the system time clock (STC) corresponds to the function

$$R(t) = f_d \times t + \hat{\theta}(t), \tag{4.2}$$

where $\hat{\theta}(t)$ is the incoming clock's phase relative to a designated time origin. Therefore, the error term after the subtractor is given by

$$e(t) = S(t) - R(t) = (f_e - f_d)t + \left(\theta(t) - \hat{\theta}(t)\right). \tag{4.3}$$

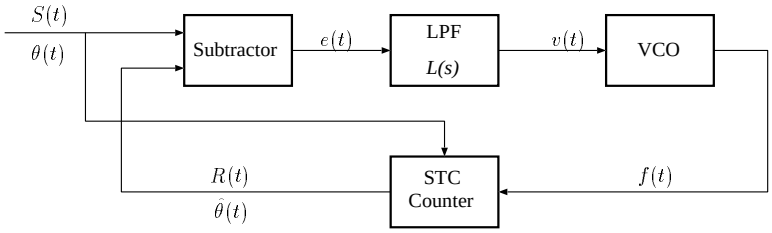

Figure 4.2: Equivalent Model of the PLL used.

Without loss of generality, we can assume that $f_e = f_d$. Let us denote this with f_o and insert any frequency difference in the phase terms. We can now work with $\theta(t)$ as being the input to our control system and with $\hat{\theta}(t)$ as being the output of the counter as shown in Figure 4.2. Thus Eq. 4.3 becomes

$$e(t) = \theta(t) - \hat{\theta}(t). \tag{4.4}$$

The frequency $f(t)$ of the VCO is a function of $v(t)$. The nominal value of this frequency is assumed to be f_0 and when $v(t)$ is applied, it becomes $f_0 + Kv(t)$ where K is the gain factor of the VCO. It is obvious that

$$\frac{dR(t)}{dt} = f_0 + Kv(t). \tag{4.5}$$

By definition,

$$R(t) = f_0 t + \hat{\theta}(t). \tag{4.6}$$

Combining Eq. 4.5 and 4.6 we get

$$\frac{d\hat{\theta}(t)}{dt} = Kv(t). \tag{4.7}$$

From Eq. 4.4 and 4.7 we obtain

$$\begin{aligned} \frac{de(t)}{dt} &= \frac{d\theta(t)}{dt} - Kv(t) \\ &= \frac{d\theta(t)}{dt} - K\int_0^\infty l(t-u)e(u)du. \end{aligned} \tag{4.8}$$

We assume that the Laplace transformations of $e(t)$ and $\theta(t)$ exist and they are $E(s)$ and $\Theta(s)$ respectively, and $L(s)$ is Low-Pass filter's transfer function. Eq. 4.8, when transformed to the Laplace domain, becomes

$$sE(s) = s\Theta(s) - KL(s)E(s). \tag{4.9}$$

We assume that $\hat{\theta}(t)$ has a Laplace transform. Using $E(s) = \Theta(s) - \hat{\Theta}(s)$, where $\hat{\Theta}(s)$ is the Laplace transform of $\hat{\theta}(t)$ and, Eq. 4.9 we can now derive the transfer function $H(s)$ of the closed-loop:

$$H(s) = \frac{\hat{\Theta}(s)}{\Theta(s)} = \frac{KL(s)}{s + KL(s)}. \tag{4.10}$$

To obtain the steady-state phase error $e(t)$, we use the final value theorem of Laplace transformation:

$$\lim_{t\to\infty} e(t) = \lim_{s\to 0} sE(s). \tag{4.11}$$

Finally,

$$E(s) = \Theta(s) - \hat{\Theta}(s) = [1 - H(s)]\Theta(s) = \frac{1}{1 + [KL(s)/s]}\Theta(s). \tag{4.12}$$

The worst-case steady-state phase error ϕ of the enhanced PLL in the case of a frequency step input occurs when all the new incoming clock values produce error terms e that fall into the low-gain zone (with gain g_2). In that case, the PLL can be approximated by a traditional PLL that has a gain of g_2 and thus ϕ is given by the equation

$$\phi = \lim_{t\to\infty} e(t) = \lim_{s\to 0}\left\{ s\frac{\Delta f}{s^2}\frac{1}{1 + [g_2 L(s)/s]}\right\} = \lim_{s\to 0}\left\{\frac{\Delta f}{s + g_2 L(s)}\right\}, \tag{4.13}$$

where Δf is the frequency difference between the encoder and the decoder, $\frac{\Delta f}{s^2}$ is the Laplace transform of the frequency step and $L(s)$ is the Laplace transform of the LPF. It is clear that if the LPF does not have a pole at the origin $s = 0$, then a non-zero phase error may be present which depends not only on $F(0)$ but also on the downpressure factor g_2 in the worst case. Therefore, only in the case of a first order PLL and a frequency change at the input, the phase error implies a tradeoff for the selection of g_2. Low values give good tracking performance but high phase error in a worst-case situation. However, if the jitter is very low, the enhanced PLL produces the same phase error as a standard PLL without the restamping algorithm.

Considering stability, if the standard PLL is stable, i.e., has its poles in the left-hand plane, the enhanced PLL will also be stable since the only change is a variable gain factor in the loop which is always less than 1. Intuitively, the tracking performance of the enhanced PLL is better than that of a standard PLL. The reasoning behind this comes from the fact that the enhanced PLL compresses the error term e with a non-linear factor, thus reducing effects of high-amplitude noise (jitter).

In order to study the loop acquisition and find the settling time T_s, we must apply a unit-step function at the input of the system. The worst-case behavior in the enhanced loop occurs again when the unit step function produces a transition from a zone with high-gain (g_1) to the zone with low-gain (g_2). According to [11], the settling time T_s is approximately equal to $3/\omega_c$, where ω_c is the crossover frequency of the system. T_s is defined as the time taken by the phase step-response error to settle within $\pm 5\%$ of its final value. For a second-order PLL with active LPF $\omega_c \approx 2\zeta\omega_n$, where $\omega_n = \left(\frac{g_2}{T_1}\right)^{1/2}$, and $\zeta = \frac{T_2\omega_n}{2}$. Note that $L(s) = -\frac{1+sT_2}{sT_1}$ is the transfer function of a first-order active LPF. As shown above, T_s depends heavily on g_2 in the worst case and in that case results in slow loop acquisition. As noted in Section 2 in order to improve the acquisition time a technique similar to the one presented in [17] can be used in which g_2 is time-varying during loop acquisition, starting from high values and ending with its final one.

5 Conclusions

In this paper, we studied the clock recovery problem for MPEG-2 Systems Layer streams. A new architecture for decoder design is proposed which is based on a jitter estimator capable of performing restamping on the incoming packets containing clock values in order to minimize the effects of jitter

from sources other than the frequency difference between sender and receiver. We also discussed a practical implementation of this architecture in which a simple heuristic was added to a standard PLL that performs restamping according to the jitter estimated by the PLL. Experimental and analytical results presented show that the use of the enhanced PLL improves performance even under excessive load conditions making it immune to noise (jitter) and responsive to frequency changes.

A potential application of the restamping method is in set-top boxes. Set-top boxes are used for distribution of digital video to the end-user. Their functions include decoding of the MPEG-2 stream, providing support for user control, and interfacing to the video distribution network. The reduction in the memory requirements afforded by the restamping scheme, and its simplicity, are both desirable attributes in the design of a set-top decoder. In addition, the restamping technique can be combined with standard dejittering approaches or with the Enhanced 2/2 scheme [1] in order to improve their performance.

A number of issues still remain to be investigated: Evaluation of the tradeoff of acquisition speed versus accuracy for the selection of the downpressure factors is one of them. Although the values used in the experiments provided satisfactory performance, it is important to study the behavior of the scheme in other environments. Also, methods of dynamically changing $\mathcal{T}_f$ to make it more responsive and immune without sacrificing robustness is another critical aspect that needs to be addressed in more detail. Finally, the effect of autocorrelation of the jitter of the arriving PCR stream on the clock recovery process needs special attention since the multiplexing of several traffic sources in current packet-switched networks may introduce such correlation.

6 Acknowledgments

The authors would like to thank Paul Pitcl and Ari Jahanian of LSI Logic Corporation, Fre Jorritsma of Philips Research Laboratories, and Bill Helms and Ji Zhang of Divicom Corporation for providing the MPEG-2 Systems Layer traces used in the simulations.

References

[1] I. F. Akyildiz, S. Hrastr, H. Uzunalioglu, and W. Yen. Comparison and evaluation of packing schemes for MPEG-2 over ATM using AAL5. In *Proceeding of ICC '96*, volume 3, pages 1411–1415, June 1996.

[2] G. F. Andreotti, G. Michieletto, L. Mori, and A. Profumo. Clock recovery and reconstruction of PAL pictures for MPEG coded streams transported over ATM networks. *IEEE Transactions on Circuits and Systems for Video Technology*, 5(6):508–514, December 1995.

[3] M. De Prycker. *Asynchronous Transfer Mode : Solution for Broadband ISDN*. Ellis Horwood, second edition, 1993.

[4] S. Dixit and P. Skelly. MPEG-2 over ATM for video dial tone networks: issues and strategies. *IEEE Network*, 9(5):30–40, September–October 1995.

[5] D. Fibush. *Subcarrier Frequency, Drift and Jitter in NTSC Systems*. ATM Forum, July 1994. ATM94-0722.

[6] S. J. Golestani. A self-clocked fair queueing scheme for broadband applications. In *Proceedings of IEEE INFOCOM '94*, volume 2, pages 643–646, June 1994.

[7] P. Hodgins and E. Itakura. *The Issues of Transportation of MPEG over ATM.* ATM Forum, July 1994. ATM94-0570.

[8] P. Hodgins and E. Itakura. *VBR MPEG-2 over AAL5.* ATM Forum, December 1994. ATM94-1052.

[9] International Organization for Standardization. *Information Technology — Generic Coding of Moving Pictures and Associated Audio: Systems, Recommendation H.222.0, ISO/IEC 13818-1*, draft international standard edition, November 1994.

[10] Y. Kaiser. Synchronization and dejittering of a TV decoder in ATM networks. In *Proceedings of PV '93*, volume 1, 1993.

[11] H. Meyr and G. Ascheid. *Synchronization in Digital Communications*, volume 1 of *Wiley Series in Telecommunications.* John Wiley & Sons, 1990.

[12] M. Nilsson. MPEG-2 over ATM. In *IEE Colloqium on "MPEG-2 – what it is and what it isn't"*, number 1995/012. The Institute of Electrical Engineers, January 1995.

[13] M. Perkins and P. Skelly. *A Hardware MPEG Clock Recovery Experiment in the Presence of ATM Jitter.* ATM Forum, May 1994. ATM94-0434.

[14] P. V. Rangan, S. S. Kumar, and S. Rajan. Continuity and synchronization in MPEG. *IEEE Journal on Selected Areas in Communications*, 14(1):52–60, January 1996.

[15] P. A. Sarginson. MPEG-2 – a tutorial introduction to the systems layer. In *IEE Colloqium on "MPEG-2 – what it is and what it isn't"*, number 1995/012. The Institute of Electrical Engineers, January 1995.

[16] M. Schwartz and D. Beaumont. *Quality of Service Requirements for Audio-Visual Multimedia Services.* ATM Forum, July 1994. ATM94-0640.

[17] R. P Singh, Sang-Hoon Lee, and Chong-Kwoon Kim. Jitter and clock recovery for periodic traffic in broadband packet networks. *IEEE Transactions on Communications*, 42(5):2189–2196, May 1994.

[18] D. Stiliadis and A. Varma. Efficient fair-queueing algorithms for packet-switched networks. *IEEE/ACM Transactions on Networking*, April 1998.

[19] C. Tryfonas. MPEG-2 transport over ATM networks. Master's thesis, University of California at Santa Cruz, September 1996. Available at `http://www.cse.ucsc.edu/research/hsnlab`.

[20] S. Varma. MPEG-2 over ATM: System design issues. LSI Logic Corp.

[21] L. Zhang. VirtualClock : A new traffic control algorithm for packet-switched networks. *ACM Transactions on Computer Systems*, 9(2):101–124, May 1991.