ITU-T Telecommunications Sect **Document AVC-647**
Study Group 15 **13 July 94**
Experts Group for Video Coding and Systems
in ATM and Other Network Environments

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION
ISO/IEC JTC1/SC29/WG11
CODING OF MOVING PICTURES AND ASSOCIATED AUDIO

ISO/IEC JTC1/SC29/WG11
MPEG 94/

Source Barry Haskell & Amy Reibman, AT&T
Title: Faster Timing Recovery in the Presence of Cell Delay Jitter
Purpose: Information

Faster Timing Recover in the Presence of Cell Delay Jitte

Barry Haskell and Amy Reibman

1. Introduction

In a previous contribution[1] we described a method of timing recovery and adaptive delay for dealing with Cell delay jitter. Here we discribe methods for significantly speeding up the timing acquisition.

In an MPEG multiplexed bit stream, the encoder periodically sends samples of its time clock, which are called Clock References. We denote these Encoder Clock References by PCR. Typically, the PCRs are sent at the rate of a few per second.

Using the methods of [1], recovery of the time clock at the decoder relies on averaging the jitter effects over many received PCR values. In cases of extreme jitter, thousands of PCR values may be needed before stable operation is achieved, which may require dozens of seconds[2]. In many applications this is unacceptable.

Here we describe a technique for synthesizing additional PCRs between the values that are actually transmitted. It relies on the fact that MPEG multiplexed bit streams are transmitted at a known data rate, which is specified in the bit stream. The data rate can change by transmitting a new specification. However, between specifications the data rate is constant.

When transmission is on ATM facilities, the PCR jitter may be made worse by the compaction of data into ATM Cells. This results in a time shift of the PCR transmission that depends on the position of the PCR in the Cell.

We also describe a method of compensating for this shift.

2. Overview

A System Decoder receives the input data stream and routes the various components to the apppropriate modules. If clock references or data rates are received, it sends them to a Decoder Time Clock Error Estimator.

We denote the Cell transmission rate as Cell_Rate. This can be computed from the value program_mux_rate transmitted in a pack header in either an MPEG-2 Program stream or a PES packet, or it can be computed from the value piecewise_rate transmitted in the adaptation field extension of an MPEG-2 Transport stream packet. These values (program_mux_rate and piecewise_rate) describe the data rate in bytes per second, and are converted into Cells per second by the System Decoder

We also define Cell_period = 1/Cell_Rate as the time between transmitted Cells. At the receiver the intercell time will usually vary because of the aforementioned delay jitter.

The System Decoder also signals the instants of Cell start and Cell end to the Decoder Time Clock Error Estimator. These are defined as the exact times just after the first and last payload bytes are received, respectively.

For each Cell, Decoder Time Clock Error Estimator generates a Decoder Time Clock Error, denoted by TC_Error, which is sent to a Filter of Decoder Time Clock Generator for phase lock loop processing as described in [1]. Since TC_Error is generated for every cell instead of only for each PCR, timing recovery is significantly speeded up.

3. Decoder Time Clock Error Estimator Startup

The Decoder Time Clock Error Estimator startup operation waits for the first byte of the next cell to be received. It then starts counting bytes. If a PCR is received, the number of bytes from the beginning of the cell up to and including the last byte of the PCR is saved as Bytes_To_PCR. Cell_rate is then received. If PCR and Cell_rate are not received, we await another cell and repeat the above process.

We assume PCR represents the time just after the next to the last byte of PCR is transmitted, assuming a constant transmission bit-rate. However, the transmission bit-rate is not instantaneously constant with ATM. Instead, cells are sent in bursts at a channel rate much higher than the average encoder output bit-rate, which distorts the timing of the PCRs.

Of course, received cells could be stored in a FIFO and read out at a constant rate in an attempt to restore a (perhaps fictitious) constant encoder bit-rate. However, this requires extra memory and also has

problems if the bit-rate changes during transmission.

What we do iis compute an Adjusted PCR that represents the time just after the first byte of the cell is transmitted. We denote this adjusted clock reference by Adj_CR, and calculate its value by

Adj_CR = PCR - (Bytes_To_PCR - 2)*Cell_period/Bytes_In_Cell

Following this, we calculate the estimated clock reference for the next cell as

Est_CR = Adj_CR + Cell_period

We then wait for the start of the next cell, whereupon we output the Decoder Time Clock startup value Dec_TC = Est_CR.

4 Decoder Time Clock Error Estimator Steady State Operation

Decoder Time Clock Error Estimator steady state operation first waits for the start of the next cell, after which it saves the current value of the Decoder Time Clock in parameter Saved_TC.

Following this, we receive any possible PCR or Cell_rate values and wait for the last byte of the cell. Upon its arrival, we check to see if a PCR was received.

If it was not, we increment Est_CR by the Cell_period so that it corresponds to the beginning of the current cell.

If PCR was received, we calculate the adjusted Clock Reference Adj_CR in the same way as above and set Est_CR = Adj_CR.

Following this we output the estimated start-of-cell Decoder Time Clock Error, which is given by

TC_Error = Est_CR - Saved_TC

An enable signal is also produced to tell the phase lock loop that input data is valid.

5. Departures and Enhancements

The descriptions so far assume no cell losses. In practice, there will be detectable cell losses, and in such cases simple modifications can minimize the effects. The simplest situation is when cell losses are detectable in the cell header. In this case, we should increment Est_CR by the Cell_period for each lost cell in order to maintain correct timing estimation.

The description so far assumes PCR and Cell_rate occur in the same cell. However, with MPEG2 Program Streams and all MPEG1 Streams such is not necessarily the case. The simplest solution is probably to disable the output of TC_Error if PCR and Cell_rate are received in different cells and not resume until a new PCR with known Cell_rate is received. However, more elaborate schemes can also be devised.

The description so far assumes the transmission rate, i.e., the

Cell_period, is known exactly. In fact, this is often the case. More specifically, the Cell_period is often an exact integer number of Encode Time Clock ticks. For example, in MPEG2 the nominal Time Clock frequency is 27MHz. With a transmission rate of 4Mbs and Bytes_In_Cell = 47, the number of ticks per cell is given by

Nc = 27Mhz * 8bits/byte * 47bytes/cell / 4Mbs = 2538 ticks/cell.

In general if R is the transmission rate in bits/second, the number of clock ticks per cell is

Nc = 216e6 * Bytes_In_Cell / R (rounded to integer, 0.5=1)

If the transmission rate is locked to the Encoder Time Clock, i.e., a cell is sent every Nc ticks, then timing recovery takes place with an effective known transmission rate defined, for example, in either program_mux_rate or piecewise_rate. If instead the encoder is transmitting at a slightly different rate, operation still proceeds as described. However, the recovered clock will have a small steady state phase error. In general, this phase error, or delay, will be small in comparison to the jitter delay computed in [1].

6. Conclusion

We have described a method for speeding up the timing recovery where bit streams contain time stamps transmitted at a low rate. The speedup is in direct proportion to the number of cells transmitted between PCR values.

7. References

[1] "Timing Recovery for Variable Bit-Rate Video on ATM Networks", Barry Haskell and Amy Reibman, MPEG92/396, AVC-315, Paris, France, July 3, 1992.

[2] "Simulation of Phase-locked Loop for Processing Jittered PCRs", Clive Holborow, MPEG94/071, AVC-628,