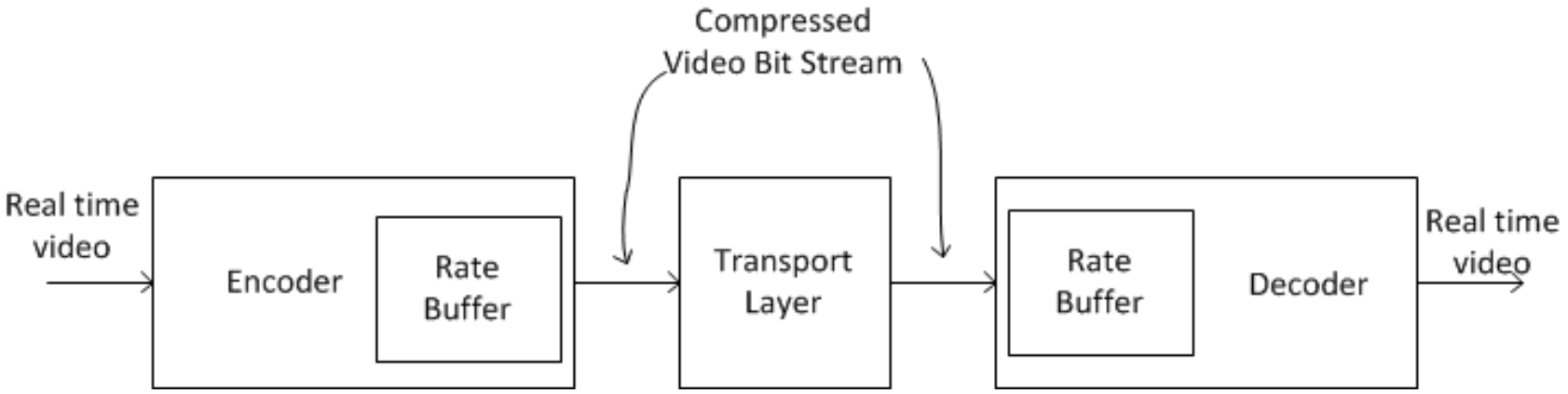
DSC Hypothetical Reference Decoder (HRD) Buffer Model Introduction

VESA DSC
Sandy MacInnis, Broadcom
Louie Kerofsky, Sharp Labs of America

DSC End-End System



- End-end system is real time. Pixels enter the encoder at pixel rate, and pixels leave the decoder at pixel rate
- Transport layer carries bits from encoder to decoder at pre-determined rate
- Encoders and decoders from different vendors must interoperate reliably for all input content
- One key issue is: How to ensure that the decoder's rate buffer never overflows nor underflows?
- Note: Some encoders / decoders may not have rate buffers. That's equivalent to having a zero-size rate buffer.

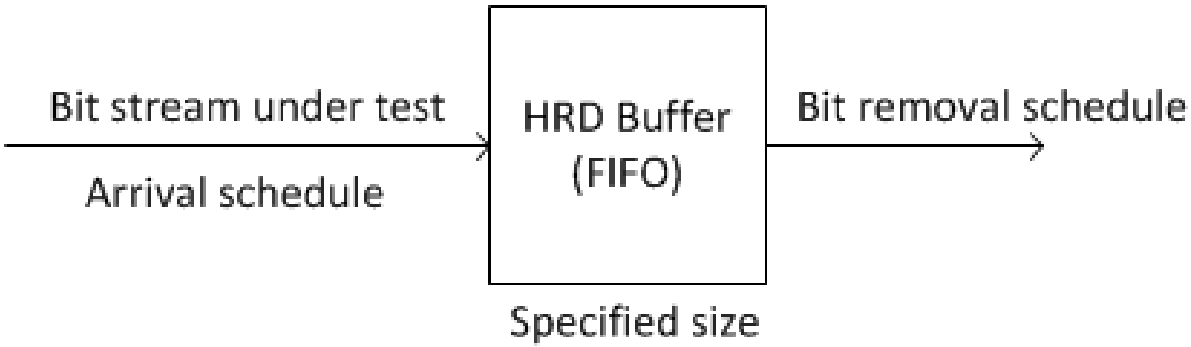
Why Have Rate Buffers?

- The number of bits used to code each pixel, or each small block of pixels (“block”), may vary, while the bit rate sent to the transport layer needs to have a fixed bits/pixel rate (CBR)
- Encoders with high coding efficiency tend to use variable numbers of bits per block
- Among other reasons, this gives the encoder the freedom to use more bits to code blocks that are require more bits to produce the desired quality, and fewer bits to code blocks that require fewer bits to produce the desired quality

Problem Statement

- When encoders use variable numbers of bits per block, what prevents them from using, for example, more bits than average for a very large number of consecutive blocks?
- When this happens, the decoder rate buffer may underflow. That is, the decoder has not received all the bits it needs to decode a block before the time when it needs to decode that block. The result is the decoder stalls, which is a system failure.
- Similarly, encoders may use fewer than average bits per block for a very large number of consecutive blocks.
- When this happens the decoder rate buffer may overflow. That is, the decoder receives more bits than it can store while it uses relatively few bits to decode the blocks in its buffer in real time. Bits are lost, causing errors in the decoder, which is a system failure.
- Underflow and overflow might be avoided by increasing the buffer size and increasing the initial start-up delay; however, these parameters are fixed at design time, and if there is no buffer model constraints, no particular values could guarantee correct operation. The standard needs to be sufficient to guarantee correct operation of decoders designed to conform to the standard.
- It's not acceptable to rely on real-time flow control from decoder to encoder to try to avoid these problems (see e.g. eDP 1.4 compression)
- When a decoder fails to decode a stream, we need an objective, formal method to determine whether fault belongs to the decoder or the stream. This is a fundamental principle of all high quality standards.

Solution: HRD (Buffer Model)

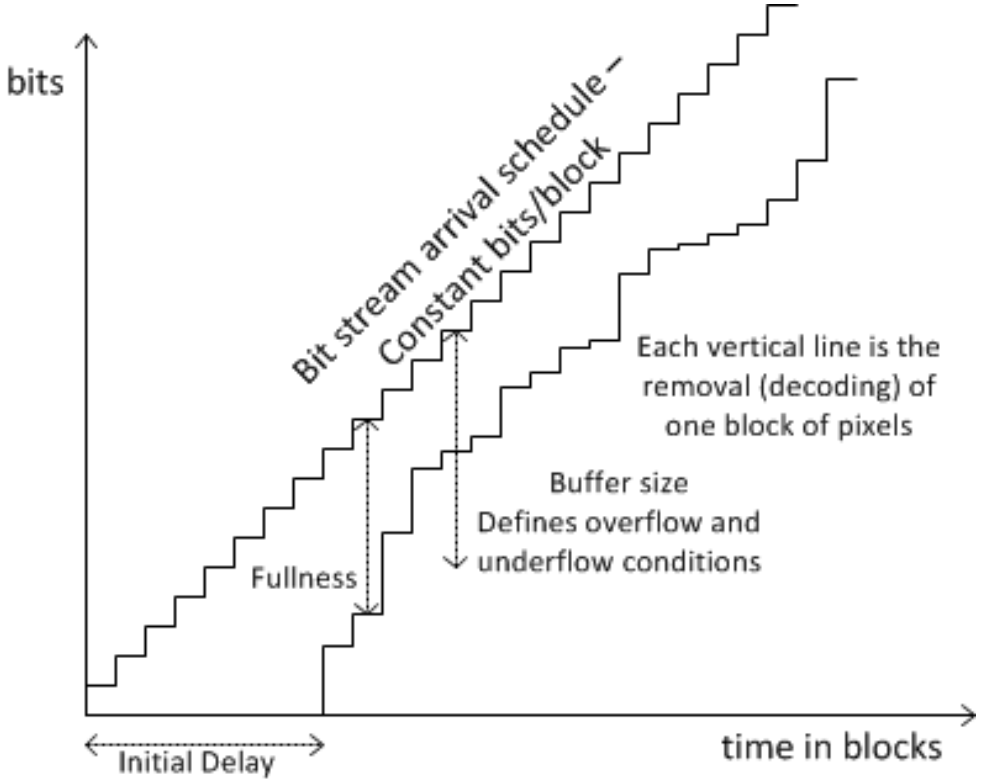


- HRD is a constraint, a test, applied to bit streams, which either pass or fail
- Bit streams that pass the HRD test have a bounded variation in the cumulative numbers of bits per block
- Real decoders can be designed to decode all streams that pass the HRD test
- HRD is a simplified, idealized model of a decoder with a rate buffer
- Bits enter the buffer on a schedule and leave the buffer on a schedule
- The buffer is never allowed to overflow nor underflow
- All major public video standards have HRDs (sometimes called VBV), going back at least to H.261 (1980s)

HRD Parameters

- Reference schedule
  - The input and output schedules must be specified with a consistent reference
  - For DSC, the simplest, most general way to specify this is in terms of a defined unit of pixels i.e. a block
  - Here a block is treated as a unit of time without requiring specification of an absolute time of the pixels
- Input schedule
  - Bit stream arrives at HRD at a specified schedule
  - A specified number of bits are added to the HRD buffer at each block time
- Output schedule
  - Bits leave the HRD at a specified schedule: Initial delay and rate both specified in terms of block times
  - Again, for DSC, the simplest, most general way to specify the rate is to remove all the bits that code each block, once per block time
  - Initial delay needs to be specified in the standard
    - Could be e.g. a fixed number of blocks after the first bit of a picture arrives, or a number of block times that is a function of the HRD buffer size and the bpp rate
- Buffer size
  - Some number of bits, which needs to be specified in the standard

Example Bit Stream



- Graph displays bits vs. time. This is convenient for analysis of HRD fullness and bit rate
- This example is CBR. Bits arrive at a specified number of bits per block time. Bits/block time does not have to be an integer.
- Arrival schedule is a number of bits every block time
- Bits are removed from the HRD at regular intervals corresponding to block times. The number of bits removed is variable.
- At every instant, the HRD buffer fullness is the distance from the arrival schedule line and the removal schedule line. We need always 0 <= fullness <= buffer size. Any exception is a failure.

Relationship of HRD to Real World

- HRD is not something that gets implemented. It is simply a constraint on bit streams.
- HRD test can be applied to any bit stream in non-real time e.g. analysis software. This is done commonly for popular video standards. If the HRD uses block as the unit of time, as proposed here, no timing information is required to analyze streams.
- Decoders can rely on all streams conforming to the HRD
- Engineers can design decoders that are guaranteed to decode all conforming streams
- The actual buffer size and actual bit arrival and removal schedules may be different in a real decoder vs. the HRD. It is the designer's responsibility to know the difference.
- Encoders need to be designed to guarantee that all output streams conform to the HRD, for all possible input video. How they do that is outside the scope of the standard. Typically an encoder has a rate buffer that is at least as large as the HRD buffer and an algorithm to ensure HRD conformance.
- Bit stream needs to incorporate schedule information i.e. bpp rate and initial delay
- The transport layer is responsible for delivering the bitstream in accordance with the bpp rate regardless of the transport characteristics i.e. HBI, bursts, etc.

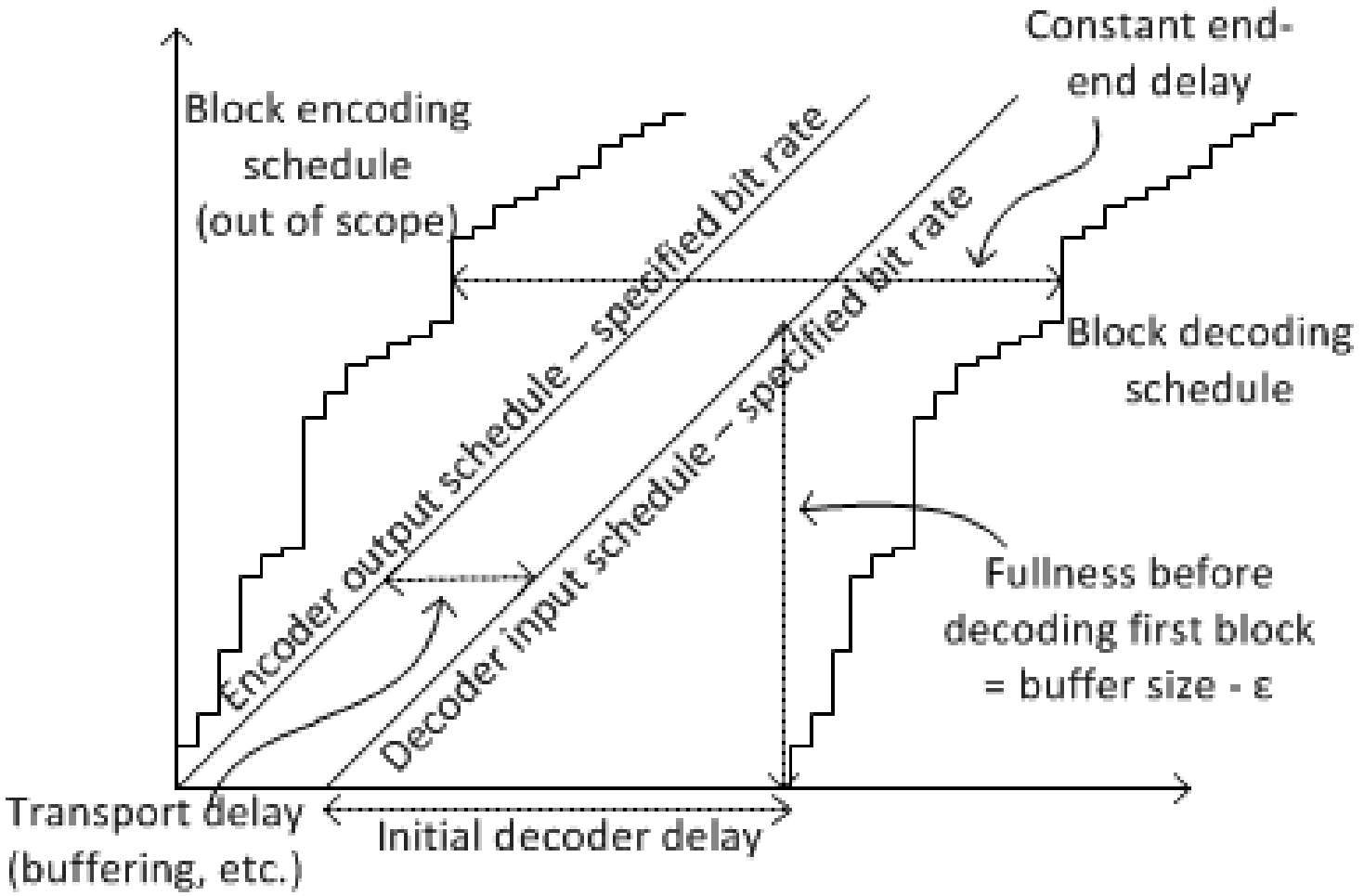
DSC Video, HRD and Transport

- How to account for transport layer timing?
  For example HBI (horizontal blanking)
- Keep video and transport layers separate, and keep them simple.
- Suggested approach:
  1. Specify video bit stream rate as a constant (nominal rate) for the full duration from start of first line of a frame to the end of the last line of the frame, not stopping for HBI. Completely independent of the transport, and very simple.
  2. If transport layer has HBI time (typically the case), the transport layer incorporates buffering in the amount necessary to map the constant rate video to the bursty rate transport and deliver video at the correct constant rate. Details are straightforward to calculate, and specific to each transport.
  3. Constant rate video with no HBI is consistent with decoders that operate continuously through HBI and use buffers to map to actual display timing, which is advantageous for throughput.
  4. Encoders produce streams with a constant bits/pixel rate (as indicated).
  5. In practice the encoder and transport may cooperate (not necessary) to minimize buffering size.

Example for Illustration

- Bit rate = 10.25bpp
- Block size = 2 pixels => 21.5 bits/block
- HRD buffer size is 2**15 bits = 32,768 bits = 4k bytes
  - Note: could be significantly smaller or larger
- Initial delay = floor(buffer_size/bitsperblock) = 1,524 blocks
- Encoder has a rate buffer size = HRD buffer size (could be larger)
- Decoder has a rate buffer size = HRD buffer size (could be larger)
- Encoder encodes blocks and inserts bits from each block in rate buffer
- Encoder transmits alternating 21 & 22 bits/block every block from its rate buffer. It inserts stuffing bits as needed to prevent underflow of its rate buffer.
- Encoder algorithm ensures that bit streams conform to HRD
- Transport layer transports compressed bits to the decoder in real time with a fixed delay, buffering as necessary to achieve this
- Decoder receives bit stream at constant rate for all active line times
- From the time when the first video bit of a frame arrives, the decoder waits 1,524 pixel block times (initial delay) before starting to decode video
- Decoder decodes a block of pixels every block time. In this example, a block time is: Total line time / number of active pixels per line * 2 pixels/block
- Decoder incorporates a decoded pixel buffer to convert constant decode rate to actual pixel clock rate with HBI. This is out of scope of the standard.

Example Operation



Note: Bit schedules are illustrated as straight lines for simplicity

Implications to DSC

- All proposals need to show formally how they prevent decoder buffer overflow and underflow. Best way to do this is via an HRD. Proposals can specify their own details of their HRDs.
- Final DSC standard needs to specify the HRD details that are appropriate to that standard, including specification of initial delay (could be constant or a fixed function of other values), and bit stream indication of bpp rate.
- Benefits to DSC of a proper HRD include:
  - Guarantee of robust real time interoperability of encoders and decoders from different vendors
  - Ensure independence of compression spec from transport layer
  - Eliminate uncertainty in the design of decoders and encoders

Appendix: Pathological Example

- A simple example is described where an algorithm needs only a single line of reference memory but huge rate buffers.
- Each line begins with a single bit copy flag.
- If the flag is set, data is copied from the previous line and no additional data is needed. Thus one line buffer is needed for prediction; not part of HRD
- If the flag is not set, data for all samples of the line follows.
- For a full HD image (1920x1080 24bpp) each line requires either a single bit or 1+1920*24 bits. The compression ratio each line can alternate between 1:1920*24 and approximately 1:1
- The encoder is free to decide which lines to copy and which to signal. For example, it could alternate essentially halving the vertical resolution or it could identify duplicate lines and allocate the bandwidth to active areas of the image to give a 12bpp average.
- This encoder freedom hides a serious implementation issue
- For example if the encoder chooses to fully code half the lines consecutively of a 1080 line frame, the rate buffer size would need to be at least 1080/2*(24-12)*1920 = 12,441,600 bits!

Buffering Needs of Example

- Simple examples stress a decoder designed for this algorithm and needing pixel reconstruction at a regular rate. Note on average the frame is compressed to 12bpp from original 24bpp in these examples.
- Top Heavy Example: majority of bits used for top half of frame
  - All lines of top half are coded by signaling pixel data, all lines of the bottom half with skip flags.
  - Consider a decoder operating to decompress this frame. During the top half of the frame, pixel data is arriving at half the rate necessary to meet the decoding needs. To avoid stalling the decoder, the decoder needs to wait until half of this data arrives before starting. This requires roughly a 25% of an uncompressed frame buffer assuming an average 2:1 compression ratio.
- Bottom Heavy Example: majority of bits used for bottom half of frame
  - All lines of the top half are coded with the skip flags, lines of the bottom half signaling pixel data.
  - Consider a decoder operating to decompress this frame. The top half of the frame is decoded in half a frame time. During this time, 540 bits have been processed by the decoder but half of the data for the frame has been received requiring a buffer to hold roughly half a frame times worth of compressed data or ¼ frame buffer assuming an average 2:1 compression ratio.
- Alternate Top Heavy and Bottom Heavy Frames.
  - Consider the frames should be produced at a regular rate.
  - The decoding of the Top Heavy frame must be delayed by ½ a frame time as above
  - For regular output frame rate, the Bottom Heavy frame must also be delayed by ½ a frame time.
  - In total, a buffer capable of holding compressed data equal ½ an uncompressed frame is needed.