# WHAT MAKES A GOOD PROFESSIONAL VIDEO CAPTURE/PLAYBACK CARD?

***How we designed a "Software Defined Interface" capture and playback card built around 21st century software practices***

## Executive Summary

One of the most important parts of a software-based encoder or decoder is the SDI card as it is the main interface for video in or out. As a result we have developed our own in-house card to add features and reliability not found from existing vendors in the market.

In particular our card provides lower latency than existing cards on the market.
In addition it handles processing in software, something that was once thought impossible. But doing this allows for improvements such as guaranteeing lipsync. At the same time having a vertically integrated card allows us to quickly diagnose any issues and add new features.

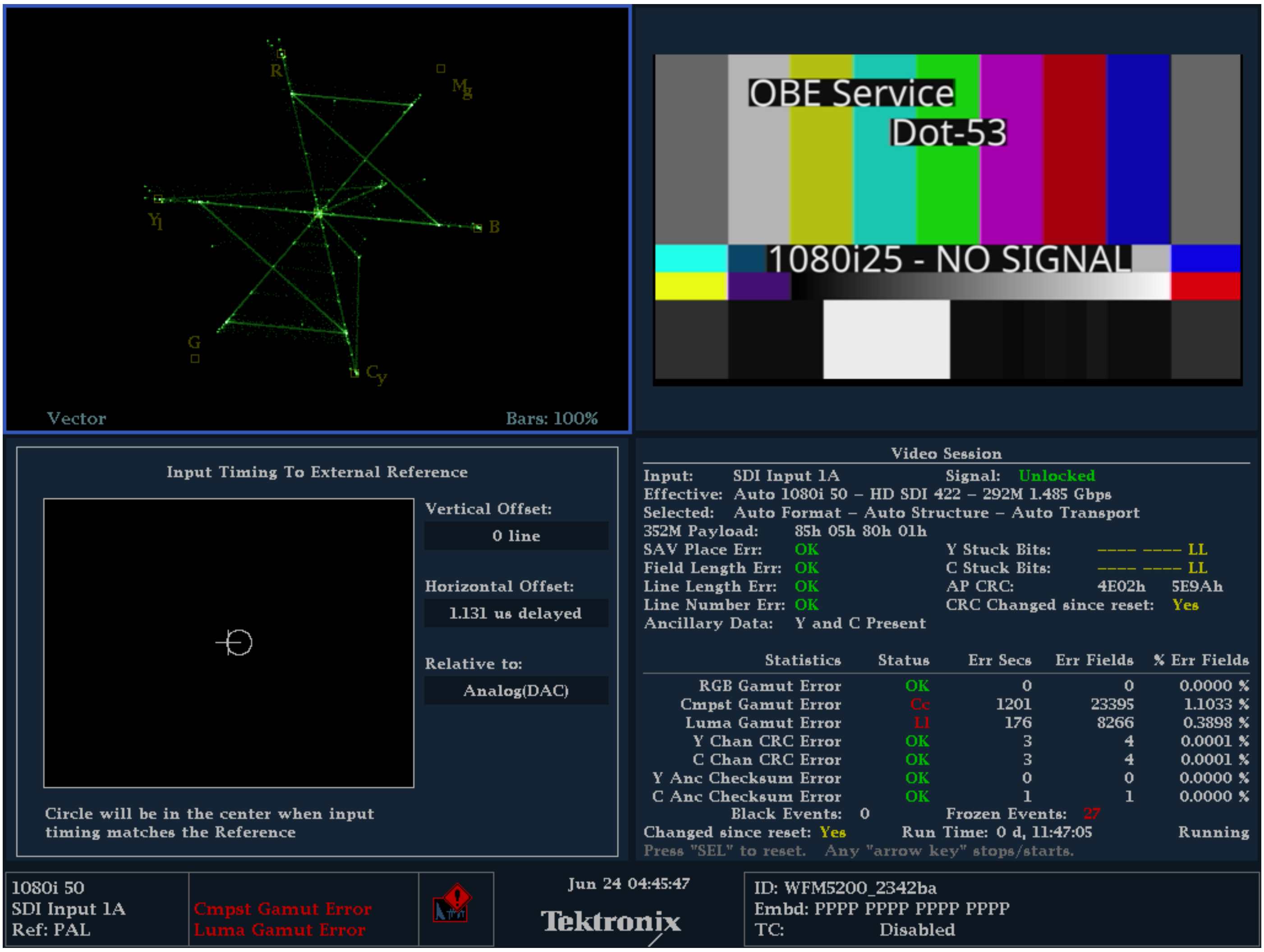
## Technical Details

To celebrate the first large scale deployment of our in-house SDI card, we're explaining a few core design decisions that we've found make the difference when they are being used in 24/7/365 professional capture and playback. There might be other use-cases (e.g keying, scaling etc.) where these decisions may not be right for everyone. But for us, these design decisions have delivered massive operational improvements and will drastically improve our end to end latency by an order of magnitude. This is what we do best: applying 21st century development practices to a legacy application. It is the culmination of years of hard work by a dedicated team.

NOTE: At this point in time our card is only available for use with our products

## Low Latency

Most capture cards on the market deliver a frame at a time (for HD usually 1000 lines or so). However, processing data frame by frame adds a large processing delay. By contrast, our card allows for around 32 lines of delay, which is already a 30x improvement on the status quo! The same applies for playback. We want to be sure that as soon as we write to the hardware that data will be on the wire in a few milliseconds maximum. On playback many cards handle genlock by buffering one or more frames in the hardware. This adds excessive latency. By minimizing the buffering in hardware we can minimize latency. Our hardware releases the frame co-timed with the genlock pulse after minimal time.



```
Input Timing To External Reference

Vertical Offset:
0 line
Horizontal Offset:
1.131 us delayed
Relative to:
Analog(DAC)

Circle will be in the center when input
timing matches the Reference

                          Video Session
Input:      SDI Input 1A                Signal:   Unlocked
Effective:  Auto 1080i 50 – HD SDI 422 – 292M 1.485 Gbps
Selected:   Auto Format – Auto Structure – Auto Transport
352M Payload:     85h 05h 80h 01h
SAV Place Err:    OK                    Y Stuck Bits:        ---- ---- LL
Field Length Err: OK                    C Stuck Bits:        ---- ---- LL
Line Length Err:  OK                    AP CRC:              4E02h    5E9Ah
Line Number Err:  OK                    CRC Changed since reset:  Yes
Ancillary Data:   Y and C Present

              Statistics    Status   Err Secs   Err Fields   % Err Fields
           RGB Gamut Error      OK          0            0       0.0000 %
         Cmpst Gamut Error      Cc       1201        23395       1.1033 %
          Luma Gamut Error      Ll        176         8266       0.3898 %
          Y Chan CRC Error      OK          3            4       0.0001 %
          C Chan CRC Error      OK          3            4       0.0001 %
     Y Anc Checksum Error       OK          0            0       0.0000 %
     C Anc Checksum Error       OK          1            1       0.0000 %
             Black Events:   0           Frozen Events:  27
Changed since reset: Yes     Run Time: 0 d, 11:47:05                Running
Press "SEL" to reset.   Any "arrow key" stops/starts.

1080i 50        Cmpst Gamut Error      Jun 24 04:45:47    ID: WFM5200_2342ba
SDI Input 1A    Luma Gamut Error                          Embd: PPPP PPPP PPPP PPPP
Ref: PAL                                                  TC:         Disabled
```

## Open Source Drivers

The vast majority of capture cards have closed source drivers. In an operational environment it is important to fault-find quickly and to be able to see what’s going on in order to pinpoint a problem (which often has nothing to do with us). This is especially true if the driver is heavily involved in the capture or playback process. But by being able to see every change we can be sure there are no regressions. The worst cases are when a vendor requires signing paperwork to have access to the drivers. There is often little to keep secret (except maybe the fact that there is not much there?).

## APIs with Locked Audio

It might seem counter-intuitive having asked for Open Source drivers then to say we don’t want to use Video4Linux2 and ALSA. But these are generally built around consumer video applications which can’t deliver many of the features that we need for professional 24/7 video.

The biggest problem with V4L2 and ALSA is that audio and video are presented separately. This means that they can’t be opened at the same time and thus are always non-deterministically out of sync by a few milliseconds. This might be acceptable for consumer applications but there are engineers out there who will measure every last millisecond. Likewise for applications which carry compressed data such as Dolby E, this slippage can be unacceptable.

A few vendors do this properly and provide either timestamps or a common callback for video and audio. But care has to be taken as to the validity of the data, this still needs to be sanitized in software otherwise it can lead to lip sync issues. The only way to guarantee lip sync is to unpack in software (our next topic).

There are also lots of issues with ancillary data such as VANC and HANC, containing data such as closed captions, which are not exposed by V4L2, likewise multichannel audio in ALSA. The same goes with providing an exposed clock so that we can generate frames at the correct rate to output.

There are often good commercial goals such as wanting to provide a simple cross-platform API that doesn’t require an understanding of SDI. Sometimes these often are used with other, more complex protocols such as HDMI which are bidirectional. But abstracting these features away often causes more problems than it’s worth. It’s very useful to have direct access to hardware and firmware to understand what’s going on.

## Software Packing and Unpacking

This one’s going to be controversial. We think that software unpacking and repacking of SDI data is the best approach in 2020. Traditional thinking has always been based around offloading as much as possible to the hardware, but this comes at the expense of fine control. Today, computers regularly process tens of gigabits of data per second with ease and it is now not a problem to build and pack SDI frames in software. There is one exception to this which is the SDI CRC, a very costly operation in software as it uses a polynomial and data size that are not software friendly. This is easily added inline as the frame is put on the wire.

As described above, lip sync is exceptionally important. By unpacking the data ourselves we can guarantee that the audio is in sync on capture.

The SDI data format itself is not software friendly but with clever use of SIMD the process can be made very fast, often 5-10 times faster than a C implementation. One of the more unusual things that we do is treat SDI as just another file format (albeit one which is very high bit rate). This means we can keep captures from a wide variety of equipment and can run regression tests to make sure that exactly the same output is produced whenever we change something, a process widely used in regression testing suites such as FATE. Likewise we can run fuzzing processes to make sure that software behaves consistently. What’s notable is the majority of this work does not require specialist knowledge. We use this in a combined SDI/2022-6 stack that saves us a massive amount of effort. In many respects the hardware is merely a FIFO as we would like it to be.

## Capture and Playback (with Genlock)

Last but not least we are often asked to support capture card X. Often these don’t have any output support and sometimes no genlock. These are out of scope for our needs and generally don’t have the reliability we need for 24/7/365 operation.
So if you’re looking for ultra-low-latency encoding and decoding with the flexibility of off-the-shelf hardware, please get in touch!

Got this far down? Does this sound interesting? Do you want to apply 21st century software practices to broadcast television? Visit https://www.obe.tv/careers/ for more information.