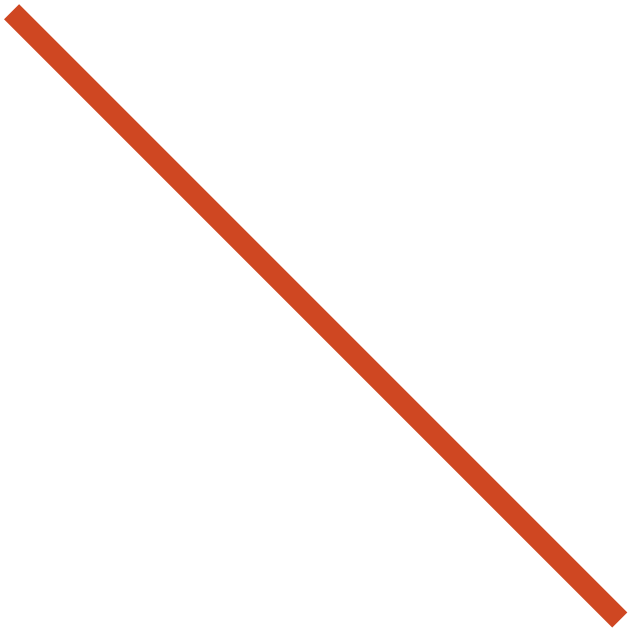
Software COTS For Real Time Broadcast



Essential Guide

EG

ESSENTIAL GUIDES

Introduction from Tomer Schechter & Gal Waldman

Hello, and thank you for downloading this Essential Guide, covering the world of 100% Software 100% COTS 100% IP for Live Production and Playout applications.

TAG Video Systems and The Broadcast Bridge strongly believe there is a lot of misinformation concerning uncompressed (and compressed) workflows, working in a 100% software, 100% COTS environment.

TAG therefore decided to partner with the premiere technical knowledge source in our industry to dig deeper than ever before and articulate exactly how this is being done in full production in the leading broadcasters in our industry.

Imagine you could easily manage SMPTE 2110, JPEG 2k, and H.264 all within the same software? What if you could use off the shelf hardware to deploy multiple applications (Probing, Monitoring & Multiviewing) with the scalability and flexibility that you define. When and where you need it.

After 11 years of deploying over 40,000 channels of 100% software on 100% COTs in 100% IP for the global leaders in broadcast and media, we have gotten used to the skepticism. So we decided to go deeper than we have even been before. Dive inside the code and inside the hardware to make you comfortable with exactly how we do it, so you no longer need to worry about the constraints of traditional broadcast hardware solutions specifically, flexibility, and scale.

Tomer Schechter & Gal Waldman: Founders of TAG Video Systems.

TAG is very active in many of the organizations committed to helping standardize IP workflows , including membership in standards organizations, participation in interoperability events and plugfests, and ensuring our API's are available for maximum utilization for both out clients and our partners.

We hope to share our experiences with you and help you help your customers have the most engaging experiences possible.

We hope you will find this Essential Guide to "Software COTS For Real Time Broadcast" useful, and that it becomes a tool you can refer to again and again.

Please share it with your friends and please let us know how we may help you by contacting us at info@tagvs.com.

Best regards,

Tomer & Gal

Supported by

Software COTS For Real Time Broadcast

By Tony Orme, Editor at The Broadcast Bridge

Traditional broadcast infrastructures relied on bespoke hardware devices to process video and audio in real time. But the advances in High Performance Computing (HPC) in unrelated industry sectors such as high-frequency-trading and telecommunications has led to a proliferation of high-speed hardware COTS server and network equipment becoming available for industries such as broadcast television.

Broadcasting has traditionally relied on hardware solutions to distribute and process uncompressed baseband video due to the high-speed data and low-latency requirements. Dedicated cables with point-to-point connectivity provided predictable bandwidths and low latency but at the expense of scalability and flexibility. It's difficult to expand an SDI router without redesigning large parts of the infrastructure resulting in quickly escalating costs. Furthermore, any significant additions to the service would often require the procurement of even more hardware. Not only would this add to the direct broadcast costs, but also to the support services such as power, air-conditioning, and space.

Software services are often seen as the optimal solution due to their scalable flexibility and faster time to market. However, the unpredictable nature of software timing introduced latency and the high-speed data-throughput required made the proposition of a software solution nearly impossible, especially with COTS solutions. Up to recently, the idea of developing software solutions on COTS hardware didn't even appear on the radar for many broadcasters. However, this has now changed.

The synchronous nature of broadcasting means we still have some challenges to overcome to make real-time media work in an asynchronous IT environment and in this Essential Guide we explain how this has been achieved. COTS servers and networks can now deliver the processing speed and data throughput needed for broadcast television, but to achieve SMPTE ST-2110 and ST-2022-6 timing constraints, more work must be done at the kernel software level.

A multiviewer is a typical example of a device that would have traditionally used FPGA's. They are a critical part of the broadcast workflow as engineers rely on them to provide accurate confidence monitoring including pay-per-view services. Software COTS multiviewers are a perfect example of the challenges that have now been overcome to provide the guaranteed data throughput and low latency demanded by uncompressed video.

Kernels

In kernel based operating systems such as Linux, the kernel is the software responsible for interfacing hardware peripherals to users to provide a layer of hardware independence. That is, standard peripherals such as network interface controllers (NIC's), keyboards, and hard disk drives have their low-level interfaces abstracted away from the user and developer through a common API (Application Programming Interface). The kernel also provides the file system and a method of time sharing so many processes can be run on the server independently of each other but giving the illusion of running in parallel.

Added to the kernel are programs that facilitate the user operation and management of the server with tools such as the bash shell for command line access, and programs including "ls" to list the files in a directory. There are network management programs such as "ifconfig" and methods of managing user rights to files and directories with "chmod". The kernel, along with all these other programs (and there could be thousands of them), all combine to form the operating system. Linux is not an operating system in itself, it's a kernel. Distributions such as Ubuntu, Debian, and SUSE are operating systems as they take the kernel and add other software to it to form a complete operating system.

This all culminates in the server running a service that allows multiple programs to be scheduled and access the hardware in a timely and organized manner without conflict.

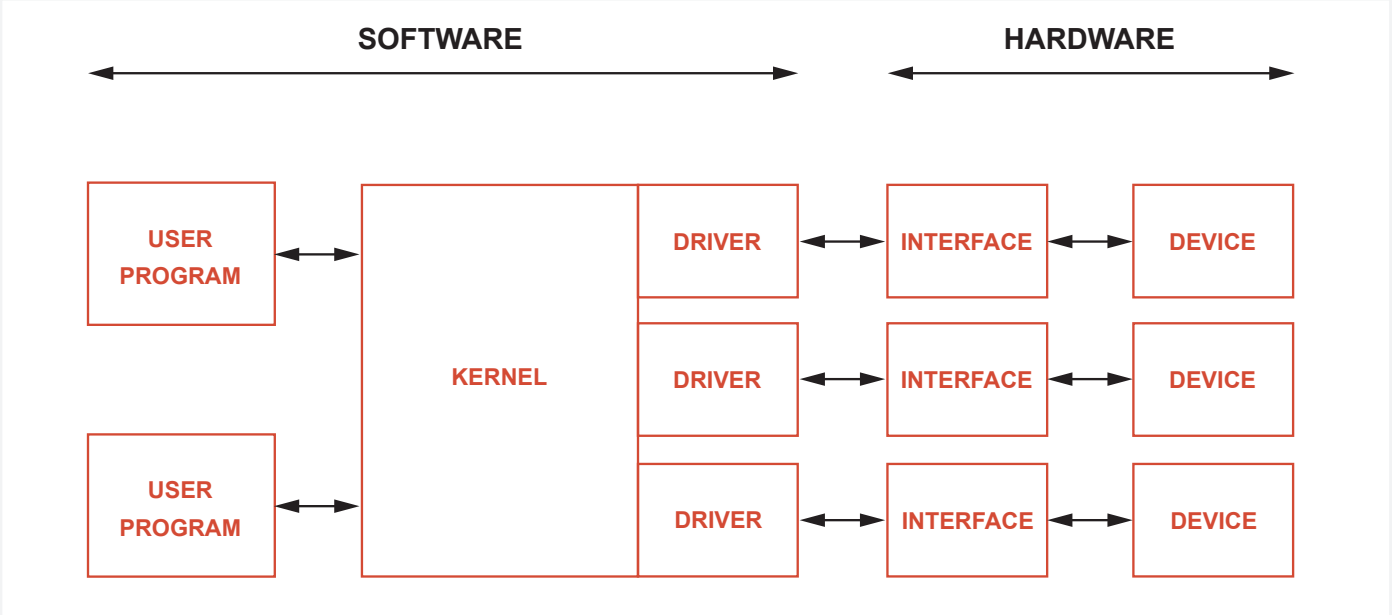


Diagram 1 – The Linux kernel showing the relationship between the hardware and user. The User Program is in the user-space and everything right of the kernel is in the kernel-space.

Generic Solutions

However, the kernel is a generic solution and asynchronous in its operation. Unlike a counter running on an FPGA that guarantees events happen within a tight time specification, a kernel contains message queues and buffers, and process scheduling leading to relatively unpredictable event timing. The kernel and operating system sacrifice timing accuracy for a generalized ease of use. And this is the key challenge we face with COTS implementations, that is, making the whole system respond in a more predictable manner.

The Linux kernel has two distinct methods of operations; user-space and kernel-space. Programs providing services such as multiviewers reside in the user-space but the low-level access to the hardware resides in the kernel-space. The kernel presents a library of system-call API's for the user program to call and abstract away the hardware to provide a generic API. This greatly simplifies access to the NIC and other peripherals for the developer.

One of the challenges of using the Linux network stack (the suite of networking software services residing in the kernel) is that it can be relatively slow. As the kernel provides a generic API through system-calls for multiple processes, it also provides many functions that may not be needed and add unnecessary overhead for a broadcast specific application.

Furthermore, services such as packet filtering, forwarding, routing, and socket allocation all take place in the kernel. A whole section is dedicated to TCP which won't be needed in a broadcast application because we are only interested in UDP for ST-2110 and ST-2022-6.

Receive

When a packet is received by the NIC it is initially stored in its own local buffer. A DMA (Direct Memory Address) process takes place to copy the packet directly into the kernel's memory space. Upon completion, the CPU is invoked and a series of house keeping tasks take place. One of these is the skbuf (socket buffer) process. This is used deep within the Linux kernel to help validate, filter and route packets. The skbuf is a form of vectored IO, that is, the skbuf data structures contain header information, populated by the kernel, about the ethernet frame and encapsulated IP packet. This allows the application data received from the network and the header information to be stored in different places.

In the generic Linux network stack solution, multiple user processes may need their own copies of the received IP packet or frame. The socket process creates an interface and data structure for each process it is attached to.

It determines which process needs a copy of the data and copies the IP packet or complete ethernet frame into its user-space memory. This is a necessary function in the general network stack solution as many processes may need copies of the data. The skbuf data structure and pointer system makes this process more efficient for multiple processes receiving the data.

A side effect is that a double memory copy has taken place, the first is a DMA process from the NIC to the kernel-space memory, this is very fast as it is hardware accelerated and it doesn't take up much CPU time, and the second is a CPU intensive process to copy the packet from the kernel-space memory to the user-space memory, and this may be repeated multiple times depending on how many sockets are configured for multiple processes. Furthermore, the kernel is acting as a bottleneck.

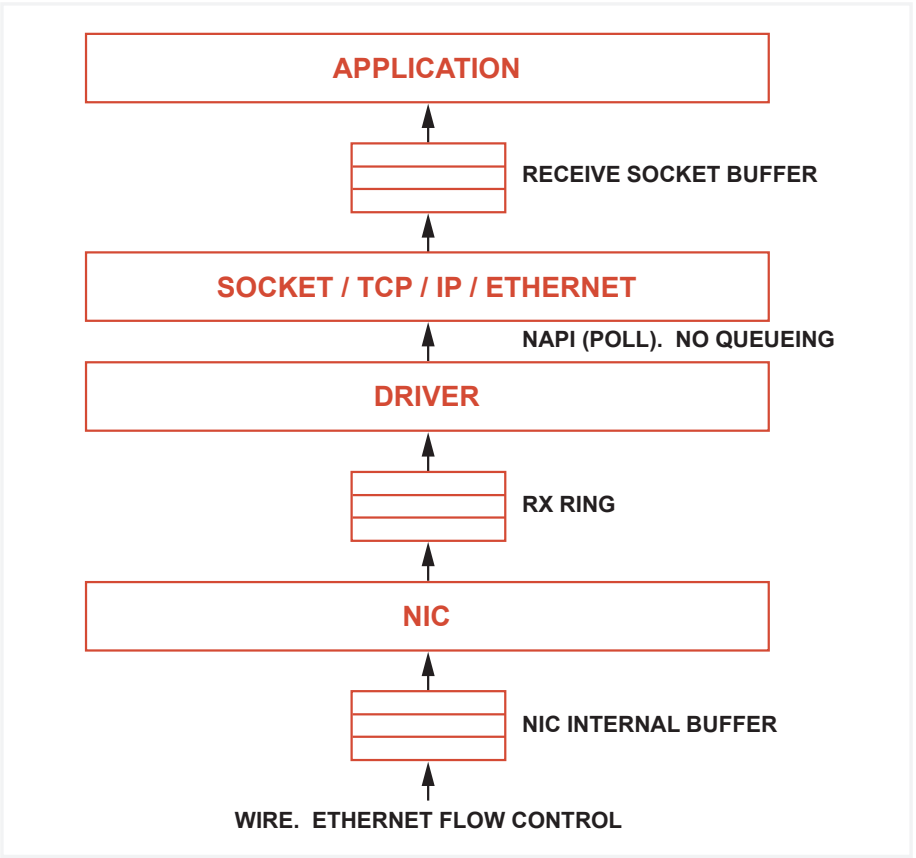


Diagram 2 – Receive process and buffers occurring between the hardware, kernel, and user program.

When the packet is initially copied to the kernel-space memory, the kernel validates the packet checksums and populates many of the values in its skbuf data structures. These are used by other processes in the kernel and user spaces. Copying the packets from the kernel-space memory to the sockets in the user-space memory involves queueing them in a buffer. Further adding latency.

Sockets are used extensively in Linux to abstract away the concept of communicating with other processes on the same server and other servers using the standard Linux descriptor file. In essence, every Linux input and output operation is accomplished by reading and writing to a descriptor file. A descriptor file can be a text file, terminal, NIC, or something else. The same system-calls such as read() and write() are used to communicate with all compliant devices thus providing a generic API. To the user program they are treated the same, but the underlying kernel code is written specifically for the device being controlled.

The user-space process either continually polls the network socket, or it waits for a signal from the kernel to invoke a user-space process indicating the packet is now available to the user application.

Transmit

When a packet is sent from the user-space process the program first writes it to the relevant network socket associated with the NIC. The socket has a queueing buffer to temporarily hold the packet (or packets) allowing the user-process to continue without too much delay. The socket process is invoked and creates the necessary packet headers and checksums such as the UDP header and then the IP and Ethernet header. At each stage the IP encapsulation, the various checksums are calculated and appended as required. A transmit skbuf is created and all the fields within the data structures are populated.

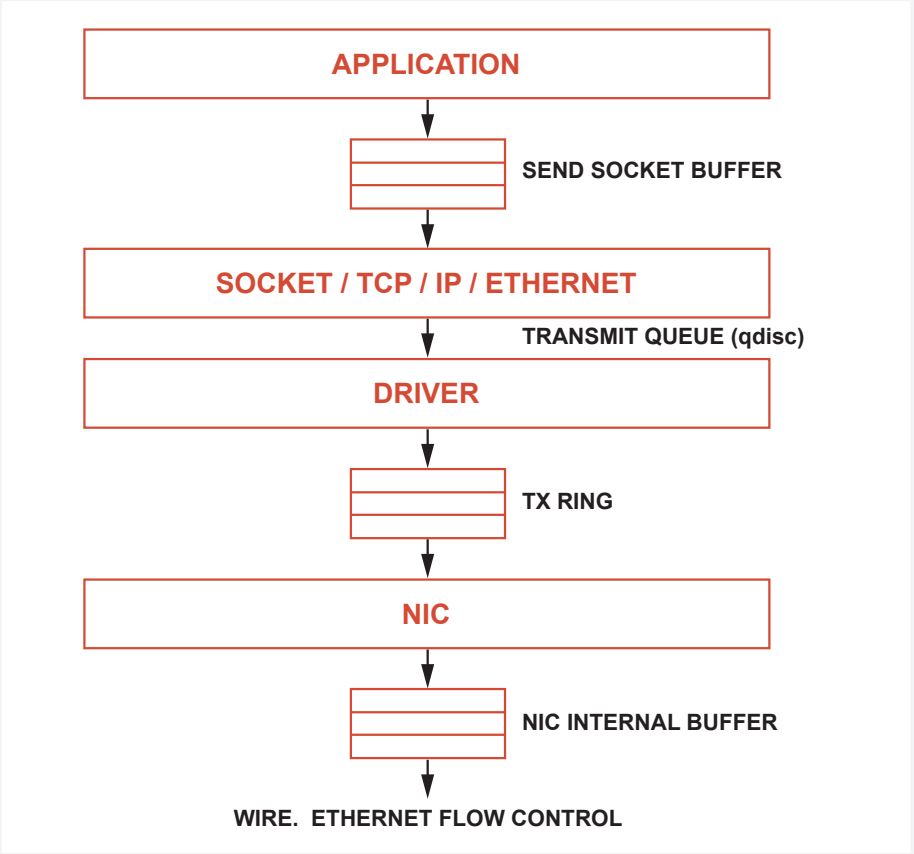


Diagram 3 – Transmit process and buffers occurring between the hardware, kernel, and user program.

The transmit skbuf data structures are required as multiple processes may need to transmit data from the same NIC, this is quite likely if the server has processes such as SSH and HTTP. The skbuf structures are linked together so the transmit part of the NIC driver can provide a many-one mapping of packets to stream them on a single NIC. This idea scales as multiple NICs can be used for load balancing across multiple network links and CPU processors.

The socket process writes the packets to the transmit queue of the NIC's driver which in turn writes the packets to another queue for the NICs transmit engine. When the NIC is ready, it finally transmits the packets across the ethernet to send them to their next location.

The speed with which the packets can be sent is initially governed by the wire speed of the NIC. If this is 10Gbps then due to packet overhead and NIC processing, the fastest the NIC could hope to transmit is approximately 90% of the line speed, or 9Gbps. If there is network congestion and the NIC cannot write packets to the wire fast enough, then the later packets queue in the drivers' buffer. If overflow occurs, then the transmit queue in the socket buffer queue starts to accumulate packets, and if this fills, the socket send buffer connected to the application eventually fills and any packets generated after that are lost.

Evenly Gapped

One important aspect of ST-2110 is that the packets must be evenly gapped at about six-microsecond intervals for HD progressive. There is some variance in these, and packets are not sent during line and field blanking, but the tolerance is very tight. It can be seen from the previous explanation of the transmit queue that the number of buffers involved and potential for latency and packet jitter will make it difficult to achieve these constraints.

To make the Linux stack as flexible and generic as possible, there is a great deal of overhead going on in the kernel-space processing. As we have a very specific application in broadcast IP infrastructures, that is ST2110 and ST-2022-6 UDP packet distribution, most of the features provided in the kernel-space processing creates an unnecessary overhead. With a specific user-application running on a COTS server such as multiviewer, we can make some very specific assumptions about the data-stream for our case. Namely, we don't need to use most of the networking processes the kernel is providing for us.

With ST-2110 and ST-2022-6 applications, the major requirement is to quickly move the data to the required memory space for the user-process application. If this was a video compression application, then our focus would be less on network data throughput and more on CPU resource efficiency as compression can be CPU intensive. But in the ST-2110 and ST-2022-6 broadcast IP infrastructures, we're mainly interested in achieving high data throughput with the smallest latency possible.

Kernel Bypass

To achieve this, we use a system called kernel-bypass. To be specific, we are only bypassing the network stack part of the processor for a specified NIC or group of NIC's. We are not bypassing the kernel in its entirety.

On the server receive side, in kernel-bypass, the NIC is polled directly by the kernel-bypass code running in user-space to determine if it has received a packet from the ethernet into its local buffer. If it has, then the kernel-bypass code uses a DMA process to transfer the data directly into the multiviewer user-space memory making it available to the application. As the packet is expected to be a UDP packet then the code can be optimized for this one specific case. There may be TCP connections, but it is up to the vendor how they deal with these. They may well just be ignored.

Using kernel-bypass on the receive side improves data throughput by several orders of magnitude as a layer of buffering has been removed and the kernel-space bottleneck serving the sockets has been negated.

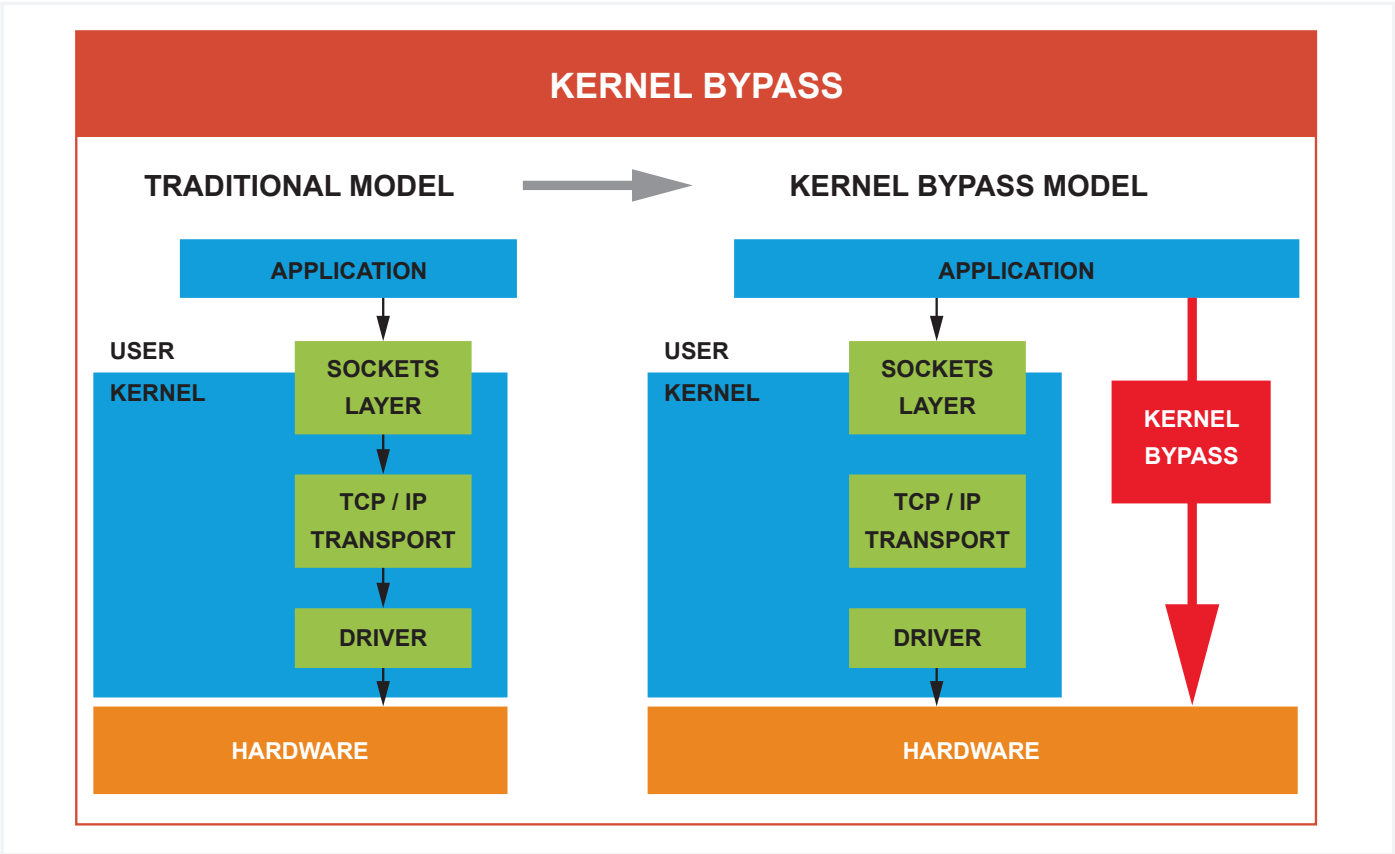


Diagram 4 – Kernel bypass allows direct connection to the hardware from the user-space to significantly improve data throughput and reduce latency for broadcast applications. The kernel network stack can still be used for monitoring and maintenance processes using a separate NIC.

The transmit side of kernel bypass is similar with one key difference, and that is related to making the packets evenly gapped. When the multiviewer application running in user-space needs to send a packet in kernel-bypass, it uses the DMA process to write the data from its memory to the NIC buffer directly. This significantly reduces kernel latency. However, it does not necessarily provide evenly gapped packets.

NIC Solutions

NIC's employ many strategies to transmit packets to the ethernet network. Some wait for their buffers to be half full before they start transmission, and some will transmit as soon as they receive a packet. Ethernet networks tend to work in burst mode to improve efficiencies so NIC's tend to follow this and provide burst transmission. But ST-2110 demands evenly gapped ethernet frames.

To date, the easiest way of achieving this is to use NIC's built specifically using hardware timers to pace packet transmission. Such NIC's can be programmed to evenly gap (within limits) packets to meet the SMPTE specifications for narrow transmission to make it compatible with hardware receivers downstream. This helps reduce latency as the receive buffers can be significantly reduced in size.

However, programming the NIC's and writing kernel-bypass code requires a deep understanding of not only the specific hardware device drivers, but the interaction between the user-space buffers, NIC buffers, and the DMA process that links them. Effective buffer management is critical in making sure the two buffers are full enough so they don't empty, but not so full that packets are dropped or latency increases.

Dedicated NICs

Kernel-bypass assumes at least one NIC is dedicated to transmitting and receiving real-time video and audio as it provides a one-to-one mapping between the NIC and the user-space process. Therefore, a sperate NIC is needed for other applications for maintenance and monitoring using SSH and HTTP type protocols. This helps enormously with system integration as the low latency time critical media multicast streams can be kept away from the TCP traffic.

However, advanced NIC processing does allow for the provision of just a single NIC within the server. Complex traffic prioritization algorithms can determine the optimal routes for kernel IP traffic as well as kernel-bypass broadcast IP streams.

The concept of moving data from user-space directly to the NIC without storing data in the kernel-space buffers is referred to as zero-copy. Quite often the DMA is used to provide hardware acceleration and further improving data throughput.

Specialist NIC's such as these are available from several vendors who specialize in low latency data distribution in other industries including high-frequency-trading and telecommunications. Consequently, they still tick the COTS box.

Some of the data throughput used in broadcast IP infrastructures is truly amazing. Not so long ago, 10Gbps was considered high-end. Now 25Gbps and 40Gbps infrastructures are finding their place in broadcast infrastructures, especially as we look at UHD and 4K. But even an HD progressive stream can generate just under 3Gbps, or about 200,000 IP packets a second - a colossal amount of data.

COTS servers can host multiple NICs within one chassis. This allows network administrators much more freedom when balancing loads for multicast streams, something most broadcast IP video and audio infrastructures will be using.

The hardware restrictions of the past have been removed by the software COTS solutions now available. Flexible and scalable infrastructures are achievable giving broadcasters unprecedented choice.

Vendors are now able to deliver software solutions that were once only achievable in hardware. Software multiviewers receiving ST-2110 and ST-2022-6 IP streams are now available in COTS servers, even for 4K and UHD. Servers are only going to get faster and software more efficient, so there is a world of opportunity opening in front of us for software COTS solutions.

The Sponsors Perspective

Multiviewing and Probing for Live Production and Playout Using 100% Software on COTS Servers

By Tomer Schechter & Gal Waldman: Founders of TAG Video Systems.

Successfully Deploying 100% SW 100% COTS with Uncompressed IP Signals.

TAG Multi-View Monitoring for Live Production and Playout.

Executive Summary

The preceding article lays down the foundation of how, thanks to IP and advanced software techniques, it is now possible to support uncompressed video signals in 100% software solutions running on Commercial Off the Shelf (COTS) computers servers. This key development means that complex live production and playout workflows can be implemented on generic IT equipment leading to the economies and added flexibility of the IT industry's data center model.

Here, we take a look at one of the more complex functions in live production and playout applications; Multi-View monitoring and probing and how TAG Video Systems works with uncompressed formats such as SMPTE ST 2110 and SMPTE ST 2022-6 as well as compressed formats such as JPEG 2k, H.264 and H.265 all within the same system to bring the maximum flexibility and scalability to these critical broadcast applications.

Supported by

Live Production

Of all the applications in a broadcast facility, none are more demanding technically and operationally than live production. Below is an overview of typical live production workflow. On the left we have a combination of local cameras and external feeds arriving by either satellite or fiber. In most cases the feeds are decoded back to baseband (uncompressed) but in some remote production applications some feeds arrive as J2K encoded (compressed) and remain compressed within the production facility. Then there are the core functions of replay, graphics and the production switcher. On the far right we see the various different rooms and operator positions where the cameras, feeds, replay and production elements are monitored. Each operational position is equipped with a monitor wall consisting of multiple displays each showing a mosaic consisting of multiple videos sources, clocks, timers, under monitor displays and tally indicators. Pictured at the center of the figure is the Multi-View processor which receives all of the signals and creates multi-view outputs to feed the monitor wall displays.

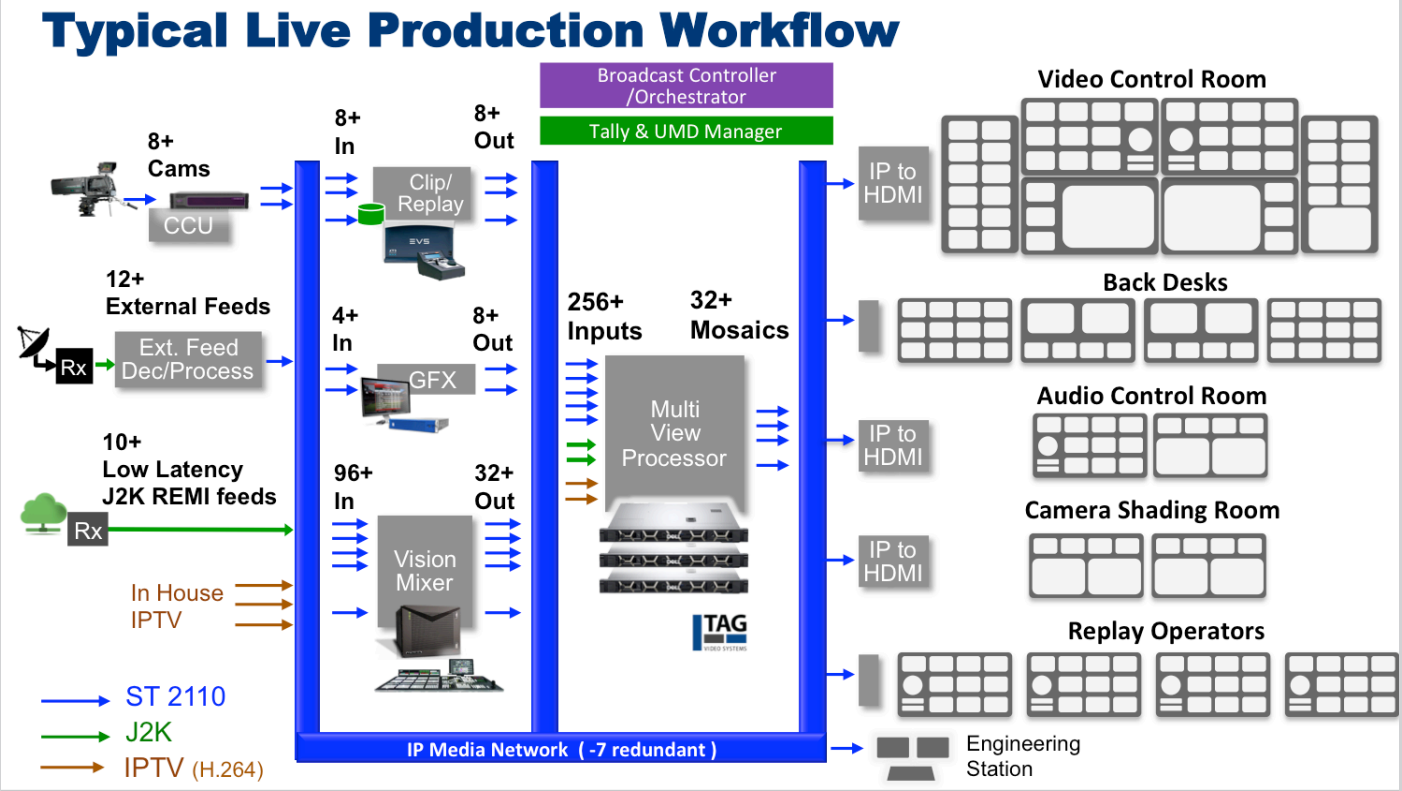
Challenges of Multi-View Monitoring in Live Production

Scale:
The sheer number of uncompressed inputs and outputs within a live production requires very high bandwidth to allow the multiviewer to handle, in the example pictured above there are over 256 inputs displayed on 32 independent multi-view screens.

Latency:
The speed in which the data must be processed in the multiviewer in order to offer the sub 2 frames per second latency that is critical in live production.

Quality:
Scale and low latency must be achieved while not scarifying the image quality when scaling HD and UHD images for the multi-view display.

Integration:
Must integrate seamlessly with tally and broadcast control systems to handle dynamic nature of a typical live production.



Supported by

The TAG Live Production Solution

Until recently, the only option to handle these performance requirements was for the multiviewer to be based on dedicated, broadcast specific hardware. TAG is more commonly recognized for being the first to provide a software based multi-viewing of compressed video signals. In 2017 TAG introduced support for SMPTE ST-2020-6 and then ST-2110 uncompressed video signals and deployed 1000's of channels combining compressed and uncompressed signals. Then in 2018 TAG added integration with tally management and broadcast control systems.

TAG Video Systems introduced a 100% software, multi-view monitoring solution for live production applications in 2017 that handles native ST 2110 and ST 2022-6 uncompressed signals while running on generic COTS server hardware. Today TAG has deployed 1000's of channels globally of live production and playout managing both uncompressed and compressed formats all within the same solution. Below is an overview of the TAG solution and how it has overcome the traditional challenges.

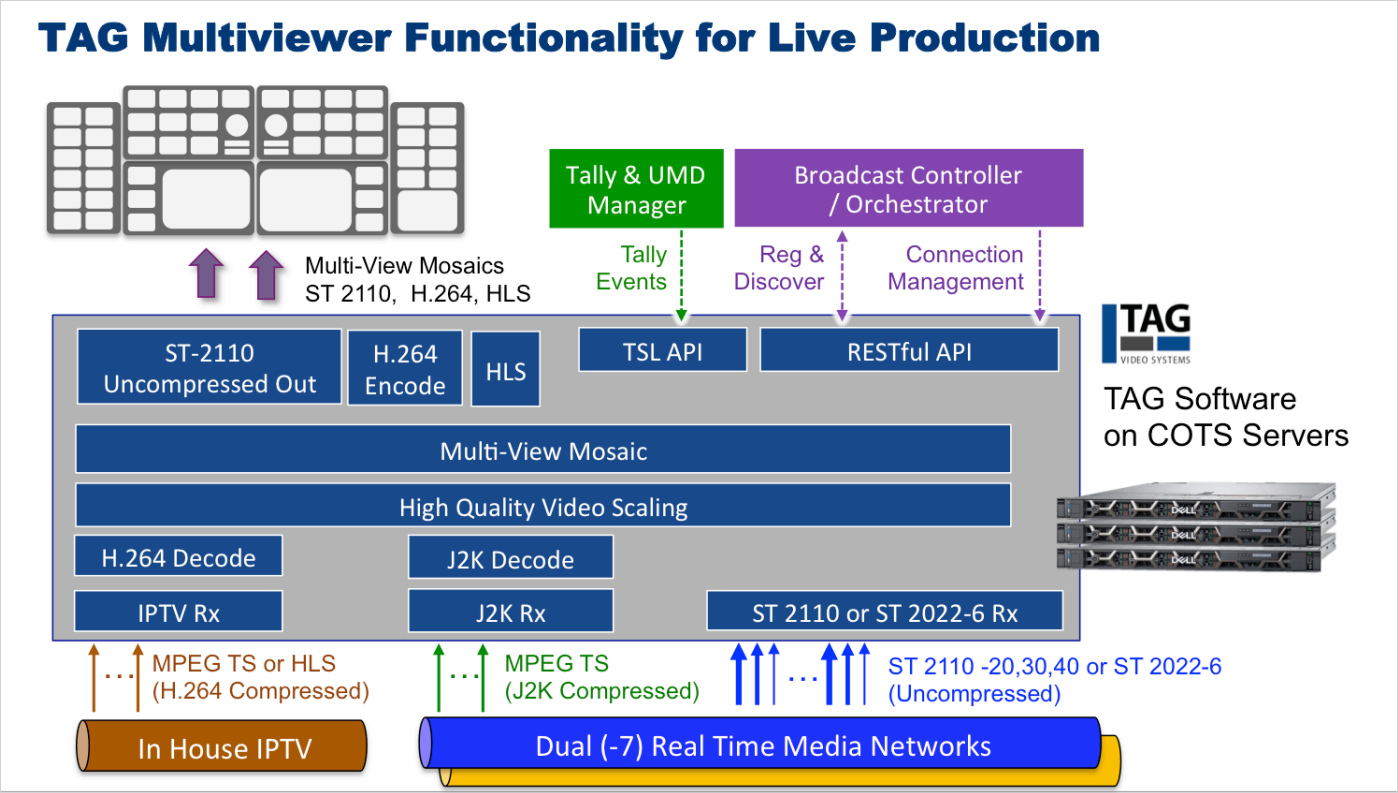
Scale:
Infinitely scalable. Unlimited number of inputs and outputs can be managed. Processing requirements are easily defined to calculate the number of COTS servers required to deploy. Scale the system when and where you want at whatever number of inputs and outputs.

Latency:
Lowest latency of any software multiviewer in the market - sub 2 frames per second.

Quality:
Supports full UHD and HD inputs and performs high quality scaling to produce UHD multi-view mosaic outputs with the image quality demanded by the most discerning TDs and Camera Shaders.

Integration:
Fully supports all major tally and broadcast control systems

Agility:
Runs on the same generic COTS server hardware used by other production and IT systems.



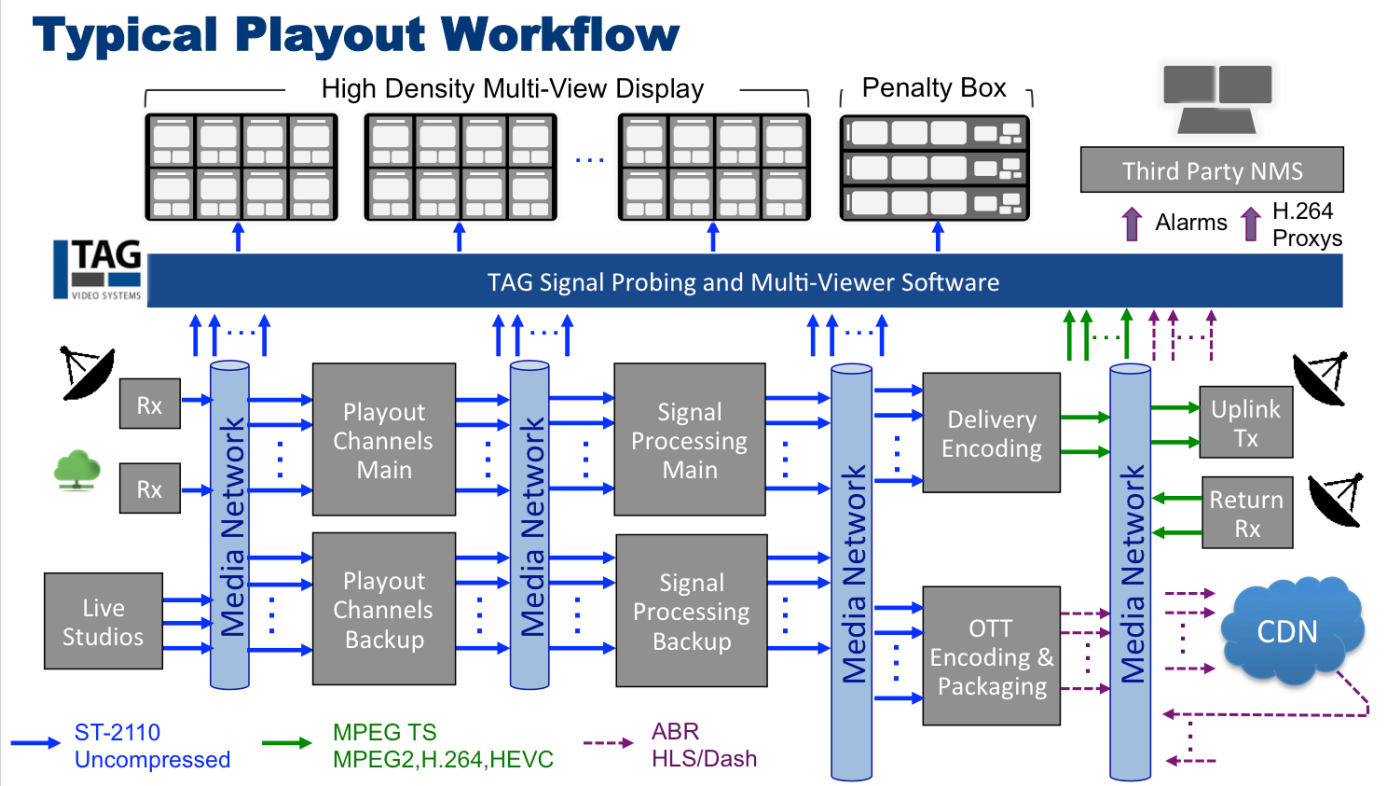
Supported by

Playout

The playout application has many of the same challenges of live production. The main differences are the need to support more different compressed and uncompressed formats and the need to do probing of the various signals to detect signal faults. Below is an illustration of the workflow of a typical playout facility.

A playout facility typically includes a combination of compressed and uncompressed signals which need to be monitored by operators managing the channels. In the facility illustrated above the playout area integrates a combination of pre-recorded material and live content originating on remote feeds or from the control room of local studios. The uncompressed is represented by the blue arrows.

Once encoded for delivery we enter the compressed world represented here by the green arrows and OTT ABR/HLS/ Dash feeds represented by purple arrows. The TAG probing, monitoring and multiviewing solution is represented above with typical high-density display layouts and integration with 3rd party network management systems (NMS).



Supported by

TAG Playout Solution

Below is an illustration of the functionality provided by the TAG solution for probing, and multviewing within the playout application. TAG's ability to work with both uncompressed and compressed formats provides the greatest flexibility in handling the most complex workflows. In addition, TAG's solution offers both the probing and multiviewing function within the same solution which provides simplified workflows for the operators. Finally, TAG's robust and open API structure allows deep integration with third party network management systems and broadcast control systems to be easily configured and provide you with the critical information you need to operate and manage the facility. All within a 100% software, 100% COTS solutions.

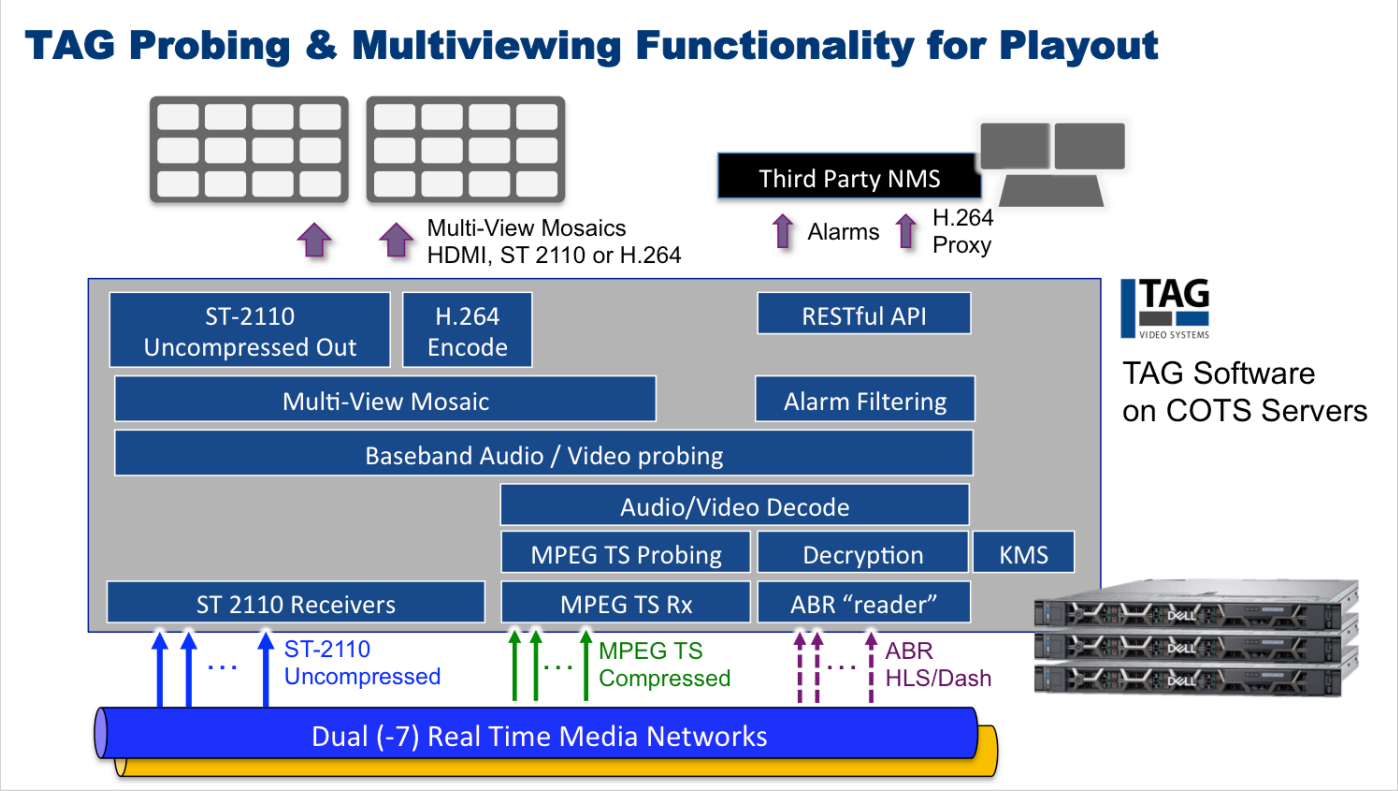
TAG Leveraging the Full Potential of IP Workflows

For years broadcasters have recognized the value of moving to an IP workflow. Until recently, this promise has not been fully leveraged due to the massive bandwidth, speed and quality requirements of the primary broadcast applications. Both the live production and playout applications push these requirements to the limit. And within these applications, the technical demands on the multiviewer with the large number of inputs, outputs, speed quality and agility, have required dedicated hardware to process the application.

TAG Video Systems has conquered this limitation and offers 100% software running on 100% standard off the shelf hardware solutions for probing, monitoring and multiviewing, for all broadcast applications (live production, playout, satellite/cable/IPTV and OTT delivery) integrated into one solution.

This now provides broadcasters with far more simplified workflows, with the highest level of scalability and flexibility in the market at the highest quality. The use of COTS server hardware frees broadcasters from having to buy and support custom, application specific hardware or vendor supplied computer appliances.

We believe strongly that in the near future, broadcasters will be able to deploy 100% software on COTS hardware for all functions across the broadcast ecosystem and finally leverage the full potential of IP.



Supported by

WP
WHITE PAPERS

EG
ESSENTIAL GUIDES

MEDIA

CS
CASE STUDIES

Find Out More

For more information and access to white papers, case studies and essential guides please visit:

thebroadcastbridge.com

9/2019

Supported by

TAG VIDEO SYSTEMS