Technische Universität München
Department of Informatics

Master's Thesis in Informatics

Evaluation of IP Communication in FPGA-based Camera Platforms

Felix Lampe

Technische Universität München

Department of Informatics

Master's Thesis in Informatics

Evaluation of IP Communication in FPGA-based Camera Platforms

Evaluation von IP-Kommunikation in FPGA-basierten Kameraplattformen

Author	Felix Lampe
Supervisor	Prof. Dr.-Ing. Georg Carle
Advisors	Paul Emmerich, M. Sc., Sebastian Gallenmüller, M. Sc.
Date	July 15, 2017

Informatik VIII
Chair for Network Architectures and Services

I confirm that this master's thesis is my own work and I have documented all sources and material used.

Garching b. München, _____ _____

Date Signature

Abstract

In recent years, the motion picture industry has been facing challenges regarding the communication infrastructure deployed in their facilities. Emerging technologies such as new image formats and applications require higher data rates than the commonly installed systems can handle. This Master's Thesis aims to evaluate the idea of using the Internet Protocol as the basis for the communication in such environments. For this task, an architecture for the integration of an Internet Protocol subsystem into professional cameras is proposed, based on an analysis of the problem and the requirements. The devised system is evaluated in a prototype implementation on Field Programmable Gate Array (FPGA) hardware using throughput and latency measurements. The results demonstrate high potential, but also indicate possible difficulties that have to be overcome before this solution can become useful for commercial adoption.

Zusammenfassung

Die Film- und Fernsehindustrie hat seit einigen Jahren zunehmend damit zu kämpfen, dass ihre Infrastruktur mit den immer schneller steigenden Datenmengen neuer Anwendungen und Bildformate kaum noch mithalten kann. Diese Master's Thesis befasst sich mit einem auf dem Internet Protocol basierenden Ansatz für die Kommunikationsinfrastruktur in Studios und an Filmsets. Basierend auf einer Anwendungsanalyse wurde eine Architektur entworfen, die diesen Ansatz umsetzt. Zur Evaluation der vorgeschlagenen Architektur wurde ein Prototyp für eine FPGA-Plattform implementiert und anschließend in Messungen hinsichtlich des erreichten Durchsatzes und der Latenz untersucht. Die so gesammelten Ergebnisse zeigen das Potential der Lösung auf, weisen allerdings auch auf einige Schwierigkeiten hin, die für eine kommerzielle Umsetzung überwunden werden müssen.

Contents

Chapter 1

Introduction

Not too long ago, the motion picture and broadcast industries switched from analog technology to new and promising digital equipment and from there to devices capable of handling High Definition, revolutionizing their infrastructure in the process. The Serial Digital Interface (SDI) standard has been developed specifically to meet the requirements of the novel systems and has been widely adopted throughout the trade. However, emerging applications such as video on demand, 3D cinema, High Frame Rate, and Ultra High Definition introduced a need for higher data rates and more flexibility than what was achievable using the prevalent infrastructure. This eventually led to new solutions being explored, one of them being the Internet Protocol suite.

1.1 Motivation

Devices on film sets and broadcast studios are highly interconnected to allow the distribution of the captured video and to facilitate the synchronization and the cooperation of cameras and other equipment. The predominant infrastructures are based on a high number of unidirectional point-to-point links, which offers poor flexibility and has higher costs and space requirements than otherwise necessary.

With the aforementioned applications and new image formats emerging, the existing equipment is close to its limit regarding bandwidths. In the face of an upcoming costly infrastructure upgrade, companies are looking for alternative solutions for the growth in bandwidth and format variety. The computer networking standards based on Ethernet and Internet Protocol (IP) are one promising option that offers the quickly growing data rates of data centers, coupled with a high flexibility because of the content-agnostic approach to communication.

1.2 Objectives

The main objectives of this thesis therefore are to analyze the benefits and requirements of integrating IP communication into professional cameras and to devise and evaluate an architecture that accomplishes this. A prototype FPGA design is developed based on that architecture which is then used in hardware to conduct performance measurements to gauge the quality of the approach. In the measurements, the latency and throughput of the prototype will be assessed to gain some insight about whether or not the concept of IP communication in a camera is useful to combat the challenges the industry is facing.

1.3 Outline

The thesis is structured as follows. Chapter 2 gives an overview of the sort of cameras this work deals with, and illustrates common environments and use cases for those devices. It should give readers that are not familiar with the inner workings of the motion picture and broadcast industry the background knowledge needed to understand the reasoning in later chapters. In Chapter 3, the requirements of networking infrastructure in the broadcast and cinema production industries are examined. Reasons are given for why the current situation is problematic, together with possible solutions, and which of them was selected as the optimal solution. The approach to the chosen solution is presented in Chapter 4, including a description of a prototype architecture as well as some implementation details. An evaluation of the prototype is performed in Chapter 5, containing performance measurements of the system in various scenarios. The throughput and the latency of the prototype are examined in two development stages, and the overall usefulness of the system in a real scenario is assessed. Chapter 6 offers some perspective on related work by presenting research similar to this thesis, and shows the similarities and differences between them. Some opportunities for future work are detailed in Chapter 7, including ideas on how to improve the weaknesses of the prototype system devised in this thesis. Finally, the conclusion can be found in Chapter 8, summarizing the key points of this work.

Chapter 2

Background

To better understand the overall topic and the specific problem at hand, it is necessary to have some background knowledge of the cameras' hardware and system structure, and the environment in which they are used. This chapter therefore aims to provide background information, so that the reasoning based on it in later chapters becomes more clear.

2.1 Type of Cameras

The cameras that are subject to this thesis are professional-grade digital motion picture cameras. They are commonly employed for cinema productions, documentaries, advertisements, high-budget TV series and large broadcast events like sports tournaments. For most productions it is unreasonable to buy them, if they will only be used for a relatively short time period. Instead, they are usually rented from an equipment rental company on a daily basis. Exceptions are some long-running TV series and teams or cinematographers who plan to produce multiple movies with the same equipment, are able to afford the initial investment, and want to save on recurring costs over a long time instead.

The rentals take care of device maintenance and firmware updates, as well as licensing of extension features like a higher variety of video codecs and formats, high-speed recording, or wireless remote controlling. The basic features like recording at a somewhat limited set of frame rates, codecs and image formats are normally available regardless of any additional licenses. To give an idea of the dimensions and look of the hardware, Figure 2.1 shows four camera families that some recent movies were shot with [1, 2].

Their higher weight, larger size and higher price compared to consumer-grade cameras are caused by their high recording quality, the wider feature set, and the increased reliability.

Figure 2.1: From top left, clockwise: ARRI ALEXA Mini (drone-mounted), ARRI ALEXA Plus, Sony PMW-F55 and RED Epic Dragon [3–6].

High-end sensors have a large dynamic range, low signal-to-noise ratio, and short readout times at increasing resolutions, allowing for high-quality video capture at high frame rates. All the details contained in these images equals a lot of data that has to be processed and stored reliably, which necessitates powerful processing hardware.

For consumers and smaller projects, the codec used to encode the raw sensor data is mostly insignificant, as long as it is supported by the platform used to view the clips. For professional productions however, certain post-production tools and workflows might require specific codecs, which means cameras should support at least some variety to allow them to be used in conjunction with common processing software and hardware. Especially important is the ability to record raw image data on the camera, which can later be arbitrarily encoded and used with any tool of choice, leaving all options open until post-production starts. However, recording raw data requires a great amount of data storage, and might not be possible at high frame rates as the data rates might exceed the power of the internal hardware or the storage medium.

Many professional cameras provide internal color processing, often with different settings for the recording and monitoring image paths, which adds a great deal of complexity and thus requires hardware resources.

The cameras also offer far more options for external monitoring of the captured image, such as live monitoring with status overlays and focus or exposure highlighting. This is

important in expensive productions, as having to repeat a shot means high additional costs. In contrast, consumers usually do not suffer any relevant consequences if a recording does not turn out as expected.

To be able to handle all of the aforementioned features, such professional cameras mainly consist of a general purpose CPU and either a FPGA or an Application-Specific Integrated Circuit (ASIC). The CPU runs an operating system (usually some derivative of Linux), which handles the user interface and hardware interactions like USB and network interfaces, while the FPGA or ASIC captures image data from the sensor, performs image processing, encoding and decoding, and provides outputs for recording and monitoring. Aside from those two central components, there are multiple pieces of highly specialized hardware that manage tasks such as temperature control, audio processing, or wireless networking.

Since different productions and settings call for a variety of combinations of image formats, frame rates, color processing, codecs, and other parameters, the FPGA or ASIC is very powerful compared to hardware built into consumer-grade video cameras. For systems that employ an FPGA, depending on the selected parameters, a fitting firmware configuration is loaded to allow many parameter combinations without wasting FPGA resources. Variants based on ASICs might need to switch between multiple components for certain settings, thus implementing the features in parallel.

Having the computationally intensive routines run on reprogrammable FPGAs, instead of on ASICs, opens up the possibility to change functionality of those subsystems via firmware updates containing new FPGA configurations. This is important, since it greatly extends a camera's product life-cycle by adding features via software and firmware updates, compared to cumbersome and expensive hardware upgrades or even replacing the entire camera with a new model.

Considering the relatively high, five-digit cost of digital cinema cameras even without accessories, being able to keep up with new developments without having to replace the hardware adds a lot of value to the system. In addition, ASICs have a low cost per unit, but a high initial cost for initializing the production and relatively long time to market [7].

Because the market for professional digital cinema cameras is rather limited, the initial expenses of ASICs can outweigh their low price per unit and make them less attractive compared to FPGAs in this kind of application, unless the expected amount of sold units is high enough. It is rather difficult to find information about the technology manufacturers use. However, ARRI AG published information on how FPGAs are used in their Alexa camera [8], and Altera Corporation (now part of Intel) revealed in 2014 that ARRI's Amira camera makes use of FPGAs as well [9]. Xilinx Inc., the biggest competitor of Altera/Intel in the FPGA market, stated that the Blackmagic Design URSA

camera is based on a Xilinx FPGA for the exact reasons listed above [10]. The RED Epic uses an ASIC, although no details have been published [11].

2.2 Typical Environment

In contrast to camcorders and system cameras targeted at private use, professional digital cameras are used in a very different environment. Consumer products have to be easy to use and must work well in a wide range of situations without the need to carry additional equipment or change a lot of settings to get a good result. In contrast, professional cameras are commonly connected to a multitude of accessories and equipment during operation, and their parameters must be fine-tuned to the current shooting conditions to get the best possible image. This applies to both studio productions and outdoor filming and has great impact on the requirements of the cameras.

Equipment on set usually includes a number of monitors of different sizes as well as viewfinders for viewing footage, both during capture and afterwards. Viewers with various roles must be able to monitor different portions of video concurrently, streamed from one of possibly multiple recording cameras, or other sources like recorders, storage units, or processing workstations in live production scenarios.

Another common practice is Return In Monitoring, in which the finished live image of a multi-camera production is sent back to the cameras as input that can be routed to a connected monitor or viewfinder, so that the operators can review the final output and adjust their shot accordingly. If cranes are used to capture a scene from above, external monitors are the only way to preview what will be recorded. In this situation, a low latency from the image sensor to the remote display is even more important than in other setups, because the operators must be able to react quickly to changes in the scene, even without a viewfinder.

While displays obviously play an important role on movie sets and TV productions, they are not the only type of devices that communicate with the cameras. Distance and brightness sensors together with lens motors can adjust lenses to allow for automatic control of focus, zoom and aperture. Signal generators for timing locking (often called Generator Locking (Genlock)) facilitate synchronization between the image sensors of multiple cameras and external equipment, e.g., for audio. Recorders, as well as processing workstations need to receive the captured video as well as metadata such as timecode or which camera a video stream has been captured with.

Many contemporary camera systems thus aim to provide a multitude of interfaces and offer great flexibility, because the environment in which they are used differs greatly from one production to another.

2.3 Hardware and Interconnect Variety

The aforementioned devices are connected to the camera using many different interconnect standards and medium types. For example, most monitors use Serial Digital Interface (SDI), the standard video link interface in the industry, and therefore have Bayonet Neill–Concelman (BNC) plugs. External viewfinders in contrast might use common video standards like HDMI, or a custom connection, and often implement a proprietary control protocol on top of the video transmission to allow controlling the camera via buttons mounted on the viewfinder, or to transport two streams of video data for the viewfinder eyepiece and a side-mounted display.

Some accessories like lens motors for adjusting focus are connected with custom plugs and proprietary protocols, while simple remote control devices that provide additional configurable buttons use a General Purpose Input/Output (GPIO) interface. Many cameras also feature a web-based user interface which can be accessed through an Ethernet port with either an RJ45 jack or yet another special connector, or via IEEE 802.11 wireless LAN. Software updates and configuration files are usually provided using USB media, which has yet another hardware interface. Genlock signals for device synchronization sometimes share the BNC connectors of SDI, but multiple other connectors are also used.

This wide variety of protocols, physical mediums and connectors necessitates interfaces for all those standards on the cameras. Because the camera is in most cases the central unit of a setup, it has to be able to communicate directly with most of the accessories. For each type of jack on the camera body, new hardware parts have to be selected, ordered and assembled, which is tedious and adds complexity compared to a system which relies on only a few uniform interfaces.

This not only affects camera production but also preparation for filming, because the crew has to ensure that the equipment they wish to use is compatible, or that they otherwise have adapters ready to facilitate interoperability. Multiple infrastructures might be necessary to provide the needed communication channels, further increasing the cost as well as weight and space requirements and setup time of the hardware. Additionally, knowledge about the different protocols and interfaces is needed to handle problems that occur during usage, such as loss of connection or quality issues.

2.4 SDI as Industry Standard for Video Transmission

The Serial Digital Interface family of SMPTE standards is the de facto standard in the film and TV industry for transmitting uncompressed live video in environments like movie sets and production studios. It transmits image content via 75 Ω coaxial copper cables

with BNC connectors at nominal data rates of up to 360 Mbit/s (SDI [12]), 1.5 Gbit/s (HD-SDI [13]), 3 Gbit/s (3G-SDI [14]), 6 Gbit/s (6G-SDI [15]) and 12 Gbit/s (12G-SDI [16]). The 6G- and 12G-SDI standards have only recently been published, so they aren't widely deployed yet [17]. A single link 24G standard is supposedly in the works since late 2013, but apparently has not officially been completed as of writing this [18, 19]. However, bandwidths of 24 Gbit/s and 48 Gbit/s can be realized with aggregated 12G-SDI links.

Current SDI installations are almost exclusively realized as copper cables with BNC connectors, both of which are becoming an issue regarding the move to high rate links. The main reason is that the medium and plugs incur relatively high losses, that can be mitigated at lower data rates, but significantly reduces the signal quality at data rates like 24 Gbit/s. The consequence is either a high bit error rate or greatly reduced maximum transmission distance [20,21], especially since only error detection, but not correction, is provided. This requires additional hardware like repeaters and switches, and would add to the already high space consumption and weight of multiple cables making up a single link, which is relevant in outside broadcasting trucks or in cramped filming locations like small rooms. As an alternative, fiber optic media have been standardized for above 10 Gbit/s [22], since those offer better characteristics in that respect. However, optical fiber cables are much less robust compared to coaxial copper, which is problematic in environments like film sets where it is common for connections to be dynamically installed and changed often, for example attached directly to the floor with tape.

To allow companies to keep using their existing infrastructure instead of having to upgrade constantly, more recent variants can be implemented by multiplexing over multiple connections of older installations [17, 23]. A 12G-SDI link for example can be built by aggregating four individual 3G-SDI links. This kind of multi-link setup is commonly used for data rates beyond 3 Gbit/s, for example in 3D monitoring use cases, for which dual link SDI was initially developed [23].

SDI is specifically tailored to transfer uncompressed video frames, which are sent line by line as time-multiplexed streams of brightness ("luminance") and color ("chrominance") samples. The sample bit depth can be either 10 bit or 12 bit, but this thesis will assume the first unless stated otherwise. The start and end of each frame is signaled via reserved codes and allow for synchronization between endpoints. Between video lines and frames, the specification allows for ancillary data to be transmitted, because these intervals were intended as time for repositioning the electron beam in CRT TVs, which is not needed in modern devices. The ancillary data can be anything, including some common uses like embedded audio, timecode, metadata and error detection [13]. Because the data is multiplexed, multiple video streams of lower data rate can be multiplexed into a higher rate link. For example, two 3 Gbit/s signals can be carried by one physical 6 Gbit/s link. The SDI standards specify exactly how the (de)multiplexing has to be performed in such cases.

This highly specialized, non-proprietary protocol ensures many parameters that are important for live video transmission, including low latency and jitter, the possibility of frame accurate switching, no frame drops and unimpaired visual quality. On the other hand, its use is limited to transmitting video signals and only small amounts of other data, and the optimization comes at the cost of flexibility.

Chapter 3

Problem Analysis

In recent years, numerous technological innovations made their way into the motion picture industry, which gave rise to novel cinematographic possibilities and practices, but also came with new difficulties. Some of those problems are related to device interconnectivity and data transmission, which are the ones that this work focuses on.

Examples of other problems include supplying power to drone-mounted cameras for extended periods of time, or creating storage media that can hold large amounts of footage data but also provide high write speeds, even at high operating temperatures.

3.1 Applications and Data Rates

Among the most challenging new applications for digital movie cameras are Ultra High Definition (UHD) and High Frame Rate (HFR), i.e., resolutions and frame rates that are multiples of what has been common for the last decades. For instance, increasing the pixels per frame by a factor of four while concurrently doubling the frames per second leads to the captured, uncompressed data to grow to 800% of the original. This is exactly the difference seen when comparing 4K UHD footage shot at 60 Frames per second (fps) to an HD clip recorded with 30 fps. Adding 120 fps, which slowly gains utilization, or stereoscopic 3D to the equation, the above example is not even the most extreme combination one could think of.

With virtual reality applications increasingly gaining attention and getting more and more affordable, high quality camera systems that can be used for recording 360° video become relevant. The captured data rates in such scenarios would exceed those of conventional setups by far as well, even without increasing pixel count and image frequency. Although cinema grade cameras currently are optimized for completely different use cases, this is a market that could offer new opportunities for manufacturers, if the data rate issue can be addressed [24, 25].

	1920 × 1080 px		3840 × 2160 px		7680 × 4320 px	
	normal	stereo 3D	normal	stereo 3D	normal	stereo 3D
30 fps	1.5	3.0	6.0	12.0	24.0	48.0
60 fps	3.0	6.0	12.0	24.0	48.0	96.0
120 fps	6.0	12.0	24.0	48.0	96.0	192.0

Table 3.1: Image formats and the nominal SDI rate in Gbit/s needed to transmit them at 10 bit pixel bit depth [20].

Currently, the SDI family of standards struggles to achieve the data rates necessary for the use cases above on the commonly deployed copper media supporting 3 Gbit/s. Even on infrastructures based on the recent multi-link 12 Gbit/s interfaces, transmitting some of the higher data rates in Table 3.1 would need eight links, which would require large amounts of expensive, space consuming wiring and a huge number of interface ports on both switching devices and data sources and sinks like cameras, editing workstations or recorders. The fact that SDI signals are uncompressed allows for great quality and compatibility, since devices do not have to support an additional codec just to support the interface, but these extremely high data rates are also one of the consequences.

While SDI is used to transport live video for monitoring and processing, different data representations are in use for recorded footage. These formats range from lossy, low quality codecs like those of the MPEG-2 family to the more modern H.264 codecs and the comparably high quality Apple ProRes codecs, and even up to raw, lossless sensor data. Most codecs compress the captured data to fit longer clips on one medium, possibly yielding much lower data rates than SDI.

The Sony PMW-F55 for example can record both MPEG-2 at a resolution of 1920 × 1080 pixels and a data rate of 50 Mbit/s, and H.264 encoded 3840 × 2160 pixels video at data rates up to 600 Mbit/s [26]. Data rates like that could easily be transferred over the existing SDI infrastructure, but while visually lossy codecs are currently acceptable for TV productions, they oppose the unimpaired quality needed for monitoring in cinema productions and the like, so lossy compression is not a universal solution. New codecs with low visual loss of detail, such as H.265, could change this, but it will take some time until they become established in the industry.

Remote control and synchronization applications in contrast only require relatively small bandwidths in the 10 to 100 Mbit/s order of magnitude, since packets are small and do not have to be transferred at image frame rate. This will probably remain mostly unchanged, as remote control data does not scale with any of the image parameters. Since usually only a small number of people are allowed to change camera settings on a set for reliability reasons, one can expect the number of users remote controlling the camera to be similarly small.

3.2 Data Flows in various Scenarios

Beyond data rates, the second big challenge that plays a role in interconnecting devices in TV studios, movie sets and the like is handling the wide variety of data streams and interfaces. As illustrated in Chapter 2, there are many completely different applications of networked devices in use on such sites. They all have their own requirements, and commonly have their own communication medium or infrastructure, i.e., SDI wiring for live video, wireless networking for remote controlling and wired IP infrastructure for file-based workflows in studios or broadcast trucks.

That approach has evolved over the years as applications and devices changed, and while it works and has been widely adopted, it is probably far from optimal concerning efficiency. Multiple coexisting infrastructures require discrete hardware for each of them, as well as the knowledge needed to set up and maintain the networks. This is both cost-intensive and requires additional space for devices like SDI switches, splitters or IP routers for each kind of infrastructure.

The above observations raise the question whether maintaining separate infrastructures for the different use cases is actually still desirable, or whether a unified approach could promise lower costs, ease of installation and maintenance, and higher flexibility. Therefore, the networking requirements of the most relevant use cases will be examined in that regard in the following.

3.2.1 Live Monitoring and External Recording

Probably the most common use of interconnecting camera equipment is live monitoring of the recorded material with monitors and electronic viewfinders. In this application, the cameras serve as sources of image data, while the monitors or viewfinders ocular are sinks that receive and display the video. The connection has to provide enough bandwidth to allow transport of the image at the used frame rate and resolution at the highest possible – and therefore uncompressed – quality. In scenarios with multiple cameras, for example the production of a live TV show, all of the cameras must be monitored both by the operators and from a central point where the live image is selected and mixed. In such cases, each camera's image must be sent to multiple displays, and the workstation that creates the mixed live image must be connected to all the cameras as well. That leads to many point-to-point links from camera outputs and downstream splitters (sources) to receiving monitors (sinks). Switching from one source's image to that of another requires time-synchronized circuit switching along the way from the sources to the sink.

Latency is relevant for monitoring with regard to adjusting focus or angle, but only in an order of magnitude of frames, i.e., tens of milliseconds. This is usually not a big

problem for any kind of local connection, as long as there are no high peaks caused by congestion in intermediate devices.

It is also crucial that all data arrives exactly in the order in which it was sent and no samples are lost in transmission, since reordered or missing samples, lines or even frames will lead to a scrambled image or stuttering, respectively [27]. In SDI, these requirements are inherently fulfilled because every link can only carry one standardized signal at a time which transports samples in a fixed order, and the network only consists of circuit-switched point-to-point links. While a signal can be composed of multiple individual video streams, the transmission standard specifies exactly in which order their samples are time-multiplexed, ensuring a known order for demultiplexing.

Recording from monitor outputs to external storage devices is sometimes used because of the ability to record on redundant drives or in additional encodings that the camera does not support directly. This application has the same requirements as monitoring regarding data rate and order, but the number of interconnects is lower, since recorders are tightly coupled to the camera they record from. Also, latency is less relevant as the data is captured frame by frame and not displayed directly, but instead stored whenever it arrives.

As new image formats and features like UHD and HFR arrive, flexibility regarding the transported data becomes an increasingly important factor. While support for new formats in processing and switching units could be added via firmware upgrades in some instances, other additions might require hardware upgrades or replacements, especially if the previously available data rates are exceeded. This observation, coupled with the large variety of consumer display resolutions and capabilities in mobile devices like tablets and smartphones, means that flexibility is needed more than ever before. SDI as a protocol that is defined by the transmitted frame format is not exactly a good fit for this new development.

3.2.2 Return-In

Section 2.2 already mentioned Return In Monitoring, which allows camera operators to compare their current capture with the aired image of a live production or the image of another camera, to coordinate shots. To set this up, the camera has to provide a video input port, to which the video source that should be viewed is connected for feedback. The video input is then routed to a connected monitor within the camera. This is a bidirectional communication on the camera side, since it both sends and receives image data concurrently. In the prevalent SDI infrastructure, bidirectional links are not supported, which means that two physical links are needed (or more for dual or quad link SDI variants) and increases cost, space requirements and complexity of the wiring.

The required bandwidth will in most cases be as high as that of the monitored live image, and therefore possibly very high, depending on the chosen image parameters. For latency and reordering, the same rules apply as for monitoring.

3.2.3 Synchronization

Synchronizing devices on set and in studios is critical, as the clock frequency and phase of image and audio sources and sinks must be aligned to ensure a constantly clean resulting signal when switching and editing multiple sources together [27, 28]. Synchronization is achieved by feeding a dedicated sync signal to all receivers, with well-defined symbols that mark the start of video frames. Without synchronization, color changes or pixel offsets could be introduced, negatively affecting image quality.

This Genlock process requires the reference signal to have the highest possible accuracy and stability, as well as the lowest possible jitter [27]. Hence, care must be taken that the signal accumulates as little jitter and latency as possible while traversing the links from the generator to the synchronized devices. The analog reference Genlock signal is commonly transferred via dedicated copper cables with BNC connectors, the same that are used for SDI.

It would be desirable to also lock other devices than just audio and video sources and sinks to the reference clock. One such example is lighting equipment. Many lights exhibit slight flickering depending on the frequency of their power supply, which might not be noticeable for the human eye, but can be amplified if captured with a camera at certain shutter settings and frame rates. To prevent this, synchronization of image sensor and light could be useful, but this apparently does not justify the addition of BNC plugs and Genlock signal processing to lighting equipment, as it is not commonly seen.

3.2.4 Accessory Communication

As has been stated in Section 3.1, communication with accessories like remote controls, external sensors and the like only has basic demands regarding data rates. Like with monitoring, total latencies shorter than the duration of a frame, e.g., 17 milliseconds at 60 fps, are acceptable. Anything above could prove problematic for applications like remote focus, iris or zoom adjustment, all of which can be timing critical. It is important to note that those transmission latencies get added to other delays occurring within the devices, like the sensor readout time, which is about as long as a frame, or display delays in monitors.

For example, if a camera assistant uses a remote control unit to change the lens's focus in a quickly changing action scene that is shot at 200 fps, a latency of 100 ms from

the remote control to the lens motor would result in 20 frames of out-of-focus footage. If the footage is going to be used for a slow motion effect with a speed factor of 8, where the video is played back at 25 fps instead of the 200 it has been recorded at, the 20 frames of delay become 0.8 seconds, which is clearly noticeable. Such amounts of latency therefore should be prevented.

The example also illustrates that reliability is very important in such use cases, as a lost message basically equals a latency of the amount of time it takes until the user notices that their command has not been followed, and repeats it.

While lighting equipment does not yet have many uses for direct communication, the crew that controls it depends on information such as illumination measurements of the scene to be able to adapt precisely. Such information can be obtained from the monitors connected to the camera via SDI, since displaying the image waveform is a common feature. However, making it available wirelessly, for example for tablets and smartphones, could offer more flexibility for the lighting crew and reduce monitor wiring. Wireless data rates should be less of a problem for this application, because waveform diagrams do not have to be delivered in high quality or frequency to be useful.

The very nature of accessories implies that they come in a wide assortment of brands, implementing various interconnect standards and proprietary protocols. That also includes personal mobile devices like smartphones and tablets, which many people are already comfortable with, because they can access HTTP-based control interfaces via their browsers. This is another dimension of communication flexibility that a set or studio has to deal with, and possibly requires additional adapters or interfaces on the cameras and workstations, especially in conjunction with wireless connections.

3.2.5 Summary

In summary, high bandwidths and low latencies are the challenges for hardware performance, while overall flexibility and interoperability are required on a more abstract level. Extensibility is of particular importance with respect to future video standards that affect parameters such as resolution, frame rate, sample bit depth.

The predominant SDI set of standards, commonly deployed on copper wires, is able to fulfill the requirements for latency and data rates for now, but is already close to its limits [17, 20, 29]. The fact that SDI relies on unidirectional point-to-point links means low flexibility and large switching matrices, and incurs high costs for installing and extending networks. That, coupled with the increasing need for flexibility with regard to both the format of carried data and the topology of a typical environment, suggests that there is a need for change in the way the industry handles communication.

3.3 IP Networks as the Solution

The most promising solution to the networking difficulties that the motion picture and broadcast industries face is the IP suite. Together with Ethernet and sometimes other link layer standards, it is slowly seeing adoption throughout studios as a universal networking standard [27, 29, 30], replacing previously common protocols to an extent, including SDI. Efforts are already being made by a number of companies in the studio equipment sector to intensify the development and acceptance of IP-based communication in the motion picture industry [29, 30].

In the following sections, the impact of the introduction of IP to broadcast and cinema production environments will be analyzed, including its benefits and possible transition mechanisms.

3.3.1 Data Rate

Networks based on Ethernet and IP have seen a steep increase in transmission rates since their introduction and adoption in the 1980s, evolving from 10 Mbit/s to the 100 Gbit/s standardized by IEEE 802.3ba in 2010 [31]. Ranging well into the multiple Gigabit domain, their transmission rates can easily compete with those of SDI, both with 10 Gbit/s on copper and even more on fiber optic media. Just like with SDI as stated in Section 2.4, copper wire length is a problem at higher speeds, therefore optical fiber are preferable for distances over 100 meters, which is the maximum transmission distance supported by 10GBASE-T Ethernet over copper [32]. That range is comparable to what can be accomplished with common SDI wiring [33].

Video payload formats of 6G-SDI and smaller can be transported on a single 10 Gbit/s network link. Multiple transmissions can be consolidated into one link as long as the total data rate does not exceed the link capacity, because Ethernet networks are packet-oriented and therefore can carry multiple connections of different types, in contrast to the format-specific single connection SDI interface. This means the links can be used to their full capacity, because the unused bandwidth between frames of one video stream can be used by arbitrary packets of other streams, which do not even have to carry image data.

While the already existing 100 Gbit/s equipment is still rather expensive, the history of Ethernet standards shows a tendency of quickly increasing adoption and dropping prices [34]. These developments are caused by the wide-spread use of Ethernet and IP networks coupled with the sustained need of the telecommunication industry for even higher data rates because of the growth of Big Data and Internet of Things applications. The recent 40 and 100 Gbit/s standards as well as the Small Form-factor Pluggable (SFP), Quad Small Form-factor Pluggable (QSFP), SFP+ and QSFP+ transceivers are results of

that process, and offer flexibility and extensibility, originally targeted at data centers. However, they also fit the requirements of the motion picture industry, providing higher data rates than the SDI infrastructure.

The evolution of transmission rate is not expected to stop at 100G Ethernet. 400 Gbit/s and 1 Tbit/s are in discussion for possible future standards, with 400G interfaces being a viable option since 2010 [35]. With data rates in those dimensions, video streaming applications in the domain of over 100 Gbit/s, such as those listed in Section 3.1, become possible. The 24 Gbit/s SDI variant that has been in the process of standardization since 2013 is one order of magnitude slower, and there is no estimated date when it will be available [18, 19]. Although a 10 Gbit/s SDI over fiber standard exists [22], it is also one order of magnitude slower than the 100 Gbit/s of Ethernet, and is not widely adopted yet.

3.3.2 Flexibility of Payload

The IP protocol suite offers flexibility in a number of ways. The Internet Protocol has no restrictions regarding the amount of parallel streams on a network segment or the type and size of data that is transmitted, apart from the limits for address and port numbers, but they are high enough to be unproblematic in almost all scenarios.

The fact that the payload content is irrelevant for correct packet delivery allows for any variety of protocols and applications on the same network infrastructure. Connections can be opened and closed at any time without having to respect frame format standards as in SDI, where multiplexing can only happen with streams of fixed bit rates that are certain fractions of the nominal link capacity. This flexibility means that a single infrastructure can be used for all applications within a site, including live monitoring, remote control and synchronization.

An IP-based network also provides extensibility with respect to new applications and image formats. Intermediate devices do not have to implement new video standards but can just forward the packets as they did before, ignoring the fact that the payload might have changed. Additional functionality that depends on higher layer information can be introduced to a network via firmware upgrades or even Software-Defined Networking (SDN), allowing cheap and quick extensibility without impacting the hardware. This could prove useful for the future of the motion picture industry because of the increasing diversity of formats and applications, as has been illustrated in Section 3.1.

File-based data transfers already necessitate IP infrastructure in some studios [30], which could be extended to also carry live video and other communication if the interfaces were unified.

3.3.3 Topological Flexibility

One of the disadvantages of the prevalent circuit switched SDI installations is that they rely on unidirectional point-to-point connections. Such a system requires more wiring than a bidirectional one and is less flexible, because dedicated channels between all communication endpoints have to be provided. Packet switched Ethernet/IP networks in contrast offer more flexibility in terms of topology, because they deliver packets based on the addresses in the headers. This simplifies connections spanning multiple subnets and does not require manual switching. It can also handle different sources and destinations on a single link.

These benefits are further enhanced by the IP multicast routing feature and Software-Defined Networking. Multicast routing makes optimal use of the network infrastructure by ensuring that data that is sent to multiple destinations only passes each network segment once, preserving bandwidth for other traffic. In the scope of this thesis, it is useful to reduce the number of point-to-point connections, e.g., by sending live video to multiple monitors via multicast instead of having to replicate the entire video stream, possibly even on a separate cable. This way, both the necessary bandwidth and the topological complexity can be reduced. SDN helps in a similar fashion because it can provide a global view of the network and automatically configure switches to find short paths with low latency or high available bandwidth.

Concerning device interoperability, the combination of Ethernet and IP enables users to connect to the production network using smartphones, tablets, and other consumer devices implementing the common IEEE 802.11 wireless LAN networking standard. Wireless routers are available as commodity hardware and are constructed to be easy to set up and use, making integration of wireless accessories simple. In an all-IP network, they could connect to services throughout the network without the need to convert between IP-based communication and proprietary protocols to access camera control interfaces or video sources.

3.3.4 SDI over IP

The above sections mentioned a number of reasons that make IP infrastructure attractive for broadcast and cinema productions and studios. However, because of the high costs of replacing an entire infrastructure, the process of switching from SDI to IP-based networks is expected to happen gradually. Such a stepwise change means that there will be a time period where both infrastructures coexist and have to interoperate with each other, as is the case with IPv4 and IPv6.

The SMPTE 2022 set of standards has been devised to allow a smooth transition as well as interoperability between the two technologies. SMPTE 2022-6 [36] defines a protocol stack for transport of an SDI data stream of any standardized SDI data rate,

as well as MPEG-2 encoded video, over an IP network, enabling a transition from SDI infrastructure to IP networking while allowing stepwise upgrades of hardware. This protocol stack, called High Bit Rate Media Transport (HBRMT), relies on the User Datagram Protocol (UDP) on the transport layer and the Real-Time Transport Protocol (RTP) on the application layer. Within RTP, a custom "High Bit Rate Media Payload Header" is added before the SDI payload is transmitted as usual in the form of 10 bit or 12 bit chrominance and luminance samples and ancillary data.

Layer	Header/Content	Size (bytes)
Application	SDI image payload	1376
	HBRMT (no extensions)	8
	RTP	12
Transport	UDP	8
Network	IPv4 (no options)	20
Data Link	MAC (no VLAN tag)	18
Physical	Preamble, Inter-frame Gap	20
	total	1462

Table 3.2: Header sizes in a SMPTE 2022-6 packet [36, 37].

Each IP packet contains 1376 bytes of SDI payload, which is padded to that size in the last packet of a frame to align the beginning of a frame with the beginning of a packet. When transmitting via IPv4 on Ethernet, a minimum total overhead of 86 bytes (in case of no header options) is added by the preamble, inter-frame gap and headers, which equals 6.25 %, or 1/16, of the payload length, giving a total packet size of 1462 bytes. That overhead allows three 3G-SDI streams or one 3G-SDI stream and one 6G-SDI stream to be transmitted over a 10 Gbit/s Ethernet link.

To simplify the implementation of HBRMT, no additional communication beyond the main RTP stream should be required between a source and a sink [36]. Because UDP and IP do not provide any mechanism for reliable packet delivery, SMPTE 2022 also defines an optional forward error correction scheme that sends error correcting codes in separate RTP streams and can conceal errors in case packets of the main stream are lost [38].

3.3.5 Time Synchronization

Synchronizing equipment throughout a site is a challenge for packet switched networks because intermediate devices such as switches might add variable latency and jitter, which reduces the accuracy of the synchronization. However, this jitter can be reduced by relying on synchronized clocks within the network, instead of using unidirectional synchronization pulses from one master [27]. The Precision Time Protocol (PTP) oper-

ating at the SMPTE PTP profile for use in professional broadcast applications offers an accuracy of “within 1 microsecond” [39].

3.3.6 Quality of Service

The IP suite and the SDI standards have been developed for different requirements and use cases, and consequently there are performance differences between them. As video production environments rely on real-time applications such as monitoring and remote control, low latency and high reliability of transmissions are crucial for smooth operation. While these requirements are naturally built into the SDI protocol, they are not fulfilled by default in Ethernet, IP, and UDP, because they are not relevant in all of the use cases of these protocols.

In IP networking, there are two main approaches of dealing with such Quality of Service (QoS) requirements: Differentiated Services (DiffServ) and Integrated Services (IntServ). IntServ provides fine-tunable QoS parameters, while DiffServ only offers a fixed number of classes of service. DiffServ is used more widely because of the fact that the classes of service it provides are sufficient for most applications, and because of the higher complexity of implementing IntServ that comes from the necessity to hold state in intermediate devices.

DiffServ recommends some widely used classes and Per-Hop Behaviors (PHBs) that represent traffic that should be optimized for low latency, low jitter, low packet loss, and combinations of these attributes [40]. The Expedited Forwarding PHB aims to provide all of these characteristics [41], which means it is suited to be used for live monitoring traffic and synchronization packets.

Since studios are closed environments where the entire network can be controlled, DiffServ is easy to deploy because all switches and routers can be set up to handle PHBs the same way. While IntServ could be used too, it is more complex and has more overhead, reducing the available net bandwidth, which is contrary to the goal of offering higher data rates for new formats. IntServ would also require the networking subsystem of the camera to hold state for all IntServ reservations and to be able to set up and tear down those reservations, which increases the complexity of the system unnecessarily. However, it might prove useful for streams from one remote site to another.

The unreliable nature of IP networks is problematic for real-time applications. While the Transmission Control Protocol (TCP) offers reliability on top of IP, it is based on retransmission of lost packets, which introduces latency in case of lost and reordered packets, and therefore contradicts the requirements of monitoring and remote control applications, which requires low latencies. For video transmission, the forward error correction (FEC) of SMPTE 2022 on top of UDP and RTP is more fit to ensure reliable communication, since it is tailored to the needs of the application, and does not have the

aforementioned negative effects. FEC does however come with the cost of bandwidth overhead caused by the additional packets that are sent. This overhead can be significant, depending on the amount of redundancy that is required [38].

Packet loss of real-time applications caused by congestion can mostly be prevented via DiffServ markings, ensuring that high-priority data is prioritized. As the peak data rates of video applications are known upfront because of the selected image format and codec, and transfer of uncompressed video is expected to be the use case with the highest data rate requirements, the network infrastructure can be designed to handle those rates plus a certain amount of background traffic such as file-transfers and remote control applications. This guarantees that congestion does not happen during regular operation, and therefore packet loss should be minimal.

To increase reliability, SDI networks frequently have redundant infrastructure which can take over in case the main system fails. Especially in live broadcasting scenarios, it is common practice to deploy an addtional link to combat the calamity of losing the live signal. Downstream devices must be able to switch between the original system and the backup without any visual impact on the video quality. In SDI, this process, called seamless protection switching, works by performing the switch in between active video frames. When transmitting SDI via IP, the same process can be used, but to extend it to integrate multicast routing and compensate the Ethernet latency jitter, multicast source address relabeling of packets by SDI over IP aware switches could be a viable approach [37].

3.3.7 Internal IP Switch

To leverage all of the above benefits in a professional video production environment, the network should to an extent be centered on the cameras, because they are the main source of video content and most of the accessories are connected – either directly or indirectly – to a camera. For cameras to become a central network component, they have to be able to communicate with all devices and relay communication between them, essentially requiring a network switch to be integrated into the camera.

An in-camera switch can serve as a unified communication interface between monitors, accessories and the camera, similar to how some camera models already provide power to attached components via a USB interface. It can transparently forward information from sensors to motors and from video stream sources to sinks based on protocols and addresses, while concurrently ensuring the necessary QoS for the various applications.

Being integrated into the camera allows the switch to connect directly to the image processing chain, removing latency otherwise caused by external converters. Wireless network segments can easily be incorporated with all existing applications, making even wireless video streaming directly from within the camera body possible.

3.4 Alternatives

Moving to an Ethernet/IP architecture is not the only solution that has been proposed for the difficulties faced by the industry. Especially for the problem of increasing data rates, multiple approaches are being researched and tested, with varying success.

The SDI standards have grown from the initial 270 Mbit/s to 12 Gbit/s, driven by the need of supporting higher resolutions and frame rates. To stay compatible with installed hardware, multi-link solutions have been employed to multiply the effective bandwidth without having to replace the existing infrastructure. This is still a valid approach, as dual and quad 12G-SDI can tackle the most urgent bandwidth problems [20]. However, multi-link SDI over coaxial copper cables consumes space and has high weight because of the added wiring, and the fact that no 24 Gbit/s variant has been standardized yet casts doubt on whether or not this really is a future-proof concept. In addition, it does not offer any of the flexibility or the lower complexity that an IP solution could provide.

There is also a trend of transmitting SDI signals over fiber optic media instead of over coaxial copper cables, because fiber can achieve higher data rates and distances at a fraction of the weight and diameter [22]. The same kind of fiber can be used with Ethernet and with SDI, but each physical link can only carry either of them, essentially still requiring two parallel infrastructures. As with SDI on copper cables, no flexibility is gained, although potential future data rates are higher because the medium benefits from advancements made in the communication and networking industries. As stated in Section 2.4, the lower robustness of optical fiber could pose a problem on movie sets.

Another way of handling the increase in video data is the introduction of visually lossless encoding. Wavelet-based codecs offer low compression ratios of around 5:1 at very low latencies. Recently standardized or proposed light codecs include Low Latency Video Codec, Tiny Codec and VC-2. Since the SDI standards transmit only uncompressed data, they are not directly compatible with encoded video, but would instead have to rely on variants of the Serial Data Transport Interface (SDTI) as a wrapping mechanism. Lightweight encoding can be combined with Ethernet transmission as well and could become a key factor in the transition from SDI to IP, because a 4K video stream at 60 fps requires 12 Gbit/s (see Section 3.1), which does not readily fit into a 10 Gbit/s Ethernet link [30].

RED cameras follow a modular approach where users can choose between multiple interfacing modules that can be attached to a generic interface on the camera body [42]. While this does offer flexibility regarding communication standards, the need to choose between a limited number of interfaces remains. The infrastructure cannot be shared either, except if a 10 Gbit/s Ethernet monitoring module would be made available, combining the modular approach with Ethernet/IP.

Overall, using IP as the universal communication standard seems the most promising, because it adds flexibility in ways that other solutions cannot provide, and promises performance that is at least comparable to the prevalent standards in other regards.

Chapter 4

Approach

As the previous chapters have shown, integrating Ethernet/IP communication into professional motion picture cameras is promising. As part of this thesis, a prototype system has been devised to demonstrate and evaluate the idea of a 10 Gbit/s network switch built into such a camera. This chapter will describe the system design and the implementation process of the prototype.

4.1 Architecture Design

Section 2.1 mentioned the two kinds of hardware used to form the core of a camera system, FPGAs and ASICs. FPGAs offer shorter design iteration cycles compared to ASICs because the design can be loaded onto a reconfigurable piece of hardware. Design development and fulfilling timing requirements is also easier on FPGAs. While the price for a high performance FPGA is not cheap, small quantities are somewhat affordable compared to producing ASIC hardware samples, which is too expensive for prototyping in a research project of such a small scale. In addition, in systems based on FPGAs, new image formats can be adopted without replacing hardware. Considering the quickly growing number of formats discussed in Chapter 3, the flexibility of FPGAs is expected to be beneficial in the future. Therefore, the FPGA variant has been selected for building the prototype.

4.1.1 Overview

On an abstract level, the networking subsystem must interconnect the modules within the camera, namely the CPU and the image processing chain, with external devices, such as monitors, remote control units or other cameras. A multilayer switch serves as the central component to which all other parts are connected. Figure 4.1 shows a block diagram of this architecture.

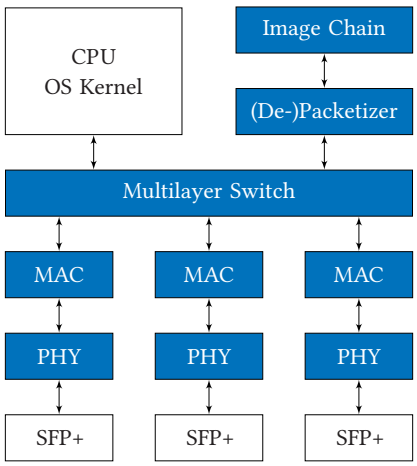

Figure 4.1: Block diagram of the proposed architecture. Blue blocks are located inside the FPGA.

The multilayer switch offers five ports, two of which are occupied by the image processing chain and the CPU. The other three are available for external connections. This configuration supports video streaming directly from and to the image chain for external monitoring and return-in functionality. The switch also allows communication between the camera software running on the CPU, including the user interface and the recording settings, and external control devices. External devices may even interact with each other transparently via the switch, which facilitates applications such as a brightness sensor that automatically adjusts the lens aperture using motors.

The data rate of 10 Gbit/s has been selected as target because hardware supporting that rate is available commercially off-the-shelf, and it represents an improvement over the 3 Gbit/s and 6 Gbit/s SDI interfaces currently common in professional cameras. Layer 1 and layer 2 functionality for this speed is also provided by Intellectual Property Cores (IP cores) for various FPGAs that support the data rate.

4.1.2 Internal Ports

The two internal ports connect the image processing chain within the FPGA and the operating system running on the CPU to the switch. A single port combined with a multiplexer could have been used instead, reducing the amount of FPGA resources used or freeing up one port for external use, respectively. The solution with two internal ports has been chosen to save programming work and to reduce the complexity and thus the number of possible error sources. The multiplexer would also have to look at destination addresses on the MAC and IP layer, basically implementing a second switching module. Instead, the switch handles all packet forwarding decisions.

The operating system running on the CPU accesses the switch by using a regular network interface in its networking stack that is connected to the FPGA memory via

Direct Memory Access (DMA). The operating system sends/receives layer 2 frames to or from that interface, which are accepted/output by the switch, so that no additional adapter in-between is required.

The same cannot be said about the image processing chain because in current cameras there are no existing mechanisms to transform video monitoring output into IP packets. The SDI video output is a continuous data stream consisting of video frames that are larger than the regular maximum frame size in Ethernet. Therefore, a special kind of adapter called Packetizer is inserted at this point in the architecture. It takes video data from the monitoring output of the image processing modules within the FPGA, slices it into packets of a fixed size and prepends all necessary headers. In the case of HBRMT, the payload size would be 1376 bytes, and headers for Ethernet, IP, UDP, RTP and HBRMT would have to be added.

For return-in monitoring, where a video signal is fed into the camera from outside, the reverse functionality is needed, where HBRMT packets are transformed back into an SDI stream. This process is performed by a Depacketizer module which is inverse to the Packetizer module. It takes a stream of IP packets and removes the headers until only the payload is left, which in the case of HBRMT is exactly the SDI signal, and hands that over to the image processing chain.

Many available cameras already contain a wireless communication module for remote control applications, which is connected to the CPU and handled by the operating system. To allow communication between wirelessly connected devices and devices connected to the switch via cables, the operating system has to be able to forward packets between the WiFi module and the switch. Alternatively, the switch has to provide an additional port that connects to a wireless networking module. For accessories that only communicate with the camera software running on the CPU, such as remote control units that change camera-internal settings, no forwarding from the CPU to the switch is necessary because such applications do not require a connection to remote devices connected to the switch.

4.1.3 External Ports

The external ports are proposed to be realized in the form of SFP+ slots because they provide flexibility in terms of the chosen physical medium, are easy to integrate into FPGA-based platforms, and have small dimensions, which helps to keep cameras compact. SFP+ transceivers supporting 10 Gbit/s Ethernet are readily available, both for copper media and optical fiber. The SFP form factor is already the de facto industry standard for SDI via optical media [20].

The number of ports that are available from outside of the camera can be chosen arbitrarily in theory, but has been set to three because this number allows for the most

common use cases, while not taking up too much surface area on the camera body. With three ports, one can be used as a monitoring and recording uplink and for return-in, while the other two can be connected to accessories, such as motors, sensors or local monitors.

As an alternative, one of the ports can be used to attach an additional switch or wireless access point to connect as many devices as necessary, regardless of the number of hardware ports on the camera. In this case, care must be taken to ensure that the latency and bandwidth requirements of the applications are met, e.g., by using a QoS-aware device.

4.1.4 Multilayer Switch

Because of the need for QoS mechanisms implemented within the switch, a pure Ethernet switch does not provide all of the functionality that is necessary to fulfill the requirements of live monitoring applications. Instead, a multilayer switch or layer 3 switch is needed, which is able to provide QoS based on protocols, addresses and DiffServ code points, since the latter will be used to enforce QoS requirements.

To use the medium to its full extent, the switch must be able to forward traffic at rates of up to 10 Gbit/s on each port separately, dropping packets only if the egress port they are destined for is congested. In that case, packets with a higher priority, especially those with a DiffServ code point representing the Expedited Forwarding (EF) Per-Hop Behavior, should be preferred, and should only be dropped if there is more than 10 Gbit/s of EF-labeled traffic being queued to the same port.

To conserve bandwidth, the switch must learn the source MAC addresses of connected devices so that it does not have to broadcast packets on links where they are not needed. For video streams to monitors or recorders that only passively receive video, the user interface of the camera might need to install forwarding entries into the switch depending on user input, so that the switch knows on which port to forward the video data.

4.2 Prototype Implementation

A prototype has been implemented based on the devised system architecture. To be able to create an FPGA implementation within the scope of this thesis, some simplifications and abstractions were necessary. The prototype includes neither the CPU part nor the image processing chain of a camera. However, a module that can generate randomized, incrementing or fixed pattern test data at up to 10 Gbit/s has been put in the place of the image chain, which allows testing of the video payload packet wrapping mechanism

implemented in the Packetizer module. The inverse process was tested with a simple monitoring module which receives the unwrapped payload on the reverse path.

Furthermore, the prototype features a multilayer switch and two external hardware ports that connect to SFP+ slots. The reduced number of ports is sufficient as a proof of concept and allows for a decent variety of performance measurements.

In the following, some of the implementation details and design choices will be explained. The resulting system design is illustrated in Figure 4.2. The implementation is based on a design example generated for the IP core that implements the link layer. The example has been modified and extended with modules written in the Verilog and VHDL hardware description languages (HDLs), and can be both simulated and synthesized to run on real hardware.

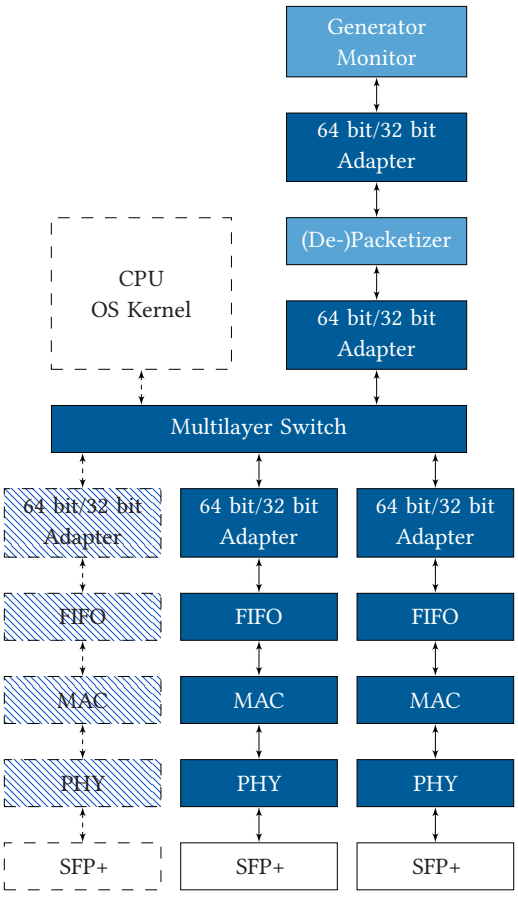

Figure 4.2: Functional block diagram of the developed prototype. Blue blocks are located inside the FPGA. Dark blue modules are realized as IP cores. Dashed blocks have been omitted in the implementation.

It should be noted that whenever the width of an interface is mentioned, this width equals the amount of bits of data that can be transferred in one cycle, excluding any bus control signals and error flags. For example, the bus between each MAC and PHY pair

is 64 bit wide, i.e., it can transfer 64 bit in every clock cycle. However, the interface also comprises start and end of packet flags, error flags and a signal that states how many bytes of the data bus are actually carrying valid data. These signals are not included in the 64 bit width.

4.2.1 Hardware

The Altera/Intel Arria 10 family of FPGAs has been chosen as the hardware platform for the prototype. The Arria 10 series is a recent midrange platform, offering computational power that is slightly above that of the Stratix V FPGAs used in ARRI Amira cameras. The specific board used is an Attila Instant-DevKit by ReFLEX CES[1] and can be seen in Figure 4.3. It is a development kit designed to showcase the features of the Arria 10 platform, although most of them, such as the LCD display, the PCIe interface, and the onboard Ethernet adapter were not used for the experiments performed as part of this thesis.

The Attila board has been extended with a Hitech Global[2] FPGA Mezzanine Card (FMC) featuring two SFP+ ports and one QSFP+ port. The FMC has been connected to the FMC connector of the ReFLEX CES board to provide the two SFP+ slots necessary for the experiments.

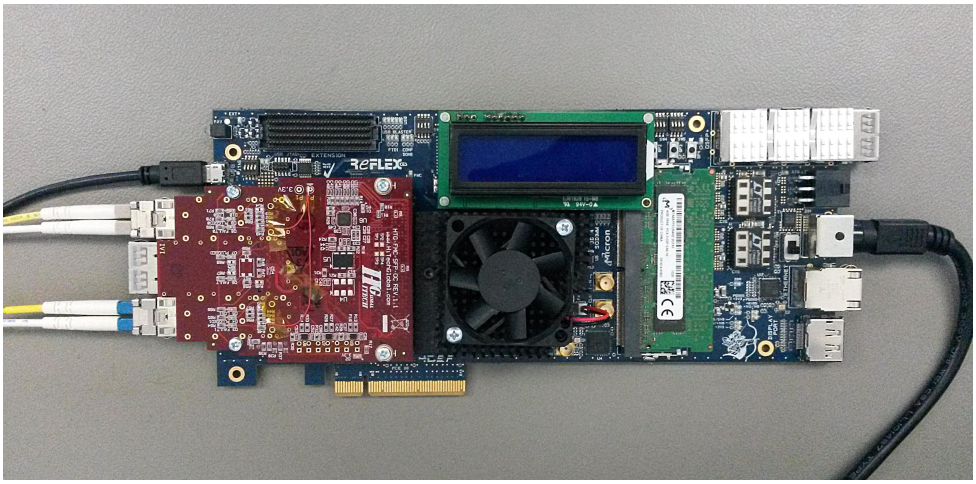

Figure 4.3: The ReFLEX CES development board (blue, background) with the attached extension card (red) and two inserted SFP+ transceivers with optical fibers plugged in (left side). The black wires provide power and a USB interface for configuration and debugging.

Multiple SFP+ transceivers were used, most of which connect to fiber optic media, while one kind supports only twisted pair copper cables with RJ45 connectors. All of them

[1] `https://www.reflexces.com/products-solutions/development-kits/arria-10/attila-instant-devkit-arria-10-fpga-fmc-idk`, accessed June 30, 2017

[2] `http://www.hitechglobal.com/FMCModules/FMC_QSFP+.htm`, accessed July 10, 2017

were compatible to the prototype board. However, multi-mode optical transceivers were used most of the time because the twisted pair transceivers were not compatible with the network interface cards (NICs) of the machines that were used as external source and sink in experiments.

FPGA designs can be loaded onto the board using the USB configuration and debugging interface. Apart from configuration, it can be used to read and write any registers within the design that have been connected to the interface.

4.2.2 Base-R PHY

The physical layer has been implemented using an Altera/Intel IP core. A number of IP cores and subvarieties are available, with varying sets of features, including the ability to switch between data rates in the range from 10 Mbit/s to 10 Gbit/s and different Physical Coding Sublayer (PCS) encoding schemes such as 64b/66b or 8b/10b.

The 10GBASE-R variant of the Arria 10 Transceiver Native PHY IP core was selected because it offers a latency that is 140 ns lower compared to other PHYs, and occupies the smallest amount of FPGA resources of all the versions that have been compared. More information on resource usage will be provided in Section 5.2.5. The 10GBASE-R PCS and the 64b/66b coding it employs are described in IEEE 802.3 [43].

The 10GBASE-R PHY connects to the MAC via a 64 bit wide 10 Gigabit Media Independent Interface (XGMII). The chosen configuration of the 10GBASE-R PHY IP core only allows for transmission at 10 Gbit/s, which is acceptable because certain copper SFP+ transceivers that support scaling down the wire speed can be used outside of the FPGA, while the connection from the internal PHY to the SFP+ transmits pause signals at full speed to lower the effective data rate. This solution has the advantage that both the PHY and the MAC become less complex and require fewer resources on the FPGA than an approach that changes the data rate throughout the design.

The PHY reference clock has to run at a frequency of 161.1328125 MHz to reach a net data rate of 10 Gbit/s when using a 64 bit interface toward the link layer, i.e., 64 bit are sent in every clock cycle. The required frequency can be calculated as follows, where f_{ref} is the reference clock frequency, R_{net} is the targeted net data rate, r_{coding} is the encoding ratio and $w_{\text{interface}}$ is the number of bits sent per clock cycle:

$$f_{\text{ref}} = \frac{R_{\text{net}}}{r_{\text{coding}} \cdot w_{\text{interface}}} = \frac{10 \cdot 10^9 \frac{\text{bit}}{\text{s}}}{\frac{64\,\text{bit}}{66\,\text{bit}} \cdot 64\,\text{bit}} = 161.132\,812\,5\,\text{MHz} \tag{4.1}$$

4.2.3 Clock Configuration

The ReFLEX CES Attila board contains a configurable clock generator that provides several clocks that can be used by the Arria 10 FPGA. However, the frequency of 161.1328125 MHz is required as the reference clock for the 10GBASE-R PHY, and is not available by default. For modules with 32 bit data bus widths that are part of the data path, the necessary frequency is doubled because only half the amount of data can be processed per cycle, therefore requiring two times as many cycles to process the same data. This results in a clock frequency of 322.265625 MHz, which is used in the Packetizer and MAC modules, and is not available by default either.

Unlike other boards, e.g., evaluation boards built directly by Altera/Intel, the clock generator of the ReFLEX CES board cannot be configured externally via the USB interface. Instead, a memory initialization file describing the desired configuration has to be generated on a PC using multiple proprietary tools. This file must be integrated into the design to load it into the FPGA's memory during initialization. From there, it is transferred to the clock generator via an I^2C bus and the configuration is activated with a timed trigger, changing the clock frequency.

However, this process takes so long that the external transceiver Phase-Locked Loops (PLLs), which use the reference clock to generate the serial transceiver clock, lock to the default clock of the wrong frequency before the clock configuration is complete. The user guide of the PHY IP core describes a recalibration procedure that can be used to lock the PLLs to a new frequency. The process has been implemented and tested, but was unsuccessful. Therefore, an alternative approach has been developed.

After the clock configuration is complete, the user is informed about it via LEDs and has to press a button which causes the entire configuration of the FPGA to be cleared and loaded from the on-board flash memory once again. This does not affect the clock generator however, which only resets on power-down. Therefore, the PLLs will calibrate to the correct frequency after the reconfiguration, allowing correct operation. For use beyond experimental tests, the button press could be replaced by a mechanism which automatically detects whether or not the clock frequency is correct, and only triggers the reconfiguration if the clock runs at the default frequency.

FPGA-internal PLLs, in contrast to the external transceiver PLLs, cannot be used to generate the reference clock from the default clock, because they introduce jitter, which is contrary to the low jitter requirements of the transceivers. Otherwise, using an internal PLL would have been a simple way to solve this problem.

In hindsight, it might have been easier to use a different PHY variant that does not employ 64b/66b coding instead of the 10GBASE-R PHY. In that case, the required reference clock frequency would have been 156.25 MHz, which is available on the board by default. Unfortunately, changing the frequency was not expected to be this complicated, as it

is not an uncommon task, which is why the amount of work that was necessary was underestimated in the beginning. Otherwise, using a different PHY configuration would probably have been the preferable solution, especially since low resource consumption is not essential for a prototype, and SFP+ modules that support multiple client side PCS variants are available.

4.2.4 MAC IP Core

Just like the PHY, the link layer modules (MACs) are based on an IP core provided by Altera/Intel. It communicates with the physical layer using the 64 bit wide XGMII interface, while the higher levels can send and receive packets using the MAC via a 32 bit data bus. The MAC core runs at the aforementioned 322.265625 MHz

The MAC instances handle the CRC32 checksum calculation and insertion for all frames that are about to be transmitted. They also calculate the checksum of all incoming frames and compare them to the value found at the end of each frame, signaling invalid frames to the upper layers. For checksum insertion, the higher layers are required to pause for one cycle whenever the end of a packet has been handed to the MAC, which allows the MAC to transmit the checksum it has calculated.

For all received and sent frames, the MAC core records statistics, such as whether the frame was oversized or undersized, broken or valid. Among other statistics, it also counts the numbers of frames that exhibit minimum and maximum payload sizes, as well as the number of unicast, multicast and broadcast frames. These statistics are helpful for debugging as well as performance measurements, because one can analyze the reasons for broken frames based on the counters and it is possible to ensure that sent frames have correct sizes and checksums. These statistics can be accessed by reading register values using the USB debugging port.

4.2.5 IPv4 versus IPv6

Whether IPv4 or IPv6 is used in communication is not relevant for the prototype as long as all modules support the respective version, because they essentially offer the same functionality. Both could also be used in parallel if all modules are design to work with both versions at the same time, which should not be difficult to implement because the version field in the IPv4 and IPv6 header is located in the same byte, allowing dynamic parsing depending on the field's value.

IPv4 has been selected for the implementation of the prototype, because accessories are assumed to better support it compared to IPv6, since IPv4 is more widely used in general as well. Additionally, the minimum header size is only 20 bytes in IPv4 compared to

40 in IPv6, which means that IPv4 offers a lower overhead compared to IPv6 at fixed packet sizes.

For measurement purposes, IPv6 can be simulated by adding the missing header length to the length of the payload. While the processing and forwarding is still based on IPv4, the impact of the higher overhead on latency and throughput can be measured this way.

4.2.6 Test Data Generator

To simulate video being streamed from the image chain of the camera, a test data generator is necessary within the FPGA design. The generator that was used is a modified version of the layer 2 frame generator found in one of the design examples that can be generated for the MAC IP core. It is a module written in the Verilog hardware description language that generates packets of variable size and has been extended for some additional functionality. It offers a 64 bit interface that was connected to the 32 bit interface of the MAC core using a 64 bit to 32 bit adapter in the original example design.

Since the generator was meant to create layer 2 frames, it offers configurable registers for the source and destination MAC addresses. For the prototype, the generator was used to create the payload of the layer 3 packets, i.e., UDP datagrams or ICMP packets. Therefore, the address registers were reinterpreted as layer 4 header fields. The generation process does not try to mimic any of the characteristics of real image payload content, except for the fact that UDP is used as the transport layer protocol.

However, the generator can be seen as the prototype equivalent to the module that slices video frames into payload chunks of a fixed size. The test data generation process and the slicing differ in the produced output, but the procedure is essentially the same. Depending on the internal state of the module and a progress counter, the output of the module is either one of multiple predefined values such as UDP ports or RTP header fields, or it is set to the current value of the input data field, which contains either video data or test data. The video slicing module could be evolved from the generator by providing it with real image data and extending it so that it pads any packets that contain the end of a frame to the correct length.

In addition to the already available options of using incrementing byte values and randomized data as payload content, a fixed pattern option has been implemented that allows to construct identical packets, which allows for static checksums in the layer 4 header. Another added feature is the ability to control the rate at which packets are generated by defining the number of clock cycles to pause generation after the end of each packet. This does not allow all rates to be selected, but it is significantly easier to implement than using a period length and the amount of active cycles per period,

while offering a control mechanism that is fine-grained enough to hit data rates such as 3 Gbit/s at realistic HBRMT packet sizes.

In the receive path, there is a simple monitor module in the location corresponding to that of the generator that counts any received payload units. The counter can be read via the USB register interface.

4.2.7 Packet Wrapping Adapters

As has been mentioned in Section 4.1.2, the design contains an adapter between the image processing chain (or generator in case of the prototype) and the switch, which wraps payload data in Ethernet, IP, and possibly other headers to allow the image data to be transported on the IP network. This adapter as well as its inverted function have been implemented as VHDL modules, called Packetizer and Depacketizer.

The modules have 32 bit wide interfaces toward the switch and toward the generator and packet monitor. The decision to use 32 bit was made because the initial specification of the switch IP core was based on a 32 bit packet interface, just like the MAC core (see Section 4.2.4).

The functionality of the Packetizer module is illustrated as a waveform diagram in Figure 4.4. The payload created by the generator is received as input in units of 32 bit per cycle. The first bytes of the Ethernet header are output two cycles later, followed by the remaining bytes and the IP header. The payload is being buffered internally until the headers have been sent. In the minimum case of 14 bytes of Ethernet header and 20 bytes of IP header, the insertion requires nine clock cycles, although the module already sends two bytes of payload in the ninth cycle.

The headers might not have lengths that are multiples of the bus width, which is why the Packetizer has to fill the remaining bytes of the output bus with the beginning of the next header or the payload in such cases. This can also be seen in Figure 4.4, where the value of a signal has two colors during one clock cycle.

For simplification, only the Ethernet and IP headers have been implemented in the Packetizer. Some transport layer fields can be configured in the test data generator, but not the entire HBRMT protocol stack is supported by this prototype. However, this does not matter unless an experiment actually includes HBRMT or RTP aware devices, because the higher layer protocols are handled transparently by all other hardware and thus they can be simulated by adding their length to the length of the application layer payload. No such HBRMT aware devices were employed in the scope of this thesis, therefore this change should not have any effects on measurement results.

The Packetizer module is based on the assumption that most of the IPv4 header fields are static values that only change rarely. This approach is adopted from a paper written

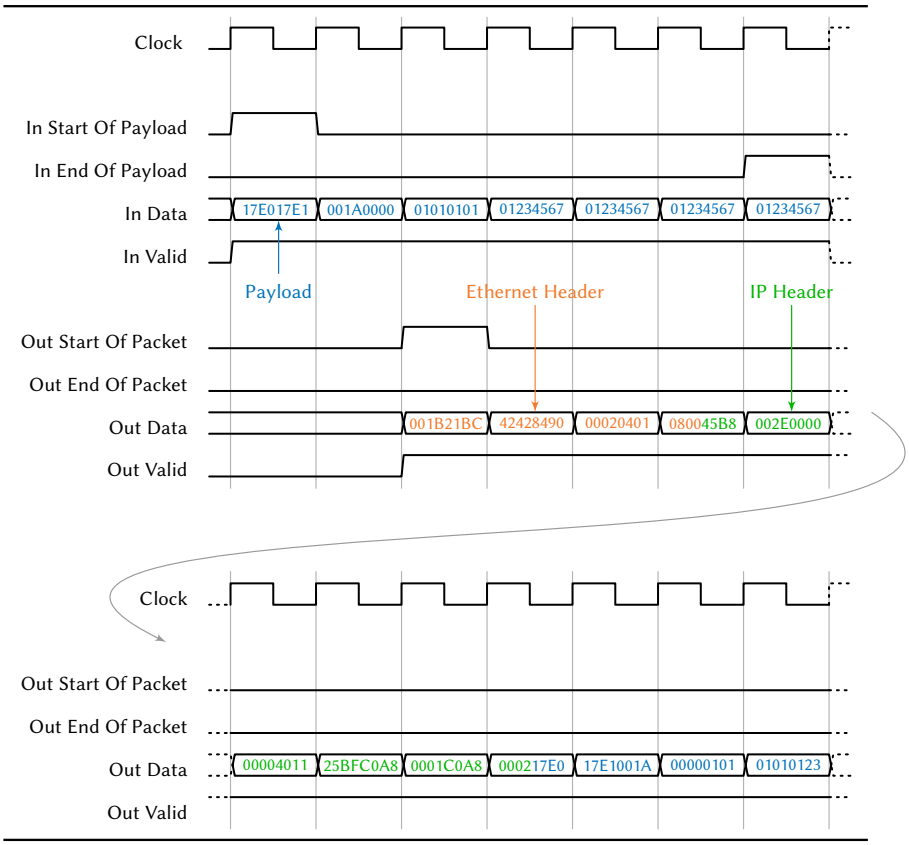

Figure 4.4: The waveform produced by the Packetizer module from a block of payload input, showing the beginning of the packet and the transition from the IP header to the payload. Blue bytes are the payload coming from the generator. Orange bytes are part of the Ethernet header. Green bytes are part of the IP header.

by Alachiotis et al. [44]. However, for the prototype in this thesis, the idea of a static Look Up Table has been realized in combination with manual configuration instead of taking the values from a packet received from another device.

The input data represents a video stream with a fixed source, a fixed destination (possibly a multicast address), and static packet sizes, e.g., 1376 bytes. The EtherType, IP version, layer 3 header length, and layer 4 protocol fields are also static, just like the fragment fields, which are not going to be used. The Time to Live can be chosen arbitrarily by the source of a packet, and therefore can be set to any static value by the Packetizer, e.g. 64. The DiffServ code point can be chosen arbitrarily as well and does not change during the existence of a stream. The Explicit Congestion Notification is optional and therefore can be set to zero. The checksum field is derived from the other values, and only has to be changed if they are modified.

The combined Ethernet and IP header is stored within the module as a VHDL record

data structure. All header fields that are not completely fixed to a single value, such as the EtherType or the fragment fields, can be configured via the configuration interface of the Packetizer module, which is accessible from the USB debug interface and can be accessed by any module on the FPGA, as well as the CPU and operating system in a production design. The correct IP checksum must be calculated and configured as well, which is reasonable because in the real system, the camera software running on the CPU would configure the header fields and could thus also calculate the correct checksum whenever a new stream, e.g., with a new destination, is created. The 32 bit CRC checksum following the Ethernet frame is omitted by the Packetizer, because the switch IP core does not require it, and the MAC core will calculate and insert it before handing the packets to the physical layer.

This solution allows for high flexibility regarding the header fields of the generated packet stream at a reasonable design complexity. While changes to fields of the IP header during operation create invalid packets until the correct checksum is supplied, this is not expected to be necessary, since destination switches can be performed by the multilayer switch via VLAN labels or multicast addresses. Changes in the MAC layer are supported during operation, because the checksum is dynamically calculated by the MAC.

The resulting header data structure can be accessed as a bit signal vector, so that the wrapping process can output the contents of the header in the correct order without having to give special attention to field boundaries. Instead, it just counts how many bits of the headers have been sent already, and outputs the following 32 bit, regardless of their meaning.

This implementation has two important advantages, apart from the ease of implementation. First, the latency added by the module is static for the entire flow, because the time taken to serialize the headers only changes if their length changes, which requires the configuration to change and therefore results in a new flow. Second, it can be easily extended to include more headers, such as the RTP header, by adding the fields of the header to the record data structure and adjusting the length of the headers that should be prepended by the Packetizer module.

As has been mentioned in Section 4.2.6, the process of splitting the continuous video payload into chunks of equal size is not performed by the Packetizer, but is handled by the generator in the prototype implementation.

The Depacketizer module removes all headers from a packet and only outputs the raw payload, effectively inverting the Packetizer process. It can be modeled by a simple state machine that differentiates between receiving header bytes, payload bytes, frame checksum bytes or invalid bytes (no input). Payload bytes are output without any further processing, while all other content is dropped.

The total length of the headers that should be dropped, starting from the beginning of the packet, is a configurable register that has to be set to the combined header lengths to ensure correct output. This was sufficient for experiments, but could be extended for a real application by defining a minimum length of 4 bit, which is enough to scan the IP version field of the packet. Depending on that, the length of the IP header can be detected, and those of higher level protocols can be found and summed up in a similar fashion. This would allow automatic handling of any length of accumulated headers.

4.2.8 Switch IP Core

Building a multilayer switch with QoS mechanisms in hardware is a complex and difficult task. Because of that, realistic options to implement the switch are either a hard IP core ASIC or a soft IP core module in the FPGA. Leaving aside the economic arguments for ASICs and FPGAs, realizing the switch as a soft IP core offers greater flexibility and extensibility through firmware upgrades, as well as easier integration, because the FPGA approach has been selected for the overall architecture. Thus, the multilayer switch has been implemented as a soft IP core within the FPGA.

The feature set provided by the IP core can be customized to fit the requirements of the specific application. The resulting variant is then automatically generated in the form of Verilog code. Customizable parameters include the number of switch ports and queues per port, the data rate supported by the ports, the amount of memory that can be used to queue packets, and the clock frequency at which the core is running.

The switch core variant that is used in the prototype offers a 64 bit wide data interface for each port. However, the initial specification of the variant that was going to be used reported an interface width of 32 bit. The change in width was caused by the difficulty of meeting the timing requirements for the 32 bit interface according to the vendor, since the transfers have to happen at twice the frequency to compensate for the narrow bus. The Packetizer module had already been developed under the expectation of a 32 bit interface, and changing the width requires drastic changes to the underlying code. Therefore, adapters have been inserted between the switch and the (De-)Packetizer as well as the MAC cores which match the buses of differing widths. The adapters are instances of Altera/Intel IP cores which were used in the design example of the MAC core between the 64 bit generator and the 32 bit MAC interface.

Integrating the switch into a design consisting of MACs, PHYs, the packet wrapping modules and the generator proved unexpectedly difficult. A data sheet was available for the switch IP core, but it was missing details on how exactly the packet interfaces are to be connected to the MACs. Because no details were given and the MAC core converts the Big Endian output by the PHY core into Little Endian before handing it to upper layers, it was assumed that the switch expects input data to be ordered in Little Endian, which would have been compatible with the MACs and the (De-)Packetizer.

When connecting the switch with the other modules this way, the packets that reach the switch are scrambled, and are dropped before being queued because the IP header checksum is not located where it should be after the order reversal. Once identified, the problem was easily solved by rewiring the signals between the switch and the other modules to revert the reordering performed by the MACs.

Another difficulty was that the core of the switch module internally runs with a clock of 135 MHz, while the packet interfaces toward the MACs and the (De-)Packetizer are clocked at 161.1328125 MHz. This mismatch requires careful synchronization between these clock domains, which was not working correctly in the initial version of the switch IP core that was used. As a consequence, severe timing issues occurred at the edge of the clock domains, leading to unexpected behavior of the switch module. After communication with the vendor, the timing was fixed in a new version of the IP core.

Corresponding to the proposed system architecture shown in Figure 4.1, the IP core variant that has been selected possesses five ports. However, out of those five, only three ports were used in the prototype design because of hardware limitations, as can be seen in Figure 4.2. The other two ports remained unconnected.

Each of the ports features eight queues to which a strict priority scheduler queues egress packets. The queue a packet is scheduled to is derived from the priority of the packet, which can be configured to be based on either the VLAN tag or the outdated IP Type of Service field, which has been replaced by the DiffServ code point and the Explicit Congestion Notification. This mechanism can be used to ensure QoS requirements regarding throughput and latency. However, guaranteed data rates for multiple flows are difficult to realize, because the scheduler performs strict priority scheduling. A different algorithm, such as Weighted Fair Queuing, would be necessary for reserving a specific fraction of the total data rate for individual flows. Nevertheless, in situations in which the link is not saturated, i.e., all flows can be handled at their maximum data rate, the strict prioritization should be able to assure low latency for packets with high priorities.

Similar to the MAC IP core, the switch has a configuration and status interface that is accessible from within the FPGA design and also connects to the debugging interface. Using that interface, forwarding tables and setting registers can be managed. Various statistics can also be obtained, including the counter values for packets dropped because of overloading, invalid checksums or configured filter rules, as well as for packets that were longer or shorter than a configurable limit.

Chapter 5

Evaluation

The developed prototype has been evaluated in two phases to enable a better examination of individual parts of the design. In this incremental approach, measurements have been conducted with two variants of an early version of the prototype, and with an advanced version. The first two can only either loop back packets or generate and count packets, respectively. They do not contain the switch module, but the packet wrapping modules are integrated in both variants. The advanced version incorporates the switch IP core and is capable of forwarding, address learning and QoS mechanisms, among other features.

The evaluation covers measurements that gauge the latency and the throughput of the prototype versions, as well as the performance impact of employing DiffServ to provide QoS. Additional findings that could be relevant for use cases of a real system are presented toward the end of the chapter.

5.1 First Measurements

The first set of measurements was conducted with two configurations of an early-stage design that features a single external port. It can be described as a hybrid between the architecture proposed in Chapter 4 and Altera/Intel's design example demonstrating the MAC IP core. One of the two SFP+ slots available on the ReFLEX CES board is connected to a PHY and a MAC IP core instance. The receiving path of the MAC hands received packets to the Depacketizer, while the transmission path sends out packets that are created by the Packetizer. The variants differ in the fact that one of them connects the receiving path with the transmitting path using a synchronized buffer (Dual Clock FIFO) that is placed between the Depacketizer and the Packetizer. This leads to received packets being stripped of their headers and prepended with new headers in the Packetizer before they are returned to the original sender. In the other variant,

the paths are separated, and the Packetizer and Depacketizer instead interact with the generator and monitor, respectively, allowing generation and counting within the FPGA design.

The first variant can forward the payload contained in the received packets by unwrapping it and inserting new Ethernet and IP headers, therefore, from now on, it will be referred to as "forwarding variant" for clarity. Its structure is shown on the left of Figure 5.1. The second variant can generate and count packets, but cannot forward incoming packets. It will be called "generating variant" and is depicted on the right of the diagram.

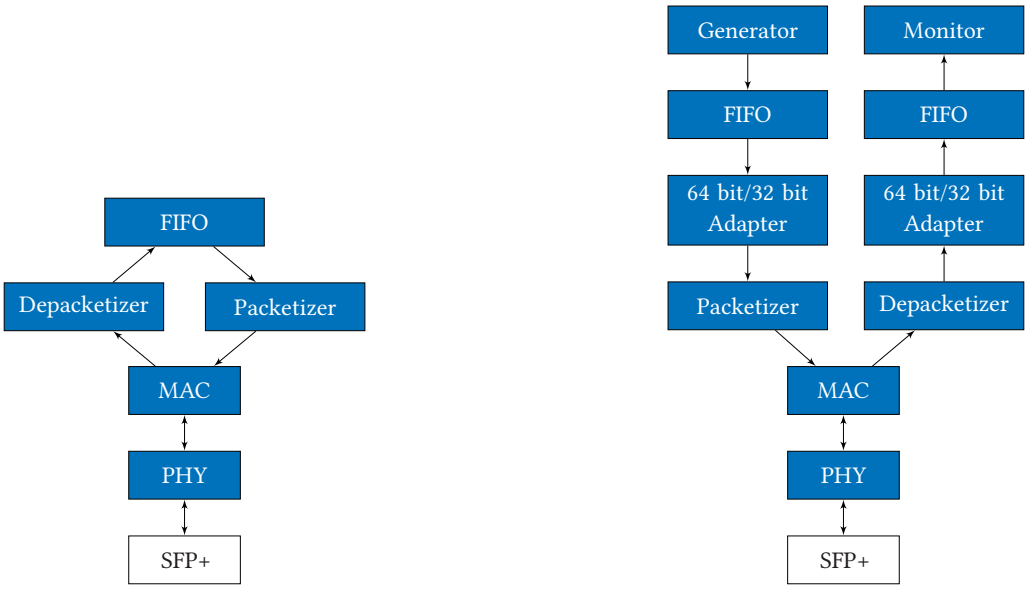

Figure 5.1: The two variants of the early prototype. The left side shows the forwarding variant, the right side shows the generating variant.

These two variants have been built because they already contain most of the modules that are part of the devised architecture and therefore are an important step both for the implementation and the evaluation of the complete system. With these two configurations, it is possible to test the throughput of the Packetizer and Depacketizer in a realistic setup using the generating variant, while the latency of those modules combined with that of the physical and link layer blocks can be gauged using the forwarding variant.

Throughput and latency in the context of this thesis are equal to the terms defined in RFC 1242 [45].

The forwarding variant allows measuring the latency of the device under test (DuT) by using an external device to send timestamped packets to the DuT which are looped back and can be analyzed. The throughput of the modules along the path of the packets can also be analyzed this way.

The generating variant of the system does not allow latency measurements from outside the FPGA, because it cannot return received data. However, the contained generator can simulate the image processing chain of a camera, which means that this variant can be used to measure the performance of the combination of the image chain and the Packetizer module in a real system. In addition, because it also contains the monitor and Depacketizer, it can be used to test the functionality of the system without the necessity for any external hardware. This was used in early phases of the implementation for debugging and measurement of the generated data rates by feeding the transmit signal of the SFP+ transceiver back to its receive signal with a single strand of optical fiber.

5.1.1 Throughput

In the first test that was performed, the generating variant of the prototype was set up to generate payload chunks that were then wrapped into Ethernet and IPv4 headers. The payload and packet lengths have been chosen in a way that results in frames of either the minimum or the maximum Ethernet frame size, i.e., 84 or 1538 bytes, including the preamble, the start of frame delimiter (SFD) and the interframe gap (IFG). The packets were looped from the transmitter of the optical transceiver back to its receptor with a single strand of fiber, and were handed to the monitor module from there (see Figure 5.2). The transceiver used was a multi-mode optical fiber SFP+ model with a wavelength of 850 nm (10GBASE-SR).

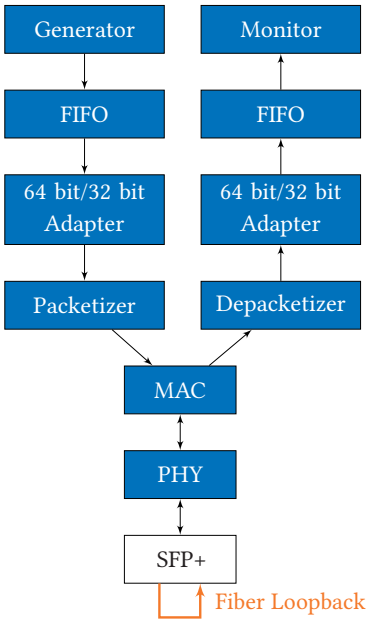

Figure 5.2: The test setup for the throughput measurements.

To measure the throughput achieved by the design, a tcl script running on the computer configured the payload length of the generator and the packet length of the Packetizer via the USB register interface. The script then started the generation of infinite data at maximum rate by writing to two more registers. After a test duration of 60 seconds, the script read the amount of received unwrapped payload chunks from the monitor module and stopped the generation. The average packet rate was then calculated from the amount of packets that were received, and the known time interval. Figure 5.3 shows the achieved throughput of this experiment for a selection of frame sizes, as well as the theoretical maximum imposed by the bandwidth limitation of the 10 Gbit/s Ethernet standard. The frame sizes have been chosen according to RFC 2544 [46], with the addition of 384 and 768 bytes to fill in the larger gaps in the diagram.

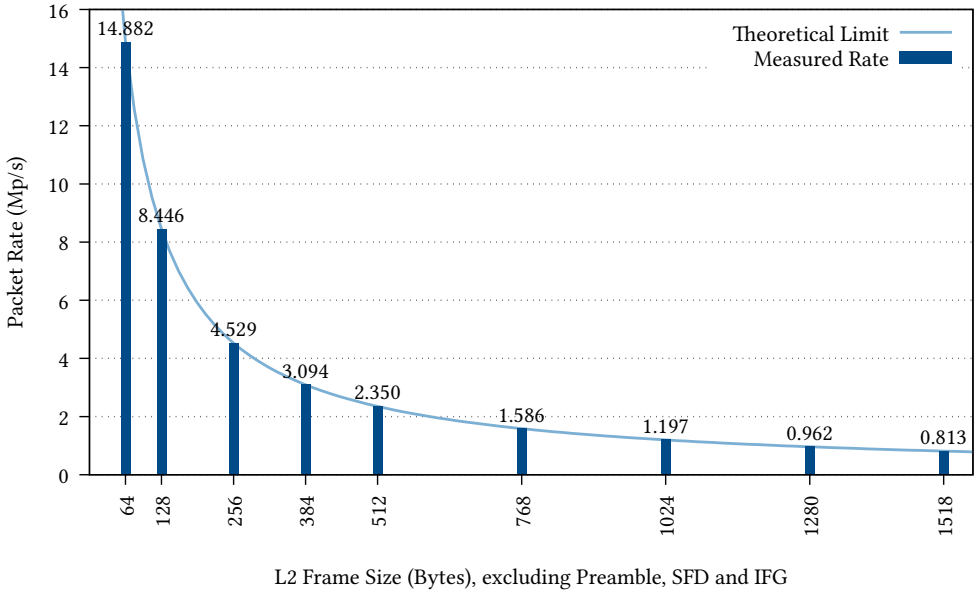

Figure 5.3: The throughput achieved by the generating variant of the early prototype at various frame sizes.

As the diagram shows, the maximum line rate has been achieved at all tested packet sizes. The measured packet rates are slightly above what is theoretically possible, which is assumed to be caused by delays in reading out the statistic register of the monitor, as well as possible inaccuracies in the measurement of the 60 seconds period in the tcl environment of the script. Another possible reason could be slight inaccuracies in frequency tuning of the transceivers. However, all of the measured packet rates deviate from the theoretical limit by less than 0.1%. This means all of the modules that make up the tested prototype are able to reach the maximum possible throughput, because if just one module could not handle the rate, it would act as a bottleneck and impact the final result. Therefore, the modules, and especially the Packetizer and Depacketizer are suited for their task regarding throughput.

To gauge the rate of errors introduced by the FPGA modules, the system was kept running in the forwarding state, with packets being generated at a rate of 10 Gbit/s for 15 hours. During this time period, the receiving MAC detected zero errors, and the number of packets sent matched the number that was generated. At a bit error rate of 10^{-13}, the expected number of occurred errors is 54, so the bit error rate of the system is assumed to be significantly lower than 10^{-13}.

5.1.2 Latency

For the latency measurements, the board with the FPGA running the forwarding variant of the early design was connected to a computer running a Linux distribution via a bidirectional 10 Gbit/s Ethernet link. Optical fiber SFP+ transceivers such as the one used before (multi-mode, 850 nm, 10GBASE-SR) have been used on both ends of the connection. The NIC installed in the computer was a 10 Gbit/s Intel X520 variant (Intel 82599ES controller) with two SFP+ slots, of which only one was used for this experiment. The right side of Figure 5.4 shows this configuration.

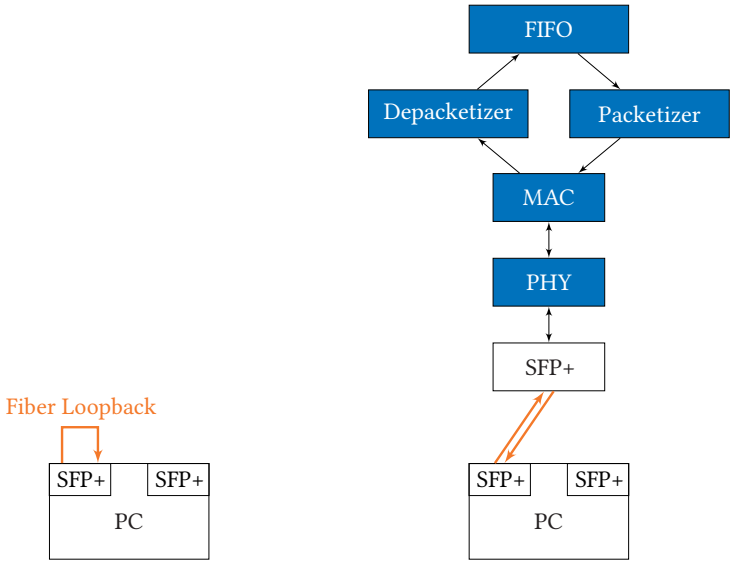

Figure 5.4: The loopback setup for measuring the latency of the transceivers and the cable, and the forwarding variant of the prototype connected to the computer.

The MoonGen packet generator [47] was used to generate and analyze the traffic in the latency experiments. It uses the Data Plane Development Kit (DPDK) for accurate latency measurements in hardware. MoonGen was running on the computer, generating traffic of selected data rates and packet sizes, with a small fraction of the packets carrying a hardware timestamp to allow measuring the latency. After the packets were processed and returned by the prototype, the difference between the timestamp in the packet and the current time of the NIC was calculated as the latency. More specifically, the

quality-of-service-test.lua script, which is one of the examples that come with MoonGen, has been used to configure its behavior. For all latency tests, the layer 2 frame length has been set to 128 bytes, including the Ethernet header and frame check sequence, but excluding the preamble, SFD and IFG. The source IP address for each packet was randomly selected from 256 addresses within the same subnet.

MoonGen can only measure the latency between the transmission and the reception of a packet in the NIC of the machine it is running on. This includes the propagation delay of the cable and the serialization delay at the host, which is not desirable, since these times depend on the test environment, and we want to determine the latency of the DuT only. To get results that are independent of the equipment used for the measurements, the propagation and serialization delays of the test setup have been measured using a loop back setup. For this, the transmitter of the optical SFP+ transceiver is directly connected to its own receptor with the same cable that is used in the actual tests, as Figure 5.4 shows (left side).

This way, the latency introduced by the cable, the NIC and the transceivers can be measured and then subtracted from the results gained in measurements of the DuT, resulting in values that only benchmark the devised prototype. To be perfectly accurate, the cable length in the loop back test would have to be of a length that is the sum of the lengths of the two fiber strands that lead from the computer to the DuT and back. Otherwise, the total propagation delay is not the same in both tests. However, since cables of less than 10 m length were used, the difference made by doubling the distance in the loop back scenario accounts to only a few nanoseconds, depending on the speed of light in the medium.

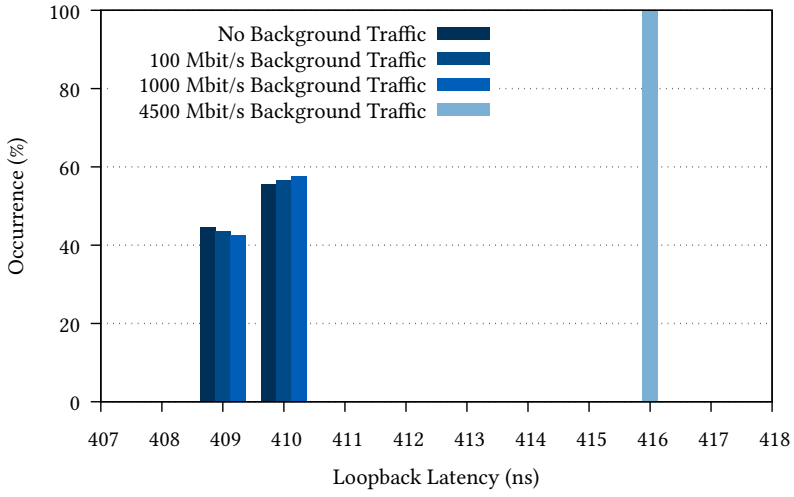

Figure 5.5: The latency measured in the loopback configuration at various data rates.

The results of the loopback latency measurement can be seen in Figure 5.5. The test has been performed with background data rates of 0 Mbit/s, 100 Mbit/s, 1000 Mbit/s and

4500 Mbit/s. The fact that the highest tested rate was only 4500 Mbit/s was caused by hardware restrictions of the computer running MoonGen, which was not able to handle a higher rate. As the diagram shows, there was hardly any variance in the measured latency.

With 4500 Mbit/s of background traffic, all packets have a constant latency of 416 ns. For other background rates, around half of the packets have 409 ns delay, while the others have 410 ns. It is possible that the actual values are between 409 ns and 410 ns, but because of the fact that the timestamp accuracy is limited to full nanoseconds, the latency is rounded up for some packets, and down for others. The offset in the case of 4500 Mbit/s of background traffic could not be explained, but as this thesis focuses on evaluating the devised architecture, this issue was not investigated further.

The low variance exhibited by the NIC allows for highly accurate latency measurements of the prototype, because any variance visible in the results of those tests is likely to be caused by the prototype, and not by external components.

After the latency in the loopback scenario had been measured, the FPGA board was added to the experiment as was already shown on the right side of Figure 5.4. The measurements were repeated with the same settings as before, i.e., a frame length of 128 bytes and background traffic rates up to 4500 Mbit/s.

Histograms plotting the resulting latencies against their frequency are shown in Figure 5.6. The latency found in the loopback has not been subtracted to allow better comparison with later charts. The reasoning for this will be given together with those advanced measurements in Section 5.2.1.

The plots of the measured latencies exhibit multiple interesting features. First, the histograms are only sparsely populated, with the three most frequent values making up more than half of the samples in all four scenarios. In the case of no background traffic, only seven different latencies were encountered in a total of over 4000 samples. This results in a standard deviation of 4.23 ns for the scenario without background traffic.

The fact that only few individual latency values were measured is assumed to be caused by the fact that all packets have the same lengths and the same header information, apart from the source IP address. Because of that, they should for the most part take the same path through the FPGA logic. That some packets were traveling faster or slower than the others might be caused by conditional behavior in the design that is triggered only after certain intervals and independently of packet content. Another reason could be that for some packets, the beginning of the frame was aligned with the clock of the PHY and MAC IP cores, while others had to be buffered longer to be synchronized to the internal clock.

Another interesting observation is that the histograms for the scenarios with more background traffic show that the variance of the latency increased with higher packet

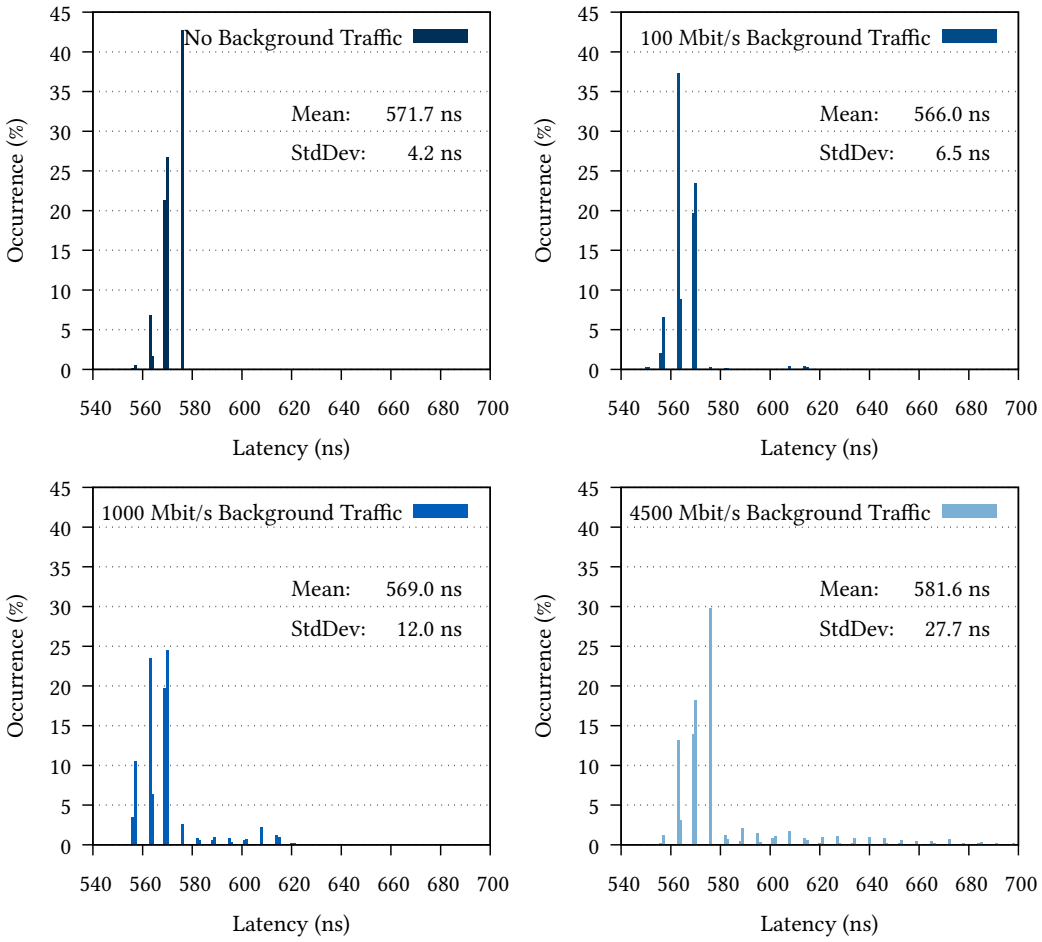

Figure 5.6: The total latency measured with the forwarding variant of the prototype at various data rates. The serialization delay at the PC and the propagation delay of the fibers are included.

rates. The graphs for 1000 Mbit/s and 4500 Mbit/s of background traffic display long tails, i.e., a comparably low number of samples with latencies that are significantly higher than the mean. For the 4500 Mbit/s scenario, the highest latency that was measured is 756 ns, which is 30% or six standard deviations above the mean. Roughly 0.9% of the samples in the 4500 Mbit/s case exceeded the 700 ns mark and were omitted from the chart to allow for a better comparison of the four cases.

The reason causing the long tails in the latency distribution for the cases with higher background traffic could not be found. Statistics tracking in the MAC module was suspected to cause small variations, but it is unlikely that such mechanisms could introduce additional latency of 175 ns.

Because of this inexplicable effect, the latency measurements were repeated with the same FPGA board, firmware, transceivers, and NIC family, and the same setup. The only difference was that a different machine was used to run MoonGen. The results of

the repeated experiment for the 1000 Mbit/s and 4500 Mbit/s background traffic cases can be seen in Figure 5.7. The histograms do not exhibit the long tails seen in the first graphs, and the standard deviations of the distributions both amount for only 4.8 ns, compared to the 12.0 ns and 27.7 ns from the previous iteration. The mean latency in all of the repeated measurements is comparable to the means of the first iteration, being around 570 ns.

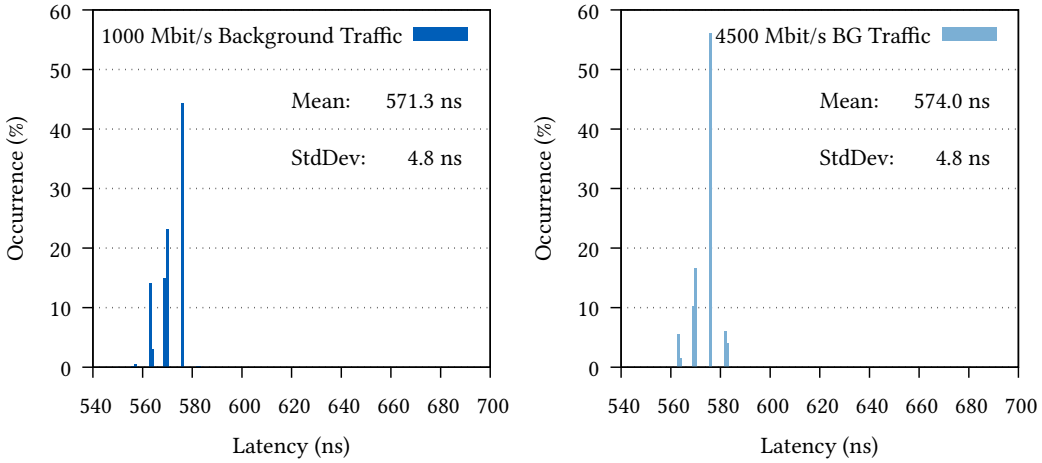

Figure 5.7: The results of the repeated experiment with the same setup as before, with the same NIC type, FPGA board, transceivers, and firmware, but with MoonGen running on a different computer.

The results of the second iteration of the experiment are assumed to be more likely to be correct than the results acquired in the first run. The main reason for that is that the prototype system only consists of few simple modules, which do not offer any feature that would be complex enough to create the long tail patterns seen in the histograms. They hardly buffer packet data because it would lower the performance, except for the Packetizer, which buffers all incoming data for a fixed amount of cycles to be able to insert the headers. However, since it is a constant time interval, this should not create a distribution as wide as in the graphs of the first iteration. The fact that the long tails could not be reproduced in the second run, although the DuT remained unchanged, indicates that the source of the long tails was not within the FPGA.

For all amounts of background traffic, the mean latency was close to 572 ns. Subtracting the average loopback latency of roughly 412 ns measured in the previous experiment leads to about 160 ns of latency introduced to the packets by the early prototype system. This value is less than half of the delay measured in the loopback case. Together with the low variance seen in the results of the repetition of the experiment, this proves that the performance of the modules that constitute this early design are suitable for real applications, regarding latency and throughput.

5.2 Advanced Evaluation with Switch

After conducting the aforementioned measurements with the early state of the system, the multilayer switch IP core was added to the system and integrated with the existing components to form a fully functional prototype for IP communication in a camera platform. The resulting system was evaluated in similar manner as the early versions, with a focus on latency and QoS.

The structure of the completed prototype has already been shown in Figure 4.2 at the beginning of Section 4.2, however including the parts that were omitted in the implementation. Without those parts, the system is structured as illustrated in Figure 5.8. The PC used for measurements is shown at the bottom, connected to the SFP+ transceivers of the FPGA board.

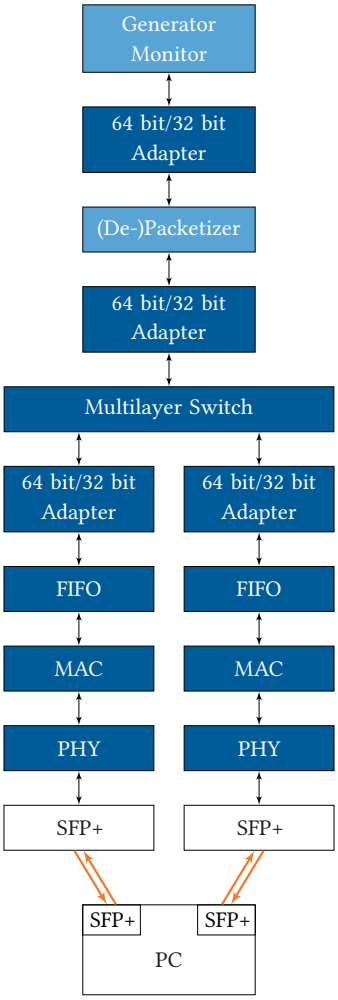

Figure 5.8: Functional block diagram of the developed prototype. Blue blocks are located inside the FPGA. Dark blue modules are realized as IP cores. The orange connections are optical fibers.

The hardware and software that was used for the evaluation of the complete system were the same as those used in the experiments conducted with the early prototype versions. However, since the final design makes use of both external ports that are available on the FMC extension card, two transceivers were used on both the FPGA side and the PC side of the connection. The devices were linked by two bidirectional multi-mode fiber cables (four strands), such as the one used in the previous tests.

One of the biggest problems encountered during the experiments was the complex feature set of the switch IP core, such as address learning and egress port resource limiting. Having to consider the possible impact of all these functionalities on a measurement makes it difficult to configure the switch in a way that allows the experiments to yield useful results.

5.2.1 Latency

The measurements performed with the complete design were focused on latency andQoS. This is because the multilayer switch represents a significant increase in complexity, as will be seen in Section 5.2.5. The switch IP core also introduces QoS mechanisms to the system, which are essential for operation in a real environment.

In a first experiment, the latency of the final prototype has been gauged using the same process as with the early prototype version. However, the packets were sent to the DuT via one of the external ports of the prototype, and were received from the other port, on which the switch forwarded them. To achieve this, a single packet was sent in the opposite direction in regular intervals, causing the switch to learn the address of the NIC at which the measurement traffic was directed.

The MoonGen framework was again used to generate background traffic at the same data rates as before, as well as the timestamped packets. Before the measurements were conducted with the DuT, the behavior of the loopback setup (see Figure 5.4) was once again determined to ensure the comparability of the results. The results obtained from this test can be seen in Figure 5.9. Only the case of 1000 Mbit/s of background traffic is shown, because the graphs for the other cases hardly differ.

Curiously, the latency distribution in the loopback case is different from what was measured in the first set of experiments, despite the fact that the same equipment has been used. In the first test, the latency was constant for all packets measured at the same rate of background traffic. The graph for this iteration, in contrast, shows a high variance, which becomes especially obvious with a logarithmic scale of the occurrence frequency, which can be seen on the right of Figure 5.9. Instead of a constant latency as in the first iteration, 150 different values were measured, resulting in a standard deviation of 32.1 ns.

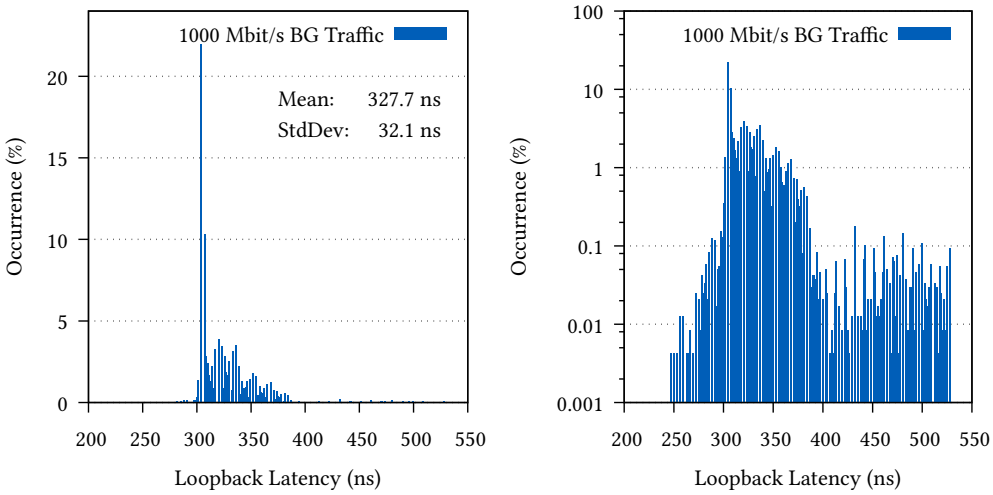

Figure 5.9: The latency results of the second loopback experiment, shown with linear and logarithmic scaling on the vertical axis.

The reason for this drastic change in variance could not be found. However, it should be noted that even though MoonGen offers hardware timestamping support, its latency granularity was reported to be 12.8 ns on a 10 cm fiber cable [48]. With the 3 m cable that was used in the tests for this work, this could be a possible cause of the variance, although it is unclear why the first execution yielded constant latencies.

Because of the high variance in the loopback measurements, it is not possible to subtract that latency from the values measured with the setup including the prototype. It is unclear whether or not the prototype would introduce this much variance on its own, even if the latencies outside of the device are constant. Therefore, all results shown in the following are influenced by all sources of delay in the path from the sending NIC to the receiving NIC, including the serialization and propagation delays.

Although the high variance in the loopback latency reduces the accuracy of the results of the measurements of the prototype, their plots still exhibit some interesting features. Figure 5.10 shows the latencies measured with the final prototype version at 1000 Mbit/s and 4500 Mbit/s of background traffic. The results were plotted for the interval from 1500 ns to 1800 ns, which includes 99.5% of the samples for both cases.

Compared to the latency distributions of the repeated experiment with the early prototype, these results have a mean latency of roughly 1670 ns, which is almost three times as high. Interestingly, although the shape of the curve is similar for the 1000 Mbit/s and 4500 Mbit/s cases, the latter has a mean latency that is almost 30 ns lower.

The standard deviation is close to seven times that of the version without the multilayer switch module, although, as has been stated, it is uncertain how much of the variance was caused by the switch. While the standard deviation of this state of the system is

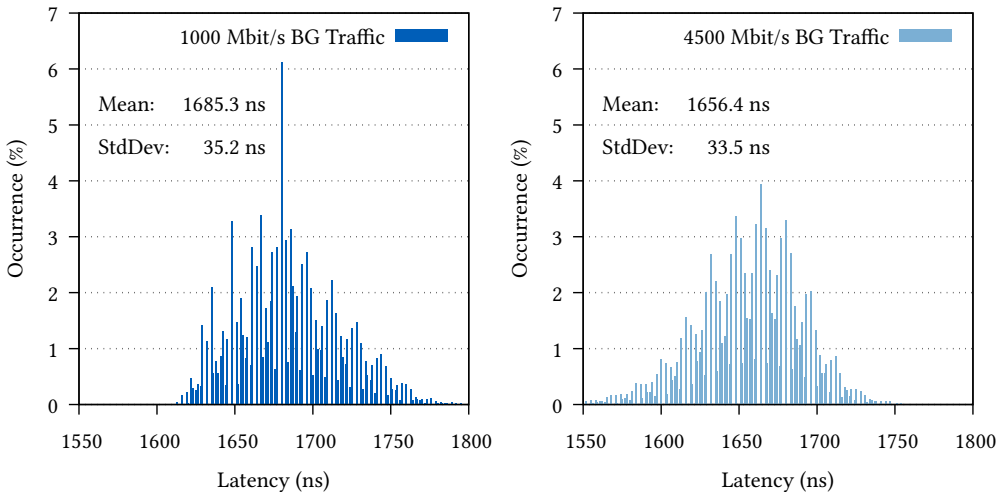

Figure 5.10: The latencies measured with the final design of the prototype.

significantly higher than that of the early version, the second loopback measurement exhibited a standard deviation of 32.1 ns, which is close to what was measured with the final design. Therefore, it seems that the DuT only introduced a small amount of additional variance, and most of it was caused by the environment of the experiment. To prove this theory, a higher accuracy in packet generation and measurement would have been necessary.

While most of the higher mean latency is certainly caused by the added complexity of the switch, some of it is expected to be caused by the use of 64 bit to 32 bit adapters in the design. As Section 4.2.8 explained, they had to be added because of specification changes, and would be omitted in an improved implementation. This is expected to reduce the latency of the system by values in the order of magnitude of 20 ns.

5.2.2 Quality of Service

As Sections 3.3.6 and 4.1.4 explained, DiffServ has been chosen for QoS enforcement. To evaluate the performance of this mechanism, the *quality-of-service-test.lua* script was extended to allow differing values for the Type of Service IP header field for the foreground and background traffic.

By default, the switch IP core determines queueing priorities based on the VLAN tags of incoming packets, or assigns a default queue for untagged packets. For the QoS tests, the switch has therefore been configured to instead assign egress queues based on the Type of Service field in the IP header, which contains the DiffServ code point. The Expedited Forwarding PHB was bound to the queue of the highest priority (queue 0),

while packets with the Best Effort PHB were configured to be scheduled to queue 7, which has the lowest priority.

As long as the switch only receives packets from one port, it should start forwarding them as soon as they are received. Each packet can be queued as soon as its header has been processed, because with only one ingress port in use, only one packet can arrive at a time. This means that queue priorities should have no influence on the latency as long as just one port receives packets.

The top row of Figure 5.11 shows the latency for packets labeled with the Best Effort PHB, while the bottom row shows the latency for the Expedited Forwarding PHB. There is no relevant difference between the two rows, which means that the assumption holds true, and prioritization has no effect in situations with only one ingress port.

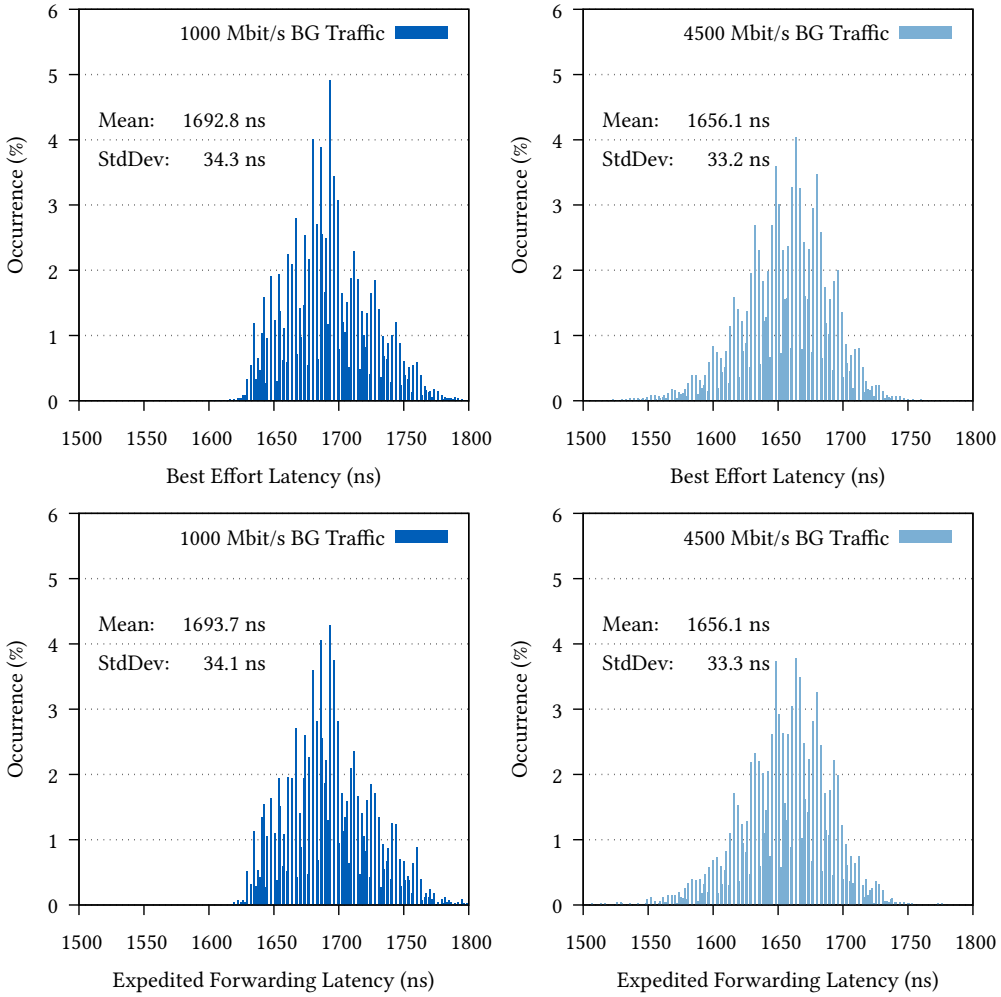

Figure 5.11: The latencies measured with the final design of the prototype, with priority queueing enabled in the switch.

For the final measurements, the generator and the Packetizer within the prototype were configured to generate packets at a rate of 4500 Mbit/s as well, simulating a situation in which the image processing chain outputs video data, while external devices communicate via the camera-internal switch. The external communication was once again simulated by traffic generated by MoonGen.

The payload data created by the generator was wrapped into packets marked with the Expedited Forwarding PHB to simulate a live video stream with high priority. To achieve this, the Packetizer module was set up to configure the DiffServ code point accordingly.

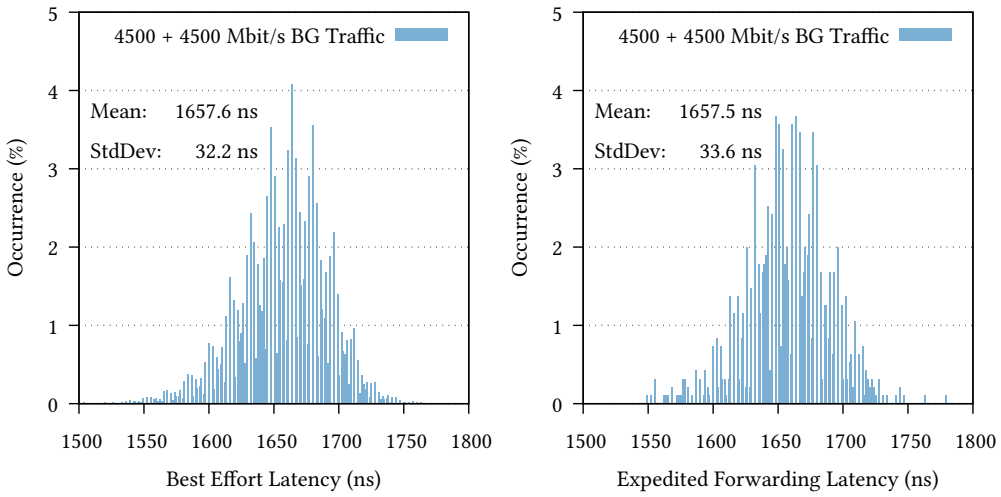

Figure 5.12: The latencies measured with the final design of the prototype, with priority queueing enabled in the switch. Apart from the traffic generated by MoonGen, additional background traffic of 4500 Mbit/s was generated inside the FPGA and sent on the same egress port.

The fact that packets are being scheduled to the same egress port from two source ports, together with the total data rate of 9 Gbit/s was expected to cause traffic with a higher priority to have a lower latency than others. The diagrams in Figure 5.12 show the results of the measurements for traffic with low priority on the left, and high priority on the right. No improvement in latency or variance is visible, as the distribution is mostly the same, and both mean and standard deviation of the two cases are almost equal to each other.

However, since the link is still not fully saturated and buffering takes place at multiple points within the device, there is a possibility that the number of packet headers that are queued perfectly simultaneously is so small, that the improvement gained by prioritization is concealed by the variance introduced by the generator and the NICs.

5.2.3 Congestion and Packet Drops

In situations where an egress port is congested, it would be desirable that the traffic with the highest priority is forwarded at a rate close to the line rate of 10 Gbit/s. However, with the switch core variant that was available, this was not the case. The resource limiter, which decides when to drop packets, acts based only on the amount of buffer memory consumed by packets queued to a certain port, but does not factor in the priorities of the packets in any way. Therefore, packets are dropped regardless of their assigned queue, essentially resulting in packets with all priorities being dropped at random. This was verified with the same setup used for the measurements shown in Figure 5.12, by increasing the data rates to reach a total of over 10 Gbit/s, leading to congestion of the egress port.

A variant that does support dropping based on packet priorities would have been available as well, but the lack of this feature has not been noticed until late in the project phase, so that an evaluation of that variant was not possible anymore.

For a real system, it would be beneficial if packets were dropped depending on their priority instead, since this would ensure that applications with higher importance can continue to function reasonably well, even if a link becomes congested. Without this feature, additional care must be taken to guarantee that the total data rate of every egress port stays below the maximum line rate of the medium.

Since the bandwidth requirements of video streams are determined by the image format and encoding, it should be relatively easy to implement a system that reserves a certain fraction of the link for video. If the remaining available data rate is sufficient to support lighter applications such as remote controlling or synchronization, preventing congestion could be possible, as long as no malfunction occurs, or new devices with higher bandwidth consumption are attached.

Even though the system including the switch has been thoroughly tested in a variety of configurations, from time to time a situation was encountered in which the switch would randomly drop packets without any apparent reason. These packets were counted as dropped because of port overuse or buffer overflow in the statistics, although they were dropped in situations, in which no ports had to handle a rate close to the maximum data rate of 10 Gbit/s.

Some minor timing issues within the FPGA design could not be fixed until the end of the project because of the complexity of the complete prototype. It is however unlikely that those timing problems are causing such behavior in the switch, as they were mainly located in other modules. Therefore it is unclear why exactly these events happen, and they had to be evaded during experiments.

5.2.4 Interoperability

As Chapters 2 and 3 have described, interoperability between a variety of devices and interconnect standards is an important requirement for broadcast and movie set environments. During the development and the measurements of the prototype system, it has been tested with a multitude of other devices.

The prototype was found to be compatible with 10 Gbit/s Ethernet SFP+ transceivers for both optical fiber and copper media. Apart from the Linux computers equipped with 10 Gbit/s NICs that were used for the measurements, the system was successfully tested in conjunction with a Cisco 10G Ethernet switch, as well as with a Laptop with a network interface that does not support data rates of 10 Gbit/s. In this case, a copper SFP+ transceiver was connected to the Laptop, and the transceiver downscaled the link speed to comply with the lower rate of the other end of the connection. This was tested with a simple echo request being sent from one computer to the Laptop. The packet had to traverse a 10 Gbit/s fiber link to the prototype, and a copper link from there to the Laptop, and was successfully received and answered by the Laptop.

5.2.5 Resource Consumption

In addition to the performance of the designed system regarding networking parameters, the resource utilization of the FPGA has been evaluated as well. In particular, the resources used by multiple combinations of PHY and MAC IP core variations used in the Altera/Intel design examples have been compared to justify the choice of which combination to base the prototype on.

Design	PHY IP core variant	MAC IP core variant
I	Native (10GBASE-R, register mode)	10GBASE-R (register mode)
II	Native (10GBASE-R)	10GBASE-R (normal mode)
III	1G/10G	10M/100M/1G/10G

Table 5.1: The combinations of PHY and MAC configurations in the example designs that have been analyzed. The final design uses the configurations of example design I.

The different design examples that were examined are given in Table 5.1. They differ in the IP cores that were used for the physical and link layers, as well as the available configuration options. The 10GBASE-R variants of the PHY and the MAC offer a register mode option, which offers a lower latency at a higher resource count compared to other single data rate configurations. The variants that support dynamic switching between data rates require more resources to implement the switching logic, as Table 5.2 shows. The amount of resources used by the final version of the prototype have also been included in the table for comparison, as well as the total count of resources available on the Arria 10 model that was used.

Design	Ports	PLLs	Registers	ALMs
I	1	3	9964	7725
II	1	5	19560	14325
III	1	4	11174	7823
Final	2	5	91598	55073
Available	48	112	–	427200

Table 5.2: A comparison of various design examples and the final design including the switch regarding the amount of Phase-Locked Loops (PLLs), registers and Adaptive Logic Modules (ALMs). The last row lists the amount of resources available on the device.

The amount of registers and ALMs that were required for the completed design amounted for almost 13% of what is available on the device. While this was not problematic for the testing and evaluation of the system, it is a rather high fraction of the capabilities of the FPGA that can not be used for applications that are crucial for digital cameras, such as image processing, encoding, and synchronization. However, it should be noted that few effort was made to optimize the spacial requirements of the design, because this work focuses on functionality and performance.

Another detail that should be noted is that the fitter process of the Altera/Intel design suite lists all 48 transceiver PLLs as being required by the design, although it actually only uses one for each external port. This is caused by the software implementing a workaround that prevents the degradation of unused clock lines over time. The unused PLLs are connected to a slow clock so that the signal connections remain intact. Since this causes a higher power drain of the system, it can be deactivated if the PLLs are planned to not be used in future firmware upgrades. If an upgrade requires additional PLLs and the compensation has been employed, some of the previously unused PLLs can be used without any negative effects.

Chapter 6

Related Work

To give some perspective of the research field of IP communication in the motion picture and broadcast industries, as well as embedded networking using FPGAs, some related research projects will be described in the following.

The idea of incorporating a network switch directly into a camera seems to be an uncommon one. While networked cameras are well-established in the surveillance sector, integrating IP into a camera as tightly as demonstrated in this thesis seems to be a novel approach. However, procedures to replace the core SDI infrastructure of studios and broadcasting venues are a rather active area of research.

For example, Montalvo et al. devised and demonstrated a prototype for a TV studio based on IP communication and the Audio/Video Bridging standards, which is used to transport video and synchronization signals [27]. Their paper focuses mostly on the requirements for the core network infrastructure instead of peripherals, in contrast to this thesis, which deals with the requirements for adding IP to the cameras. They base their architecture on the assumption that cameras and other communication endpoints continue to communicate via SDI, and thus add SDI to IP converters to integrate those devices.

For the 2014 International Broadcasting Convention, Gardiner et al., working with Sony, released a paper describing a similar architecture for live production environments based on IP communication [30]. The paper also gives an overview of standards and mechanisms that might prove useful to realize such infrastructures.

M. Laabs summarized the SMPTE 2022-6 standard for SDI over IP, highlighting the opportunities gained by combining those two worlds, and proposed an improved concept for seamless switching of SDI signals in IP networks [37]. His publication has been used as a reference for the standards surrounding SDI transport via HBRMT packets.

As Section 4.2.7 mentioned, the Packetizer module implemented for the prototype was to an extent based on a paper by Alachiotis et al., in which they present an implementation

for a UDP/IP FPGA IP core [44]. The core supports a data rate of 1000 Mbit/s compared to the 10 Gbit/s targeted by this thesis, but only occupies 1% of the resources of a Xilinx Virtex 5 FPGA. They also reference a similar project, which however supports only even lower rates [49]. Neither of the projects included a network switch, making the overall system less complex, although both implement the entire stack from scratch instead of using readily available IP cores.

Finally, J. Hudson and E. Frlan propose the use of SDI infrastructure with higher data rates in their 2014 paper. They hope to achieve data rates up to 192 Gbit/s, which would be able to transport 3D video at a resolution of 8K, using multi-link SDI setups as an intermediate step until higher data rates are being standardized [20].

Chapter 7

Future Work

The evaluation of the prototype has shown that it performs decently in situations that resemble real scenarios. However, there clearly is room for improvement in some regards, both in the implementation and in the evaluating measurements. This chapter will give an overview of possible extensions to the system and the tests performed on it, which could not be performed in the scope of this thesis because of time and hardware restrictions.

7.1 Design Improvements

One of the drawbacks of the described implementation is the amount of adapters that are necessary to connect modules with interface widths of 64 bit and 32 bit. They consume additional resources and introduce latency and complexity to the design. Removing those adapters would require resizing the interfaces on one side of each adapter. This means that the internal logic would have to be changed, too.

For example, the internal state machine of the Packetizer depends on the input that has been received in the previous clock cycle. Changing the interface from 32 bit to a width of 64 bit would introduce new states and thus require changes to the state machine. Such complex changes were expected to require too much time to perform them as part of this thesis. Inserting adapters was a fast and simple approach to the problem, which is why it was preferred over restructuring the interfaces.

Another limitation is the fixed payload length of the Depacketizer modules. While the length of the payload that should be unwrapped can be configured, it is not dynamically inferred from the packet contents. The Ethernet, IP, and UDP headers all have length fields that inform a receiver of the length of the payload a packet is carrying, which could be used by the Depacketizer to remove all headers until only the payload remains. This

feature was omitted to simplify the implementation, but would be a useful extension that adds flexibility regarding the payload format for return-in monitoring applications.

The Packetizer also offers some potential for improvements. For the prototype, only the Ethernet and IP layer have been implemented, while an additional UDP or ICMP header can be added by the generator. To allow full HBRMT support, the UDP, RTP, and HBRMT header fields would have to be added to the combined header data structure. This extension should be comparably easy to realize, because the Packetizer has been built with this enhancement in mind.

The most important improvement to the system however would be a version of the multilayer switch IP core which supports dropping packets based on their DiffServ priority in case of a congested egress port. This change would significantly increase the reliability of the system, as well as the flexibility regarding dynamic data rates and new devices being added to the network at runtime.

7.2 Further Measurements

The measurements documented throughout Chapter 5 revealed various characteristics of the prototype implementation of the devised architecture. Some of the results were somewhat lacking in accuracy however, because of the variance introduced by the external equipment and the high requirements with respect to latency jitter of synchronization applications. Hence it would be helpful to conduct further tests to gain more precise results for the latency of the system. A possible approach could be to measure the latency within the system itself by adding a synchronization between the generator and the monitor modules, e.g., by marking packets with a counter value that increments every clock cycle and is shared between the two units. This would allow for latency measurements without the influences of external software and hardware.

Varying the packet sizes in the latency experiments could also provide additional insight about the impact of enabling priority queueing within the switch. Longer packets could increase the difference between the latencies of packets with high and low priorities.

7.3 Security and Safety Considerations

Finally, networking interfaces always pose a security risk. This aspect of the topic has not been covered by this thesis, but it does hold some relevant implications. Integrating the network gateway directly into the FPGA, with a direct connection to the image processing chain, bears the risk of the core of the camera being damaged or sabotaged, e.g., by injecting malicious packets which break image synchronization or attaching devices with excessive voltages.

It is also imaginable that misconfigured or broken intermediate devices forward data in a different way than planned. This could lead to signal loss and disruptions, but if misused deliberately, this could lead to video data being streamed into networks for which the data was not designated, possibly leading to parts of unreleased productions to be leaked. Therefore, a secure configuration of intermediate devices is necessary to prevent misuse.

However, multiple cameras that are available today already offer a network connection for purposes such as remote control or maintenance. As of now, there has not been a relevant case of damage or misuse caused by this interface, which might be due to tight access control to the cameras for reliability reasons. Thus it is unclear how much of a problem the additional sensitivity of the proposed architecture poses in real environments.

The lower robustness of optical fiber could present a problem to studio environments. However, to reach the data rates of emerging image formats, higher SDI data rates will probably have to be implemented with optical fiber as well, which means the industry has to adopt this new medium in any case.

Chapter 8

Conclusion

Newly emerging applications and disruptive technologies have recently found their ways into the broadcast and motion picture industries, posing major challenges for the workflow and the installed infrastructure. There is a need for a fresh approach to handling the data in such environments because the prevalent SDI standards are not practical anymore for all applications.

As the analysis in the beginning of this thesis showed, using the Internet Protocol suite is a promising solution that offers flexibility and extensibility, while allowing backwards compatibility and scalability. To meet the requirements of applications such as live monitoring, synchronization, and remote control, mechanisms for Quality of Service and SDI-to-IP conversion have to be implemented. The combination of IP and Ethernet was found to be a suitable platform which can satisfy these demands.

Other options, such as multi-link SDI or lightweight video encoding could perform well regarding certain aspects, but are rather limited in their application and thus do not bear as much potential.

An architecture for enabling 10 Gbit/s Ethernet within cameras has been proposed. It integrates IP communication closely into a camera system by embedding a multilayer network switch into the FPGA fabric in which the core of the camera resides. The individual modules of the architecture have been explained and partially implemented in a prototype system.

The prototype was evaluated with a focus on QoS and latency, and was found to perform reasonably well, especially considering that no radical optimizations have been performed. It achieves the maximum line rate of the underlying Ethernet standard and exhibits a round-trip latency of roughly 1660 nanoseconds.

In summary, IP communication seems to be a promising solution to the problems the motion picture and broadcast industries are facing. Incorporating a multilayer network switch into the camera has proven to be a useful approach to the challenges posed by

the requirements of some common applications, even though further improvements and tests will be necessary before such a system can be put to commercial use in a future camera system.

Glossary

ASIC – Application-Specific Integrated Circuit

An integrated circuit specifically made for one particular task, which cannot be changed after manufacturing, but offers high speed and a low cost per unit, once the production has been initialized [7].

BNC – Bayonet Neill–Concelman

A universal type of coaxial cable connector used in various industries and especially common in the broadcast sector.

DiffServ – Differentiated Services

A simple mechanism to provide a variety of classes of Quality of Service in a network that relies on marking packets as belonging to a certain class.

FPGA – Field Programmable Gate Array

An integrated circuit which can be reconfigured after production by loading a hardware design defined in a hardware description language. They offer short time to market and high flexibility, but are relatively expensive [7].

fps – Frames per second

GPIO – General Purpose Input/Output

Genlock – Generator Locking

The process of locking the clock frequency and phase of a device to those of a generator to allow synchronization between multiple devices on a site.

HBRMT – High Bit Rate Media Transport

HFR – High Frame Rate

Frame rates that are higher than the typical rates used in the industry are referred to as High Frame Rate. For cinema productions, this can mean rates of 48 frames

per second, but can go much higher, up to 200 frames per second. In this thesis, video footage recorded at rates of 60 fps and above will be considered as high frame rate footage.

IntServ – Integrated Services

A set of mechanisms to provide specific Quality of Service attributes for flows in a network that is based on making resource reservations across the entire network for individual flows.

IP core – Intellectual Property Core

A reusable circuit design, often provided in the VHDL or Verilog hardware description languages, that implements a certain function and is intellectual property of a party. Free and paid variants exist. Hard IP cores are delivered as fixed hardware layouts, while soft IP cores usually come as hardware description language source code or netlists and are more flexible.

IP – Internet Protocol

A data transmission protocol that transmits "datagrams from sources to destinations" [50].

PTP – Precision Time Protocol

QSFP – Quad Small Form-factor Pluggable

A compact hot-pluggable transceiver type that succeeds SFP and supports 4×1 Gbit/s. It can provide four times the data rate of a SFP transceiver at 1.5 times the size. QSFP+ is an extension of the standard that comes in a variety of data rates, and can support up to 4×28 Gbit/s.

QoS – Quality of Service

The quality of a networking connection regarding attributes such as latency, packet loss and throughput. It is also used to refer to a set of mechanisms used to implement certain levels of quality, e.g., prioritization.

RTP – Real-Time Transport Protocol

SFP – Small Form-factor Pluggable

A compact hot-pluggable transceiver type that supports data rates from 1 Gbit/s to 5 Gbit/s. The enhanced variant SFP+ offers up to 16 Gbit/s, while SFP28 offers up to 25 Gbit/s.

SDN – Software-Defined Networking

A relatively new approach to network configuration that focuses on dynamically programmable and openly available interfaces, usually by separating the control plane from the data plane and providing a centralized view of the network [51].

TCP – Transmission Control Protocol

UHD – Ultra High Definition

Image formats with resolutions of 3840×2160 or 7680×4320 pixels [52], which are also referred to as 4K or UHDTV-1 and 8K or UHDTV-2, respectively. In the motion picture industry, a resolution of 4096×2160 pixels is also used.

UDP – User Datagram Protocol

List of Figures

List of Tables

Bibliography

[1] "Cannes 2016: What cameras were used to shoot this year's films," May 2016, accessed June 14, 2017. [Online]. Available: http://www.indiewire.com/2016/05/cannes-2016-what-cameras-were-used-to-shoot-this-years-films-289100/

[2] "Sundance 2017: Here are the cameras used to shoot this year's acclaimed films," Jan. 2017, accessed June 14, 2017. [Online]. Available: http://www.indiewire.com/2017/01/sundance-2017-cameras-arri-canon-red-sony-1201770071/

[3] V. Hyvönen, "Helicam Arri Alexa Mini," Aug. 2015, accessed June 16, 2017. Licensed under https://creativecommons.org/licenses/by-sa/2.0/legalcode. [Online]. Available: https://www.flickr.com/photos/villehoo/19818005793/

[4] S. P. Anderson, "Arri Alexa," Apr. 2014, accessed June 16, 2017. Licensed under https://creativecommons.org/licenses/by/2.0/legalcode. [Online]. Available: https://www.flickr.com/photos/seanpanderson/14390225393/

[5] Morio (user at Wikimedia Commons), "2013 Sony CineAlta 4k camcorder PMW-F55," Feb. 2013, accessed June 16, 2017. Licensed under https://creativecommons.org/licenses/by-sa/3.0/legalcode. [Online]. Available: https://commons.wikimedia.org/wiki/File:2013_Sony_CineAlta_4K_cam_2013_CP%2B.jpg

[6] V. Hyvönen, "RED Epic Dragon 6k," Dec. 2013, accessed June 16, 2017. Licensed under https://creativecommons.org/licenses/by-sa/2.0/legalcode. Brightness reduced. [Online]. Available: https://www.flickr.com/photos/villehoo/11216966276/

[7] Xilinx Inc., "FPGA vs. ASIC," accessed June 12, 2017. [Online]. Available: https://www.xilinx.com/fpga/asic.htm

[8] ARRI AG, "Image processing hardware," accessed June 19, 2017. [Online]. Available: http://www.arri.com/camera/alexa/technology/arri_imaging_technology/image_processing_hardware/

[9] PR Newswire Association LLC, "Hollywood's top-selling camera maker ARRI chooses Altera for AMIRA documentary camera," accessed June 19, 2017. [Online]. Avail-

able: http://www.prnewswire.com/news-releases/hollywoods-top-selling-camera-maker-arri-chooses-altera-for-amira-documentary-camera-274790171.html

[10] Xilinx Inc., "Blackmagic URSA, world's first user-upgradable 4K digital film camera, has a Xilinx Kintex-7 FPGA inside," Apr. 2015, accessed June 19, 2017. [Online]. Available: https://forums.xilinx.com/t5/Xcell-Daily-Blog/Blackmagic-URSA-world-s-first-user-upgradable-4K-digital-film/ba-p/588553

[11] R. Koo, "RED to cut prices on EPIC and SCARLET cameras november 1st, Dragon sensor delayed," Oct. 2012, accessed June 19, 2017. [Online]. Available: http://nofilmschool.com/2012/10/red-dragon-sensor

[12] "SDTV digital signal/data – serial digital interface," *SMPTE ST 259:2008*, pp. 1–18, Jan. 2008.

[13] "1.5 Gb/s signal/data serial interface," *SMPTE ST 292-1:2012*, pp. 1–20, Jan. 2012.

[14] "3 Gb/s signal/data serial interface," *SMPTE ST 424:2012*, pp. 1–10, Oct. 2012.

[15] "2160-line and 1080-line source image and ancillary data mapping for single-link 6G-SDI," *SMPTE ST 2081-10:2015*, pp. 1–22, Mar. 2015.

[16] "2160-line source image and ancillary data mapping for 12G-SDI," *SMPTE ST 2082-10:2015*, pp. 1–21, Mar. 2015.

[17] J. Hudson, "1080p50/60, 4k and beyond: Future proofing the core infrastructure to manage the bandwidth explosion," in *The 2012 Annual Technical Conference Exhibition*, Oct. 2012.

[18] "SMPTE standards quarterly report," Mar. 2014, accessed May 30, 2017. [Online]. Available: https://www.smpte.org/sites/default/files/2014-03%20Niagra%20Outcome%20Report%20FINAL.pdf

[19] "SMPTE working group 32NF-70 website," Sep. 2013, accessed May 30, 2017. [Online]. Available: https://kws.smpte.org/higherlogic/ws/public/projects/project/details?project_id=180

[20] J. Hudson and E. Frlan, "Toward a hierarchy of SDI data rates," *SMPTE Motion Imaging Journal*, vol. 123, no. 3, pp. 17–40, Apr. 2014.

[21] F. Rumsey, *Digital Interface Handbook*, 3rd ed., J. Watkinson, Ed. Oxford [England]: Focal Press, 2004.

[22] "10 Gb/s serial signal/data interface," *SMPTE OV 435-0:2012*, pp. 1–2, Sep. 2012.

[23] "Dual 1.5 Gb/s serial digital interface for stereoscopic image transport," *SMPTE ST 292-2:2011*, pp. 1–10, Jun. 2011.

[24] ARRI AG, "In virtual reality on the red carpet," Feb. 2017, accessed May 22, 2017. [Online]. Available: http://www.arri.com/news/news/in-virtual-reality-on-the-red-carpet/

[25] Road to VR, "NextVR's stereoscopic 360-degree VR cam uses $180,000 worth of RED 6k cameras," Sep. 2014, accessed May 22, 2017. [Online]. Available: http://www.roadtovr.com/nextvr-stereoscopic-360-degree-vr-cam-uses-180000-worth-of-red-6k-cameras/

[26] S. Corporation, "PMW-F55 – specifications," accessed June 19, 2017. [Online]. Available: https://www.sony.co.uk/pro/product/broadcast-products-camcorders-digital-motion-picture-camera/pmw-f55/specifications/

[27] L. Montalvo, G. Macé, C. Chapel, S. Defrance, T. Tapie, and J. L. Roux, "Implementation of a TV studio based on Ethernet and the IP protocol stack," in *2009 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting*. IEEE, 2009.

[28] "Definition of vertical interval switching point for synchronous video switching," *SMPTE RP 168:2009*, pp. 1–23, Jul. 2009.

[29] Arista Networks Inc., "Broadcast transition from SDI to Ethernet," 2016, accessed May 23, 2017. [Online]. Available: https://www.arista.com/assets/data/pdf/Whitepapers/Broadcast_Transition_from_SDI_to_Ethernet.pdf

[30] P. N. Gardiner, J. J. Stone, J.-R. Chen, and T. Kojima, "IP live production," *IET Conference Proceedings*, pp. 6.3–6.3(1), Jan. 2014. [Online]. Available: http://digital-library.theiet.org/content/conferences/10.1049/ib.2014.0019

[31] "IEEE standard for information technology – local and metropolitan area networks – specific requirements – part 3: CSMA/CD access method and physical layer specifications amendment 4: Media access control parameters, physical layers, and management parameters for 40 Gb/s and 100 Gb/s operation," *IEEE Std 802.3ba-2010 (Amendment to IEEE Standard 802.3-2008)*, pp. 1–457, Jun. 2010.

[32] "IEEE standard for information technology – telecommunications and information exchange between systems – local and metropolitan area networks – specific requirements part 3: Carrier sense multiple access with collision detection (csma/cd) access method and physical layer specifications," *IEEE Std 802.3an-2006 (Amendment to IEEE Std 802.3-2005)*, pp. 1–167, 2006.

[33] D. Schelle, "12 Gbps SDI cable performance estimation," Nov. 2015, accessed June 26, 2017. [Online]. Available: http://www.edn.com/design/analog/4440765/12-Gbps-SDI-cable-performance-estimation

[34] A. F. Benner, P. K. Pepeljugoski, and R. J. Recio, "A roadmap to 100g ethernet at the enterprise data center," *IEEE Communications Magazine*, vol. 45, no. 11, pp. 10–17, Nov. 2007.

[35] P. J. Winzer, "Beyond 100G Ethernet," *IEEE Communications Magazine*, vol. 48, no. 7, pp. 26–30, Jul. 2010.

[36] "Transport of high bit rate media signals over ip networks (hbrmt)," *SMPTE ST 2022-6:2012*, pp. 1–16, Oct. 2012.

[37] M. Laabs, "SDI over IP – seamless signal switching in SMPTE 2022-6 and a novel multicast routing concept," *EBU Technical Review*, 2012.

[38] "Forward error correction for transport of high bit rate media signals over IP networks (HBRMT)," *SMPTE ST 2022-5:2013*, pp. 1–22, Feb. 2013.

[39] "Smpte profile for use of ieee-1588 precision time protocol in professional broadcast applications," *SMPTE ST 2059-2:2015*, pp. 1–19, Apr. 2015.

[40] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated services," Tech. Rep., Dec. 1998.

[41] B. Davie, A. Charny, J. Bennet, K. Benson, J.-Y. L. Boudec, W. Courtney, S. Davari, V. Firoiu, and D. Stiliadis, "An expedited forwarding phb (per-hop behavior)," Tech. Rep., Mar. 2002.

[42] RED.COM, "RED Weapon/Epic-W operation guide," Jun. 2017, accessed July 12, 2017. [Online]. Available: http://docs.red.com.s3.amazonaws.com/955-0138/REV_K/PDF/955-0138_v6.3%20REV-K%20%20%20RED%20PS%2C%20WEAPON-EPIC-W%20Operation%20Guide.pdf

[43] "IEEE standard for ethernet," *IEEE Std 802.3-2015 (Revision of IEEE Std 802.3-2012)*, pp. 1–4017, Mar. 2016.

[44] N. Alachiotis, S. A. Berger, and A. Stamatakis, "Efficient PC-FPGA communication over gigabit ethernet," in *2010 IEEE 10th International Conference on Computer and Information Technology (CIT)*. IEEE, 2010, pp. 1727–1734.

[45] S. Bradner, "Benchmarking terminology for network interconnection devices," Tech. Rep., 1991.

[46] S. Bradner and J. McQuaid, "Benchmarking methodology for network interconnect devices," Tech. Rep., 1999.

[47] P. Emmerich, S. Gallenmüller, D. Raumer, F. Wohlfart, and G. Carle, "Moongen: A scriptable high-speed packet generator," in *Proceedings of the 2015 ACM Conference on Internet Measurement Conference*. ACM, 2015, pp. 275–287.

[48] P. Emmerich, S. Gallenmüller, G. Antichi, A. W. Moore, and G. Carle, "Mind the gap: A comparison of software packet generators," in *Proceedings of the Symposium on Architectures for Networking and Communications Systems*. IEEE Press, 2017, pp. 191–203.

[49] A. Löfgren, L. Lodesten, S. Sjöholm, and H. Hansson, "An analysis of fpga-based udp/ip stack parallelism for embedded ethernet connectivity," in *NORCHIP Conference, 2005. 23rd*. IEEE, 2005, pp. 94–97.

[50] J. Postel *et al.*, "Rfc 791: Internet protocol," Sep. 1981.

[51] E. Haleplidis, K. Pentikousis, S. Denazis, J. H. Salim, D. Meyer, and O. Koufopavlou, "Rfc 7426: Internet protocol," Jan. 2015.

[52] International Telecommunication Union, "Parameter values for UHDTV systems for production and international programme exchange, ITU-R BT.2020-2," 2012.