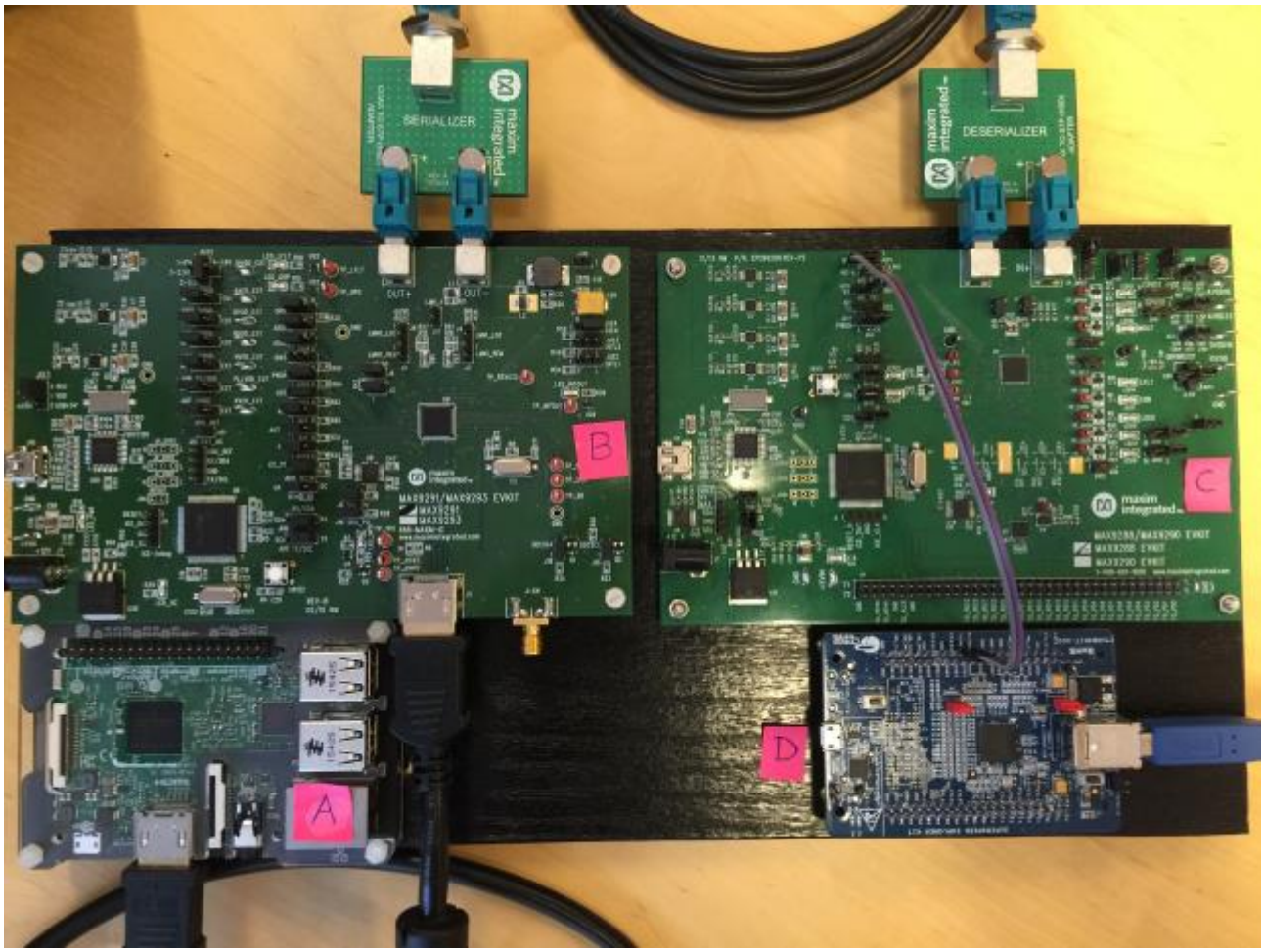
CHALMERS
UNIVERSITY OF TECHNOLOGY



LVDS Gateway for Image Sensors

Wilhelm Råbergh
Joel Norell

Bachelor's thesis in electrical engineering.

Department of Signals and Systems
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2016

PREFACE

This is a Bachelor's thesis as a part of the last year of the program in Electrical Engineering at Chalmers University of Technology in Gothenburg. This project has been carried out in behalf of Diadrom in Gothenburg with the purpose of investigating the possibilities of an LVDS gateway for high speed video stream and control channel. The project has mainly been carried out at Diadrom's office in Gothenburg.

We would like to express our gratitude to our supervisor, Henrik Fagrell at Diadrom for the opportunity to carry through our thesis work with his help and guidance in the project. We would like to thank Manne Stenberg at Chalmers for being our mentor.

We would also like to thank Jonas Hellberg at Diadrom for supplying us with hardware and for his patience and help.

ABSTRACT

The amount of electronic equipment in vehicles today is huge, there are computers controlling and monitoring almost every moving part and there are more adding up for every new model and update. With all systems together, cars are closing in to autonomous driving and it is all about enhancing the experience with functionality, comfort and safety. One of many important systems in the development of these vehicles is the camera with multiple purposes and functions. Implementation of cameras to existing infotainment systems often require modifications of the software and hardware, in this case a camera with LVDS (Low Voltage Differential Signalling) interface. The LVDS interface provides a high speed video stream and great noise cancelling. The camera today is known to help the driver get a better view of the surroundings or simplify the driving in narrow places which is why it is quite common with a rear-view camera in cars today. But as the systems in the vehicles get more complex and closing in to autonomous, the camera is no longer just a reversing aid. The camera itself has not necessarily changed, but the purpose of the information it is providing. With advanced imaging software, other systems in the vehicle can make use of the information to. This is why the high speed data transfer is becoming more and more important, and the LVDS camera is a great asset.

The purpose is to analyse the possibilities of implementing an LVDS gateway for video and control channel transfer over USB 3.0. An UART control channel over USB 3.0 to a camera was successfully implemented. A “proof of concept” with high speed video transfer over USB 3.0 was accomplished.

TABLE OF CONTENTS

GLOSSARY

ADAS	Advanced Driver Assistance Systems
LVDS	Low Voltage Differential Signalling
GMSL	Gigabit Multimedia Serial Link
HDMI	High Definition Multimedia Interface
CSI-2	Camera Serial Interface, version 2
STP	Shielded Twisted Pair
HSD	High Speed Data
USB 3.0	Universal Serial Bus, version 3.0
FFC	Flat Flex Connector
IC	Integrated Circuit
FIFO	First in - First out
SerDes	A composition of a serializer and associated deserializer.
UART	Universal Asynchronous Receiver/Transmitter
I2C	Inter-Integrated Circuit
LIDAR	Light Detection and Ranging
RADAR	Radio Detection and Ranging

1. INTRODUCTION

In this project we will discuss our research of the possibilities of simplifying the implementation of cameras in vehicles, primarily image sensors with LVDS interface. The idea is to convert the LVDS video stream into a more common interface to make it possible to implement a fully functioning camera system into an infotainment system with lack of LVDS communication. This would make the whole process of camera implementation in previous systems easier and minimize the needed hardware and software modification of it. There are also some performance requirements on the systems, both aesthetically and as safety measures. The system need to be able to run with the same image quality and at the same time has as little data loss and as little delay as possible, this is also something that we will need to take in consideration.

1.1 Background

The automotive industry is experiencing a massive transformation where a couple of young manufacturers has grown to change the concept of what a car is. We are talking about the return of the electrical motor, and not just the quite common hybrid cars. But the pure electric cars with the same specifications and performance as the cars with petrol and diesel engines that we are familiar with, and that is not all. There are cars with functions that can be described as autopilot, where the driver has more of a monitoring role rather than actually driving the car. The fact that a lot of these manufacturers also is experimenting with fully autonomous cars tells us that we should familiarize with the idea of all being passengers in a car. Along with the progress of this huge transformation of the automotive industry there are a lot of new equipment which needs to be fitted into the cars, where one category is sensors of different types.

The given assignment is an LVDS gateway with the intention to simplify implementation of LVDS image sensors in already existing infotainment systems without support for LVDS. The project is highly relevant as explained before, since a lot of car manufacturers today are working on their systems for autonomous driving cars. These cars are fitted with different kinds of sensors that closely monitor the area around the car for example cameras, radar and LIDAR. The cameras which have been used as a complement to the rear-view mirror now provides information for other systems in the car. Radar can be used to determine distance to a vehicle ahead using radio waves. Lidar which functions similar as radar, only with light. More precisely with lasers. Lidar sends a pulse with a laser, when measuring the delay of the reflection it is possible to determine the distance to the object [14].

Diadrom wants to research the possibility to simplify the implementation of cameras in vehicles, focusing on the car industry, while maintaining the same quality and responsiveness that the already existing system have. Companies that develop and manufacture cars today have to customize their own systems if they want to have a rear view camera in a car and that process can be very time consuming and expensive, therefore making a rear view camera system that you can implement into already existing infotainment systems would make this process a whole lot easier, a lot cheaper and less time consuming. This would mean that car manufacturers with already existing infotainment systems could implement a camera without the need of modifying their own hardware, only implement some new software to show the image on the infotainment system screen.

1.2 Purpose

- Investigate the possibilities of an LVDS gateway to a more common interface. Can a high speed video stream with including control channel be gatewayed and still be able to maintain the same quality and data rate with very little data loss?
- Implement a suitable control channel in the gateway and verify its functionality.
- Implement the video stream together with its control channel and verify its functionality.

Research if it is possible to simplify the already existing rear view camera system that use LVDS and create a gateway where the video output stream is connected to the infotainment system via already existing interfaces in the infotainment system for a more modular and cheaper implementation for new and existing systems.

1.3 Limitations

There will be no image processing or editing of the data, only data conversion. It will be the same data in as out but with different interfaces.

The request from Diadrom is an LVDS gateway, preferably with USB 3.0 and an embedded UART control channel.

A pre-configured USB 3.0 board will be acquired, preferably with existing drivers and to some extent code example. This to minimize time spent on just getting the USB module up and running.

1.4 Problem

Which standardized interfaces are available that can match the data rate of LVDS?
Does the interface support both video stream and control channel?
Can you simplify video transfer interface LVDS with conversion to USB 3.0, maintain same quality, have very little data loss and keep the delay small?

2. TECHNICAL BACKGROUND

This section presents the technical facts and conclusions for the project and intends to give clarity for what applications it might be useful.

2.1 Differential signalling

Differential signalling refers to a method of transmitting information rather than an interface. A differential pair of the same signal is sent through separate wires with opposite polarity, the receiver compares the voltage between the two signals. Compared to the more traditionally used single ended signalling with a common ground as reference it can withstand noise and electromagnetic interference in a more effective way. The signals with opposite polarity is affected in the exact same way by external interference and the receiver which compares the difference between the two signals will have a high accuracy.

2.1.1 LVDS - Low voltage differential signalling

LVDS is a high speed data transfer interface that has electrical characteristics of a differential, serial communications protocol. It has become a commonly used interface in a wide range of industries, automotive, aerospace and defence for example. Some of the reasons this interface is so widely adopted by different industries might be because of its low power consumption and really high data rate with high noise immunity, but also its robustness and inexpensive components [10].

The big difference between LVDS and other more traditional communication protocols is transmitting the signal over two lines instead of one. It produces two signals with opposite polarity and on the receiver end, the two signals are referenced against each other compared to the traditional way of comparing the one signal to a common ground. This gives a noise immunity and data transfer quality that a traditional protocol cannot provide [11].
One might think that LVDS compared to more common data transfer protocols require the double amount of wires but with the high data rate it actually use less. To achieve the same data rate with one of the common ones it would be necessary to use a parallel interface which means a lot more wires. LVDS with its serial interface is mostly used with a combined SerDes were the parallel data is serialized and transmitted through a STP cable and then deserialized on the receiver end into parallel data.

2.1.2 Typical LVDS setup

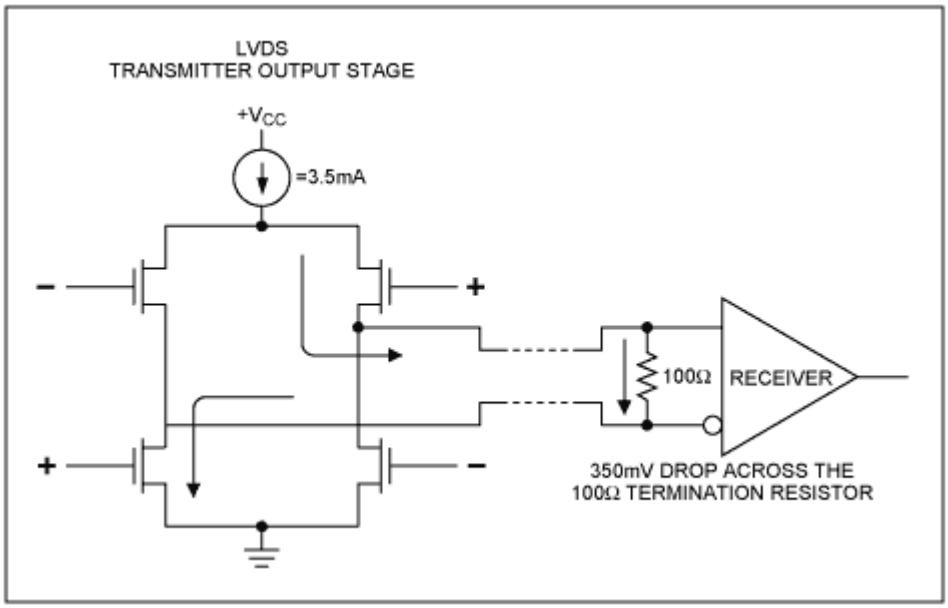


Figure 1. LVDS transmitter and receiver.

The transmitter, also referred to as the driver has a current-mode driver that delivers 3.5 mA through the STP output over the transmission lines to the receiver. The transmission line often consists of a coax cable with 100 ohms' impedance, to keep good signal integrity it is very important to have a close impedance matching at the receiver's end [11]. The characteristics of the termination resistor helps maintaining the quality of the signal and reduces noise and other interference which arise from reflections.

2.2 GMSL

Gigabit Multimedia Serial Link is a LVDS composition intended for the automotive industry designed by Maxim Integrated. It is primarily created with focus on video applications in the ADAS category for high precision and low cost. The video stream also includes an embedded control channel which enables UART communication. It is compatible with STP cabling and can be used with different interfaces in each of the SerDes ends [12].

2.3 USB 3.0 - Universal Serial Bus Version 3.0

Superspeed USB, or more commonly known as USB 3.0 is the more updated and faster version of one of the most commonly used communication standards in computers and home electronics. The major update compared to the predecessor USB 2.0 is the much higher transfer rate of 5 Gbit/s compared to 480 Mbit/s, hence the name Superspeed.

2.4 HDMI - High Definition Multimedia Interface

A multimedia interface for transferring uncompressed digital audio and video in one cable. It is probably most common in our homes connected to our television and delivering multimedia from our computers, TV decoders and what else one might be interested to pair with the TV. The main drawback with HDMI is the expensive cabling which has resulted in several other popular interfaces in high volume industries such as Automotive [13].

2.5 MIPI CSI-2 - Camera Serial Interface

CSI-2 is a camera interface that is often used in the mobile industry because the need of a robust, low-power, high-speed and cost-effective standard. But this also fit in very well with the automotive industry now when autonomous driving becomes something that many of the coming cars will have. The autonomous function will then need a lot of cameras and sensors which will increase the need of the CSI-2 standard.

CSI-2 protocol is layered in the Application and transport layer in the OSI model and supports D-PHY and C-PHY in the physical layer. It is possible to choose between 1 to 4 of data transfer lanes and each lane can typically transmit 1.5-1.78 Gbps depending on which physical layer(C-PHY/D-PHY) you use. This can differ a lot depending on what type of microprocessor you use which can result on both less and in some cases more bandwidth [7][8].

The layers are sorted in the following order:

1. Application layer
2. Pixel to Byte Conversion
3. Low Level Protocol Layer
4. Lane Merger Layer
5. Physical Layer (C-PHY/D-PHY)

2.6 Cable properties

Cable	Max length (m)	Bandwidth (Gbps)
USB 3.0 (3.1)	3	5 (10)
LVDS (STP)	>18	3

[16].

3. METHOD

This is a research project with the purpose of investigating how an LVDS gateway can be designed within the criteria of speed and quality. The results will be highly affected by the hardware and software because of the transition from one interface to another.

A serializer evaluation kit is used to simulate the camera which is connected to a deserializer evaluation kit via an STP cable to establish the LVDS link. The serializer is used instead of the camera to enable features like different video and control channel interfaces. It also makes troubleshooting of the data easier. It was noticed that I2C was set as standard control channel in favour of the featured interfaces. This meant that the video source had to be forced to a specific resolution to cope with the UART control channel and for the LVDS link to be active. The specific video source is created with a Raspberry Pi by changing the configuration settings which then results in a forced HDMI output at start-up.

Research and initial testing of the current setup is done to determine which functionality is needed to supply the system with USB 3.0 connectivity. It is found that a connection can be established with either MIPI CSI-2 or Parallel RGB888. A development kit for USB 3.0 connectivity with an ARM processor and a FPGA is acquired to potentially complete the systems hardware setup. With a HDMI input and USB 3.0 output, the development kit delivers the video from the source to a PC. The implementation of the USB 3.0 development kit to LVDS setup is begun by connecting the UART control channel to the FPGA, throughout to the ARM processor. Implementation of the video stream with FFC cables and adaptors struggles due to the high signal integrity required.

Unfortunately, at this stage the ARM processor on the development kit enters a bootloader mode from which there is no recovery. For a possible reset of the processor, disassembly of the board is necessary for which there is no time.

A different kind of USB 3.0 development kit is acquired with same capabilities but with lack of an FPGA. The new board has all I/O's accessible which enables full reset.
The UART control channel is implemented in the ARM processor of the USB 3.0 development kit and connected to the rest of the system. Drivers on a PC is matched with the UART over USB 3.0 and a fully functioning UART control channel is setup. It is now possible to communicate with the camera from a PC.

4 HARDWARE DESCRIPTION

The main component in this project is obviously the LVDS-camera but while doing the research and development, evaluation kits and development boards are used to enable full access to the video stream and control channel. Apart from that, at least one computer is needed for receiving output from the system. Although in the testing phase of the project a HDMI source is going to simulate the camera where the HDMI input will be converted into a LVDS link. The hardware setup is shown in Figure 2.

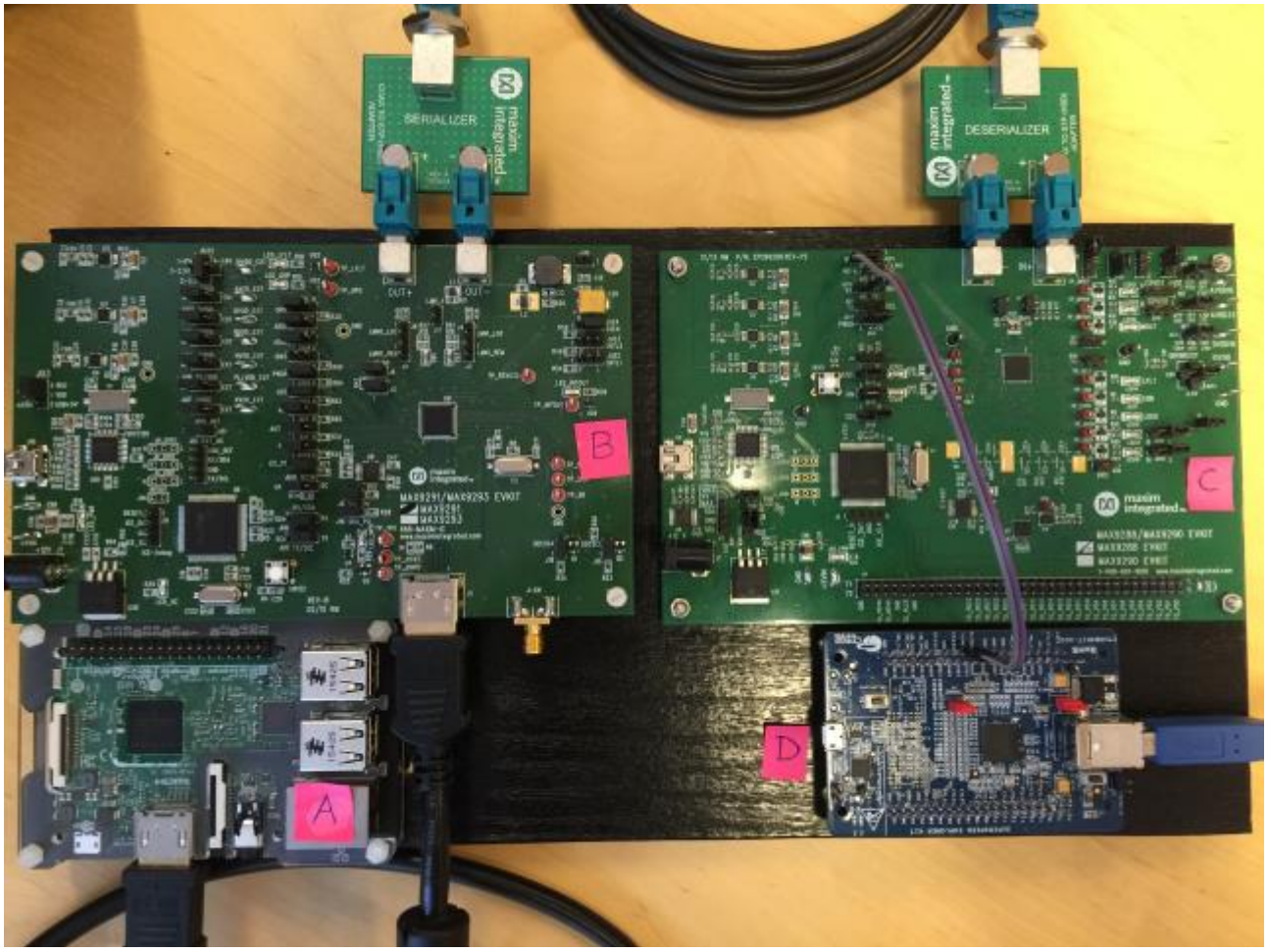


Figure 2. Hardware setup containing the following.

A: RPI3 microprocessor unit from Raspberry Pi.
B: MAX9291 serializer evaluation kit from Maxim Integrated.
C: MAX9288 deserializer evaluation kit from Maxim Integrated.
D: USB 3.0 SuperSpeed explorer kit from Cypress Semiconductor Corp.

4.1 MAX9291 Serializer

The evaluation kit is intended to substitute the LVDS components of the camera during the research to enable outputs of video stream and control channel when debugging. Fitted on the board amongst other things is a MAX9291 IC described in Figure 3, an HDMI input and dual coax FAKRA outputs. The HDMI input signal is converted to a GMSL output for transmission of the video stream with an STP cable. The coax FAKRA outputs are connected to a serializer adapter which enables the coax output. The LVDS link constitutes of the serial GMSL output over the STP cable.

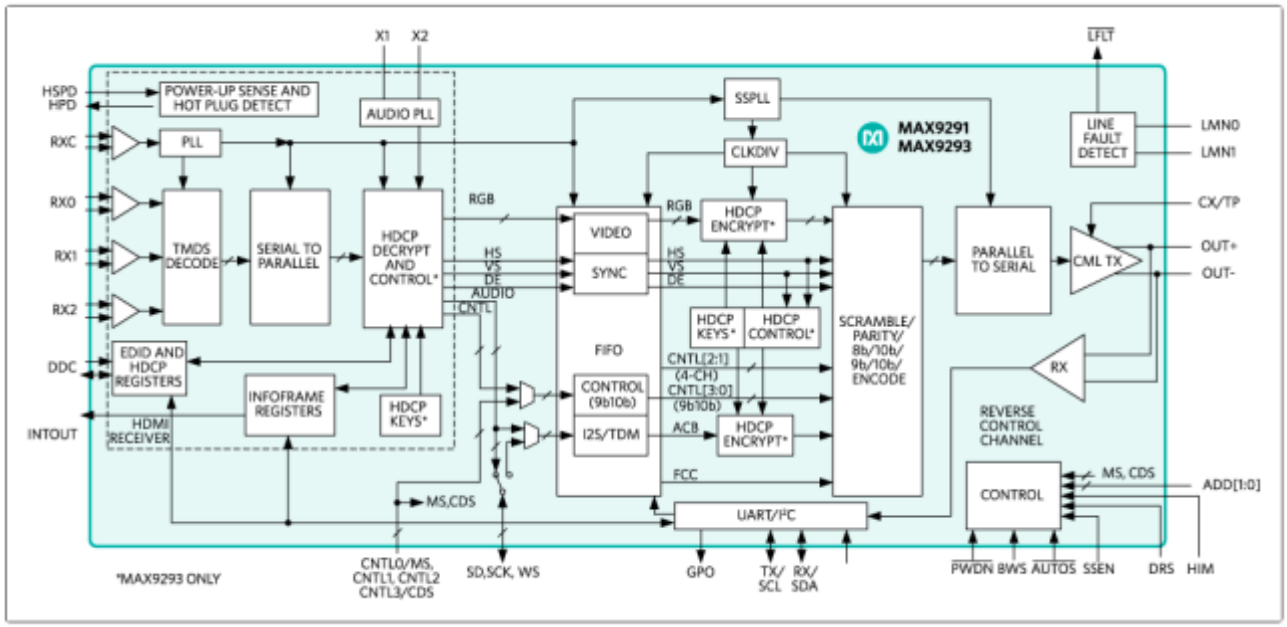


Figure 3. MAX9291 functional diagram.[1]

The MAX9291 is compatible with both UART and I2C control channel and has an LVDS output which is similar to the actual camera. Figure 3 reveals a detailed diagram over the conversion from the HDMI input signal to a LVDS output link.

[1] Copyright Maxim Integrated Products (http://maximintegrated.com). Used by permission.

4.2 MAX9288 Deserializer

The MAX9288 chip is used to convert the GSML input to a four lane MIPI CSI-2 output as shown in figure 4. The evaluation kit is built to match the MAX9291 evaluation kit as a fully functioning SerDes system. It is also fitted with a chip for converting the CSI-2 output to a parallel output and with associated defined pins. Maxim Integrated has also provided a GUI, compatible with the SerDes where the settings of the setup can be managed. One can choose what interface to use for the control channel, or which video format is preferred.

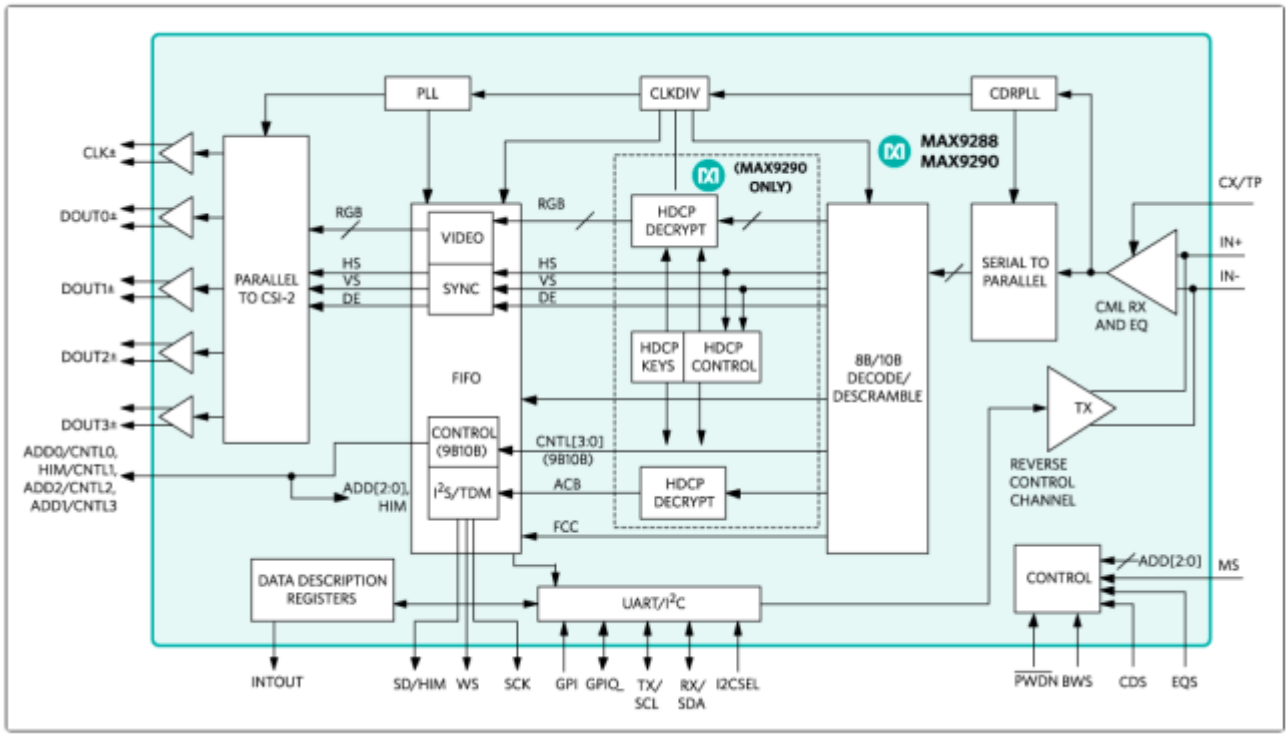


Figure 4. MAX9288 Functional Diagram.[2]

4.3 STP to Coax Adapters

The output from the MAX9291 evaluation kit and the input to the MAX9288 evaluation kit is the two coaxial FAKRA connectors. The FAKRA connectors is designed for high speed data transfer and intended for use in the automotive industry. To establish the SerDes link, the two coaxial FAKRA connectors has to be merged which is done with an adapter from coax to STP. On the other end of the link there is a corresponding adapter to separate the STP connector to the two coaxial inputs. The two adapters are then connected to each other using a STP cable with close impedance matching for maximum noise cancelling.

[2] Copyright Maxim Integrated Products (http://maximintegrated.com). Used by permission.

4.4 Lattice USB3

This manufacturer built kit features the EZ-USB FX3 chip which is built around an ARM9 processor with added functionality to provide USB 3.0 connectivity. It also comes with an FPGA which is interfacing the FX3 chip and provides the possibility to implement different kinds of image sensors. The board is shown in figure 5[1].

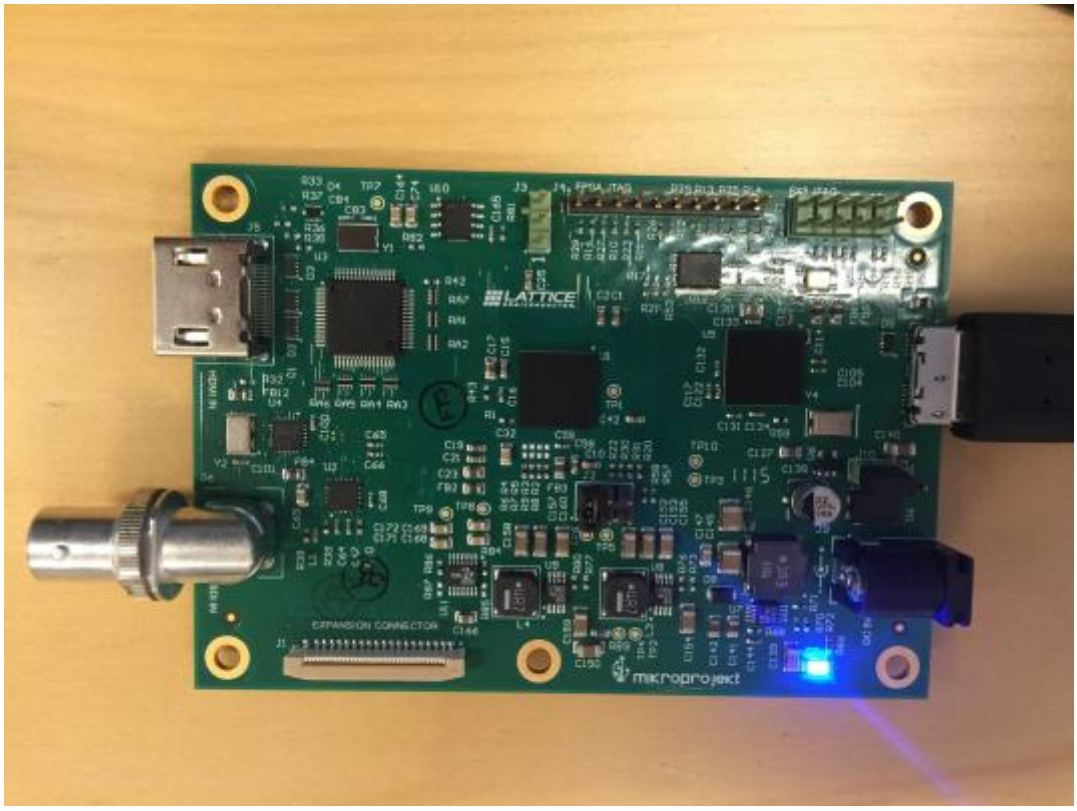


Figure 5. The Lattice USB3 video bridge.

4.5 SuperSpeed Explorer Kit

This manufacturer built kit features the EZ-USB FX3 chip which is built around an ARM9 processor with added functionality to provide USB 3.0 connectivity. which is supposed to be the LVDS gateway. It is fitted with a custom GPIO interface called GPIF II which has capability of higher speeds than a regular GPIO interface. There are different kinds of control channel interfaces such as UART and also available resources with example code and project guides. The board is shown in figure 6[9].

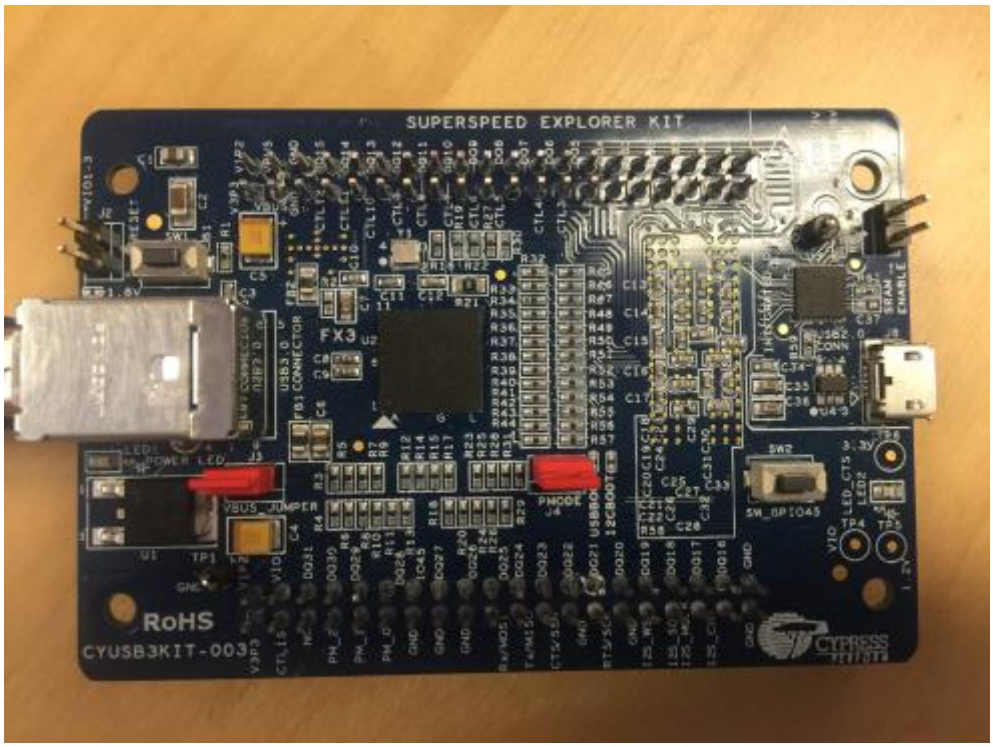


Figure 6. Cypress SuperSpeed explorer kit.

5. COMMON AVAILABLE INTERFACES

When it comes to choosing the best interface for the gateway it is crucial that the data rate of LVDS is met, it also has to be a digital interface with capability of transmitting video data. The speed of the video stream needs to be met, otherwise the system will be affected by the delay and it will not be sufficient for an LVDS camera. Since there are a lot of restrictions and regulations in the automotive industry it also should be an interface that is suitable in the environment of a car. There are good reasons for it, the system should be able to withstand temperature changes, vibrations and even damp that can occur in a car from cold seasons to hot. In terms of cost, the interface including connectors and cabling obviously has to be in a price range that does not exceed the cost of rebuilding the existing infotainment system which then would be pointless. To rule out the interfaces that does not comply to the task, we have some criteria's.

- The data rate has to be at least 3 Gbit/s.
- It should be a digital interface.
- Preferably a well-known and thoroughly tested interface durable enough for the environment in a car.

The LVDS interface can reach data rates up to 3 Gbit/s which eliminates the most common interfaces of them all, USB 2.0 with its 480 Mbit/s. The more updated version of the interface, USB 3.0 can reach data rates up to 5 Gbit/s which should be enough for our intended system. The idea with USB was to create a standard interface and physical connector for personal computers and replace the different serial and parallel interfaces used for different devices. The interface has been adapted in a wide range of industries for its simplicity and amount of universal accessories.

Another interesting interface when it comes to video streams is HDMI, with its almost monopoly like share of the market. It has grown to be one of the most commonly used interfaces for transferring video when it comes to home electronics. It has a capacity of data rates past 10 Gbit/s with the latest version 1.3 and up to 5 Gbit/s with the original version 1.0 which is enough. The disadvantage with HDMI is the overall expensive cables and connectors, and the fact that it is not that common in the automotive industry.

In terms of functionality both USB 3.0 and HDMI would be okay as interface for the intended gateway. Since USB 3.0 was eligible by Diadrom already from the start we decided to go with that interface and to start looking for all the components needed.

6. CONTROL CHANNEL

The request for the control channel is that it consists of UART interface. To make sure it works properly an initial test of the control channel is carried out. The Maxim evaluation kits are pre-configured with I2C as control channel as a result of the available interfaces on the boards. The MAX9291 that are used to simulate the LVDS camera is equipped with HDMI input, which has I2C embedded in the interface as control channel. The MAX9288 that is our LVDS deserializer with MIPI CSI-2 output which is also predefined with I2C as standard control channel. However, both kits support UART communication with a couple of setting changes. According to the datasheet this can be done by changing the position of jumpers directly on the board. Maxim is also providing an API to communicate with the board via USB. The API provides functionality to update the LVDS chipset register settings however, these settings only last as long as the power supply is connected. After a reset the board will use the settings according to the jumpers.

6.1 UART

Universal Asynchronous Receiver/Transmitter, more commonly known as UART is a serial communications protocol often used in computers and other circuits. Its purpose is to send and receive data either between ICs or two different systems for example in our case, a PC and a camera. There are also a number of types of UART which makes it useful in different setups. The parallel data are converted into a serial sequence of data bits sent through a FIFO buffer and then deserialized by the receiver.

6.1.1 Communication

There are different kinds of UART communication which differ in their extensiveness, referred to as Simplex, Duplex and Full Duplex. Simplex is used for communication in only one direction with no possibilities for the receiver to send any feedback. There is half duplex with communication both ways one at the time, and full duplex capable of communicating both ways at the same time. The setup is usually consisting of three signals, Tx(transmitter), Rx(receiver) and ground for reference. Where the Tx signal on one side is connected to the Rx signal on the other side of the communication as in figure 7.

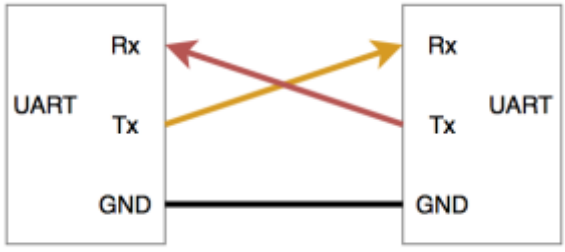


Figure 7. Interaction of UART.

6.1.2 Data sequence

There are a couple of factors and settings for the UART protocol that can differ from device to device which can be crucial to follow to make the communication work properly. Since it is an asynchronous protocol, meaning there are no actual clock signal to reference the data sent it is important to use start and stop bits. If the signal is high in idle, the start bit should be low and indicate that there are data incoming to the receiver. The actual data normally consist of 8 data bits but can sometimes also be defined by either manufacturer or user to be more or less data, depending on use. A sequence of data bits is followed by a stop bit to indicate that all the data has been received.

6.1.3 Parity bit

The parity bit (also called "check bit") is often used at the end of a string of bits (a byte for example) and it functions as a failure detection and indicates if there have been any faults during the transmission. The parity bit tells the receiver whether the 1-bits in the byte should be even (1) or odd (0).

In some cases, it is required to have a parity bit, it can be even or odd. The reason for the extra bit is to tell the receiver whether it should count all the bits with a value of 1(even parity) or with a value of 0(odd parity).

6.1.4 Baud rate

Baud or Baud rate is a unit for symbol rate corresponding to how many changes per second a signal can do. There are standards for the Baud in electronics from 300 to 256000 and in modern devices it can even be up to 1M. In theory it could be even greater, as long as the device has the capability of working in at the same frequency. What is important is that both ends of the communication has the same Baud, otherwise the data will be faulty on the receiving end.

6.2 Setting up UART on the Maxim evaluation kits

As mentioned before the evaluation kits from Maxim is preconfigured with I2C as primarily control channel because of the data types and interfaces available. After studying the datasheets for each evaluation kit it was found that there are designated pins for UART which makes it possible to setup a system with an UART control channel.

6.2.1 FTDI (USB to serial)

To interface the UART to the PC we are using an FTDI cable which connects directly to the pins on the board to the USB port on the PC. FTDI (Future Technology Device International) is actually the company who is manufacturing the cables which features a TTL to USB converter for serial UART communication. FTDI produces a range of cables with different setups where we are using the TTL-232R-3V3 model, but they are often referred to as just FTDI cables.

TTL(Transistor-Transistor Logic) is a type of bipolar transistor ICs, 232R is the name of the chip used in the cable and 3V3 is referring to the voltage level.

Figure 8. Basics of the FTDI cable.

6.2.2 HTerm

On the PC we are using a program called HTerm which is an emulator for the serial interface also known as COM (Communication port). COM ports do still occur, mainly on desktop computers but there are many alternative ways to use it. As mentioned above we are using an FTDI cable which can be referred to as an COM adapter. HTerm is a software that is easy to use for this purpose because of its layout. All the settings we need to think about is shown directly on the screen which can be seen below in figure 9.

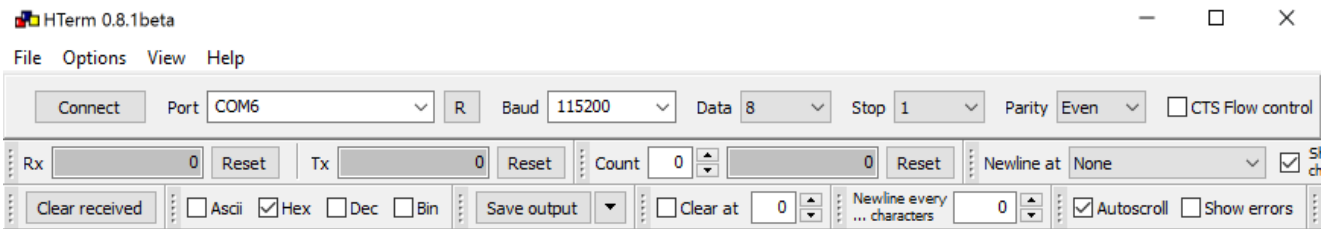


Figure 9. Settings for UART communication in HTerm.

On the top row of the layout the baud rate can be changed, the number of data bits can be set, number of stop bits that should be used and also the parity bit.

6.2.3 Communicating with MAX9288

To initiate UART communication with the MAX9288 chip it uses a sync byte specified as Hex (0x79) to set the correct baud rate of the SerDes. All Maxim's chips have the same sync byte, in case a system is using more than one chip (which this system are) this would cause a problem since the UART buses often are merged together. To avoid this problem Maxim has added what they call a device ID byte which depends on the type of chip. Basically it is a device ID used to address a specific chip in a larger design and for the MAX9288 it is specified as Hex (0x91) by default. This is followed by the register address and then the number of bytes to be read or written.

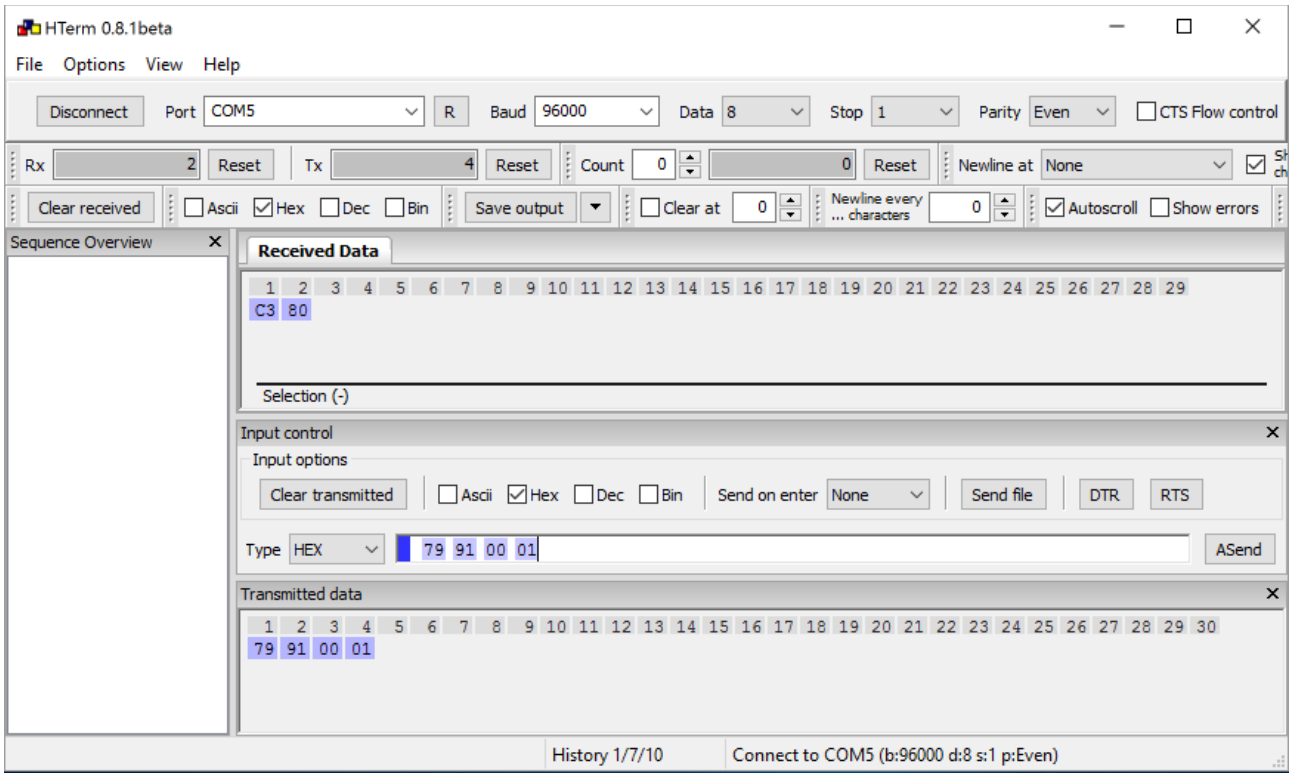


Figure 10. UART communication with MAX9288.

Figure 7 shows a successful communication with the MAX9288 over UART where the chip is called with the line Hex (0x79 0x91 0x00 0x01). The chip should respond with whatever is stored in the address, and Hex (0xC3 0x80) is the expected data where Hex (0xC3) is the acknowledge byte and Hex (0x80) is the response data byte.

7. VIDEO STREAM

7.1 Lattice Demo

After the arrival of the Lattice development kit it was of interest to make sure the board worked as intended. Therefore, a test was set up with the demo code provided by the manufacturer. Provided was also a GUI called "Lattice USB 3.0 Video Bridge Configurator" with changeable settings to enable configuration of the board. The device was configured with HDMI input and a resolution of 1920x1080p at 60 FPS. When the board was configured and everything was connected we got a video stream from the USB 3.0 which was captured by another program called "AMCap". We found that when we had 1920x1080p resolution we got the FPS to be around 35-38 FPS and 58-60 FPS while having a lower resolution of 1280x720. Depending on what type of screen you are using in a potential infotainment system something it might be more suitable to have lower resolution and get higher frame rate than having the opposite, because a car for example moves and therefore it is favourable to have higher frame rate and lower delay. This might change depending on the application this system is used for [2].

7.2 Camera simulation

This project does not use a real camera when we are building our test rig, instead we are using a board from Maxim Integrated which will simulate the functionality of a camera. MAX9291 evaluation kit do not have a way to create an image itself therefore it needs to get an image from an external source. In this case we'll use the HDMI input on the Maxim board and a Raspberry Pi 3 to send the video stream, also using its HDMI interface, into the MAX9291 evaluation kit [15].

7.2.1 Raspberry Pi 3

The Raspberry Pi 3 is used as a part of the simulation of an image sensor and is forced to a specific resolution. This is necessary for a couple of reasons:

- To implement a pixel clock
- Show an image so that we know that the video stream works.
- Enables the UART communication between the two Maxim boards, 9288 and 9291 evaluation kits.

A pixel clock is required to activate the link between the maxim boards to which is crucial, otherwise nothing can be sent over the LVDS link, no picture nor control signals. The control channel will not reach the Raspberry Pi which is not necessary though its functionality is only to supply the LVDS SerDes with a pixel clock and image source. The serializer is then the endpoint of which the communication will reach, this is also the part of the setup where an actual camera would be connected. The image that will be forced is in the resolution of

640x480 at 60Hz and has a pixel clock of 24 MHz. The image itself is just a spectrum of colours but this is just to make sure the video stream works as intended.

7.2.2 MAX9291 evaluation kit

The MAX 9291 serializer is a part of the MAX9291 evaluation kit and its role in the camera simulation is to work as the "computer" and it makes sure that the right format is sent over the LVDS link and also enables the UART communication. The Board needs a pixel clock which is provided by the Raspberry Pi as mentioned in the previous section to enable the communication both video stream and the control channel, UART.
In our project we will use the registers in the MAX 9291 serializer as a reference when we are going to test the UART communication. We are going to send a few bytes of data [x2] to the address of a specific register and in case the UART communication works as intended the microcontroller will respond with a certain string of bytes and by knowing what we expecting as an answer we can determine whether the control channel works or not.
The Video stream will be generated by the Raspberry Pi and then sent out via the HDMI interface into the MAX9291 board and then it will be converted and serialized so that it can be sent out to the next board via the LVDS interface.

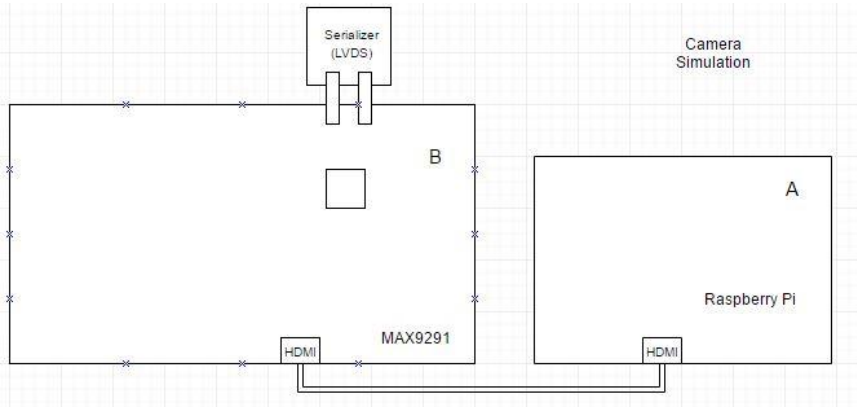


Figure 11. Block schematic of the Camera simulation.

7.3 Video conversion

This project includes a lot of different video formats and interfaces and each one of the boards we have included serves its own purpose and handles one thing or another when it comes to video conversion. Although we will not be coding and construction all of the conversions that is being used due to time limitations, therefore in a few of the steps we'll be using already finished code.

7.3.1 MAX9291 evaluation kit - conversion

The MAX 9291 serializer is a part of the MAX9291 evaluation kit and this serves as the HDMI to LVDS converter and is also part of the simulated camera part and will therefore be replaced if this project was to be realised and build with real hardware. The software on the MAX 9291 serializer is already finish and the only thing we are doing on this board is to configure the settings and making sure we can talk to the registers and getting the right respon se so that we know the UART communication is working.

7.3.2 MAX9288 evaluation kit - conversion

The MAX 9288 deserializer is a part of the MAX9288 evaluation kit and serves as the LVDS to CSI-2 and parallel using RBG888. This board is constructed as a test board and it is the parallel interface that has pins as output but they board is made so that you could also use the CSI-2 interface if you would prefer that, although you would need to remove the parallel interface part of the board otherwise you might get stability issues when you are running at very high frequency.

7.3.3 Lattice USB 3.0 Video Bridge Development Kit

Lattice USB 3.0 Video Bridge Development Kit is a fairly closed system where cannot change very much in the hardware, except power source and voltage levels, so on this board we will mainly focus on software development unlike to the two previous boards where we have focused on the hardware. This board contain several inputs for the video stream and they include, HDMI, Expansion Connection (for CSI-2, LVDS etc.) and a SDI Equalizer, and it got one output, which is the USB 3.0 connection.

Lattice USB 3.0 Video Bridge Development Kit is designed into two different parts, the first one being the FPGA and the second being the microcontroller. The FPGA is a Lattice ECP3-17EA and the microcontroller is a Cypress EZ-USB FX3, they will handle the video conversion from CSI-2/parallel RGB 888 to USB 3.0 that will be sent into the Computer Host where the video can be viewed [3].

7.3.4 Lattice FPGA ECP3-17EA

The FPGA from lattice that the lattice board uses are the first part of the conversion it is in this chip the control signal and the video stream merge and are sent together to the Computer-host via the FX3 microcontroller and the USB 3.0. In the demo of the lattice board we used HDMI as input but in the real tests we want to use the Expansion connector instead because we want to use CSI-2 that we have as an output from the MAX9288 board. [4][5]

7.3.5 Cypress EZ-USB FX3

The microcontroller on the Lattice boards is a EZ-USB FX3 from Cypress that is going to handle the last part of the conversion and also going to handle the USB → PC interface. The chip has an interface called General Programmable Interface or GPIF™ II and is configurable for a wide variety of processors, ASIC's, image sensor's or FPGA's. In the Lattice Video Bridge dev kit, it is already connected to a FPGA but in the EZ-USB FX3 SuperSpeed Explorer Kit you have the option to configure and connect your own device depending on what type of video device you have. After the GPIF™ II interface the FX3 chip then packet the data and make it compatible with USB 3.0.[6]

8. RESULTS

- The system was successfully modified to work with an UART control channel. Full duplex communication over USB 3.0 between a PC and the camera module is possible.
- A "proof of concept" was performed with data conversion from HDMI to USB 3.0 with a video stream quality of 1080p, 30fps and 720p, 60fps.

9. DISCUSSION

This project has been a difficult one due to a lot of the hardware has been fairly new and the information available has been limited, although this also meant that we have been forced to read and learn a lot of new things which is great experience for future projects. The goals that we set up in the beginning of the project has been met, we made a proof of concept that worked but we did not manage to put everything together and try the system as a whole. Because some of the hardware from Maxim Integrated did not have complete datasheets, we spent a lot of time figuring out how the image interfaces (video stream out from the Maxim 9288 board) was configured. The lack of pin configuration in the datasheet made so that we couldn't use the parallel(RGB888) or the CSI-2 interface from the maxim 9288 board and this forced us to skip this step and focusing more on the control channel.

The Lattice Video Bridge development kit that we ordered worked fine when we tested the demo on it and the FPGA was easy to program and worked fine but the FX3 chip wasn't as easy to work with in this board. When we started working with the FX3 chip we loaded by mistake a .zip file into the fx3 and then it got stuck in a bootloader state which could not be fixed because all the hardware pins were not available and therefore render the lattice board useless, and we had to order a new different board. We tried to contact Lattice and see if they had a solution to this but we never got any response.

Cypress FX3 SuperSpeed Explorer Kit was the other board we ordered and this was much better to work with due to being more flexible with all the pins available and the program or firmware that you downloaded (to the FX3) was by default in the RAM so that a simple power reset always reset the device so it never got stuck like the previous did. FX3 also had a lot more example software for testing different functions, like UART communication and video stream examples. One big drawback was that the FX3 board did not have a finished video stream interface like, HDMI or CSI-2 so that was a trade-off for us so that we were able to finish the project in time with all the test that was remaining.

All of these hardware problems was not something we could have known before and it is also something that often or maybe always occur when you are working with hardware so planning the project more before starting or maybe have a backup plan for when hardware fails might be something to have in mind for future project.

The project also changed along the way when we found new problems with both hardware and software, this in return changes the project and may alter the original question. One of the first and maybe the most important question was:

"Are there interfaces available that can match the data rate of LVDS"

This was one of the harder one to test out due to hardware problems but the data transfer rates on both the hardware we got from Maxim Integrated and the boards we ordered it was sufficient enough to run at fairly high resolution. We tested the Lattice Development kit and we got 35-38 fps at 1920x1080p resolution and if you lower the resolution to 1280x720p you get 60+ fps which is more than enough to run the camera smoothly.

Choosing the resolution depends on what type of application the camera is going to be used for but if we use the camera as a rear view camera 1280x720p with higher fps and low delay is more preferable because the vehicle is in movement. When you are reversing your car you want it to update quickly so that you don't miss if anything comes in the way.

The USB 3.0 is a versatile interface which you can use for many types of applications and a lot of different types of data, because of this transferring a video stream were not hard to achieve. To split the USB 3 into two different devices that could serve as a video stream and a control channel was not as easy as doing one of the thing on their own, because we had to use something called a "composite device" we had to use a code that was a bit more complicated which also made it harder to edit and get the functionality we wanted. If we changed to much in the code the host PC did not recognize the drivers for either the FX3 video stream or the USB 3 COMPORT, so we had a lot of problems trying to alter the functionality of an already existing program so that it fit our specification. This was the last thing we tried to achieve so that we could have something more "real" to show as a demonstration but we were only able to show each function on its own and not a working composite device. Although we got a composite device to work and got both a FX3 video stream device and a COM-PORT device running the functionality wasn't as we intended due to the COMPORT was a loopback function and did not send anything out via the UART bus.

We reached our goals in the form of a proof of concept but unfortunately we were not able to get the demonstration up and running to show how a video stream could run alongside a separate UART control channel.

10. REFERENCES

1. Lattice Semiconductor (2016) [Retrieved 2016-04-26] Lattice USB3 Video Bridge Development kit retrieved from: http://www.latticesemi.com/en/Products/DevelopmentBoardsAndKits/LatticeUSB3VideoBridgeDevelopmentKit.aspx

2. Lattice Semiconductor (2016) [Retrieved 2016-04-26] Lattice USB3 Video Bridge Development Kit Quick Start Guide - QS024 - version 1.0 - 2014-09-30. Retrieved from: http://www.latticesemi.com/en/Products/DevelopmentBoardsAndKits/LatticeUSB3VideoBridgeDevelopmentKit.aspx

3. Lattice Semiconductor (2016) [Retrieved 2016-04-26] Lattice USB3 Video Bridge Development Kit Users Guide - EB88 - version 1.0 - 2014-09-30. Retrieved from: http://www.latticesemi.com/en/Products/DevelopmentBoardsAndKits/LatticeUSB3VideoBridgeDevelopmentKit.aspx

4. Lattice Semiconductor (2016) [Retrieved 2016-04-26] LatticeECP3 Family Data Sheet DS1021 Version 02.8EA, March 2015. Retrieved from: http://www.latticesemi.com/Products/FPGAandCPLD/LatticeECP3.aspx

5. Lattice Semiconductor (2016) [Retrieved 2016-04-27] Lattice Diamond Software - version 3.7 - 2016-02-22. Retrieved from: http://www.latticesemi.com/Products/DesignSoftwareAndIP/FPGAandLDS/LatticeDiamond.aspx

6. Cypress Semiconductor Corp (2016) [Retrieved 2016-04-27] EZ-USB® FX3: SuperSpeed USB Controller - 2016-04-07. Retrieved from: http://www.cypress.com/documentation/datasheets/cyusb301x-cyusb201x-ez-usb-fx3-superspeed-usb-controller?source=search&cat=technical_documents

7. MIPI Alliance (2016) [Retrieved 2016-05-03] Evolving CSI-2 Specification. Retrieved from: http://mipi.org/sites/default/files/files/MIPI%20CSI-2%20Specification%20Brief.pdf

8. MIPI Alliance (2016) [Retrieved 2016-05-03] Camera Interface Specifications. Retrieved from: http://mipi.org/specifications/camera-interface

9. Cypress Semiconductor Corp (2016) [Retrieved 2016-06-15] EZ-USB® FX3: SuperSpeed USB 3.0 peripheral Controller. Retrieved from: http://www.cypress.com/products/ez-usb-fx3-superspeed-usb-30-peripheral-controller

10. National Instruments (2016) [Retrieved 2016-05-03] Understanding LVDS for digital test systems. Retrieved from: http://www.ni.com/white-paper/4441/en/

11. Maxim Integrated (2016) [Retrieved 2016-05-03] LVDS basics for base stations. Retrieved from: https://www.maximintegrated.com/en/app-notes/index.mvp/id/1058
12. Maxim Integrated (2016) [Retrieved 2016-06-18] High speed signalling, GMSL. Retrieved from: https://www.maximintegrated.com/en/products/interface/high-speed-signaling/gmsl.html
13. Kjell & Company (2016) [Retrieved 2016-08-11] HDMI. Retrieved from: https://www.kjell.com/se/fraga-kjell/hur-funkar-det/hembio/bild-och-ljudoverforing/hdmi
14. Lidar-UK (2016) [Retrieved 2016-07-06] Lidar. Retrieved from: http://www.lidar-uk.com/how-lidar-works/
15. Maxim Integrated (2016) [Retrieved 2016-06-19] MAX9291 3.12Gbps GMSL serializer. Retrieved from: https://www.maximintegrated.com/en/products/interface/high-speed-signaling/MAX9291.html
16. Maxim Integrated (2016) [Retrieved 2016-08-30] Retrieved from: https://www.maximintegrated.com/en/app-notes/index.mvp/id/1856

APPENDICES

I bilagor redovisas med fördel aktiviteter och övriga handlingar som inte är helt nödvändiga för förståelsen av examensarbete i sin helhet, men som kan hjälpa den detaljintresserade läsaren. Sidnumrering sker i övre högra hörnet och varje bilaga numreras individuellt enligt följande exempel: BILAGA 3. Sid 1(7) eller BIL.1 s. 1(7).

A. Config file for Raspberry Pi

NOTE: Lines marked in red are changes done to force the desirable HDMI configuration.

```
# For more options and information see
# http://www.raspberrypi.org/documentation/configuration/config-txt.md
# Some settings may impact device functionality. See link above for details

# uncomment if you get no picture on HDMI for a default "safe" mode
#hdmi_safe=1

# uncomment this if your display has a black border of unused pixels visible
# and your display can output without overscan
disable_overscan=1

# uncomment the following to adjust overscan. Use positive numbers if console
# goes off screen, and negative if there is too much border
#overscan_left=16
#overscan_right=16
#overscan_top=16
#overscan_bottom=16

hdmi_ignore_edid=0xa5000080
framebuffer_ignore_alpha=1
framebuffer_depth=32

# uncomment to force a console size. By default it will be display's size minus
# overscan.
#framebuffer_width=1280
#framebuffer_height=720

# uncomment if hdmi display is not detected and composite is being output
hdmi_force_hotplug=1

# uncomment to force composite
#hdmi_ignore_hotplug=1
```

```
# uncomment to force a specific HDMI mode (this will force VGA)
hdmi_group=2
#hdmi_mode=19      #CEA 1280x720pX50Hz
#hdmi_mode=47      #DMT 1440x900x60 PCLK=106MHz
#hdmi_mode=35      #DMT 1280x1024x60 PCLK108MHz
#hdmi_mode=18      #DMT 1280x768x75 PCLK78MHz
hdmi_mode=4                #DMT 640x480x60 PCLK24MHz?
#hdmi_mode=11      #PAL CEA

#hdmi_mode=87      #manual settings
#hdmi_cvt=1280 1024 60 4 0 0 1    #DMT override 1280x1024x60 reduced blanking

# uncomment to force a HDMI mode rather than DVI. This can make audio work in
# DMT (computer monitor) modes
#hdmi_drive=2

# uncomment to increase signal to HDMI, if you have interference, blanking, or
# no display
#config_hdmi_boost=4

# uncomment for composite PAL
#sdtv_mode=2
#sdtv_aspect=1

#uncomment to overclock the arm. 700 MHz is the default.
arm_freq=800

core_freq=250
sdram_freq=400
over_voltage=0
gpu_mem=128
```