

# INTERNSHIP REPORT

---

**Industry internship at Enclustra GmbH,  
Räffelstrasse 28, CH-8045 Zürich**

---

presented by  
Arthur Ruder  
Matr.-Nr. 310697

Supervised by Dr.Sc. Jelena Dragas FPGA System Design Engineer

Aachen, 17.09.2019



# Contents

<b>List of Figures</b>	<b>I</b>
<b>List of Tables</b>	<b>III</b>
<b>Abbreviations</b>	<b>V</b>
<b>1. Introduction</b>	<b>1</b>
<b>2. Internship report</b>	<b>3</b>
2.1. Week 1 . . . . .	3
2.1.1. Company profile . . . . .	3
2.1.2. Market research . . . . .	3
2.2. Week 2 . . . . .	4
2.2.1. Out-of-box testing . . . . .	4
2.2.2. Wiki updates and ZCU 104 testing . . . . .	6
2.3. Week 3 . . . . .	6
2.3.1. Presentation 'Introduction to AI' . . . . .	6
2.3.2. <i>Deep Neural Network Development Kit (DNNDK)</i> testing on ZCU 104 . . . . .	7
2.4. Week 4 . . . . .	8
2.4.1. Presentation 'Introduction to AI' . . . . .	8
2.4.2. Xilinx Vivado and DNNDK workflow . . . . .	8
2.5. Week 5 . . . . .	10
2.5.1. Presentation 'Introduction to Machine Learning (ML) on Field Programmable Gate Arrays (FPGAs)' . . . . .	10
2.5.2. Xilinx ML event Munich . . . . .	12
2.6. Week 6 . . . . .	12
2.6.1. Xilinx ML event report . . . . .	12
2.6.2. Presentation 'Introduction to ML on FPGAs' . . . . .	12
2.6.3. Bugfix for ZCU 104 evaluation board . . . . .	13

2.6.4.	Synthara collaboration . . . . .	13
2.7.	Week 7 . . . . .	13
2.7.1.	Presentation 'Introduction to ML on FPGAs' . . . . .	13
2.7.2.	Vivado hardware design and embedded OS . . . . .	14
2.8.	Week 8 . . . . .	16
2.8.1.	Vivado hardware design and embedded OS . . . . .	16
2.8.2.	Synthara collaboration conference call . . . . .	18
2.9.	Week 9 . . . . .	18
2.9.1.	Intel evaluation . . . . .	18
2.9.2.	Petalinux workflow . . . . .	19
2.10.	Week 10 . . . . .	20
2.10.1.	DNNDK update . . . . .	20
2.10.2.	Petalinux workflow . . . . .	20
2.10.3.	Enclustra hardware integration . . . . .	21
2.11.	Week 11 . . . . .	22
2.11.1.	Enclustra hardware integration . . . . .	22
2.11.2.	Petalinux build . . . . .	22
2.11.3.	Synthara visit and collaboration discussion . . . . .	24
2.12.	Week 12 . . . . .	24
2.12.1.	Mars ST3 demonstrator . . . . .	24
2.13.	Week 13 . . . . .	27
2.13.1.	Mars ST3 demonstrator preparation . . . . .	27
2.13.2.	ROCK-PAPER-SCISSOR demonstration data collection preparation . . . . .	27
2.14.	Week 14 . . . . .	28
2.14.1.	ROCK-PAPER-SCISSOR demonstration data collection preparation . . . . .	28
2.14.2.	Shell scripting and presentation on data set collection . . . . .	29
2.15.	Week 15 . . . . .	29
2.15.1.	Data set collection . . . . .	29
2.15.2.	Intel ML seminar in Lausanne . . . . .	30
2.16.	Week 16 . . . . .	31
2.16.1.	Data set collection . . . . .	31
2.16.2.	Robot hand ordering . . . . .	32
2.16.3.	Internship documentation . . . . .	32

2.17. Week 17 . . . . .	33
2.17.1. Robot hand ordering . . . . .	33
2.17.2. Synthara update . . . . .	33
2.17.3. Internship documentation . . . . .	33
2.18. Week 18 . . . . .	34
2.18.1. Internship documentation . . . . .	34
2.18.2. Synthara update . . . . .	34
2.19. Week 19 . . . . .	35
2.19.1. Website content about AI . . . . .	35
2.19.2. Embedded World demo discussion . . . . .	35
2.19.3. DNNDK Tensorflow workflow . . . . .	36
2.20. Week 20 . . . . .	36
2.20.1. DNNDK Tensorflow workflow . . . . .	36
2.20.2. Data set publication . . . . .	37
2.20.3. Robot hand assembly . . . . .	38
2.20.4. Mettler Toledo <i>Artificial Intelligence (AI)</i> demo interest . . . . .	38
2.21. Week 21 . . . . .	38
2.21.1. Robot hand testing and documentation . . . . .	38
2.21.2. Mettler Toledo AI visit . . . . .	38
2.21.3. FPGA PWM VHDL module . . . . .	39
2.22. Week 22 . . . . .	39
2.22.1. FPGA PWM VHDL module . . . . .	39
2.22.2. Synthara design porting . . . . .	40
2.22.3. Xilinx proprietary software trial . . . . .	40
2.23. Week 23 . . . . .	40
2.23.1. Synthara design porting . . . . .	40
2.23.2. Mars XU3 alternative . . . . .	40
2.24. Week 24 . . . . .	41
2.24.1. MIPI demonstrator Petalinux Mars ST3 . . . . .	41
2.24.2. I2C debugging . . . . .	42
2.25. Week 25 . . . . .	42
2.25.1. New <i>Deep Learning Processor Unit (DPU)</i> version . . . . .	42
2.25.2. ECC 2019 presentation . . . . .	42
2.26. Week 26 . . . . .	43
2.26.1. ECC 2019 presentation . . . . .	43

2.26.2. Petalinux debug . . . . .	44
2.27. Week 27 . . . . .	44
2.27.1. ECC 2019 presentation . . . . .	44
2.27.2. Mars ST3 demonstrator polishing . . . . .	45
2.27.3. Synthara design porting . . . . .	45
<b>Bibliography</b>	<b>47</b>
<b>A. Workday reports</b>	<b>49</b>

# List of Figures

2.1.	Hardware platform overview . . . . .	4
2.2.	High level ZYNQ family overview [1] . . . . .	5
2.3.	ILSVRC winners . . . . .	8
2.4.	Example system with integrated DPU [4, p. 8] . . . . .	9
2.5.	FPGA architecture overview . . . . .	11
2.6.	Rock-paper-scissors demonstrator setup . . . . .	14
2.7.	D-PHY MIPI IP core overview [6] . . . . .	15
2.8.	Receiver MIPI IP core subsystem [7] . . . . .	15
2.9.	MIPI CSI block design . . . . .	17
2.10.	OpenVINO toolkit overview [3] . . . . .	19
2.11.	Neural network framework usage overview [2] . . . . .	20
2.12.	AI SDK overview [8] . . . . .	21
2.13.	DPU hierarchy system . . . . .	23
2.14.	resnet50 demonstrator overview . . . . .	25
2.15.	resnet50 classification example . . . . .	26
2.16.	Data set overview and requirements for a good quality data set . . . . .	28
2.17.	Data set collection setup . . . . .	30
2.18.	Data set segmentation . . . . .	31
2.19.	Robotic hand chosen for the demonstrator . . . . .	32
2.20.	PWM signal servo motor position mapping . . . . .	33
2.21.	Mars ST3 DNNDK demonstrator . . . . .	34
2.22.	Embedded World demonstration setup . . . . .	36
2.23.	CIFAR-10 data set example . . . . .	37
2.24.	PWM module . . . . .	39
2.25.	Hardware design language testbench overview . . . . .	39
2.26.	Xilinx MPSoC product overview [5] . . . . .	41
2.27.	I2C protocol data transfer . . . . .	42
2.28.	DPU performance overview [4] . . . . .	43

2.29. Qualitative comparison of available hardware platforms . . . . .	44
2.30. Vivado usage report Rock-Paper-Scissor demonstrator . . . . .	45
A.1. Week 1 . . . . .	49
A.2. Week 2 . . . . .	49
A.3. Week 3 . . . . .	50
A.4. Week 4 . . . . .	50
A.5. Week 5 . . . . .	50
A.6. Week 6 . . . . .	51
A.7. Week 7 . . . . .	51
A.8. Week 8 . . . . .	51
A.9. Week 9 . . . . .	52
A.10. Week 10 . . . . .	52
A.11. Week 11 . . . . .	52
A.12. Week 12 . . . . .	53
A.13. Week 13 . . . . .	53
A.14. Week 14 . . . . .	53
A.15. Week 16 . . . . .	53
A.16. Week 17 . . . . .	54
A.17. Week 18 . . . . .	54
A.18. Week 19 . . . . .	54
A.19. Week 20 . . . . .	55
A.20. Week 21 . . . . .	55
A.21. Week 22 . . . . .	56
A.22. Week 23 . . . . .	56
A.23. Week 24 . . . . .	56
A.24. Week 25 . . . . .	57
A.25. Week 26 . . . . .	57

# **List of Tables**



# Abbreviations

<b>AI</b>	Artificial Intelligence
<b>ANN</b>	Artificial Neural Network
<b>API</b>	Application Program Interface
<b>APU</b>	Application Processing Unit
<b>ASIC</b>	Application Specific Integrated Circuit
<b>AXI</b>	Advanced eXtensible Interface
<b>CIFAR</b>	Canadian Institute For Advanced Research
<b>CPU</b>	Central Processing Unit
<b>CSI</b>	Camera Serial Interface
<b>DMA</b>	Direct Memory Access
<b>DNNDK</b>	Deep Neural Network Development Kit
<b>DP</b>	DisplayPort
<b>DPU</b>	Deep Learning Processor Unit
<b>DSP</b>	Digital Signal Processor
<b>DUT</b>	Device Under Test
<b>FAE</b>	Field Application Engineer
<b>FPGA</b>	Field Programmable Gate Array
<b>GPU</b>	Graphics Processing Unit
<b>GUI</b>	Graphic User Interface
<b>HDL</b>	Hardware Description Language
<b>IP</b>	Intellectual Property
<b>MAC</b>	Multiply and Accumulate

<b>MIPI</b>	Mobile Industry Processor Interface
<b>ML</b>	Machine Learning
<b>MNIST</b>	Modified National Institute of Standards and Technology database
<b>MPSoC</b>	Multiple Processor System on Chip
<b>OS</b>	Operating System
<b>PCB</b>	Printed Circuit Board
<b>PCIe</b>	Peripheral Component Interconnect Express
<b>PLL</b>	Phase Locked Loop
<b>PL</b>	Programmable Logic
<b>PS</b>	Processing System
<b>PWM</b>	Pulse Width Modulation
<b>PWM</b>	Pulse Width Modulation
<b>QEMU</b>	Quick Emulator
<b>RPU</b>	Real-time Processing Unit
<b>SDK</b>	Software Development Kit
<b>SDR</b>	Software Defined Radio
<b>SDSoC</b>	Software Development System on Chip
<b>SoC</b>	System on Chip
<b>USB</b>	Universal Serial Bus
<b>VPU</b>	Vector Processing Unit

# 1. Introduction

In this internship my work can be divided into three main blocks, DNNDK testing, DNNDK demonstrator on Enclustra hardware and finally a collaboration project.

After investigating the possibilities of available *Artificial Neural Network (ANN)* accelerators and market research, the Xilinx DNNDK was studied in more detail. The available examples were evaluated using the Xilinx ZCU 104 evaluation board. This was used as a test platform to check the published sample applications in the field of image recognition. The main application area is image classification and detection.

The next step was porting the available examples and use cases to Enclustra custom hardware. In addition, a demonstrator for ANN applications was developed using the Enclustra Mars ST3 baseboard and Mars XU 3 FPGA module. Image classification using resnet50 and face detection were implemented and tested. The whole design included using Vivado for the DPU *Intellectual Property (IP)* core integration and then building an embedded Linux *Operating System (OS)* with Xilinx Petalinux.

The other main part of the internship was developing a more sophisticated ANN demonstrator in collaboration with Synthara. Synthara is an ETH spin-off and offers their own IP core for neural network applications. The idea for the demonstrator was for a human player to play Rock-Paper-Scissors against a robot hand. The decision which symbol to choose would be done by using a camera and running a hand gesture detection neural network on the FPGA. For this purpose, a data set was collected and a custom neural network trained. The hardware implementation was also started by using the Synthara neural network accelerator IP core.

Throughout the internship several presentations were held as well. The topics ranged from introduction to AI and ML in general, to AI on FPGAs and data set collection.



## 2. Internship report

### 2.1. Week 1

#### 2.1.1. Company profile

The first two days of the week were spent getting to know all colleagues and familiarize myself with internal processes and guidelines. Zurich office is the headquarters of Enclustra GmbH and therefore the majority of hardware and software design is being done here. Around fifty people, most of which are hardware and software engineers, work in the Zurich office. The company itself is divided into two areas, FPGA Design Center and FPGA Solution Center. The former is offering customer-specific design services implementing applications on FPGAs and providing support. Areas of expertise include wired networks and switching, wireless communications (*Software Defined Radio (SDR)*), smart cameras, embedded interfaces (*Peripheral Component Interconnect Express (PCIe)*, *Universal Serial Bus (USB)*, *Advanced eXtensible Interface (AXI)*, ethernet, etc.), test and measurement (sensors, data acquisition, *Digital Signal Processor (DSP)*) and drive/motion control. The FPGA Solution Center designs custom FPGA/*System on Chip (SoC)* modules and custom IP cores. Several baseboard families and FPGA module families are developed and supported and can be adapted to the needs of the application by offering different performance key points. Reference designs for each combination of base board and module are provided as a starting point for customers.

#### 2.1.2. Market research

Furthermore, my task was to do market research on artificial intelligence and artificial intelligence on FPGAs especially. The four key platforms for artificial neural network applications are shown in figure 2.1. A qualitative design trade-off is shown on the  $x$ - and  $y$ -axis in terms of power efficiency and performance versus flexibility and ease-of-use. As Enclustras focus is on the embedded market, the market survey has

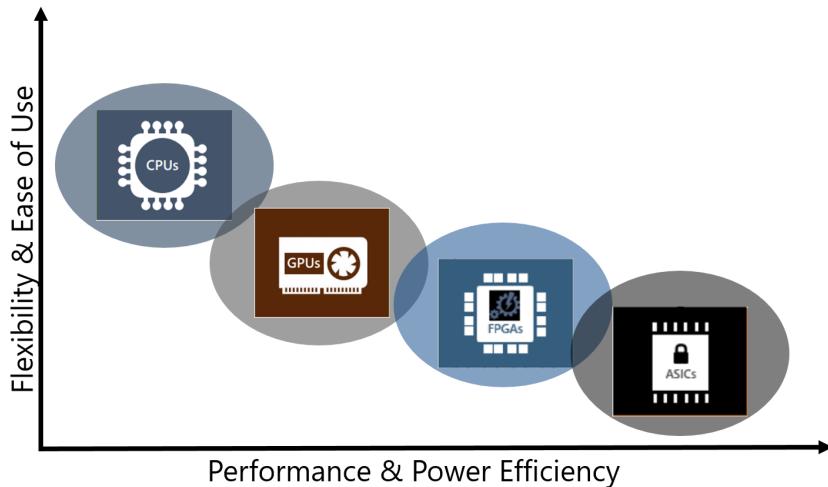


Figure 2.1.: Hardware platform overview

been mainly on *Graphics Processing Units (GPUs)*, FPGAs and *Application Specific Integrated Circuits (ASICs)* as full blown *Central Processing Units (CPUs)* are too inefficient for embedded applications. Possible competitors as well as toolkits provided by FPGA manufacturers such as Intel, Xilinx and Lattice have been evaluated. The results have been presented in a meeting in which a discussion has been held, where Enclustras products and services can fit. One of the results was to start planning an AI demonstrator using Enclustra hardware. The purpose of this demonstrator was to showcase machine learning applications running on Enclustra hardware. As a preliminary step it was decided to check the Xilinx DNNDK on an evaluation board, the ZCU 104.

## 2.2. Week 2

### 2.2.1. Out-of-box testing

At the beginning of the week a task unrelated to AI was given to check upon internal documentation and customer support. Together with another recently hired employee, a day of out-of-box testing was scheduled. An Enclustra base board (Mercury+ PE1-400) together with a fitting FPGA module (Mercury+ XU1) featuring a Xilinx *Multiple Processor System on Chip (MPSoC)* shuold be tested. Some details will be given for this specific FPGA family as the Zynq-7000 SoC and the Zynq-MPSoC Xilinx product family are unique in the way dedicated ARM processors are combined

with traditional FPGAs. A high level overview is shown in figure 2.2. It shows

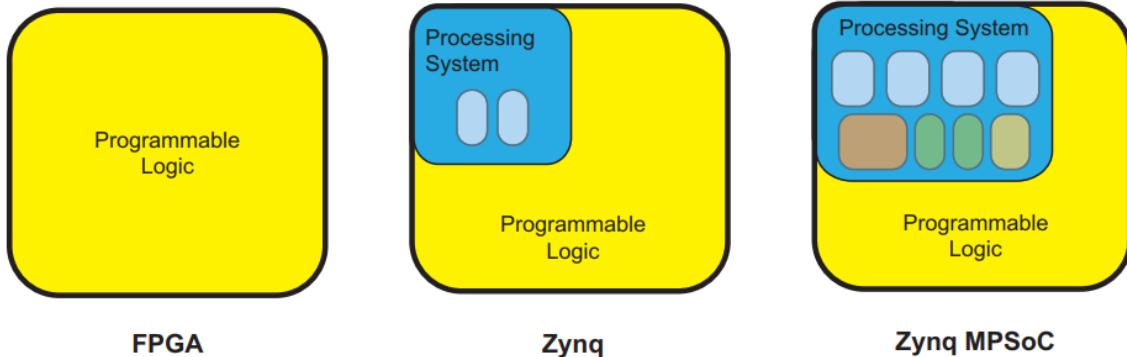


Figure 2.2.: High level ZYNQ family overview [1]

the difference between traditional FPGAs and the Zynq product family. The main benefit here is the division between *Processing System (PS)* and *Programmable Logic (PL)*. This allows to combine the benefits of traditional processing units with the flexibility of FPGAs. Custom logic and all peripheral devices can be implemented in the PL part, while system control and even complete OSs (such as embedded Linux) can be done in the PS. As this system is completely integrated in one package, the communication between the two fabrics is extremely fast and can be done using AXI interfaces. The MPSoC family even integrates several different types of processing units, *Application Processing Units (APUs)*, GPUs, *Vector Processing Units (VPUs)* and *Real-time Processing Units (RPUs)* as well as additional hard cores. The Enclustra module uses an MPSoC Xilinx FPGA and our task was to go through the whole process of setting up the base board together with a corresponding module to test customer experience using the provided documentation, user manual and reference design. The goal was to find unclear instructions in the documentation and provide feedback as to the overall experience setting up the hardware out-of-the-box. First, the hardware reference design was loaded in the Vivado Design Suite and the bit stream generated. After, the hardware description file was exported so it can be used using the Xilinx *Software Development Kit (SDK)*. This allows to create applications in C/C++ against the custom hardware design. All of the provided sample applications have been tested and verified. Some unclear instructions were identified and discussed with the employee in charge to improve customer experience.

### 2.2.2. Wiki updates and ZCU 104 testing

The rest of the week was spent updating the internal Wiki page for AI. Furthermore, the DNNDK sample applications were tested on the ZCU 104 evaluation board. The provided examples include several state-of-the-art neural networks demonstrating key applications for neural network inference, such as image classification, face detection, object detection and pose detection. As only the image classification example worked directly for this particular evaluation a fix needed to be found. Another task was to introduce the topic of AI to the whole company as ANNs was a completely new design field for a majority of the technical staff. Two PowerPoint presentations should be prepared, namely 'Introduction to AI' and 'Introduction to ML on FPGAs'. I started with the preparation of the first one in parallel with finding a bug fix for the other DNNDK sample applications, as these should be part of the second presentation.

## 2.3. Week 3

### 2.3.1. Presentation 'Introduction to AI'

The main focus of this week was research and starting to layout the first presentation. It was assumed that the audience is technology savvy but has no particular background in AI. Thus, the presentation had to introduce the whole field and key concepts that enabled the rise of AI applications in recent years. The first draft of the presentation was discussed in a meeting and some changes were made to the overall structure, the content and the degree of complexity. The rough structure of the presentation is as follows:

- **Motivation:** To get the viewers interest it was shown that AI applications are already part of daily life for everyone. This was achieved by showing that all of the major companies such as Google, Apple, Facebook, Microsoft as well as Tesla, Netflix and Amazon use AI in their datacenters and products and allocate huge resources to AI research. The importance of AI was further enhanced by showing the rapid growth of annually published AI papers and startups developing AI systems. The trend from 1995 to 2015 resembles almost exponential growth in AI research and products.
- **Definition:** As AI has become such a buzz word in media a definition of the

term was needed and what part of AI is actually used in all of the common applications. ANN that perform typical computer vision and language processing tasks are all part of ML, which is a subset of AI. ML itself can then be divided into further subsets using roughly three learning methods, supervised learning, unsupervised learning and reinforcement learning. As supervised learning is the most commonly used method, the presentation focused on this method used to train ANN. Furthermore, the different parts that comprise a ANN are introduced, namely the neuron and how neurons are formed into layers. These layers are then stacked together to form an ANN.

- **Key concepts:** An explanation of supervised learning was given with two distinct examples, linear regression and deep learning neural networks to illustrate the idea behind supervised learning: predict a value  $y$  given an input  $x$  by deploying a function  $f(x)$ . This function  $f(x)$  is acquired by deploying a learning algorithm and usage of a so called training set, consisting of input pairs  $x$  and  $y$ . The concept of inference and training were also explained with an emphasis on inference. Once a network is trained, only inference needs to be run, so this is the crucial part application wise.
- **Deep Learning:** Figure 2.3 shows the winners of the state-of-the-art ImageNet Large Scale Visual Recognition Challenge. It illustrates that the breakthrough in performance came not only from more sophisticated networks, but mainly from stacking different kinds of layers deeply, hence the name. The presentation was ended with the question, what hardware is best suited for ANN applications. This question would be addressed in the second presentation 'Introduction to ML on FPGAs'.

### 2.3.2. DNNDK testing on ZCU 104

Alongside preparing the presentation the error preventing the more sophisticated DNNDK examples from running was identified as the board crashed while performing tasks related to video analysis (face detection, pose detection, etc.). At first, it was suspected there were some problems with heat management and using the system monitor the temperature of the FPGA was investigated during operation. As this seemed to be well within allowed borders specified by the Xilinx data sheet, other causes had to be found.

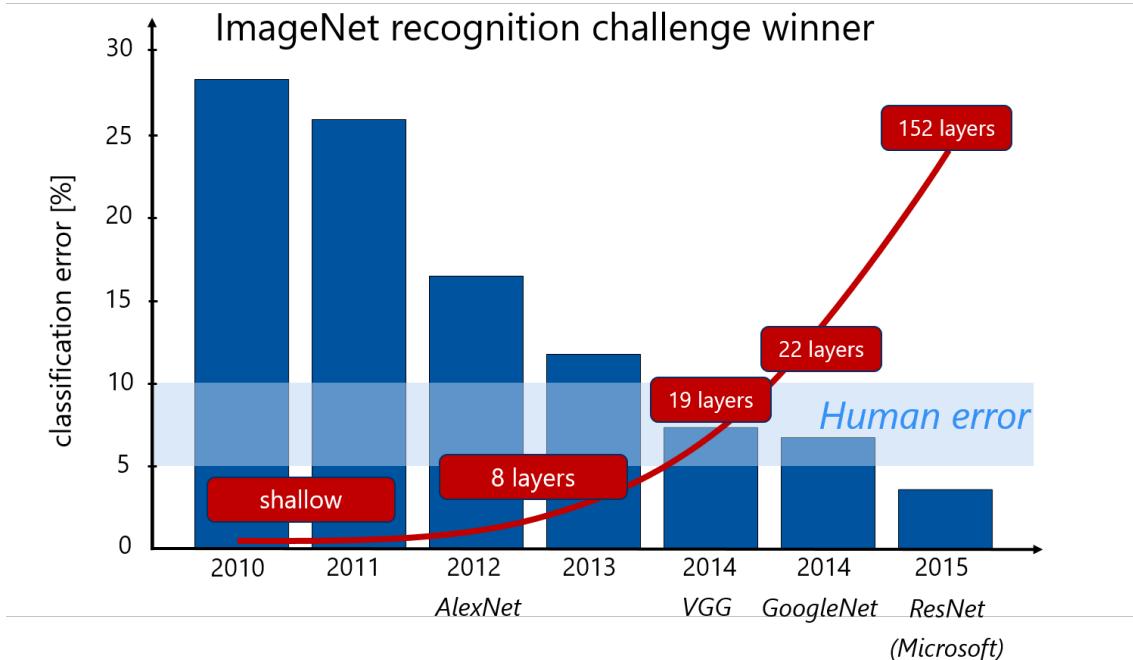


Figure 2.3.: ILSVRC winners

## 2.4. Week 4

### 2.4.1. Presentation 'Introduction to AI'

At the beginning of the week the first presentation 'Introduction to AI' was held before the technical stuff of the company. The general background and principles of ML have been introduced and an outlook given to the second presentation, which would go more into detail about the actual hardware realization.

### 2.4.2. Xilinx Vivado and DNNDK workflow

The rest of the week was spent going through various tutorials provided by Xilinx to familiarize myself with the workflow and the DNNDK toolkit. As the state of tools used for AI applications on FPGA is still in flux, several approaches needed to be evaluated:

- **DNNDK workflow:** Version 2.08 of the toolkit supported only the Caffe neural network training framework and needs a network description file and the trained weights as input. The key component here is the DPU IP core

provided by the DNNDK toolkit. This core is integrated via Vivado into the block design of the hardware and can be configured and adjusted for several performance and power profiles.

- **DNNDK Software Development System on Chip (SDSoC):** Another option is to abstract away the whole Vivado block design process and use Xilinx SDSoc to implement the whole system in a higher programming language, C++. Supported functions can then be flagged as being executed in the PL part of the system. This approach makes using a traditional *Hardware Description Language (HDL)* obsolete and is deemed more accessible. This approach uses the established Xilinx reVision stack for development providing high level *Application Program Interfaces (APIs)* for computer vision.

Furthermore, the IP core provided by the DNNDK toolkit was studied in more detail. A new Enclustra baseboard was in the final production phase and the idea was to have a ML design ready to showcase the capabilities of the new board. Figure 2.4

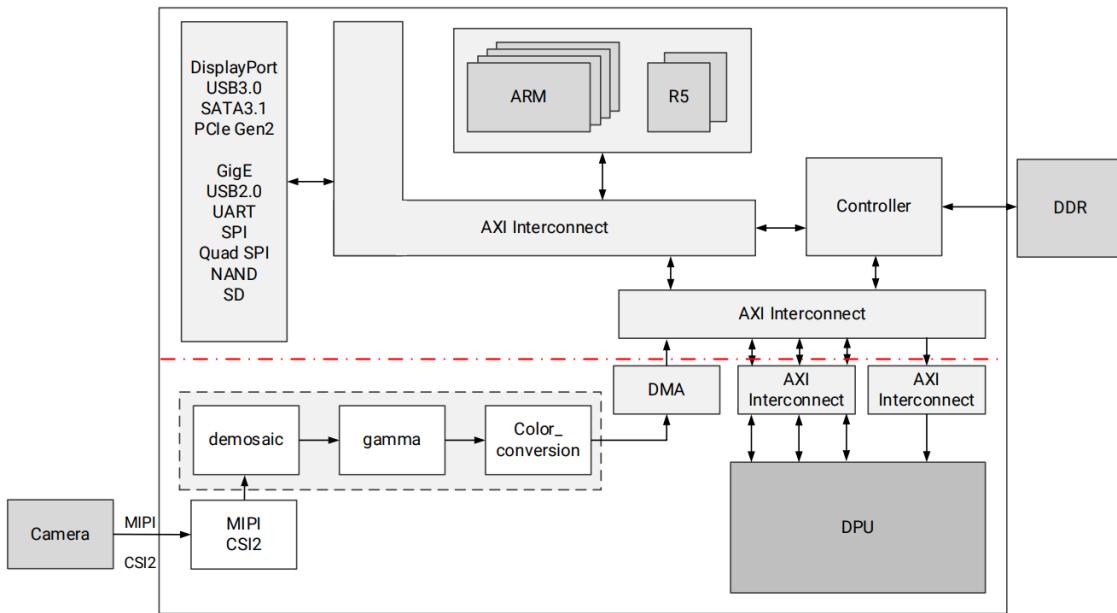


Figure 2.4.: Example system with integrated DPU [4, p. 8]

shows an example hardware design with integrated DPU module. In this example a camera is connected via the *Mobile Industry Processor Interface (MIPI) Camera Serial Interface (CSI)* interface to the PS. *Direct Memory Access (DMA)* is usually

used in conjunction with an AXI interconnect to communicate with the PS. The captured images are used as the input to the neural network and the DPU itself can be viewed as a co-processor to the PS implemented in the PL fabric. The IP core itself is customizable and the number of DPU processor units, the size of the DPU and the usage of DSP blocks among other parameters are configurable. The decision on which size to use is based upon the performance demands of the application.

## 2.5. Week 5

### 2.5.1. Presentation 'Introduction to ML on FPGAs'

In this week I started working on the second presentation 'Introduction to ML on FPGAs'. This time the focus should be on the hardware needed to handle typical ML workloads, namely inference and training along all major fields of applications where ANN are used. The main areas are: image classification, object detection, semantic segmentation, optical character recognition and speech recognition. The presentation structure is as follows:

- **ANN workload:** Using a state-of-the-art neural network (resnet50) the number of operations needed to process one image in the ANN were illustrated to show the vast amount of compute and memory needed for a single image. This was done for inference and training respectively with the purpose of laying out the challenges involved in ML applications. Moreover, the majority of operations are the costly *Multiply and Accumulate (MAC)* operations which take several clock cycles to complete.
- **Hardware for training:** The industry right now in terms of ANN training is dominated by NVIDIA and so the clear answer here was GPU. There are a number of start-ups developing alternative solutions to get into the ANN training market. The main advantage these start-ups have is that they can design hardware specifically tailored to ANN workloads from scratch and use an architecture tailored to the specific requirements of ANN training. The sheer number of floating point 32 operations and the requirements for memory are strictly not suited for FPGAs.
- **Hardware for inference:** The picture is different for inference. Here, a lot of research has been done in using quantization and pruning of ANN models

without impeding the performance of these networks. The reason for this is, that neural networks are inherently over-parametrized and this is necessary for the training algorithms to work. Once a trained network is obtained however, the network can be compressed severely (up to 90 %) without network degradation. A qualitative comparison of the available platforms was made to show the strengths and weaknesses of each platform. The flexibility of FPGAs make them a suitable platform for ANN inference.

- **FPGA architecture:** Figure 2.5 shows the two possible high-level architectures that are typically used in neural network implementations. On the left you have the streaming architecture where basically the structure of the neural network is mirrored by the hardware implementation. The main benefit is the efficiency and customizability as the hardware can be tailored specifically to each network. The other approach is shown on the right. This is a more general approach in that it has a single computation engine which breaks down the operations needed for neural network inference. These operations are controlled by a host and executed on demand. The main benefit here is the flexibility: I can use the same compute engine for different neural networks. As the types of layers in a neural network are fixed, efficient implementation of these different types of layers enables the deployment of arbitrary neural networks. The downside is that the implementation is limited by the architecture in terms of tailoring the hardware implementation to the neural network.

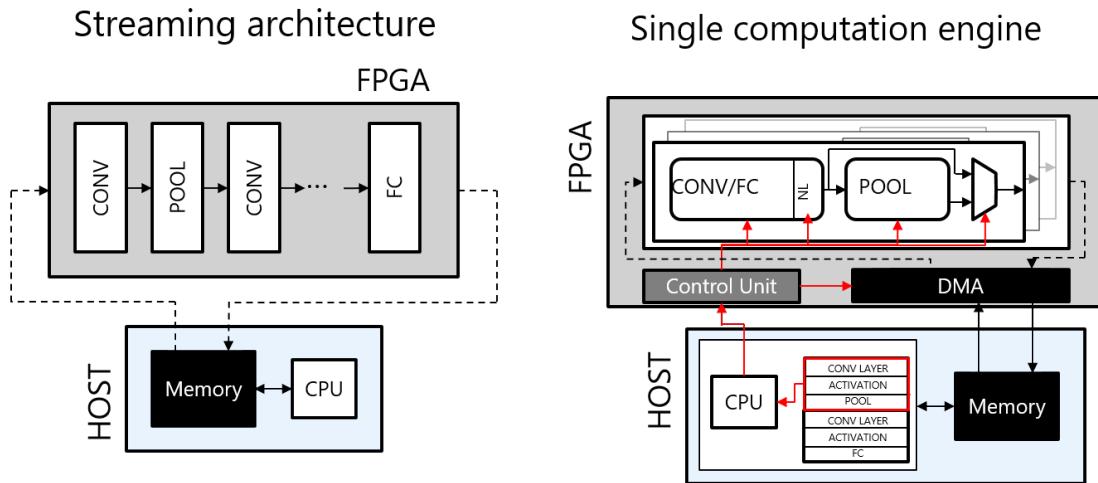


Figure 2.5.: FPGA architecture overview

- **DNNDK work flow:** Lastly, the DNNDK work flow is introduced with a focus on adjusting the DPU IP core to custom boards. Some demonstrations implemented on the ZCU 104 evaluation board were used to finish the presentation and show the employees, what is possible with the tools available.

### 2.5.2. Xilinx ML event Munich

In this week on Thursday there was a Xilinx ML seminar in Munich, where I went to with my supervisor to get more information about Xilinx AI solutions. This was a whole day event with several segments, showing off the capabilities and the work flow of Xilinx cloud and edge AI tools. This was also a great opportunity for networking and speaking in person to top Xilinx *Field Application Engineer (FAE)* engineers.

## 2.6. Week 6

### 2.6.1. Xilinx ML event report

All of the new information gathered at the Xilinx seminar last week needed to be transferred to the internal Wiki and properly documented. Three main tasks were worked upon in this week, namely preparing the second presentation 'Introduction to ML on FPGA', getting all of the DNNDK sample applications to work on the ZCU 104 evaluation board and evaluating a possible collaboration with an ETH start-up called Synthara.

### 2.6.2. Presentation 'Introduction to ML on FPGAs'

Extensive market research has been conducted to find resources and ideas on how to present the different hardware platforms. The difficulty lies therein, that there are no standardized performance metrics for neural networks. Performance is strongly dependent on the network used and the individual use case. This leads to a lot of unfair comparisons, both in research and industry when numbers are shown. Therefore, a qualitative approach was chosen as doing all of these comparisons would have taken an extreme amount of work time and effort, requiring special hardware as well.

### 2.6.3. Bugfix for ZCU 104 evaluation board

After reading through a vast amount of documentation and employing the help of online resources, mainly the Xilinx official forums, a solution was found to the problem. It turned out to be a specific problem of the ZCU 104 evaluation board which made it hard to track down. The solution to this was provided by an unofficial patch by one of the Xilinx employees online. The board has power issues when running at full load resulting in the already mentioned problem of freezing the board in the middle of running the sample applications. After applying the patch all of the sample applications worked. The patch changed a current setting on the board. These included image classification with resnet50 and inception-v1 as well as real time face detection, object detection and pose detection using other popular ANN

### 2.6.4. Synthara collaboration

During research for neural network accelerator implementations I read about an ETH start-up providing this service in the form of an ASIC. However, their prototypes as a proof-of-concept are implemented on FPGAs. Thus, we reached out to them and scheduled a meeting. During this meeting we discussed the possibility of a cooperation. The idea was to implement a demonstrator for the Embedded World 2020 conference showing off Enclustra hardware and using the Synthara neural network accelerator. The demo is a game of rock-paper-scissors played by a human player against a robotic hand. The setup can be seen in figure 2.6. The robot hand is controlled via USB using *Pulse Width Modulation (PWM)* to control each finger individually. The human players movement is captured by a camera connected via MIPI to the FPGA board. The FPGA handles all image preprocessing and is running a custom neural network capable of detecting hand gestures. The control signals for the robot hand are also given by the FPGA.

## 2.7. Week 7

### 2.7.1. Presentation 'Introduction to ML on FPGAs'

At the beginning of the week I held the second presentation 'Introduction to ML on FPGAs' in front of the assembled engineering employees. At the end of the presentation a live demonstrator was shown at my work place. The ZCU 104 was

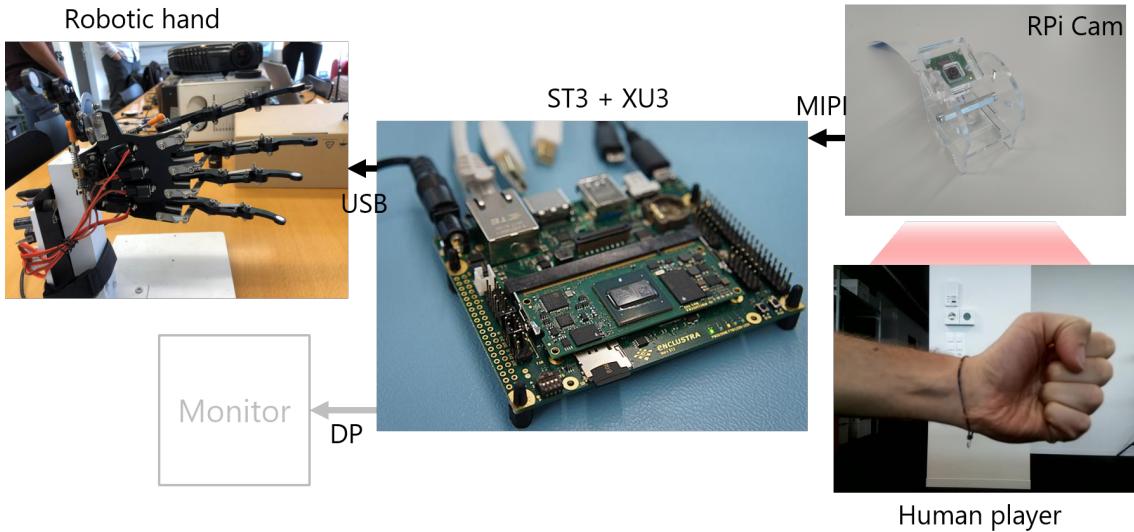


Figure 2.6.: Rock-paper-scissors demonstrator setup

used as the platform to show off all of the ML examples provided by the DNNDK and upcoming questions answered.

### 2.7.2. Vivado hardware design and embedded OS

The rest of the week was spent doing in depth research on integrating MIPI into a Vivado block design. The reason for this is that Enclustra developed a new base board, the Mars ST3. This board features a MIPI CSI connector allowing high-speed video streaming with up to four lanes and a bit rate of 2.5 Gbit/s per lane (or 2.9 Gbit/s depending on the chosen clock rate). The camera chosen was the Raspberry Pi camera with a SONY image sensor. As MIPI is not an open standard, research has been conducted into open source implementations of interfacing with the MIPI protocol. After discussing the open source alternatives with more experienced employees, those solutions were deemed unsuitable for the task at hand. The alternative solution is to use a Xilinx IP core which is available as a time limited evaluation license. The starting point was the ZCU 104 reference design, which included the whole MIPI IP core design. It consists of two main parts, the MIPI D-PHY and the MIPI Tx/Rx subsystem. A high-level view of the Xilinx MIPI D-PHY IP core system is shown in figure 2.7. The communication takes place between a Master and Slave with one clock lane and up to 4 data lanes. This IP core allows proper communication on this high-speed I/O interface standard. The complete receiver subsystem is shown in

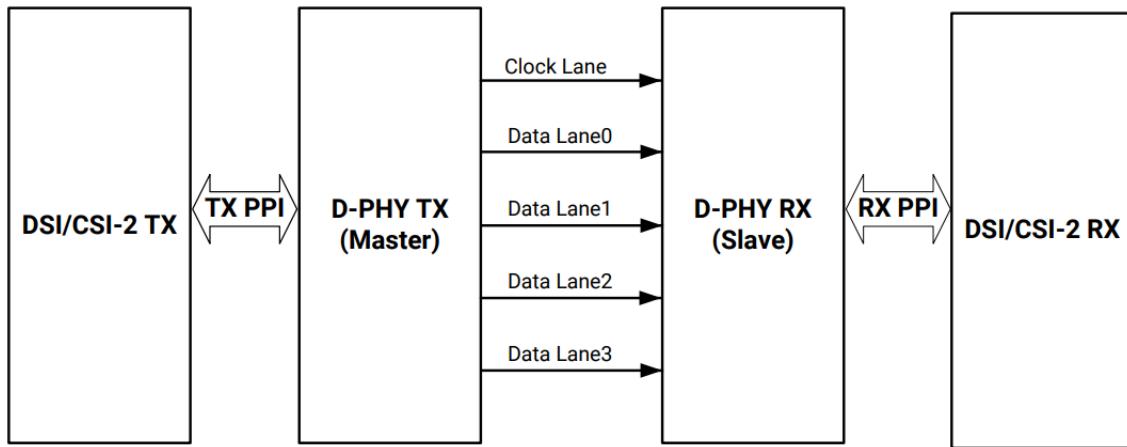


Figure 2.7.: D-PHY MIPI IP core overview [6]

figure 2.8. The D-PHY IP core is part of this subsystem and in combination with the

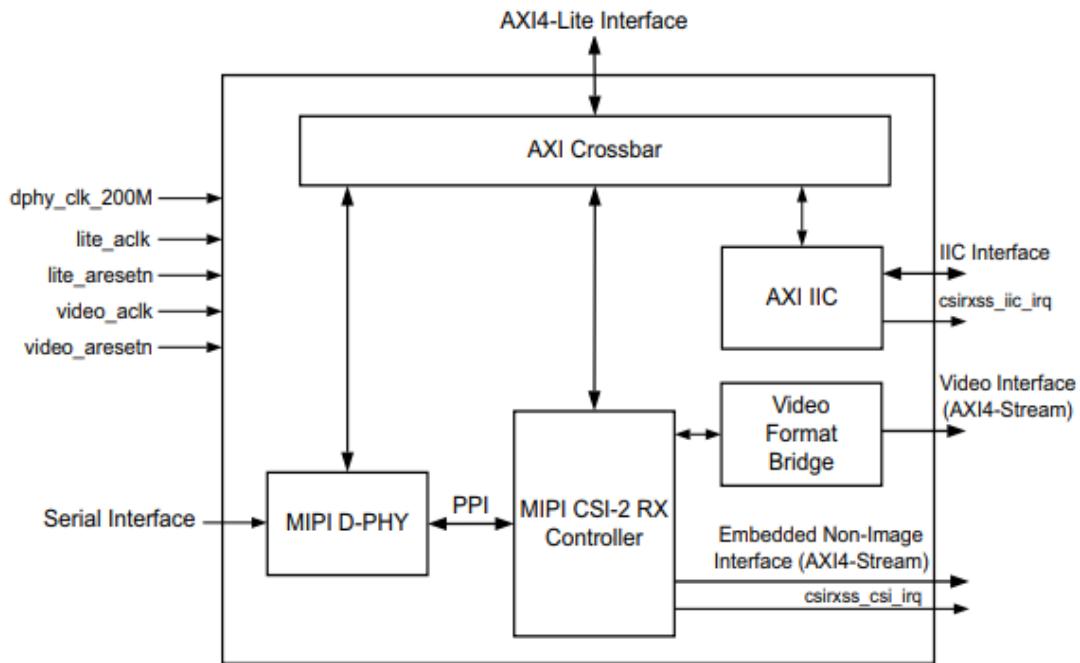


Figure 2.8.: Receiver MIPI IP core subsystem [7]

rest of the Rx subsystem allows the integration of a MIPI based image sensor and an image sensor pipe. The captured images can then be accessed via AXI interfaces. This part of the system needed to be integrated into the complete hardware design in Vivado consisting of the ZYNQ MPSoC, the DPU IP core and the usual peripheral

interfaces (USB, DisplayPort, HDMI, etc.). On top of that, an embedded Linux OS needed to be built to control the applications and provide a working demonstration environment. The chosen Linux distribution for this task was the Xilinx supported Petalinux, which is itself based on Yocto.

## 2.8. Week 8

### 2.8.1. Vivado hardware design and embedded OS

Figure 2.9 shows the reference design provided by Xilinx for the ZCU 104 board showing the complete hierarchy of the MIPI CSI block design with all additional blocks. These blocks are needed for further image processing. The data coming from the image sensor is in an unprocessed format called RAW. Therefore, an image processing pipeline needs to be integrated to convert this RAW image format into useable data in RGB format for example. The following Xilinx IPs are used to accomplish this task: 'Sensor Demosaic', 'Gamma LUT', 'Video Processing subsystem' and 'Video Frame Buffer Write'. The data transfer is handled by AXI streaming interfaces. Using the reference design as a starting point, the MIPI subsystem was integrated into the hardware design together with the DPU block. Afterward, Petalinux had to be configured and built. This enabled control of the whole system via an embedded Linux OS host system running on the ARM cores. Several steps are necessary for setting up a Petalinux environment:

- **Vivado hardware design:** First of all a working hardware design needed to be created in Vivado and successfully synthesized. This hardware design then needs to be exported in .hdf file format. This allows importing the hardware design as a template for the Petalinux system generation.
- **Creation of Petalinux project:** Petalinux is a command line tool for a Linux OS which abstracts away some of the details of building an embedded Linux OS. During generation of a new project, the previously used hardware design file is imported so that the system can access all of the implemented features.
- **Configuration:** In the next steps, necessary packages, user written apps, file system packages and custom modules can be added to the Petalinux project via

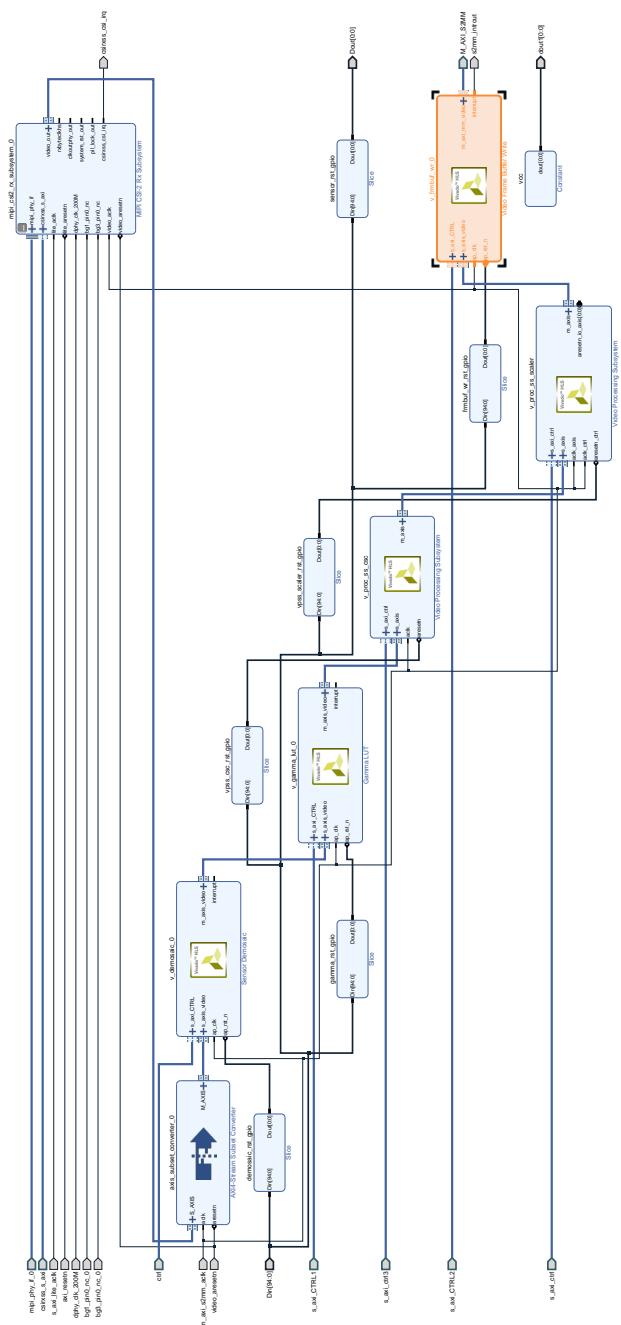


Figure 2.9.: MIPI CSI block design

console commands and a *Graphic User Interface (GUI)* environment simplifying interaction with all of the possible options.

- **Building the system:** After all of the system is configured, the necessary packages and files need to be downloaded and a root file system and kernel image constructed. This can also be done via console commands. After successfully building the whole system the necessary files need to be generated for the system to boot. This includes a BOOT.bin file, an image.ub file and the root file system. These files and directories are the minimum necessities for the Petalinux OS

To create a bootable image, an SD card is used and properly formatted. The SD card needs to partitioned into two primary partitions, BOOT formatted as FAT32 and ROOTFS, formatted as ext4. The Petalinux files are then copied over into the respective directories and the SD card can be used as the boot image for the FPGA board.

### **2.8.2. Synthara collaboration conference call**

Another Synthara conference call was due to further discuss details about the Embedded World 2019 demonstrator and a visit was organized to the company in order to create a schedule for the collaboration and labor division between Enclustra and Synthara.

## **2.9. Week 9**

### **2.9.1. Intel evaluation**

In this week a more closer evaluation of available neural network inference tools by Intel was done. The reason for this is that the dependency solely on one FPGA supplier is not ideal. In order to be flexible with product design and familiar with all of the tools on the market for neural network inference, it was decided that some time should be spend on evaluating alternatives to Xilinx. Figure 2.10 shows an overview over Intels OpenVINO toolkit. The workflow itself is similar to Xilinx DNNDK. The starting point is a trained model using popular neural network training frameworks (Caffe, Tensorflow, etc.). The trained network is then passed to a model optimizer which performs tasks such as quantization, stripping away layers only needed for

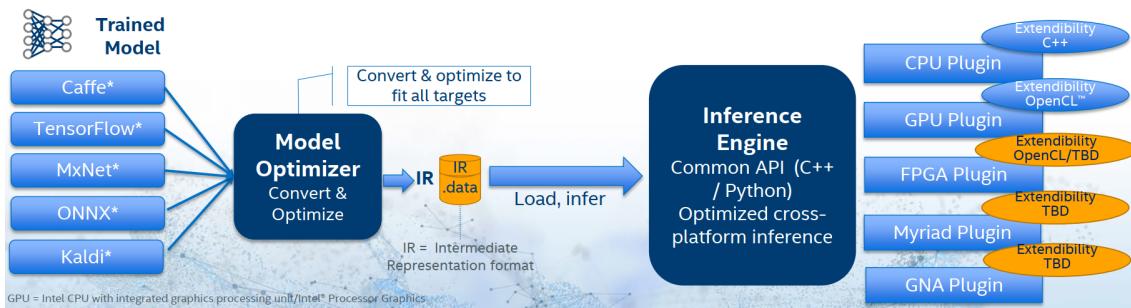


Figure 2.10.: OpenVINO toolkit overview [3]

training and other tasks. This operation is hardware independent. An intermediate representation is generated and passed on to the inference engine. This is a high level API allowing the implementation of neural networks on the target hardware. The main difference is its universal approach compared to the Xilinx DNNDK. Intel acquired the FPGA company Altera and took over their FPGA modules. Therefore, the OpenVINO toolkit does not only support FPGAs but also all of other product families Intel offers, such as CPUs, GPUs as well as other more specialized hardware. One of the main problems with Intels offering is its support of only one FPGA family, namely the Arria 10 FPGAs. Moreover, the model optimizer is not as powerful as the Xilinx DNNDK one. No pruning is taking place and as of this date, there is no support for INT8 precision, only reduced precision floating point, which is not ideal. Therefore, the Xilinx approach is deemed superior.

## 2.9.2. Petalinux workflow

Integration and building of a custom Petalinux distribution continued this week with familiarizing myself with the overall workflow and possible debugging features. The DPU IP core was successfully integrated into the ZCU 104 reference design and synthesized. As the DNNDK tool is still in a beta phase, there are frequent updates. A major update was released this week. This updated version was investigated and documented in the internal Enclustra Wiki. New sample applications were tested with the ZCU 104 evaluation board.

## 2.10. Week 10

### 2.10.1. DNNDK update

In this week some further testing of the new DNNDK version was done. The biggest change here is the added support for the Tensorflow framework. Tensorflow is the most popular neural network training framework right now as figure 2.11 shows. The x-axis represents the timeline and the y-axis is the percentage of papers mentioning a specific neural network framework. Although the output formats of Tensorflow are

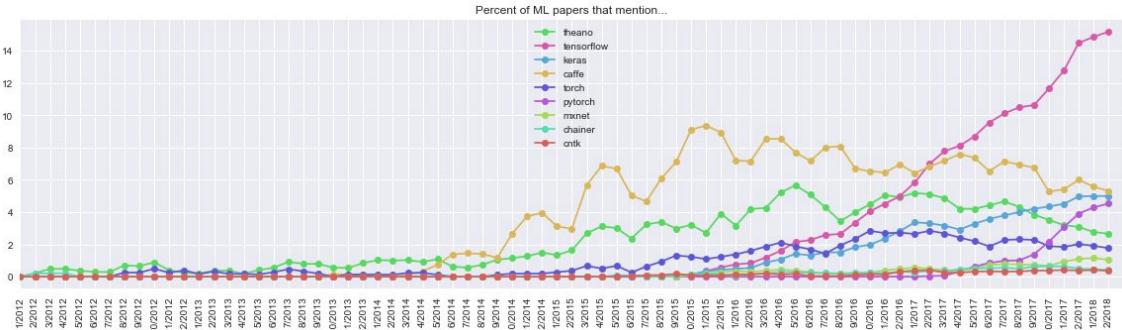


Figure 2.11.: Neural network framework usage overview [2]

different than the Caffe output files, the work flow with the DNNDK is basically the same as before: take the output of the framework, typically the trained weights and the network structure and feed it to the DNNDK tools. Another big change is the division of the DNNDK into sub groups.

- **IP core:** DPU IP core to be integrated in hardware design
- **DNNDK tools:** quantization and compression tools to acquire a binary file encapsulating the trained neural network model
- **AI SDK:** unified interface providing efficient implementations of common neural network layers as well as common neural networks (see figure 2.12)

### 2.10.2. Petalinux workflow

The Petalinux documentation was studied as well, especially setting up the device tree correctly. The device tree is a description of the hardware components that are present in a computer system so that the kernel can access these hardware

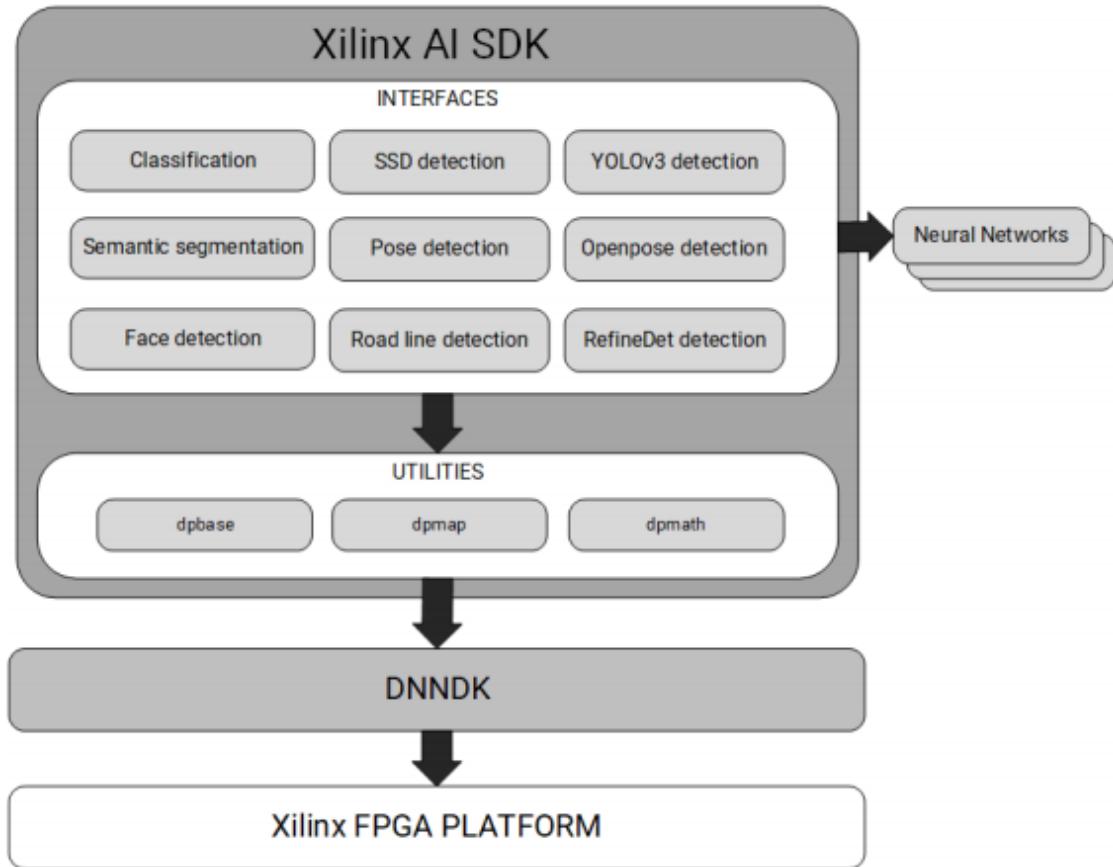


Figure 2.12.: AI SDK overview [8]

components correctly. Part of this process is automated by the Petalinux tools, however, there are also manual additions that need to be added for the system to work as intended.

### 2.10.3. Enclustra hardware integration

After successfully integrating the DPU IP core into the ZCU 104 evaluation board the implementation of the DPU on Enclustra modules was undertaken. Two different modules were chosen from the companies portfolio. One lower end ZYNQ Ultrascale device featured on the Mars XU3 and a higher end module, the Mercury+ XU1. The main difference concerning ML performance is the number of DSP blocks on the FPGAs. The DPU can be instantiated in different sizes, the bigger the size the more performance. This is strongly dependent on the number of DSP blocks that can be used. A comparison was made which sizes can be synthesized on these two

modules. The much bigger Mercury+ XU1 can fit any of the available DPU sizes (and has enough room for several instances of the DPU IP core) whereas the smaller Mars XU3 can only fit small DPU sizes.

## 2.11. Week 11

### 2.11.1. Enclustra hardware integration

The DPU integration into the reference designs available for the Mars XU3 and the Mercury+ XU1 continued in this week. For a successful integration the DPU user guide was studied and the design adopted to the respective hardware platform. Figure 2.13 shows an overview over the DPU subsystem. Two main blocks comprise this subsystem: the clock generation and the DPU IP core itself. For generating the correct clock signals at appropriate frequencies a 'Clocking Wizard' IP core is used. This block is configured to take an asynchronous reset and reference clock as input. Internally, a *Phase Locked Loop (PLL)* is used to generate faster clock frequencies which are needed for the DPU IP core. As this is a highly optimized hardware block, the frequency can be relatively high compared to the system block. In this design the DSP slices are clocked with a frequency of 650 MHz. For each clock signal an asynchronous reset needs to be generated as well. This can be done via the 'Processor System Reset' IP core. There, an asynchronous reset signal is generated for each respective clock domain. The data exchange of the DPU IP is handled by AXI interfaces.

### 2.11.2. Petalinux build

The next part of the process is building the Petalinux image files. After extracting the hardware description file and building a Linux OS the generated system can be simulated to verify correct functionality. The tool used for this is part of the Xilinx Petalinux tool flow called *Quick Emulator (QEMU)*. This tool is able of emulating a Xilinx ZYNQ system in software running on a Linux host system and offers some debugging tools. After some problems trying to emulate the kernel boot operation a Xilinx employee answered my forum question regarding my particular error message. As the DNNDK is not yet fully released, QEMU is not able to emulate the DPU, which is not mentioned in the documentation of either tool.

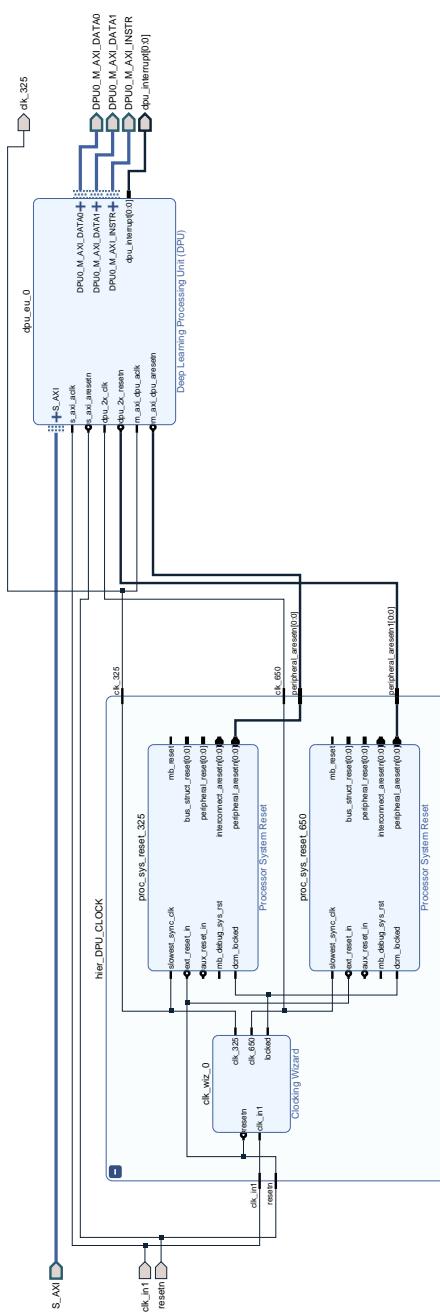


Figure 2.13.: DPU hierarchy system

### 2.11.3. Synthara visit and collaboration discussion

In this week on Thursday the visit to Synthara at ETH campus Irchel was scheduled and I went there with my supervisor Jelena and our marketing associate Melvin to discuss a possible collaboration in more detail. Synthara presented their rock-paper-scissors demonstrator and told us about their work flow which is quite similar to the Xilinx DNNDK. They also presented their custom neural network used for the demonstrator, which is a very basic convolutional neural network. We agreed to developing a demonstrator together for Embedded World 2020 and made a project timetable. The work was separated among the different areas of expertise of each company with Synthara handling training the neural network and providing their custom neural network accelerator IP. Enclustra would handle the necessary data collection and the hardware integration, both of which were my tasks.

## 2.12. Week 12

### 2.12.1. Mars ST3 demonstrator

This week the first samples of the Mars ST3 base board were available at the company after the usual bring up process for internal use. The bring up typically involves testing the hardware as thoroughly as possible and weed out errors in the *Printed Circuit Board (PCB)* design and the providing reference designs for the new board, which is compatible with all Mars FPGA modules. My task was to make a small AI demonstrator using this new base board and a Mars XU3 module. The reason for choosing the XU3 with the Zynq MPSoC instead of the smaller Zynq 7000 SoC was the compatibility of the Xilinx DPU IP core. The publicly released version was only suitable for Zynq Ultrascale devices. Support for Zynq 7000 SoC would be added in a future release. I decided to port the DNNDK sample applications to the Enclustra hardware. There were several reasons for this:

- As the CEO of Enclustra would be attending a Xilinx conference in early June, having an AI demonstrator to showcase would be great. This left only a few weeks for development.
- I was already familiar with the Xilinx DNNDK work flow and due to the timing constraints this course of action was the most promising.

- No neural network needed to be trained as the already trained models from the DNNDK can be used.
- The necessary hardware integration was completed successfully in the previous weeks for the XU3 module and therefore could be used for porting the sample applications to the complete hardware design with the new Mars ST3 board.

The main difficulty was posed by the Petalinux implementation as there are no official guidelines by Xilinx for custom boards. However, the official Xilinx tutorial found at <https://github.com/Xilinx/Edge-AI-Platform-Tutorials/tree/master/docs/DPU-Integration> provides an excellent starting point for custom projects. After a few adjustments to the device tree specific to the Enclustra hardware and Petalinux configuration adjustments for the exported .hdf file, the resnet50 image classification application was successfully running on the Enclustra hardware. The setup is shown in figure 2.14. This first version of the demonstrator consists of a *DisplayPort (DP)*

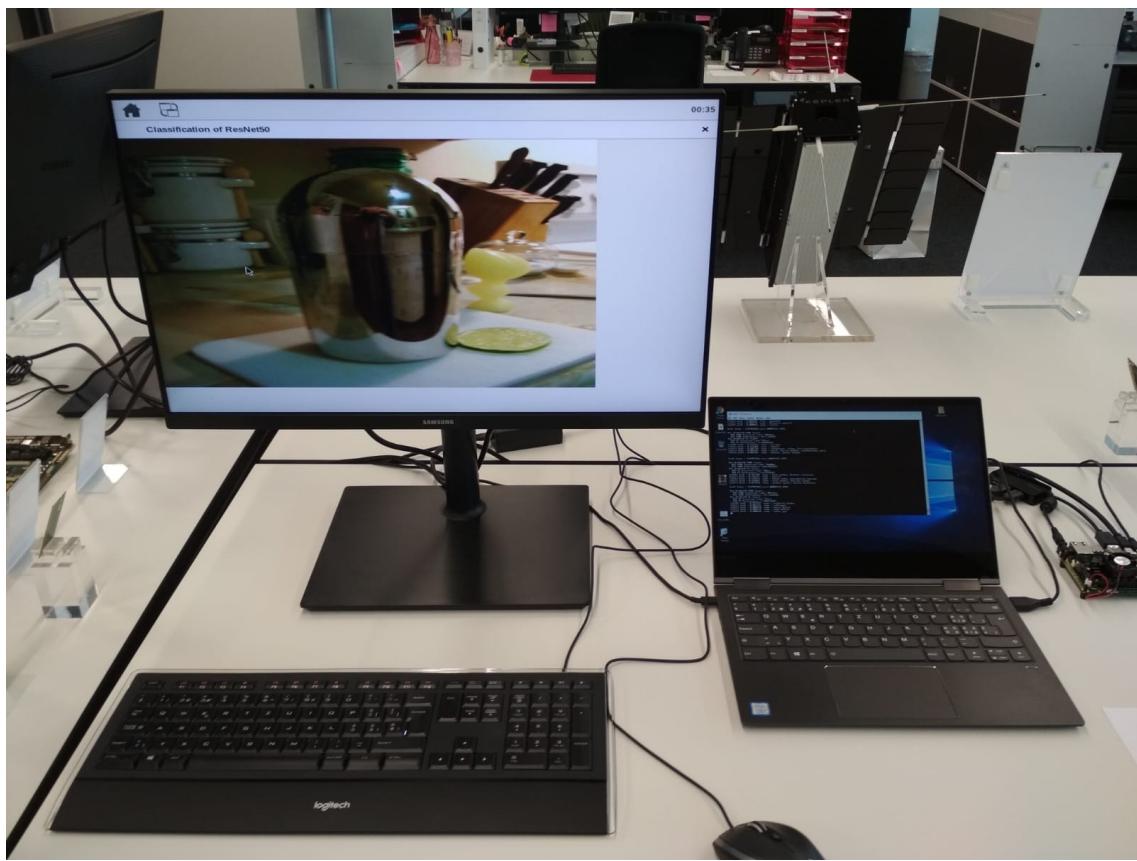


Figure 2.14.: resnet50 demonstrator overview

## 2. Internship report

---

monitor, the Mars ST3 base board, the Mars XU3 module and a serial connection to control the application via terminal from the host PC (adding keyboard and mouse to the FPGA module itself would make this step obsolete). A toggle button was implemented to allow switching between two modes, going through the data set one by one or classifying all images in the data set as fast as possible. One mode is to show the correct result of the classification as otherwise the classification is too fast to observe by eye. The other mode is to show the classification speed. An example for this can be seen in figure 2.15.

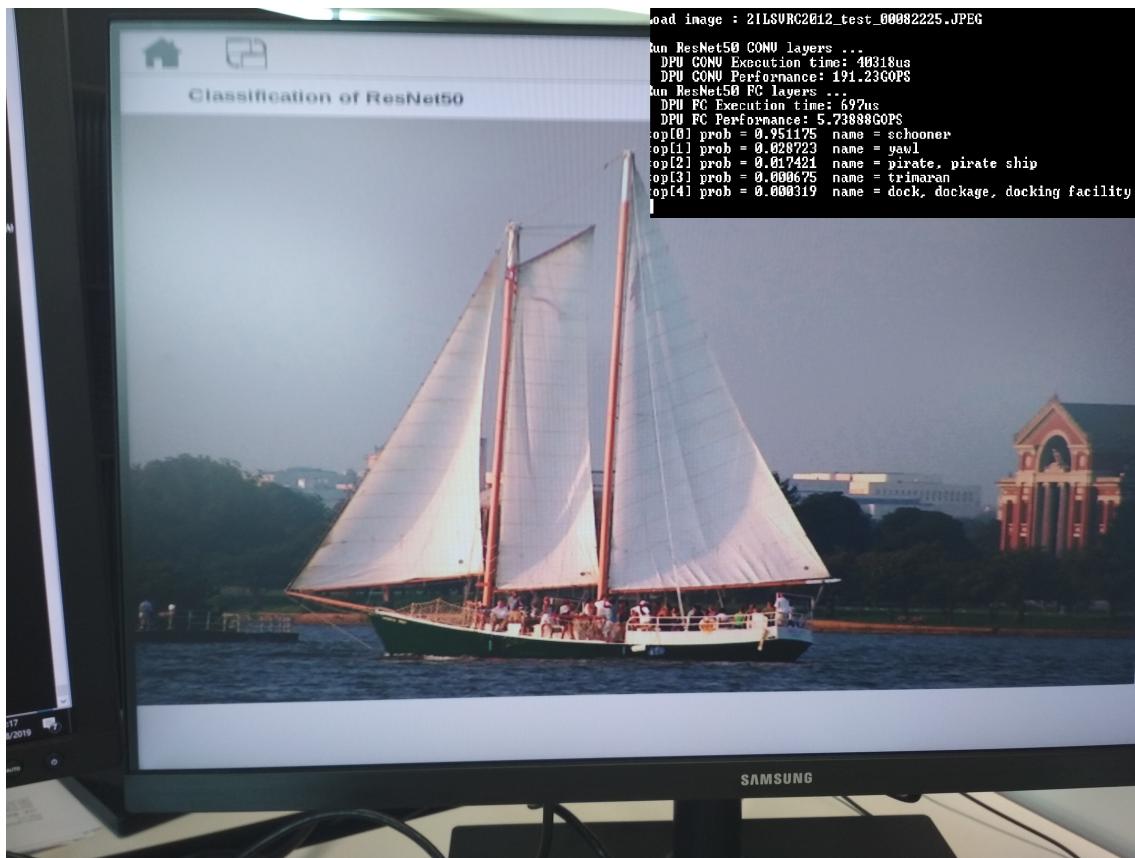


Figure 2.15.: resnet50 classification example

## 2.13. Week 13

### 2.13.1. Mars ST3 demonstrator preparation

The deadline to finish the AI demonstrator was this week Tuesday. So in the beginning of the week I worked on polishing the resnet50 example to have a smooth demo application. Furthermore, the face detection demo was also ported to the Enclustra hardware. The camera used was the same one from the Xilinx evaluation kit. A USB 3.0 3.4 MP camera module. The reason for using the USB camera for the demonstrator again was due to the strict timing constraints for the project. Furthermore, the camera used utilizes the standard V4L2 Linux kernel driver. This enables plug and play of the camera module and auto detection of the Petalinux OS. Therefore, no additional driver needed to be written. The applications were written using the Xilinx SDK against the exported root file system and libraries from the Petalinux build. In order to have access to all of the necessary functions of the DNNDK the correct Linker flags and environmental variables needed to be set. Additionally, the utilized libraries needed to be specified. The resnet50 application and the face detection application have different requirements of course, so two applications have been implemented with the SDK. The output of the DNNDK compiler was included as well into the project. These output binary files are the result of the neural network compilation containing the neural network model. Compiling the whole application combines these DNNDK binary files with the application binary file into a hybrid binary file, that can be used to run the application. Pose detection as a third demonstration application was also started to be implemented but ultimately forfeited due to the limited time. Consequently, the demonstrator has two sample applications, image classification using resnet50 and face detection, running on Enclustra hardware. All the project files and scripts necessary to replicate the design have been collected and documented for re-usability.

### 2.13.2. ROCK-PAPER-SCISSOR demonstration data collection preparation

The second part of the week was used to prepare the data collection for the rock-paper-scissors demo. After the Synthara visit the information of how to obtain a good quality data set were shared and it was my task to create the setup for the data collection. First of all, the number of classes needed to be defined: There was one

class for each of the correct symbols (rock, paper, scissors), one illegal symbol class and one background class, resulting in a total of five classes. Figure 2.16 summarizes the data set structure and points out the requirements for a good quality data set.

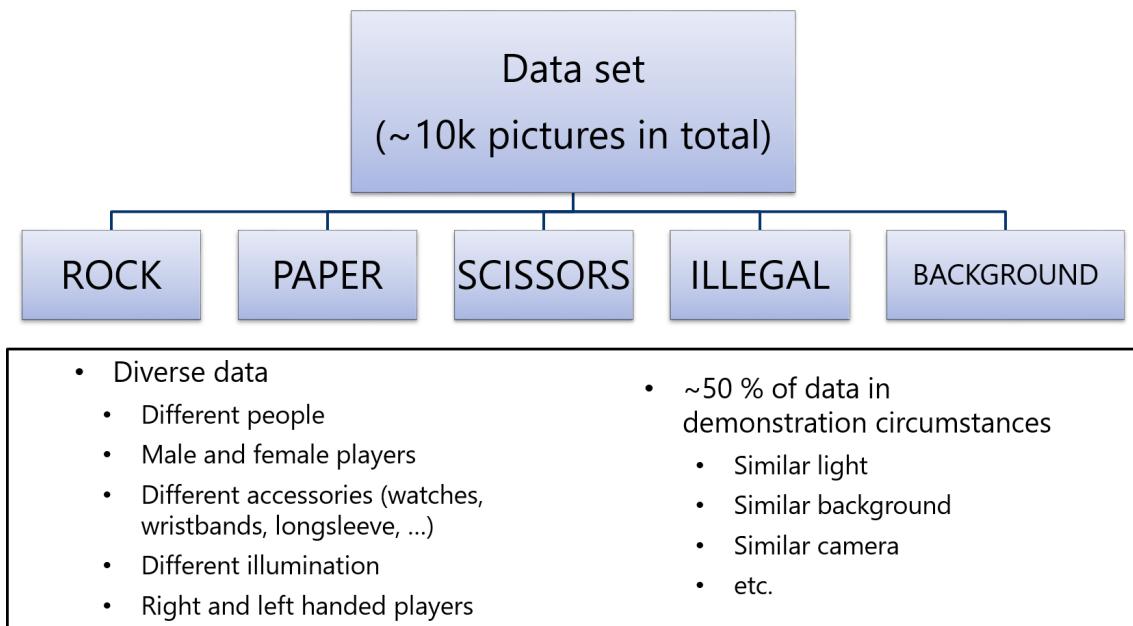


Figure 2.16.: Data set overview and requirements for a good quality data set

## 2.14. Week 14

### 2.14.1. ROCK-PAPER-SCISSOR demonstration data collection preparation

The need for a quality data set and the amount of data that needed to be collected made automation of the process necessary to speed things up. The hardware that we wanted to use for the demonstrator was the Mars XU3 module and the Mars ST3 base board. Furthermore, the RPi camera utilizing a MIPI connector was to be used. The easiest way to obtain the necessary amount of data (about 10k pictures in total) needed was to use the Raspberry Pi together with the camera. The operating system on the Raspberry Pi provides all the functions to obtain videos quickly and extract frames from videos. For the collection of a good quality data set variation

is key. Therefore, the contribution of as many people as possible is necessary. The setup for collecting the data set looked as follows:

- take videos of each subject
- 150 seconds per symbol per subject
- extract frames from the video
- structure the data so it gets labeled directly (ROCK, PAPER, SCISSORS, ILLEGAL)

## 2.14.2. Shell scripting and presentation on data set collection

To automatize this process, a shell script was written handling all of the tasks mentioned above. The script generates the necessary folder structure, takes the video and saves it. A conversion from .h264 to .mp4 format is done to easier extract the frames from the video. After discussing the frame size with Synthara, a video resolution of 320x240 with a framerate of 60 fps was chosen. This is the lowest resolution supported by the camera drivers for the Raspberry Pi. The test setup was build up in the kitchen area, so as to get as many people as possible to contribute to the data set collection. In order to have everybody know what to do, a short presentation was prepared to inform the colleagues of the purpose of this data set collection and also the procedure. Details were given about what constitutes a quality data set and how to obtain it. Emphasis was laid upon how to move the subjects hand and arm in order to generate useful data. Emphasis was laid on not moving the hand out of the camera frame, as this would generate a lot of falsely labeled data, which in turn degrades the network performance during training significantly. A diagram of the data collection setup is shown in figure 2.17.

# 2.15. Week 15

## 2.15.1. Data set collection

The majority of this week was spent collecting the data set. Therefore, the data collection setup was build up in the kitchen area and various employees were asked to participate. I went from one to the after I was done with one subject. A total of 10 minutes per subject of data was captured via video. As mentioned before, a

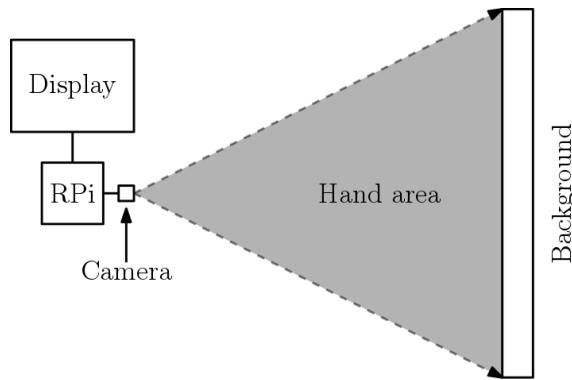


Figure 2.17.: Data set collection setup

Raspberry Pi was used with the script I created to collect the data and automatize as much of the process as possible. During the data set collection I assisted in keeping the movements of the subject in check and talk them through the collection process. For each of the symbols (ROCK, PAPER, SCISSORS, ILLEGAL) a video of 150 seconds was recorded. After each symbol, a short break was incorporated, as this seemingly small time period already puts a strain on the arm. This process was repeated for any employee that was available. A monitor was used to directly show the camera video feed so that the position of the hand within the frame was visible and could be easily corrected.

### 2.15.2. Intel ML seminar in Lausanne

On Wednesday, I traveled to Lausanne to attend an Intel seminar on AI inference with Intel products. The main reason for going to this event was to acquire more knowledge of the OpenVINO toolkit for FPGA. In addition, a block of the seminar dealt with data set preparation and augmentation for training neural networks. One of the key methods to create the necessary amount of data is data augmentation. This can be described as oversampling the data set, basically creating more data from existing data by rotating, shifting, zooming and other image manipulations. In this way, the data set can be evenly balanced as well so that each class has a similar amount of images. The data set is then separated into three subsets, which are shown in figure 2.18.

- training set: data used to train the network
- validation set: subset of the test set used for hyperparameter tuning (learning

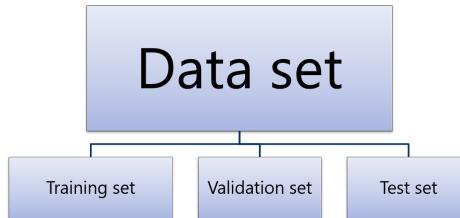


Figure 2.18.: Data set segmentation

rate, momentum, etc.)

- test set: data that the network has not seen during training, which is used for a full evaluation of the performance of the network

The data augmentation process only applies to the training set. As to new information regarding the OpenVINO toolkit, there was no additional information given as to what is available online. Overall, the focus was not on FPGA programming. The main idea of Intel is to provide a unified platform to develop AI applications using OpenCL and then provide plugins to optimize the device independent code for the individual hardware, like FPGAs. The rest of the week was spent collecting more data from Enclustra employees with the created data collection setup.

## 2.16. Week 16

### 2.16.1. Data set collection

The final data set collection amounted to 11503 images with a distribution of 2838 images for the 'ROCK' class, 2820 images for the 'PAPER' class, 2814 images for the 'SCISSORS' class and 2749 images for the 'ILLEGAL' symbol class. The 'BACKGROUND' class consists of 282 images. To add to this number a publicly available data set was also added to the collection containing 2188 images in total distributed almost equally amongst the classes 'ROCK', 'PAPER' and 'SCISSORS'. The complete data set was then handed off to Synthara for the training process as agreed when starting the collaboration. The data set was augmented as described in the chapter before and then iteratively quantized to compress the network.

### 2.16.2. Robot hand ordering

As the training process would take up to two weeks at least, the next reasonable step was to work on the hardware setup with the Enclustra board. A crucial part for this is the robotic hand. At first, I conducted research into the available possibilities: There are numerous options ranging from cheap ( $\sim 50 \$$ ) servo motor controlled basic hands to more sophisticated 3D-printed medical variants (several thousand \$). For this demonstrator the key parameter was speed, so that the robot hand would react in time to the classification results provided by the neural network. The key parameter for this is how quickly the attached servos can go from  $0^\circ$  to  $180^\circ$ . The typical value for this is around 1 to 2 ms for the cheaper robotic hands. Figure 2.19 shows the chosen robotic hand. The benefits of this system is the added

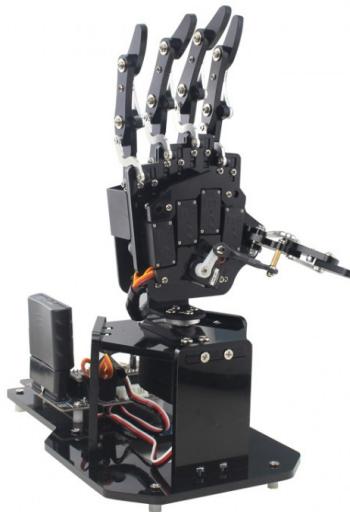


Figure 2.19.: Robotic hand chosen for the demonstrator

PCB together with a microcontroller, so that the basic functionality can be tested easily and directly. The servos used in this hand are very standard servo motors with three connectors, VCC, GND and DATA. The DATA line is used to determine the angle which the servo has to drive to. The signal used for this is a PWM. Figure 2.20 shows the mapping of the PWM signal to the motor position.

### 2.16.3. Internship documentation

The rest of the week was spent on writing documentation of all of the tasks I completed during my internship so far and integrate this information into the company Wiki.

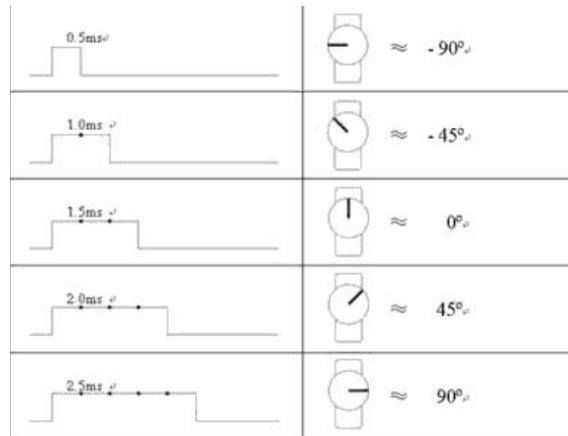


Figure 2.20.: PWM signal servo motor position mapping

## 2.17. Week 17

### 2.17.1. Robot hand ordering

More expensive robot hands were also investigated. These are mostly 3D printed hands capable of much more sophisticated movements as their intended use is in medicine as limb replacements and bionics. The cost was the main deciding factor in the end and after discussions with my supervisor and the CEO of Enclustra the order was placed for a cheap robot hand from a Chinese manufacturer. This also meant several work days of shipping had to be taken into account.

### 2.17.2. Synthara update

The first training results were also shared by Synthara. The training was progressing well and a floating point base line was established. The floating base line with the data set so far is: 100 % train accuracy, 88 % validation accuracy and 75 % test accuracy. The goal before going to the next step is to bring test accuracy above 85 %. However, there was still fine tuning of the parameters to do and also the quantization step.

### 2.17.3. Internship documentation

The rest of the week was spent on writing documentation of all of the tasks I completed during my internship so far and integrate this information into the company Wiki. The main part here was documenting all the steps included in recreating the DNNDK

design for the Mars ST3 demonstrator board. A picture of the hardware is shown in figure 2.21.

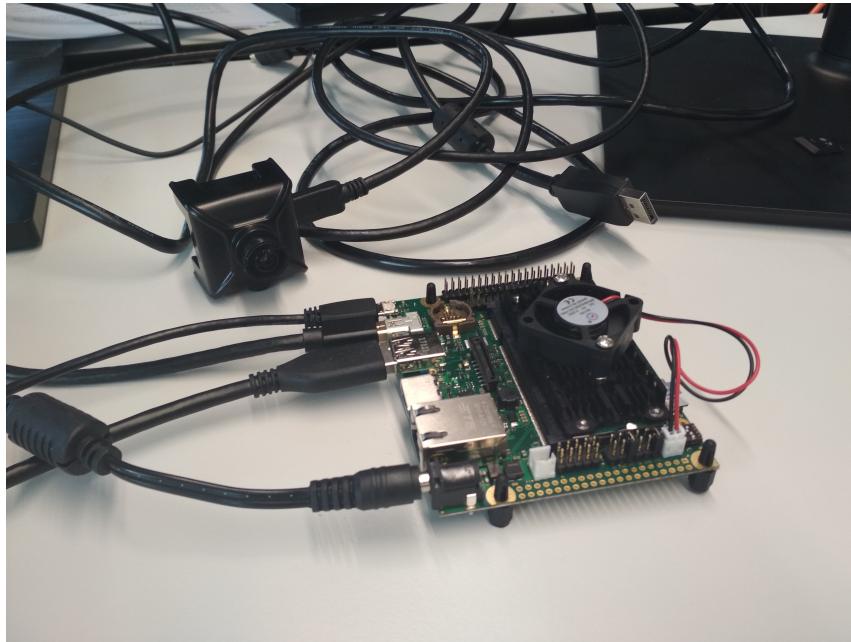


Figure 2.21.: Mars ST3 DNNDK demonstrator

## 2.18. Week 18

### 2.18.1. Internship documentation

While waiting for further training results, some more documentation was worked on. A company profile for Synthara was written on the internal Wiki page as well as a summary of the collaboration project.

### 2.18.2. Synthara update

The new training results were also shared by Synthara. After establishing the floating point base line, the limited precision training phase was started. One characteristic of running neural network inference is that the network performance does not degrade when reducing number precision. This is actually the key for making FPGAs a viable platform for ML applications among other things. The reason for this is that reducing the precision can be thought of as adding noise to the neural network and

helps generalize the network performance, preventing overfitting and resulting in the same accuracy (or slightly worse or slightly better) in reduced precision compared to 32 bit floating point precision. Reducing the number of bits needed to represent the numbers is beneficial on the whole system as all hardware components needed for computation need less resources. The results with limited precision are as follows: training 79 %, validation 92 % and test accuracy 86 %. Table 2.18.2 shows the number of bits used for each parameter of the neural network.

	number of bits
CNN weights	5 (6 for the first layer)
FC weights	11
Activations	16
Biases	16

## 2.19. Week 19

### 2.19.1. Website content about AI

As part of the official Enclustra newsletter and to leverage the new Mars ST3 AI demonstrator, the 'Projects' section of the website was updated. The main purpose is to show that Enclustra also has some expertise on this new emerging field that is AI. The website content was created together with the head of marketing of the company.

### 2.19.2. Embedded World demo discussion

The setup for the Embedded World demonstrator was also discussed. The whole setup of the demonstrator within the Enclustra booth had to be addressed: the position of the hand, the background, the position of the FPGA and so on. Figure 2.22 shows the final setup for the demonstrator. This is just a sketch and subject to further change, but the general setup would look more or less like this. The monitor could be used to show a video feed with a virtual hand. This might be useful because the neural network and the image processing on the FPGA are estimated to be much faster than the robot hand assuming its position, so a possibility to show the output of the neural network on the screen was added to the setup.

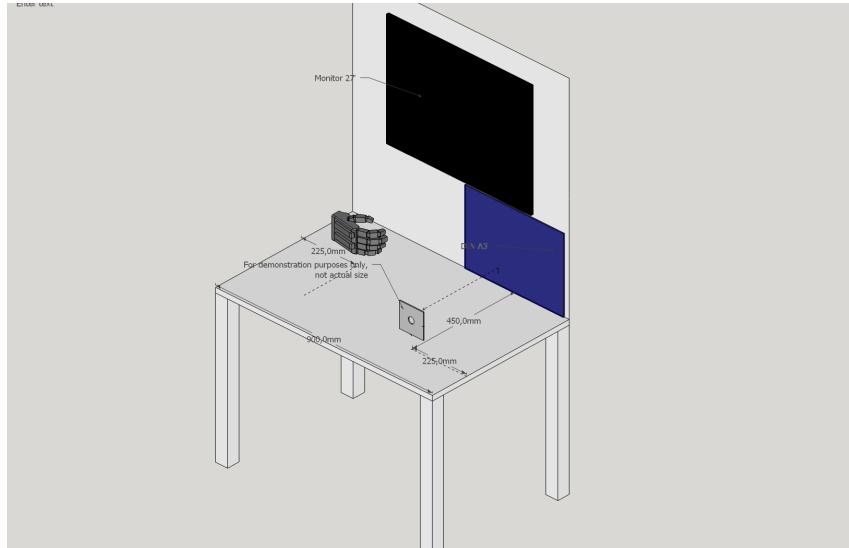


Figure 2.22.: Embedded World demonstration setup

### 2.19.3. DNNDK Tensorflow workflow

With the release of a new version of the DNNDK support for Tensorflow was added. Tensorflow is the most popular ML framework for training neural networks. It is developed by Google and provides a Python API to simplify neural network training by providing optimized libraries to handle typical neural network operations. A Xilinx tutorial was also posted alongside the new release and with some modifications I managed to follow along. First of all I trained a convolutional neural network to recognize handwritten human digits using the standardized *Modified National Institute of Standards and Technology database (MNIST)* data set. Then, the trained model was exported and could be fed into the DNNDK tool chain to generate a fixed point 8 bit quantized neural network model ready for deployment on an FPGA.

## 2.20. Week 20

### 2.20.1. DNNDK Tensorflow workflow

The second part of the tutorial consists of training a more sophisticated neural network capable of image recognition using the *Canadian Institute For Advanced Research (CIFAR)-10* data set. It contains 60.000 32x32 color images in 10 different classes. Figure 2.23 shows these 10 different classes and examples for each class. Compared to the MNIST data set network it is much bigger and the training

consequently takes a lot longer. The resulting trained weights and the network structure were again fed into the DNNDK tool chain to obtain the necessary binary file that can be used in the Xilinx SDK to create ML applications. The neural network specific tasks can be off-handed to the DPU IP core. A documentation for the internal Wiki page of Enclustra was added and the small adjustments to the original Xilinx tutorial documented.

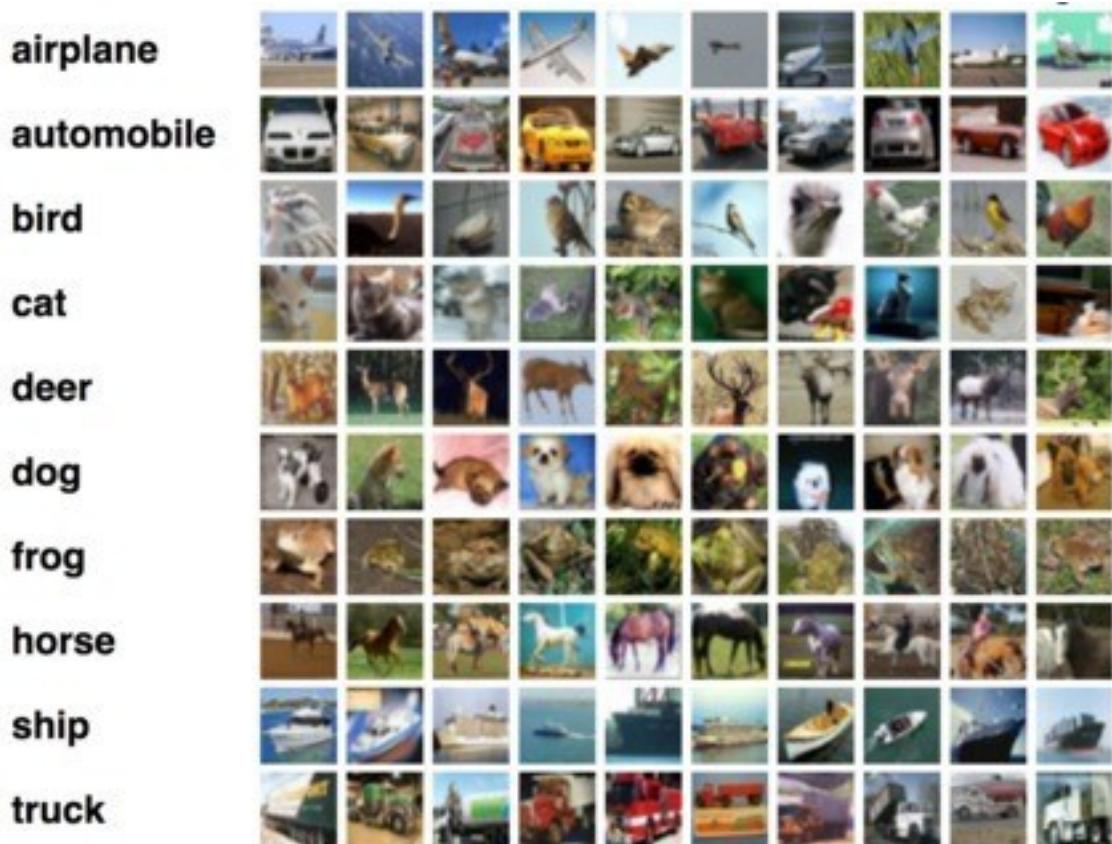


Figure 2.23.: CIFAR-10 data set example

## 2.20.2. Data set publication

In order to show Enclustras interest in ANN the decision was made to publish the collected Rock-Paper-Scissors data set on the Kaggle website. Kaggle offers a community and platform dedicated to neural network research. Various data sets are available for free to advocate and challenge neural network research. A detailed

description was written containing the amount of data collected as well as the method of collection.

### **2.20.3. Robot hand assembly**

The robot hand arrived in this week and needed to be assembled. As the description and all of the instructions were in Chinese, this took longer than anticipated. Moreover, the data was only available via Baidu which made acquisition difficult because a Chinese phone number is needed to make a Baidu account. Fortunately, Enclustra also has an office in Shenzhen and I asked a Chinese employee there to download all of the files and make them available to me.

### **2.20.4. Mettler Toledo AI demo interest**

During my internship Enclustra celebrated its 15th anniversary and during that visit, a Mettler Toledo official showed interest in the Mars ST3 AI demonstrator. The Mars ST3 demonstrator was handed over to Mettler Toledo so they could present it within their company. A visit for next week was also arranged.

## **2.21. Week 21**

### **2.21.1. Robot hand testing and documentation**

After the robot hand was assembled I tested the functionality with the included Arduino microcontroller. The servo motors controlling the fingers of the robot hand and the palm are using PWM to move the fingers. The pulse width of the modulation scheme corresponds to the degree the motor has to rotate to. The pulse width is controlled by the Arduino in this case.

### **2.21.2. Mettler Toledo AI visit**

As mentioned before I visited Mettler Toledo with my supervisor Jelena to showcase our AI demonstrator. We discussed possible scenarios where AI could be interesting for a customer. The main focus of Mettler Toledo is industrial scales and measurement equipment. Image recognition in this case could be a prime use case for ANN.

### 2.21.3. FPGA PWM VHDL module

In order to get rid of the Arduino, the PWM scheme needed to be created on the FPGA. A VHDL module was written to make the Arduino obsolete. The



Figure 2.24.: PWM module

function of the VHDL module is shown in figure 2.24. The *dutycycle* is given and *pwmout* is generated. This is done by using two counters, one of which is used to divide the system clock and generate a slower pulse (*clkdiv*). This slower pulse is used to increase the second counter. In this way, *pwmout* can be generated with a conditional statement depending on the state of the second counter.

## 2.22. Week 22

### 2.22.1. FPGA PWM VHDL module

Further work was put into the PWM module. After writing synthesizable and syntax error free code, a testbench was written for the PWM module. Figure 2.25 shows

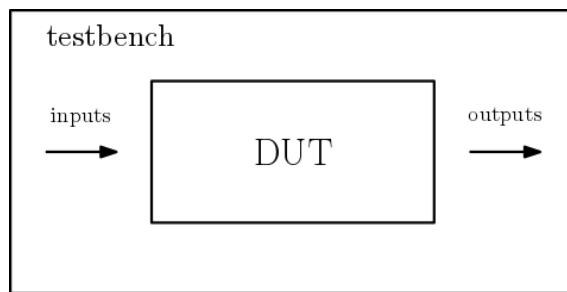


Figure 2.25.: Hardware design language testbench overview

the structure of a VHDL module testbench. A testbench is a VHDL entity without inputs or outputs. The *Device Under Test (DUT)* is instantiated as a component within the testbench and stimuli are provided as signals to the inputs of the DUT. By observing the outputs, the functionality of the design can be verified in simulation. Modelsim was used as the tool for testing and debugging the PWM module.

### **2.22.2. Synthara design porting**

I received the synthesized netlist from Synthara containing their IP core and started to incorporate their module into the MIPI design.

### **2.22.3. Xilinx proprietary software trial**

Enclustra as a close Xilinx partner could register for early access to a new unified design software. Any further information is part of a non disclosure agreement and I can not go into further detail in my report.

## **2.23. Week 23**

### **2.23.1. Synthara design porting**

The Synthara IP core was implemented into the MIPI hardware design. However, the design could not be fit on the FPGA because the Mars XU3 FPGA has a low LUT and DSP count in comparison to other Ultrascale+ architectures. The design was synthesized successfully after the block design was created in Vivado. After discussing this issue with Synthara the possibility of getting a smaller version of the accelerator from Synthara was investigated.

### **2.23.2. Mars XU3 alternative**

If it would not be possible to use the Mars XU3 an alternative solution was investigated. The Mercury XU1+ features a much bigger FPGA and was chosen as a possible alternative. Figure 2.26 shows the difference between the two FPGAs used in the Mars XU3 and the Mercury XU1+. The unaltered Synthara design fits easily on the Mercury XU1+ module. One downside of this adjustment would be the usage of the Mercury+ PE1 which does not have a MIPI port. In this case we would use the USB 3.0 camera for the camera feed. This would mean recollecting part of the data set with the USB 3.0 camera, so this was put on hold for now and I waited for Synthara to reduce the size of their accelerator. This would allow the usage of the intended Mars XU3 module.

	Device Name <sup>(1)</sup>	ZU2EG	ZU3EG	ZU4EG	ZUSEG	ZU6EG	ZU7EG	ZU9EG	ZU11EG	ZU15EG	ZU17EG	ZU19EG
Application Processor Unit	Processor Core									Quad-core ARM® Cortex™-A53 MPCore™ up to 1.5GHz		
	Memory w/ECC				L1 Cache 32KB I / D per core, L2 Cache 1MB, on-chip Memory 256KB							
	Processor Core									Dual-core ARM Cortex-R5 MPCore™ up to 600MHz		
	Memory w/ECC				L1 Cache 32KB I / D per core, Tightly Coupled Memory 128KB per core							
Real-Time Processor Unit	Graphics Processing Unit									Mali™-400 MP2 up to 667MHz		
	Memory									L2 Cache 64KB		
Graphic & Video Acceleration	Dynamic Memory Interface				x32/x64: DDR4, LPDDR4, DDR3, DDR3L, LPDDR3 with ECC							
	Static Memory Interfaces									NAND, 2x Quad-SPI		
External Memory	High-Speed Connectivity			PCIe® Gen2 x4, 2x USB3.0, SATA 3.1, DisplayPort, 4x Tri-mode Gigabit Ethernet								
	General Connectivity			2xUSB 2.0, 2x SD/SDIO, 2x UART, 2x CAN 2.0B, 2x I2C, 2x SPI, 4x 32-bit GPIO								
Connectivity	Power Management									Full / Low / PL / Battery Power Domains		
	Security									RSA, AES, and SHA		
Integrated Block Functionality	AMS - System Monitor									10-bit, 1MSPS – Temperature and Voltage Monitor		
	PS to PL Interface									12 x 32/64/128b AXI Ports		
Programmable Logic (PL)	System Logic Cells (K)	103	154	192	256	469	504	600	653	747	926	1,143
	CLB Flip-Flops (K)	94	141	176	234	429	461	548	597	682	847	1,045
	CLB LUTs (K)	47	71	88	117	215	230	274	299	341	423	523
	Max. Distributed RAM (Mb)	1.2	1.8	2.6	3.5	6.9	6.2	8.8	9.1	11.3	8.0	9.8
Memory	Total Block RAM (Mb)	5.3	7.6	4.5	5.1	25.1	11.0	32.1	21.1	26.2	28.0	34.6
	UltraRAM (Mb)	-	-	13.5	18.0	-	27.0	-	22.5	31.5	28.7	36.0
Clocking	Clock Management Tiles (CMTs)	3	3	4	4	4	8	4	8	4	11	11
	DSP Slices	240	360	728	1,248	1,973	1,728	2,520	2,928	3,528	1,590	1,968
Integrated IP	PCI Express® Gen 3x16	-	-	2	2	-	2	-	4	-	4	5
	150G Interlaken	-	-	-	-	-	-	-	1	-	2	4
	100G Ethernet MAC/PCS w/RS-FEC	-	-	-	-	-	-	-	2	-	2	4
	AMS - System Monitor	1	1	1	1	1	1	1	1	1	1	1
Transceivers	GTH 16.3Gb/s Transceivers	-	-	16	16	24	24	24	32	24	44	44
	GTY 32.75Gb/s Transceivers	-	-	-	-	-	-	-	16	-	28	28
Speed Grades	Extended <sup>(2)</sup>	-1	-2	-2L		-1	-2	-2L	-3		-2	-2L
	Industrial									3		

Notes:  
1. For full part number details, see the Ordering Information section in [DS891](#), Zynq UltraScale+ MPSoC Overview.  
2.-ZLE (I) = 0°C to 110°C). For more details, see the Ordering Information section in [DS891](#), Zynq UltraScale+ MPSoC Overview.

Figure 2.26.: Xilinx MPSoC product overview [5]

## 2.24. Week 24

### 2.24.1. MIPI demonstrator Petalinux Mars ST3

During the wait for Synthara to adjust the accelerator IP core the demonstrator for the Mars ST3 board was adjusted to use the MIPI camera. For this task, adjustments to the Petalinux system needed to be made. The device tree as well as the Linux driver needed to be modified and added respectively. The camera driver for the Sony sensor used in the Raspberry Pi camera is controlled via the I2C protocol. I2C is a simple master-slave interface for data communication. It uses only two wires, SCL (Serial Clock Line) and SDA (Serial Data). The normal state is defined by both lines being high. The master initiates the communication by generating the start condition *S* followed by the slaved device address (*B1*). Read or write is defined by the 0 bit of the slave address where 0 means 'write'. Once all bytes are read or written (*B<sub>N</sub>*) the stop condition (*P*) is generated. This signals the end of the communication transaction.

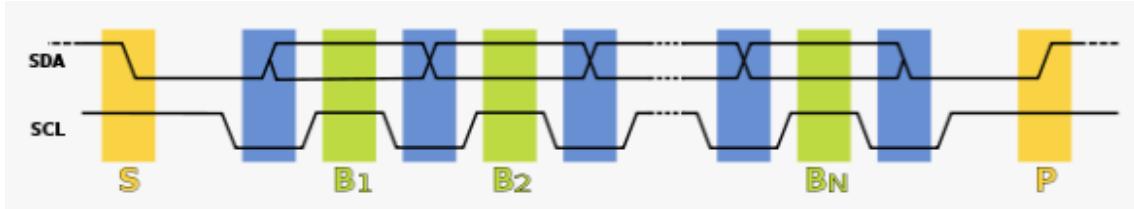


Figure 2.27.: I2C protocol data transfer

### 2.24.2. I2C debugging

Two problems occurred during testing the camera driver and the camera itself. The first one is a hardware specific problem of the first revision of the Mars ST3 board where the MIPI connector pins are not configured in a standard way and the need to be adjusted. The I2C device also needed to be included in the device tree. After these adjustments, the driver could be loaded successfully on the board. The second problem is that the test application for the testing the camera feed gets stuck at preparing the data transfer.

## 2.25. Week 25

### 2.25.1. New DPU version

Xilinx released a new version of the DPU IP core showing much better performance. In this week I started to adjust the original AI demonstrator in order to use the new, more powerful version. Figure 2.28 shows the achievable performance for different image classification and object detection state-of-the-art neural networks. The workload as well as the input image resolution is shown together with the framerate that is acquired. The hardware design and Petalinux needed to be adjusted for the new DNNDK version. Previously, the softmax function needed to be calculated in the CPU, whereas now, this can also be handled by the DPU. It is up to 160 times faster to calculate the softmax in hardware than it is in software.

### 2.25.2. ECC 2019 presentation

Enclustra is a sponsor of the Electronic Computing Conference in Winterthur, which is one of the biggest embedded computing conferences in Switzerland. I was tasked

<b>Network Model</b>	<b>Workload (Gops per image)</b>	<b>Input Image Resolution</b>	<b>Frame per second (FPS)</b>
Inception-v1	3.2	224*224	405
ResNet50	7.7	224*224	175
SqueezeNet	0.698	224*224	1048
Tiny-YOLO	6.97	448*448	220
YOLO-V2	82.5	640*640	24
Pruned YOLO-V2	18.4	640*640	120
YOLO-V3	53.7	512*256	43
Pruned YOLO-V3-	4	512*256	115

Figure 2.28.: DPU performance overview [4]

with presenting the possibilities of AI on FPGAs. I started putting together a presentation with a brief company profile, going to introducing artificial neural networks and ultimately focusing on FPGAs as a viable platform for neural network inference.

## 2.26. Week 26

### 2.26.1. ECC 2019 presentation

I continued working on the presentation for the ECC 2019. A section about available hardware platforms and their strengths and weaknesses in terms of neural network inference was added. This comparison could only be done in a qualitative manner as there is no standardized test for neural network workloads and thus making a quantitatively fair comparison is very difficult. Figure 2.29 shows a qualitative hardware comparison for the three major hardware platforms for running neural network inference in embedded devices. This qualitative study was conducted by looking at research papers as well as industry white papers and product pages. The main advantage of FPGAs is the flexibility of the devices. Low latency in conjunction with sensor integration and a high performance per Watt.

Requirements	GPU	FPGA	ASIC
Low latency		✓	✓
High throughput	✓	✓	✓
Power efficiency		✓	✓
Sensor fusion		✓	
Robustness		✓	✓
Programmability	✓	✓	
Flexibility	✓	✓	
Ease-of-use	✓		
(Development) cost	✓		

Figure 2.29.: Qualitative comparison of available hardware platforms

## 2.26.2. Petalinux debug

In order to get the new version of the DPU running on the Mars ST3 the hardware design in conjunction with the Petalinux device tree. The softmax integration also added another interrupt to the system which needed to be in sync with the device tree. After rebuilding the Petalinux system the two demos also needed to be adjusted to the new API DNNDK version.

## 2.27. Week 27

### 2.27.1. ECC 2019 presentation

The presentation deadline was this week and the ECC itself was to take place in the first week of September. I polished the presentation and discussed possible improvements with my supervisor. A test run to check the amount of time needed for the presentation was also held in front of my supervisor. The feedback was incorporated into the presentation slides. The final structure is as follows: company presentation, neural networks and neural network training, inference requirements, available hardware platforms, FPGA workflow and Xilinx DNNDK workflow in detail. In total, the presentation would be 20 minutes and 10 minutes are reserved for questions.

## 2.27.2. Mars ST3 demonstrator polishing

The previous version of the Mars ST3 DNNDK demonstrator was updated with the newer version. The main benefit is better performance and smoother operation. It was planned to use the demonstrator at the Enclustra booth at the ECC. This fits the topic of the presentation as well as it shows Enclustras interest and expertise in AI.

## 2.27.3. Synthara design porting

Synthara managed to provide a smaller version of their neural network accelerator core. I incorporated this new version into the MIPI design and now Vivado successfully fit the whole design on the Mars XU3 module. Figure 2.30 shows the usage report of

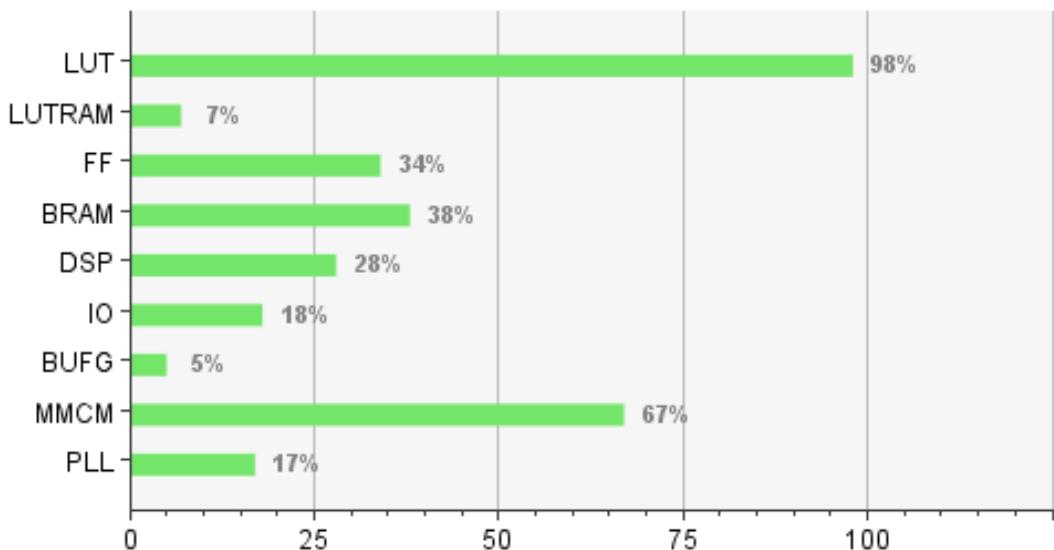


Figure 2.30.: Vivado usage report Rock-Paper-Scissor demonstrator

the available hardware resources on the used FPGA after the place and route step. The design barely fits the FPGA with a 98 % usage of the available LUTs. The reason why the DSP block count usage is so low is that Syntharas IP core is not optimized for FPGAs. So there is definitely some potential for improving the design when running on FPGAs.



# Bibliography

- [1] Loise Crocket et al. *Exploring Zynq MPSoC: With PYNQ and Machine Learning Applications*. Strathclyde Academic Media, 2019.
- [2] Devopedia. *Deep Learning Frameworks*. Ed. by Devopedia. Jan. 22, 2019. URL: <https://devopedia.org/deep-learning-frameworks>.
- [3] Intel. “AI on Intel - From data center to the edge - AI solutions using Intel architecture”. 2019.
- [4] Xilinx. *DPU for Convolutional Neural Network v1.2*. Ed. by Xilinx. Mar. 26, 2019. URL: [https://www.xilinx.com/support/documentation/ip\\_documentation/dpu/v3\\_0/pg338-dpu.pdf](https://www.xilinx.com/support/documentation/ip_documentation/dpu/v3_0/pg338-dpu.pdf).
- [5] Xilinx. *MPSoC product guide*. Ed. by Xilinx. 2016. URL: <https://www.xilinx.com/support/documentation/selection-guides/zynq-ultrascale-plus-product-selection-guide.pdf>.
- [6] Xilinx. *PG202 MIPI D-PHY*. Ed. by Xilinx. July 2, 2019. URL: [https://www.xilinx.com/support/documentation/ip\\_documentation/mipi\\_dphy/v4\\_1/pg202-mipi-dphy.pdf](https://www.xilinx.com/support/documentation/ip_documentation/mipi_dphy/v4_1/pg202-mipi-dphy.pdf).
- [7] Xilinx. *PG232 MIPI RX subsystem*. Ed. by Xilinx. Apr. 4, 2018. URL: [https://www.xilinx.com/support/documentation/ip\\_documentation/mipi\\_csi2\\_rx\\_subsystem/v3\\_0/pg232-mipi-csi2-rx.pdf](https://www.xilinx.com/support/documentation/ip_documentation/mipi_csi2_rx_subsystem/v3_0/pg232-mipi-csi2-rx.pdf).
- [8] Xilinx. *UG1354 Xilinx AI SDK User Guide*. Ed. by Xilinx. July 3, 2019. URL: [https://www.xilinx.com/support/documentation/user\\_guides/ug1354-xilinx-ai-sdk.pdf](https://www.xilinx.com/support/documentation/user_guides/ug1354-xilinx-ai-sdk.pdf).



# A. Workday reports

<b>Fr</b> 8.3	13:00 16:15 3,25  0000.909 (Enclustra AI Study)	002 (Testing DNNDK & revision on ZC104)
	08:00 12:30 4,50  0000.909 (Enclustra AI Study)	001 (Miscellaneous)
	Total/ 7,75	
<b>Th</b> 7.3	12:45 16:15 3,50  0000.909 (Enclustra AI Study)	002 (Testing DNNDK & revision on ZC104)
	08:00 12:15 4,25  0000.909 (Enclustra AI Study)	002 (Testing DNNDK & revision on ZC104)
	Total/ 7,75	
<b>We</b> 6.3	08:00 16:30 8,50  0000.909 (Enclustra AI Study)	005 (Market research AI and AI on FPGA)
	Total/ 8,50	
<b>Tu</b> 5.3	13:45 16:30 2,75  0000.002 (Enclustra Human Resources)	003 (Einarbeitung)
	08:00 13:15 5,25  0000.002 (Enclustra Human Resources)	003 (Einarbeitung)
	Total/ 8,00	
<b>Mo</b> 4.3	14:00 18:00 4,00  0000.002 (Enclustra Human Resources)	003 (Einarbeitung)
	09:00 13:00 4,00  0000.002 (Enclustra Human Resources)	003 (Einarbeitung)
	Total/ 8,00	

Figure A.1.: Week 1

<b>Fr</b> 15.3	13:00 17:00 4,00  0000.909 (Enclustra AI Study) 003 (Presentation about AI on FPGA)	int Presentation Introduction to AI
	08:00 12:30 4,50  0000.909 (Enclustra AI Study) 003 (Presentation about AI on FPGA)	int Presentation Introduction to AI
	Total/ 8,50	
<b>Th</b> 14.3	15:00 17:30 2,50  0000.909 (Enclustra AI Study) 001 (Miscellaneous)	int Presentation Introduction to AI
	08:15 12:30 4,25  0000.909 (Enclustra AI Study) 001 (Miscellaneous)	int Started to prepare the presentation / Wiki Updates
	Total/ 6,75	
<b>We</b> 13.3	15:00 18:15 3,25  0000.909 (Enclustra AI Study) 001 (Miscellaneous)	int FPGA examples and samples
	08:00 13:00 5,00  0000.909 (Enclustra AI Study) 001 (Miscellaneous)	int FPGA examples and samples
	Total/ 8,25	
<b>Tu</b> 12.3	13:45 17:30 3,75  0000.909 (Enclustra AI Study) 002 (Testing DNNDK & revision on ZC104)	int FPGA examples and samples / Out of Box testing with Melvin
	07:45 13:00 5,25  0000.909 (Enclustra AI Study) 002 (Testing DNNDK & revision on ZC104)	int FPGA examples and samples / Out of Box testing with Melvin
	Total/ 9,00	
<b>Mo</b> 11.3	15:00 17:30 2,50  0000.909 (Enclustra AI Study) 002 (Testing DNNDK & revision on ZC104)	int FPGA examples and samples / Wiki Updating
	08:30 13:30 5,00  0000.909 (Enclustra AI Study) 002 (Testing DNNDK & revision on ZC104)	int FPGA examples and samples / Wiki Updating
	Total/ 7,50	

Figure A.2.: Week 2

## A. Workday reports

---

Fr 22.3	14:30 - 16:00	1,50	0000.909 (Enclustra AI Study)	005 (Market research AI and AI on FPGA)	int
	09:00 - 14:30	5,50	0000.909 (Enclustra AI Study)	003 (Presentation about AI on FPGA)	int
	Total/	7,00			
Th 21.3	15:00 - 16:15	1,25	0000.909 (Enclustra AI Study)	001 (Miscellaneous)	int
	14:15 - 15:00	0,75	0000.909 (Enclustra AI Study)	003 (Presentation about AI on FPGA)	int
	08:15 - 14:00	5,75	0000.909 (Enclustra AI Study)	003 (Presentation about AI on FPGA)	int
	Total/	7,75			
We 20.3	13:00 - 17:45	4,75	0000.909 (Enclustra AI Study)	003 (Presentation about AI on FPGA)	int
	09:30 - 12:30	3,00	0000.909 (Enclustra AI Study)	003 (Presentation about AI on FPGA)	int
	Total/	7,75			
Tu 19.3	13:00 - 17:15	4,25	0000.909 (Enclustra AI Study)	003 (Presentation about AI on FPGA)	int
	08:00 - 12:30	4,50	0000.909 (Enclustra AI Study)	003 (Presentation about AI on FPGA)	int
	Total/	8,75			
Mo 18.3	13:00 - 17:15	4,25	0000.909 (Enclustra AI Study)	003 (Presentation about AI on FPGA)	int
	08:00 - 12:30	4,50	0000.909 (Enclustra AI Study)	003 (Presentation about AI on FPGA)	int
	Total/	8,75			

Figure A.3.: Week 3

Fr 29.3	13:15 - 16:45	3,50	0000.909 (Enclustra AI Study)	002 (Testing DNNDK & revision on ZC104)
	08:30 - 12:45	4,25	0000.909 (Enclustra AI Study)	003 (Presentation about AI on FPGA)
	Total/	7,75		
Th 28.3	13:00 - 16:45	3,75	0000.909 (Enclustra AI Study)	003 (Presentation about AI on FPGA)
	08:45 - 12:45	4,00	0000.909 (Enclustra AI Study)	003 (Presentation about AI on FPGA)
	Total/	7,75		
We 27.3	13:00 - 17:30	4,50	0000.909 (Enclustra AI Study)	003 (Presentation about AI on FPGA)
	08:45 - 12:30	3,75	0000.909 (Enclustra AI Study)	003 (Presentation about AI on FPGA)
	Total/	8,25		
Tu 26.3	13:00 - 18:30	5,50	0000.909 (Enclustra AI Study)	003 (Presentation about AI on FPGA)
	08:30 - 12:30	4,00	0000.909 (Enclustra AI Study)	003 (Presentation about AI on FPGA)
	Total/	9,50		
Mo 25.3	12:45 - 15:00	2,25	0000.909 (Enclustra AI Study)	003 (Presentation about AI on FPGA)
	08:00 - 12:30	4,50	0000.909 (Enclustra AI Study)	003 (Presentation about AI on FPGA)
	Total/	6,75		

Figure A.4.: Week 4

Fr 5.4	15:00 - 19:15	4,25	0000.909 (Enclustra AI Study)	001 (Miscellaneous)
	Total/	4,25		
Th 4.4	09:00 - 17:00	8,00	0000.909 (Enclustra AI Study)	001 (Miscellaneous)
	Total/	8,00		
We 3.4	13:15 - 15:45	2,50	0000.909 (Enclustra AI Study)	001 (Miscellaneous)
	08:30 - 13:00	4,50	0000.909 (Enclustra AI Study)	001 (Miscellaneous)
	Total/	7,00		
Tu 2.4	13:15 - 18:00	4,75	0000.909 (Enclustra AI Study)	005 (Market research AI and AI on FPGA)
	09:00 - 13:00	4,00	0000.909 (Enclustra AI Study)	001 (Miscellaneous)
	Total/	8,75		
Mo 1.4	13:00 - 18:00	5,00	0000.909 (Enclustra AI Study)	001 (Miscellaneous)
	09:30 - 12:30	3,00	0000.909 (Enclustra AI Study)	001 (Miscellaneous)
	Total/	8,00		

Figure A.5.: Week 5

<b>Fr</b> 12.4	08:30 15:00	6,50	0000.909 (Enclustra AI Study)	003 (Presentation about AI on FPGA)
	Total	6,50		
<b>Th</b> 11.4	13:30 18:00	4,50	0000.909 (Enclustra AI Study)	003 (Presentation about AI on FPGA)
	08:15 13:15	5,00	0000.909 (Enclustra AI Study)	001 (Miscellaneous)
	Total	9,50		
<b>We</b> 10.4	15:15 18:45	3,50	0000.909 (Enclustra AI Study)	001 (Miscellaneous)
	09:30 15:00	5,50	0000.909 (Enclustra AI Study)	003 (Presentation about AI on FPGA)
	Total	9,00		
<b>Tu</b> 9.4	13:00 19:00	6,00	0000.909 (Enclustra AI Study)	002 (Testing DNNDK & revision on ZC104)
	09:00 12:45	3,75	0000.909 (Enclustra AI Study)	002 (Testing DNNDK & revision on ZC104)
	Total	9,75		
<b>Mo</b> 8.4	13:15 17:00	3,75	0000.909 (Enclustra AI Study)	001 (Miscellaneous)
	08:30 13:00	4,50	0000.909 (Enclustra AI Study)	001 (Miscellaneous)
	Total	8,25		

Figure A.6.: Week 6

<b>Fr</b> 19.4	09:00 17:00	8,00	0000.909 (Enclustra AI Study)	001 (Miscellaneous)
	Total	8,00		
<b>Th</b> 18.4	08:00 17:00	9,00	0000.909 (Enclustra AI Study)	001 (Miscellaneous)
	Total	9,00		
<b>We</b> 17.4	09:00 17:00	8,00	0000.909 (Enclustra AI Study)	001 (Miscellaneous)
	Total	8,00		
<b>Tu</b> 16.4	09:00 17:00	8,00	0000.909 (Enclustra AI Study)	003 (Presentation about AI on FPGA)
	Total	8,00		
<b>Mo</b> 15.4	09:00 17:00	8,00	0000.909 (Enclustra AI Study)	003 (Presentation about AI on FPGA)
	Total	8,00		

Figure A.7.: Week 7

<b>Fr</b> 25.4	09:00 17:00	8,00	0000.909 (Enclustra AI Study)	001 (Miscellaneous)	int	MIPI; DPU integration
	Total	8,00				
<b>Th</b> 25.4	09:00 17:00	8,00	0000.909 (Enclustra AI Study)	001 (Miscellaneous)	int	MIPI; DPU integration
	Total	8,00				
<b>We</b> 24.4	09:00 17:00	8,00	0000.909 (Enclustra AI Study)	001 (Miscellaneous)	int	Synthara conference call; MIPI; DPU integration
	Total	8,00				
<b>Tu</b> 23.4	09:00 17:00	8,00	0000.909 (Enclustra AI Study)	001 (Miscellaneous)	int	MIPI CSI interface
	Total	8,00				
<b>Mo</b> 22.4						

Figure A.8.: Week 8

## A. Workday reports

---

Fr 3.5	09:30 - 17:30	8.00	0000.909 (Enclustra AI Study)	002 (Testing DNNDK & revision on ZC104)	int	New DNNDK version; Petalinux
	Total/	8.00				
Th 2.5	09:00 - 17:00	8.00	0000.909 (Enclustra AI Study)	002 (Testing DNNDK & revision on ZC104)	int	New DNNDK version; Petalinux debug for MIPI test
	Total/	8.00				
We 1.5	09:00 - 17:30	8.50	0000.909 (Enclustra AI Study)	001 (Miscellaneous)	int	Intel OpenVINO; ZCU104 Mipi DPU integration Petalinux
	Total/	8.50				
Mo 29.4	09:00 - 17:00	8.00	0000.909 (Enclustra AI Study)	001 (Miscellaneous)	int	DPU integration: Wiki Synthara Update; Market Research; Intel OpenVino
	Total/	8.00				

Figure A.9.: Week 9

Fr 10.5	09:00 - 17:00	8.00	0000.909 (Enclustra AI Study)	006 (AI Demonstrator)	int	DPU integration into XU3 reference design; DPU integration into XU1 reference design
	Total/	8.00				
Th 9.5	09:30 - 17:30	8.00	0000.909 (Enclustra AI Study)	006 (AI Demonstrator)	int	DPU integration into XU3 reference design;
	Total/	8.00				
We 8.5	09:00 - 18:00	9.00	0000.909 (Enclustra AI Study)	006 (AI Demonstrator)	int	DPU integration into XU3 reference design;
	Total/	9.00				
Tu 7.5	09:00 - 17:00	8.00	0000.909 (Enclustra AI Study)	002 (Testing DNNDK & revision on ZC104)	int	New DNNDK version; Petalinux
	Total/	8.00				
Mo 6.5	09:00 - 17:00	8.00	0000.909 (Enclustra AI Study)	002 (Testing DNNDK & revision on ZC104)	int	New DNNDK version; Petalinux
	Total/	8.00				

Figure A.10.: Week 10

Fr 17.5	08:00 - 13:00	5.00	0000.909 (Enclustra AI Study)	006 (AI Demonstrator)	int	Petalinux debug
	Total/	5.00				
Th 16.5	08:00 - 18:00	10.00	0000.909 (Enclustra AI Study)	006 (AI Demonstrator)	int	Synthara visit; Petalinux debug
	Total/	10.00				
We 15.5	08:30 - 16:30	8.00	0000.909 (Enclustra AI Study)	006 (AI Demonstrator)	int	Petalinux kernel debugging
	Total/	8.00				
Tu 14.5	09:00 - 17:30	8.50	0000.909 (Enclustra AI Study)	006 (AI Demonstrator)	int	DPU integration into XU3 reference design; DPU integration into XU1 reference design
	Total/	8.50				
Mo 13.5	15:00 - 18:00	3.00	0000.909 (Enclustra AI Study)	006 (AI Demonstrator)	int	DPU integration into XU3 reference design; DPU integration into XU1 reference design
	09:00 - 13:00	4.00	0000.909 (Enclustra AI Study)	006 (AI Demonstrator)	int	DPU integration into XU3 reference design; DPU integration into XU1 reference design
	Total/	7.00				

Figure A.11.: Week 11

Fr 24.5	08:00 15:30 7.50  0000.909 (Enclustra AI Study)	006 (AI Demonstrator)	int	Enclustra AI demonstrator
Th 23.5	08:00 17:00 9.00  0000.909 (Enclustra AI Study)	006 (AI Demonstrator)	int	implemented resnet50 on ST3
We 22.5	08:00 16:00 8.00  0000.909 (Enclustra AI Study)	006 (AI Demonstrator)	int	Petalinux debug
Tu 21.5	08:00 16:00 8.00  0000.000 (Enclustra Abwesenheit)	001 (Ferien)	int	
Mo 20.5	08:00 16:00 8.00  0000.000 (Enclustra Abwesenheit)	001 (Ferien)	int	

Figure A.12.: Week 12

Fr 31.5	13:00 17:00 4.00  0000.909 (Enclustra AI Study)	006 (AI Demonstrator)	int	work on script; prepare presentation for data collection
Th 30.5	09:00 12:00 3.00  0000.909 (Enclustra AI Study)	006 (AI Demonstrator)	int	work on script; prepare presentation for data collection
Total	7.00			
We 29.5	09:30 18:00 8.50  0000.909 (Enclustra AI Study)	006 (AI Demonstrator)	int	start collecting data set for Synthara collaboration; created script for data collection
Total	8.50			
Tu 28.5	09:30 18:00 8.50  0000.909 (Enclustra AI Study)	006 (AI Demonstrator)	int	package AI demonstrator; work on face detection; work on pose detection
Total	8.50			
Mo 27.5	09:00 17:00 8.00  0000.909 (Enclustra AI Study)	006 (AI Demonstrator)	int	package AI demonstrator; work on face detection; work on pose detection
Total	8.00			

Figure A.13.: Week 13

Fr 7.6	08:00 16:00 8.00  0000.000 (Enclustra Abwesenheit)	003 (Krankheit)	int	Cold
Th 6.6	08:00 16:00 8.00  0000.000 (Enclustra Abwesenheit)	003 (Krankheit)	int	Cold
Total	8.00			
We 5.6	09:30 16:30 7.00  0000.909 (Enclustra AI Study)	006 (AI Demonstrator)	int	data set presentation and collection
Total	7.00			
Tu 4.6	09:30 17:30 8.00  0000.909 (Enclustra AI Study)	006 (AI Demonstrator)	int	prepare presentation for data collection; camera setup preparation
Total	8.00			
Mo 3.6	08:30 17:30 9.00  0000.909 (Enclustra AI Study)	006 (AI Demonstrator)	int	prepare presentation for data collection; 3D model for embedded world
Total	9.00			

Figure A.14.: Week 14

Fr 14.6	10:30 17:00 6.50  0000.909 (Enclustra AI Study)	006 (AI Demonstrator)	int	data collection; frame extraction
Th 13.6	09:30 17:30 8.00  0000.909 (Enclustra AI Study)	006 (AI Demonstrator)	int	data collection; intel seminar wrapup
Total	8.00			
We 12.6	08:00 17:00 9.00  0000.909 (Enclustra AI Study)	001 (Miscellaneous)	int	Intel Lausanne Seminar
Total	9.00			
Tu 11.6	17:00 20:00 3.00  0000.909 (Enclustra AI Study)	006 (AI Demonstrator)	int	Data correction.
08:00 12:00 4.00  0000.000 (Enclustra Abwesenheit)	001 (Ferien)	int		
Total	7.00			

Figure A.15.: Week 16

## A. Workday reports

---

Fr 21.6	08:00 16:00 8.00  0000.000 (Enclustra Abwesenheit)	001 (Ferien)	int	St. Petersburg
Total/ 8.00				
Th 20.6	08:00 16:00 8.00  0000.000 (Enclustra Abwesenheit)	001 (Ferien)	int	St. Petersburg
Total/ 8.00				
We 19.6	08:00 16:00 8.00  0000.000 (Enclustra Abwesenheit)	001 (Ferien)	int	St. Petersburg
Total/ 8.00				
Tu 18.6	08:00 16:00 8.00  0000.000 (Enclustra Abwesenheit)	001 (Ferien)	int	St. Petersburg
Total/ 8.00				
Mo 17.6	08:00 16:00 8.00  0000.000 (Enclustra Abwesenheit)	001 (Ferien)	int	St. Petersburg
Total/ 8.00				

Figure A.16.: Week 17

Day	From	To	Duration	Project	Task	Act	Remark
Su 30.6							
Sa 29.6							
Fr 28.6	10:00 17:00 7.00  0000.909 (Enclustra AI Study)	002 (DNNDK Toolchain)	int	documentation internship			
	Total/ 7.00						
Th 27.6	10:45 18:15 7.50  0000.909 (Enclustra AI Study)	002 (DNNDK Toolchain)	int	documentation internship			
	Total/ 7.50						
We 26.6	15:00 18:00 3.00  0000.909 (Enclustra AI Study)	002 (DNNDK Toolchain)	int	documentation internship			
	10:00 15:00 5.00  0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	placed order of the robotic hand;			
	Total/ 8.00						
Tu 25.6	10:00 18:00 8.00  0000.909 (Enclustra AI Study)	006 (Miscellaneous)	int	Looked into candidates for robotic hands; picked suitable candidate; also checked 3D printed hands; Synthara update and further talks			
	Total/ 8.00						
Mo 24.6	09:30 18:00 8.50  0000.909 (Enclustra AI Study)	006 (Miscellaneous)	int	Looked into candidates for robotic hands; Synthara update and further talks; Meeting with Jelena			
	Total/ 8.50						

Figure A.17.: Week 18

Day	From	To	Duration	Project	Task	Act	Remark
Su 7.7							
Sa 6.7							
Fr 5.7	14:30 17:00 2.50  0000.909 (Enclustra AI Study)	002 (DNNDK Toolchain)	int	tensorflow workflow			
	14:00 14:30 0.50  0000.909 (Enclustra AI Study)	001 (Meetings)	int	robot hand FPGA packaging for demonstrator with Patrick			
	09:00 14:00 5.00  0000.909 (Enclustra AI Study)	002 (DNNDK Toolchain)	int	tensorflow workflow			
	Total/ 8.00						
Th 4.7	14:00 17:30 3.50  0000.909 (Enclustra AI Study)	002 (DNNDK Toolchain)	int	tensorflow workflow			
	09:00 13:00 4.00  0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	internship documentation			
	Total/ 7.50						
We 3.7	08:30 15:30 7.00  0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	internship documentation			
	Total/ 7.00						
Tu 2.7	08:30 17:30 9.00  0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	robot hand order processed; synthara checkup; internship documentation			
	Total/ 9.00						
Mo 1.7	13:30 18:00 4.50  0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	website content; synthara checkup;			
	08:00 12:00 4.00  0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	documentation internship			
	Total/ 8.50						

Figure A.18.: Week 19

Day	From	To	Duration	Project	Task	Act	Remark
SU 14.7							
Sa 13.7							
Fr 12.7	⌚⌚ 08:30	17:00	8.50	⌚⌚ 0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	Toledo demonstration; documentation; ECC presentation
		Total	8.50				
Th 11.7	⌚⌚ 09:00	17:00	8.00	⌚⌚ 0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	Synthara check up; robot hand assembly; started presentation for ECC 2019
		Total	8.00				
We 10.7	⌚⌚ 09:00	17:00	8.00	⌚⌚ 0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	preparation of data set for publication; kaggle publication; wiki updates
		Total	8.00				
Tu 09.7	⌚⌚ 12:30	17:30	5.00	⌚⌚ 0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	preparation of data set for publication
	⌚⌚ 08:30	12:30	4.00	⌚⌚ 0000.909 (Enclustra AI Study)	002 (DNNDK Toolchain)	int	documentation tensorflow CIFAR10;
		Total	9.00				
Mo 08.7	⌚⌚ 15:00	18:15	3.25	⌚⌚ 0000.909 (Enclustra AI Study)	002 (DNNDK Toolchain)	int	documentation
	⌚⌚ 09:30	15:00	5.50	⌚⌚ 0000.909 (Enclustra AI Study)	002 (DNNDK Toolchain)	int	tensorflow workflow; tensorflow training; tensorflow documentation
		Total	8.75				

Figure A.19.: Week 20

Day	From	To	Duration	Project	Task	Act	Remark
SU 21.7							
Sa 20.7							
Fr 19.7	⌚⌚ 09:15	17:15	8.00	⌚⌚ 0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	Robot hand PWM control VHDL coding
		Total	8.00				
Th 18.7	⌚⌚ 13:00	17:30	4.50	⌚⌚ 0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	Robot hand documentation translation; Synthara followup Email; FPGA control servo motors
	⌚⌚ 09:30	13:00	3.50	⌚⌚ 0000.909 (Enclustra AI Study)	006 (Miscellaneous)	int	Xilinx Scout testing
		Total	8.00				
We 17.7	⌚⌚ 12:00	17:00	5.00	⌚⌚ 0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	robot hand testing; documentation; FPGA control of servo motors
	⌚⌚ 08:30	12:00	3.50	⌚⌚ 0000.909 (Enclustra AI Study)	006 (Miscellaneous)	int	Mettler Toledo visit
		Total	8.50				
Tu 16.7	⌚⌚ 10:00	17:00	7.00	⌚⌚ 0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	robot hand test; robot hand documentation; internship documentation
		Total	7.00				
Mo 15.7	⌚⌚ 09:30	18:00	8.50	⌚⌚ 0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	robot hand documentation; internship documentation
		Total	8.50				

Figure A.20.: Week 21

## A. Workday reports

---

Day	From	To	Duration	Project	Task	Act
Su 28.7						
Sa 27.7						
Fr 26.7	08:00	16:00	8.00	0000.000 (Enclustra Abwesenheit)	001 (Ferien)	int
	Total		8,00			
Th 25.7	08:00	16:00	8.00	0000.000 (Enclustra Abwesenheit)	001 (Ferien)	int
	Total		8,00			
We 24.7	08:00	16:00	8.00	0000.000 (Enclustra Abwesenheit)	001 (Ferien)	int
	Total		8,00			
Tu 23.7	08:00	16:00	8.00	0000.000 (Enclustra Abwesenheit)	001 (Ferien)	int
	Total		8,00			
Mo 22.7	08:00	16:00	8.00	0000.000 (Enclustra Abwesenheit)	001 (Ferien)	int
	Total		8,00			

Figure A.21.: Week 22

Day	From	To	Duration	Project	Task	Act	Remark
Su 4.8							
Sa 3.8							
Fr 2.8							
Th 1.8							
We 31.7	09:30	17:30	8.00	0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	Synthara design porting; Synthara design; Scout trial
	Total		8,00				
Tu 30.7	16:00	18:45	2.75	0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	Synthara design checkup
	10:30	14:15	3.75	0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	PWM control debugging and testing
	10:00	10:30	0.50	0000.909 (Enclustra AI Study)	001 (Meetings)	int	Meeting with Martin
	09:00	10:00	1.00	0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	Robot hand PWM control VHDL coding;
	Total		8,00				
Mo 29.7	09:00	17:30	8.50	0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	Robot hand PWM control VHDL coding; internship documentation
	Total		8,50				

Figure A.22.: Week 23

Day	From	To	Duration	Project	Task	Act	Remark
Su 11.8							
Sa 10.8							
Fr 9.8	08:15	16:15	8.00	0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	Porting Synthara Design: problems with fitting the design; implementation on XUI+ as backup; check USB 3.0 camera; MIPI ST3 demonstrator
	Total		8,00				
Th 8.8	08:30	17:00	8.50	0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	Porting Synthara Design: received all necessary files from Synthara; problems with fitting the design; implementation on XUI+ as backup; check USB 3.0 camera;
	Total		8,50				
We 7.8	09:30	18:00	8.50	0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	Porting Synthara Design: received all necessary files from Synthara; problems with including and using them in Vivado
	Total		8,50				
Tu 6.8	09:00	17:00	8.00	0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	Porting Synthara Design: received all necessary files from Synthara; problems with including and using them in Vivado
	Total		8,00				
Mo 5.8	09:00	18:00	9.00	0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	Scout Early Access porting of ST3 demonstrator
	Total		9,00				

Figure A.23.: Week 24

Day	From	To	Duration	Project	Task	Act	Remark
SU 18.8							
Sa 17.8							
Fr 16.8	09:00	17:00	8.00	0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	MIPI ST3 demonstrator Petalinux setup; Hardware design Vivado
			Total	8.00			
Th 15.8	09:00	17:00	8.00	0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	MIPI ST3 demonstrator Petalinux setup; Hardware design Vivado
			Total	8.00			
We 14.8	09:00	17:00	8.00	0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	MIPI ST3 demonstrator Petalinux setup; Hardware design Vivado
			Total	8.00			
Tu 13.8	09:00	17:00	8.00	0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	MIPI ST3 demonstrator Petalinux setup; Hardware design Vivado
			Total	8.00			
Mo 12.8	10:00	18:00	8.00	0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	MIPI ST3 demonstrator Petalinux setup; Hardware design Vivado
			Total	8.00			

Figure A.24.: Week 25

Day	From	To	Duration	Project	Task	Act	Remark
SU 25.8							
Sa 24.8							
Fr 23.8	08:30	16:00	7.50	0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	ECC presentation; Petalinux Debug
			Total	7.50			
Th 22.8	09:00	15:45	6.75	0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	ECC presentation; Petalinux Debug
			Total	6.75			
We 21.8	09:00	16:30	7.50	0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	MIPI ST3 demonstrator Petalinux setup; ECC presentation; Petalinux Debug
			Total	7.50			
Tu 20.8	08:30	16:30	8.00	0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	MIPI ST3 demonstrator Petalinux setup; ECC presentation; Petalinux Debug
			Total	8.00			
Mo 19.8	08:30	16:30	8.00	0000.909 (Enclustra AI Study)	005 (Embedded World 2020 Demo)	int	MIPI ST3 demonstrator Petalinux setup; Hardware design Vivado
			Total	8.00			

Figure A.25.: Week 26