Chair of Computer Architecture and Parallel Systems
School of Computation, Information and Technology
Technical University of Munich

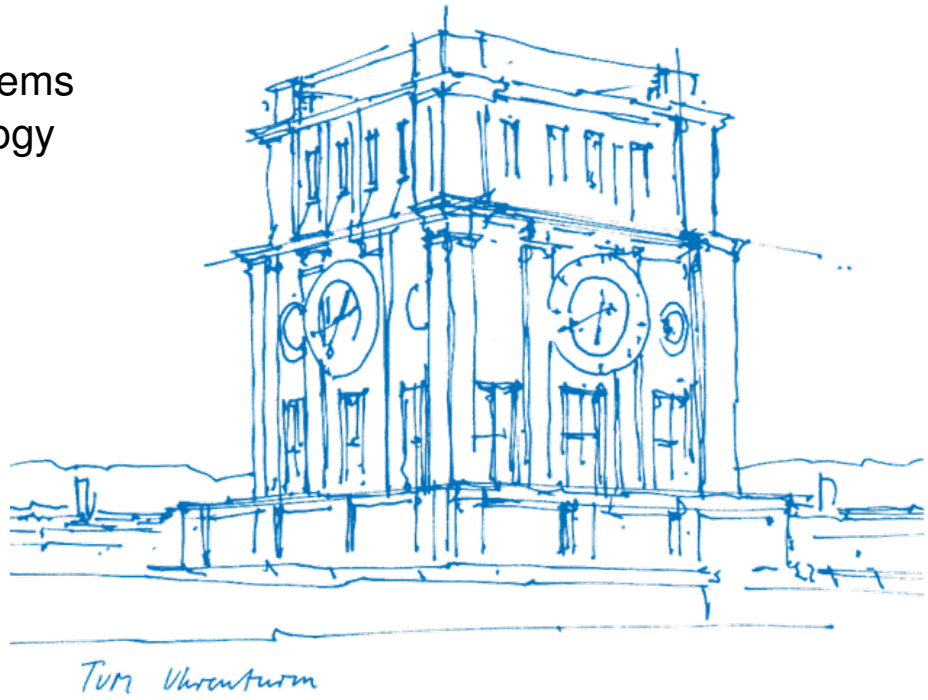
Accelerating CNNs using Programmable Logic

Preliminary Meeting

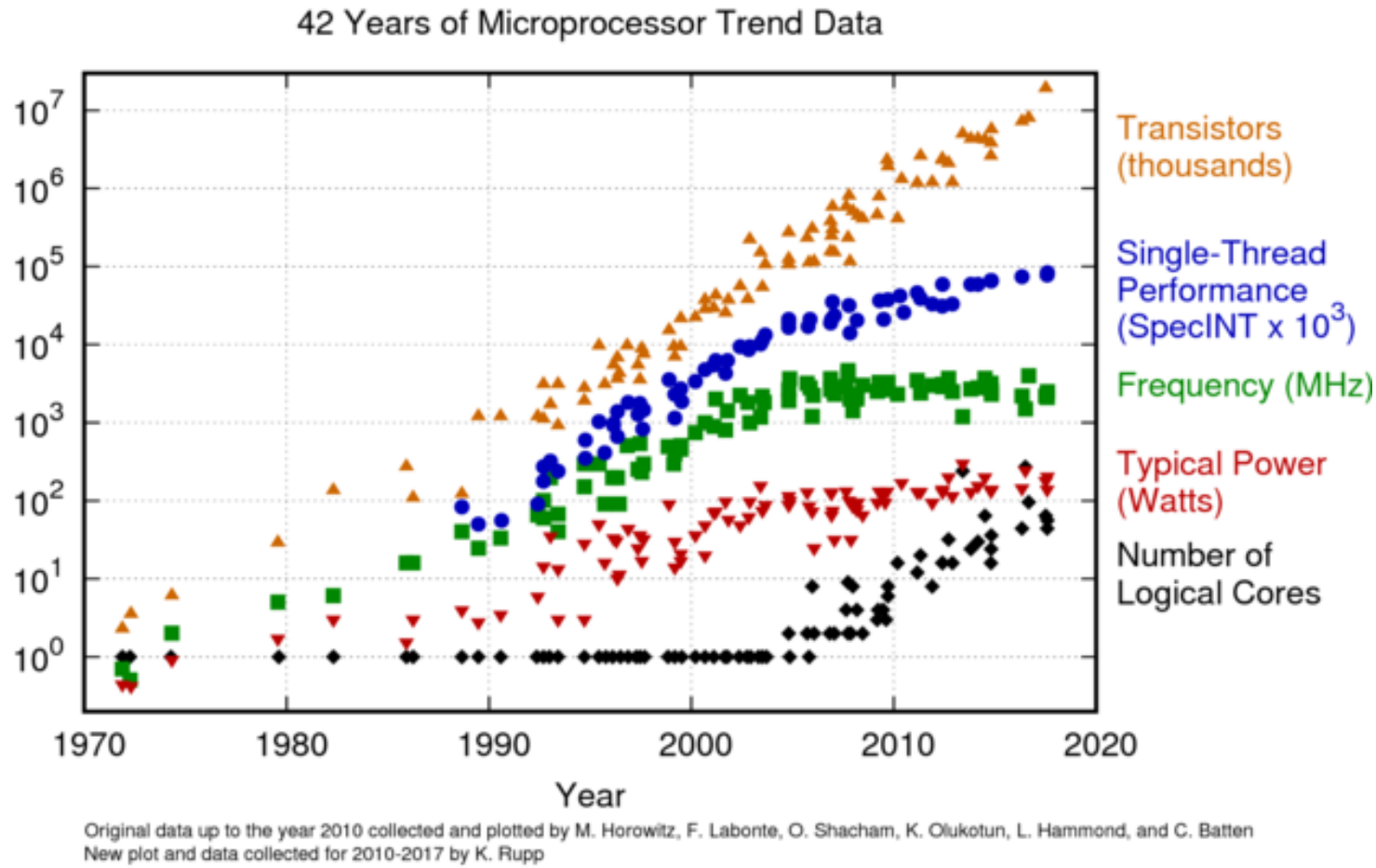
Dirk Stober

Chair of Computer Architecture and Parallel Systems
School of Computation, Information and Technology
Technical University of Munich

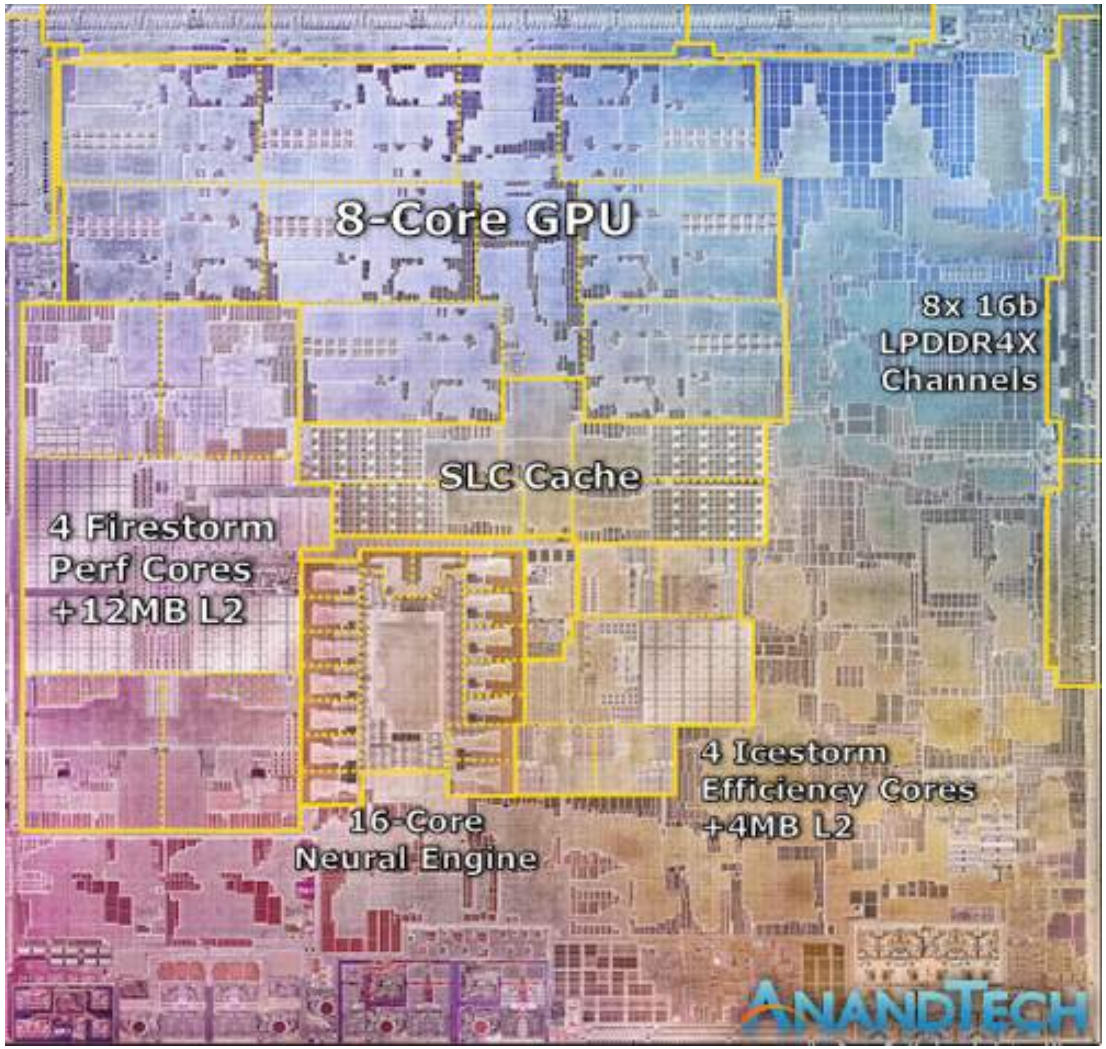
08.02.2023



Domain-Specific Architectures (DSA)



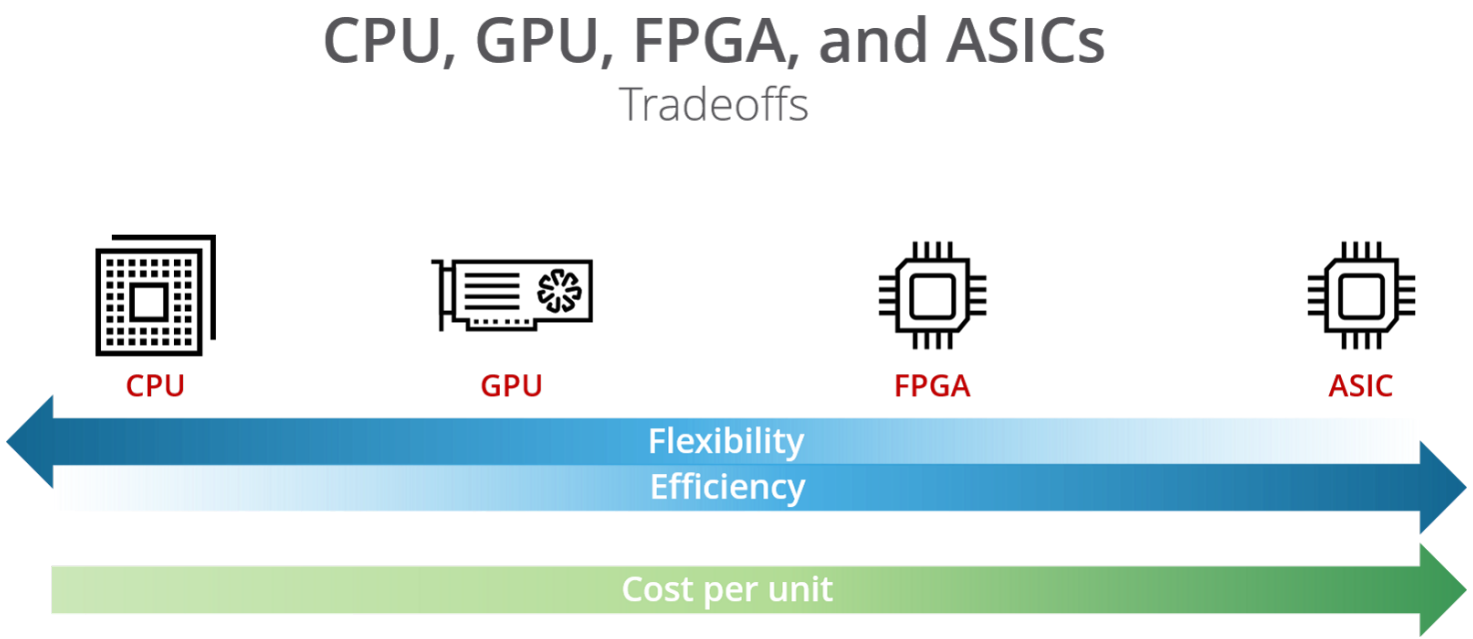
Apple M1



[1]

[1] https://www.anandtech.com/show/16252/mac-mini-apple-m1-tested

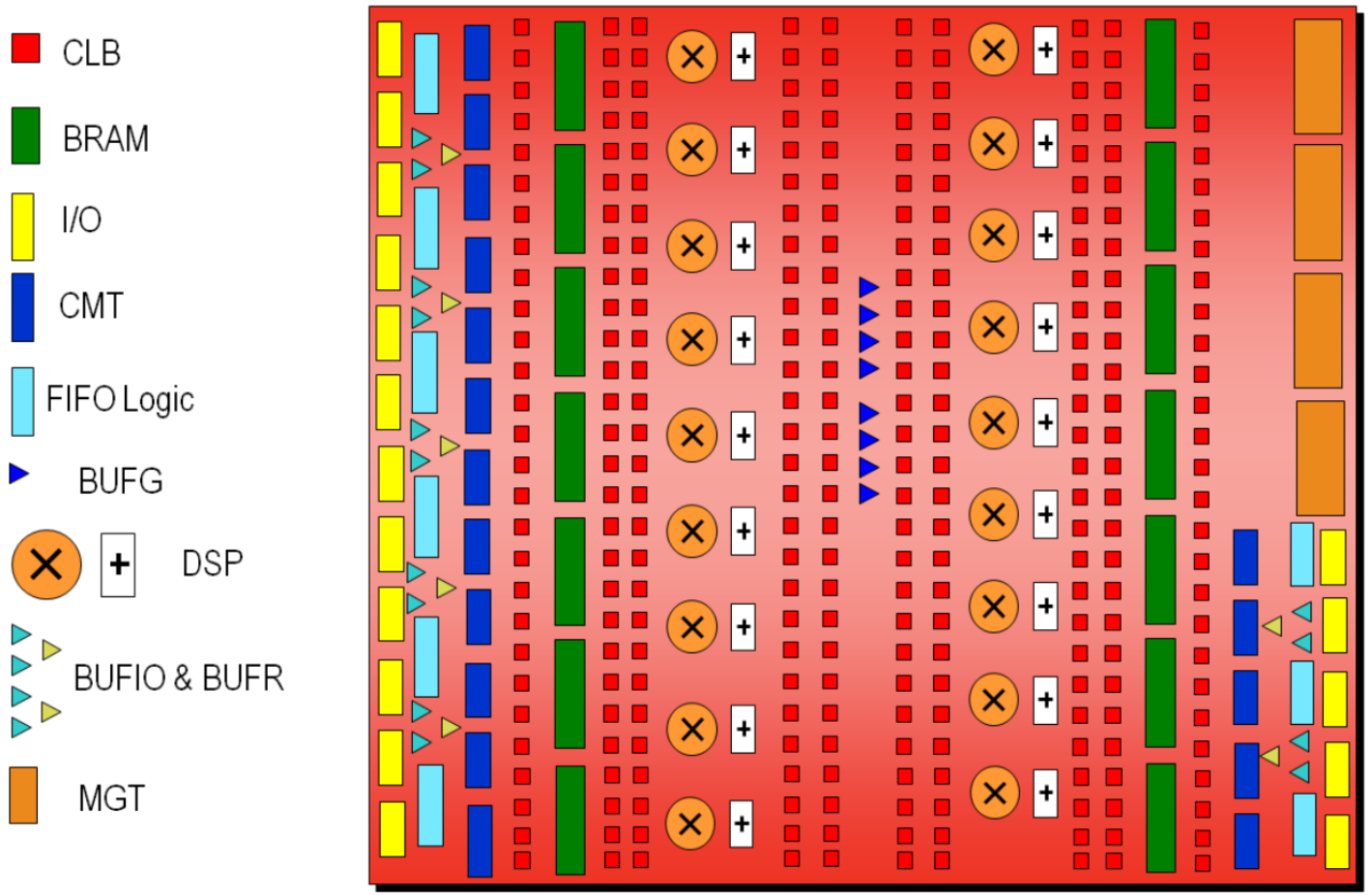
FPGA



FPGA

- Reconfigurable Hardware with lower clock speed
- Flexible accelerator, good for small volume
- Can be used to prototype ASIC design

Artix-7 Overview



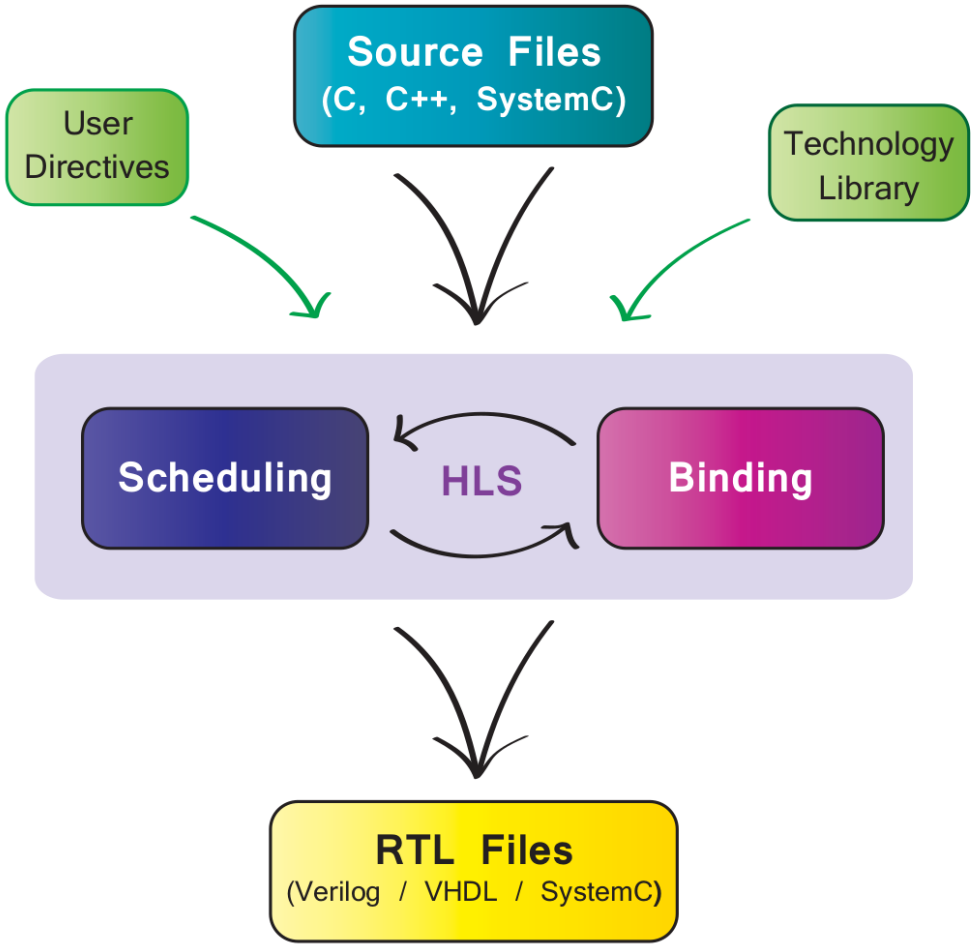
[2]

[2] https://xilinx.eetrend.com/files-eetrend-xilinx/forum/201509/9204-20390-7_series_architecture_overview.pdf

Register Transfer Level (RTL)

```
module bit_reverse
#(
    parameter addr_width = 10
)
(
    input [addr_width - 1 : 0] addr_in,
    output [addr_width - 1 : 0] addr_out
);
generate
    genvar i;
    for(i = 0; i < addr_width;i = i + 1) begin
        assign addr_out[i] = addr_in[addr_width - 1 - i];
    end
endgenerate
endmodule
```

High-Level Synthesis (HLS)



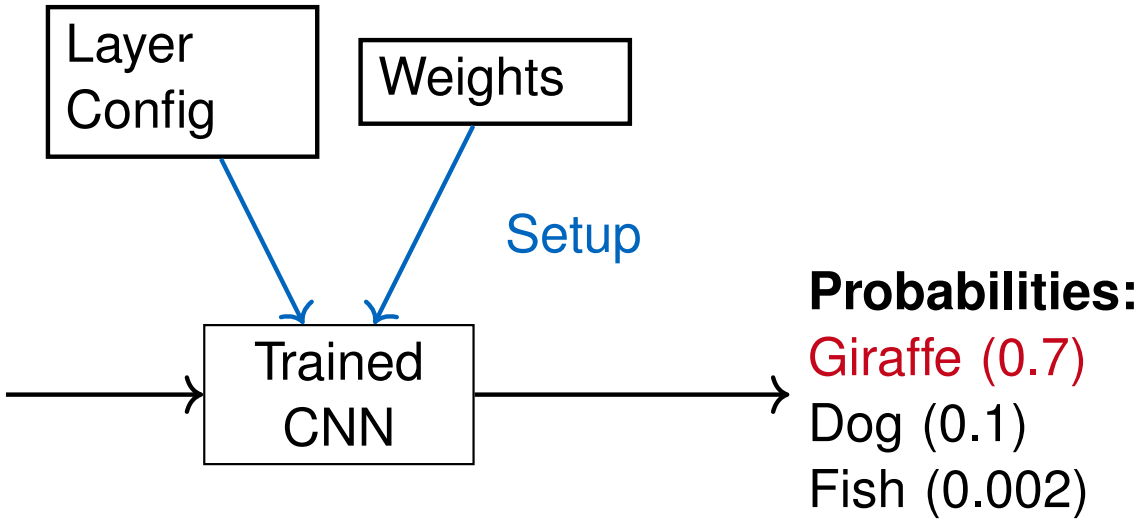
[3]

[3] Crockett et al., *The Zynq Book*

High-Level Synthesis (HLS)

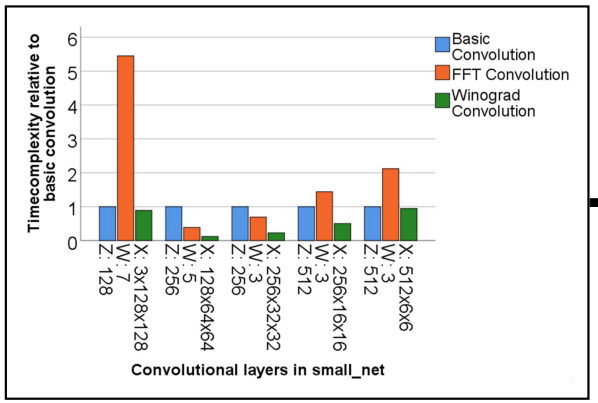
```
void dut(int in[N], int out[N], int n)
{
  hls_thread_local hls::split::round_robin<int, NP> split1;
  hls_thread_local hls::merge::round_robin<int, NP> merge1;
  #pragma HLS dataflow
  read_in(in, n, split1.in);
  // Task-Channels
  hls_thread_local hls::task t[NP];
  for (int i = 0; i < NP; i++) {
  #pragma HLS unroll
      t[i](worker, split1.out[i], merge1.in[i]);
  }
  write_out(merge1.out, out, n);
}
```

Inference

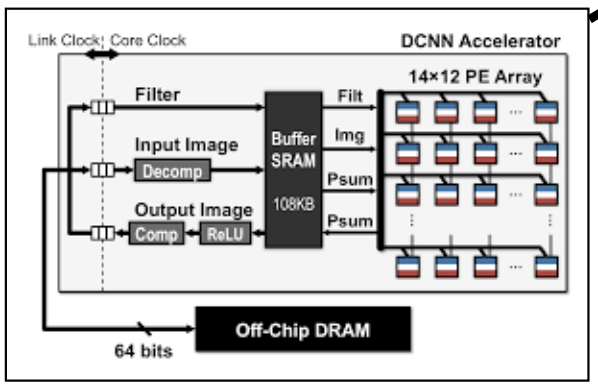


- Focus inference of CNNs
- Learn required kernels and implement them in C++ and analyse algorithm
- Higher Latency and Energy requirements than training (e.g. Smartphone)
- Reduce Latency and/or increase throughput using HLS + FPGA

Project



Analysis of SW



Existing Architectures

Student Synthesis

```
#define N = 10
#define M = 3
#define ZM = N - M + 1
void conv1d(float x[N], float w[M], float z[ZM])
{
        #pragma HLS interface m_axi port=x,w
        #pragma HLS interface m_axi port=z
        #pragma HLS INTERFACE s_axilite port=return
        for(int i =0 ; i < ZM; i++){
                float acc = 0;
                for(int j = 0; j < M; j++){
                        acc += x[i + j]*w[j];
                }
                z[i] = acc;
        };
}
```

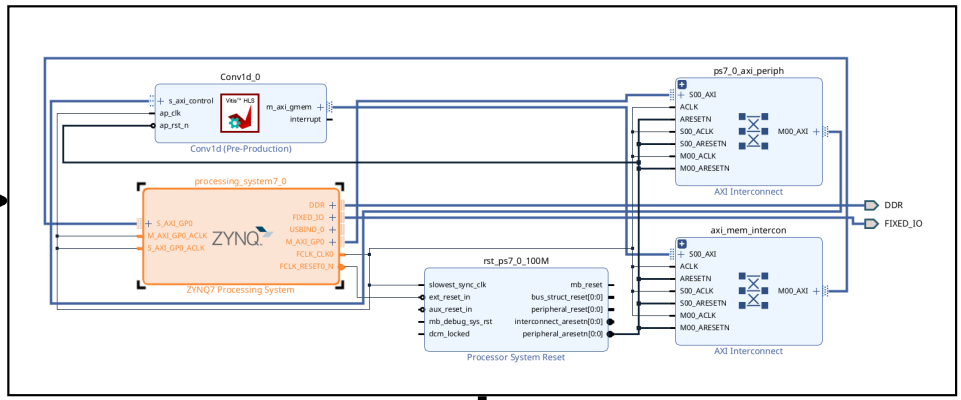
HLS Design

Integration

```
#define N = 10
#define M = 3
#define ZM = N - M + 1
void conv1d(float x[N], float w[M], float z[ZM])
{
        #pragma HLS interface m_axi port=x,w
        #pragma HLS interface m_axi port=z
        #pragma HLS INTERFACE s_axilite port=return
        for(int i =0 ; i < ZM; i++){
                float acc = 0;
                for(int j = 0; j < M; j++){
                        acc += x[i + j]*w[j];
                }
                z[i] = acc;
        };
}
```

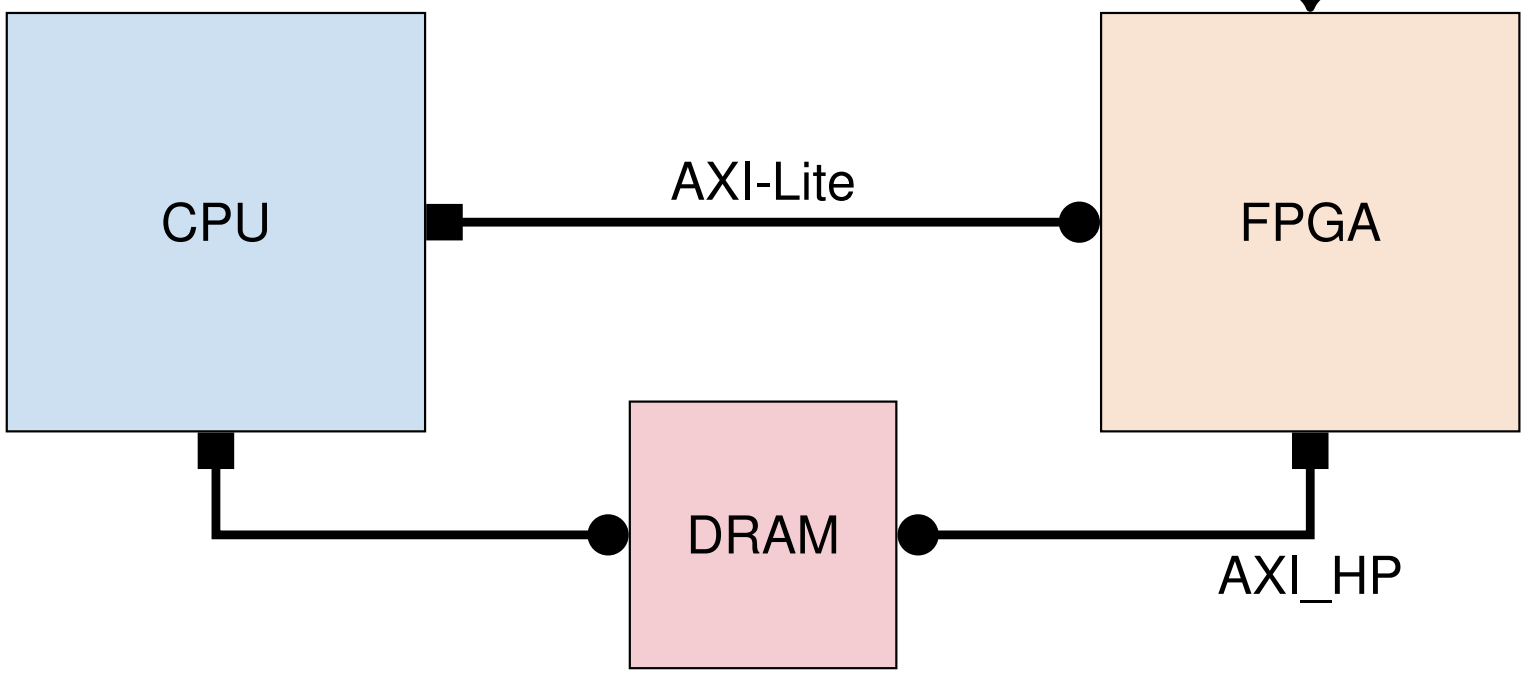
HLS Design

Export IP



Vivado BD

Flash bitstream



Structure (1/2)

1. **Domain Specific Architectures**
   - **Why do we need Domain Specific Architectures?**
   - **Existing DSAs and design approaches**

Structure (1/2)

1. Domain Specific Architectures
   - Why do we need Domain Specific Architectures?
   - Existing DSAs and design approaches
2. **FPGAs & Digital Design Basics (Verilog)**
   - **Components of FPGAs (BRAM,LUTs,DSPs and more)**
   - **Recap on Sequential, Combinational Logic, FSMs and Pipelining**

Structure (1/2)

1. Domain Specific Architectures
   - Why do we need Domain Specific Architectures?
   - Existing DSAs and design approaches
2. FPGAs & Digital Design Basics (Verilog)
   - Components of FPGAs (BRAM,LUTs,DSPs and more)
   - Recap on Sequential, Combinational Logic, FSMs and Pipelining
3. **High-Level Synthesis**
   - **Different Pragmas (Optimizations)**
   - **Compilation workflow and Debugging**

Structure (1/2)

1. Domain Specific Architectures
   - Why do we need Domain Specific Architectures?
   - Existing DSAs and design approaches
2. FPGAs & Digital Design Basics (Verilog)
   - Components of FPGAs (BRAM,LUTs,DSPs and more)
   - Recap on Sequential, Combinational Logic, FSMs and Pipelining
3. High-Level Synthesis
   - Different Pragmas (Optimizations)
   - Compilation workflow and Debugging
4. **Use Case: CNN-Inference & C/C++ recap**
   - **Kernels of CNNs (Conv,ReLU, Linear, etc.)**
   - **Implementation in C++ and performance analysis of different layers**
   - **Write a short report per group about the SW implementation**

Structure (2/2)

5. **Paper Presentation**
    - **Per group presentation of an AI-Accelerator**

Structure (2/2)

5. Paper Presentation
   - Per group presentation of an AI-Accelerator
6. **High-Level Synthesis Project**
   - **Accelerate the CNN-Inference using the PYNQ-Z2 board**
   - **Lectures will deal with more advanced HLS topics**

Structure (2/2)

5. Paper Presentation
   - Per group presentation of an AI-Accelerator
6. High-Level Synthesis Project
   - Accelerate the CNN-Inference using the PYNQ-Z2 board
   - Lectures will deal with more advanced HLS topics
7. **Grading**
   - **Final Report and Implementation (50%)**
   - **Presentation (10%)**
   - **Individual Project Discussion (40%)**

Structure (2/2)

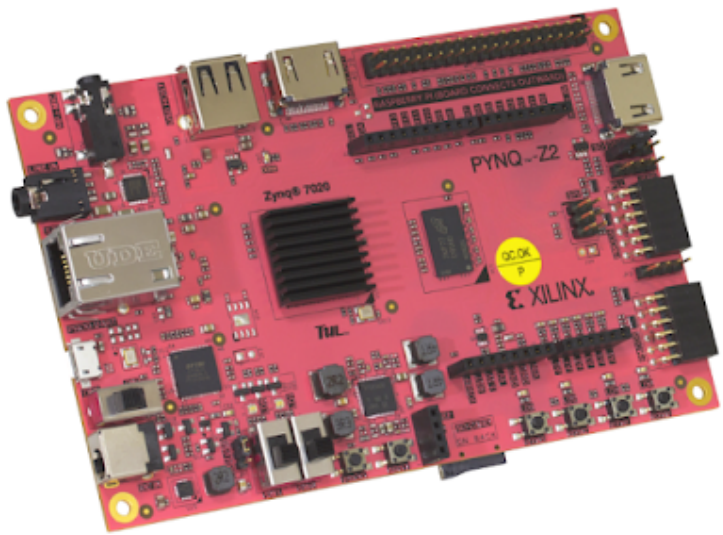
5. Paper Presentation
   - Per group presentation of an AI-Accelerator
6. High-Level Synthesis Project
   - Accelerate the CNN-Inference using the PYNQ-Z2 board
   - Lectures will deal with more advanced HLS topics
7. Grading
   - Final Report and Implementation (50%)
   - Presentation (10%)
   - Individual Project Discussion (40%)
8. **Groups:**
   - **Up to 3 students per group**
   - **Less students are possible**
   - **8 boards available so 8 groups max**

Course Structure

- Contact: dirk.stober@tum.de
- Lecture session Tuesday (10:00 - 12:00)
  - In room 01.06.020
- Lab session Friday (14:00 - 16:00)
  - In room 01.06.020
  - **Will have some Lectures/Presentations on Friday as well!**
- Additional Pre-recorded Videos

Equipment & Software

- Lab exercises will be designed for Linux machines
- Windows and Apple (x86) will also work
- Apple ARM laptops (M1-M3) will have to use a server for Vivado & Vitis (Limited Supply)
- Python interpreter + Pytorch , C++ compiler
- Vivado & Vitis (Free Student License)
- Board: Pynq Z2 accessible through a server



Summary Deliverables

- Verilog and HLS labs
- SW-Implementation (Mandatory)
  - C++ Implementation
  - SW Report
- Paper Presentation (Mandatory)
- Graded Project: HLS implementation of an CNN accelerator
  - Presentation (20 min)
  - Report
  - Discussion (Week of presentation or week afterwards)

Use this course in the machting System:

Practical: Accelerating Convolutional Neural Networks using Programmable Logic (IN0012) (IN2106, IN4345)