

# Computer Organization and Networks

(INB.06000UF, INB.07001UF)

## Welcome

Winter 2021/2022



Stefan Mangard, [www.iaik.tugraz.at](http://www.iaik.tugraz.at)

# COVID-19 Prolog

# Content

# What this course is about

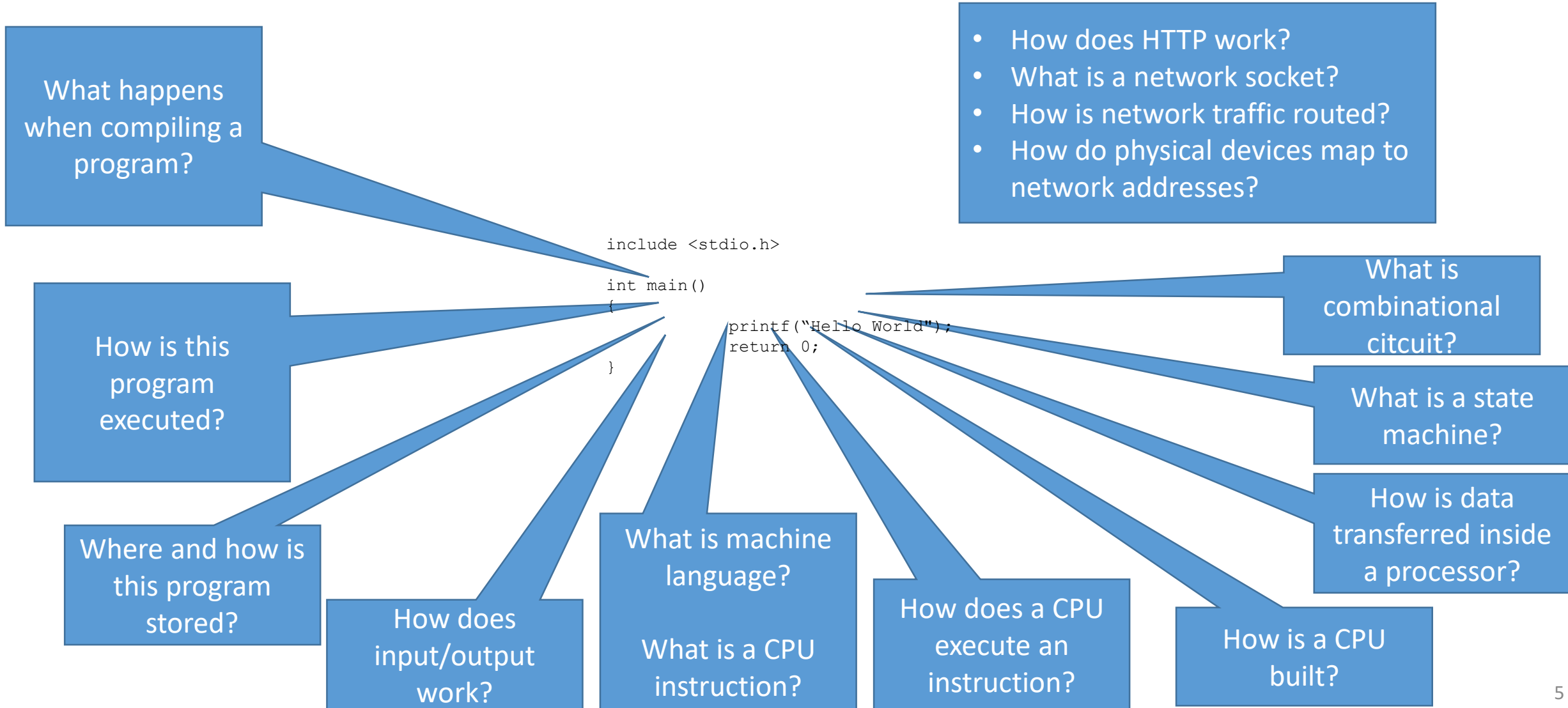
- How does a computer work?
- What does actually happen, if I compile and run this code?

```
include <stdio.h>

int main()
{
    printf("Hello World");
    return 0;
}
```

- How do computers communicate?

# Hardware and Software - It's all one Thing

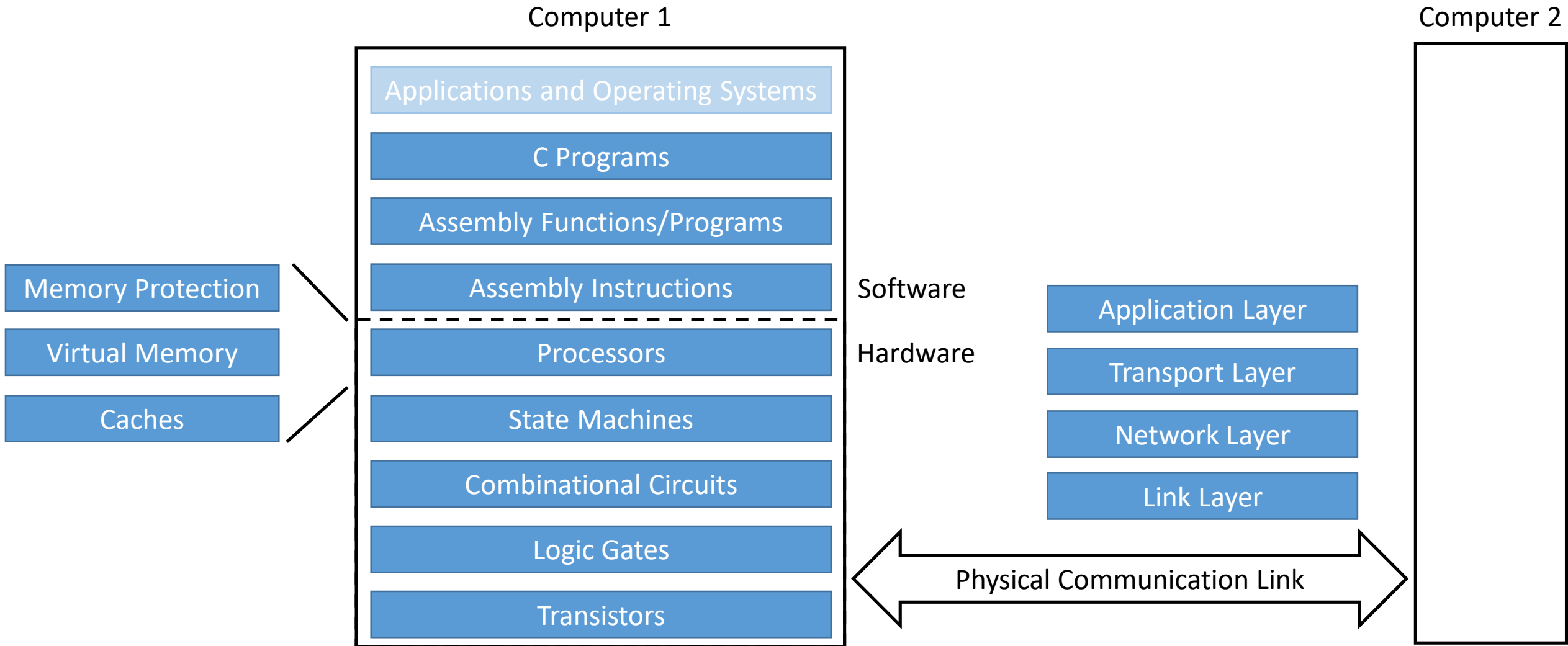


# The Lecture follows a Bottom-up Approach

- Abstraction will be our most important tool
- We “play Lego” and we constantly build larger and more powerful bricks

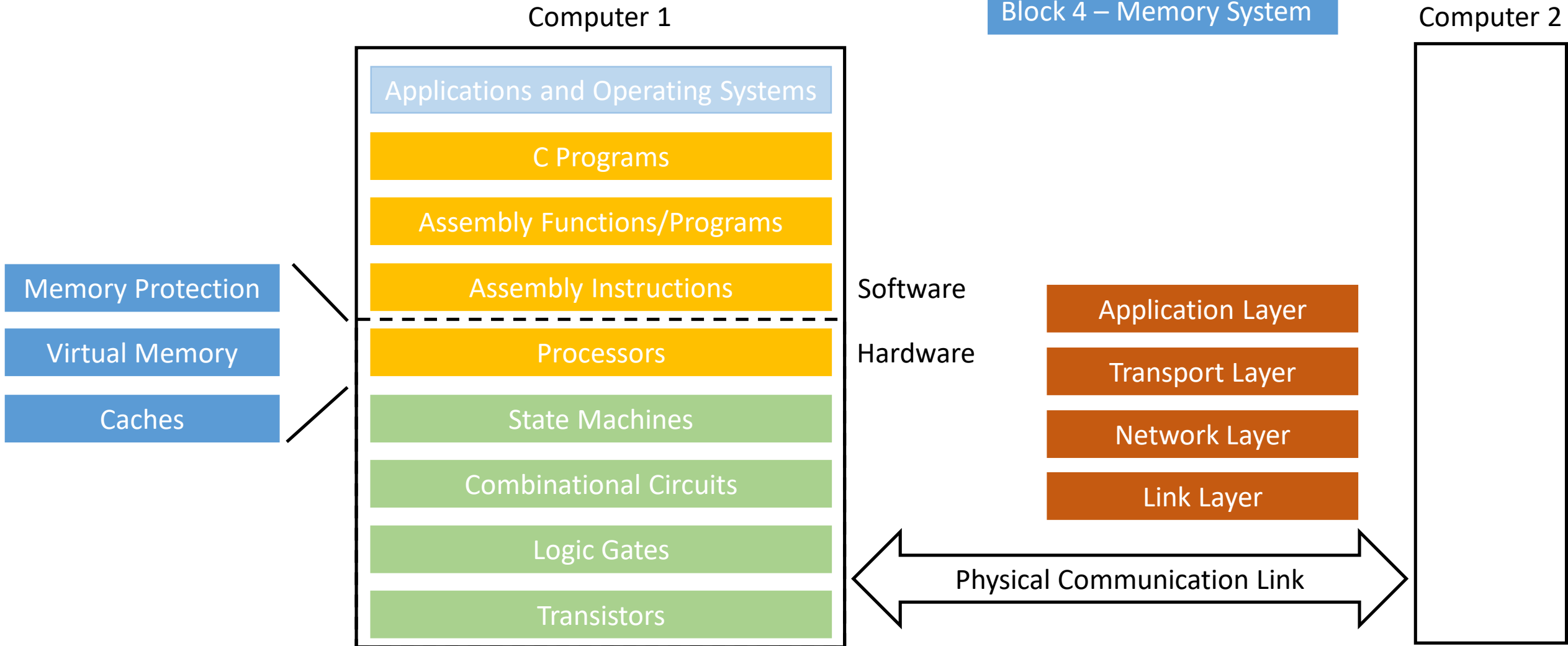


# The Big Picture



# The Big Picture

- Block 1 - Basics
- Block 2 - Processors
- Block 3 - Networks
- Block 4 - Memory System





# Goal

- Get to know the machine you program → only this allows to write highly optimized code
- Understand the specifications of your device
  - Which device does this spec belong to?
    - 64-bit six-core CPU implementing ARMv8 ISA
    - Two high performance cores @3.1 GHz, four energy efficient cores
    - Caches of performance cores: 192 KiB L1I and 128 KiB L1D; Shared L2 with 8 MiB
    - TSMC 5nm, 11.8 billion transistors

# ACM Turing Awards

- The Turing Award is the most prestigious award in computer science – it is the Noble Price of Computer Science
- David A. Patterson and John L. Hennessy received the Turing Award 2017 for their work on computer architectures and organization

**Watch their Turing Lecture:**

**<https://www.acm.org/hennessy-patterson-turing-lecture>**



# Computer Organization and Networks

- In this course, we learn the basics to get the **big picture** → dig deeper in follow-up courses!

Networks

Application Layer

Transport Layer

Network Layer

Link Layer

Software

Applications and Operating Systems

C Programs

Assembly Functions/Programs

Assembly Instructions

Hardware

Processors

State Machines

Combinational Circuits

Logic Gates

Transistors

# More?

- In this course, we learn the basics to get the big picture → dig deeper in follow-up courses!

- **System-Level Programming**
- **Operating System**

“Build your own OS”

Networks

Application Layer

Transport Layer

Network Layer

Link Layer

Applications and Operating Systems

C Programs

Assembly Functions/Programs

Assembly Instructions

Processors

State Machines

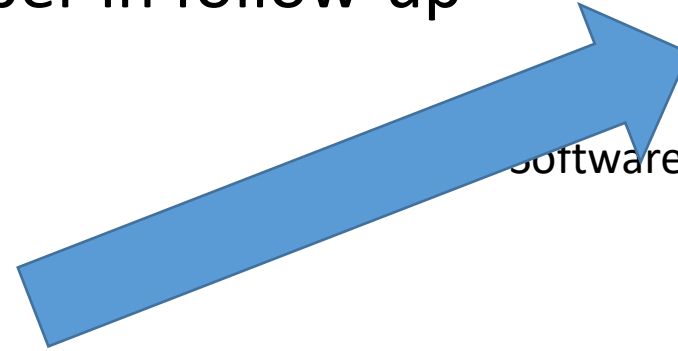
Combinational Circuits

Logic Gates

Transistors

Software

Hardware



# More?

- In this course, we learn the basics to get the big picture → dig deeper in follow-up courses!

- **Digital System Design**

“Build your own hardware”



<https://opentitan.org/>

Networks

Application Layer

Transport Layer

Network Layer

Link Layer

Software

Applications and Operating Systems

C Programs

Assembly Functions/Programs

Assembly Instructions

Hardware

Processors

State Machines

Combinational Circuits

Logic Gates

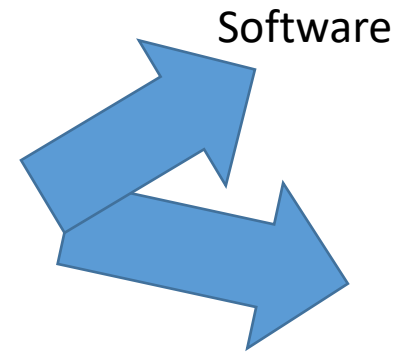
Transistors

# More?

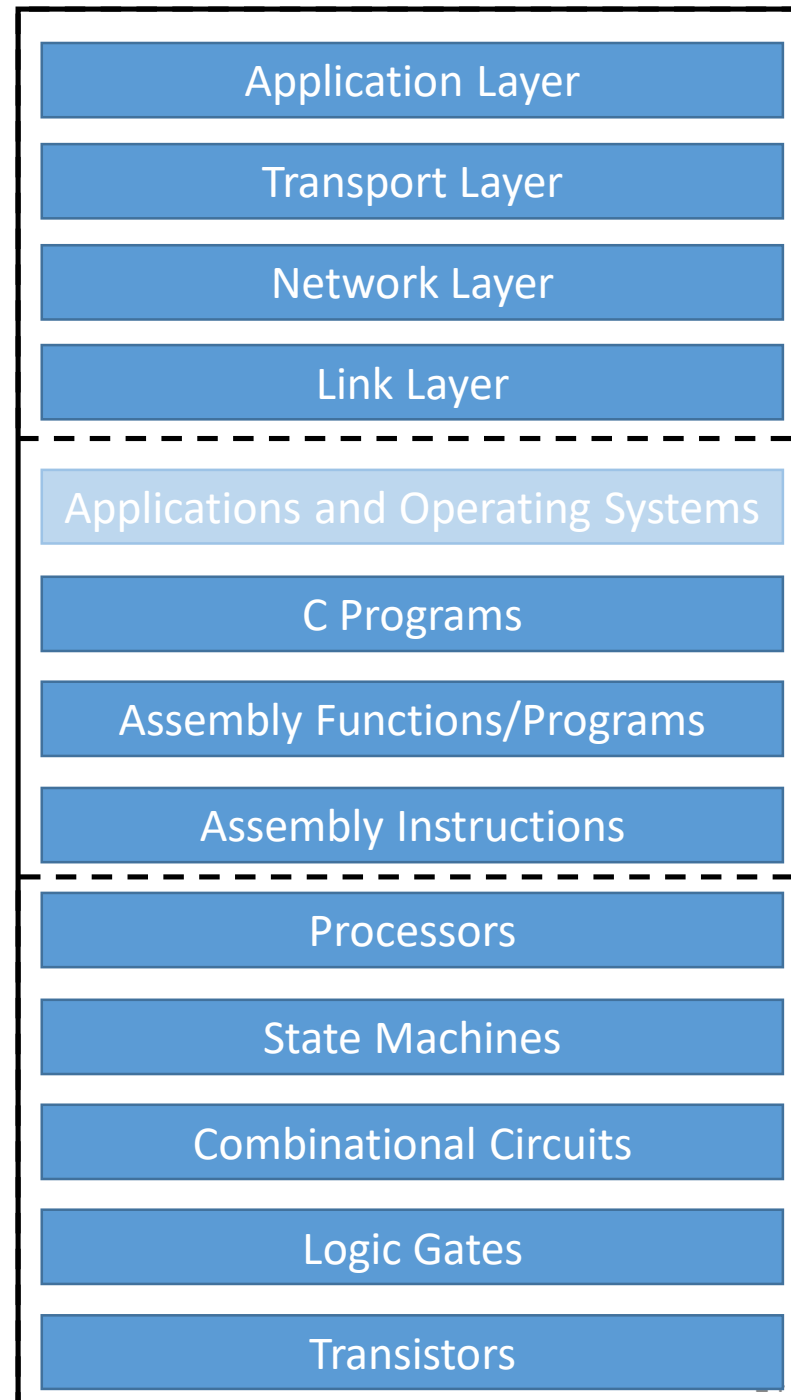
- In this course, we learn the basics to get the big picture → dig deeper in follow-up courses!

- **System Integration and Programming**

“Build your own hardware and integrate it in Linux”



Networks



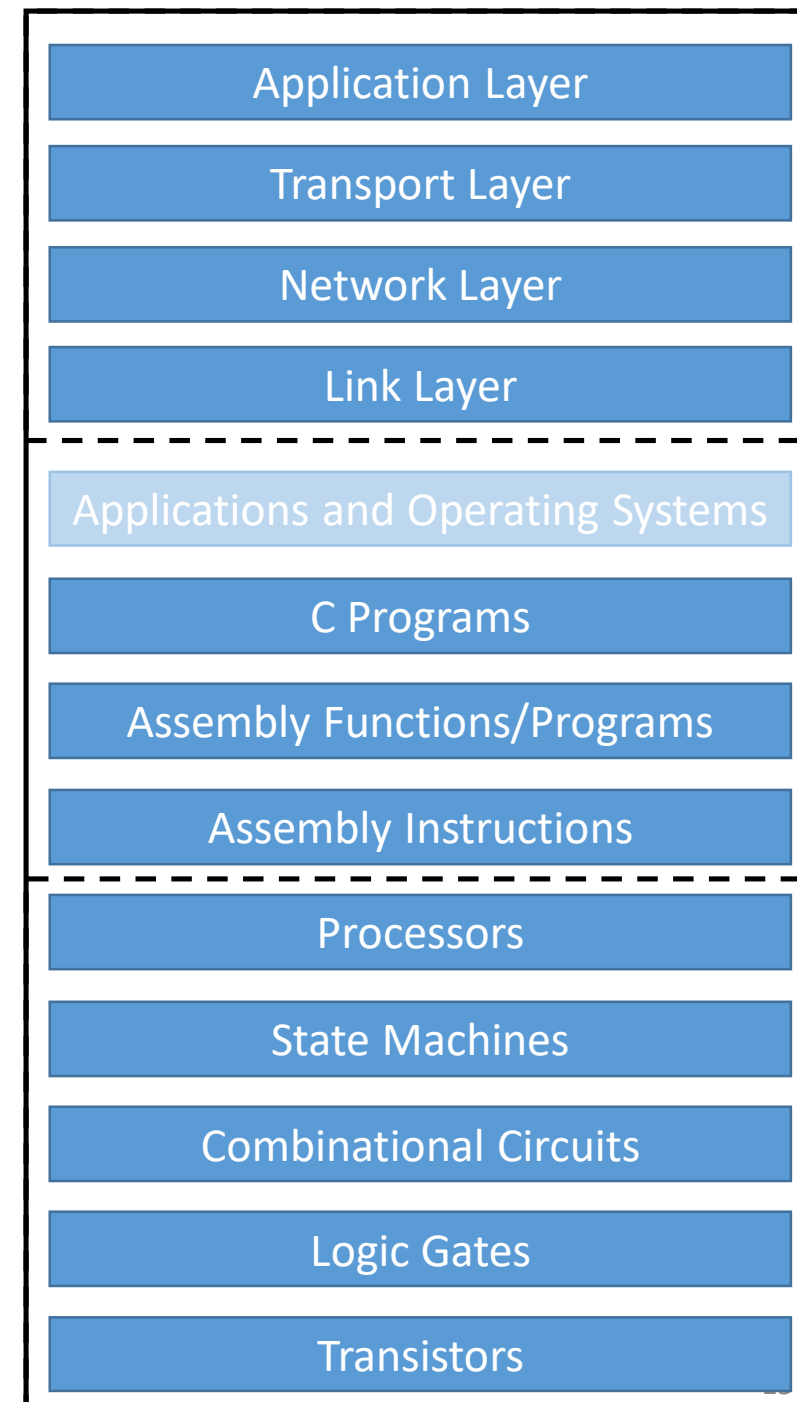
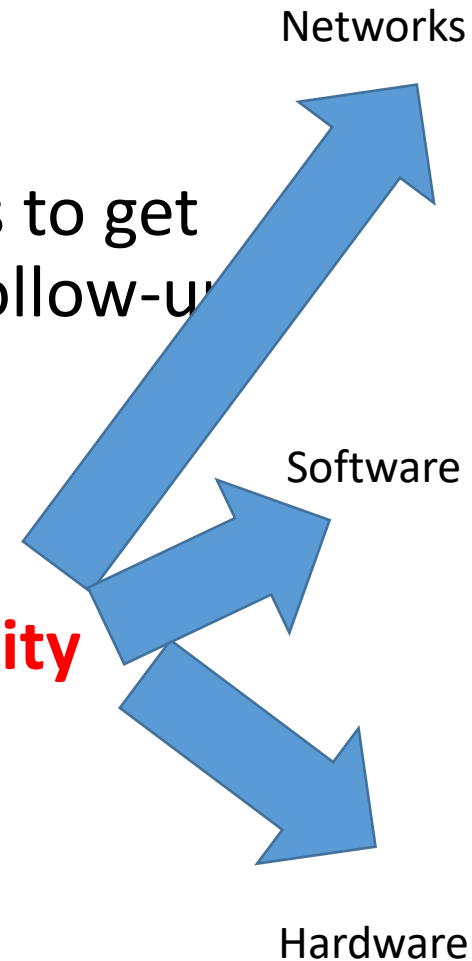
Hardware

# More?

- In this course, we learn the basics to get the big picture → dig deeper in follow-up courses!

- **Introduction to Information Security**

“Learn about Security on all Layers”



# More?

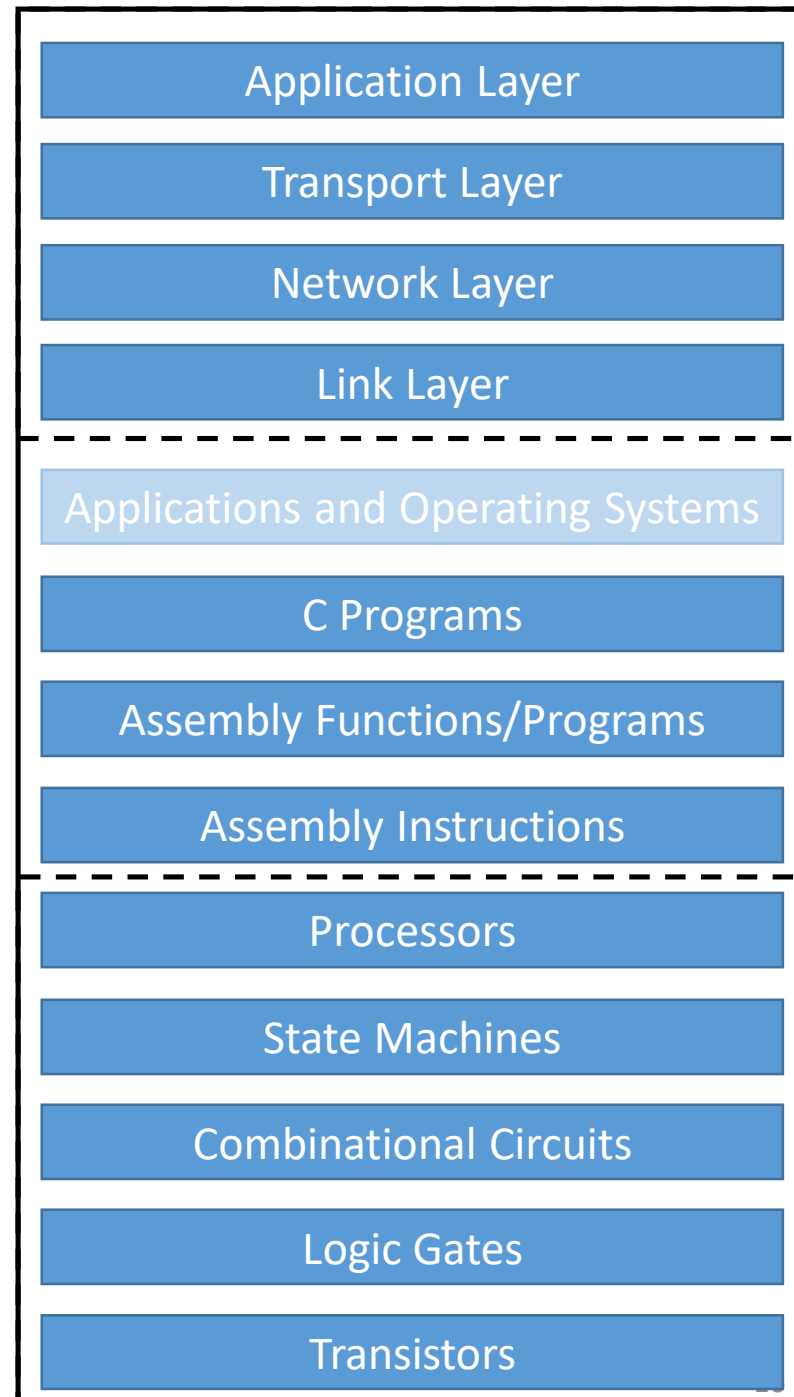
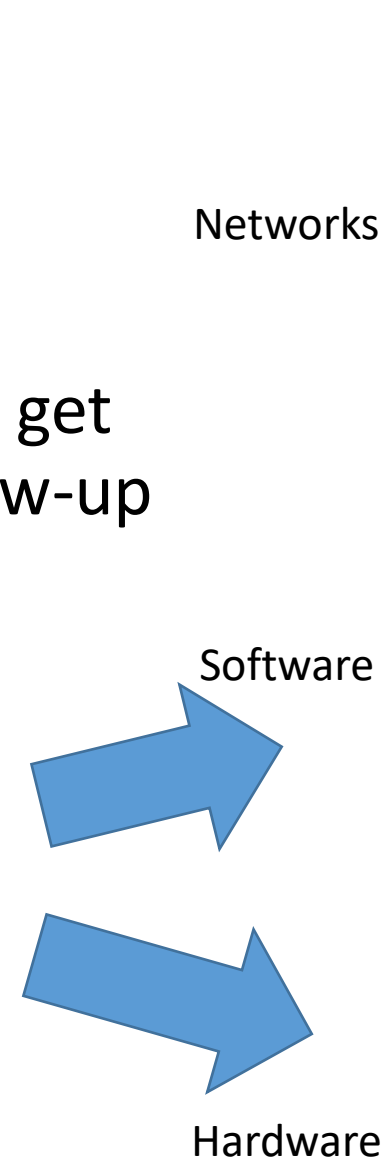
- In this course, we learn the basics to get the big picture → dig deeper in follow-up courses!

- **Side Channel Security**

“Learn about the fatal consequences of side channels”



<https://meltdownattack.com/>





# **Administrative Stuff**

# Position in Curricula

- **Compulsory course in semester 3** for
  - 211 Information and Computer Engineering
  - 285 Digital Engineering
  - 521 Computer Science
  - 524 Software Engineering and Management
- **Elective compulsory course in semester 3** for
  - 054, 414 Supplementary Bachelor's program Teacher Training: Secondary Schools (General Education), Subject: Informatics
  - 198 Teacher Education Programme for Secondary Level

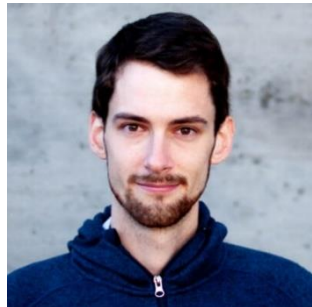
# Team



Stefan  
Mangard



Jakob  
Heher



Lukas  
Prokop



Robert  
Schilling

## Teaching Assistants

- Constantin Piber
- Fabian Burgmann
- Ferdinand Bachmann
- Katrin Schupfer
- Marcel Maderecker
- Martin Unterguggenberger
- Matthias Fischer
- Moritz Waser
- Niklas Skardelly
- Stefan Weiglhofer

# Teaching Assistants



Katrin Schupfer



Stefan Weiglhofer



Fabian Burgmann



Matthias Fischer



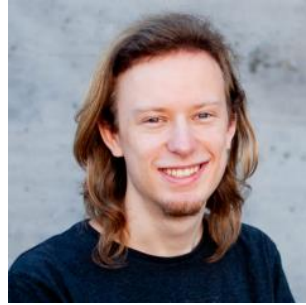
Marcel Maderecker



Moritz Waser



Niklas Skardelly



Ferdinand Bachmann



Constantin Piber



Martin Unterguggenberger

# Material and Contact

- Email

[con@iaik.tugraz.at](mailto:con@iaik.tugraz.at)

- Course website including all material

<http://www.iaik.tugraz.at/con>

- Discord invitation link

<https://discord.com/invite/mxuUnjP>

# Lecture

**From Hardware**

**To Software**

# Lecture

- Location
  - HS i13
  - Online Streaming via TU Graz Tube
- Time

Each week on Wednesday there is a block of 120min lecture plus a 10 min break

  - 13:00 – 14:00 lecture block 1
  - 14:00 – 14:10 break
  - 14:10 – 15:10 lecture block 2

The positioning of the break may vary ;-)
- Programming examples are available from <https://extgit.iaik.tugraz.at/con/examples-2021/>



# Lecture Content and Timeline

- **Block 1: Basics (Stefan Mangard)**

- Chapter 1: Combinational Circuits
- Chapter 2: Number representation and arithmetic
- Chapter 3: Finite State Machines

Block 1 - Basics

- **Block 2: Processors (Stefan Mangard)**

- Chapter 4: Basics of Processor Design
- Chapter 5: Pipelining
- Chapter 6: Pipelining Issues
- Chapter 7: Hardware/Software Contract, Stack

Block 2 - Processors

- **Block 3: Networks (Jakob Heher)**

- Chapter 8: Network Basics
- Chapter 9: Network Layer
- Chapter 10: Transport Layer
- Chapter 11: Application Layer

Block 3 - Networks

- **Block 4: Memory System (Stefan Mangard)**

- Chapter 12: Caches
- Chapter 13: Virtual Memory

Block 4 – Memory System

# Material

- Slides
  - Central source
- This course is based on the RISC-V instruction set
  - Many tutorials and materials can be found on the web
  - <https://riscv.org/>



- Textbook:
  - Computer Organization & Design: The Hardware/Software Interface (David A. Patterson / John L. Hennessy)
  - We partly cover the content of this book

# Practical

# Tasks

Deadline	Topic	Toolchain	Points
29.10.2021	Divider	Digital / SystemVerilog	15
05.11.2021	Divider and CPU integration	SystemVerilog	20
26.11.2021	Pipeline a RISC-V CPU	SystemVerilog	20
03.12.2021	Quicksort in RISC-V	RISC-V Assembly	15
14.01.2022	Switching	C/C++	10
21.01.2022	A tiny HTTP server	C/C++	20

88–100	Sehr gut (1)
76–87	Gut (2)
63–75	Befriedigend (3)
51–62	Genügend (4)
0–50	Nicht genügend (5)

# Mode of Operation

- There is a PDF assignment for each task
- There is a video tutorial for each assignment (+ extra video tutorial for “Getting Started” and “SystemVerilog”)
- All tutorials take place online via Discord → organized as Q & A sessions
- Three tasks with 2 subtasks each.  
One week between deadlines of subtasks.

# Assignment

- PDF Assignment for Task 1 is distributed with your git repository this week
- Video tutorials for the next task will be published no later than with the deadline of the task before

Publication Date	Tutorial Video Content
06.10.2021	Getting started
06.10.2021	Task 1.a
06.10.2021	SystemVerilog
06.10.2021	Task 1.b
05.11.2021	Task 2.a
05.11.2021	Task 2.b
03.12.2021	Task 3.a
03.12.2021	Task 3.b

# Question hours

- 10 groups, 10 teaching assistants (TAs)
- Question hours start next week (13.10.2020)
- Weekly question hours specific for each group

	Monday	Tuesday	Wednesday	Thursday	Friday
08:00	Martin		Stefan	Katrin	
09:00	Matthias	Marcel		Fabian	
14:00	Niklas				
16:00		Ferdinand	Moritz		
17:00					Constantin

# Submissions

- Submission via GitLab
- GitLab repositories will be distributed via Email by the end of this week



# Interviews

- Interviews will happen on Discord (microphone needed!)
- TA will pick you up from #con-waiting-room

Week	Interview scope
06–10.12.2021	Task 1.a, 1.b, 2.a
24–28.01.2022	Task 2.b, 3.a, 3.b

# Resources

- Course Web:
  - <https://www.iaik.tugraz.at/con>
- Virtual machine with all tools installed
  - <https://seafile.iaik.tugraz.at/f/d91aa405a7584e7c847f/>
- Upstream Repository and Assignment:
  - <https://extgit.iaik.tugraz.at/con/practicals-2021>
- Tutorials:
  - Regularly, almost every week

# Plagiarism

- **We perform plagiarism checks!**
- **All involved people** fail the practical
  - We will not invest time on researching who copied from whom
- If you plagiarize parts of the program, it is still a case of plagiarism
- How to avoid plagiarism?
  - **Do not share code!**
  - Do not tell/dictate others your solution!
  - Commit regularly to your git repository!
  - The practical is **no group work!**

# Plagiarism example: Identical program

```
#include <stdio.h>

if(X8 > 23) goto Print;
X5 = X4 + X8;
X7 = X3 - X8;

if(X5 < 0) goto LessThanZero;

X3 = X3 + X5;
X4 = X4 >> X7;
X2 = *(X1 + X8);
X5 = X5 - X2;
X8++;
goto Start;

LessThanZero:
X3 = X3 >> X5;
X2 = *(X1 + X8);
X5 = X5 * X2;
X8++;
goto Start;

return 0 ;
}

#include <stdio.h>


if(X8 > 23) goto Print;
X5 = X4 + X8;
X7 = X3 - X8;

if(X5 < 0) goto LessThanZero;


X3 = X3 + X5;
X4 = X4 >> X7;
X2 = *(X1 + X8);
X5 = X5 - X2;
X8++;
goto Start;

LessThanZero:
X3 = X3 >> X5;
X2 = *(X1 + X8);
X5 = X5 * X2;
X8++;
goto Start;

return 0 ;
}
```



# Plagiarism example: Variables renamed

<pre> #include &lt;stdio.h&gt;  if(X8 &gt; 23) goto Print; X5 = X4 + X8; X7 = X3 - X8;  if(X5 &lt; 0) goto LessThanZero;  X3 = X3 + X5; X4 = X4 &gt;&gt; X7; X2 = *(X1 + X8); X5 = X5 - X2; X8++; goto Start;  LessThanZero: X3 = X3 &gt;&gt; X5; X2 = *(X1 + X8); X5 = X5 * X2; X8++; goto Start;  return 0 ; } </pre>		<pre> #include &lt;stdio.h&gt;  if(X6 &gt; 23) goto Print; X3 = X2 + X6; R4 = X1 - X6;  if(X5 &lt; 0) goto Negative;  X1 = X1 - X3; X2 = X2 &gt;&gt; R4; X8 = *(X7 + X6); X5 = X5 - X8; X6++; goto Begin;  Negative: X1 = X1 &gt;&gt; X3; X8 = *(X7 + X6); X5 = X5 * X8; X6++; goto Begin;  return 0; } </pre>
---	--	--

This is still the same program!

# Plagiarism example: Branches flipped

```

#include <stdio.h>

if(X8 > 23) goto Print;
X5 = X4 + X8;
X7 = X3 - X8;

if(X5 < 0) goto LessThanZero;

X3 = X3 + X5;
X4 = X4 >> X7;
X2 = *(X1 + X8);
X5 = X5 - X2;
X8++;
goto Start;

LessThanZero:
X3 = X3 >> X5;
X2 = *(X1 + X8);
X5 = X5 * X2;
X8++;
goto Start;

return 0 ;
}

```

```

#include <stdio.h>

if(X6 > 23) goto Print;
X3 = X2 + X6;
R4 = X1 - X6;

if(X5 >= 0) goto Positive;

X1 = X1 >> X3;
X8 = *(X7 + X6);
X5 = X5 * X8;
X6++;
goto Begin;

Positive:
X1 = X1 + X3;
X2 = X2 >> R4;
X8 = *(X7 + X6);
X5 = X5 - X8;
X6++;
goto Begin;

return 0;
}

```

This is still the  
same program!

Do not invest time on trying to bypass detection of plagiarism

**Invest your time on the assignments**



# Your First Actions for the Practical

- **Register** for one of the groups in TUGRAZonline (**deadline: tomorrow**)
- **Install** the CON2021 Virtual Machine

On Friday, Oct 8<sup>th</sup>, all registered students receive their git repositories including assignment sheet

- **Clone** the upstream repository
- **Read** the assignment sheet
- **Watch** the video tutorial for assignment 1
- **Attend** the online tutorials next week



# Effort

- This is a 7 ECTS course – this is approx. one quarter of your semester (approx. 200 working hours)
- There are 120 minutes lecture per week
- This lecture and the practical runs through many abstraction layers with many different tools – work on the course every week (“Am Ball bleiben”)