

Digital Design & Computer Arch.

Lecture 3b: Introduction to the Labs and FPGAs

Prof. Onur Mutlu
(Lecture by Hasan Hassan)
ETH Zurich
Spring 2021
4 March 2021

Lab Sessions

■ Where?

- **Online (Zoom Meetings)**

■ When?

- Tuesday 16:15-18:00
- Wednesday 16:15-18:00
- Friday 08:15-10:00
- Friday 10:15-12:00

Grading

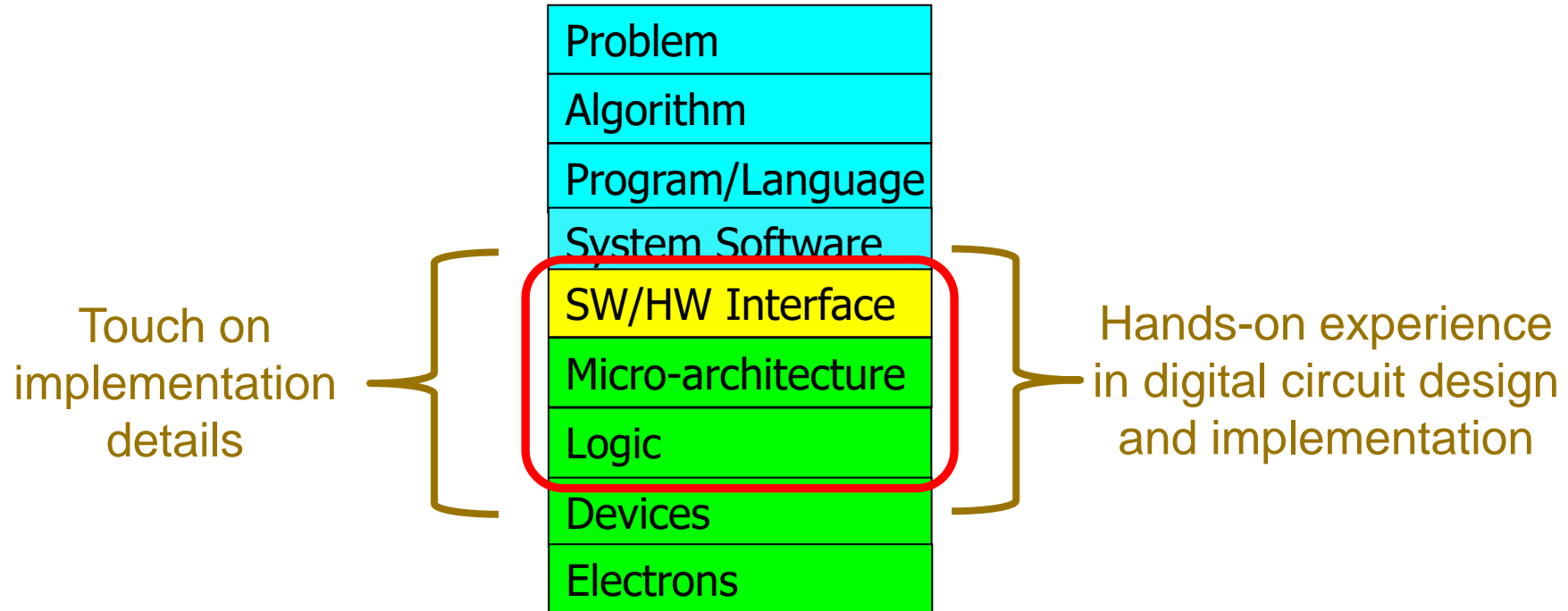
- **10** labs, **30** points in total
- We will put the **lab manuals** online
 - <https://safari.ethz.ch/digitaltechnik/doku.php?id=labs>
- **Grading Policy**
 - In-class evaluation (70%) and *mandatory* lab reports (30%)
 - *1-point penalty for late submission of the report*
 - You can use your grades for labs from past years
 - You can find your grades in last year's Moodle page: <https://moodle-app2.let.ethz.ch/course/view.php?id=12285>
You should finish the labs within 1 week after they are announced
- **For questions**
 - Piazza (preferred)
 - digitaltechnik@lists.inf.ethz.ch (Emails are sent to all TAs)

Agenda

- Logistics
- **What Will We Learn?**
- FPGAs in Today's Systems
- Overview of the Lab Exercises
- What is an FPGA?
- Programming an FPGA
- Tutorial and Demo

What Will We Learn?

The Transformation Hierarchy



Understanding how a processor works
underneath the software layer

What Will We Learn? (2)

- How to make **trade-offs** between **performance** and **area/complexity** in your hardware implementation
- **Hands-on experience** on:
 - ❑ Hardware **Prototyping** on FPGA
 - ❑ **Debugging** Your Hardware Implementation
 - ❑ Hardware Description Language (**HDL**)
 - ❑ Hardware Design Flow
 - ❑ Computer-Aided Design (**CAD**) Tools

Agenda

- Logistics
- What Will We Learn?
- **FPGAs in Today's Systems**
- Overview of the Lab Exercises
- What is an FPGA?
- Programming an FPGA
- Tutorial and Demo

FPGAs in Today's Systems: Project Brainwave

- “Microsoft’s *Project Brainwave* is a **deep learning platform** for real-time AI inference in the cloud and on the edge. A soft Neural Processing Unit (NPU), based on a high-performance **field-programmable gate array (FPGA)**, accelerates deep neural network (DNN) inferencing, with applications in computer vision and natural language processing. Project Brainwave is transforming computing by augmenting CPUs with an interconnected and configurable compute layer composed of programmable silicon.”

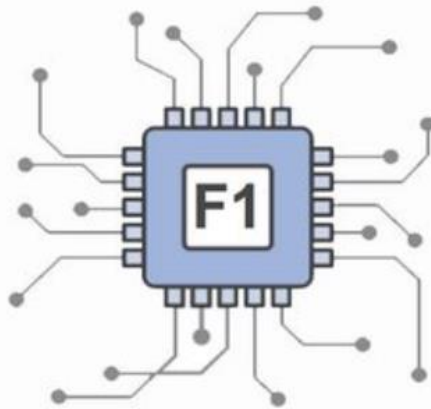


<https://www.microsoft.com/en-us/research/project/project-brainwave/>

<https://www.microsoft.com/en-us/research/blog/microsoft-unveils-project-brainwave/>

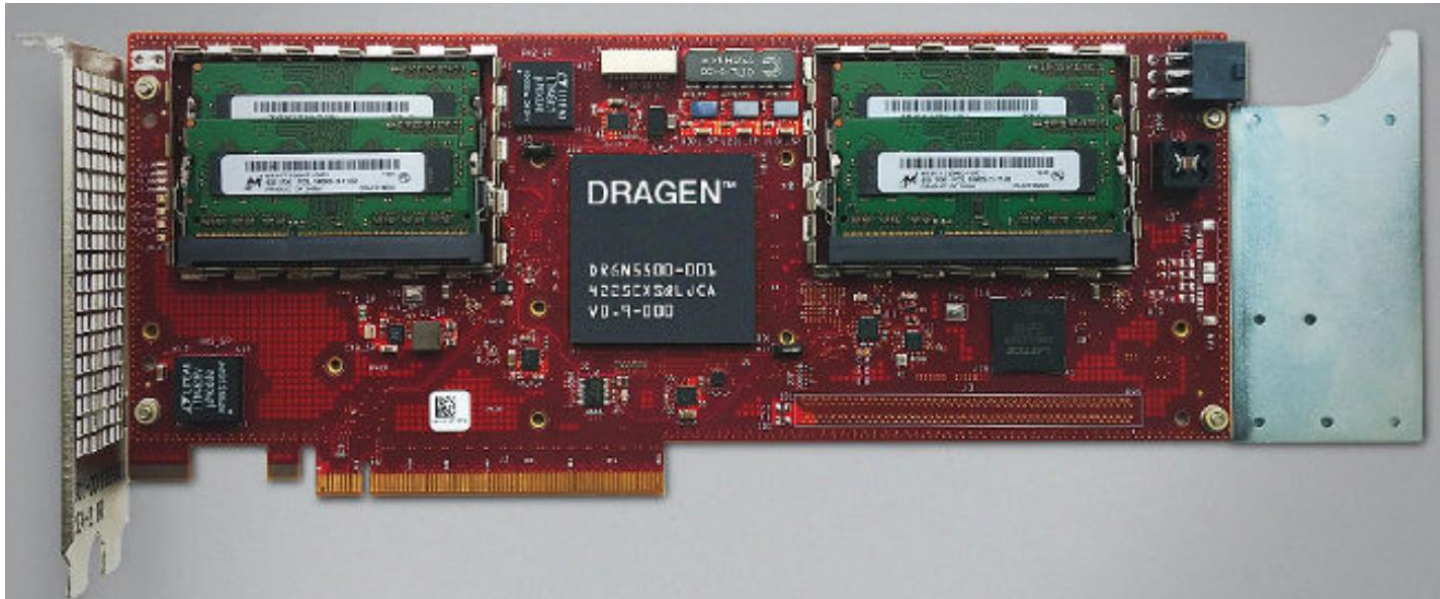
FPGAs in Today's Systems: Amazon EC2 F1

- “Amazon EC2 F1 instances use **FPGAs** to enable delivery of **custom hardware accelerations**. F1 instances are *easy to program* and come with everything you need to develop, simulate, debug, and compile your hardware acceleration code, including an FPGA Developer AMI and supporting hardware level development on the cloud. Using F1 instances to deploy hardware accelerations can be useful in many applications **to solve complex science, engineering, and business problems that require high bandwidth, enhanced networking, and very high compute capabilities.**”



FPGAs in Today's Systems: DNA Sequencing

- DRAGEN's suite of analysis pipelines are engineered to run on **FPGAs**, offering hardware-accelerated implementations of genomic analysis algorithms, including BCL conversion, mapping and alignment, sorting, duplicate marking and haplotype variant calling.



Illumina DRAGEN (Dynamic Read Analysis for GENomics) Bio-IT Platform



Illumina NextSeq 2000

<https://www.illumina.com/products/by-type/informatics-products/dragen-bio-it-platform.html>

FPGAs in Today's Systems: More Bioinformatics

Bioinformatics



Article Navigation

GateKeeper: a new hardware architecture for accelerating pre-alignment in DNA short read mapping ^{FREE}

Mohammed Alser ✉, Hasan Hassan, Hongyi Xin, Oğuz Ergin, Onur Mutlu ✉, Can Alkan ✉

Bioinformatics, Volume 33, Issue 21, 01 November 2017, Pages 3355–3363,

<https://doi.org/10.1093/bioinformatics/btx342>

Published: 31 May 2017 **Article history** ▼

Bioinformatics

doi.10.1093/bioinformatics/xxxxx

Advance Access Publication Date: Day Month Year

Manuscript Category

OXFORD

Subject Section

SneakySnake: A Fast and Accurate Universal Genome Pre-Alignment Filter for CPUs, GPUs, and FPGAs

Mohammed Alser ^{1,2,*}, Taha Shahroodi ¹, Juan Gómez-Luna ^{1,2},
Can Alkan ^{4,*}, and Onur Mutlu ^{1,2,3,4,*}

Alser+, "[GateKeeper: A New Hardware Architecture for Accelerating Pre-Alignment in DNA Short Read Mapping](#)", *Bioinformatics*, 2017.

Alser+, "[SneakySnake: A Fast and Accurate Universal Genome Pre-Alignment Filter for CPUs, GPUs, and FPGAs](#)", *Bioinformatics*, 2020.

FPGAs in Today's Systems: SoftMC

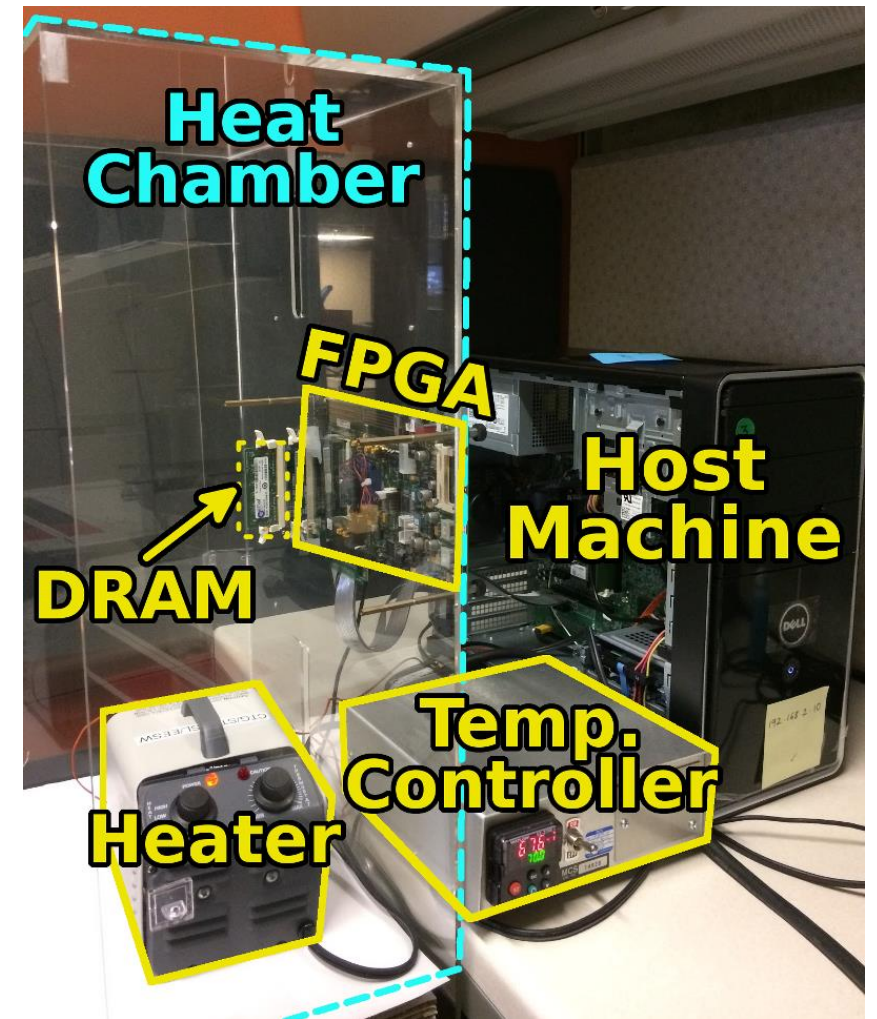
- An open-source FPGA-based infrastructure for experimental studies on DRAM

- **Flexible**

- **Easy to Use (C++ API)**

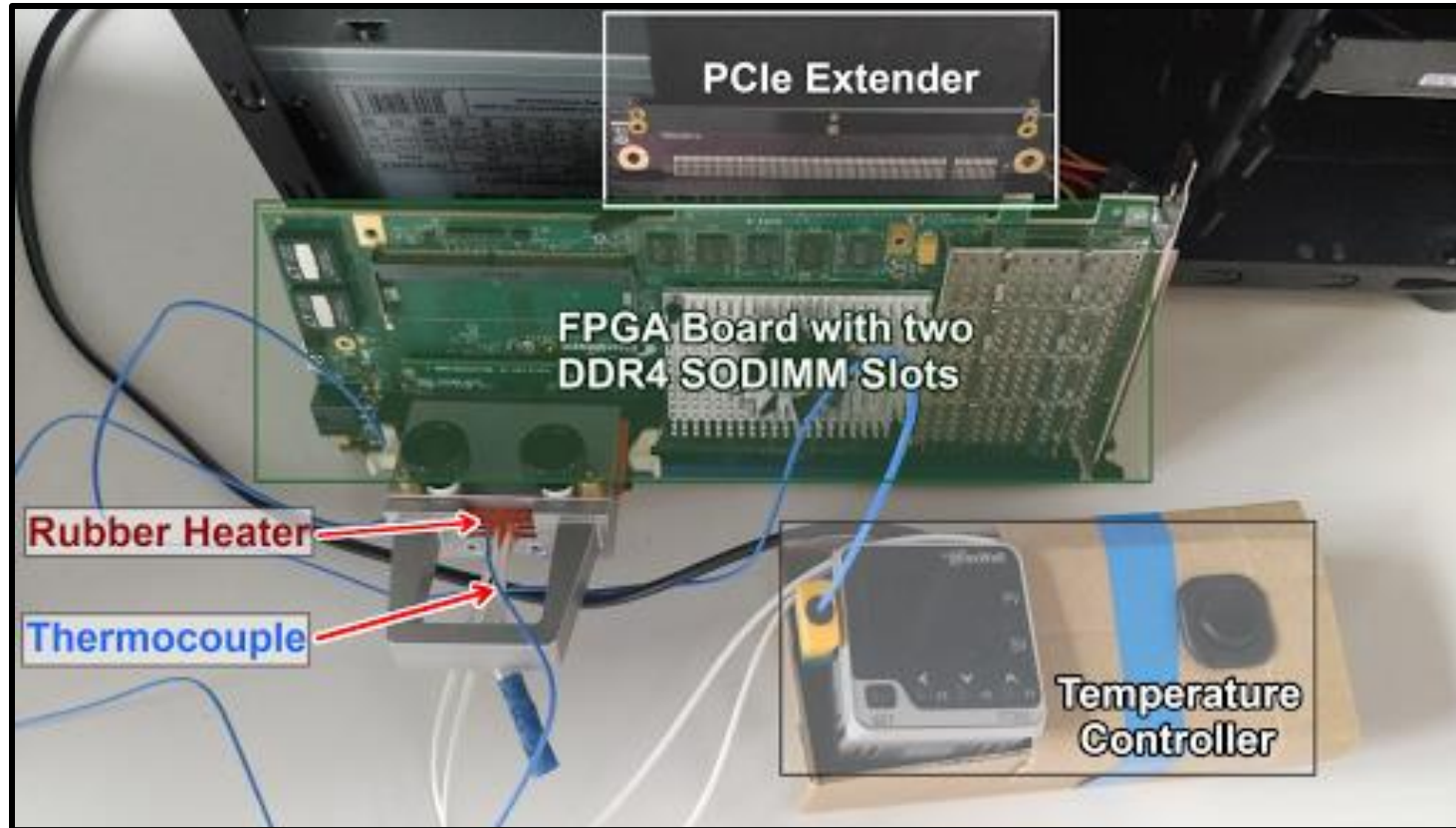
- **Open-source**

github.com/CMU-SAFARI/SoftMC



Hassan+, "[SoftMC: A Flexible and Practical Open-Source Infrastructure for Enabling Experimental DRAM Studies](#)," HPCA 2017.

SoftMC for DDR4



Frigo+, "[TRRespass: Exploiting the Many Sides of Target Row Refresh](#)", S&P 2020

Kim+, "[Revisiting RowHammer: An Experimental Analysis of Modern Devices and Mitigation Techniques](#)", ISCA 2020

RowHammer: Seven Years Ago...

- Yoongu Kim, Ross Daly, Jeremie Kim, Chris Fallin, Ji Hye Lee, Donghyuk Lee, Chris Wilkerson, Konrad Lai, and Onur Mutlu,

"Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors"

Proceedings of the 41st International Symposium on Computer Architecture (ISCA), Minneapolis, MN, June 2014.

[\[Slides \(pptx\) \(pdf\)\]](#) [\[Lightning Session Slides \(pptx\) \(pdf\)\]](#) [\[Source Code and Data\]](#)

Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors

Yoongu Kim¹ Ross Daly* Jeremie Kim¹ Chris Fallin* Ji Hye Lee¹
Donghyuk Lee¹ Chris Wilkerson² Konrad Lai Onur Mutlu¹

¹Carnegie Mellon University ²Intel Labs

RowHammer in 2020 (I)

- Jeremie S. Kim, Minesh Patel, A. Giray Yaglikci, Hasan Hassan, Roknoddin Azizi, Lois Orosa, and Onur Mutlu,
"Revisiting RowHammer: An Experimental Analysis of Modern Devices and Mitigation Techniques"
Proceedings of the 47th International Symposium on Computer Architecture (ISCA), Valencia, Spain, June 2020.
[[Slides \(pptx\)](#)] [[pdf](#)]
[[Lightning Talk Slides \(pptx\)](#)] [[pdf](#)]
[[Talk Video](#) (20 minutes)]
[[Lightning Talk Video](#) (3 minutes)]

Revisiting RowHammer: An Experimental Analysis of Modern DRAM Devices and Mitigation Techniques

Jeremie S. Kim^{§†} Minesh Patel[§] A. Giray Yağlıkçı[§]
Hasan Hassan[§] Roknoddin Azizi[§] Lois Orosa[§] Onur Mutlu^{§†}

[§]*ETH Zürich*

[†]*Carnegie Mellon University*

RowHammer in 2020 (II)

- Pietro Frigo, Emanuele Vannacci, Hasan Hassan, Victor van der Veen, Onur Mutlu, Cristiano Giuffrida, Herbert Bos, and Kaveh Razavi,

"TRRespass: Exploiting the Many Sides of Target Row Refresh"

Proceedings of the 41st IEEE Symposium on Security and Privacy (S&P), San Francisco, CA, USA, May 2020.

[[Slides \(pptx\)](#) ([pdf](#))]

[[Lecture Slides \(pptx\)](#) ([pdf](#))]

[[Talk Video](#) (17 minutes)]

[[Lecture Video](#) (59 minutes)]

[[Source Code](#)]

[[Web Article](#)]

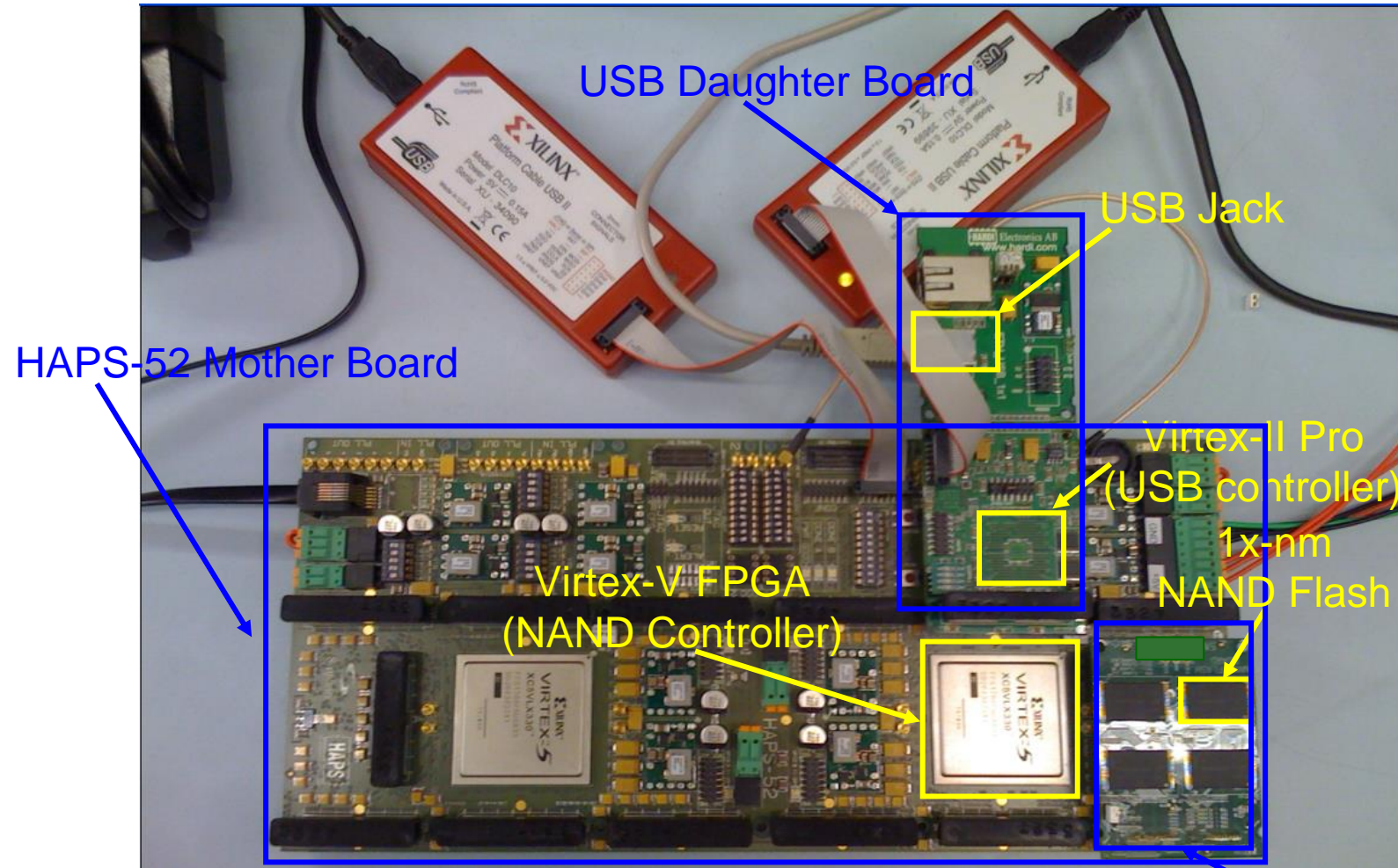
Best paper award.

Pwnie Award 2020 for Most Innovative Research. [Pwnie Awards 2020](#)

TRRespass: Exploiting the Many Sides of Target Row Refresh

Pietro Frigo^{*†} Emanuele Vannacci^{*†} Hasan Hassan[§] Victor van der Veen[¶]
Onur Mutlu[§] Cristiano Giuffrida^{*} Herbert Bos^{*} Kaveh Razavi^{*}

FPGAs in Today's Systems: Characterizing Flash Memories

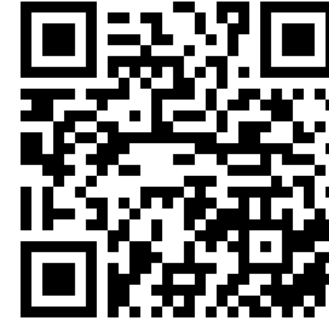


[DATE 2012, ICCD 2012, DATE 2013, ITJ 2013, ICCD 2013, SIGMETRICS 2014, HPCA 2015, DSN 2015, MSST 2015, JSAC 2016, HPCA 2017, DFRWS 2017, PIEEE 2017, HPCA 2018, SIGMETRICS 2018]

NAND Daughter Board



Proceedings of the IEEE, Sept. 2017



Error Characterization, Mitigation, and Recovery in Flash-Memory-Based Solid-State Drives

This paper reviews the most recent advances in solid-state drive (SSD) error characterization, mitigation, and data recovery techniques to improve both SSD's reliability and lifetime.

By YU CAI, SAUGATA GHOSE, ERICH F. HARATSCH, YIXIN LUO, AND ONUR MUTLU

<https://arxiv.org/pdf/1706.08642>

Accelerating Climate Modeling Using FPGAs

- Gagandeep Singh, Dionysios Diamantopoulos, Christoph Hagleitner, Juan Gómez-Luna, Sander Stuijk, Onur Mutlu, and Henk Corporaal,

"NERO: A Near High-Bandwidth Memory Stencil Accelerator for Weather Prediction Modeling"

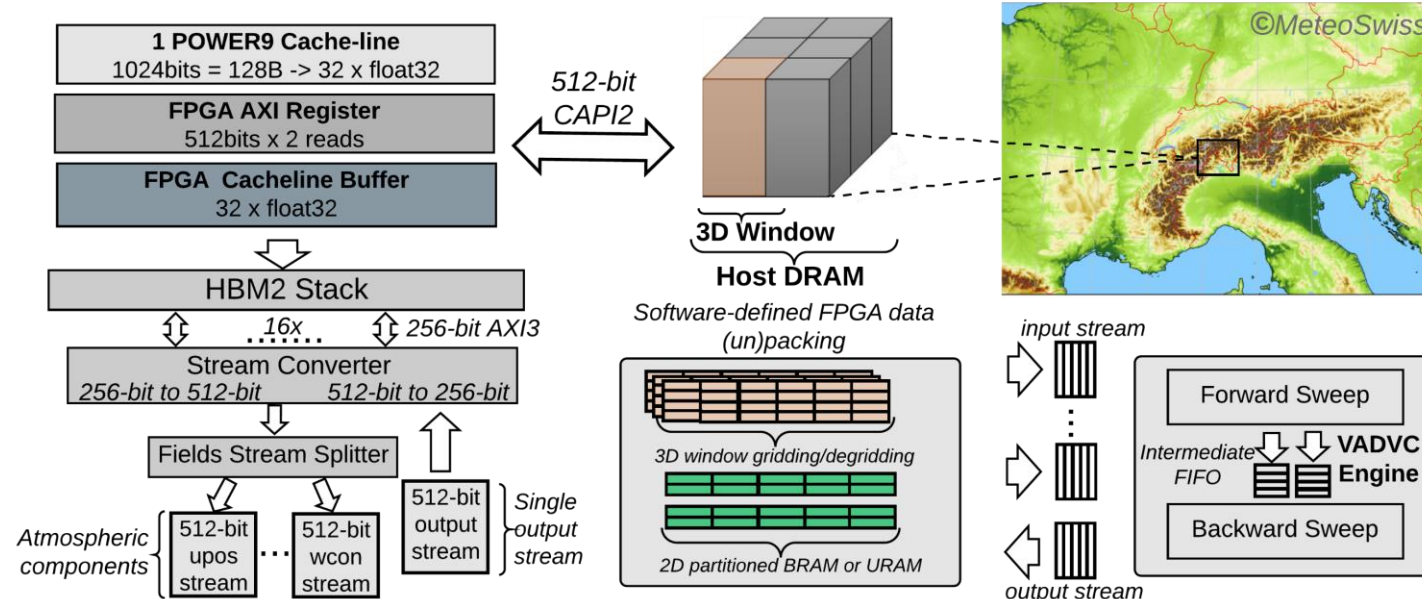
Proceedings of the 30th International Conference on Field-Programmable Logic and Applications (FPL), Gothenburg, Sweden, September 2020.

[[Slides \(pptx\)](#) ([pdf](#))]

[[Lightning Talk Slides \(pptx\)](#) ([pdf](#))]

[[Talk Video](#) (23 minutes)]

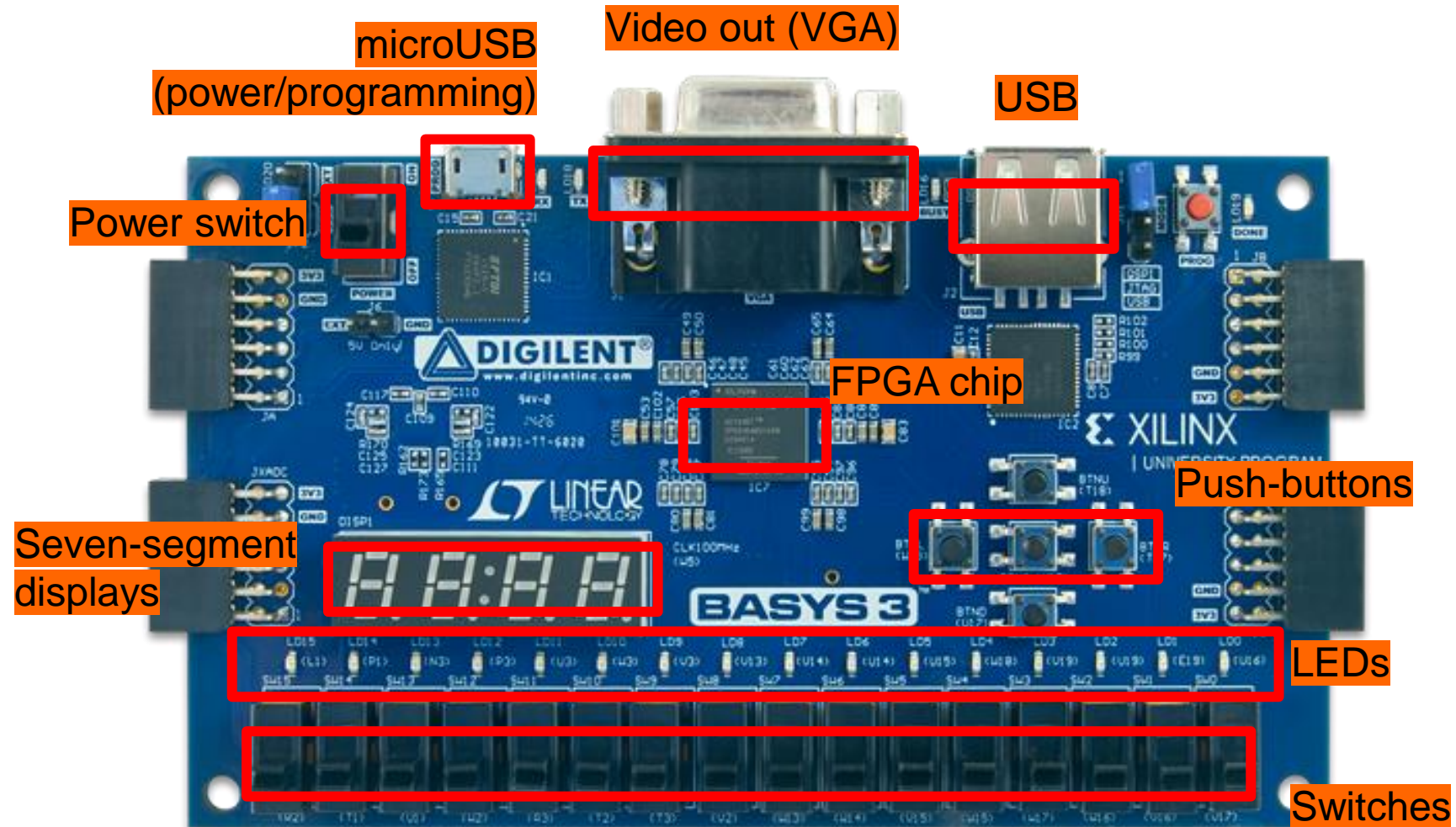
One of the four papers nominated for the Stamatis Vassiliadis Memorial Best Paper Award.



Agenda

- Logistics
- What Will We Learn?
- FPGAs in Today's Systems
- **Overview of the Lab Exercises**
- What is an FPGA?
- Programming an FPGA
- Tutorial and Demo

Basys 3: Our FPGA Board



<https://reference.digilentinc.com/reference/programmable-logic/basys-3/start>

High Level Labs Summary

- At the end of the exercises, we will have built a 32-bit microprocessor running on the FPGA board
 - It will be a small processor, but it will be able to execute pretty much any program
- Each week we will have a new exercise
 - Not all exercises will require the FPGA board
- You are encouraged to experiment with the board on your own
 - We may have some extra boards for those who are interested – unlikely, but ask
 - It is not possible to destroy the board **by programming!**

Lab 1: Drawing a Basic Circuit

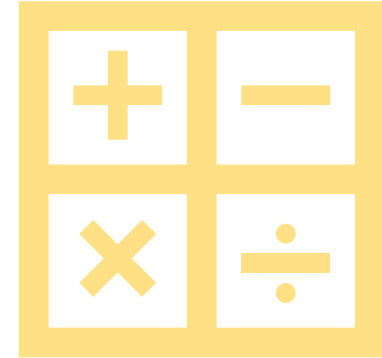
- **Comparison** is a common operation in software programming
 - We usually want to know the **relation** between two variables (e.g., $<$, $>$, $==$, ...)
- We will compare two *electrical signals* (inputs), and find whether they are same
 - The result (output) is also an *electrical signal*
- No FPGA programming involved
 - We encourage you to try later



Lab 2: Mapping Your Circuit to FPGA

- Another common operation in software programming?

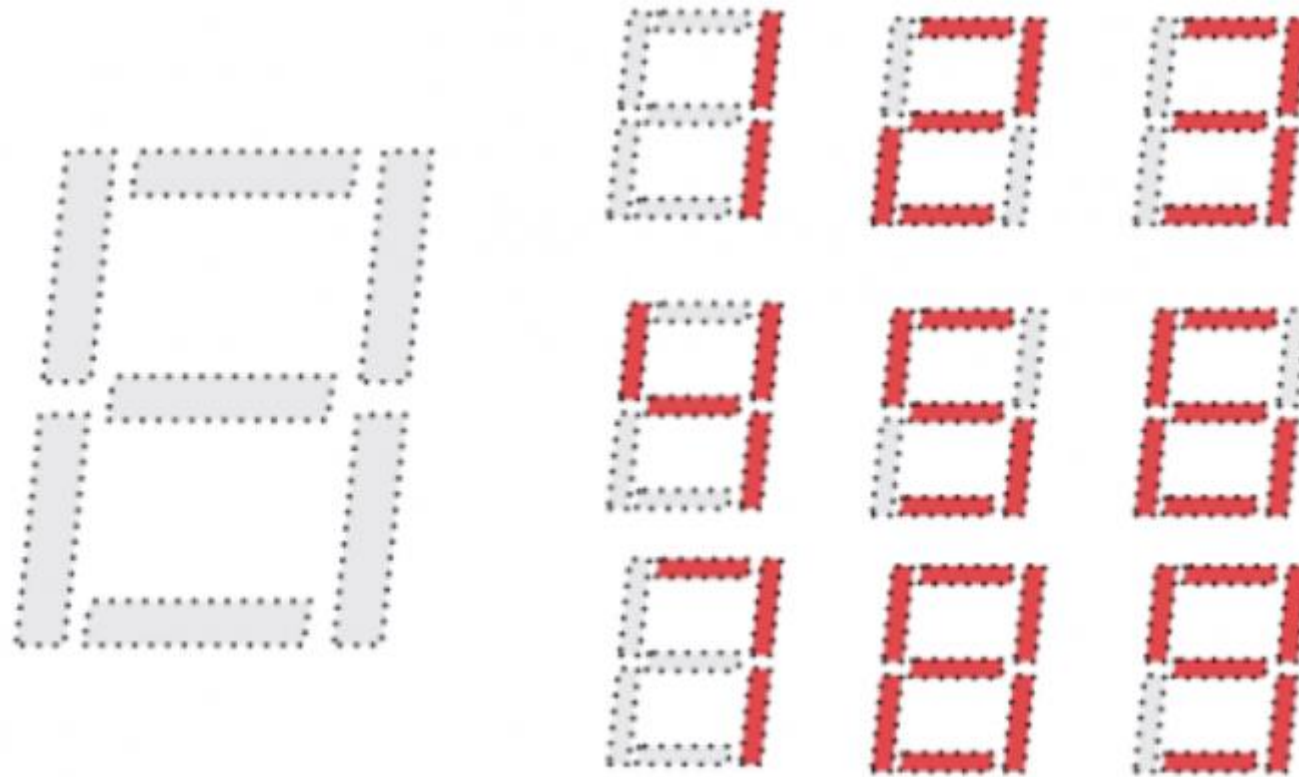
- Addition



- Design a circuit that adds two 1-bit numbers
- Reuse the 1-bit adder multiple times to perform 4-bit addition
- Implement the design on the FPGA board
 - Input: switches
 - Output: LEDs

Lab 3: Verilog for Combinatorial Circuits

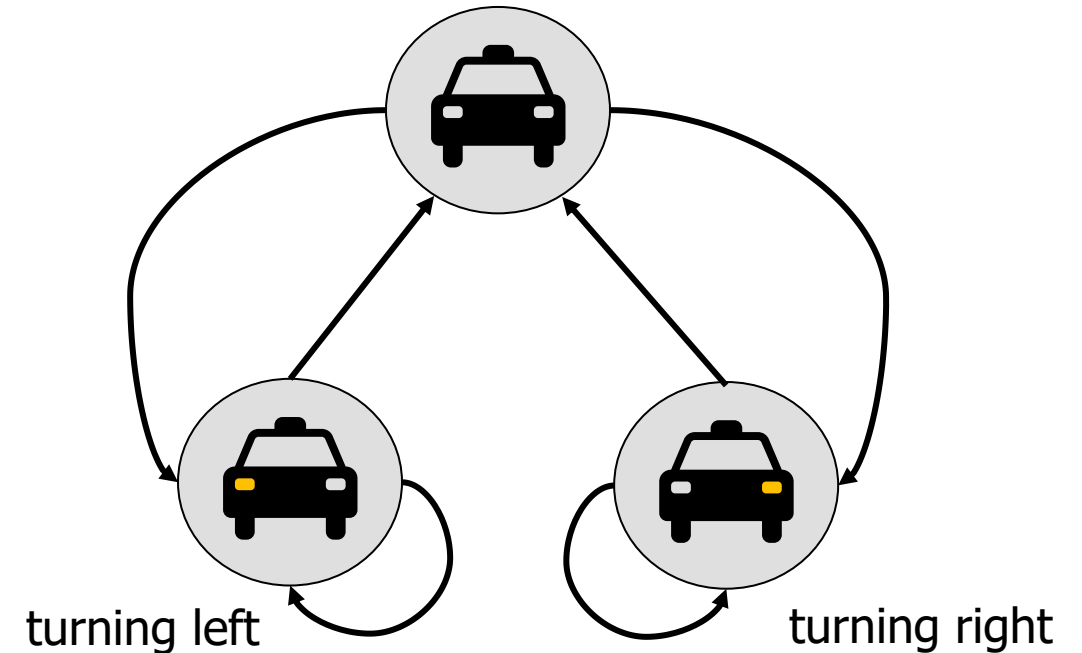
- Show your results from Lab 2 on a **Seven Segment Display**



<https://reference.digilentinc.com/reference/programmable-logic/basys-3/reference-manual>

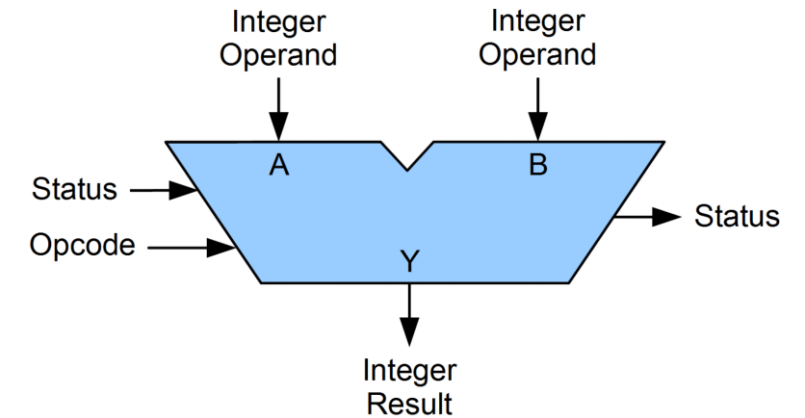
Lab 4: Finite State Machines

- Blinking LEDs for a car's turn signals
 - Implement and use **memories**
 - Change the blinking speed



Lab 5: Implementing an ALU

- Towards implementing your very first **processor**
- Implement your own **Arithmetic and Logic Unit (ALU)**
- An ALU is an important part of the CPU
 - ❑ Arithmetic operations: add, subtract, multiply, compare, ...
 - ❑ Logic operations: AND, OR, ...



Source:

https://en.wikipedia.org/wiki/Arithmetic_logic_unit

Lab 6: Testing the ALU

- **Simulate** your design from Lab 5
- Learn how to **debug** your implementation to resolve problems



Lab 7: Writing Assembly Code

- Programming in **assembly language**
 - MIPS
- Implement a program which you will later use to run on your processor
- Image manipulation

High-level code

```
int sum = 0;

for(int i = 0; i <= 10; i += 2)
{
    sum += i;
}
```

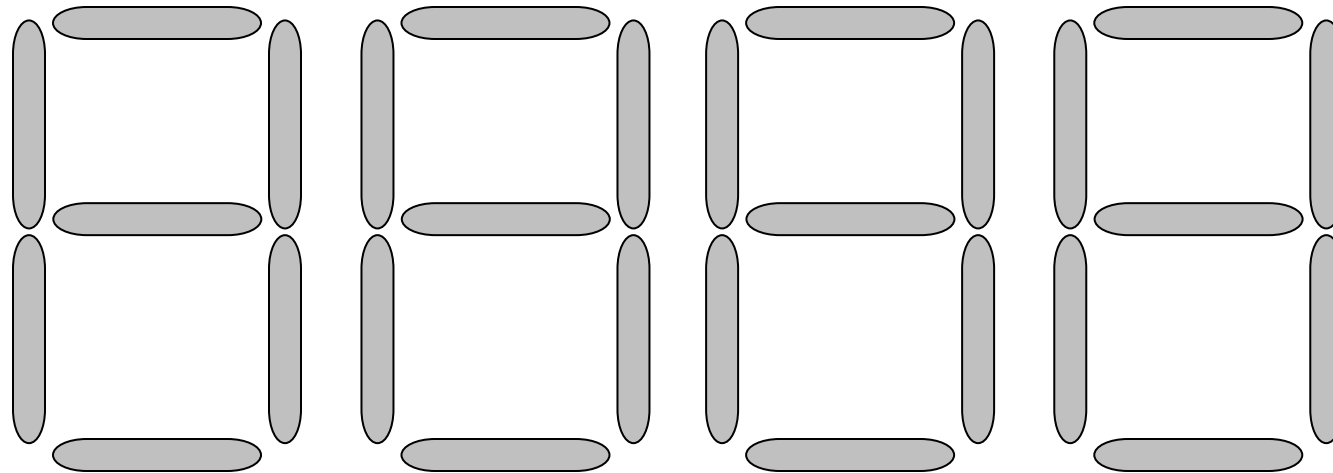
MIPS assembly

```
# i=$s0; sum=$s1

        addi $s0, $0, 0
        addi $s1, $0, 0
        addi $t0, $0, 12
loop:    beq  $s0, $t0, done
        add  $s1, $s1, $s0
        addi $s0, $s0, 2
        j    loop
done:
```

Lab 8: Full System Integration

- Will be covered in two weeks
- Learn how a processor is built
- Complete your first design of a MIPS processor
- Run a "snake" program



Lab 9: The Performance of MIPS

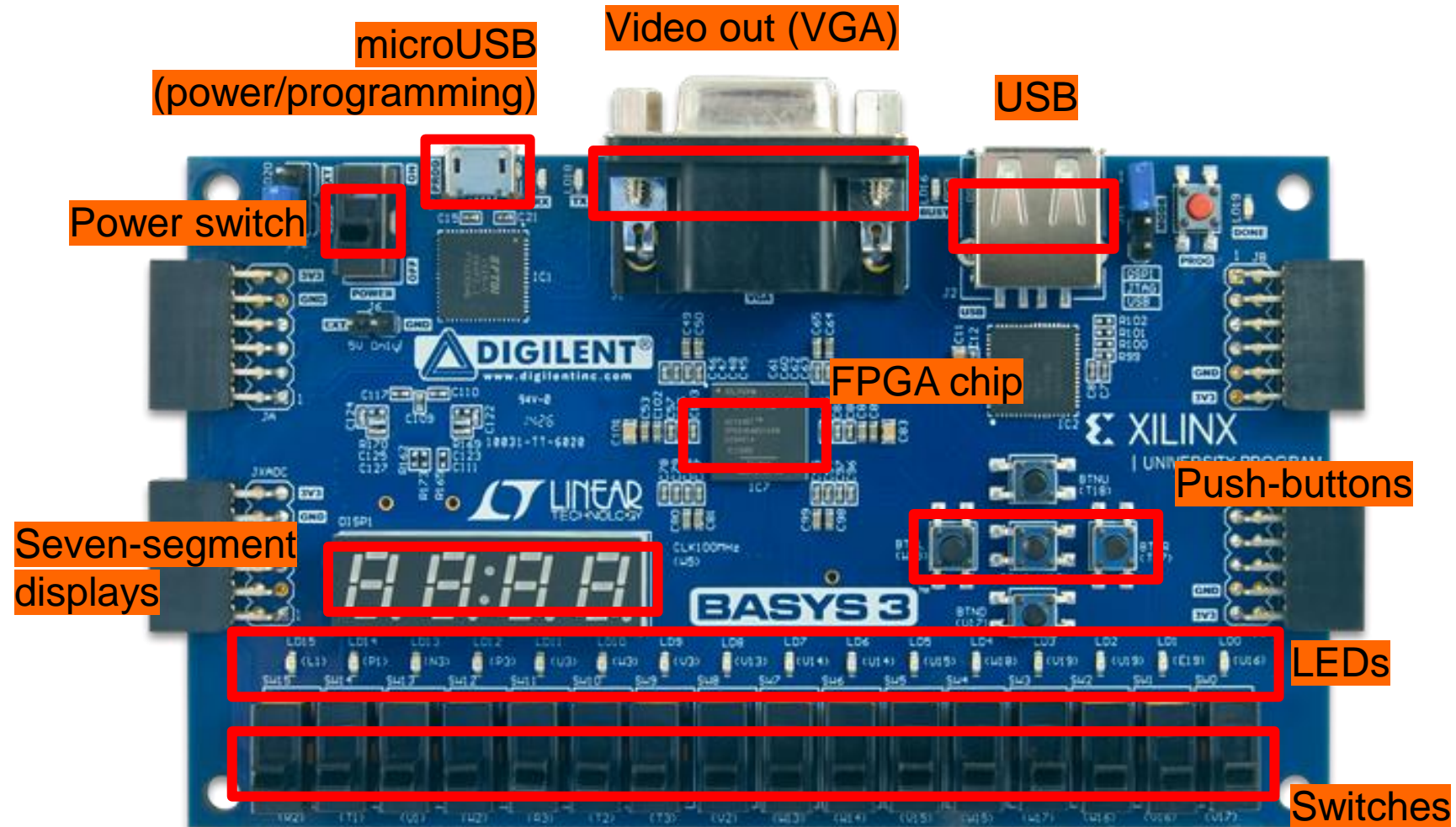
- Improve the **performance** of your processor from Lab 8 by adding new **instructions**
 - ❑ Multiplication
 - ❑ Bit shifting



Agenda

- Logistics
- What Will We Learn?
- FPGAs in Today's Systems
- Overview of the Lab Exercises
- **What is an FPGA?**
- Programming an FPGA
- Tutorial and Demo

Basys 3: Our FPGA Board

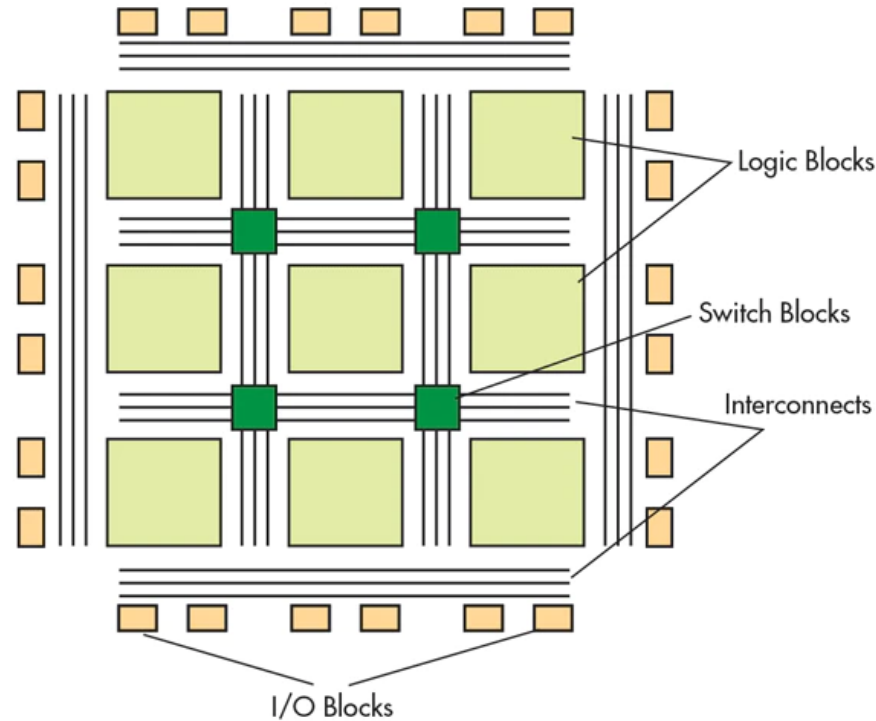


<https://reference.digilentinc.com/reference/programmable-logic/basys-3/start>

What is an FPGA?

- Field Programmable Gate Array: FPGA
- FPGA is a **software-reconfigurable** hardware substrate
 - Reconfigurable **functions**
 - Reconfigurable **interconnection** of functions
 - Reconfigurable **input/output (IO)**
 - ...
- When used well, FPGAs:
 - Provide **higher performance** than running software **on a CPU**
 - Provide **more flexibility of implementation** than **dedicated hardware or a CPU**

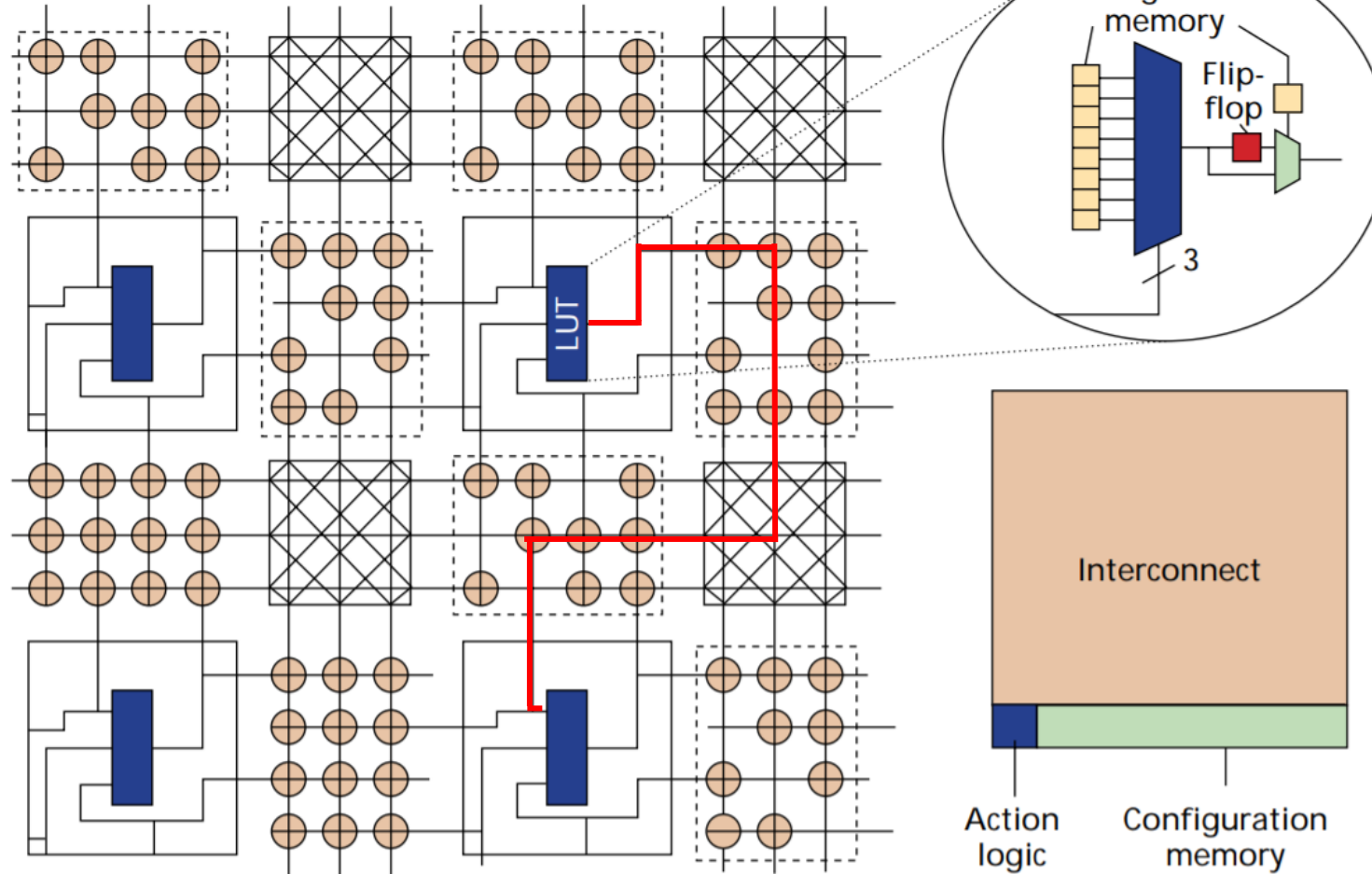
A High-Level Overview of FPGAs



We **configure** logic blocks, their connections, and IO blocks to create hardware circuits and map programs onto those circuits

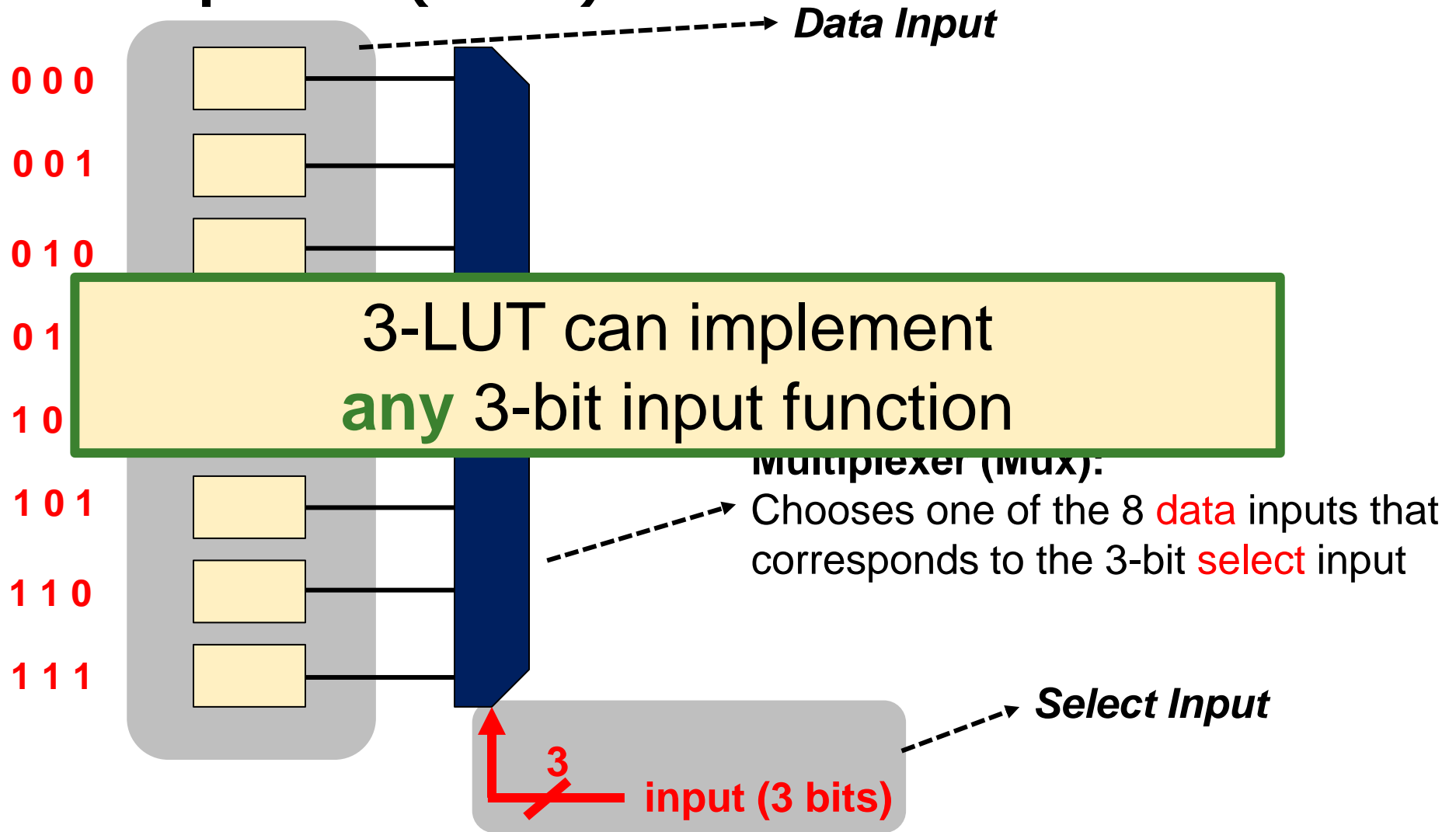
FPGA Architecture - Looking Inside an FPGA

- Two main building blocks:
 - Look-Up Tables (LUT) and Switches



How Do We Program LUTs?

■ 3-bit input LUT (3-LUT)



An Example of Programming a LUT

- Let's implement a function that outputs '1' when there are at least two '1's in a 3-bit input

In C:

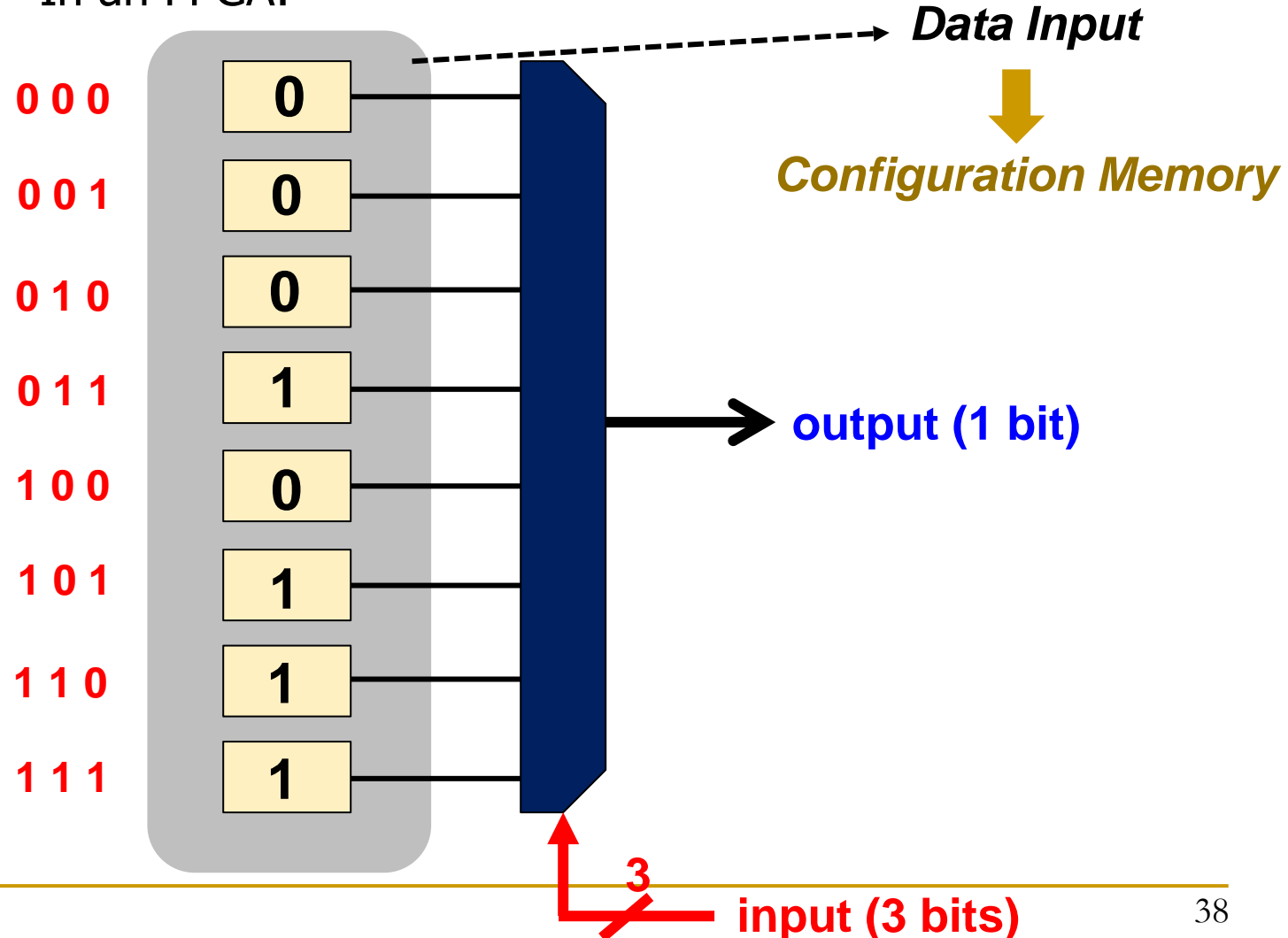
```
int count = 0;
for(int i = 0; i < 3; i++) {
    count += input & 1;
    input = input >> 1;
}

if(count > 1) return 1;

return 0;
```

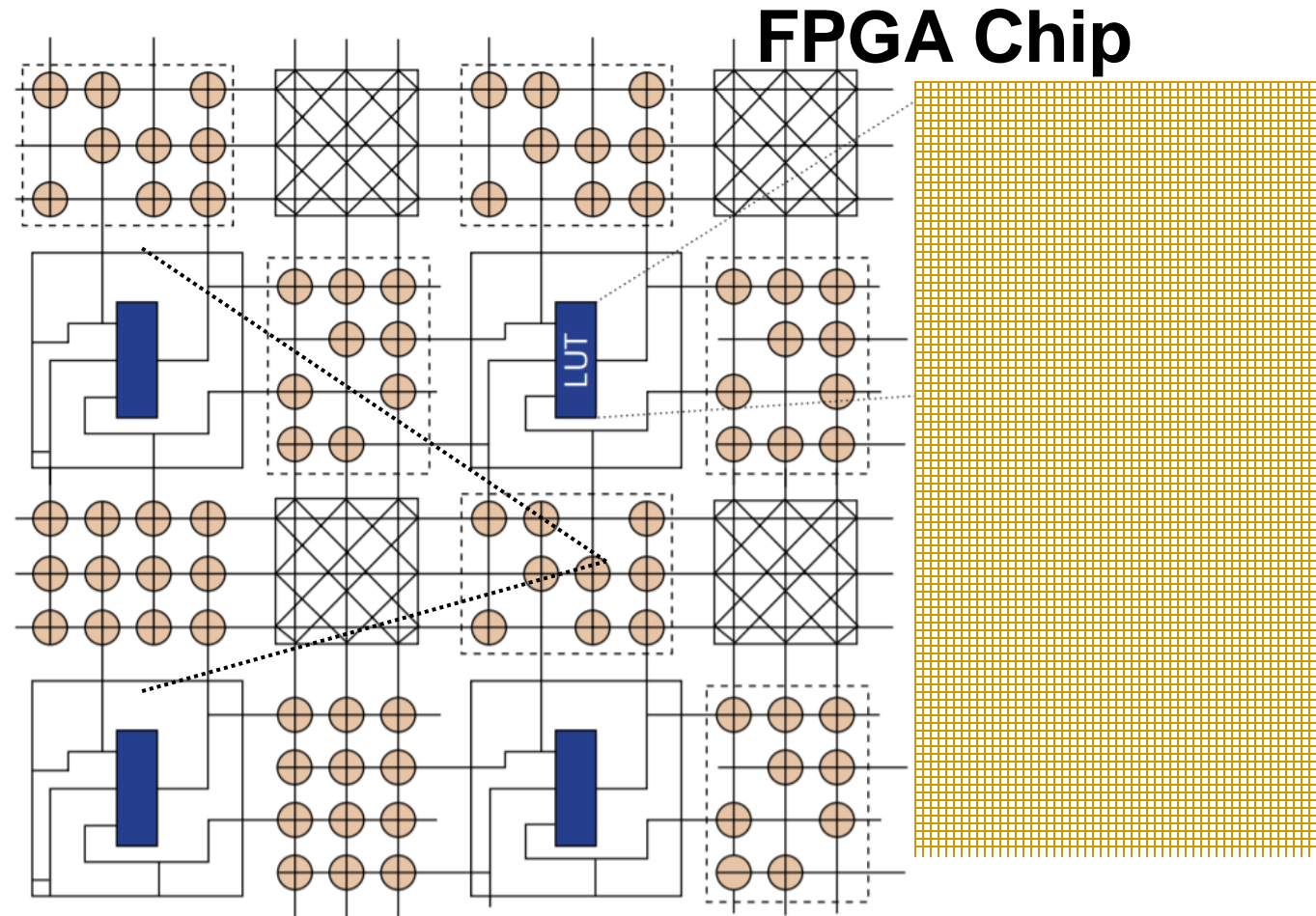
```
switch(input){
    case 0:
    case 1:
    case 2:
    case 4:
        return 0;
    default:
        return 1;}
```

In an FPGA:



How to Implement Complex Functions?

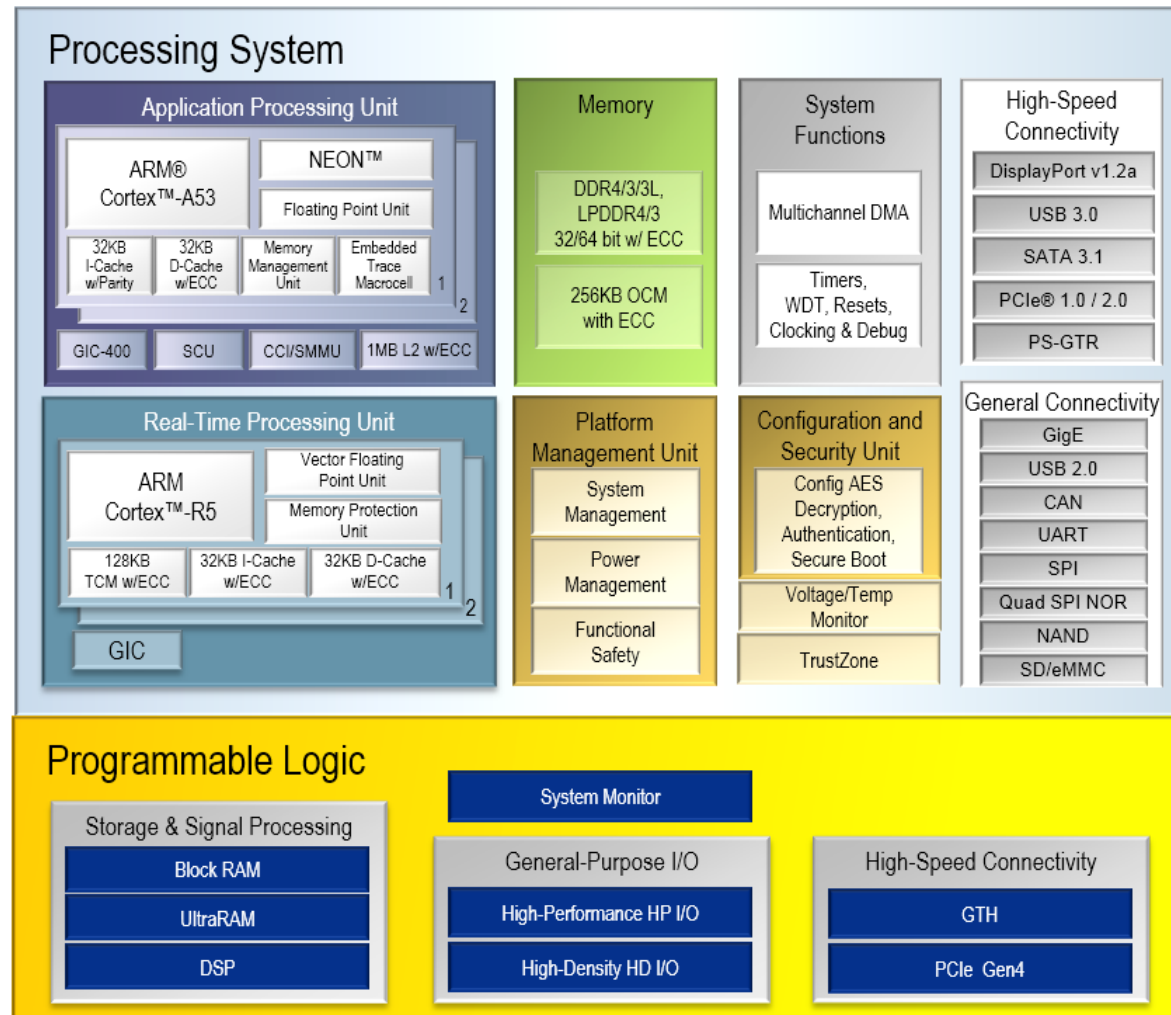
- FPGAs are composed of **many** LUTs and switches



Modern FPGA Architectures

- Typically use LUTs with 6-bit select input (6-LUT)
 - Thousands of them
- MegaBytes of distributed on-chip memory
- Hard-coded special-purpose hardware blocks for high-performance operations
 - Memory interface
 - Low latency and high bandwidth off-chip I/O
 - ...
- Even a general-purpose processor embedded within the FPGA chip

Xilinx Zynq Ultrascale+



<https://www.xilinx.com/products/silicon-devices/soc/zynq-ultrascale-mpsoc.html>

Advantages & Disadvantages of FPGAs

■ Advantages

- ❑ An algorithm can be implemented directly in hardware
 - No ISA, high specialization → high performance, high energy efficiency
- ❑ Low development cost (vs. a custom hardware design)
- ❑ Short time to market (vs. a custom hardware design)
- ❑ Reconfigurable in the field
- ❑ Usable and reusable for many purposes
 - Good for both prototyping and application acceleration

■ Disadvantages

- ❑ Not as **fast** and **power efficient** as *dedicated hardware customized for an algorithm*
- ❑ Reconfigurability comes at a cost: significant **area overhead**

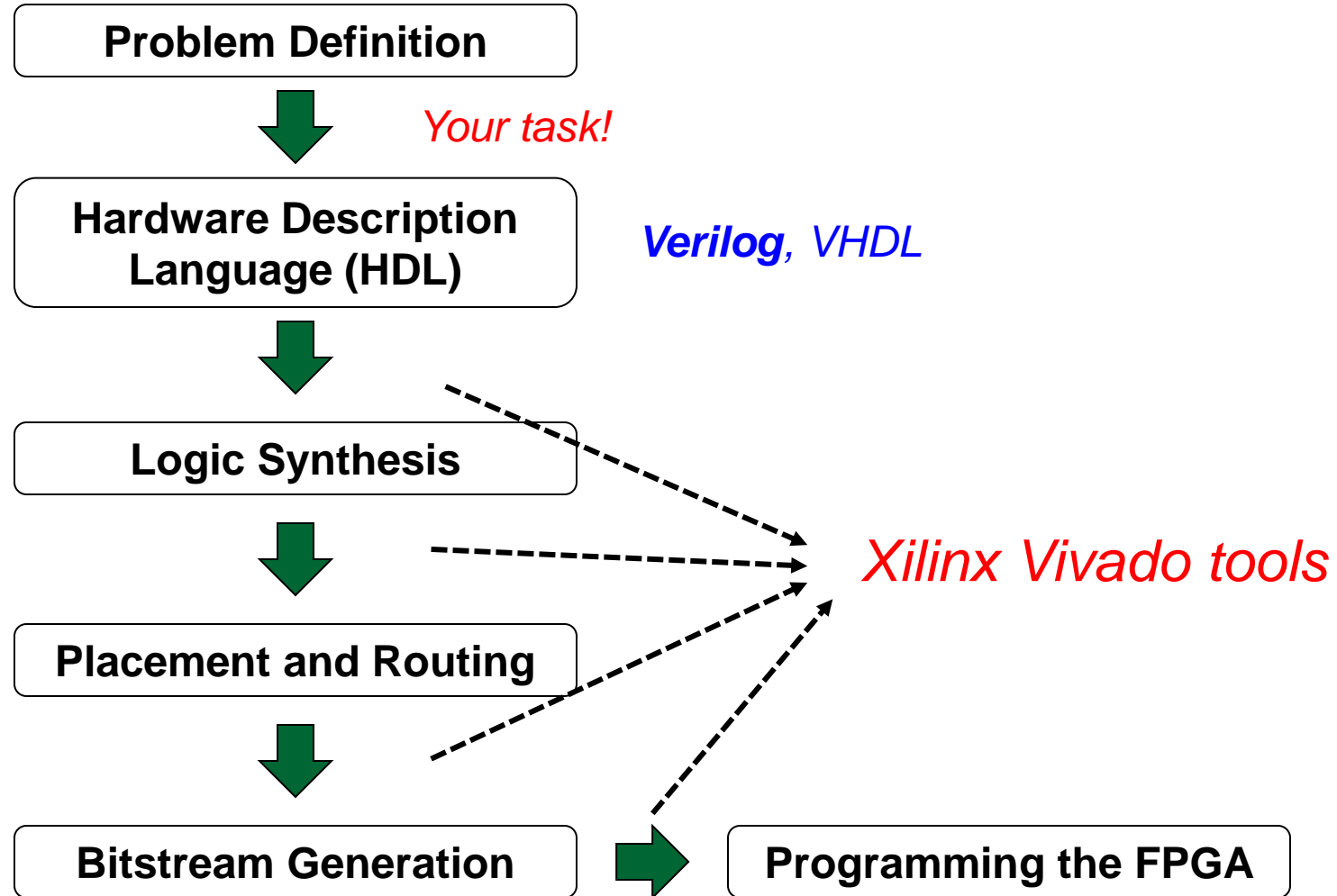
Agenda

- Logistics
- What Will We Learn?
- FPGAs in Today's Systems
- Overview of the Lab Exercises
- What is an FPGA?
- **Programming an FPGA**
- Tutorial and Demo

Computer-Aided Design (CAD) Tools

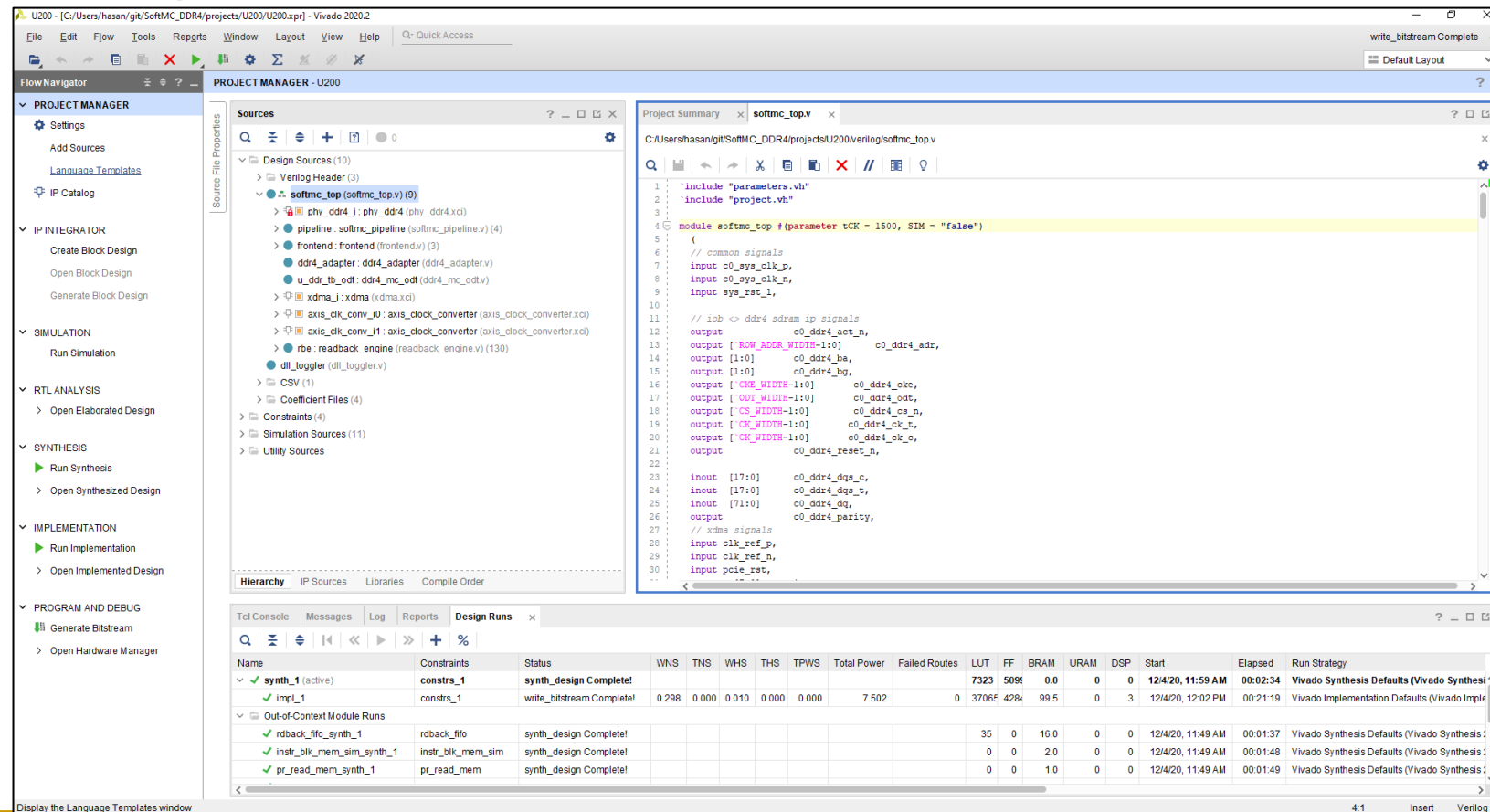
- FPGAs have many **resources** (e.g., LUTs, switches)
- They are **hard** to program manually
- How can we
 - represent a **high-level functional description** of our hardware circuit using the FPGA resources?
 - select the resources to **map** our circuit to?
 - **optimally configure the interconnect** between the selected resources?
 - generate a final **configuration file** to properly configure an FPGA?

FPGA Design Flow



Vivado

- A software that helps us throughout the FPGA design flow
- Provides tools to **simulate** our designs
 - ❑ Validate the correctness of the implementation
 - ❑ **Debugging**
- Provides drivers and graphical interface to easily **program** the FPGA using a USB cable
- ~~Installed in computer rooms in HG (E 19, E 26.1, E 26.3, E 27)~~



Tutorial and Demo

- We will see how to
 - use Vivado to write Verilog code
 - follow the FPGA design flow steps
 - download the bitstream into the FPGA
- Simple Keyboard Demo
 - An example for a simple hardware that you can easily develop by the end of semester

<https://reference.digilentinc.com/learn/programmable-logic/tutorials/basys-3-keyboard-demo/start>

Introduction to FPGAs: Tutorial and Demo

In this video, we will show how to program a **Max 3 FPGA board** using an example project that interfaces a **keyboard** and sends the **ASCII code of the pressed key** over a serial interface to a computer

<https://www.digilent.com/en/products/programmable-logic/tutorials/keyboard-demo>

Today We Covered:

- Logistics
- What Will We Learn?
- FPGAs in Today's Systems
- Overview of the Lab Exercises
- What is an FPGA?
- Programming an FPGA
- Tutorial and Demo

Digital Design & Computer Arch.

Lecture 3b: Introduction to the Labs and FPGAs

Prof. Onur Mutlu
(Lecture by Hasan Hassan)
ETH Zurich
Spring 2021
4 March 2021