TU Graz

SCIENCE
PASSION
TECHNOLOGY

Digital System Integration and Programming

Barbara Gigerl, Rishub Nagpal

October 5th, 2022

Outline

1. Digital system integration and programming

Outline

1. Digital system integration and programming
2. About this course

Outline

1. Digital system integration and programming
2. About this course
3. Outlook: Projects

What is digital system integration and programming?

Digital system integration

- Digital systems: very complex
- System integration: connect multiple complex systems to achieve a certain goal

What is digital system integration and programming?

Digital system integration

- Digital systems: very complex
- System integration: connect multiple complex systems to achieve a certain goal

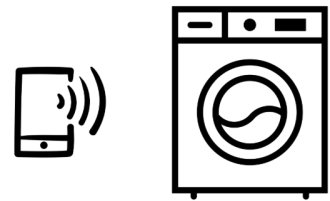
...and programming

- Hardware and Software

What is digital system integration and programming?

Digital system integration

- Digital systems: very complex
- System integration: connect multiple complex systems to achieve a certain goal



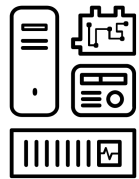
...and programming

- Hardware and Software

What is a System-on-a-Chip?

A **System-on-a-Chip (SoC)** is a complex system which:

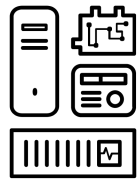
- consists of several components.



What is a System-on-a-Chip?

A **System-on-a-Chip (SoC)** is a complex system which:

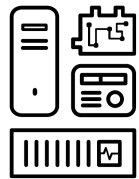
- consists of several components.
- Each component itself is a complex system.



What is a System-on-a-Chip?

A **System-on-a-Chip (SoC)** is a complex system which:

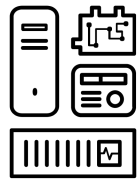
- consists of several components.
- Each component itself is a complex system.
  - CPU



What is a System-on-a-Chip?

A **System-on-a-Chip (SoC)** is a complex system which:

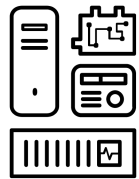
- consists of several components.
- Each component itself is a complex system.
  - CPU
  - Memory



What is a System-on-a-Chip?

A **System-on-a-Chip (SoC)** is a complex system which:

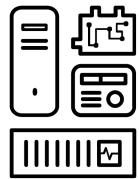
- consists of several components.
- Each component itself is a complex system.
  - CPU
  - Memory
  - Bus architectures



What is a System-on-a-Chip?

A **System-on-a-Chip (SoC)** is a complex system which:

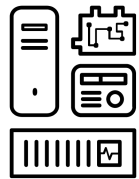
- consists of several components.
- Each component itself is a complex system.
  - CPU
  - Memory
  - Bus architectures
  - I/O modules



What is a System-on-a-Chip?

A **System-on-a-Chip (SoC)** is a complex system which:

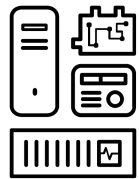
- consists of several components.
- Each component itself is a complex system.
  - CPU
  - Memory
  - Bus architectures
  - I/O modules
  - Co-processors



What is a System-on-a-Chip?

A **System-on-a-Chip (SoC)** is a complex system which:

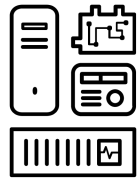
- consists of several components.
- Each component itself is a complex system.
  - CPU
  - Memory
  - Bus architectures
  - I/O modules
  - Co-processors
  - Analog circuits



What is a System-on-a-Chip?

A **System-on-a-Chip (SoC)** is a complex system which:

- consists of several components.
- Each component itself is a complex system.
  - CPU
  - Memory
  - Bus architectures
  - I/O modules
  - Co-processors
  - Analog circuits
  - ...



What is a System-on-a-Chip?

A quick history

- 1970s: VLSI design

What is a System-on-a-Chip?

A quick history

- 1970s: VLSI design
  - VLSI = Very large-scale integration

What is a System-on-a-Chip?

A quick history

- 1970s: VLSI design
  - VLSI = Very large-scale integration
  - Combining millions of MOS transistors into an integrated circuit

What is a System-on-a-Chip?

A quick history

- 1970s: VLSI design
  - VLSI = Very large-scale integration
  - Combining millions of MOS transistors into an integrated circuit
- 1990s: System-on-a-chip

What is a System-on-a-Chip?

A quick history

- 1970s: VLSI design
  - VLSI = Very large-scale integration
  - Combining millions of MOS transistors into an integrated circuit
- 1990s: System-on-a-chip
  - System integration: integration of a complete system, that until recently consisted of multiple ICs, onto a single IC (a SoC)

What is a System-on-a-Chip?

A quick history

- 1970s: VLSI design
  - VLSI = Very large-scale integration
  - Combining millions of MOS transistors into an integrated circuit
- 1990s: System-on-a-chip
  - System integration: integration of a complete system, that until recently consisted of multiple ICs, onto a single IC (a SoC)
- Today: SoC is the state-of-the-art principle for designing chips.

SoCs are everywhere

Smartphones

SoCs are everywhere

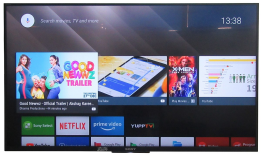
Smartphones

Tablets

SoCs are everywhere

Smartphones

Tablets

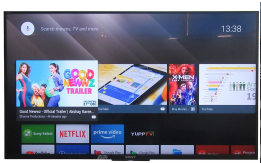


Smart TVs

SoCs are everywhere

Smartphones

Tablets



Smart TVs

Cars

Example: Apple A12 Bionic

- Used in iPhone XS, XS Max, XR

Example: Apple A12 Bionic

- Used in iPhone XS, XS Max, XR
- 7nm CMOS, 6.9 billion transistors

Example: Apple A12 Bionic

- Used in iPhone XS, XS Max, XR
- 7nm CMOS, 6.9 billion transistors
- Components:

Example: Apple A12 Bionic

- Used in iPhone XS, XS Max, XR
- 7nm CMOS, 6.9 billion transistors
- Components:
  - 64-bit ARMv8.3A (6 performance CPUs, 4 energy-efficient CPUs)

Example: Apple A12 Bionic

- Used in iPhone XS, XS Max, XR
- 7nm CMOS, 6.9 billion transistors
- Components:
  - 64-bit ARMv8.3A (6 performance CPUs, 4 energy-efficient CPUs)
  - Four-core GPU

Example: Apple A12 Bionic

- Used in iPhone XS, XS Max, XR
- 7nm CMOS, 6.9 billion transistors
- Components:
  - 64-bit ARMv8.3A (6 performance CPUs, 4 energy-efficient CPUs)
  - Four-core GPU
  - *Neural Engine* with 8 cores

Example: Apple A12 Bionic

- Used in iPhone XS, XS Max, XR
- 7nm CMOS, 6.9 billion transistors
- Components:
  - 64-bit ARMv8.3A (6 performance CPUs, 4 energy-efficient CPUs)
  - Four-core GPU
  - *Neural Engine* with 8 cores
  - ...

Example: Qualcomm Snapdragon 865

- Used in smartphones by ZTE, Sony, OnePlus, LG, ...

Example: Qualcomm Snapdragon 865

- Used in smartphones by ZTE, Sony, OnePlus, LG, ...
- 7nm CMOS

Example: Qualcomm Snapdragon 865

- Used in smartphones by ZTE, Sony, OnePlus, LG, ...
- 7nm CMOS
- Components:

Example: Qualcomm Snapdragon 865

- Used in smartphones by ZTE, Sony, OnePlus, LG, ...
- 7nm CMOS
- Components:
  - Several ARM Cortex-A77 and Cortex-A55-based CPUs

Example: Qualcomm Snapdragon 865

- Used in smartphones by ZTE, Sony, OnePlus, LG, ...
- 7nm CMOS
- Components:
  - Several ARM Cortex-A77 and Cortex-A55-based CPUs
  - Deidcated processor for ISP for photos and videos

Example: Qualcomm Snapdragon 865

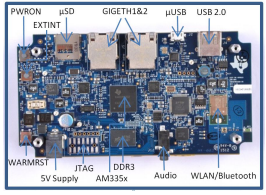
- Used in smartphones by ZTE, Sony, OnePlus, LG, ...
- 7nm CMOS
- Components:
  - Several ARM Cortex-A77 and Cortex-A55-based CPUs
  - Deidcated processor for ISP for photos and videos
  - Wi-Fi

Example: Qualcomm Snapdragon 865

- Used in smartphones by ZTE, Sony, OnePlus, LG, ...
- 7nm CMOS
- Components:
  - Several ARM Cortex-A77 and Cortex-A55-based CPUs
  - Deidcated processor for ISP for photos and videos
  - Wi-Fi
  - SPU: dedicated subsystem for boot-loader, key management unit, crypto accelerators, ...

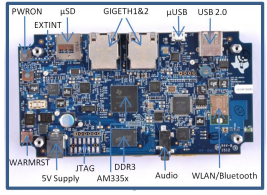
Example: Sitara Processors of TI

- SoC for industrial applications



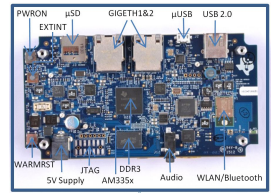
Example: Sitara Processors of TI

- SoC for industrial applications
- Used in thermostats, firewalls, Lego Mindstorms



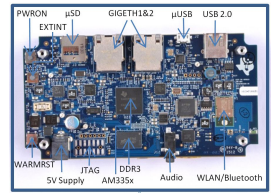
Example: Sitara Processors of TI

- SoC for industrial applications
- Used in thermostats, firewalls, Lego Mindstorms
- 1 GHz ARM CPU



Example: Sitara Processors of TI

- SoC for industrial applications
- Used in thermostats, firewalls, Lego Mindstorms
- 1 GHz ARM CPU
- On-chip quad-core PRU (Programmable Realtime Unit)



A Typical SoC

A **traditional SoC** consists of:

- Processor(s): mostly ARM cores

A Typical SoC

A **traditional SoC** consists of:

- Processor(s): mostly ARM cores
- GPU: depending on the field of application, ranging from simple cores for small LCDs to 4k screens

A Typical SoC

A **traditional SoC** consists of:

- Processor(s): mostly ARM cores
- GPU: depending on the field of application, ranging from simple cores for small LCDs to 4k screens
- Co-processors: for security, real-time signal processing, ...

A Typical SoC

A **traditional SoC** consists of:

- Processor(s): mostly ARM cores
- GPU: depending on the field of application, ranging from simple cores for small LCDs to 4k screens
- Co-processors: for security, real-time signal processing, ...
- I/O interfaces: Ethernet, SPI, USB, ADC, ...

A Typical SoC

A **traditional SoC** consists of:

- Processor(s): mostly ARM cores
- GPU: depending on the field of application, ranging from simple cores for small LCDs to 4k screens
- Co-processors: for security, real-time signal processing, ...
- I/O interfaces: Ethernet, SPI, USB, ADC, ...
- A bus connecting all components: AMBA, AXI, CoreConnect, ...

Why use SoCs?

Advantages

Why use SoCs?

Advantages

- Low silicon area

Disadvantages:

Why use SoCs?

Advantages

- Low silicon area
- Power efficiency (no need for complex component **wiring**)

Disadvantages:

Why use SoCs?

Advantages

- Low silicon area
- Power efficiency (no need for complex component **wiring**)
- Low manufacturing costs

Disadvantages:

Why use SoCs?

Advantages

- Low silicon area
- Power efficiency (no need for complex component **wiring**)
- Low manufacturing costs
- Smaller power supply unit

Disadvantages:

Why use SoCs?

Advantages

- Low silicon area
- Power efficiency (no need for complex component **wiring**)
- Low manufacturing costs
- Smaller power supply unit

Disadvantages:

Why use SoCs?

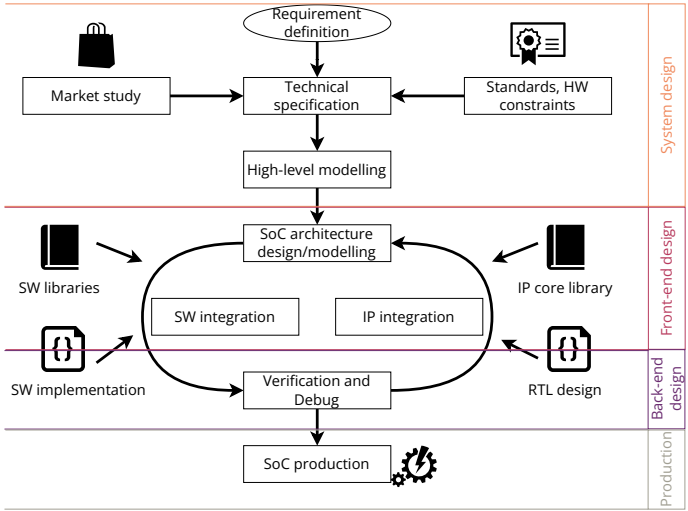
Advantages

- Low silicon area
- Power efficiency (no need for complex component **wiring**)
- Low manufacturing costs
- Smaller power supply unit

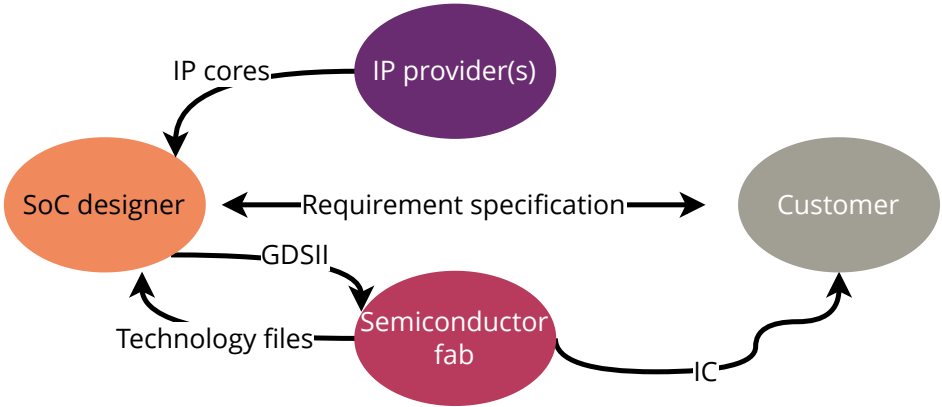
Disadvantages:

- Resulting system is very complex
- High design and development costs

SoC Design Methodology



SoC Players



- GDSII: data format to describe ICs
- Technology file: information about manufacturing (metals, IC layers, ...)

Who are we?

Barbara Gigerl

PhD student @ Graz University of Technology

Formal Verification of Side-Channel Protected Implementations

barbara.gigerl@iaik.tugraz.at

sip-team@iaik.tugraz.at

Who are we?

Rishub Nagpal

PhD student @ Graz University of Technology

Power side-channel attacks and defenses for cryptographic implementations

rishub.nagpal@lamarr.at

sip-team@iaik.tugraz.at

Topics for Master Thesis

Looking for a master thesis?
→ https://www.iaik.tugraz.at/teaching/master-thesis/
We have lots of interesting open topics :)

Alternatively, email us:

- barbara.gigerl@iaik.tugraz.at
- rishub.nagpal@lamarr.at

Contact

- General information:
  https://www.iaik.tugraz.at/sip
- Questions and concerns by E-Mail
  mailto:sip-team@iaik.tugraz.at
- Questions and concerns via Discord
  https://discord.gg/9KKGfndsD5

We build our own SoC

- We focus on the front-end design

We build our own SoC

- We focus on the front-end design
- We use an FPGA in order to build a prototype of our SoC

We build our own SoC

- We focus on the front-end design
- We use an FPGA in order to build a prototype of our SoC
- Our Platform: Zybo Zynq Boards

We build our own SoC

- We focus on the front-end design
- We use an FPGA in order to build a prototype of our SoC
- Our Platform: Zybo Zynq Boards
  - Xilinx FPGA

We build our own SoC

- We focus on the front-end design
- We use an FPGA in order to build a prototype of our SoC
- Our Platform: Zybo Zynq Boards
  - Xilinx FPGA
  - 650Mhz dual-core Cortex-A9 processor

We build our own SoC

- We focus on the front-end design
- We use an FPGA in order to build a prototype of our SoC
- Our Platform: Zybo Zynq Boards
  - Xilinx FPGA
  - 650Mhz dual-core Cortex-A9 processor
  - HDMI, VGA, USB, SPI, Ethernet, Audio, ...

We build our own SoC

- We focus on the front-end design
- We use an FPGA in order to build a prototype of our SoC
- Our Platform: Zybo Zynq Boards
  - Xilinx FPGA
  - 650Mhz dual-core Cortex-A9 processor
  - HDMI, VGA, USB, SPI, Ethernet, Audio, ...
  - Connected via AXI bus

Goals

- Build a working prototype

Goals

- Build a working prototype
- Project management and self-organization

Goals

- Build a working prototype
- Project management and self-organization
- Presentation of: ideas, results, technology in English

Goals

- Build a working prototype
- Project management and self-organization
- Presentation of: ideas, results, technology in English
- Preparation for project/thesis

Required Previous Knowledge

SIP addresses **advanced-level students**. **You need:**

- Knowledge about hardware including a HDL (Verilog/VHDL)
  (Computer Organization and Networks is probably not enough)

Required Previous Knowledge

SIP addresses **advanced-level students**. **You need:**

- Knowledge about hardware including a HDL (Verilog/VHDL)
  (Computer Organization and Networks is probably not enough)
- Very good C/C++ skills

Required Previous Knowledge

SIP addresses **advanced-level students**. **You need:**

- Knowledge about hardware including a HDL (Verilog/VHDL)
  (Computer Organization and Networks is probably not enough)
- Very good C/C++ skills
- Knowledge about Linux

Required Previous Knowledge

SIP addresses **advanced-level students**. **You need:**

- Knowledge about hardware including a HDL (Verilog/VHDL)
  (Computer Organization and Networks is probably not enough)
- Very good C/C++ skills
- Knowledge about Linux
  - Buildroot/Yocto, kernel modules, drivers, device trees, GPIO

Required Previous Knowledge

SIP addresses **advanced-level students**. **You need:**

- Knowledge about hardware including a HDL (Verilog/VHDL)
  (Computer Organization and Networks is probably not enough)
- Very good C/C++ skills
- Knowledge about Linux
  - Buildroot/Yocto, kernel modules, drivers, device trees, GPIO
- Knowledge about FPGAs, bus protocols, CPUs, networks

Required Previous Knowledge

SIP addresses **advanced-level students**. **You need:**

- Knowledge about hardware including a HDL (Verilog/VHDL)
  (Computer Organization and Networks is probably not enough)
- Very good C/C++ skills
- Knowledge about Linux
  - Buildroot/Yocto, kernel modules, drivers, device trees, GPIO
- Knowledge about FPGAs, bus protocols, CPUs, networks
- Very good time-management skills

Required Previous Knowledge

SIP addresses **advanced-level students**. **You need:**

- Knowledge about hardware including a HDL (Verilog/VHDL)
  (Computer Organization and Networks is probably not enough)
- Very good C/C++ skills
- Knowledge about Linux
  - Buildroot/Yocto, kernel modules, drivers, device trees, GPIO
- Knowledge about FPGAs, bus protocols, CPUs, networks
- Very good time-management skills
- Good presentation skills

Teaching method

We offer:

- Project driven work (group-oriented, project-centric)

Teaching method

We offer:

- Project driven work (group-oriented, project-centric)
- Hands-on project with real hardware

Teaching method

We offer:

- Project driven work (group-oriented, project-centric)
- Hands-on project with real hardware
- Upgrading of soft skills

Teaching method

We offer:

- Project driven work (group-oriented, project-centric)
- Hands-on project with real hardware
- Upgrading of soft skills
  - Presentations

Teaching method

We offer:

- Project driven work (group-oriented, project-centric)
- Hands-on project with real hardware
- Upgrading of soft skills
  - Presentations
  - Speaking English

Teaching method

We offer:

- Project driven work (group-oriented, project-centric)
- Hands-on project with real hardware
- Upgrading of soft skills
  - Presentations
  - Speaking English
  - Group communication

Teaching method

We expect:

- Investment of time

Teaching method

We expect:

- Investment of time
  - SIP: 3 VU (5 ECTS)

Teaching method

We expect:

- Investment of time
  - SIP: 3 VU (5 ECTS)
  - $5 \times 25 = 125$ hours work = 28 days of 8 hours

Teaching method

We expect:

- Investment of time
  - SIP: 3 VU (5 ECTS)
  - $5 \times 25 = 125$ hours work = 28 days of 8 hours
- Active communication within your group

Teaching method

We expect:

- Investment of time
  - SIP: 3 VU (5 ECTS)
  - $5 \times 25 = 125$ hours work = 28 days of 8 hours
- Active communication within your group
- Active participation, presence during lectures

Grading

Your grade consists of:

- Project 1: 20%

Grading

Your grade consists of:

- Project 1: 20%
  - Individual work, independent submissions

Grading

Your grade consists of:

- Project 1: 20%
  - Individual work, independent submissions
- Project 2: 50%

Grading

Your grade consists of:

- Project 1: 20%
  - Individual work, independent submissions
- Project 2: 50%
  - Team work in groups of 2 or 3 students

Grading

Your grade consists of:

- Project 1: 20%
  - Individual work, independent submissions
- Project 2: 50%
  - Team work in groups of 2 or 3 students
- Seminar presentation: 30%

Grading

Your grade consists of:

- Project 1: 20%
  - Individual work, independent submissions
- Project 2: 50%
  - Team work in groups of 2 or 3 students
- Seminar presentation: 30%
  - Selection from course catalog OR suggest your own topic

Grading

Your grade consists of:

- Project 1: 20%
  - Individual work, independent submissions
- Project 2: 50%
  - Team work in groups of 2 or 3 students
- Seminar presentation: 30%
  - Selection from course catalog OR suggest your own topic
  - Slides are reviewed by us (submit until Monday evening)

Grading

Your grade consists of:

- Project 1: 20%
  - Individual work, independent submissions
- Project 2: 50%
  - Team work in groups of 2 or 3 students
- Seminar presentation: 30%
  - Selection from course catalog OR suggest your own topic
  - Slides are reviewed by us (submit until Monday evening)
- Bonus points for questions during/after seminar presentations

Team work

- Team Size for Project 1: 1
- Team Size for Project 2: 3
- Team Size for Seminar presentation: 1

Registration Process

1. Find a group

Deadline: Monday, 10.10., 23:59

[1] You can try your luck before/after

Registration Process

1. Find a group
2. Find and register your group: mailto:sip-team@iaik.tugraz.at

Deadline: Monday, 10.10., 23:59

[1] You can try your luck before/after

Registration Process

1. Find a group
2. Find and register your group: mailto:sip-team@iaik.tugraz.at
3. Wait for the confirmation mail to get your group number

Deadline: Monday, 10.10., 23:59

[1] You can try your luck before/after

Registration Process

1. Find a group
2. Find and register your group: mailto:sip-team@iaik.tugraz.at
3. Wait for the confirmation mail to get your group number
4. Decide when to pick up the hardware (IF01052, Mo-Fr 10:00-16:00[1])

Deadline: Monday, 10.10., 23:59

[1] You can try your luck before/after

Registration Process

1. Find a group
2. Find and register your group: mailto:sip-team@iaik.tugraz.at
3. Wait for the confirmation mail to get your group number
4. Decide when to pick up the hardware (IF01052, Mo-Fr 10:00-16:00[1])
5. Choose a seminar topic

Deadline: Monday, 10.10., 23:59

[1] You can try your luck before/after

Registration Process

1. Find a group
2. Find and register your group: mailto:sip-team@iaik.tugraz.at
3. Wait for the confirmation mail to get your group number
4. Decide when to pick up the hardware (IF01052, Mo-Fr 10:00-16:00[1])
5. Choose a seminar topic
6. Register for a seminar topic: https://bit.ly/3dZjipp

Deadline: Monday, 10.10., 23:59

[1] You can try your luck before/after

Registration Process

1. Find a group
2. Find and register your group: mailto:sip-team@iaik.tugraz.at
3. Wait for the confirmation mail to get your group number
4. Decide when to pick up the hardware (IF01052, Mo-Fr 10:00-16:00[1])
5. Choose a seminar topic
6. Register for a seminar topic: https://bit.ly/3dZjipp
7. Receive your git repositories (by email)

Deadline: Monday, 10.10., 23:59

[1] You can try your luck before/after

Schedule of SIP 2022

Project meetings

- Regular weekly meetings: Wednesday 10:00 - 12:00, IFEG042

Schedule of SIP 2022

Project meetings

- Regular weekly meetings: Wednesday 10:00 - 12:00, IFEG042
- Program for each project meeting

Schedule of SIP 2022

Project meetings

- Regular weekly meetings: Wednesday 10:00 - 12:00, IFEG042
- Program for each project meeting
    1. Questions, problems about the project

Schedule of SIP 2022

Project meetings

- Regular weekly meetings: Wednesday 10:00 - 12:00, IFEG042
- Program for each project meeting
  1. Questions, problems about the project
  2. Each team briefly (1-2 sentences) comments on own project progress

Schedule of SIP 2022

Project meetings

- Regular weekly meetings: Wednesday 10:00 - 12:00, IFEG042
- Program for each project meeting
  1. Questions, problems about the project
  2. Each team briefly (1-2 sentences) comments on own project progress
  3. Seminar talk + discussion

Schedule of SIP 2022

Project meetings

- Regular weekly meetings: Wednesday 10:00 - 12:00, IFEG042
- Program for each project meeting
  1. Questions, problems about the project
  2. Each team briefly (1-2 sentences) comments on own project progress
  3. Seminar talk + discussion
  4. Seminar talk + discussion

Schedule of SIP 2022

Project meetings

- Regular weekly meetings: Wednesday 10:00 - 12:00, IFEG042
- Program for each project meeting

1. Questions, problems about the project
2. Each team briefly (1-2 sentences) comments on own project progress
3. Seminar talk + discussion
4. Seminar talk + discussion
5. ...

Preliminary timeline

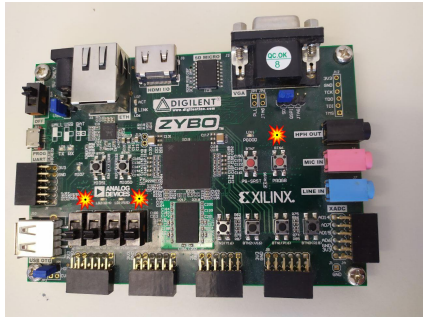
Date	Topic
05.10.	Kick-off / Introduction to Seminar Topics / SoC Design Flow Tutorial
12.10.	Embedded Linux Tutorial / Presentation Project 1
19.10.	Debugging Tutorial
26.10.	Bank holiday (no meeting)
2.11.	Bank holiday (no meeting)
9.11.	Presentation Project 2a+2b / Seminar talks
16.11.	Seminar talks
23.11.	Seminar talks
30.11.	Seminar talks
7.12.	Seminar talks
14.12.	Seminar talks
11.01.	Seminar talks
18.01.	Seminar talks
25.01.	Seminar talks

Important Dates and Deadlines

Date	Topic
10.10., 23:59	Deadline Group Registration
8.11., 23:59	Deadline Project 1
9.11.-16.11.	Exercise Interviews Project 1
6.12., 23:59	Deadline Project 2a
7.12.-13.12.	Exercise Interviews Project 2a
17.01., 23:59	Deadline Project 2b
18.01.-24.01.	Exercise Interviews Project 2a

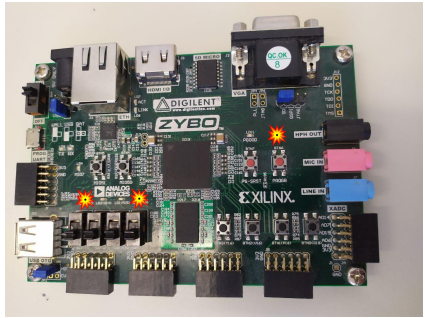
Project 1: Fancy Lights

- Get to know the board and run through all steps



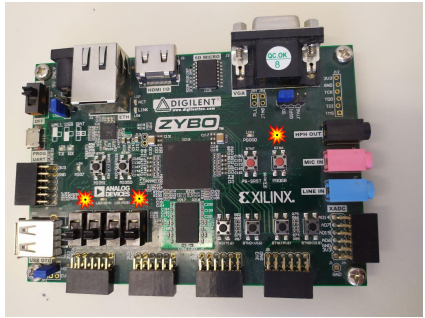
Project 1: Fancy Lights

- Get to know the board and run through all steps
- Design hardware, build a driver, write an application



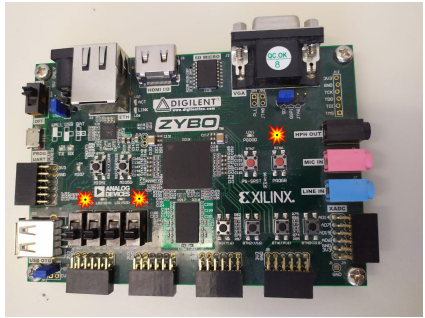
Project 1: Fancy Lights

- Get to know the board and run through all steps
- Design hardware, build a driver, write an application
- Access the LEDs from a bare-metal application and from within Linux



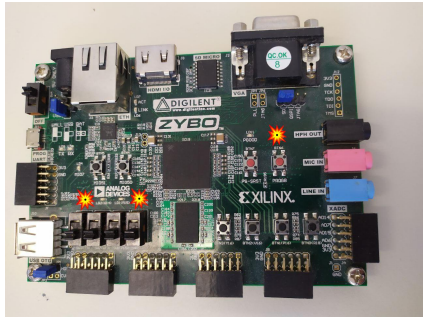
Project 1: Fancy Lights

- Get to know the board and run through all steps
- Design hardware, build a driver, write an application
- Access the LEDs from a bare-metal application and from within Linux
- No team work; everybody should do all steps (share your board within group)



Project 1: Fancy Lights

- Get to know the board and run through all steps
- Design hardware, build a driver, write an application
- Access the LEDs from a bare-metal application and from within Linux
- No team work; everybody should do all steps (share your board within group)
- Aim: After completing, everybody should have the same basic knowledge.



Project 2: Display for Encrypted Images

- Use knowledge from Project 1 to build larger system
- Receive encrypted image via Ethernet, decrypt it in hardware and display it via HDMI
- Team work
- Aim: Get some deeper understanding of the topic