# Cyber-Physical Systems

## Course Syllabus

Amirkabir University of Technology
Department of Electrical Engineering

courses.aut.ac.ir


### Instructor

Dr. Ahmad Shabani
Assistant Professor
ah.shabani@ut.ac.ir
ahmadshabani89@gmail.com


### Class Information

Saturday–Monday
7:45–9:15
Room #410
TAs: Will be introduced

### Course Theme

This course studies Cyber-Physical and embedded computing systems from the architectural, software, and system-design perspectives. The course connects modern processor architecture, embedded platforms, hardware acceleration, memory systems, communication interfaces, sensors, real-time operating systems, and FPGA-based embedded design.

## Course Description

Cyber-Physical Systems combine computation, communication, control, sensing, and physical interaction. This course introduces the foundations required to design and understand modern embedded and cyber-physical platforms. The course begins with embedded system trends, system-on-chip design, technology scaling, and dark/dim silicon concepts. It then moves into modern processor architecture, including RISC-V datapaths, pipelining, hazards, superscalar execution, out-of-order execution, and VLIW processors.

The second part of the course focuses on data-level parallelism, vector processors, SIMD extensions, GPUs, and CUDA programming. The course then introduces application-specific instruction-set processors, ASIP design methodology, Transport-Triggered Architecture, ESP32-based development, memory hierarchy, sensors, communication interfaces, embedded C building process, RTOS concepts, FreeRTOS, synchronization mechanisms, and FPGA-based embedded system design.

## Learning Outcomes

By the end of the course, students should be able to:

- Explain the structure and characteristics of embedded and cyber-physical systems.
- Understand system-on-chip design and embedded processor trends.
- Analyze the impact of technology scaling, dark silicon, dim silicon, power walls, and bandwidth walls.
- Explain RISC-V ISA concepts and single-cycle RISC-V architecture.
- Analyze pipelined RISC-V processors, hazards, forwarding, and stalling.
- Understand superscalar, out-of-order, EPIC, and VLIW processor concepts.
- Explain vector processors, SIMD extensions, GPUs, and CUDA-C programming.
- Describe ASIP design flow and Transport-Triggered Architecture processors.
- Work with ESP32-based embedded development tools and bare-metal programming concepts.
- Explain memory hierarchy, DRAM, cache architecture, cache coherence, and flash memory.
- Understand sensors, actuators, and communication interfaces such as I2C, SPI, CAN, WiFi, BLE, ZigBee, and RS-485.
- Describe embedded C building process, memory layout, RTOS concepts, FreeRTOS, tasks, queues, mutexes, and semaphores.
- Understand the basics of FPGA-based embedded design using ZYNQ, VIVADO, Vitis, and PYNQ.

## › Course Modules

| **Module** | Topics |
|---|---|
| **Chapter Zero** | Introduction to embedded system design; embedded system trends, components, and characteristics; system-on-chip concepts; technology scaling; dark and dim silicon; power and bandwidth walls. |
| **Chapter One** | Instruction-level parallelism in modern processors; review of single-cycle RV32I RISC-V processor and datapath; single pipelined RISC-V datapath; hazards; stalling and forwarding; superscalar RISC-V; in-order and out-of-order execution; EPIC and VLIW processors. |
| **Chapter Two** | Data-level parallelism in modern processors; array and vector processors; SIMD instruction-set extensions; GPUs and execution model; CUDA-C programming. |
| **Chapter Three** | Application-specific instruction-set processors; ASIP design flow and methodology; hardware accelerator design; Transport-Triggered Architecture processors; TCE tool; Tensilica Xtensa LX6 ISA; ESP32 SOC; embedded development tools. |
| **Chapter Four** | Memory, sensors, and communication interfaces; memory hierarchy; DRAM architecture; cache architecture and coherency; flash architecture; sensors and actuators; on-board interfaces including I2C, SPI, and CAN; external interfaces including WiFi, BLE, ZigBee, and RS-485; MQTT and client-server communication. |
| **Chapter Five** | Compilers and RTOS in embedded systems; C program building process; memory layout of C programs; RTOS fundamentals; FreeRTOS on ESP32; tasks, threads, scheduling, synchronization, memory management, queues, mutexes, and semaphores. |
| **Chapter Six** | FPGA-based embedded system design; ZYNQ-based embedded systems; AXI, AXI-Lite, and AXI-Stream; VIVADO block design; Vitis platform; PYNQ; bare-metal and Linux-based development for ZYNQ devices. |

**Note:** Some advanced topics, especially Chapter Six and selected parts of Chapter Five, may be used as further reading depending on the semester schedule and available course time.

## Tentative Course Schedule

| L | Description | Category | CA |
|---|---|---|---|
| **1** | Introduction to embedded systems, features, and characteristics; introduction to System-on-Chip in embedded systems. | Computer Organization & Architecture | |
| **2** | Technology scaling and its effect on modern processors; dark and dim silicon; power and bandwidth walls. | Computer Organization & Architecture | |
| **3** | RISC-V instruction-set architecture; single-cycle RISC-V architecture. | Computer Organization & Architecture | |
| **4–5** | Single-pipeline RISC-V processors; dependency and hazards including data, branch, and structural hazards; forwarding and stalling; static multi-issue RISC-V; superscalar processors; branch prediction. | Computer Organization & Architecture | |
| **6** | Out-of-order execution in modern processors; dynamic renaming using Tomasulo's algorithm; exception handling and in-order commit. | Computer Organization & Architecture | CA1 |
| **7** | Introduction to Very-Long-Instruction-Word processors. | High Performance Computing | Midterm |
| **8** | Introduction to vector processing and SIMD extensions with RISC-V edition. | High Performance Computing | |
| **9** | Introduction to graphical processing units in embedded systems. | High Performance Computing | |
| **10** | Introduction to CUDA-C and parallel programming. | High Performance Computing | CA2 |
| **11** | ASIP design flow and methodology; source profiling and hardware accelerator design. | ASIP | |
| **12** | Introduction to Transport-Triggered Architecture processor as a basic platform for ASIP design. | ASIP | |
| **13** | Introduction to TCE, the TTA-based Co-Design Environment tool for ASIP design. | ASIP | CA3 |
| **14** | Introduction to Tensilica Xtensa LX6 ISA; introduction to ESP32 as a low-cost dual-core Xtensa-based SOC. | ASIP / ESP32 | |
| **15** | Embedded development tools including Arduino, PlatformIO, and ESP-IDF; bare-metal ESP32 programming and super-loop programming. | ESP32 | |
| **16** | Introduction to memory system and hierarchy; DRAM cell and architecture. | Memory | |
| **17** | Cache architecture and cache coherency. | Memory | |
| **18** | Flash architecture including NAND, NOR, and SPIFFS flashes. | Memory | |
| **19** | Sensors and actuators for embedded applications; on-board communication interfaces including I2C, SPI, and CAN; connecting sensors, OLED, LCD, and peripherals to ESP32. | Peripherals | |
| **20** | External communication interfaces including WiFi, BLE, ZigBee, and RS-485; WiFi access-point and client setup in IoT applications; BLE setup in energy-limited ESP32 applications. | Interfaces | CA4 |
| **21** | Client-server communication using ESP32; communicating ESP32 with remote servers. | Interfaces | |
| **22** | MQTT protocol in embedded systems. | Interfaces | |
| **23** | C programming building process including preprocessor, compiler, assembler, and linker. | Building Process | |
| **24** | Memory layout of C programs including stack, heap, data, and BSS. | Building Process | |
| **25** | Introduction to real-time operating systems. | RTOS | |

## Assessment and Grading Policy

| Component | Weight |
|---|---|
| Mid-term Exam | 30% |
| Final Exam | 45% |
| Computer Assignments and Homeworks | 20% |
| Presence and Contribution in Class Activity | 5% |

**Assessment Philosophy**

The course evaluation combines theoretical understanding, design-oriented problem solving, practical assignments, and active participation. Computer assignments and homeworks are intended to connect architectural concepts with embedded and cyber-physical system design.

## Penalty and Submission Policy

- Homework and computer assignment deadlines are firm.
- Late submissions receive a penalty of 3% per day.
- The maximum late penalty is 35%.
- Submissions more than one week late will not be accepted.
- Attendance is mandatory.
- Contribution in class activities will be considered as part of the final grade.

## Textbooks and References

### Main Course Materials

- The notes of the course.
- David A. Patterson and John L. Hennessy, *Computer Organization and Design: The Hardware/Software Interface*, RISC-V Edition, 2021.
- Sarah Harris and David Harris, *Digital Design and Computer Architecture*, RISC-V Edition, 2021.
- Lecture notes of Prof. James C. Hoe, *Computer Architecture*, Carnegie Mellon University, Spring 2022.

### Parallelism, GPUs, and CUDA

- Lecture notes of Prof. Onur Mutlu, *Computer Architecture*, Carnegie Mellon University, Vector and Array Processors, Spring 2018.
- Lecture notes of Prof. Onur Mutlu, *Computer Architecture*, ETH Zürich, SIMD Processors and GPUs, Fall 2021.
- Jason Sanders and Edward Kandrot, *CUDA by Example: An Introduction to General-Purpose GPU Programming*, 2011.

### ASIP and Embedded Processors

- Tilman Glökler and Heinrich Meyr, *Design of Energy-Efficient Application-Specific Instruction-Set Processors*, 2007, Chapters 4 and 5.
- H. Corporaal and P. van der Arend, "MOVE32INT, a Sea of Gates realization of a high performance Transport Triggered Architecture," *Microprocessing and Microprogramming*, 1993.
- Henk Corporaal, "Design of Transport Triggered Architectures," Proceedings of the 4th Great Lakes Symposium on VLSI, IEEE, 1994.
- Application-Specific Instruction-Set Tool Set: openasip.org.

### Memory, Communication, and Embedded Systems

- Lecture notes of Dr. Mehdi Modarressi, University of Tehran, Memory Systems, DRAM Part.
- Dawoud Shenouda and Peter Dawoud, *Microcontroller and Smart Home Networks*, River Publishers, 2020.
- *A Primer on Memory Consistency and Cache Coherence*, Synthesis Lectures, 2011.
- Tianhong Pan and Yi Zhu, *Designing Embedded Systems with Arduino: A Fundamental Technology for Makers*, 2017.
- Dake Liu, *Embedded DSP Processor Design: Application Specific Instruction Set Processors*, Elsevier, 2008.
- David E. Simon, *An Embedded Software Primer*, Addison-Wesley Professional, 1999.

## Platforms, Tools, and Practical Skills

**RISC-V** **CUDA-C** **PlatformIO** **ESP-IDF** **WiFi** **BLE** **MQTT** **ZYNQ** **GPU Computing** **ASIP** **TTA** **FreeRTOS** **I2C** **SPI** **CAN** **VIVADO** **Vitis** **PYNQ** **TCE** **ESP32** **Arduino**

## Course Policies

- Students are expected to attend classes regularly.
- Active participation in discussions, problem solving, and class activities is encouraged.
- All assignments must be submitted before the announced deadline.
- Academic honesty is required in exams, homework, computer assignments, and reports.
- Any update to the schedule, references, or assignments will be announced through the official course page.