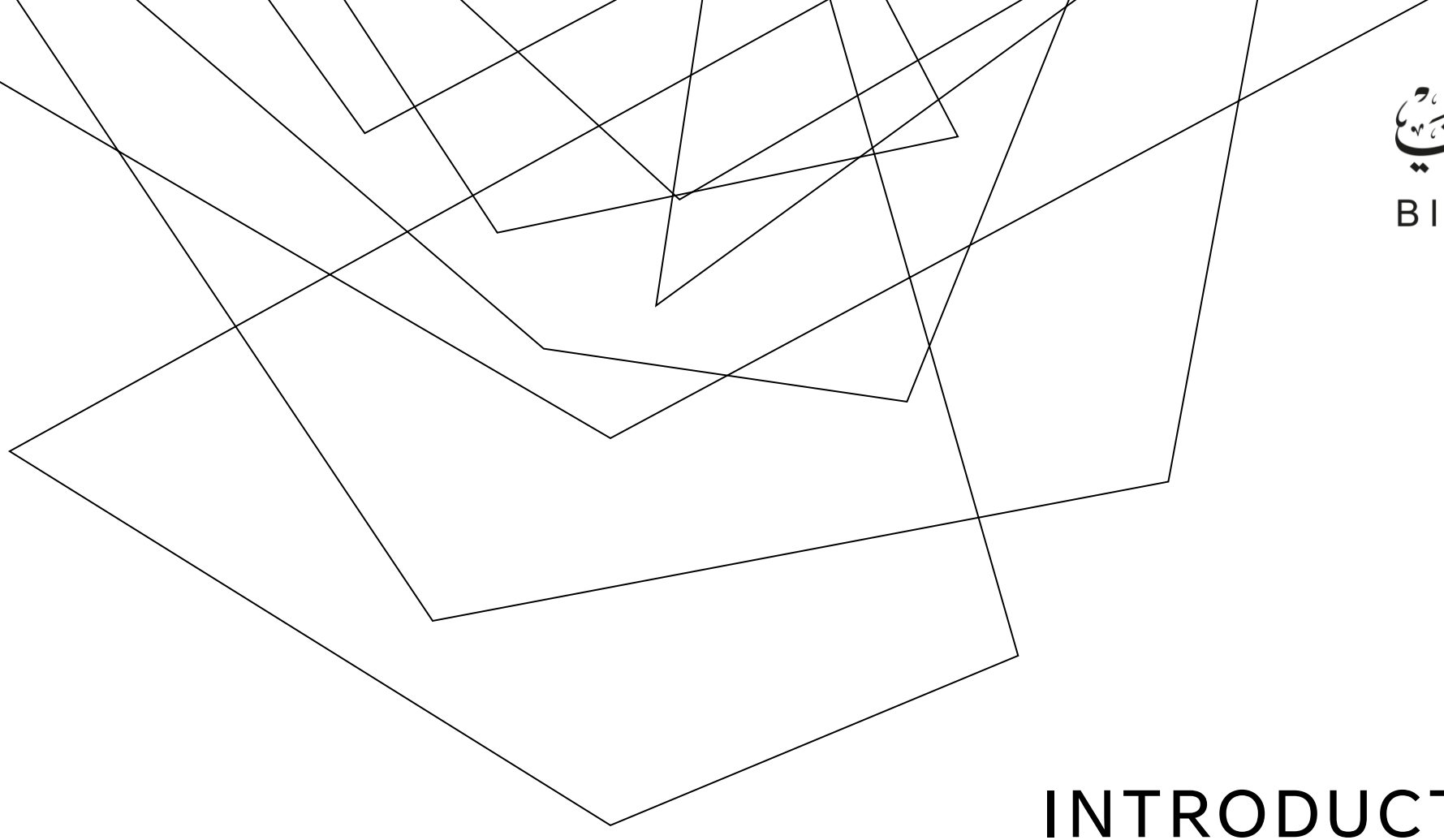


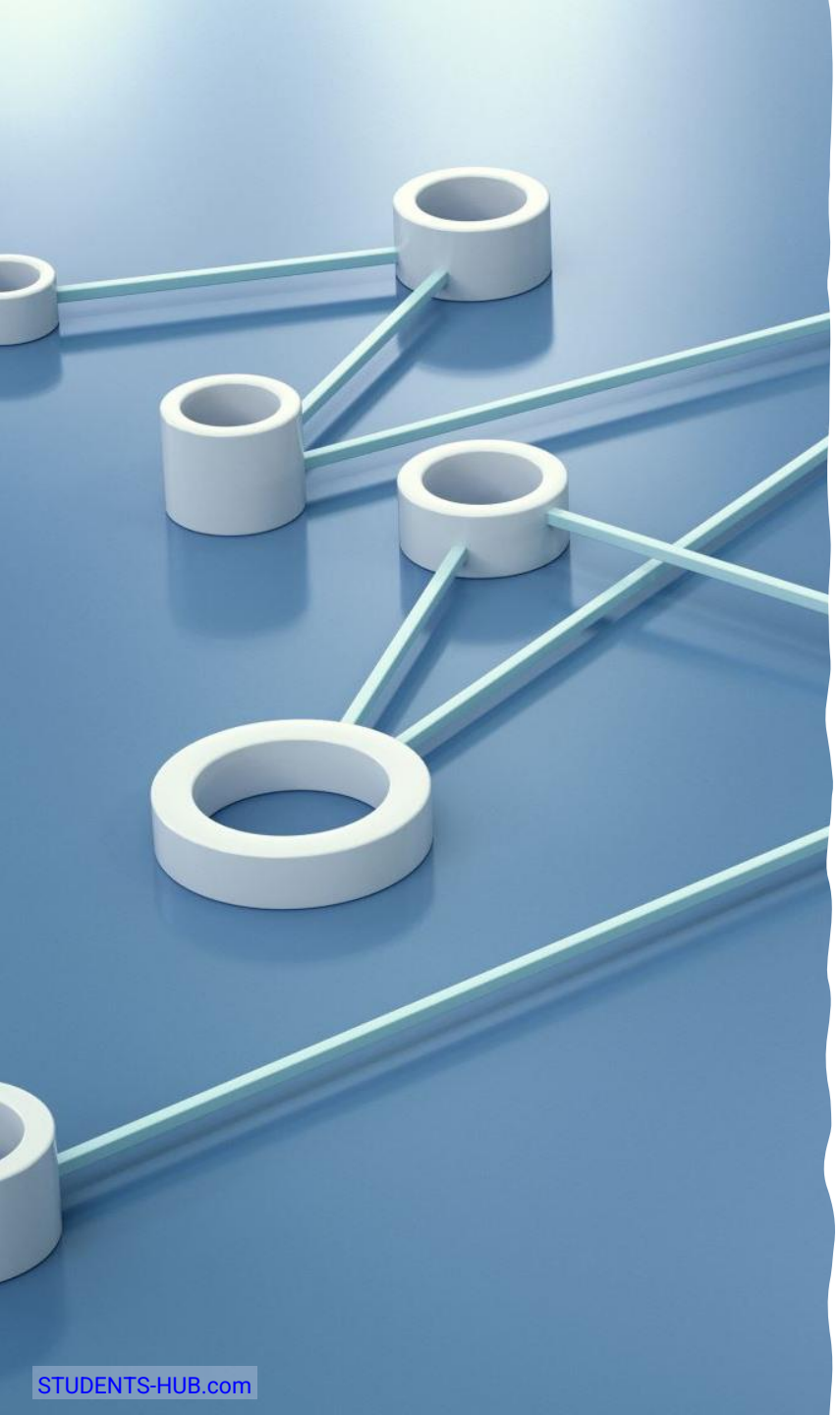
جامعة بيرزيت
BIRZEIT UNIVERSITY

INTRODUCTION TO EMBEDDED SYSTEMS

Shadi Daana

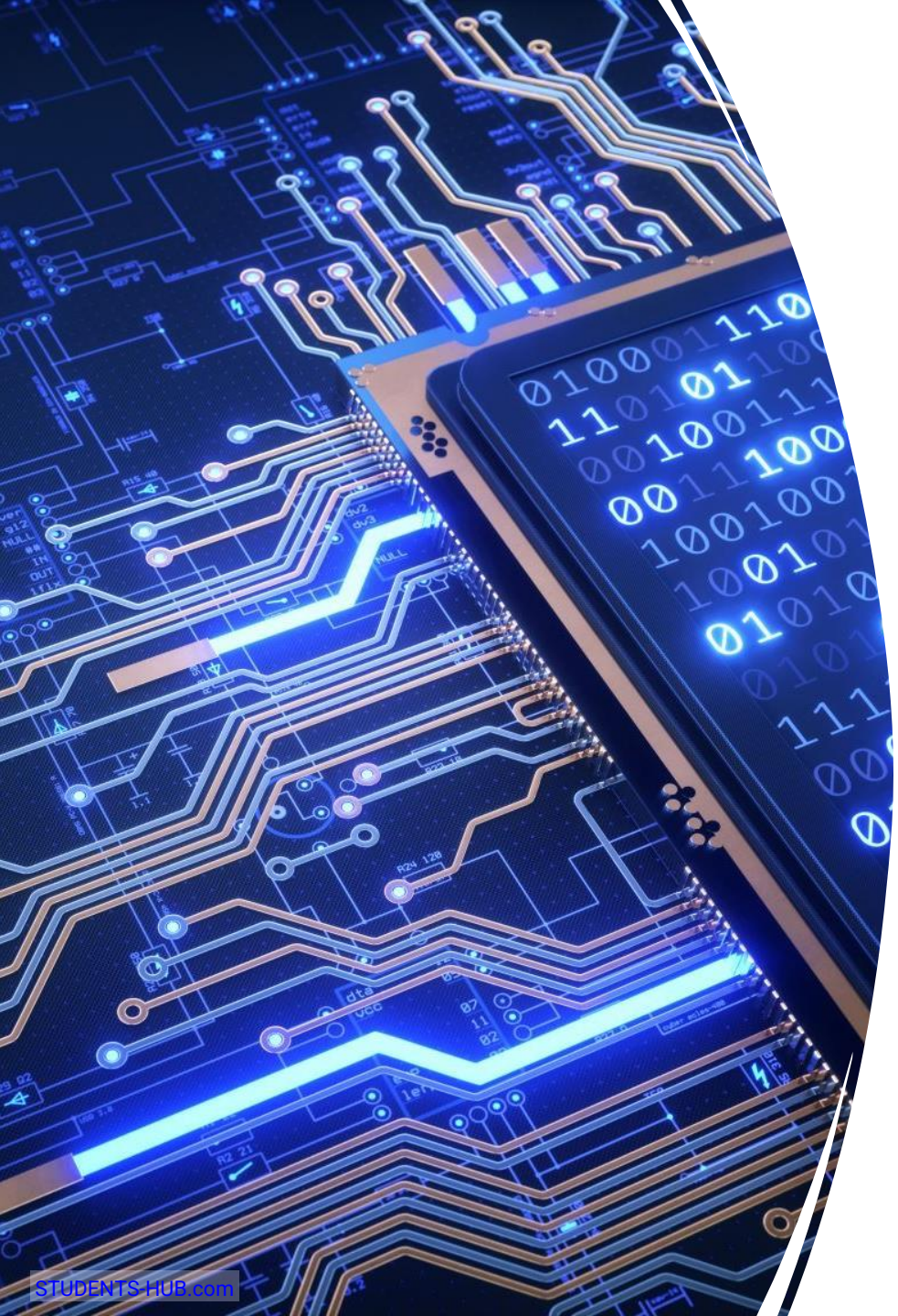
Course Objectives

- **Understand Embedded Systems**: Gain a comprehensive understanding of what embedded systems are, their characteristics, and their importance in various industries.
- **Components and Architecture**: Explore the key components of embedded systems, including microcontrollers, memory, input/output devices, and their roles in system architecture.
- **Design and Development**: Learn the embedded system design process, encompassing requirements analysis, hardware and software design, and testing and validation.
- **Programming Skills**: Acquire proficiency in programming embedded systems using C language and assembly, and become familiar with development tools and debugging techniques.



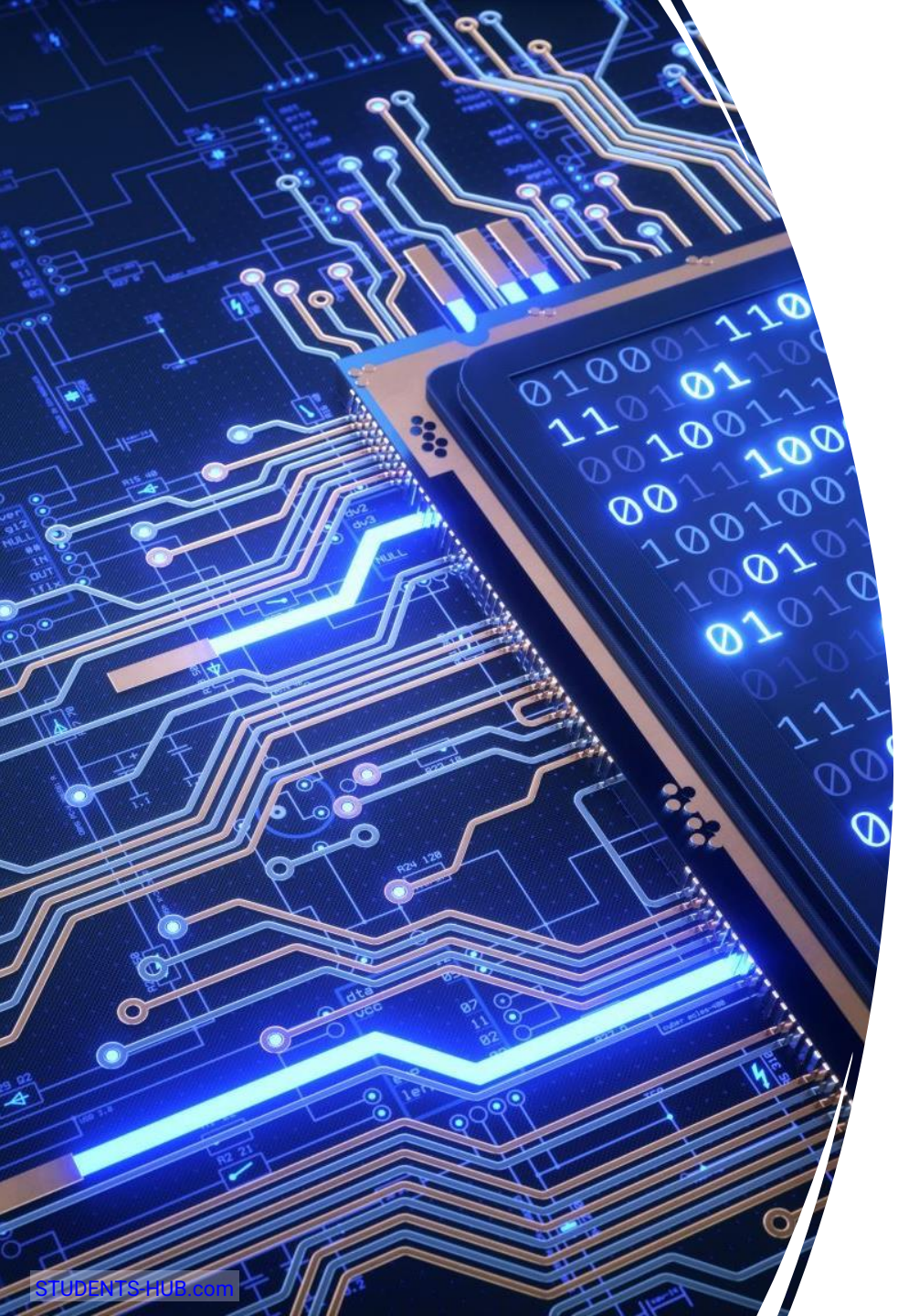
Course Objectives

- **Communication Protocols**: Master various communication protocols used in embedded systems, including serial communication, and networking. wireless technologies are not part of this course.
- **Sensors and Actuators**: Explore the diverse range of sensors and actuators used in embedded systems and their practical applications.



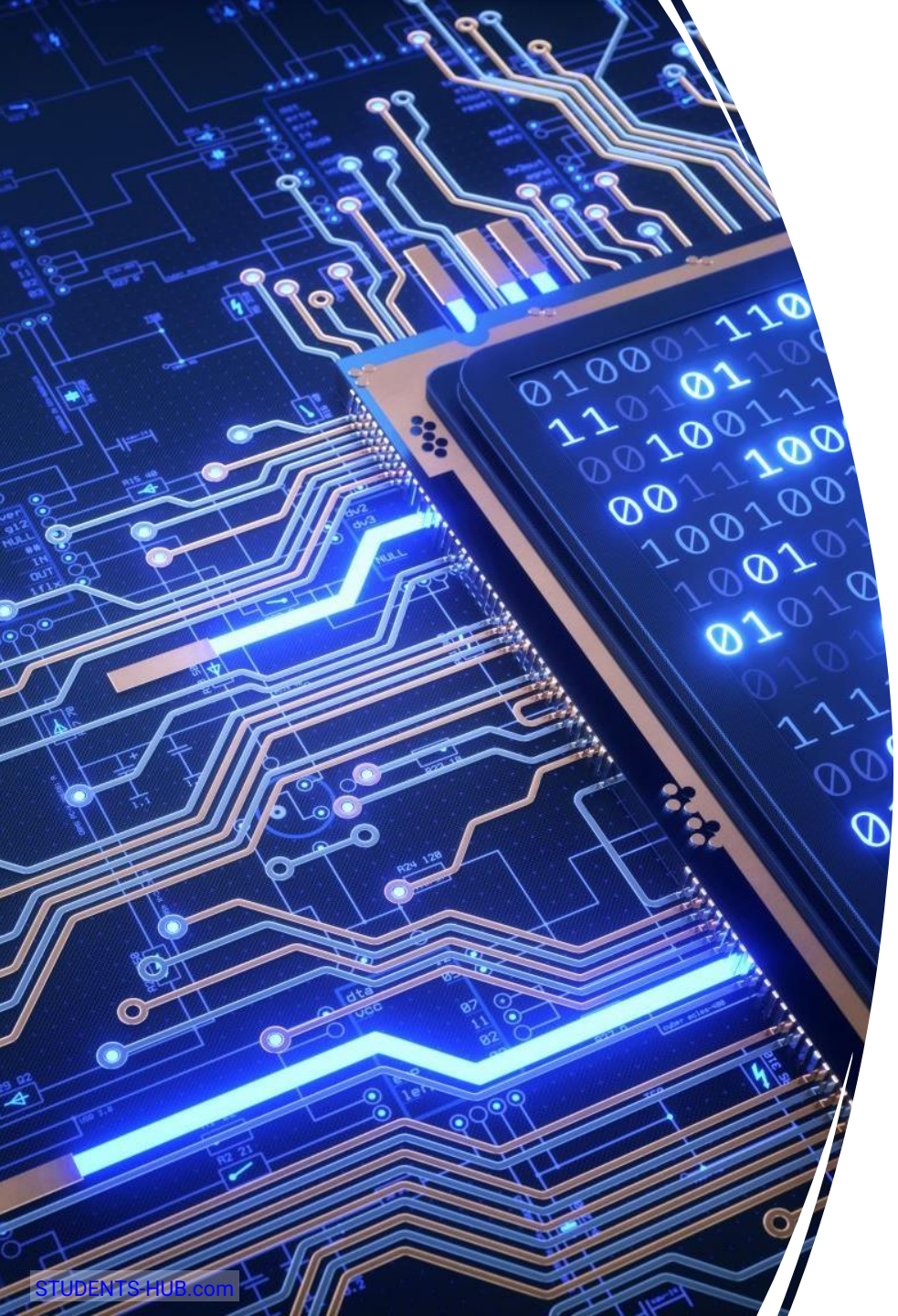
What are Embedded Systems?

- *We can think of an embedded system as a computing device developed for solving a specific problem.*
- *It interacts with the environment as acquiring data, processes the acquired data, and produces the corresponding output accordingly*
- *The designer developing the embedded system must know how the hardware works and how the dedicated software should be formed for it*



What are Embedded Systems?

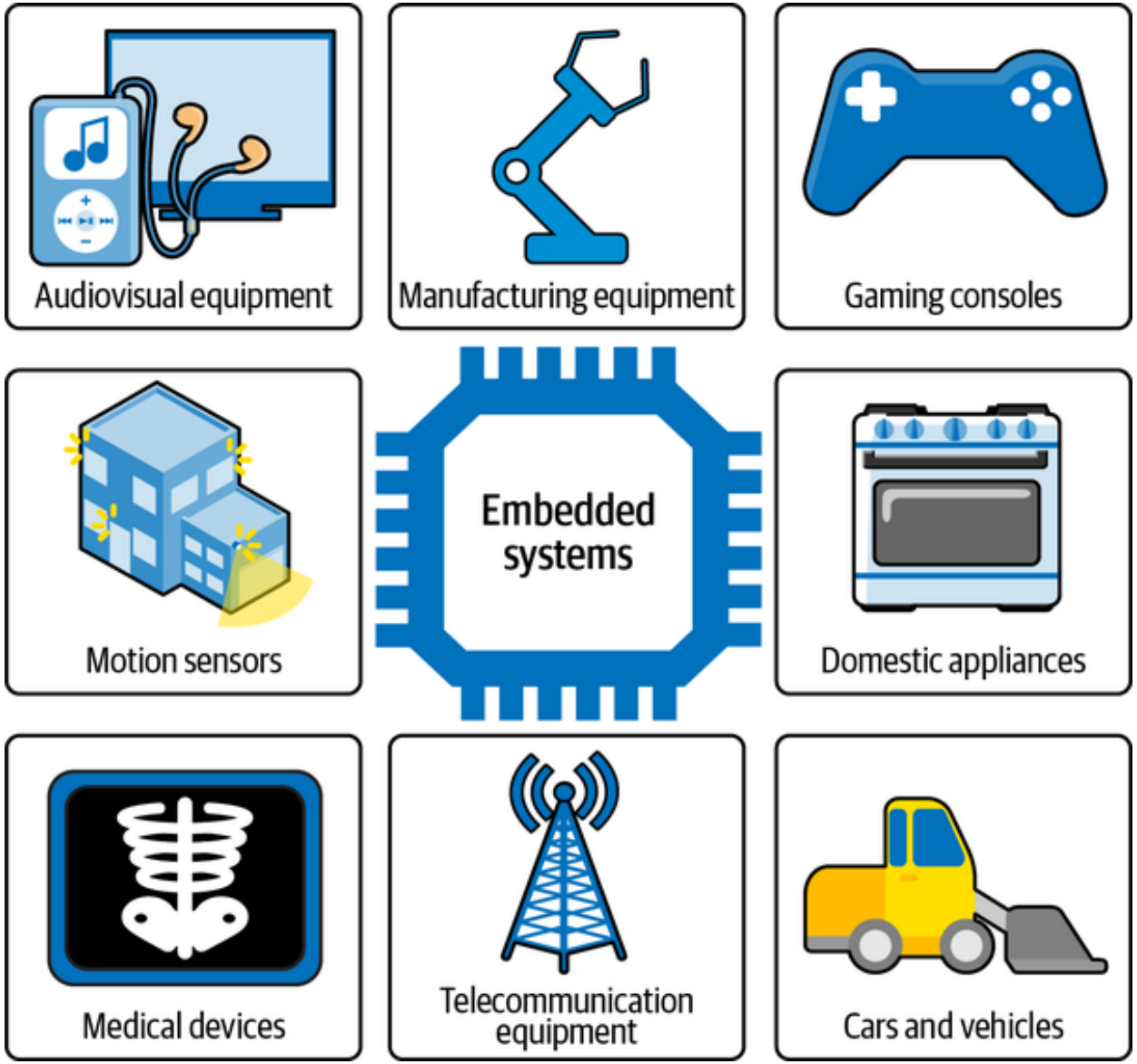
- *We can group embedded systems based on their hardware properties as follows*
  - *System-on-Chip (SoC)*
  - *Field-Programmable Gate Arrays (FPGA)*
  - *Application-Specific Integrated Circuits (ASIC)*
  - *Microcontrollers (MCUs)*



What are Embedded Systems?

- *In this course, It is a system built around a microcontroller and intended to control a certain set of tasks.*
- *microcontrollers are programmable chips that contain a built-in processor, memory, and programmable peripherals for input and output.*
- *Microcontrollers installed inside a mechanical or electrical devices to enable the electrical equipment to carry out certain duties or jobs automatically.*

Examples of Embedded Systems



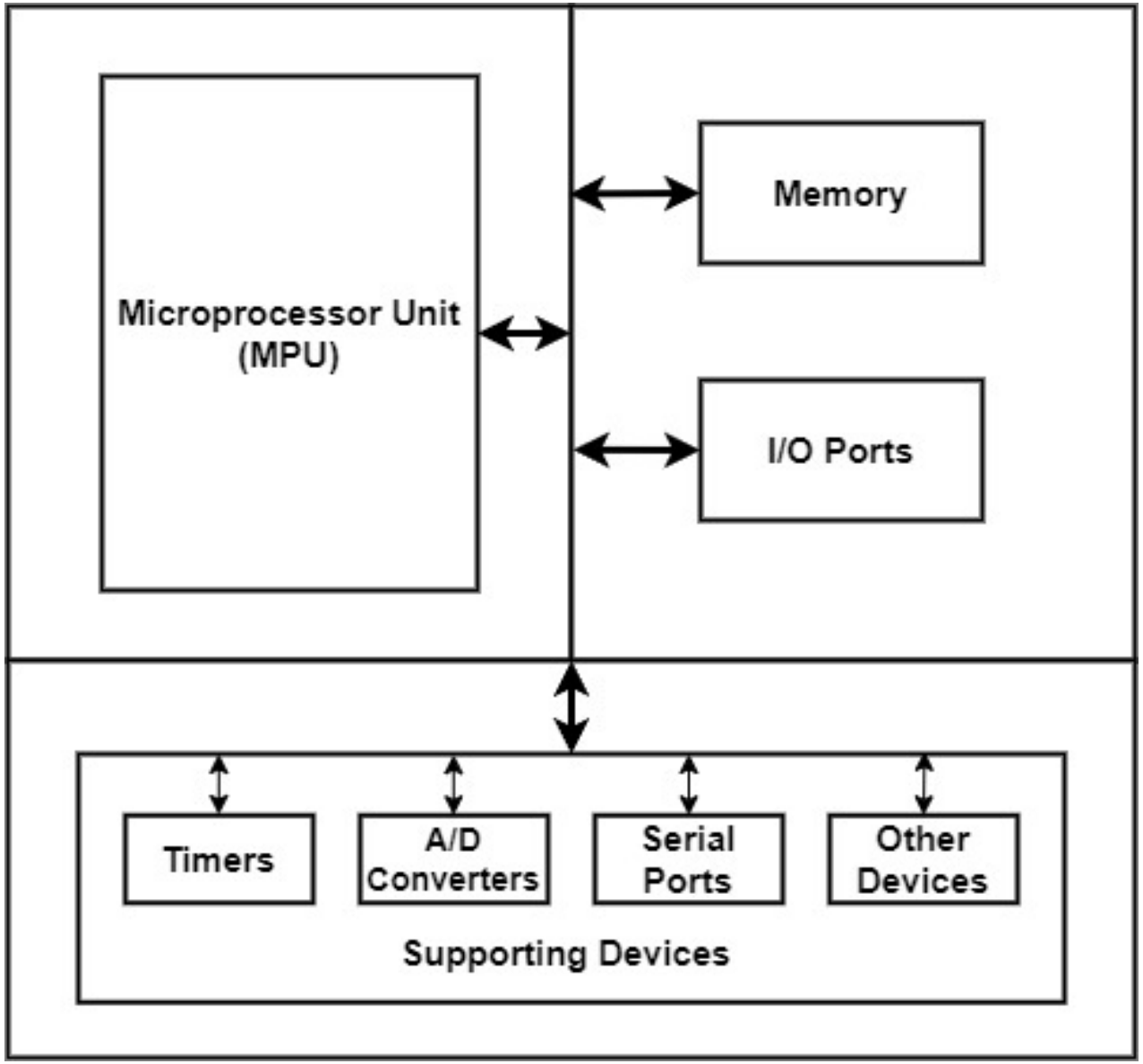
Embedded Systems Vs General Purpose Computer

- **Reliability and Stability**: Embedded systems are expected to be highly reliable and stable. Failure can have significant consequences in applications like medical devices, automotive systems, and aerospace systems. Therefore, they are engineered to minimize errors and downtime.
- **Minimal User Interface**: Embedded systems often have minimal or no user interface, as they are meant to run autonomously. User interaction is limited, and feedback is typically through indicators or simple interfaces, like buttons or sensors.
- **Fixed or Changeable Functionality**: Some embedded systems have fixed functionality and cannot be easily updated or modified once deployed (e.g., a microwave oven). Others, particularly those connected to the internet of things (IoT), may have the ability to receive over-the-air updates to change or enhance their functionality.
- **Long Lifecycle**: Embedded systems often have a longer product lifecycle compared to consumer electronics. This is because they are integrated into critical applications where stability and long-term support are essential.
- **Cost Constraints**: Cost is a significant consideration in embedded system design. Manufacturers aim to keep production costs low while meeting performance and reliability requirements.

Embedded Systems Vs General Purpose Computer

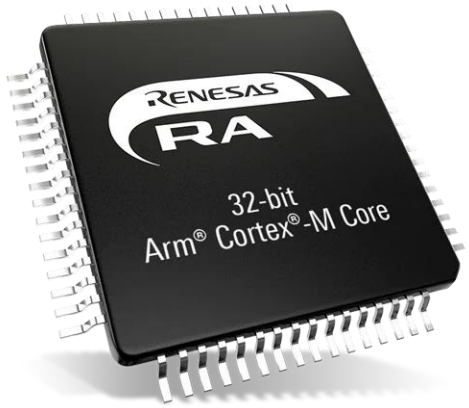
- **Real-time Operation**: Many embedded systems are required to operate in real-time, meaning they must respond to external events or inputs within a predefined timeframe. Real-time systems are often classified into hard real-time (strict timing constraints) and soft real-time (less strict timing constraints) categories.
- **Resource Constraints**: Embedded systems typically have limited resources compared to general-purpose computers. These constraints include limited processing power (CPU speed), memory (RAM and ROM), and storage. Designers must optimize their code and resource usage to meet performance requirements.
- **Integration**: Embedded systems are often integrated into larger systems or devices. They interact with other hardware components or sensors and actuate devices like motors or relays. Integration is a crucial aspect of embedded system design.
- **Efficiency**: Embedded systems are designed to be highly efficient in terms of power consumption and performance. This efficiency is vital for battery-powered devices and systems that need to run continuously for extended periods.

Microcontroller Block Diagram



ARM Cortex-M4

- Every microcontroller out there contains a processor which is responsible for performing all the actions on that microcontroller.
- Each processor is designed, based on a certain instruction set Architecture.
- That architecture can be based on any type, for instance, ARM.



STM32 Microcontrollers

- **STM32** is a family of 32-bit microcontroller integrated circuits by STMicroelectronics.
- The STM32 chips are grouped into related series that are based around the same 32-bit ARM processor core: Cortex-M0, Cortex-M0+, Cortex-M3, Cortex-M4, Cortex-M7.
- Internally, each microcontroller consists of ARM processor core(s), flash memory, static RAM, debugging interface, and various peripherals

Programming Embedded Systems

- Programming Languages (C, C++, Assembly)
- Development Tools (IDEs, Compilers)
- Debugging Techniques

Communication in Embedded Systems

- **Serial Communication (UART, SPI, I2C)**
  - Serial communication is a method of data transmission where data is sent sequentially, one bit at a time, over a single communication channel. It is a straightforward and widely used method for connecting devices in embedded systems, microcontrollers, and computers.
- **CAN (Controller Area Network):**
  - CAN is a more advanced and robust communication protocol designed specifically for use in automotive and industrial applications. It was originally developed by Bosch for automotive applications but has since found use in various other domains due to its reliability and efficiency

Questions?