Embedded Systems

Semester 1, 2025

Dr Chris Lehnert

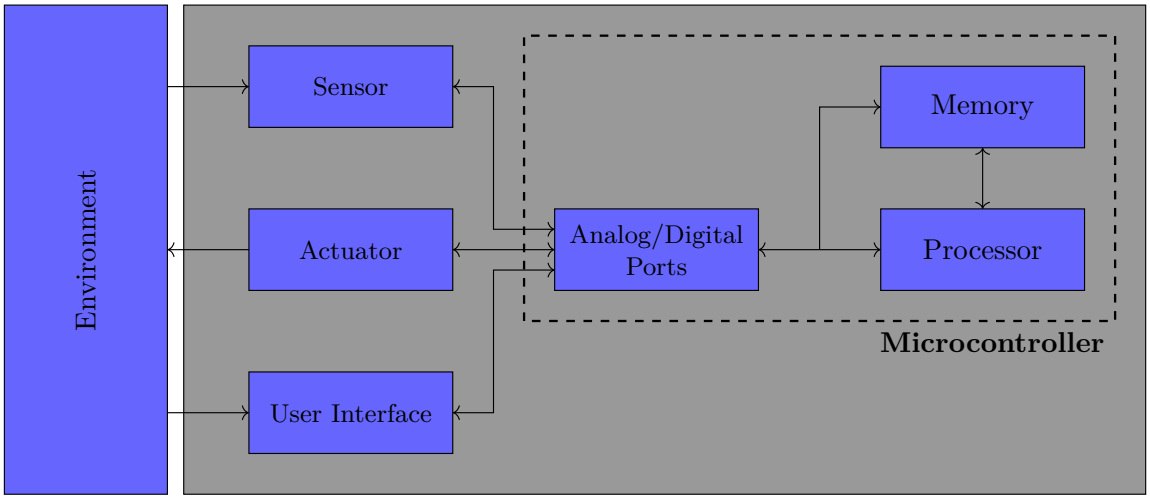
Tarang Janawalkar

Contents

1 Introduction

1.1 Definition of an Embedded System

An embedded system is a combination of computer hardware and software designed for a specific function or functions within a larger system. These systems typically contain computer hardware *within* their implementation and are used in devices to simplify system design and provide flexibility. Often, the user is unaware that a processor is present in the device as an embedded system comprises a suite of different components that communicate with each other to perform a specific task. These components include the processor, memory, and analog/digital ports—which form the microcontroller—and ports that are connected to various input/output devices such as sensors, actuators, and user interfaces—each of which interact with the environment. This is illustrated in the figure below, where the grey box represents the embedded system.



1.1.1 Types of Embedded Systems

Embedded systems can be classified into three main categories:

- **Centralised:** One node performs all work.
- **Distributed:** Nodes distribute work across sub-nodes.
- **Decentralised:** Nodes are only connected to peers in a network.

1.2 Advanced RISC Machines

Advanced RISC Machine (ARM) is a family of Instruction Set Architectures (ISAs) for computer processors. These ISAs are developed and designed by Arm Holdings so that they can be licensed to other companies that design their own ARM-based processors. ARM processors are found in many battery operated devices such as mobile phones, tablets, embedded systems, and some newer laptops.

Reduced Instruction Set Computer (RISC) processors are popular in such applications due to their high performance per watt and ability to execute all instructions in a single cycle. Additionally, because the architecture uses fixed-length instructions, instructions are also easier to pipeline, leading to increased parallelism. The RISC architecture focuses on small and highly-optimised instructions rather than the highly-specialised set of instructions found on Complex Instruction Set Computer (CISC) architectures such as x86. Although this may seem restrictive, this allows instructions to be executed at a greater frequency resulting in improved performance. Complex operations can then be performed in software using these instructions.

1.3 Characteristics of an Embedded System

Embedded systems are characterised by several features. At a high level, they may be designed to be:

- Highly stable
- Time specific
- Task specific
- Cost effective
- Minimal in interface
- Easy to operate
- Real-time
- High-efficiency
- Reliable
- Memory constrained
- Power constrained
- Fault tolerant

1.3.1 Design Goals

These characteristics lead to several design goals in embedded systems such as:

- Reliability: Some systems may be critical to a mission, or life-threatening, and must be able to operate 24/7 without rebooting.
- Performance: Systems may need to respond to many events within a time frame using resources such as computing speed and power effectively. Constraints may need to be placed on inputs to prevent buffer overflows, and inaccuracies from floating-point calculations must be properly handled.
- Cost: Systems may be marketed to consumers and must therefore manufacturing minimise cost and be easy to produce.

1.4 Real-Time Applications

A system is said to be real-time if the total correctness of an operation not only depends on its logical correctness, but also upon the time in which it is performed. A primary design goal of real-time systems is **meeting deadlines**.

- **Soft real-time systems** execute as fast as possible requiring on explicit deadline on the response time.
- **Hard real-time systems** impose a strict deadline on the response time. If the deadline is missed, the system fails.

1.4.1 Real-Time Operating Systems

Embedded systems are typically developed using low-level programming languages such as C, C++, and assembly, for their performance and reliable compilation. The compilation process is different from that of a desktop application where code is compiled into an executable file which can be executed by the operating system. Instead, embedded systems (or those with sufficient resources) make use of **real-time kernel** libraries alongside application code to produce a single binary image that is flashed onto the device. These systems are known as real-time operating systems (RTOS). The kernel is software that manages this real-time system by providing abstractions for creating threads (tasks), scheduling, input/output operations, memory management, and other functions in an operating system.

1.5 Tiva C Series Microcontrollers

This unit uses the Texas Instruments Tiva C series TM4C1294NCPDT microcontroller which is housed on the EK-TM4C1294XL evaluation board. This microcontroller chip is based on an ARM Cortex-M4 core and includes several on-chip peripherals such as an Ethernet controller, USB interface, analog-to-digital converters (ADCs), and timers. The evaluation board also provides additional hardware such as LEDs, switches, a touch screen, and other input/output devices, all of which can be interfaced with the microcontroller.