# Embedded Systems — Lab Assignment 2

***Instructions***

***Use the same groups used to do assignment 1***

**To be sent to** – peterkimanga.m@gmail.com **on or before 14/01/2026**
**Email Subject: Embedded Systems Assignment 2**

For each lab, students will submit the following deliverables:

- Source code
- Simulator waveforms or screenshots
- A brief interpretive report
- A reflection on data integrity and reproducibility

## Lab 4: Architecture and Memory Layout (Simulation)

### ObjectivesS

- Compare Von Neumann and Harvard architecture concepts
- Explore microcontroller memory maps

### Tools

8051 simulator with memory viewing capabilities or a teaching model that exposes code and data memory.

### Tasks

- Inspect the MCU memory map, distinguishing between code and data memory, within the simulator
- Write a program illustrating code versus data access in both architectures (conceptual if the simulator does not separate spaces)
- Demonstrate stack growth and show the impact of ISRs on the stack
- Create a diagrammatic comparison (such as a table) outlining access characteristics and memory regions

### Deliverables

- Report containing diagrams, code snippets, and a short analysis

## Lab 5: Timers, Interrupts, and Real-Time Behaviour (Simulated)

### Objectives

- Implement timer-based events and interrupt handling
- Contrast polling and interrupt-driven approaches

### Tools

8051 simulators supporting Timer0, Timer1, and interrupts.

### Tasks

- Configure Timer0 in mode 1 to generate periodic interrupts
- Implement an interrupt service routine (ISR) that toggles an LED at 1 Hz independently from the main loop
- Create a polling-based version of the same functionality and compare CPU usage, either conceptually or using simulator metrics
- Experiment with nested interrupts and interrupt priority, as supported by the simulator

### Deliverables

- ISR code
- Timing diagrams
- Short comparison of interrupt-driven versus polling approaches

## Lab 6: Data Acquisition and Logging (NI LabVIEW-like Software Simulation)

### Objectives

- Introduce concepts of integrating hardware and software using LabVIEW-like tools or free alternatives
- Simulate data acquisition and a basic control loop

## Tools

Free LabVIEW-like environments (such as LabVIEW Student Edition, if available), Python-based simulation with PyDAQmx, NI's online simulators, or visual programming tools that emulate LabVIEW's VI concepts.

## Tasks

- Create a virtual instrument (VI) that simulates a temperature sensor input
- Implement a simple control and logging loop that triggers an action when a threshold is exceeded
- Visualise data in graphs and export results to CSV files
- Discuss aspects of data integrity, including timestamping, validation, and export integrity

## Deliverables

- VI diagrams and flowcharts
- Data logs
- Short report addressing reproducibility and data integrity