Embedded Systems All Around Us

Course: Embedded Computing (IoT Pathway) | **Unit 1** | **Instructor:** Mr. Norman

---

Before We Start: The Frame

Every embedded system in these notes follows the same three-part pattern. As we go through each example, name the parts out loud.

> **INPUT → PROCESS → OUTPUT**
>
> A sensor takes in data → a microcontroller decides what it means → an actuator or display does something about it.

Working definition: An *embedded system* is a computer built into a larger device to do one specific job. It usually has no keyboard, no monitor, and no "apps." You don't think of it as a computer — but it is one.

The question that drives this unit: What happens when it fails?

---

1. Embedded Systems in Vehicles

Your car is not one computer. It's 50–150 of them networked together.

System	What it does
ABS (anti-lock braking)	Detects wheel lock and pulses the brakes automatically — faster than any human foot can
Airbag sensors	Accelerometers detect collision force and trigger deployment in **milliseconds**
ECU (engine control unit)	The central system managing fuel injection, ignition timing, and emissions
Lane departure warning	Camera-based system detects lane markings and alerts the driver
Backup camera	Image processor overlays steering guidelines onto a live video feed

> 💬 **Discussion:** What would happen if the ECU in your car suddenly failed at 70 mph?
>
> Think through it in order: What stops working first? What still works? What does the driver actually experience?

---

2. Embedded Systems in Healthcare

This is where failure stops being an inconvenience.

- **Pacemaker** — Implanted system monitors heart rhythm and delivers an electrical pulse when needed.
    - *Input:* electrical heart signals → *Output:* corrective pulses
    - Failure consequence: **fatal**

- **Insulin pump** — Wearable system that delivers measured insulin doses.
    - 🔗 *Connects back to the Insulin Pump Hack from Lesson 1.2*

- **MRI machine** — Controls magnetic field timing, data acquisition, and patient safety interlocks.

- **Infusion pump** — Delivers precise medication doses intravenously; tracks flow rate and alarms on error.
-

> ⚠️ **Key idea: LIFE-CRITICAL SYSTEMS**
>
> In these devices, failure consequences are **immediate and severe**. There is no "restart it and see." This is why embedded engineering has different standards than app development — and why the code review process is completely different.

---

3. Embedded Systems in Criminal Justice

Here, the embedded system isn't just doing a job. It's producing **evidence**.

System	How it works
GPS ankle monitors	Location tracking with documented failure modes — and specific populations affected → *preview of Lesson 1.5*
Speed cameras	Captures vehicle speed and an image, then triggers an automated citation
Breathalyzers	Sensor measures blood alcohol content from a breath sample; used as legal evidence in court
Body cameras	Recording, storage, activation, and chain of custody are all embedded system functions

💬 **Discussion:** Should embedded systems be allowed to generate legal evidence automatically, without human review?

Before you answer: who wrote that code? Was it audited? Who is responsible if the sensor is out of calibration?

---

4. Embedded Systems in Everyday Life

You've used every one of these. You've probably never called any of them a computer.

- **Smart speakers** (Alexa, Google Home) always-on system running voice recognition AI
- **Washing machines** timer and sensor system managing water level, temperature, and cycle timing
- **Security cameras** image capture and transmission
- **Video game controllers** reads button inputs and transmits them over a wireless protocol
- **Elevators** controls the motor, doors, floor selection, and safety interlocks

✏️ **Quick task:** Pick one item from this list. Name its inputs, its processing job, and its outputs. Then name one realistic failure mode.

---

5. Two Platforms You Need to Know: Arduino vs. Raspberry Pi

	Arduino Uno	**Raspberry Pi**
What it is	Microcontroller board	Single-board computer
Operating system	None — runs one program, forever	Full Linux OS
Language	C++	Python (and others)
Best for	Physical computing, direct hardware control	Complex IoT projects, media servers, computer vision
Multitasking	One dedicated program	Many programs at once
In this course	✅ **This is what we use**	Used in other pathway courses

🎯 **The distinction to remember: Arduino is a microcontroller** — dedicated, single-purpose, predictable. **Raspberry Pi is a full computer** — flexible, powerful, more overhead.

Both are used in industry for prototyping and in education for learning. Neither one is "better" — they solve different problems.

---

6. Careers in Embedded Computing

This is not a hypothetical list. Every skill in this course maps to a line on one of these job descriptions.

Role	What you'd actually do	Typical education	Pay (entry–mid)
Embedded systems engineer	Design hardware and write firmware	Electrical or computer engineering	$75,000–$115,000
Firmware developer	Write low-level C/C++ that runs directly on microcontrollers	CS or computer engineering	$70,000–$110,000
Hardware engineer	Design physical circuit boards (PCBs)	Electrical engineering	$70,000–$105,000
IoT engineer	Connect embedded devices to networks and cloud platforms	CS or electrical engineering	$75,000–$120,000
Embedded security engineer	Test and secure embedded systems against attack — **growing field**	CS, CE, or cybersecurity	$85,000–$130,000+

Certifications — the other path in

These can supplement a degree, and in some roles substitute for one:

- CompTIA A+
- CompTIA IoT+
- Arduino certification
- AWS IoT
- Certified Embedded Systems Engineer (CESE)

> 📌 **Career path note:** The work you do in this course — wiring circuits, writing C++ for a microcontroller, documenting a build, analyzing a failure — goes on a résumé. Start the Developer Log with that in mind.

---

7. SkillsUSA — Our CTSO

CTSO = Career and Technical Student Organization. Ours is **SkillsUSA**.

Mission: Prepare students for skilled trade, technical, and service careers through hands-on competition and leadership development.

Who joins: Students enrolled in CTE courses. That's you.

Competitive events that connect directly to this course

- **Electronics Technology** — hands-on circuit building, troubleshooting, and testing
- **Internetworking** — network design and configuration
- **Cybersecurity** — defending systems, identifying vulnerabilities
- **Engineering Technology / Design** — technical design and problem solving

How it works

Local / school → State → National

What you get out of it: leadership conferences, scholarships, employer connections, and a résumé entry that means something to people who hire in this field.

> 🙋 **How to join:** Through Mr. Norman. Ask today if you're interested.

Terms to Know

Term	Definition
Embedded system	A computer built into a larger device to perform one dedicated function
Microcontroller	A single chip containing processor, memory, and I/O — runs one program
Firmware	Low-level software written to run directly on hardware
Sensor	Component that converts a physical condition into an electrical signal (input)
Actuator	Component that converts an electrical signal into physical action (output)
Life-critical system	A system where failure causes immediate injury or death
Safety interlock	A control that prevents unsafe operation under defined conditions
Failure mode	A specific way a system can fail, and what happens when it does

---

Exit Questions

Answer in complete sentences in your Developer Log.

1. Name one embedded system you interacted with **before you got to school today**. Identify its input, its processing job, and its output.
2. Explain the difference between a microcontroller and a single-board computer in your own words.
3. Choose one system from these notes and describe a realistic failure mode. Who is affected, and how badly?
4. Which of the five careers listed interests you most, and what is one specific skill from this course that role requires?
