Roadmap to become Embedded design engineer

Silicon Community - Session 1

Agenda

1. What is embedded system design?
2. What are all the blocks in the embedded system?
3. Embedded development life cycle?
4. Steps involved from idea to prototype?
5. What are the roles in the embedded industry?
6. Skill Set required for the respective roles

Embedded System Design - Introduction (1)

Embedded systems are specialized computer systems designed to perform **specific tasks** within larger systems or devices.

They are typically integrated into other **hardware or machinery** and are dedicated to executing a specific function or set of functions.

These systems are widely used across industries and play a vital role in various applications.

Embedded System Design - Introduction (2)

Example:

The embedded system within a **digital camera** is responsible for controlling various tasks such as **autofocus, exposure control, white balance adjustment, image compression, and storage management.**

It receives input from the user through **buttons or touchscreens, processes the image data captured by the image sensor, applies various image processing algorithms, and stores the resulting image or video on a memory card.**

The embedded system in a digital camera must operate in real-time, ensuring smooth operation, quick response times, and accurate image processing. It optimizes power consumption to maximize battery life and provides user-friendly interfaces for settings and controls.

Embedded System Design - characteristics

Characteristics of Embedded Systems:

1. **Specific Functionality:** Embedded systems are designed to perform specific tasks and are optimized for efficiency and reliability in executing those functions.
2. **Real-Time Operation:** Many embedded systems require real-time operation, meaning they must respond to events or input within strict timing constraints.
3. **Limited Resources:** Embedded systems often have limited resources in terms of processing power, memory, and energy consumption. Designers must optimize their solutions to work within these constraints.
4. **Connectivity:** Embedded systems can be connected to other devices or networks to exchange data or communicate with external systems.
5. **Reliability:** Embedded systems are expected to operate reliably for extended periods without human intervention, often in harsh environments.

Embedded System Design - Applications

1. **Consumer Electronics:** Embedded systems are widely used in everyday devices like smartphones, tablets, televisions, smart appliances, and wearable devices. They provide functionalities such as user interfaces, data processing, connectivity, and control.
2. **Automotive:** Embedded systems are essential in modern vehicles, controlling various aspects such as engine management, entertainment systems, safety features, navigation, and communication.
3. **Industrial Automation:** Embedded systems play a crucial role in industrial automation, including robotic systems, process control, monitoring, and data acquisition. They help improve efficiency, accuracy, and safety in manufacturing and production processes.
4. **Healthcare:** Embedded systems are used in medical devices and equipment like pacemakers, infusion pumps, diagnostic equipment, and patient monitoring systems. They assist in diagnosis, treatment, and patient care.

Embedded System Design - Applications (2)

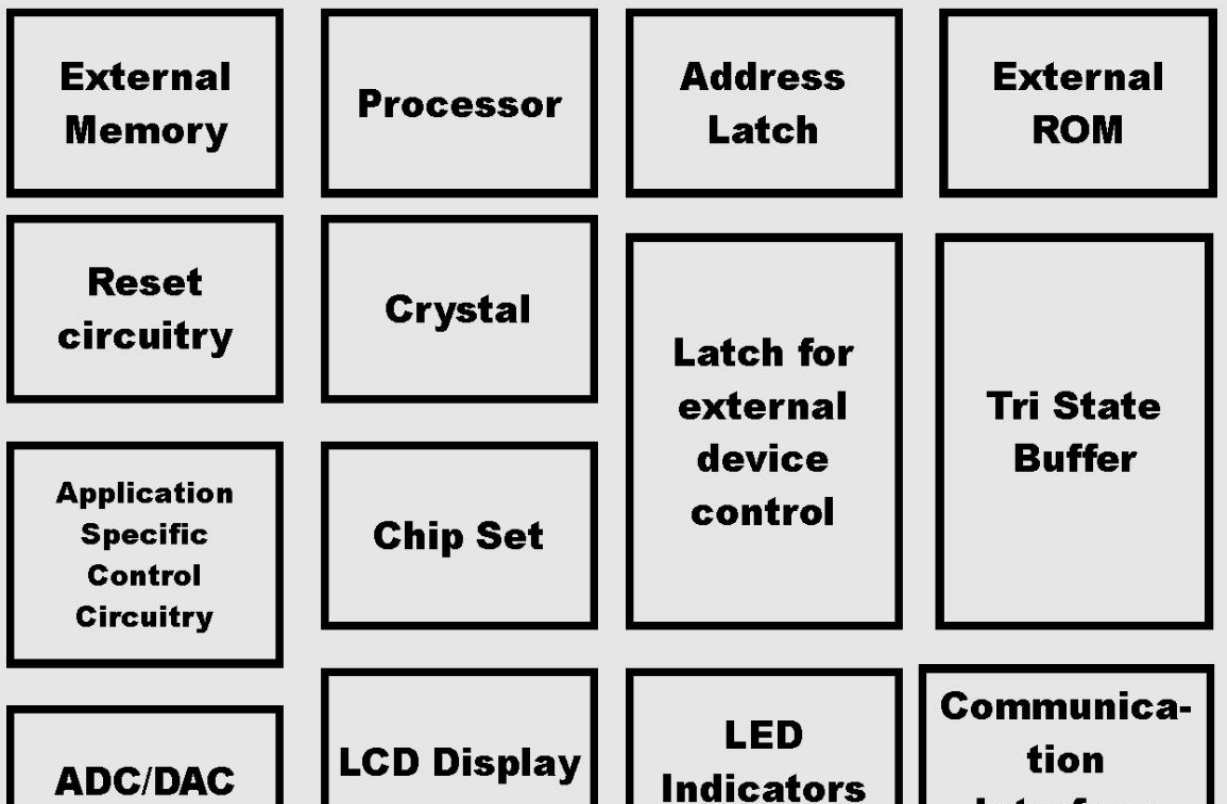
5. **Aerospace and Defense:** Embedded systems are integral to aerospace and defense applications, including flight control systems, guidance systems, communication systems, radar systems, and surveillance equipment.

6. **Home Automation:** Embedded systems enable automation and control of home devices and systems, such as security systems, smart thermostats, lighting controls, and home entertainment systems.

7. **Internet of Things (IoT):** Embedded systems are the backbone of IoT devices, connecting physical objects to the internet, collecting and exchanging data for various applications like smart cities, agriculture, energy management, and environmental monitoring.

8. **Energy and Utilities:** Embedded systems are used in smart grid systems, energy management systems, and utility monitoring devices to optimize energy usage,

Embedded System Design - Various blocks



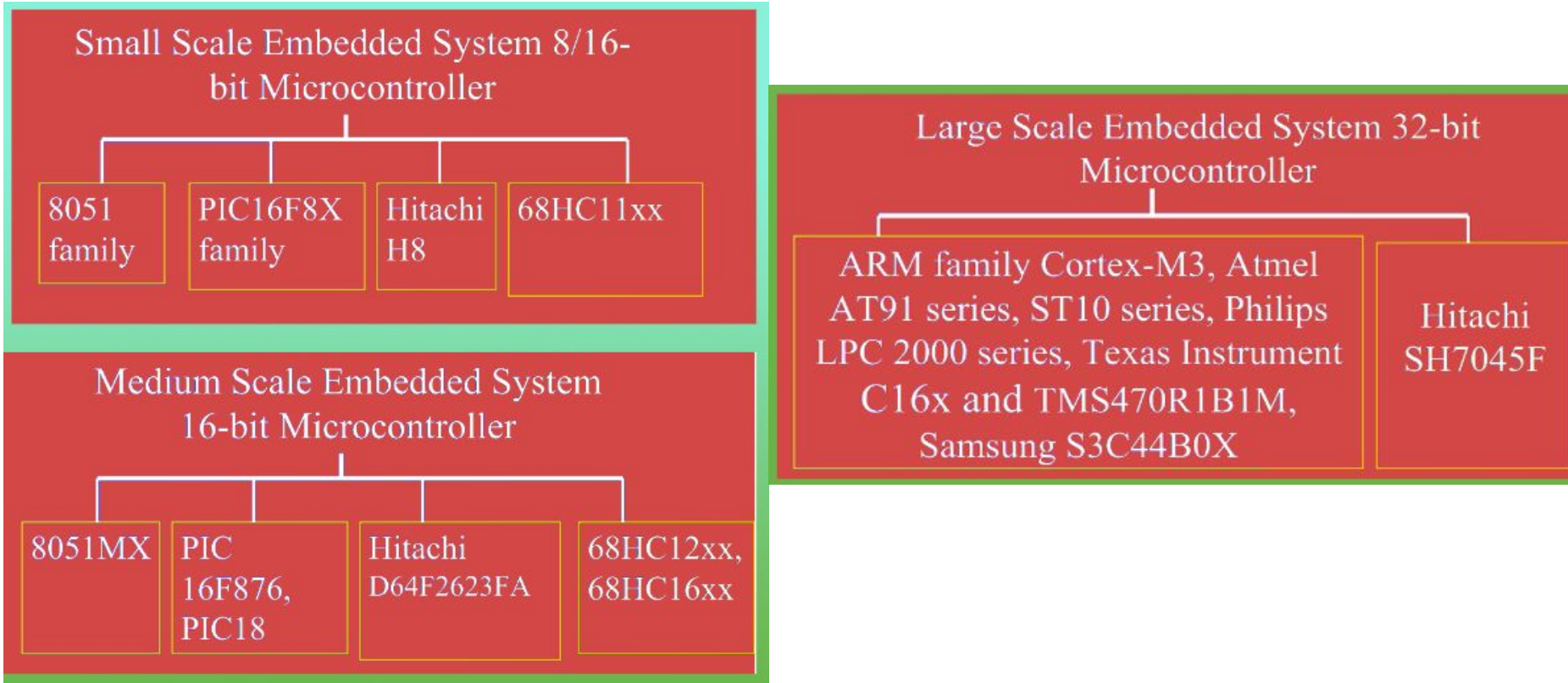
Processor

The processor used in embedded systems can be of three types:

- Micro-Controllers
- Microprocessors
- Digital Signal Processor

Each of them is specified by clock speed(100MHz, etc) and data word length(8-bit, 16-bit, 32-bit).
The higher the clock speed, the faster the processor.
Same way, bigger data word-length leads to higher precision.
The word length also indicates the processor's capability to address the memory. This addressing capability is specified by the number of bits used for addressing.

Processor - Microcontroller (1)



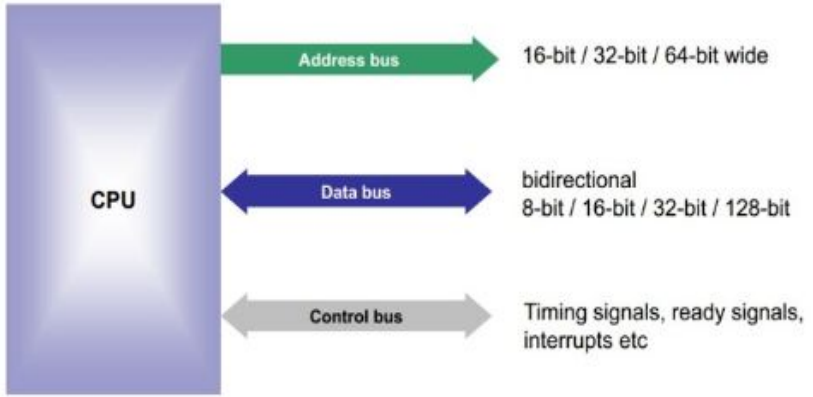
Processor - Microcontroller (2)

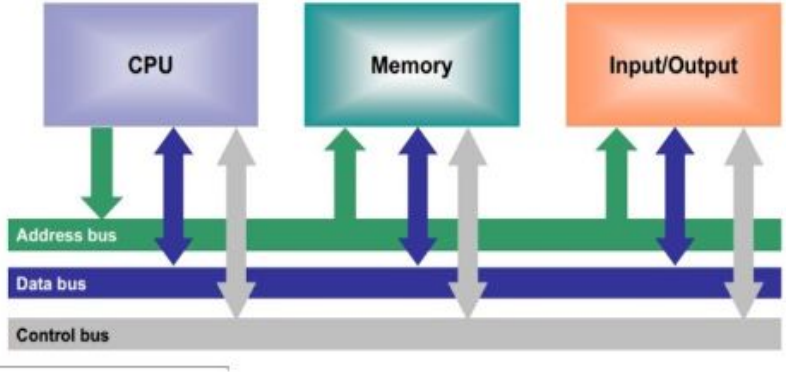
Micro-controllers are mainly of Intel's 805x family, Motorola's 68HCxx, etc. A typical micro-controller contains a CPU, interrupts, timer/counter, memory (RAM, ROM, or both) and other peripherals in same Integrated Circuit(IC).

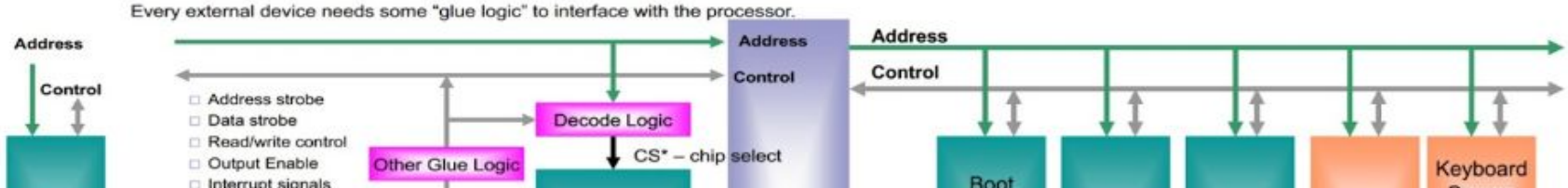
Micro-controller are often an ideal solution for control applications because you can use them to build an embedded system with little additional circuitry.

The 8-bit microcontrollers are used for process control applications, such as the ones found in toys and smart cards. If your processing power and memory requirements are high, you need to choose a 16-bit or 32-bit processor

Processor - MicroProcessor (1)







Processor - MicroProcessor (2)

Based on the instructions, a microprocessor does three basic things:

- Using its ALU(Arithmetic/Logic Unit), a microprocessor can perform mathematical operations like addition, subtraction, multiplication and division. Modern microprocessors contain complete floating-point processors that can perform extremely sophisticated operations on large floating-point numbers.
- A microprocessor can move data from one memory location to another.
- A microprocessor can make decisions and jump to a new set of instructions based on those decisions.

Processor - MicroProcessor - Types(3)

Based on hardware characteristics

- **Complex Instruction Set Computer (CISC)**

- large number of complex addressing modes
- many versions of instructions for different operands
- different execution times for instructions
- few processor registers
- microprogrammed control logic

- **Reduced Instruction Set Computer (RISC)**

- one instruction per clock cycle
- memory accesses by dedicated load/store instructions
- few addressing modes
- hard-wired control logic

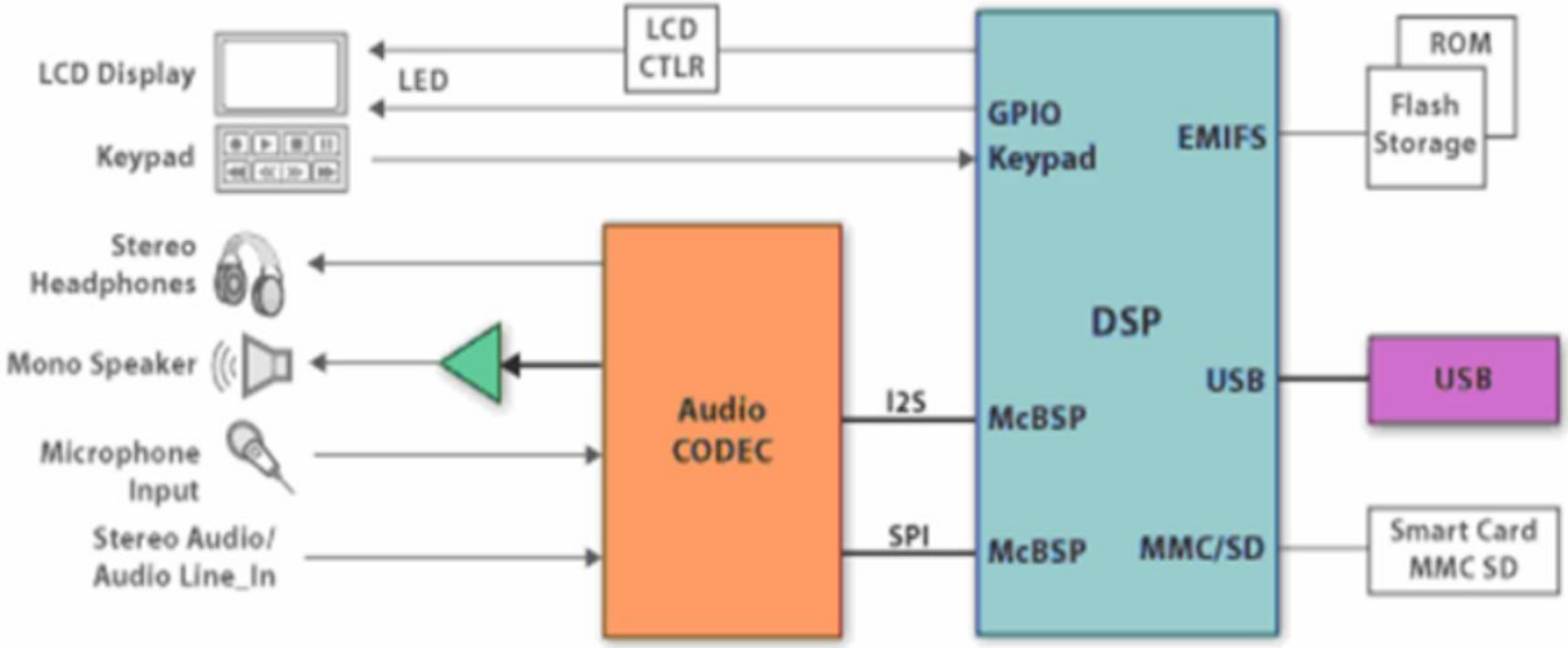
Based on the application areas

- GPP (General Purpose Processor)
- SPP (Special Purpose Processor)

Processor - Digital signal Processor (DSP) (1)

- *The digital signal processor (DSP)* is a special designed processor to handle signals, rather than data.
- Processing signals (whether audio or video) is much more complex than processing digital signals. To process audio and video signals, the hardware/software needs to perform an operation called filtering, in which unwanted frequencies are removed.
- In signal processing, another important task is to convert the signal in frequency domain.
- Analyzing a signal in frequency domain requires intensive mathematical computation, which general-purpose processors take a lot of time to carry out.
- The DSP carries out such mathematical computations quickly using a special module called the Multiplier and Accumulator.
- DSPs are available with various clock frequencies and word-lengths, with each catering to different market segments (such as speech processing, high-fidelity music processing, image compression, video processing, etc).

Processor - Digital signal Processor (DSP) (2)



Memory

The memory used in embedded systems can be either internal or external. The internal memory of a processor is very limited. For small applications, if this memory is sufficient, there is no need to use external memory.

Internal Memory:

Internal memory is found in the same silicon as the processor. It is accessible to the processor without any use of input and output. If the capacity of the internal memory is high, the cost is likely to be more. The advantage of larger memory internal to the chip is that there will be less program swapping, resulting in fast instruction and data access.

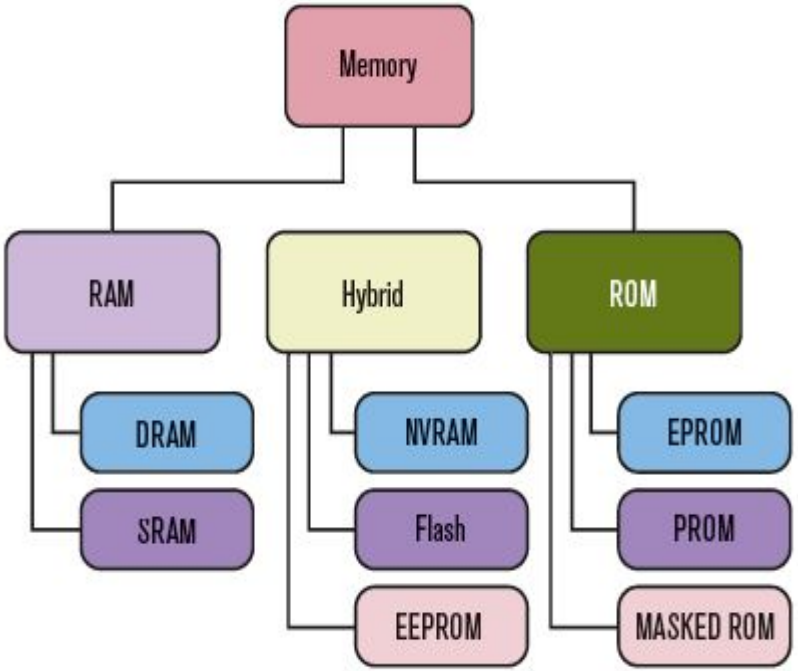
External Memory:

Separate devices are used to store the program and temporary data. When the program and data are stored in external devices, data access takes more time than it would with internal memory. External devices require additional circuitry to be built around the processor, to generate the addresses for retrieving the memory contents.

Memory (2) - Volatile VS Non-volatile

	Volatile memory	Non-volatile memory
Description	This type of memory loses all of its data when the power source is turned off	This type of memory wil retain all the data when it loses its power source
Examples	RAM Cache Registers	Solid state drives Hard disk drives Read only memory USB sticks

Memory (3) - Common memories in ESD



Common Embedded System Memory Types

Memory (4) - Random Access Memory

Its a type of memory that can be accessed randomly - that is, a memory location can be accessed without touching other locations. RAM is also called **Read-Write Memory**, as you can perform both read and write operations on this memory device.

RAM are of two types

- **Static RAM** : Retains its contents as long as electrical power is being supplied to the device. For Example: The desktop PC contains SRAM, which loses data when power is switched.
- **Dynamic RAM** : DRAM has short life cycle, typically one fourth of second, even if power is being supplied. Its uses DRAM controller in conjunction with DRAM to refresh the memory periodically They generally have low costs. Handheld computers contains DRAM to store data.

Memory (5) - Read Only Memory

It is a memory device from which the processor can read data but to which it cannot write data. Like CD. The programs and data that needs to be permanently stored are kept in ROM devices. ROM Retains its contents even if power is switched off; therefore it is used to store program codes and any permanent data retained to initialize and operate embedded systems. A veriety of ROM devices are available, each with different capabilities.

- **PROM(Programmable ROM):** It can be programmed only once.
- **EPROM (Erasable ROM):** It can be programmed many times.An EPROM programmer is required to program the EPROM chip.

Memory (5) - Hybrid Memories

- **Electrically Erasable Programmable ROM (EEPROM)** : It is similar to EPROM, but it can be erased electrically by applying an electrical signal to one of the pins.
- **Non-Volatile RAM (NVRAM)** : It is a SRAM with battery backup so that the contents are not erased even if power is switched off. Its is very expensive but data access through it is fast.
- **Flash Memory** : it is also a non - volatile memory, fast EEPROM. The main attraction of flash is that it can be erased one block at a time and programmed one bit at a time. Flash memory devices are high density, low cost, fast (to read but not to write), and electrically programmable. It is being extensively used for embedded systems that contain embedded OS and the application program such as handheld computers.

Crystal and Reset circuits

- The CPU needs a clock source, and a crystal oscillator generates the clock.
- The Crystal is chosen based on the cock frequency of the processor.
- Micro-controllers provide an on chip oscillator and you connect an external crystal or ceramic resonator. The clock generation circuitry determines the various states of machine cycles

Reset circuits

It is generally built in the hardware to take care of any unforeseen problems.
This circuit handles software handles software hang-ups, power supply failures, etc.
The processor sends a status signal to this circuit periodically. If in case this signal is not received, it is an indicator that something is wrong, which is then reset by this circuit. Single chip solutions are also available.

Watchdog Timer

- Most of the embedded systems have no provision of resetting the processor in such cases a watchdog timer is used.
- It is set to a large value and is counted down.
- When value reaches zero, the processor resets.
- If things are fine and there is no need to reset the processor, the processor resets the watchdog timer to that large value again.

ADC and DAC

- Embedded systems receive their inputs from the external world in the form of analog signals.
- An analog signal (amplitude varies continuously) needs to be converted into a digital signal as processor only takes digital signals (a series of ones and zeros represented by voltages).
- Thus these conversions are performed by Analog-to-Digital(ADC) and the reverse conversion of digital to analog by Digital-to-Analog(DAC).

Roadmap to become ES designer (1)

1. **Education:**
   - Obtain a bachelor's degree in electrical engineering, computer engineering, or a related field. This provides a solid foundation in electronics, digital systems, microprocessors, and programming.
2. **Gain a Strong Foundation:**
   - Develop a solid understanding of digital systems, microcontrollers, and programming languages like C (and then C++). Study topics such as digital logic, microprocessor architecture
3. **Learn Embedded System Concepts:**
   - Acquire knowledge of embedded system concepts, including real-time operating systems (RTOS), device drivers, interfacings (such as I2C, SPI, UART)
4. **Hands-on Experience:**
   - Gain practical experience by working on embedded system projects. Start with small-scale projects, like building simple microcontroller-based circuits, and gradually progress to more complex projects. This hands-on experience will help you understand hardware-software integration, debugging techniques, and troubleshooting.

Roadmap to become ES designer (2)

5. **specialize in Relevant Technologies:**

    - Explore and specialize in technologies commonly used in embedded systems, such as ARM Cortex-M microcontrollers, embedded Linux, FPGA, wireless communication protocols (such as Bluetooth, Wi-Fi), and sensor integration.

6. **Learn Industry Standards and Tools:**

    - Familiarize yourself with industry standards and tools used in embedded system development. Examples include the C programming language, debugging tools (such as JTAG debuggers), simulation software (such as Proteus, MATLAB/Simulink), and integrated development environments (IDEs) like Keil, IAR Embedded Workbench, or Eclipse.

7. **Collaborate and Network:**

    - Engage with the embedded system community by participating in forums, online communities, and social media groups. Collaboration and networking with professionals in

Various roles and skills required (1)

1. **Embedded Systems Engineer:**
   - Skills: Strong knowledge of embedded systems design, microcontrollers/microprocessors, C/C++ programming, real-time operating systems (RTOS), digital electronics, hardware-software integration, debugging and troubleshooting, circuit design, and familiarity with communication protocols (e.g., I2C, SPI, UART).
2. **Firmware Engineer:**
   - Skills: Proficiency in embedded systems programming (C/C++), low-level hardware interactions, firmware development and debugging, knowledge of microcontrollers/microprocessors, familiarity with peripheral interfaces (e.g., I2C, SPI, UART), real-time constraints, and experience with version control systems.
3. **Embedded Software Engineer:**
   - Skills: Strong programming skills in C/C++, knowledge of embedded software development, software architecture and design patterns, familiarity with real-time operating systems (RTOS), debugging and testing techniques, communication protocols (e.g., TCP/IP, MQTT), and experience with software development tools and methodologies.

Various roles and skills required (2)

4. Embedded Hardware Engineer:

- Skills: Expertise in digital and analog circuit design, microcontroller/microprocessor selection and integration, schematic and PCB layout design, signal integrity analysis, hardware testing and debugging, familiarity with communication protocols (e.g., I2C, SPI, UART), and knowledge of hardware description languages (HDL) like VHDL or Verilog.

5. Embedded Systems Architect:

- Skills: In-depth understanding of system-level design and architecture, ability to define system requirements and specifications, knowledge of hardware and software interactions, familiarity with communication protocols and interfaces, proficiency in embedded systems programming, strong problem-solving and analytical skills, and experience in selecting

Various roles and skills required (3)

6. Embedded Linux Developer:

- Skills: Proficiency in Linux kernel and device driver development, understanding of Linux file systems, knowledge of embedded Linux distributions (e.g., Yocto Project), C/C++ programming, debugging and troubleshooting on Linux-based systems, familiarity with network protocols and system administration, and experience with cross-compilation and build systems.

7. Verification and Validation Engineer:

- Skills: Expertise in testing and validation of embedded systems, knowledge of test methodologies and tools, experience with test automation, proficiency in scripting languages (e.g., Python), understanding of hardware-software interactions, debugging and troubleshooting skills, and familiarity with software and hardware verification