arm Education

Embedded Systems Essentials with Arm: Getting Started

Module 1

KV2 (1): Benefits and Constraints of Embedded Systems

Although the term “embedded system” is typically used for microcontroller-based systems, it is not restricted to just those. Instead it encompasses systems with as much functionality as possible, embedded into few components. This can range from discrete hardware components to fully embedded PCs, while microprocessors with dedicated hardware elements sit in between.

Microcontroller-based embedded systems have many advantages.

For instance, software-based applications are more flexible than pure hardware, and can be adapted more easily to changes in system requirements. This can even translate to better performance and efficiency.

By partitioning the system into hardware and software components, the total system cost can be minimized. The system will be cheaper to make, operate, and maintain.

Microcontroller-based designs can implement more features. Through system optimization, memory and on-chip components redundancy can also be reduced.

Finally, microcontroller-based systems are more reliable. Failures can be detected, debugged, and compensated for.

However, embedded systems are also subject to constraints.

Cost is very important in high-volume production, especially in today’s competitive markets. It is important to balance manufacturing, design, and production costs.

Device size and weight can be a constraint, especially for portable and wearable systems.

Battery capacity is another constraint, which in turn impacts the power consumption profile affordable to embedded devices.

Finally, designers must consider the effects of any heating issues that may arise due to a system’s operation or the external environment.

These constraints have various impacts on microcontroller-based embedded systems, which make operation and interfacing more difficult.

For instance, since peripherals are already integrated on the chip, they are usually operated in a way that is unique to every controller type. Although this reduces manufacturing costs, it increases the engineering costs for design, development, and debugging. In other words, the board and design complexity is reduced, but the chip-and-coding complexity is increased.

Programming is mostly done in C or Assembly language due to inherent code efficiency and low programming overhead. These languages also allow control of the required clock cycles per operation more easily.

Operating systems are less common for embedded systems, although this is changing in the Internet of Things era. Arm Mbed OS is one example. OSs can simplify development, but the real-time operating system overhead extends code size and reduces transparency around what is really happening inside the MCU.

In summary, in most embedded systems, MCUs are the best choice because they are cheap, flexible, available in multiple configurations, and consume relatively low power. However, they require special development environments as well as support for each processor type for uploading code and debugging.