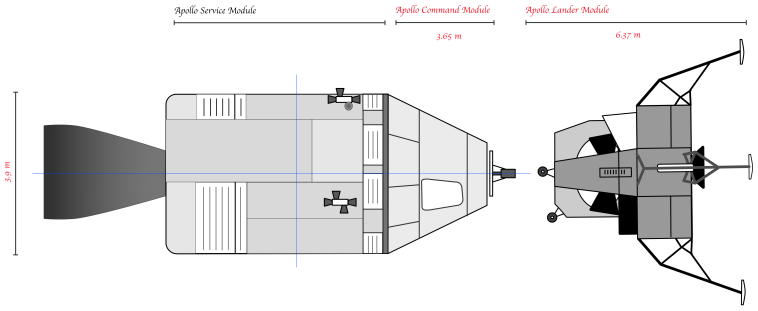
What is an *Embedded System*?

Andrew N. Sloss

October 12, 2023



Apollo Command, Service, Lander Modules

Discussion

> "*Juggling can be thought of as the art of controlling patterns in time and space*"
> Ronald Graham, Mathematician-Juggler [Gra14]

What is an Embedded System? It is a subclass of computing and, as such, involves information processing. Embedded Systems are practical solutions to real-world problems. These problems cover everything from controlling the light in a fridge to landing a rover on the moon. Determining a precise definition can be surprisingly tricky. The first use or creation of the term was by *Dr. Charles Stark Draper* at the *Massachusetts Institute of Technology* (MIT). Draper was responsible for the design of the *Apollo Guidance Computer* (circa 1961). The goal was to *get to the moon*. The guidance system controlled the *Apollo Command Module* and the *Apollo Lunar Module*. This development leans the subject towards uniqueness, i.e., unique problems with unique solutions. Also, it highlights the long close relationship between *Space exploration* and Embedded Systems that continues today with the likes of the *NASA Jet Propulsion Laboratories*, *Blue Origin* and *SpaceX*.

Guidance
Uniqueness

As well as the world of Space exploration, there have been close ties to the military. *Autonetics*, now part of *Boeing*, developed the *Minuteman I missile guidance system*. This guidance system is said to be the first *mass-produced* Embedded System. It highlights the other significant leaning —*economics of scale*. By producing systems at scale, the price per system goes down. Building one or two of *something* is expensive, but we could make 10,000 and be significantly more *cost-effective*.

Production
Scale

Early examples of Embedded Systems centered on specialized *guidance* systems. Guidance systems are all about *control* and *Control Theory*. They gather environmental data (< *input*) to help make the correct *active* decisions (> *output*), e.g., maneuvering a rocket, autonomous driving, or manipulating a robot arm. They handle *events*, which can be anything from pressing a button on a keyboard to receiving a communication signal. By contrast, many of today's Embedded Systems are *passive*, as-in monitor without interfering with the environment, *input-only* and either storing or broadcasting the information. Control and monitoring have been fundamental aspects from the beginning.

Control
Input/Output
Events

A problem may require strict response times (e.g., rocket control) or not (e.g., data logging). We call these types of stimuli-response times *deterministic* or *non-deterministic*. A determinis-

Response
Deterministic

tic response reliably replies to an external stimuli within a strict timing window, i.e., a *Real-Time System*. By contrast, a non-deterministic stimuli-response provides no time guarantee. Real-Time

Let's take a closer look at the term Embedded Systems. First, the word *Embedded* means something firmly connected to something else. In the case of the Apollo Guidance Computer, the device is in the Command Module. Generally, we can say it is any device connected to another object, e.g., a heart monitor connected to a human or a security camera connected to a warehouse. Connection implies interacting with an environment. The word *Systems* means a form of network. It is a device that is part of a bigger goal. The network could be physical or digital, e.g., a pollution monitoring device storing data locally or an audio player connected to the internet. Taken together, we have a device that interacts with an environment, translates the environment into information, and shares the information with other devices. Network

Embedded Systems have the difficult task of bridging the gap between perfect theoretical concepts and the chaotic reality of nature. Nature covers classical mechanics (if lucky), statistical mechanics, quantum mechanics, non-linear systems, and complex biological systems. They are translators that connect worlds together as best as technically possible. Nature

From an implementation point of view, the hardware ranges from a tiny circuit on the back of a Bee (recording movements) to an extensive dedicated system (dealing with autonomous cars). In other words, these devices are not necessarily small or low-cost; they can be expensive and large. In some ways, Embedded Systems are more closely related to specialized Supercomputers than Personal Computers. They lean towards specialization over generalization. Specialization comes from a combination of hardware and dedicated *software*. Specialization

Embedded Systems operate within strict constraints. These constraints include the physical attributes. Size, power consumption, pressure, or radiation (electromagnetic or acoustic) are all forms of physical characteristics. An Embedded System has to accommodate these constraints within the design. Constraints

There are other less physical constraints that play an important role in Embedded Systems, namely *reliability*, *safety*, *security*, and *privacy*. These constraints are challenging. Reliability is about understanding weaknesses and strengthen those areas to make the device more reliably, i.e., taking unreliable components and making them reliable. Safety is about making a device less likely to cause economic damage or harm. Security is about making it harder to compromise a device. Finally, privacy is about storing personal information without the risk of disclosure. These constraints are not *mutually exclusive*. Reliability Safety Security Privacy

There are subclasses, and we already mentioned one subclass Real-Time Systems, but another critical subclass is *Extreme Embedded Systems*. These are a systems where an environmental constraint or constraints far exceed typical experiences. For example, a Mars rover falls under this definition, a tag system to track whales in the Ocean or a system measuring extreme temperatures. These are the abnormal Extreme Embedded Systems; most standard platforms fail to service these problems, and special-purpose solutions are required. Subclass Extreme

Lastly, we can describe Embedded Systems purely from the behavioral side, i.e., *what are the problems we want to solve?* We use Embedded Systems for everything. They monitor, control, and make decisions, from fridges to autonomous cars. To achieve these tasks, we must interact with the natural world and learn from or modify the environment. These interactions occur through transducers called *sensors* and *actuators*. Transducer Sensors Actuators

KEY TERMS

- Actuators
- Deterministic
- Cost-effective
- Constraint
- Control (Guidance)
- Deterministic
- Economics of Scale
- Environment
- Embedded Systems
- Events
- Extreme Embedded Systems
- Input/Output, Input-only

- Nature
- Privacy
- Production
- Real-Time Systems
- Reliability
- Safety
- Security
- Sensors
- Specialization
- Stimuli-response
- Time
- Transducers
- Uniqueness

References

[Gra14] Ron Graham. Juggling mathematics and magic. *Notices of the International Congress of Chinese Mathematicians*, 1(1):7–9, 2014.