Preface

Importance of embedded systems

Embedded systems can be defined as information processing systems embedded into enclosing products such as cars, telecommunication or fabrication equipment. Such systems come with a large number of common characteristics, including real-time constraints, and dependability as well as efficiency requirements. Embedded system technology is essential for providing ubiquitous information, one of the key goals of modern information technology (IT).

Following the success of IT for office and workflow applications, embedded systems are considered to be **the** most important application area of information technology during the coming years. Due to this expectation, the term **post-PC era** was coined. This term denotes the fact that in the future, standard- PCs will be a less dominant kind of hardware. Processors and software will be used in much smaller systems and will in many cases even be invisible (this led to the term **the disappearing computer**). It is obvious that many technical products have to be technologically advanced to find customers' interest. Cars, cameras, TV sets, mobile phones etc. can hardly be sold any more unless they come with smart software. The number of processors in embedded systems already exceeds the number of processors in PCs, and this trend is expected to continue. According to forecasts, the size of embedded software will also increase at a large rate. Another kind of Moore's law was predicted: *For many products in the area of consumer electronics the amount of code is doubling every two years* [Vaandrager, 1998].

INTRODUCTION

1.1 Terms and scope

Until the late eighties, information processing was associated with large mainframe computers and huge tape drives. During the nineties, this shifted towards information processing being associated with personal computers, PCs. The trend towards miniaturization continues and the majority of information processing devices will be small portable computers integrated into larger products. Their presence in these larger products, such as telecommunication equipment will be less obvious than for the PC. Hence, the new trend has also been called the **disappearing computer**. However, with this new trend, the computer will actually not disappear, it will be everywhere. This new type of information technology applications has also been called **ubiquitous computing** [Weiser, 2003], **pervasive computing** [Hansmann, 2001], [Burkhardt, 2001], and **ambient intelligence** [Koninklijke Philips Electronics N.V., 2003], [Marzano and Aarts, 2003]. These three terms focus on only slightly different aspects of future information technology. Ubiquitous computing focuses on the long term goal of providing "information anytime, anywhere", whereas pervasive computing focuses a somewhat more on practical aspects and the exploitation of already available technology. For ambient intelligence, there is some emphasis on communication technology in future homes and smartbuildings. Embedded systems are one of the origins of these three areas and they provide a major part of the necessary technology. **Embedded systems are information processing systems that are embedded into a larger product** and that are normally not directly visible to the user. Examples of embedded systems include information processing systems in telecommunication equipment, in transportation systems, in fabrication equipment and in consumer electronics.

Common characteristics of these systems are the following:

- Embedded systems are connected to the physical environment through **sensors** collecting information about that environment and **actuators**[1] controlling that environment.

- Embedded systems have to be **dependable**.

  Many embedded systems are safety-critical and therefore have to be dependable. Nuclear power plants are an example of extremely safety-critical systems that are at least partially controlled by software. Dependability is, however, also important in other systems, such as cars, trains, airplanes etc. A key reason for being safety-critical is that these systems are directly connected to the environment and have an immediate impact on the environment.

Dependability encompasses the following aspects of a system:

1 **Reliability:** Reliability is the probability that a system will not fail.

2 **Maintainability:** Maintainability is the probability that a failing system can be repaired within a certain time-frame.

3 **Availability:** Availability is the probability that the system is available. Both the reliability and the maintainability must be high in order to achieve a high availability.

4 **Safety:** This term describes the property that a failing system will not cause any harm.

5 **Security:** This term describes the property that confidential data remains confidential and that authentic communication is guaranteed.

- Embedded systems have to be **efficient**. The following metrics can be used for evaluating the efficiency of embedded systems:

1 **Energy:** Many embedded systems are mobile systems obtaining their energy through batteries. According to forecasts [SEMATECH, 2003], battery technology will improve only at a very slow rate. However, computational requirements are increasing at a rapid rate (especially for multimedia applications) and customers are expecting long run-times from their batteries. Therefore, the available electrical energy must be used very efficiently.

2 **Code-size:** All the code to be run on an embedded system has to be stored with the system. Typically, there are no hard discs on which code can be stored. Dynamically adding additional code is still an exception and limited to cases such as Java-phones and set-top boxes.

Due to all the other constraints, this means that the code-size should be as small as possible for the intended application. This is especially true for **systems on a chip** (SoCs), systems for which all the information processing circuits are included on a single chip. If the instruction memory is to be integrated onto this chip, it should be used very efficiently.

3 **Run-time efficiency:** The minimum amount of resources should be used for implementing the required functionality. We should be able to meet time constraints using the least amount of hardware resources and energy. In order to reduce the energy consumption, clock frequencies and supply voltages should be as small as possible. Also, only the necessary hardware components should be present. Components which do not improve the worst case execution time (such as many caches or memory management units) can be omitted.

4 **Weight:** All portable systems must be of low weight. Low weight is frequently an important argument for buying a certain system.

5 **Cost:** For high-volume embedded systems, especially in consumer

electronics, competitiveness on the market is an extremely crucial issue, and efficient use of hardware components and the softwaredevelopment budget are required.

- These systems are **dedicated towards a certain application**.

  For example, processors running control software in a car or a train will always run that software, and there will be no attempt to run a computer game or spreadsheet program on the same processor. There are mainly two reasons for this:

  1 Running additional programs would make those systems less dependable.

  2 Running additional programs is only feasible if resources such as memory are unused. No unused resources should be present in an efficient system.

- Most embedded systems do not use keyboards, mice and large computer monitors for their user-interface. Instead, there is a **dedicateduser-interface** consisting of push-buttons, steering wheels, pedals etc. Because of this, the user hardly recognizes that information processing is involved. Due to this, the new era of computing has also been characterized by the disappearing computer.

- Many embedded systems must meet **real-time constraints**. Not completing computations within a given time-frame can result in a serious loss of the quality provided by the system (for example, if the audio or video qual- ity is affected) or may cause harm to the user (for example, if cars, trains or planes do not operate in the predicted way). *A time-constraint is called* ***hard*** *if not meeting that constraint could result in a catastrophe* [Kopetz, 1997]. All other time constraints are called**soft**.

  Many of today’s information processing systems are using techniques for speeding-up information processing *on the average*. For example, caches improve the average performance of a system. In other cases, reliable communication is achieved by repeating certain transmissions. For example, Internet protocols typically rely on resending messages in case the original messages have been lost. On the average, such repetitions result in a (hopefully only) small loss of performance, even though for a certain message the communication delay can be orders of magnitude larger than the normal delay. In the context of real-time systems, arguments about the average performance or delay cannot be accepted. **A guaranteed system response has to be explained without statistical arguments** [Kopetz, 1997].

- Many embedded systems are **hybrid systems** in the sense that they include analog and digital parts. Analog parts use continuous signal values incontinuous time, whereas digital parts use discrete signal values in discrete time.

- Typically, embedded systems are **reactive systems**. They can be defined as follows: *A reactive system is one that is in continual interaction with its environment and executes at a pace determined by that environment* [Berge´ et al., 1995]. Reactive systems can be thought of as being in a certain state, waiting for an input. For each input, they perform some computation and generate an output and a new state. Therefore, automata are very good models of such systems. Mathematical functions, which describe the problems solved by most algorithms, would be an inappropriate model.

- Embedded systems are **under-represented in teaching and in public discussions**. *Embedded chips aren't hyped in TV and magazine ads* ...[Ryan, 1995]. One of the problems in teaching embedded system design is the equipment which is needed to make the topic interesting and practical. Also, real embedded systems are very complex and hence difficult to teach.

Due to this set of common characteristics (except for the last one), it does make sense to analyze common approaches for designing embedded systems, instead of looking at the different application areas only in isolation.

Actually, not every embedded system will have all the above characteristics. We can define the term "embedded system" also in the following way: *Information processing systems meeting most of the characteristics listed above are called **embedded systems***. This definition includes some fuzziness. However, it seems to be neither necessary nor possible to remove this fuzziness.

Most of the characteristics of embedded systems can also be found in a recently introduced type of computing: ubiquitous or pervasive computing, also called ambient intelligence. The key goal of this type of computing is to make information available *anytime, anywhere*. It does therefore comprise communication technology. Fig. 1.1 shows a graphical representation of how ubiquitous computing is influenced by embedded systems and by communication technology.

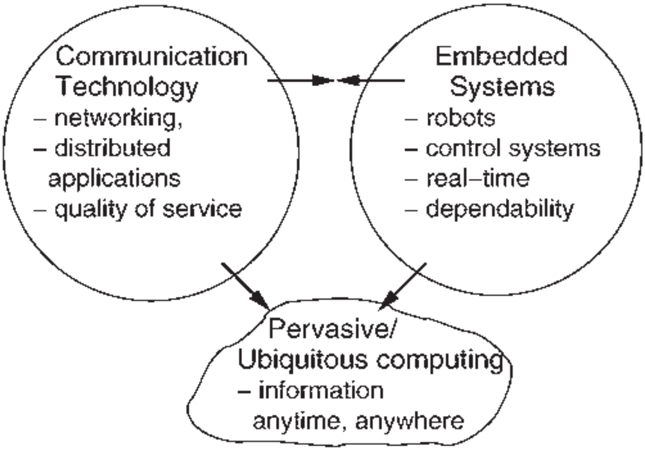

Figure 1.1. Influence of embedded systems on ubiquitous computing

For example, ubiquitous computing has to meet real-time and dependability requirements of embedded systems while using fundamental techniques of communication technology, such as networking.

1.2 Application areas

The following list comprises key areas in which embedded systems are used:

- **Automotive electronics:** Modern cars can be sold only if they contain a significant amount of electronics. These include air bag control systems, engine control systems, anti-braking systems (ABS), air-conditioning, GPS- systems, safety features, and many more.

- **Aircraft electronics:** A significant amount of the total value of airplanes is due to the information processing equipment, including flight control systems, anti-collision systems, pilot information systems, and others. Dependability is of utmost importance.

- **Trains:** For trains, the situation is similar to the one discussed for cars and airplanes. Again, safety features contribute significantly to the total value of trains, and dependability is extremely important.

- **Telecommunication:** Mobile phones have been one of the fastest growing markets in the recent years. For mobile phones, radio frequency (RF) design, digital signal processing and low power design are key aspects.

- **Medical systems:** There is a huge potential for improving the medical service by taking advantage of information processing taking place within medical equipment.

- **Military applications:** Information processing has been used in military equipment for many years. In fact, some of the very first computers analyzed military radar signals.

- **Authentication systems:** Embedded systems can be used for authentication purposes.

  For example, advanced payment systems can provide more security than classical systems. The SMARTpen [IMEC, 1997] is an example of such

  an advanced payment system (see fig. 1.2).

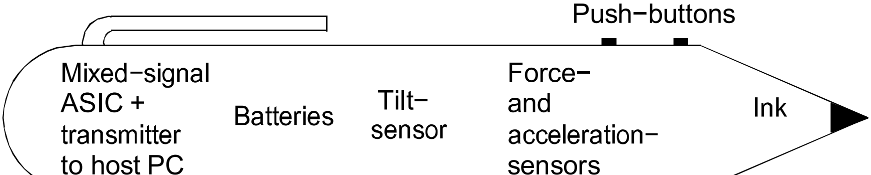
Push−buttons
Mixed−signal ASIC + transmitter to host PC
Batteries
Tilt− sensor
Force− and acceleration− sensors
Ink

Figure 1.2. SMARTpen

The SMARTpen is a pen-like instrument analyzing physical parameters while its user is signing. Physical parameters include the tilt, force and acceleration. These values are transmitted to a host PC and compared with information available about the user. As a result, it can be checked if both the image of the signature as well as the way it has been produced coincide with the stored information.

Other authentication systems include finger print sensors or face recognition systems.

- **Consumer electronics:** Video and audio equipment is a very important sector of the electronics industry. The information processing integrated into such equipment is steadily growing. New services and better quality are implemented using advanced digital signal processing techniques.

Many TV sets, multimedia phones, and game consoles comprise high-performance processors and memory systems. They represent special cases of embedded systems.

- **Fabrication equipment:** Fabrication equipment is a very traditional area in which embedded systems have been employed for decades. Safety is very important for such systems, the energy consumption is less a problem. As an example, fig. 1.3 (taken from Kopetz [Kopetz, 1997]) shows a container connected to a pipe. The pipe includes a valve and a sensor. Using the readout from the sensor, a computer may have to control the amount of liquid leaving the pipe.

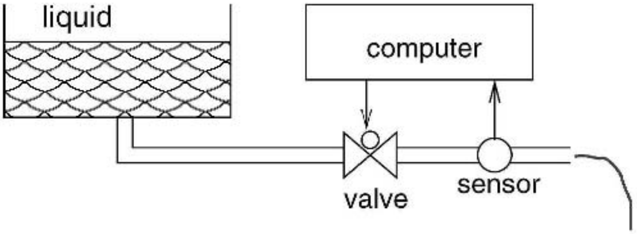

Figure 1.3. Controlling a valve

The valve is an example of an actuator (see definition on page 2).

- **Smart buildings:** Information processing can be used to increase the comfort level in buildings, can reduce the energy consumption within buildings, and can improve safety and security. Subsystems which traditionally were unrelated have to be connected for this purpose. There is a trend towards integrating air-conditioning, lighting, access control, accounting and distribution of information into a single system. For example, energy can be saved on cooling, heating and lighting of rooms which are empty. Available rooms can be displayed at appropriate places, simplifying ad-hoc meetings and cleaning. Air condition noise can be reduced to a level required for the actual operating conditions. Intelligent usage of blinds can optimize lighting and air-conditioning. Tolerance levels of air conditioning subsystems can be increased for empty rooms, and the lighting can be automatically reduced. Lists of non-empty rooms can be displayed at the entrance of the building in emergency situations (provided the required power is still available).

  Initially, such systems will mostly be present only in high-tech office buildings.

- **Robotics:** Robotics is also a traditional area in which embedded systems have been used. Mechanical aspects are very important for robots. Most of the characteristics described above also apply to robotics. Recently, some new kinds of robots, modeled after animals or human beings, have been designed. Fig. 1.4 shows such a robot.

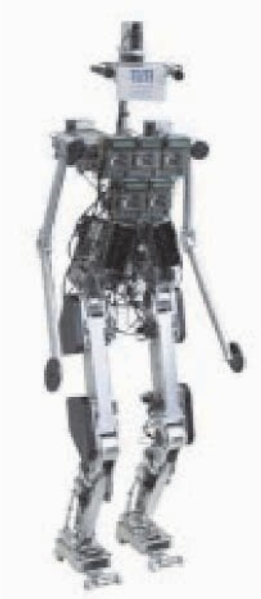

Figure 1.4. Robot "Johnnie" (courtesy H. Ulbrich, F. Pfeiffer, Lehrstuhl für Angewandte Mechanik, TU München), ©TU München

This set of examples demonstrates the huge variety of embedded systems. Why does it make sense to consider all these types of embedded systems in one book? It makes sense because information processing in these systems has many common characteristics, despite being physically so different.

1.3 Growing importance of embedded systems

The size of the embedded system market can be analyzed from a variety of perspectives. Looking at the number of processors that are currently used, it has been estimated that about 79% of all the processors are used in embedded systems[2]. Many of the embedded processors are 8-bit processors, but despite this, 75% of all 32-bit processors are integrated into embedded systems [Stiller, 2000]. Already in 1996, it was estimated that the average American came into contact with 60 microprocessors per day [Camposano and Wolf, 1996]. Some

[2] Source: Electronic design.

high-end cars contain more than 100 processors[3] . These numbers are much larger than what is typically expected, since most people do not realize that they are using processors. The importance of embedded systems was also stated by journalist Mary Ryan [Ryan, 1995]:

... embedded chips form the backbone of the electronics driven world in which we live. ... they are part of almost everything that runs on electricity.

According to quite a number of forecasts, the embedded system market will soon be much larger than the market for PC-like systems. Also, the amount of software used in embedded systems is expected to increase. According to Vaandrager, *for many products in the area of consumer electronics the amount of code is* ***doubling every two years*** [Vaandrager, 1998].

Embedded systems form the basis of the so-called **post-PC era**, in which information processing is more and more moving away from just PCs to embedded systems.

The growing number of applications results in the need for design technologies supporting the design of embedded systems. Currently available technologies and tools still have important limitations. For example, there is still a need for better specification languages, tools generating implementations from specifications, timing verifiers, real-time operating systems, low-power design techniques, and design techniques for dependable systems. This book should help teaching the essential issues and should be a stepping stone for starting more research in the area.

The structure of this book corresponds to that of a simplified design information flow for embedded systems, shown in figure 1.5.

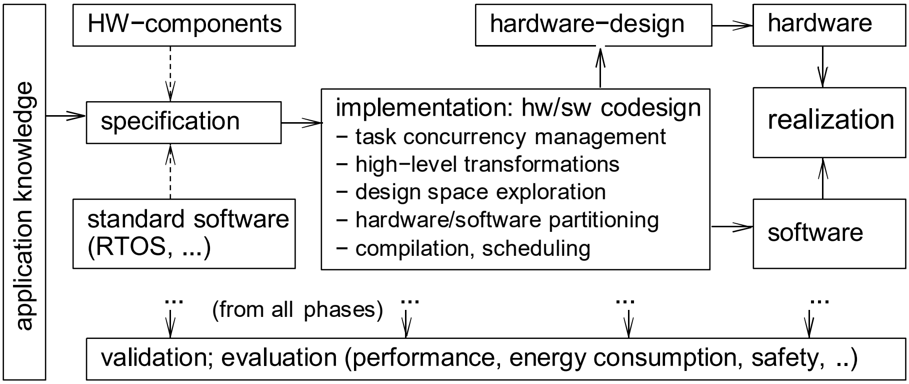
application knowledge
HW−components
hardware−design
hardware
specification
implementation: hw/sw codesign
− task concurrency management
− high−level transformations
− design space exploration
− hardware/software partitioning
− compilation, scheduling
realization
standard software
(RTOS, ...)
software
...
(from all phases)
...
...
...
validation; evaluation (performance, energy consumption, safety, ..)

Figure 1.5. Simplified design information flow

The design information flow starts with ideas in people's heads. These ideas must be captured in a design specification. In addition, standard hardware and software components are typically available and should be reused whenever possible.

Design activities start from the specification. Typically, they involve a consideration of both hardware and software, since both have to be taken into account for embedded system design. Design activities comprise mapping operations to concurrent tasks, high-level transformations (such as advanced loop transformations), mapping of operations to either hardware or software (called hardware/software partitioning), design space exploration, compilation, and scheduling. It may be necessary to design special purpose hardware or to optimize processor architectures for a given application. However, hardware design is not covered in this book. Standard compilers can be used for the compilation. However, they are frequently not optimized for embedded pro- cessors. Therefore, we will also briefly cover compiler techniques that should be applied in order to obtain the required efficiency. Once binary code has been obtained for each task, it can be scheduled precisely. Final software and hardware descriptions can be merged, resulting in a complete description of the design and providing input for fabrication.

At the current state of the art, none of the design steps can be guaranteed to be correct. Therefore, it is necessary to validate the design. Validation consists of checking intermediate or final design descriptions against other descriptions. Evaluation is another activity that is required during various phases of the de- sign. Various properties can be evaluated, including performance, dependabil- ity, energy consumption, manufacturability etc.

Note that fig. 1.5 represents the flow of **information about the design object**. The sequence of design **activities** has to be consistent with that flow. This does not mean, however, that design activities correspond to a simple path from ideas to the final product. In practice, some design activities have to be repeated. For example, it may become necessary to return to the specification or to obtain additional application knowledge. It may also become necessary to consider additional standard operating systems if the initially considered operating system cannot be used for performance reasons.

Consistent with the design information flow, this book is structured as fol- lows: in chapter 2, we will discuss specification languages. Key hardware components of embedded systems will be presented in chapter 3. Chapter 4 is devoted towards the description of real-time operating systems, other types of such *middleware*, and standard scheduling techniques. Standard design tech- niques for implementing embedded systems - including compilation issues - will be discussed in chapter 5. Finally, validation will be covered in the last chapter.