IL2206 EMBEDDED SYSTEMS

Reflection Assignment 1

YANG WANG

2021-09-12

1 Reflection Assignments for Seminar 1

1.1 Give three new examples (not mentioned previously in the lectures or lecture notes) for typical saftey-critical embedded systems from different application domains.

notes) for typical safety-critical embedded systems from different application domains. Safety-critical embedded systems are a very important embedded system. It often controls more critical components. If something goes wrong, it will bring serious consequences, including loss of equipment or property, loss of personal health and safety, and damage to the environment. The faults here include various errors, such as uninitialized variables, logical errors in the code, and physical damage due to various problems. Failures may not cause serious problems, but as they accumulate level by level in the system, they can cause serious problems. So small problems can quickly become life-threatening problems.

There are many such embedded systems, and the more common ones are flight control systems on airplanes and nuclear reactor control systems. There are also some more common ones, such as elevator door controllers in daily life.

1.2 What do you consider the main problems and challenges in the design process of safety-critical embedded systems?

The most critical issue in the design process of safety-critical embedded systems is to ensure the safety and stability of the system. The focus is on safety, not on the speed of response and whether it is powerful. In most of the design process, people will mainly pay attention to whether the designed software has strong performance and complete functions. In safety-critical embedded systems, the most important thing is to ensure the safe and stable operation of the system. In the design process of safety-critical embedded systems, people must find all possible harm situations and the ways in which harm can occur under these situations. While ensuring safety, it is also necessary to ensure that the functions of the system can operate stably. Achieving these requirements will greatly increase the complexity of the system.

A challenge in the design process is how to ensure the safety of the system when the system fails. That is to say, in any case where the system is used in accordance with the system design instructions, if there is a problem, including misoperation, etc., no safety problem can occur. And even if the system fails, the user's declaration must be secured to the greatest extent possible. For example, when using a washing machine, if you open the lid, the drum should stop running immediately. This situation is relatively simple, but for some more complex safety-critical embedded systems, such as the control system of a nuclear power plant, it is very complicated to figure out all the situations. For example, in the Chernobyl nuclear accident in 1986, because of the test plan, engineers reduced the power of the equipment from 1600MW to 700MW, which caused the reactor to be poisoned (the accumulation of xenon 135), and the power was reduced to 30MW. The duty officer judged that the reactor was poisoned and suggested to shut down, wait 24 hours for the xenon 135 to be consumed by the reaction, and then gradually increase the reactor power. The

chief engineer was worried that it would affect the daily operation and insisted on restoring the power to 700MW. He lifted a series of safety measures, reduced the number of control rods in the reactor to only six, and wanted to quickly increase the reactor power. When the accumulated xenon 135 was exhausted, the power of the reactor controlled by only 6 control rods rose rapidly. After a few seconds, the power quickly rose to 33000MW, which was about ten times the design power, and eventually caused a steam explosion. So for this kind of complicated control system, it is very complicated to understand all the situations. This makes safety requirements far more than functional requirements and greatly increases the complexity of safety-critical embedded systems.

1.3 What do you consider the most relevant properties for an embedded computing platform designed for safety-critical systems. Motivate! What does this mean for the characteristics of the individual components (processor, memory system, interconnection network, peripherals) that build the platform for a safety-critical embedded systems?

The most important thing for safety-critical embedded systems is not function and speed, but safety and stability. Therefore, ensuring stability should be the primary requirement for the platform of this type of embedded system. This is also true in most cases in reality. For example, most computer systems used in spacecraft. The computer of the Apollo Moon Landing Project: Apollo Guidance Computer, 16-bit, clocked at 2MHz, contains a 2K RAM and 36K ROM, which is worse than the current home calculator. Hubble's computer: DF-224, clock frequency is 1.25MHz, 32K memory, later replaced with 25MHz 486CPU in 1999, and the performance of the most common civilian mobile phones is still very different. The CPU of the Curiosity Mars rover: 256M RAM, 2G Flash external storage, and the CPU capacity can reach 400MIPS. It is already a very good configuration in the spacecraft, but this configuration is also used in a home computer ten years ago. level. It is because for this type of system, it is most important to ensure that they can operate safely and stably. Once they have problems, the world will threaten the lives of astronauts and cause a lot of losses. In order to ensure safety, a lot of work can only be invested in designing measures to ensure safety. Therefore, for this type of system platform, security and stability are the most important.

The processor is a very important part of the embedded system platform. When choosing a processor for a safety-critical embedded system, it is important to know whether it has been verified and meets the safety requirements of the system to be used. Unlike processors used in consumer products such as mobile phones, validating processors for safety-critical systems requires more knowledge and time. A large number of security verifications, such as parity check, error correction code, logic correction, watchdog timer, hardware stack protection, etc. are all required. So whether to pass the safety verification and meet the system requirements is the requirement of a safety-critical embedded system processor.

Stability is also the most important for the memory, internetwork and peripherals of safety-critical embedded systems. For memory, compared to the equipment used by ordinary consumers, there should be enough redundancy to ensure that when some of them fail, the memory is still sufficient. Also design some protection measures, such as memory protection unit. It can define and assign some access attributes to combine different schemes to protect against misbehavior of the code in key programs. The memory of safety-critical embedded systems is limited in terms of hardware and flexibility, but is more stable. The same is true for the interconnection network. The speed of the interconnection network is not the primary consideration, and the design focus is to ensure that the information exchange can proceed smoothly. In most designs, multiple disjoint paths are added between input and output. The easiest way is to add a redundant stage of switches to obtain redundant paths. These measures will increase system security, but will lead to system performance degradation. If the task is not completed before the deadline, it will also cause catastrophic consequences. Although there are various algorithms to optimize its structure, this is still a big problem. Therefore, the key to choosing an interconnection network is to make a balance between security and speed, and to choose the architecture and algorithm most suitable for this system. For peripherals, it is still important to ensure the stable provision of information. For the peripherals of safety-critical systems, they need to work stably in harsh environments, or they must not fail in long-term use. For example, vari-

ous detectors outside the spacecraft are working in harsh environments, and cosmic rays and huge changes in temperature will affect its working results. The detector at the entrance of the elevator will be used for a long time and cannot fail, otherwise it will cause injury to the user. Many systems still avoid the above situation by adding redundant equipment to the greatest extent. Although it will increase the cost, it is necessary for safety-critical embedded systems.

1.4 Safety-critical systems execute a lot of software. How can a time-predictable behaviour of the software be supported by the hardware?

Various methods can be used to predict the time of the task. A large number of data sets can be input and run to get the best and worst-case execution time. This is the most commonly used test method for task performance on target hardware. Another method is static timing analysis, which uses mathematical models abstracted by hardware to predict the execution time of tasks. This method may cover all possible situations of data.