FPGAs enable development of safety-critical embedded systems

By Michael Henze, MEN Mikro Elektronik

Electronics are always safety-critical when faults or failures endanger human lives or can cause major environmental and/or material damage. Safety-critical systems must therefore always function reliably. Essential functional safety can be realized with FPGA logic.

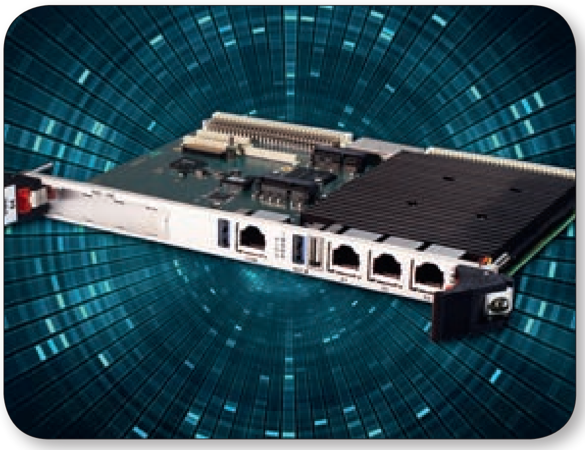
Figure 1. A board with FPGA implementation is the 6U high-performance SBC with 16 cores and FPGA-based VMEbus interface. It is used in the LHC of the CERN Institute and is not only technically considered a safe board among developers.

■ The specifications for safety-critical embedded systems are often given for specific industries and are subject to strict standards. There is no scope for errors in hardware or software. Typical applications can be found in trains, buses, ships and aircraft as well as in more complex applications in industrial automation, medical technology or the energy industry. The factor of functional safety must be accorded particular importance in this context. But is it possible to design a system in such a way that it absorbs all known risks through its design? This includes accidental failures caused by component failures, EMC influences or cosmic radiation as well as possible design errors that can be avoided during development by appropriate processes. And is it possible to certify systems according to the safety standards of the different markets if most of the available standard components do not cover these standards by default? A self-generated verification is usually very time-consuming - especially if the components are complex. In addition, it is sometimes only possible in cooperation with the component manufacturer, as this also requires an insight into the production processes. But do they go along with it? Often not, because functional safety is a niche market for most suppliers of standard components used in embedded computers. But how can this dilemma be overcome and yet functionally safe systems developed? A very good alternative to the retrospective testing of standard components, as prescribed by the EASA certification memorandum EASA CM - SWCEH - 001, is the use of Field Programmable Gate Arrays, or FPGAs for short, in which the function is mapped in a new and compliant manner to the respective safety standards. This solution is ideally suited for the exact fulfilment of the safety-critical requirements of the respective industries. It even opens the possibility of efficiently implementing even highly customer-specific designs with small batch sizes and offering them at attractive prices. This makes them the suited basis for ensuring application-specific functional safety. The advantage of FPGAs is that it is not necessary to redevelop everything. Rather, functional IP building blocks can be combined as required, which saves both costs and development time. This is possible not only for the design of a specific FPGA, but also for the design of a board or system with several FPGAs, which receive their highly individual functionality via the respective FPGA adaptations.

However, before a safety-critical design can be qualified and certified, a proof of its behavior in the event of an error must be provided. This is comparatively easy with development tools for FPGAs. In the virtual development environment for FPGAs, even serious or complicated errors can be provoked to test the error behavior of the system or to check whether the system has a defined error behavior. This form of simulation is not common in software, but is part of the basic tool for FPGA design - it is also used for so-called normal developments, which do not have to meet any functional safety requirements. In this respect, FPGAs do not require any additional toolchain effort. The simulation can also be used not only to prove the correct error behavior, but also to prove the correct implementation of a function. It is therefore possible to create complete simulation reports, which can then be submitted to TÜV or other certification service providers, for example.

Monitoring of proper conditions is also extremely important in the safety-critical area, as this is the only way to detect failures and initiate appropriate actions. For example, temperatures, the functionality of components or the receiving of data must be monitored and analyzed to achieve a safe state in the event of deviations from setpoint values - such as stopping a machine or stopping a train in a controlled manner. However, finished components for the connection of input and output units - such as serial interfaces or GPIOs - rarely contain such monitoring functions as are required for functional safety, for example in accordance with EN 50129 for railways or in accordance with IEC 61508 for electronic systems with a safety function. Such functions can also be mapped very efficiently in FPGAs if there are no suitable microcontrollers. The integration of such monitoring functions into FPGAs also offers the advantage over microcontrollers that they are freely configurable and can be adapted to the application.

The saying never change a running system particularly applies to functionally safe systems. On the one hand, the costs for verifying functional safety in accordance with the standards are high and must be reproduced every time a change is made - which means that they are very cost-intensive. On the other hand, when changes are made, there is always the risk of installing a new error. For this reason, systems have been used for decades without revision, especially in rail transport and aviation. This also necessitates an obsolescence strategy for components, since standard components for industry are rarely available for more than 5 to 10 years. FPGAs offer decisive advantages here. The function is not in a dedicated component, but in the programming itself. As a result, component discontinuations are comparatively easy to handle, since the code can be ported to new FPGAs with identical functionality. Project lifetimes of more than 30 years are no problem, even if the FPGA manufacturer has to be changed. This also provides independence from a certain supplier.

With FPGAs, it is also always possible to integrate additional functions at a later date - for example, to upgrade the system. This flexibility naturally also has an effect at the beginning of the product lifecycle: if some of the hardware functions are implemented in FPGAs, this part can be tested parallel to further development. Such a procedure saves time during subsequent commissioning and testing of the entire system.

One of the most common requirements, especially in the safety-critical area, is the support of extended temperature ranges - usually from -40 to +85°C. Even here, there are often problems to find appropriately qualified standard components. However, at the latest with an extremely extended temperature range of -55 to +125°C, it becomes considerably more difficult or impossible to get components for the various hardware functions. However, FPGAs offer a sufficiently wide range for these extreme temperatures.

The most important strategy for making a system less risk-prone is redundancy - i.e. the functionally identical multiplication of important components in order to be able to compensate failures of individual components without any problems. A component that paralyses the entire system due to its failure is called Single Point of Failure (SPOF). Any important building block can be such an SPOF. In aerospace applications, for example, memory errors caused by cosmic radiation are a problem. These lead to effects such as Single Event Upsets (SEU) or Multi-Bit Upsets (MBU), where one or more bits in memory elements jump from 0 to 1 or vice versa. If critical components such as a CPU in multiple redundancy with voting are present, this increases functional safety and availability. Such redundancies including voting functionality can be built up with FPGAs, which offers the advantage that this logic can be easily copied in every instance by copying and pasting the IP logic. In the FPGA, this redundancy is repeated again in order to be able to complete its calculation, if an FPGA flip/flop fails. As a result, an almost SEU-immune implementation can be realized, when using a flash-based FPGA.

In the safety-critical environment, predictable execution times are also required in addition to reliability. The system must react to an external event in a defined time, even in the worst case. However, typical computer architectures use interrupts and DMA topologies that can negatively affect the response times of individual tasks when

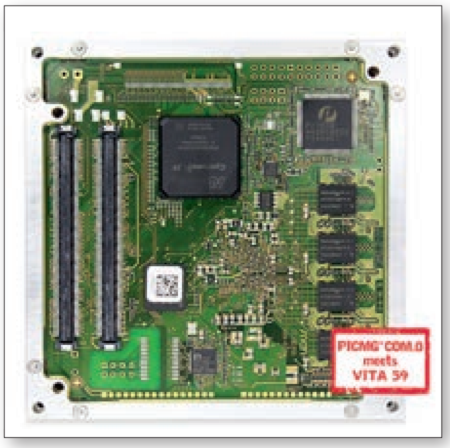

Figure 2. The standard FPGA on the Rugged-COM Express module CC10C with ARM i.MX6 processor provides flexible, customizable I/Os.

another task requests the same resources. The required deterministic behavior - i.e. exactly predictable in terms of time - is then difficult to achieve. For this reason, such solutions are not used for hard real-time requirements. FPGAs support real-time capability, however, since they are built in parallel. This means that the different processes do not compete with each other, but take their own predefined path, which is not disturbed by other events. This makes it much easier to ensure deterministic real-time capability with clearly defined behavior over time.

In the context of functional safety in times of the Internet of Things, Industry 4.0 and Mobility 4.0 one sooner or later also comes across security in the sense of protection against manipulation. Here too, FPGAs offer many possibilities to protect the application against manipulation, unauthorized access or duplication of data. For example, a unique key can be programmed in the FPGA. There it is stored in encrypted form in a non-volatile memory. This key can then ensure that data can only be read by applications and people who know this key. The key can also be used to identify the device communication with other devices. Because it is hardware-based, it can no longer be manipulated on the software side, which always uniquely identifies the device. A code, which is implemented in hardware cannot be copied as easily as software. In this way, an FPGA can assume valuable security functions that go much further than, for example, a Trusted Platform Module. They even have a very specific advantage over standard solutions, because if they are programmed individually, they are much less susceptible to hacking.

In addition to all these advantages of FPGAs, there are also limits to their use. On the one hand, there are the costs. They are, of course, more expensive than standard components manufactured in large batches. FPGAs can also only be used to a limited extent to implement complex solutions, because from a certain level of functionality it is better to switch to a combination of software and hardware, since microcontrollers and application processors already have a fundamental logic including various I/O interfaces off-the-shelf, which would have to be developed for FPGAs.

Nevertheless, you can do a lot with FPGAs of course. For example, x86 logic already exists in FPGAs. But we are still a long way from reproducing the entire software logic that exists for x86 in FPGAs. Advantages and disadvantages must therefore be weighed up depending on the application and existing standard components. In principle, FPGAs offer flexible and safe alternatives for almost all challenges, where on the hardware or software side you would long ago have come up against development limits. Many applications that rely on functional safety cannot do without FPGA logic today. Companies such as MEN Mikro Elektronik specialize in such FPGA-based platforms for safety-critical embedded systems and are also familiar with the requirements of specific industries. In some industries it is common practice, for example, that FPGA development is not considered in the standards, but solution providers today like to rely on the simulation results of FPGA development tools to save documentation effort for certification.

In spite of all the effort involved in development, FPGAs can also save considerable time in terms of certification expenses, which sometimes cost more than the development itself. And there are already many MEN function blocks that are used in certified applications. On the one hand, they fulfill core functions of the boards. On the other hand, they represent specific I/O functions. For example, MEN IP building blocks for FPGAs include: graphic and touch display controllers, fieldbus interfaces such as CAN and MVB, different UARTs like RS232 or RS485, Ethernet and HDLC interfaces, SRAM and flash memory controllers, and GPIO, digital I/O, counter, quadrature decoder and PWM functions.

All these IP cores can be combined with cores provided by Altera (Avalon Bus) or the Open Cores Community (Wishbone Bus). Bridges developed by MEN from Wishbone-to-Avalon and Avalon-to-Wishbone round off the application-ready FPGA logic range, which is continuously being further developed and can of course also be adapted and or extended to customer-specific requirements. ■