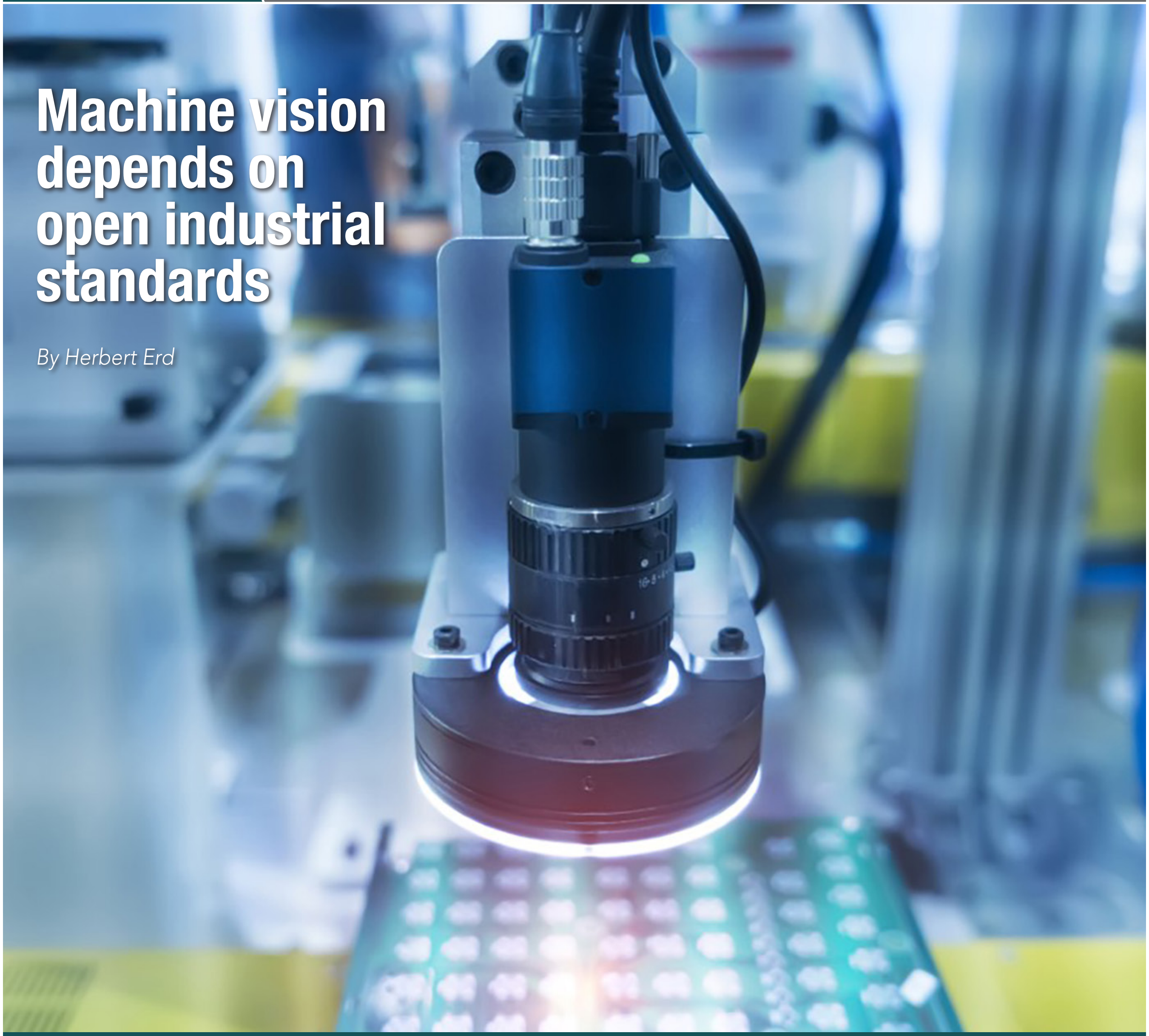

Machine vision depends on open industrial standards

By Herbert Erd

Machine vision is widely used in automation for failure detection, quality inspection, medical applications, transportation, and public safety.

For the past 15-plus years, optical inspection – also called machine vision – has expanded worldwide, leaving its footprint in different vertical markets. These days machine vision is mainly performed in three ways: offline, where parts are removed from the manufacturing process; near-line, during which parts are inspected close to the manufacturing process, leading to a quicker return to the production line; and online, where components are inspected without removing them from the manufacturing process.

Machine vision is used in automation for failure detection and quality inspection, while in medical markets its main purpose is picture analysis and merging. In addition, the public safety and transportation industries use machine-vision technology for camera link aggregation, filtering, analytics, compression, and recoding.

Input sensors used in machine-vision applications can consist of different kinds of cameras for the visible or invisible spectrum; they may also include ultrasonic and X-ray laser sensors, or even a combination of any of these.

The amount of data per input link (bandwidth) as well as the number of links are growing with every new camera generation, because of 3D capabilities as well as real-time requirements with low latency.

In order to terminate and process such data streams from different sensor types, hardware based on FPGAs has become very popular for image processing, particularly when latency matters. On top of existing applications, growing demands from AI [artificial intelligence] as well as the IoT [Internet of Things] will increase the need for processing power and intelligent interconnects. Any new feature has an additional impact on data input, throughput, processing, and data output, which means that every development team should consider several aspects before choosing any specific hardware platform.

Developers' headache

For machine vision, two types of software developers – with completely different backgrounds – are involved for solutions based on FPGAs. First off, digital image-processing engineers or C programmers may have limited hardware skills; in addition, real FPGA programmers can be hard to find. Developers in this realm must understand the overall effects of the software requirements on the suitable hardware platform for a system in production.

As a first approach, we often see FPGA-based PCIe cards in a PC, actually still one of the most common development environments with a simple master/slave architecture.

However, the following simple calculation might highlight the limitation of such a development environment: In a typical machine-vision application, a camera generates a data stream of 672 Mbit/sec (2.4 Mpix/frame x 35 frames/s x 8 bit/pix = 672 Mbit/sec).

This data stream needs to be terminated by a processing resource whereby the image gets further manipulated and processed. In order to transport the data stream from the camera to the processing engine, in this case a 1 GbE link on GiGE (GbE vision) is sufficient.

Upgrading the camera to a more enhanced version could easily increase the data stream from 672 Mbit/sec to 4.5 Gbit/sec to be transported (5.2 Mpix/frame x 107 frames/s x 8 bit/pix = 4.5 Gbit/sec).

So it's fair to assume that with any new or additional sensor being added to the system, the data stream between camera and processing device increases significantly. Therefore, one can conclude that a growing demand for bandwidth will soon drive the chosen processing engine to its limit. As a consequence, a system design with a fixed and dedicated number of I/O ports and processing engines – either CPUs or FPGAs by design – can soon generate major issues for the performance and/or ability to upgrade of systems deployed to the field. This is a typical development issue.

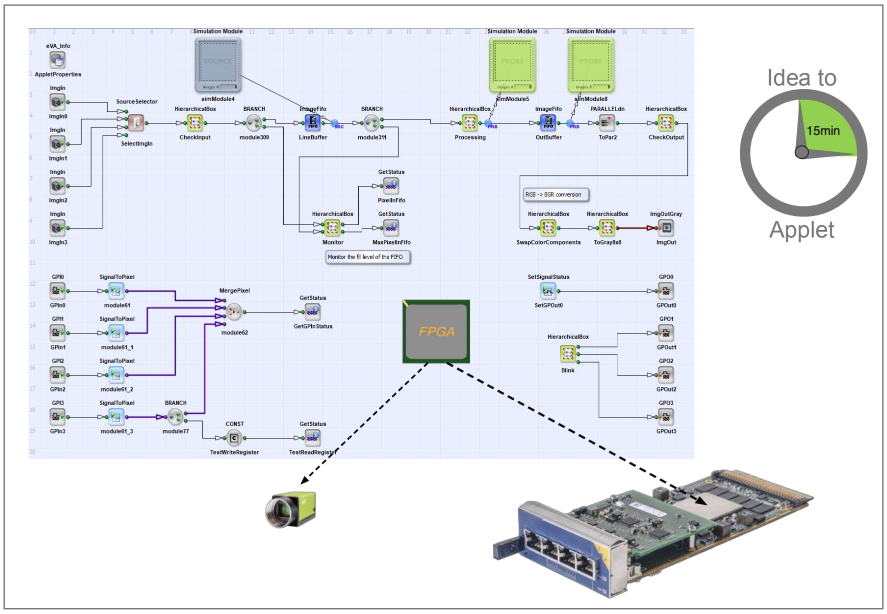

Figure 1 | Visual Applets enables creation and programming of image-processing algorithms on FPGA hardware without the need for HDL knowledge.

Machine vision based on MicroTCA for embedded applications

N.A.T. developed its distinctive NATvision platform concept based on the open industrial standard MicroTCA for embedded applications in industrial, medical, transportation, and public-safety markets. NATvision enables users to have a complete environment for development amd serial production of sophisticated image- and video-processing applications.

The platform consists of a wide range of different FPGA and CPU boards fully integrated in different proven MicroTCA chassis and is intended to reduce both development costs and time.

It uses a set of preselected software tools with different libraries to optimize FPGA/Arm-based programming for machine vision. For larger systems that need high-speed interconnects between different boards plus timing and trigger signals, MicroTCA's switching and clocking capability provides scalability and flexibility for machine vision applications. Add-on requirements during the system life cycle can also be implemented.

By combining proven hardware components with Visual Applets, NATvision eliminates the need for deep FPGA knowledge. Visual Applets (design software) is a graphical design tool for creating and programming image-processing algorithms on FPGA hardware. Vision algorithms are described by block design operators, available from included libraries. Due to a platform-specific glue logic layer, HDL [Hardware Description Language], skills are not required. This capability enables engineers to focus on the vision algorithm, thus achieving impressive results in very short time. (Figure 1.)

NATvision allows combining any HDL-based logic with software libraries such as OpenCV or Halcon to expand the field of possible applications. Software algorithms can either run on an external CPU (x86, x64) or on the system-on-chip's processing system (Arm).

It is even possible to accelerate software algorithms with an FPGA. This can be done by using high level synthesis (HLS), which compiles C code into HDL language. By using FGPA resources instead of executing algorithms sequentially on a CPU, image processing can be accelerated by multiples of 10. This is mainly archived by rolling out loops into parallel logic structures. Developers will benefit from HLS as it is much easier to describe and test HDL algorithms when they're written in C.

Today, NATvision is available with different FPGAs from Xilinx and Altera including FMC (FPGA mezzanine cards)-based I/O. In order to meet special customer requirements, different processing engines such as x86 and RISC-/ARM-based CPUs are available. (Figure 2.)

What's the benefit?
Based on N.A.T.'s NATvision platform, any development of new algorithms can start immediately and independent of the final system design. Due to NATvision's flexibility, several parameters and solution characteristics can be easily altered during development – for example, amount and types of cameras, sensor resolution, frames per second, processing power, or new functions for the foreseen application – as well as the addition of new and additional output requirements without delaying the application development in any way.

The deployment system for serial production can be designed independently and in parallel to the software development based on a small NATvision development system. Based on MicroTCA's design rules, larger enclosures for additional cards can also be used. High-speed switching capability enables handling whatever number of boards in a flexible manner. If needed, new or more powerful functions can be implemented by adding hardware in spare slots by reconfiguring the board to board communication via the switch. In this way, software tasks also can be split or distributed between different boards.

NATvision provides flexibility in several ways:

- Provides scalability in terms of number and types of boards.
- Provides independence of hardware requirements from software application development
- Provides a graphical software-development tool and enables the use of C and C++ codes instead of HDL
- Enables distribution of application functions afterwards, even for different boards
- Timing and latency aspects are covered by hardware

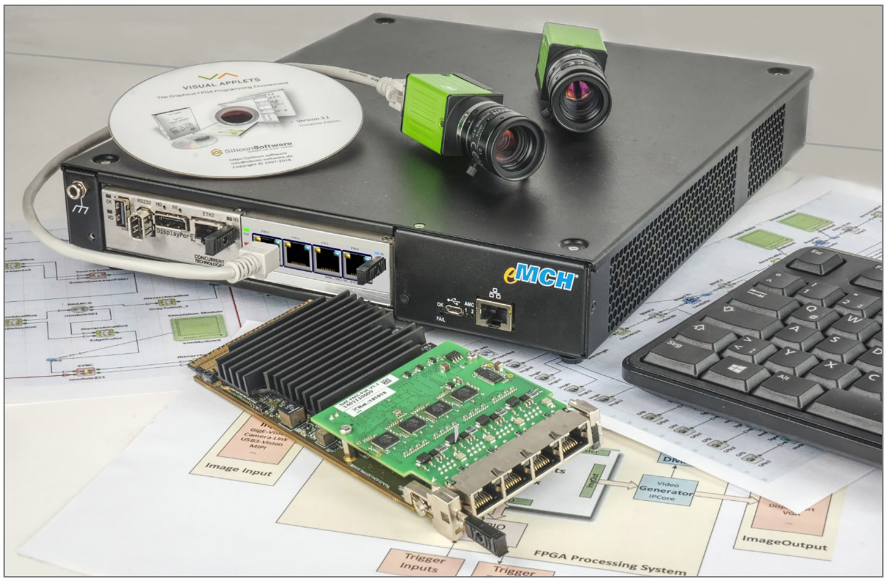

Figure 2 | NATvision is customizable with different FPGAs, software libraries, and more.

- Remote management and system health are also covered as standard functions by design
- On-the-fly support for modular hardware without impact on the used software leads to scalable, flexible, and future-proven systems solutions.

Companies' machine-vision needs can change
Today's machine-vision systems are facing additional and/or changing requirements during the development process already, even after deployment during their lifespan. For industrial applications, an open industrial standard like MicroTCA provides the needed flexibility to extend hardware as needed. NATvision – based on this approach – supports development and deployment and naturally provides software tools for remote upgrades.

NATvision enables a change of hardware if required, independent of whether new I/O requirements show up or additional or more powerful processing cards are needed. As soon as a common clock domain is required, a MicroTCA-based vision system proves its worth, as this scenario enables even distribution of a system across several enclosures located in different areas.

__Herbert Erd__ is the Business Development Manager at N.A.T. With 35 years business experience and 15+ years of experience in business development, Herbert has an extensive technical background in embedded, automation/IoT, and communication technology. Since 2005 he has initiated and designed several major mTCA and ATCA projects for communication and automation applications. He holds an engineering degree from the University of Applied Sciences for Electronic and Radio Communication Academic – grade of graduate engineer – at FH Koblenz.

N.A.T.
www.nateurope.com