Embedded Vision: Highest Efficiency with ARM Cortex-A15 and Cortex-A72

Motivation

Embedded Vision, ARM, Linux – new terms are circulating through the image processing industry. Are real changes coming, or do these new terms describe an evolution of existing technologies? In which direction are the trends going concerning smart cameras or the decentralized vision computers? The following deliberations consider alternatives to the classical x86/Windows solution approach but also show where the classical approach is still sensible.

Embedded – What is That?

In relation to a vision system, "Embedded" means that the vision system is integrated into the application, the device or the machine, being an integral part of it. The complete system can no longer work without the functionality of the vision system. Consider these examples: a reverse vending machine – without image processing, the machine is useless. A machine detecting suspended solids in vials needs image processing as much as a track and trace machine, whose function is the examination of a printed code. The way such an "embedded system" is ultimately realized – whether as a vision sensor, smart camera, or high-performance computer-camera-system – plays a secondary role for the definition. The functionality is what is essential, the interfaces to other components. Also, it is important not to associate Embedded Vision with simple single-board computers (like a commercial Raspberry Pi). These are neither designed for industrial use, nor are they equipped with the computing power and interfaces often necessary for today's demanding application software.

CPU in the Camera or as a Separate Box?

Judging from heat sinks mounted onto smart cameras, it becomes clear: There is not hardly enough space! The heat generation of the CPU is high, mounting conditions (form factor, air flow) are limited. But the conflict shows the transition between smart camera and separate computing power in a compact computer (also fanless). It depends (primarily) on the CPU power dissipation. Another restriction of smart cameras are limited interfaces. A lot of applications require a range of interfaces not only for the camera, but also for I/O and Ethernet. The number of I/Os, as well as Encoder, fieldbus and multi-camera functionality are only some of the parameters relevant for decision making.

Smart Camera CPU: x86, DSP or ARM?

X86 CPUs (ATOM) are rarely built into smart cameras. But why? Usually because the – always critical – form factor increases. A PC requires interfaces like monitor, keypad, mouse – in the case of the smart camera over a separate adapter box. But these devices cannot be operated as elegantly as the following alternatives.

The DSP, especially from the Texas Instruments DaVinci family, can be found in various Vision Sensors. However, the small applications in the cameras "blossom out" to become full-grown applications which do not only perform a few filters. A universal software – not optimized for signal processors – runs faster on an ARM architecture. Moreover, the infrastructure of the ARM processors is more comprehensive and

better – just think of the Linux operating system. Still, DSPs prevail when it comes to the use of a robust real-time operating system. The DSPs are also still available on the chips as coprocessors.

The trend obviously goes in the direction of ARM processors and SoC: System on Chip. The company ARM provides the CPU architecture, whilst the chip developers (TI, NXP, Altera, …) locate their own knowhow around the CPU. In the case of an Altera module, this would be the FPGA, in the case of the chip TI AM5728 used in the IMAGO Smart Camera, these would be a number of other processors for special tasks.

New Performance Class for Smart Cameras

Studying the Cortex-A CPUs of the company ARM shows that these are subdivided into ultra-high-efficient (Cortex-A5; -A7; -A32; -A35), high-efficient (Cortex-A8; -A9; A53) and high-performance (Cortex-A15; -A17; -A57; A-72; -A73). For a smart camera, the CPU power consumption is critical– however, the engineer wants a computing power as high as possible. The IMAGO team has succeeded to integrate a Dual-Cortex-A15 from the high-performance class into a compact camera housing.

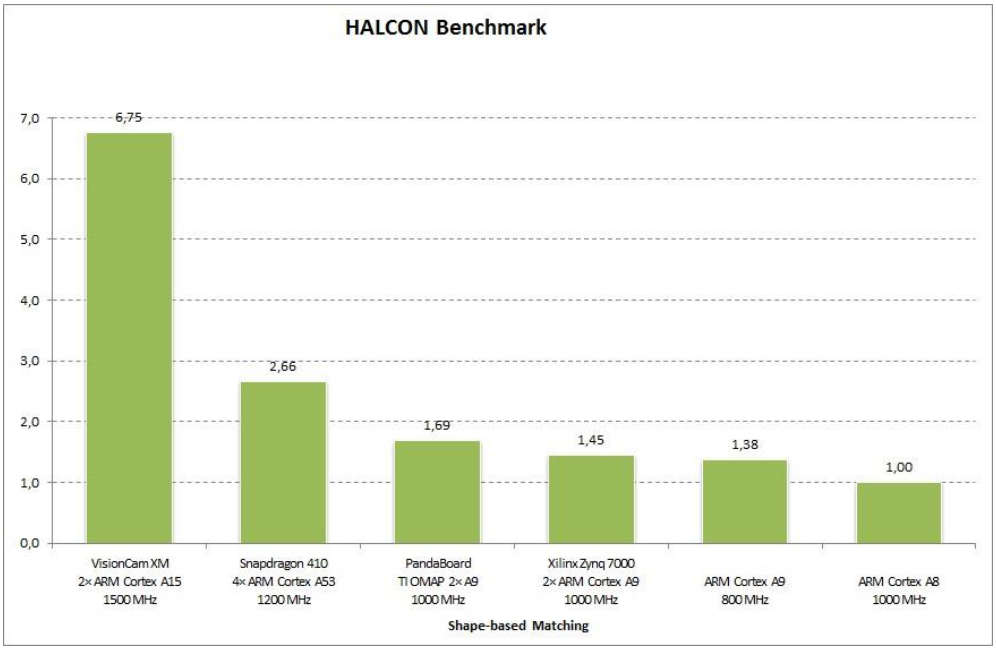

Fig. 1: Halcon shape-based matching on ARM CPUs

The benchmark of a Halcon operator shows the superiority of the Cortex-A15 in contrast to other ARM-architectures. And that is exactly what the new smart camera applications are about: Can I realize my demanding application in the form factor of a smart camera? Does it have to be a box computer in the control cabinet? In this context, the term “Personal Vision Sensor” was created, referring to the “Personal Computer”.

Fig. 2: Personal Vision Sensors

This “Personal Vision Sensor” is a computer platform enabling a demanding, individual application to run in the form factor of a sensor. In addition to the computing power, the goal is achieved through further interfaces as well as the OS Linux. The Linux market provides additional programs for a range of tasks accompanying applications.

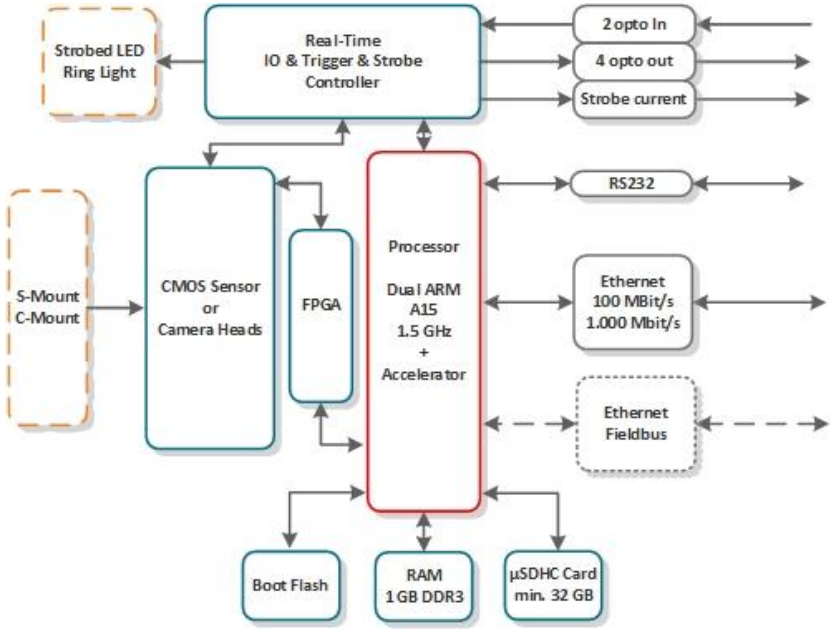

Fig. 3: Block diagram VisionCam XM

IMAGO Technologies GmbH

Strassheimer Str. 45; 61169 Friedberg - Germany; Tel. +49(0)6031-6842611; Fax +49(0)6031-6842612

info.itf@imago-technologies.com; www.imago-technologies.com

Further details can be taken from the block diagram – however, one thing worth noting is the option of equipping the camera with a fieldbus. This real-time Ethernet is provided through a further RISC processor integrated onto the CPU chip. The firmware running there has the task to process the respective stack of one of the fieldbuses (EtherCAT, ProfiNet, SercosIII, Powerlink, etc.). Apart from the drivers and the RJ45 sockets, no further upgrade is necessary.

On the camera's side, a range of CMOS sensors is available, beginning with WVGA ranging up to 5 MPixel resolution. With S- or C-Mount video lenses as well as integrated LED strobe lighting, a range of different models has been developed, which covers the everyday requirements of image processing engineers. However, it is important to emphasize that the application development of a Personal Vision Sensor is only economically effective for series. The MOQ (minimum order quantity) of 25/year is low and allows the pursuit of newest product ideas.

Development Environment: Traditional

"Traditional" – what is meant here is the development toolchain MS Visual Studio, which is justly successful with developers. Eclipse is also common, but the IMAGO developers have found a method to develop Code via MS Visual Studio for the smart camera running on Linux. A plug-in transfers the source code into the camera, executes the compiler there and sends debugging information back to the Visual Studio. This has become elegant and easy in comparison to former procedures – reservations about a Linux-based CPU cannot remain. In addition, a lot of developers are by now more or less familiar with the ARM processors. A base application is quickly running, and the developing engineer can focus on the actual task. Moreover, a 2 hour long expert seminary is available via video training.

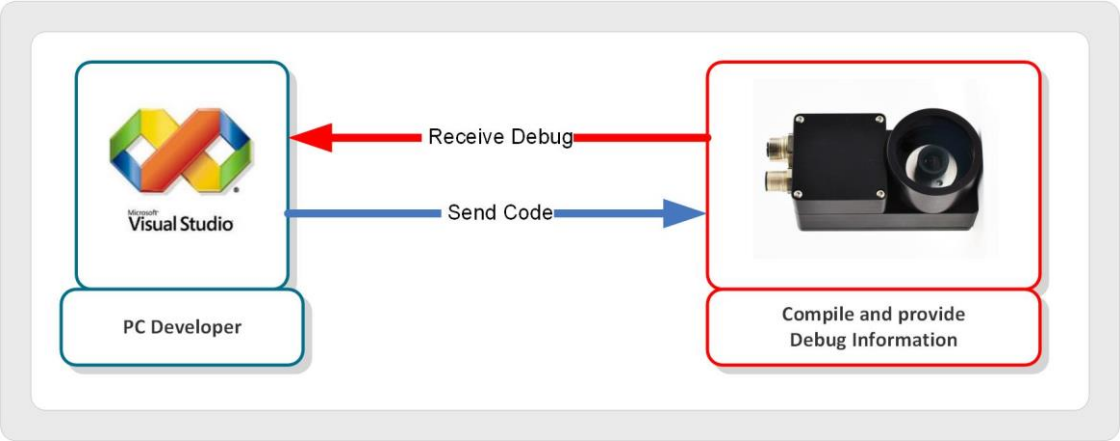

Fig. 4: Development environment

Interactively Extending Image Processing: Halcon Embedded

Halcon as a popular image processing library principally runs on Linux-based computers, too – the condition for this is the compiling for the respective target architecture. If you develop a supporting program for the Personal Vision Sensor which serves the interfaces including the camera, you can change and extend the core of the image processing routines via HDevelop. HDevelop generates a script which is directly interpreted on the intelligent camera. Via remote access, this change can take place globally on the camera computer.

Fig. 6: The VisionBox LE MANS with 10 GBit/s Ethernet, PoE, ToE, LED Controller

New Performance Class for VisionBoxes

With the ARM Cortex-A72, a new performance class is also available for the VisionBoxes. These vision computers are equipped with all interfaces necessary for image processing and partly also for automation. Furthermore, the Real-Time Communication Controller allows the generation of real-time capable signals from input signals. For example, a sensor input is manipulated (delayed, interpreted, counted) and from this, a camera trigger signal or a signal for the control of the integrated LED Controller is generated. The camera trigger signal itself is available at the exit or alternatively as an Action Command for Trigger-over-Ethernet cameras. All these functions are available on the x86 VisionBoxes – not only for the OS Linux, but also for Windows Embedded. Why, then, should you consider a CPU alternative here?

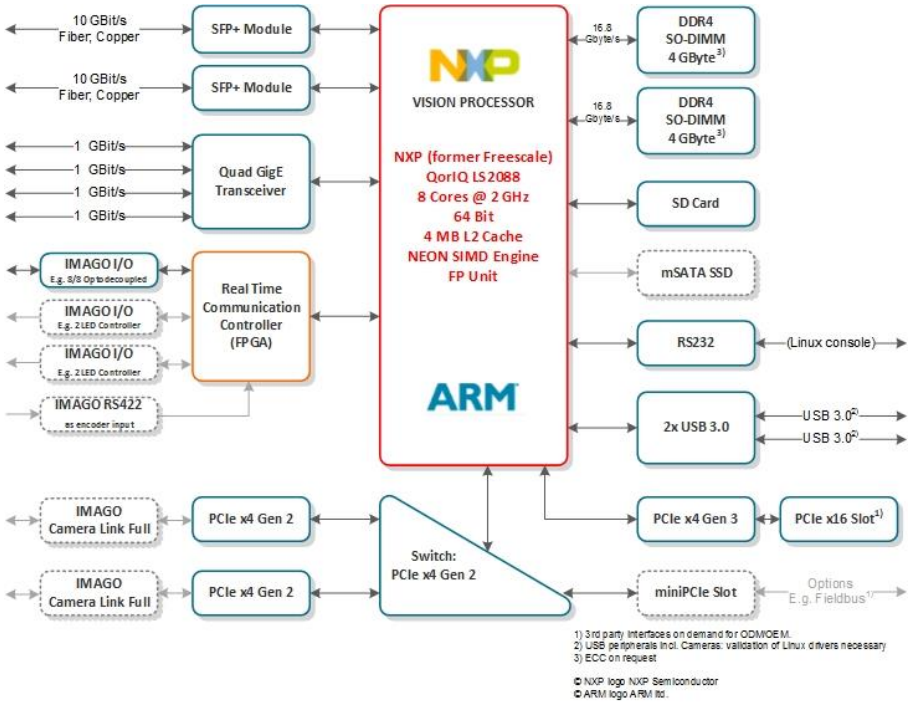

Fig. 7: Block diagram VisionBox LE MANS

There are three essential reasons: if you assume the power loss of a CPU and define f. ex. a limit of 45 watts for a fanless computer, the ARM architecture provides more computing power compared with the x86. This computing power is even permanently available, which means that the CPU is permanently clocked with 2 GHz, whilst an x86 processor speeds up (and down) sporadically and is often only performant in this power range.

The second reason lies in the long-term availability of processors: manufacturers of industrial CPUs are able to build them “as long as possible”, meaning for more than 10 years. Commercial x86 CPUs are only deliverable for a few years, the more expensive Embedded series 5-6 years. When a system is solidly designed, the machine engineering industry often requires up to 10 years of availability plus the following availability of spare parts. All this is also possible with generation changes of x86 CPUs, but every expert is aware of the effort behind that. In other words: if a vision system has to be produced for 10 years, for instance, an ARM-based architecture can become cheaper per unit regarding the total costs than a comparable x86 computer. What should not be ignored are the efforts behind a new compatible x86 generation like e.g. development resources, changes in logistics, administration.

The third reason stems from the technology: today’s 8-core Cortex-A72 CPU is produced with a 28nm process, a current Core x86 with 14nm. The number of cores will rise, that is certain. With the 28nm process, there is simply much more future potential to increase the number of cores or to reduce the power consumption than with the 14nm process.

Fig. 8: For line scan camera applications: LE MANS incl. Camera Link Grabber

Back to the VisionBox: you can already guess that the new flagship of the company NXP is brought into action here: the LS2088 with a Cortex-A72, 8 cores. The DDR4 memory interface allows the fastest data transfer of image data. Additionally, two 10 GBit/s Ethernet interfaces for camera and/or other communication are directly connected with the processor. 4 additional Gigabit/s Ethernet connectors allow the connection of further network devices. And instead of thinking about accelerator cards like GPU, DSP or FPGA, the computing power can be scaled via the 10 GBit/s Ethernet. Faster system solutions, homogenous developing teams, safety for the future are just a few keywords for decision makers.

Summary, Conclusion and Outlook

ARM-based CPUs are increasingly moving into the architectures of vision computers and smart cameras and offer a plus of computing power and long-time availability of components in the respective form factor. In many industry applications, Linux is no longer a foreign concept, young engineers are trained in it and senior engineers go strong with Linux experiences.

Sub-functions of a machine – where image processing belongs to – are becoming more complex, have to work safely and autarkic. Everything done in a PC increases the risk of not being able to act for too long in the case of an error. Here, embedded vision systems offer a lot of advantages. And the machine without control cabinet is already being promoted – but where are the computers in this solution?

Whether x86 or ARM, whether Linux or Windows Embedded – what is target-aimed for the customers remains their own decision. Still, the IMAGO team with their experience and an individually elaborate decision matrix can contribute to choosing the best architecture.

In the future, the evolution will continue – algorithms that are more intelligent and software need hardware that is more powerful in order to equip new machine generations for Industry 4.0.

Authors:

Dipl.-Ing. Carsten Strampe; Dipl.-Phys. Oliver Barz