# Labview with embedded Linux on ARM

*More and more embedded systems develop into distributed, networked systems. To solve this new complexity, Labview combined with embedded Linux and a dual core ARM system-on-module is one useful method.*

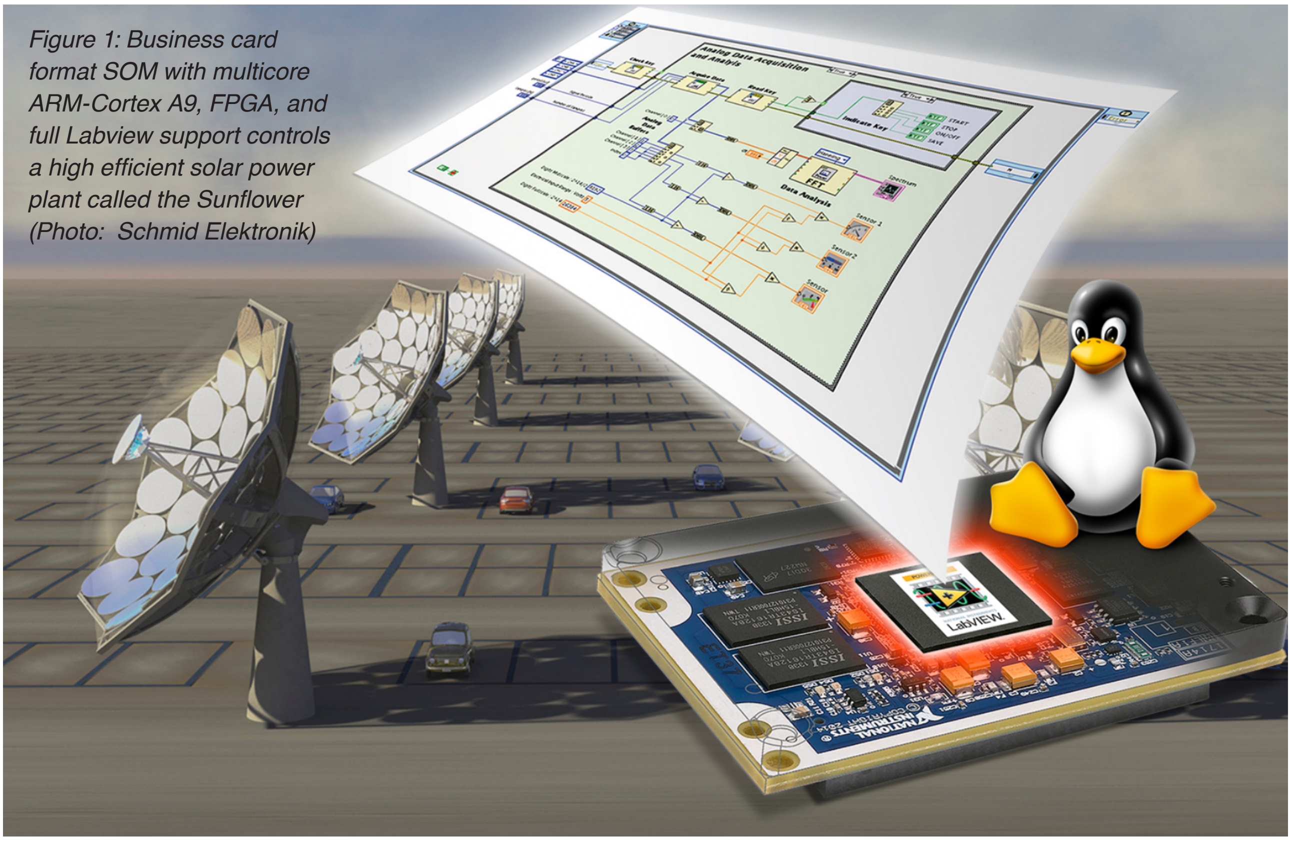



*Figure 1: Business card format SOM with multicore ARM-Cortex A9, FPGA, and full Labview support controls a high efficient solar power plant called the Sunflower (Photo: Schmid Elektronik)*

Conventional product development with embedded systems usually requires dedicated hardware with specific I/Os. A micro-controller, DSP, FPGA, or a combination of these serves as the brain. The programming languages C and VHDL have been the de facto standard so far, but the industry is changing. The conventional approach is being questioned as more and more embedded systems develop into distributed, networked systems, thus forming a completely new league. New ways of thinking and new techniques are required to manage the related increase in complexity. The high degree of abstraction of the graphical programming language Labview is one efficient approach to this end, combined with the flexibility and stability of Embedded-Linux and the raw power of a dual core ARM system-on-module (SOM) with FPGA.

The RIO SOM by National Instruments is as small as a business card. Its operating temperature range of -40 °C to +85 °C makes it suitable for industrial applications. It is energy efficient and provides the functionalities typically required by smart embedded systems. The high-speed CPU consists of a ZYNQ system-on-chip (SOC) by Xilinx, with a 667 MHz dual core ARM Cortex-A9, Artix-7 FPGA, external 512 MiB DRAM, and 512 MiB flash memory as well as communication functions like Gigabit Ethernet, CAN, USB 2.0 host/device, SD card, and serial interfaces. These are linked to the baseboard via a rugged 320-pin connector, including 160 GPIOs of the FPGA. The baseboard conditions the bare TTL signals and contains further customer specific hardware devices as well as supplies. These features do not differentiate the SOM from conventional embedded systems. However, it can be programmed in Labview, which is a major advantage resulting in a higher development speed for programmers who often work under time pressure.

## Development time – the new currency

How can a small team achieve big results? First of all, different programming models can be used as flexibly as a Swiss Army knife, tailored to suit the task or the developer's preference. A control engineering concept, for instance, can be implemented and executed in the Matlab notation. It can be included in the graphical Labview block diagram like differential equations created with model based design. Statecharts on the other hand simplify ▷

the complex processing logic, and a C code interface provides the inclusion of existing C algorithms or direct access to the Linux operating system. Second, Labview programmers directly benefit from support without any further development effort: from libraries for mathematics and signal processing, toolkits (e.g. filters, controllers, sound and vibration, vision, motion) and real-time synchronization of decentralized systems to advanced operating strategies (smartphones, tablets) and data communication (Cloud, real-time Ethernet). Third, the timing, operating system, multitasking, multicore, and underlying hardware are conveniently abstracted. Thus, the Labview application can be downloaded and executed in real-time to own embedded hardware, without any in-depth knowledge of the underlying details.

## Flexibility and security of Linux

The widely used, popular operating system Linux and the related ecosystem open up new possibilities to the Labview community:

- The Ångström Distribution, optimized for embedded applications, is installed with a repository on the NI servers. The Labview diagram is mapped 1:1 onto the Linux operating system according to the Posix standard. There, the Sandbox of the Linux ecosystem can be deployed, for example, for downloading an SQL database, Apache webserver, or QT GUI with the package manager Opkg.
- The Busybox is a tool for typical embedded tasks, from file system access, system time and, small DHCP client to sleep mode and system reboot.
- Labview has access to the Linux command line, facilitating the direct execution of system commands as well as live management of file system and user permissions.
- Powerful script languages are available, such as Python.
- With TCP/IP, Labview uses Localhost to tap the services of other Linux processes, the so-called daemons.
- The libraries of the Linux operating system can be accessed via the native C API of Labview. In addition, experienced Linux users can configure and recompile the Kernel individually.

U boot – widely used, not only in embedded Linux applications – serves as a bootloader. The kernel is started and initialized based on System V, which facilitates the start of user specific processes. Via one of the shells, the system can be managed by means of an external terminal like Putty (Figure 2). A web-based graphical configuration tool would facilitate the process even further. Instead of using FTP, data is exchanged via WebDAV, an industry standard for secure data transmission on HTTP basis, which is also used by Dropbox.

Particularly in the network of smart embedded systems, timing is the greatest challenge. Timing is an integral element of the language in Labview, as it has always gone hand in hand with industrial process measurement ▷

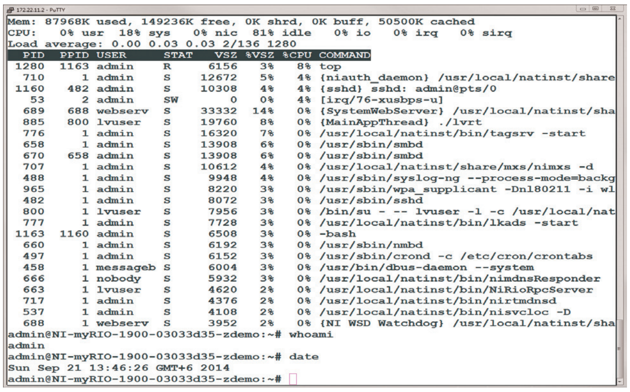



*Figure 2: The SSH (secure shell) facilitates the management of Linux on ARM and the use of various command line services (Photo: Schmid Elektronik)*

and control hardware. Graphical multitasking programming requires operating system functionalites like scheduling. Six schemes are available to this end. Cron facilitates the execution of repetitive tasks down to [min] resolution, e.g. deletion of logfiles or frequent e-mail checks. The CFS (completely fair scheduler) mainly serves to implement work tasks that are not time-critical but still efficient. If [ms] response times are required, the kernel can be configured as preemptive. For time-critical tasks with a required jitter of 10 $\mu$s to 100 $\mu$s, the Linux kernel is patched with "PREEMPT_RT". The FPGA is used for hard real-time in the one-digit [$\mu$s] or even [ns] range. Due to Labview's multicore support, graphical tasks can be directly assigned to a processor core.

## Programming the FPGA with a mouse

The Artix-7 FPGA hardware is software-reconfigurable and suitable for parallel processing (Figure 3). Time-critical tasks and I/Os are delegated to this hardware to reduce ARM processor load. Practicable functionality as well as the related timing are two crucial parameters. So far, the programming of FPGAs has required specific expertise. With Labview, even engineers without such expertise get access to the powerful technology of reconfigurable logic. Using intuitive functional blocks, a range of analog, digital, and serial process signals can be integrated, linked, and preprocessed in parallel before being transferred to the ARM processor. The functionality of the application to be mapped is directly limited by the number of available gates. Consequently, available information on how many gates are required for each specific operation (addition, filter, FFT) is a valuable basis of decision-making regarding the functional scope that can be implemented as a maximum. In the context of timing, even application programmers with graphical programming preference think in ticks, the smallest FPGA time increment in the nanosecond range. It defines how much time is required by logical and mathematical operations and I/O accesses, providing target values for system timing.

In the software design process, the embedded application has to be split between ARM and FPGA. The ARM processor is in charge of high-level main functions. Low-level details like device drivers, time-critical code, digital filters, combinational and sequential logic, scaling, fixed point, and integer arithmetic, are preferably outsourced to the FPGA. The NI Labview FPGA diagram is synthesized in VHDL code, compiled to firmware/bitfile by the FPGA tools and downloaded to the FPGA.

Almost any commercially available I/O component can be connected to the SOM through a baseboard (Figure 6) and controlled with Labview, e.g. via digital I/Os, synchronous (SPI, I²C) and asynchronous (UART) serial interfaces or parallel high-speed bus systems. Typical examples are analog I/O, PWM, counters, encoders and digital I/O, wireless/WLAN, RFID, GSM/GPRS, GPS, Zigbee, and color TFTs with CAP/multi-touch. The hardware can be adapted to any task as regards shape and function. To achieve such flexibility, the first step is to develop hardware in the form of a baseboard, a task usually required in a dual-plate approach. The most critical circuits around the CPU and memory are already implemented on the plugin SOM.

## Individual hardware drivers with Eclipse

In the Labview application, external I/O devices are activated through intuitive virtual instruments (VIs). Low-level drivers are required to this end. If the devices are connected to the 160 FPGA pins, drivers can be implemented with a similar technique as for Compact RIO. If they are connected directly to the ARM processor and the BSP (board support package) has no driver, Linux offers the following three options:

- If the driver is available as executable, it can be executed in Labview directly through the command line VI. Here, Labview is logged in as "lvuser".
- Otherwise, the executable is generated with the Eclipse IDE (Figure 4) and executed like in the first option. To this end, the driver has to be available as C/C++ source code. Due to access through the operating system, response times in the two-digit millisecond range can be achieved.
- A library (shared object) is generated with Eclipse and addressed in Labview via the C API (Figure 5), similar to a DLL under Windows. Compared to the both previous options, direct access to the C library now facilitates timings in the two-digit µs-range.

## System-on-module in a solar power plant

With the Sunflower, a Swiss company brings solar energy to the remotest places on earth. No matter where you live,

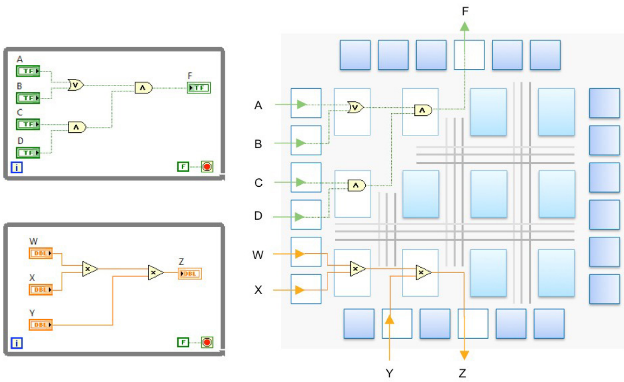



*Figure 3: The true parallelism of FPGA programming comes closest to the Labview data flow paradigm (Photo: National Instruments)*

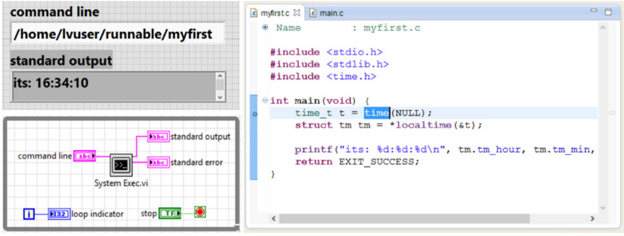



*Figure 4: Labview executes a Linux executable compiled in Eclipse (right) via the command line VI (virtual instrument, left) (Photo: Schmid Elektronik)*

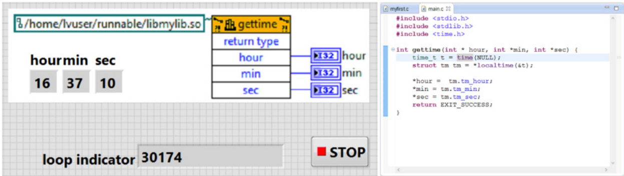



*Figure 5: Through its C API (application programming interface, left), Labview accesses a Linux library generated in Eclipse (*.so = shared object, right) (Photo: Schmid Elektronik)*

you get cooling in summer, heating in winter, clean drinking water, and even fuel. The key of the solar plant is to concentrate solar radiation in the focal point of a dish with a factor of 2000 and extract energy with an efficiency of 80 %. This is achieved with ultra-high efficiency solar cells and a cooling system of IBM. The Sunflower weighs 18 t, is 10 m high and produces up to 300 kWh of energy during a sunny day. It consists of three main elements: the optics, the receiver, and the tracker.

The optics is represented by a huge 40-m$^2$ dish. Its main function is to concentrate solar rays in its focal point. To achieve this, the dish contains 36 elliptic mirrors. On each of these mirrors, a very thin, silver coated foil similar to chocolate wrappers is applied through a very small vacuum. That's why they are also called pneumatic mirrors. A solution like this is much lower in cost compared to warped solid state mirrors. The dish is covered with an inflated, robust plastic membrane. The reason for this is twofold: First, the dish has to be protected from the environment such as rain, sand, and other dirt in order to keep the mirrors clean. Second, to protect wildlife such as birds from the sunflower. The inflation is achieved with a pneumatic system consisting of fan blower, compressor, and an accumulator that are located in the pole of the sunflower.

An sbRIO-9651 SOM as the system's brain moves the dish continuously towards the sun and connects to dozens

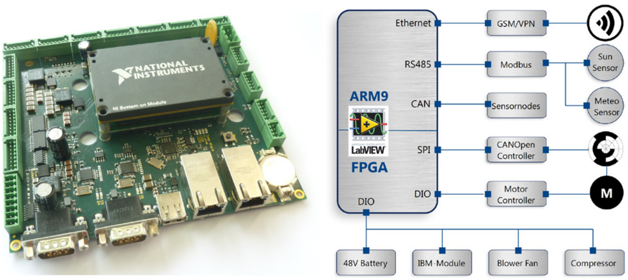



*Figure 6: The SOM (black) is plugged onto a baseboard (left) of an individual form factor, which contains customer specific hardware (right) (Photo: Schmid Elektronik)*

of sensors through a baseboard (Figure 6). Stepper motors move the two axes of the tracking system using absolute encoders in the feedback loop. In order to keep production costs low, the motor driver has been implemented with discrete electronics such as transistors to drive the PWM. The tricky thing to handle was to connect to the encoder using CANopen to the FPGA. The solution is a micro-controller, featuring CANopen in its silicon, connecting to the system-on-module with SPI and acting like a translator. On top of that, the CAN network of the ZYNQs ARM, linked with Labview Realtime, was also connected to the CANopen network. This link is mainly for servicing the CANopen sensors during configuration mode. As for the encoder, this meant configuring the sampling rate of the encoder signals on the CAN network.

Next, a commercially available sun sensor is connected via Modbus RTU. A local weather station delivers the local temperature, humidity, pressure, wind force, and direction over the TCP/IP-Modbus. The VPN access is realized with a netmodule, connected via Gigabit Ethernet. A distributed sensor network including pressure-, temperature-, and humidity sensors is connected by CAN and helps the controlling unit to maintain a stable climate within the covering membrane. All other devices such as the fan blower, compressor, and electronic valves are controlled by a robust 24-V PLC type of digital signals. ◀

**Author**


Marco Schmid
Schmid Elektronik
www.schmid-elektronik.ch
info@schmid-elektronik.ch

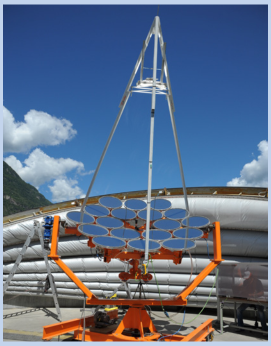

## 80 % energy efficiency from solar concentration

Airlight Energy, a Swiss-based supplier of solar power technology, has partnered with IBM Research to bring affordable solar technology to the market. Their High Concentration Photovoltaic Thermal (HCPVT) system can concentrate the sun's radiation 2000 times and convert 80 % of it into useful energy to generates 12 kW of electrical power and 20 kW of heat on a sunny day — enough to power several average homes. The stand-alone dish provides electricity, heat, hot water, and conditioned air (through a chiller) all at the same time. The system, which resembles a sunflower, uses a parabolic dish made of patented fiber-based concrete. The concrete can be molded into nearly any shape in less than four hours and has mechanical characteristics similar to those of aluminum at one-fifth the cost. The photovoltaic chips, similar to those used in orbiting satellites, are mounted on micro-structured layers that pipe treated water within fractions of millimeters of the chip to absorb the heat and draw it away 10 times more effectively than with passive air cooling. The +85 °C to +90 °C hot water maintains the chips at safe operating temperatures of +105 °C, which otherwise would reach over +1500 °C. The entire system sits on a sun tracking system, which positions the dish at the best angle throughout the day to capture the sun's rays.

"The direct cooling technology with very small pumping power used to cool the photovoltaic chips with water is inspired by the hierarchical branched blood supply system of the human body," said Dr. Bruno Michel, Manager of advanced thermal packaging at IBM Research. With such a high concentration and based on its radical design, researchers believe that with high-volume production they can achieve a cost two to three times lower than comparable systems. Based on its current design, scientists estimate that the operating lifetime for the HCPVT structure will be up to 60 years with proper maintenance. The protective foil and the plastic elliptic mirrors will need to be replaced every 10 to 15 years depending on the environment, and the photovoltaic cells need replacing every 25 years.

Greenpeace named the Sunflower the number one solar wonder of the world because it can "not only provide electricity – it can also desalinate water for sanitation and drinking. A group of several solar generators could provide enough fresh water for an entire town. The sunflower operates by tracking the sun, so that it always points in the best direction for collecting the rays – just like a real sunflower!" The system can be customized with further equipment to provide drinkable water and air conditioning from its hot water output. For example, salt water can pass through a porous membrane distillation system, where it is vaporized and desalinated. Such a system could provide 30 l to 40 l of drinkable water per square meter of receiver area per day, while still generating electricity with a more than 25 % yield or 2 kWh per day – a little less than half the amount of water the average person needs per day according to the United Nations, whereas a large multi-dish installation could provide enough water for a town.

Scientists at Airlight and IBM envision the HCPVT system providing sustainable energy to locations that need both outputs of the sunflower: energy and heat. Possible locations around the world include southern Europe, Africa, the Arabian peninsula, the southwestern part of North America, South America, Japan, and Australia. In addition to residences, additional applications could include remote hospitals, medical facilities, hotels and resorts, and shopping centers.



*Annegret Emerich*

**Youtube**

Airlight Energy and IBM Bring Solar Electricity and Heat to Remote Locations