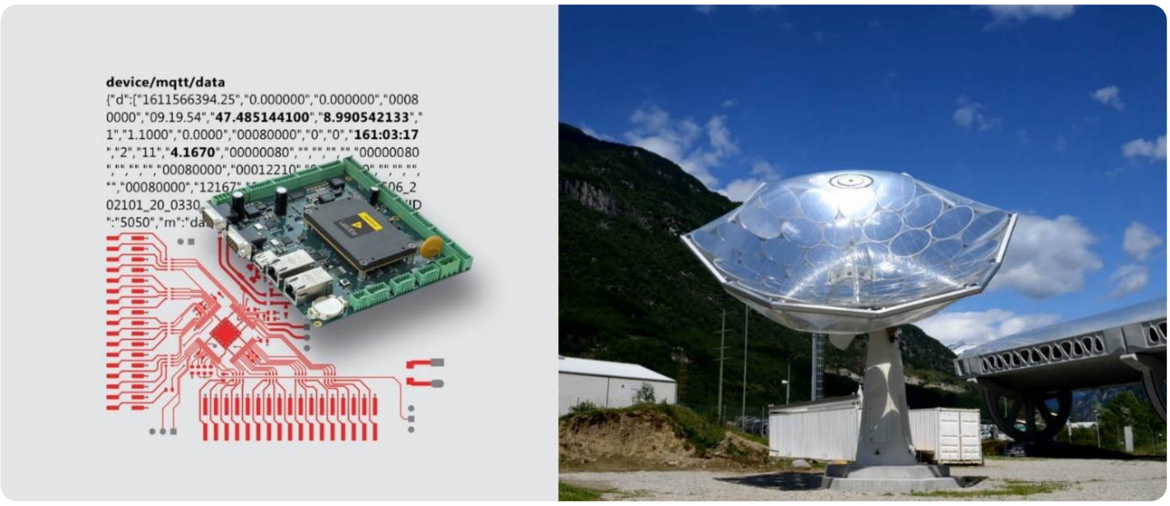
Whitepaper

LabVIEW Embedded with Linux on ARM and FPGA



For those interested in learning more about the use of LabVIEW in embedded applications and the underlying technology, this whitepaper provides a deep dive. It offers additional insights and knowledge that some might not expect from LabVIEW. In the image gallery, we are accompanied by a real project based on the NI System-on-Module (SOM). It is the electronic heart and brain of the "sunflower". This is a compact solar power plant with an 80% efficiency of energy utilization.

Contents

1 What is LabVIEW Embedded?

LabVIEW is often mentioned in the same breath as laboratory work or testing. LabVIEW Embedded, on the other hand, scales further, namely into the area of embedded systems. The advantage is obvious: use LabVIEW on your own hardware for graphically programmable series products! The catch: many convenient features that we appreciate from NI's off-the-shelf hardware have to be developed ourselves. This includes the integration of custom specific analog converters and their driver development. In addition, special requirements are placed on application development, such as a scalable architecture and consistent error handling.

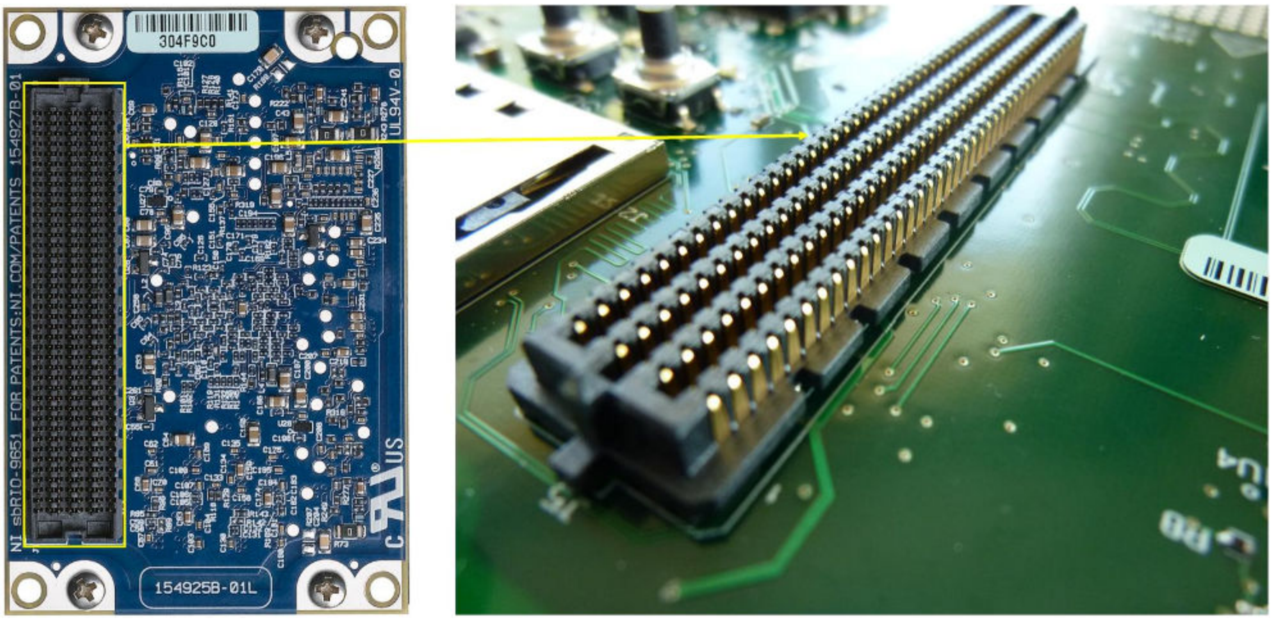


Fig. 1 | The 320-pin SEARAY connector between the SOM and the baseboard is a BGA (Ball Grid Array).

This whitepaper provides an overview of how the NI standard can be combined with customer-specific requirements using the case study of a real project. This standard is synonymous here with the NI System-on-Module, sbRIO9651 or SOM for short. This module with the dual-core ARM Cortex A9 and FPGA is again available for the long term. The two-board approach is described with this universal, flexible SOM, plugged into a baseboard with I/O of your choice (Fig. 1, right).

2 The Flexibility of Linux

LabVIEW Embedded runs on Open Source Linux. Thanks to this widespread operating system and the associated ecosystem, new possibilities are opening up for the developer community. Installed on the SOM is the SUMO distribution with the 4.14 kernel optimized for the embedded sector with a repository on NI's servers. The LabVIEW code is mapped to the operating system in accordance with the Linux standard. There, the Linux ecosystem can be leveraged in LabVIEW with the usual functions. For example, the graphical environment has direct access to the "command-line" using a VI, allowing system commands to be executed directly. It is furthermore possible to access operating system libraries via the C-API (Application Programming Interface). In addition, experienced programmers even have the option of configuring the kernel individually. The system can be easily managed via one of the shells using an external terminal such as PuTTY (Fig. 6). It is even easier with the web-based, graphical configuration tool WebDAV.

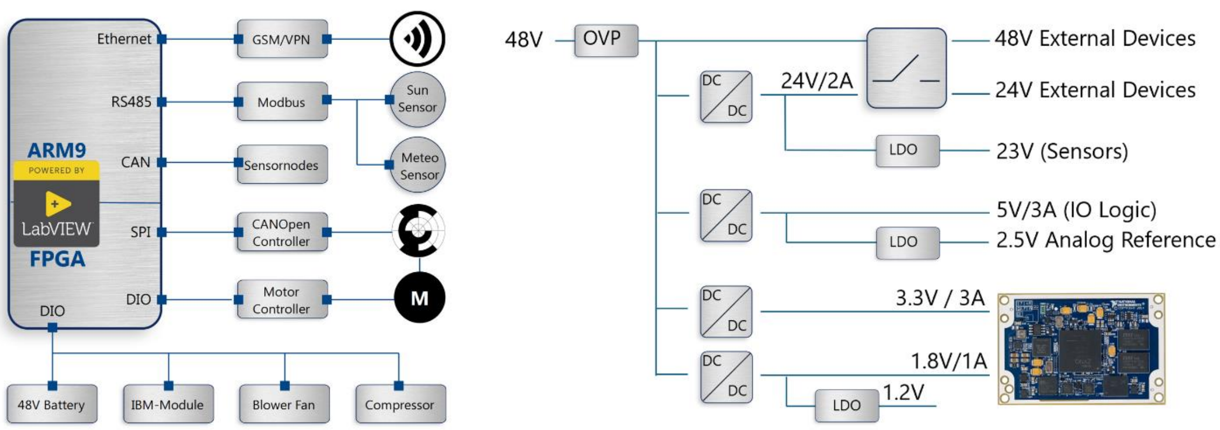


Fig. 2 | The first step is to design the baseboard.
On the left are the functions, on the right the power concept.

3 My own LabVIEW Hardware, please!

Almost every electronic I/O module available on the market can be connected to the SOM and controlled with LabVIEW (Fig.2, left). The possibilities are manifold: analog and digital I/O (memory mapped), synchronous (SPI, I^{2}C) and asynchronous (UART) serial interfaces or parallel high-speed bus systems. Typical examples of such components are analog to digital converters (ADC).

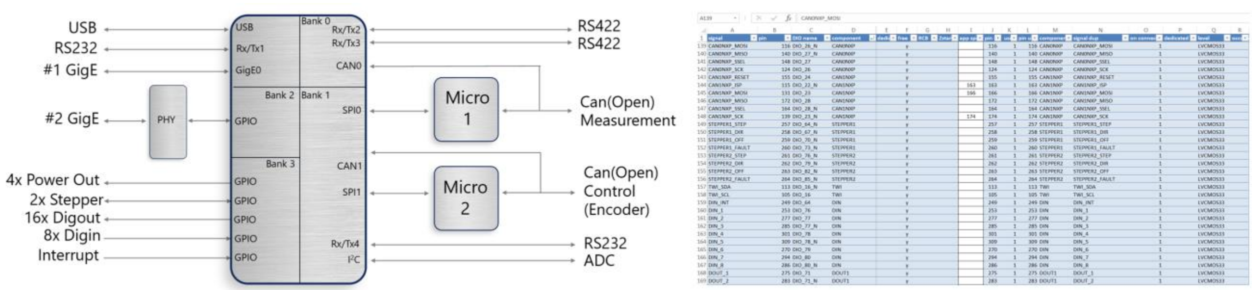


Fig. 3 | The next step is to assign the pins of the SEARAY connector (Fig. 1) to the I/O.
Due to the multiple assignment, we recommend defining a standard (right).

The advantage: the form and function of the hardware can be adapted to any task. This flexibility comes at a price, however, since this is where the two-board approach that is popular in the embedded sector comes into play. Hardware in the form of a baseboard with a proper power supply must always be developed, as the SOM alone does not function. The risk is manageable, as the time-critical high-speed circuits around the microcontroller and memory are already routed on the plugged-in SOM.

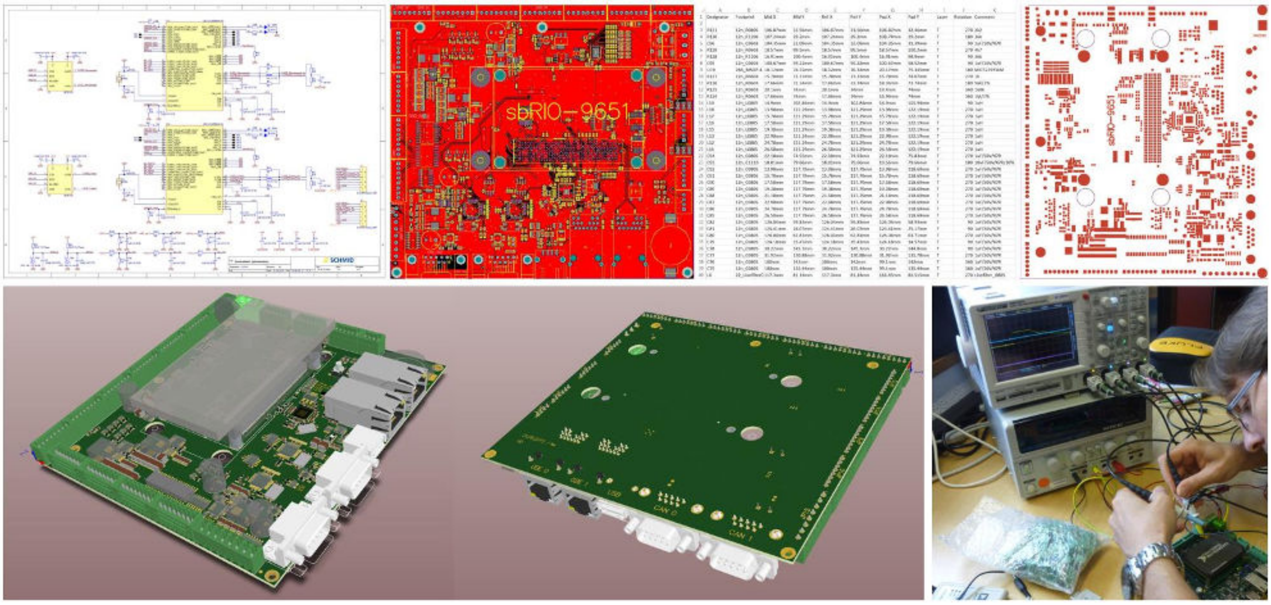


Fig. 4 | The development steps for embedded systems: create schematic design, create layout, generate production data (pick & place, template data), putting into operation after production

Due to the densely packed BGA connector, advanced developer knowledge is nevertheless required for the baseboard design. A hardware developer usually goes through the following steps from the initial idea to the final baseboard:

1. Analyze the requirements and evaluate the I/O components, e.g. A/D converters
2. Design of the baseboard (Fig. 2) and assignment of the SOM pins (Fig. 3)
3. The function, operating and limit values as well as timings can be found in the data sheets.
4. Create the schematics and incorporate knowledge from the data sheet and the application notes
5. Generate the bills of materials (BOM) and parts list for purchasing and production
6. Define the physical footprints according to production specifications and IPC standards
7. Create the board geometry and locate connectors and mounting holes
8. Map the mass and supply concept to the multilayer structure
9. Place electronic components and unbundle the connections
10. Route the layout, fill ground/power areas, check signal integrity
11. Create production data for PCB manufacturer and SMT line

4 Production in SMT Machinery in Batch Size 1

The 320-pin Searay connector between the SOM and the baseboard is a BGA (Ball Grid Array, Fig. 1). The soldering points are therefore not accessible from the outside. That's why it is advantageous for production to be carried out by an EMS (Electronic Manufacturing Service provider) with professionally equipped SMT machinery (Fig. 5). At the start of development, the quantities are often small, so it is beneficial to choose a manufacturer that can handle prototyping, NPI (New Product Introduction) and the batch size 1. Technically, on schedule and also economically.

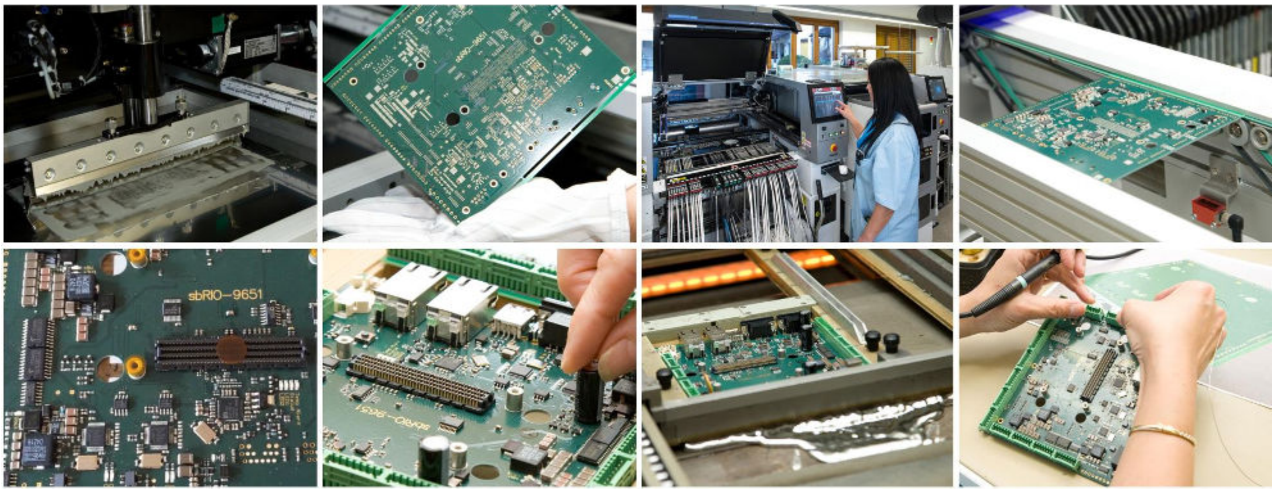


Fig. 5 | How an embedded system is manufactured: Solder paste printing, automatic assembly, reflow soldering, manual assembly, wave soldering and repairs

5 Tailored LabVIEW Hardware from the One-Stop-Shop

If the EDA infrastructure (Electronic Design Automation), the required hardware, software and operating system expertise or the SMT machinery are missing, NI partner Schmid Elektronik offers a one-stop-shop for tailored LabVIEW hardware. This includes a development and production service for individual baseboards with customer-specific industrial electronics. A production team optimized for low-volume/high-mix and a high-tech machinery optimized for prototyping and series production are also available, supplemented by a test and EMC laboratory. On the software side, this includes modification of the Linux kernel as well as the development of individual device drivers and their integration into the LabVIEW environment. All in all, a convenient service for anyone who wants to run LabVIEW on their own hardware. A customer once put it this way:

"We value the mutual trust and professional cooperation with our partner Schmid Elektronik, which offers absolutely reliable quality in the development of hardware and software as well as in production and testing. The agile and open-minded way of working enables flexible, lean processes, especially for our prototypes and small series. Schmid Elektronik is one of our pillars in adapting the NI platform to our special needs in the field of high-voltage DC transmission."
(Julian Lange, Siemens Energy)

6 Developing Low-Level Device Drivers

As soon as the hardware is available, the low-level drivers are developed. Drivers access external I/O modules and they are abstracted with a high-level VI. If the components are connected to the FPGA pins, the drivers are written graphically on the FPGA using LabVIEW. If they are connected directly to the microcontroller, three options are offered under Linux:

1. If the driver is available as a binary, it can be executed directly from LabVIEW by the command line VI. In this case, LabVIEW is logged on to Linux as "lvuser". The achievable response times are in the double-digit millisecond range due to access via the operating system (Fig. 6, top left).

2. If the driver is available as a C source, the executable is built using the Eclipse IDE.

3. Finally, a library (shared object) can be created with Eclipse and accessed from LabVIEW via the C-API-VI, similar to a DLL under Windows. Thanks to direct access to the C library, response times in the two-digit microsecond range are possible (Fig. 6, bottom left).

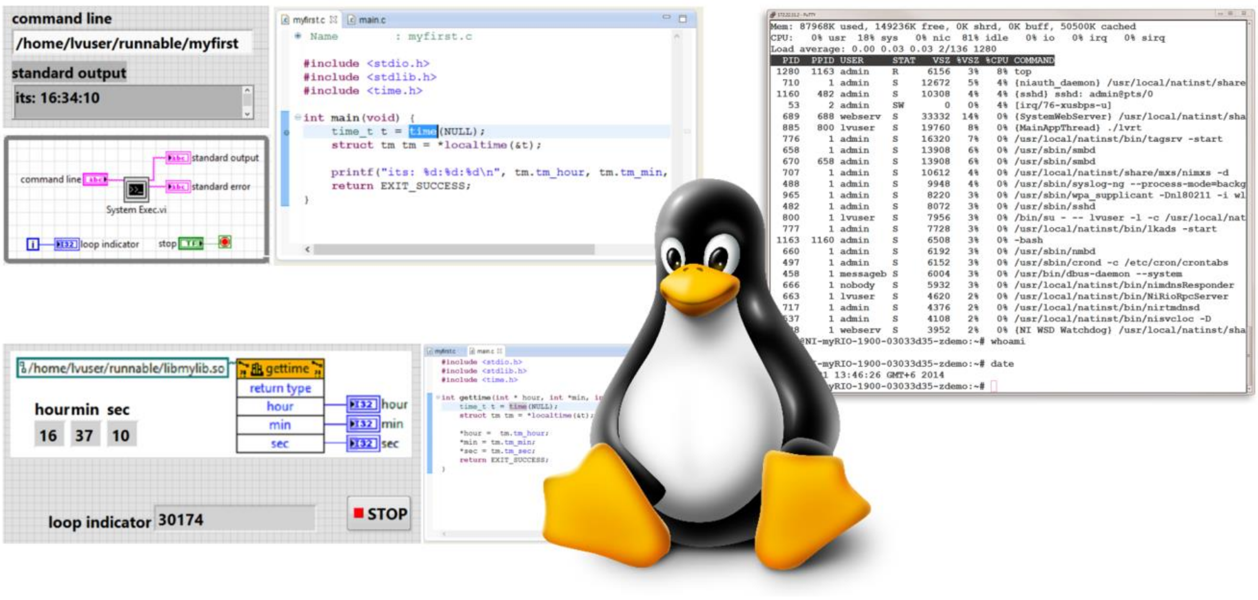


Fig. 6 | Interfaces between LabVIEW and Linux: top left, option 1) and bottom left, option 3. On the right, the PuTTY external terminal commonly used with Linux.

7 Program an FPGA with the Mouse

The Artix 7 FPGA in the SOM is software-configurable, parallel hardware. Time-critical tasks and I/Os are delegated to it in order to relieve the microcontroller. Decisive key data are the functions that can be realized and the timing. Until now, specialists were needed to program FPGAs. With LabVIEW, even engineers without this experience gain access to the powerful technology of reconfigurable logic (Fig. 7). Using intuitive function blocks, a wide range of analog, digital and serial process signals can be integrated, linked and pre-processed in parallel before they are forwarded to the microcontroller.

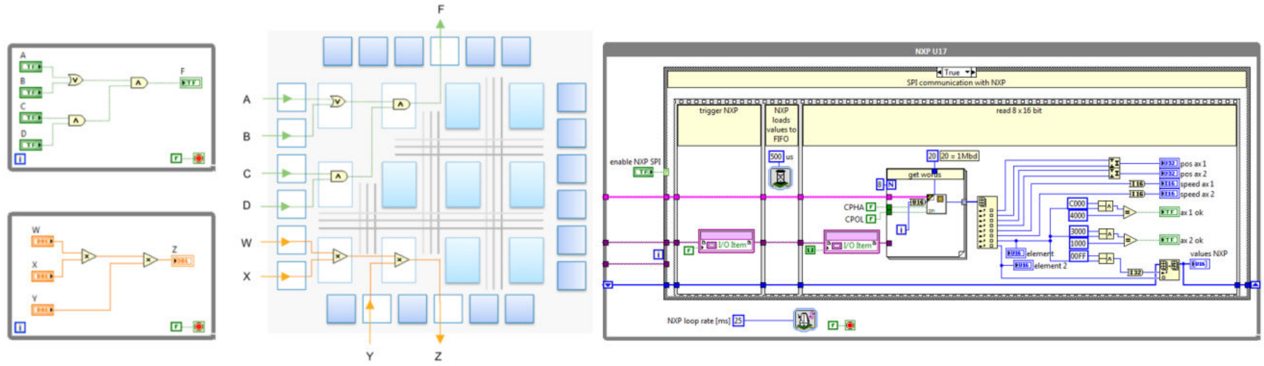


Fig. 7 | Developing low-level drivers with LabVIEW FPGA. Example: communication between the FPGA and an external microcontroller, which can integrate an encoder via CANOpen

The functionality of the application is directly limited by the number of gates available. Known characteristic metrics, which operation (addition, filter, FFT) requires how many gates, provide a valuable basis for deciding on the maximum possible range of functions. When it comes to timing, even the graphically-minded application programmer thinks in "ticks", the smallest FPGA time increment in the order of nanoseconds. This defines how much time is required for logical and mathematical operations and I/O accesses. The result: guide values for the overall system timing.

8 The Balance between Microcontroller and FPGA

When designing the software, the embedded application should be carefully balanced between the microcontroller and the FPGA. The former is responsible for the main high-level functions. Low-level details such as device drivers, time-critical code, digital filters, combinatorial and sequential logic, scaling, fixed-point and integer arithmetic are better outsourced to the FPGA. The LabVIEW FPGA diagram is finally translated into VHDL code with the FPGA tools, compiled into a bitfile (firmware) and loaded into the FPGA via LabVIEW Realtime.

9 Possibilities and Limits

Due to the usually high project requirements of our customers, we have often reached technical limits over the last 15 years. Robust device drivers in multitasking mode required quite some know-how and were often real time wasters. Every now and then, the graphical overhead put a spoke in our (performance) wheel and we had to optimize code in a targeted manner. Consistent error handling turned out to be the key. Apart from the limitations in Table 1, Schmid Elektronik's Zbrain offers a flexible platform with possibilities that can cover the entire area of value creation with just a single programming paradigm: proof-of-concept, minimum viable products (MVPs), prototypes, series products and testing.

Possibilities	Limits (No-Mans-Land)
• Complex application logic, multitasking • Robust 24/7 real-time operation • ADC block sampling standard 450kHz, scalable up to 40MHz • Flexible data communication • Customized I/O modules • Floating point arithmetic • Safety functions (in the FPGA)	• Continuous lowest-power applications in mW. Low-power for LabVIEW can be solved with a combination of the SOM with an RTC or a low-power coprocessor. • LabVIEW on your own microcontroller. This was previously possible with the C-Generator. This entered the end-of-life cycle (EOL) in 2017. • This puts 8-bit small targets out of reach • Price-sensitive quantities

Table 1 | Possibilities and limits of LabVIEW Embedded

10 Appendix: Cyber-Security Low-Hanging Fruit

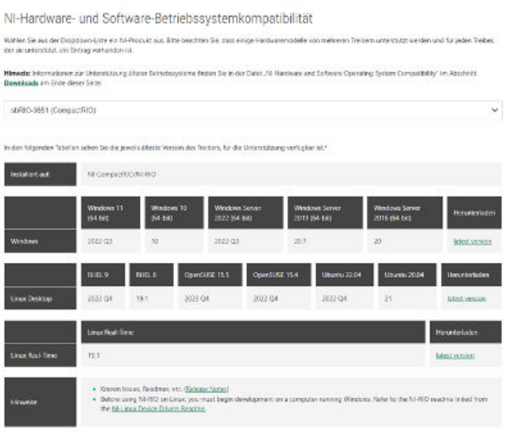
Security of the SOM against external manipulation and attacks

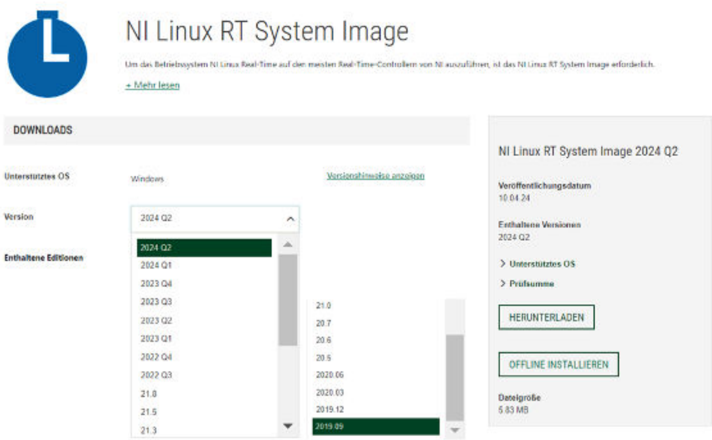
It depends on how strongly you want to secure the SOM against such attacks. As shown below in the links to the security best practices, NI offers various levels of security: from "Open" (i.e. not secure at all, but convenient to work with), to "Recommended" and "Extreme" (completely sealed off). The topic of "network" is also addressed. SELinux (Security Enhanced Linux) can be used to restrict access. This limits the access of processes to the minimum necessary resources. The higher the security level, the more complex the configuration and administration. Documentation on the NI website:

- SELinux - Addressing Access Control Security in LabVIEW RIO Devices - NI
- NI Linux Real-Time Security User Guide
- Overview of Best Practices for Security on RIO Systems - NI
  - Best Practices for Security on RIO Systems: Part 1 Recommended
  - Best Practices for Security on RIO Systems: Part 2 Optional
  - Best Practices for Security on RIO Systems: Part 3 Extreme

Information on the kernel and the NI RT system image

- Single-board RIOs based on the ARM architecture, including the NI System-on-Module, are delivered with the Linux kernel 4.14 and the Sumo distribution.
- The RTOS team announces important changes and corrections via the open source NILRT repository on GitHub.
- NI Linux RT images have a backwards compatibility of 4 years.
  For example, the 2022Q4 image supports LabVIEW 2022, 2021, 2020 and 2019.
- An NI Linux RT image can be downloaded here.
- NI hardware and software operating system compatibility is described here.
- Regular updates on critical security features are available here.
- Companies such as neosoft.ca offer further security components, e.g. a firewall
- See also NI Linux RT Knowledge Base from Hampel Software Engineering





11 Glossary

Chapter 1	Embedded Systems	Combination of hardware and software that performs a sub-function within a larger system in which they are “embedded”.
	SOM, sbRIO9651	The NI **S**ystem-**O**n-**M**odule in the size of a business card.
	Dualcore	A microcontroller with two computing cores.
	ARM-Cortex A9	A dual-core 32-bit microcontroller with an ARM v7 instruction set.
	FPGA	Reconfigurable logic: **F**ield **P**rogrammable **G**ate **A**rray.
	Two-Board Approach	Here: a business card sized module contains only the microcontroller and its chipset and is generally backwards and forwards compatible in form, fit and function. It is plugged into a customised baseboard. This is a popular approach in the embedded industry, as microcontrollers and memory are usually fast-moving, unlike application I/O.
	Baseboard	Contains all application-specific components and accepts the microcontroller module via a high-pin connector, in this case a 320-pin SEARAY.
Chapter 2	Linux	Widely used open source operating system.
	Kernel	The main component of an operating system, in this case Linux.
	Linux Distribution	A selection of harmonised software, tools, apps and services around a Linux kernel.
	Repository, Repo	A central repository that developers use to make changes to the source code of an application or an image.
	Command-Line	Here it is a Linux interface that accepts text lines.
	C-API	**A**pplication **P**rogramming **I**nterface. In this context, it is a special LabVIEW VI that can execute external C code. This enables hardware access, for example.
	Shell	A command line interface to the Linux operating system, e.g. PuTTY.
	PuTTY	Is a **S**ecure **Sh**ell (SSH) to establish a connection to Linux.
	WebDAV	**Web**-based **D**istributed **A**uthoring and **V**ersioning: Data exchange with a Linux system.
Chapter 3	I/O-Component	Here: components that integrates sensor **I**nput and actuator **O**utput.
	Memory Mapped	Fast, parallel access from the microcontroller to an I/O module via data bus.
	Synchronous	Here: Timed serial communication, e.g. SPI.
	Asynchronous	Here: Time-delayed communication, synchronisation with start/stop bit.
	SPI	Serial synchronous and very fast onboard bus for I/O modules.
	I^2C	TWI (Two-Wire Interface), originally I^2C: two-wire onboard bus for I/O.
	UART	**U**niversal, **A**synchronous **R**eceiver & **T**ransmitter for asynchronous data transmission. Very common for RS232, for example.
	A/D-Converter	Analogue converter, more precisely : Analogue to digital converter.
	Microcontroller	The NI SOM here is an ARM Cortex-A9 microcontroller.

	BGA-Connector	**B**all **G**rid **A**rray: a type of housing in which the connections are located at the bottom of the component. This is in the form of small solder beads that are arranged next to each other in a grid of columns and rows. In this way, a large number of connections are possible in a small space. However, such components can only be soldered by machine.
	Schematics	The graphical representation of an electronic circuit with components.
	Layout	Geometric arrangement of components/parts on an electronic circuit board and connecting them according to the diagram.
	Datasheet	This is a product description with properties, functions, specifications and materials of an electronic component.
	Application Notes	Detailed description of how an electronic component described in the data sheet is used in a specific application or use case.
	Billd of Materials (BOM)	A list of all electronic components contained in an assembly. It is the interface between development and production.
	IPC-Standards, Footprint	IPC creates standards for the component pads in the layout so that they can be easily produced by different electronics manufacturers.
	Multilayer	A printed circuit board consisting of several layers (4,6,8,12, etc).
	Signal Integrity	When the signals on the circuit board along the conductor tracks behave exactly as desired in terms of voltage and current.
	SMT and SMD	**SMT**: **S**urface **M**ounted **T**echnology. **SMD**: **S**urface **M**ounted **D**evices.
Chapter 4	SEARAY	This is a 320-pin connector between two circuit boards that enables fast signals. The pitch of the BGA (see above) is 1.27mm.
	EMS	**E**lectronic **M**anufacturing **S**ervices.
	NPI	**N**ew **P**roduct **I**ntroduction. A product innovation is optimally transferred to series production.
	Batch Size 1	Production of individual items in one-piece flow. The background to this is a growing customer need to customise products. Changeover costs are reduced to a minimum thanks to lean processes.
	Solder Paste Printing	This is the very first step in SMT assembly. The solder paste - a mixture of tiny solder balls and flux - is applied precisely to the PCB to be assembled using a metal stencil.
	SMD Assembly	A pick & place robot picks the components from a reel with a gripper, measures them with an optical camera and places them on the board. That's why it's called pick & place. The components remain attached to the sticky solder paste until they are soldered.
	Reflow Soldering	The solder paste applied by the solder paste printer is re-melted/re-flowed in an oven via a controlled temperature profile and connects the solder connections of the components to the pads/landing surfaces of the PCB.
	THT-Assembly	This is called "**T**hrough-**H**ole **T**echnology" and is used when wired components are inserted into the circuit board and soldered over the wave.
	Wave Soldering	A previously hand-assembled circuit board passes through a solder bath on a conveyor belt. The entire underside of the circuit board is wetted by a literal wave of solder tin.

Chapter 5	EDA	**E**lectronic **D**esign **A**utomation: Software for the development of electronic hardware with schematic and layout. This is a sub-area of CAD.
	One-Stop-Shop	A customer receives several services from a single source. In this case, they can have customised LabVIEW hardware developed and produced and also receive the low-level drivers. This allows them to concentrate on the LabVIEW application from the very first hour.
	Low-Volume/ High-Mix	Here: electronics production of small batches (e.g. 50 units) of different products with several changeover operations during one working day.
	EMC Laboratory	A laboratory in which devices are tested for their immunity to interference and emitted interference. **EMC**: **E**lectro-**M**agnetic-**C**ompatibility.
	Device Driver	Hardware-related control of electronic components, e.g. AD converters.
Chapter 6	High-Level-VI	A function block in LabVIEW that calls subroutines.
	FPGA	Reconfigurable logic: **F**ield **P**rogrammable **G**ate **A**rray.
	C-Source	Source code written in the "C" programming language.
	Eclipse-IDE	An integrated development environment for many programming languages, e.g. C
	Shared-Object	A code library under Linux (*.so).
	DLL	**D**ynamic **L**inked **L**ibrary: a code library under Windows (*.dll).
Chapter 7	Artix-7-FPGA	Alongside the microcontroller, a key component of the ZYNQ-7000 family.
	Gates	Logic circuits/cells in an FPGA: about 85,000 in the SOM.
	FFT	**F**ast **F**ourier **T**ransformation: converts signals from the time domain to the frequency domain.
	Ticks	A clock cycle in the FPGA in the order of nanoseconds.
	Timing	Here it is a time synchronisation of different software parts.
Chapter 8	Digital Filters	Amplify or attenuate certain frequencies in a signal.
	Combinatorial Logic	The output value depends directly on the input value.
	Sequential Logic	Output values also depend on stored states.
	Fixed-Point	Consists of a fixed number of digits before and after the decimal point. The position of the decimal point is fixed. Advantage: can be calculated much faster than floating point.
	Floating-Point	A number that is split into an exponent for the order of magnitude and a mantissa. Used for the practical and dynamic notation of very large and very small numbers.
	VHDL-Code	**V**ery High Speed Integrated Cirquits **H**ardware **D**escription **L**anguage: Hardware description language for FPGAs.
	Bitfile	Contains all the information of the LabVIEW block diagram, to configure the logic of the FPGA.
	Firmware	Software that issues machine commands to the hardware components.

	LabVIEW Realtime	A LabVIEW variant that is executed in real time down to microseconds on a microcontroller.
	LabVIEW FPGA	A LabVIEW variant that is executed in real time down to [ns] on an FPGA.
Chapter 9	Multitasking	Execution of multiple, independent tasks simultaneously.
	Overhead	Data that is not part of the user data but is required as additional information, e.g. for the abstraction of graphical code.
	Error handling	Intercept errors in a targeted manner and react to them in a controlled manner.
	Zbrain	Platform for graphical programming with LabVIEW Embedded, combined with a toolbox of modular hardware and scalable software as well as an SDK (Software Development Kit) from NI partner Schmid Elektronik.
	MVPs	**M**inimum **V**iable **P**roducts contain only the most necessary functions and are often needed to test an idea.
	24/7	Continuous industrial operation: the system runs around the clock.
	ADC-Blocksampling	Analogue values are recorded in blocks. High sampling rates are possible within these blocks. These blocks are then processed before the next block is recorded.
	Low-Power mW	Scalable, energy-saving operation of embedded systems.
	RTC	**R**eal-**T**ime-**C**lock
	Coprocessor	A second microcontroller in addition to the main microcontroller. In this context, a low-power microcontroller controls the much more powerful main microcontroller. The system can then switch back and forth between low power and high power as required.
	C-Generator	In this context, C code is generated from the graphical LabVIEW code, which is then converted into firmware using standard tools. EOL since 2017.
	8Bit-Small Targets	Miniature microprocessors that process 8 bits during one clock cycle.
Appendix	Cyber Security	Measures to protect computers against malicious attacks.
	SELinux	A Linux kernel extension that controls access to certain resources.
	Best Practices	Best practice based on experience.
	ARM-Architecture	**A**dvanced **R**ISC **M**achine: these are microcontrollers based on the RISC architecture (RISC: **R**educed **I**nstruction **S**et **C**omputer) and dominate in embedded systems. They take a different approach to hardware design than the better-known x86 architecture.
	Linux Kernel 4.14	A kernel from 2017 with long-term support until 2024.
	SUMO Distribution	The NI Linux RT Distribution for the System on Module.
	RTOS	**R**eal-**T**ime **O**perating **S**ystem.

Münchwilen, Switzerland, June 2025, Marco Schmid, marco.schmid@schmid-elektronik.ch