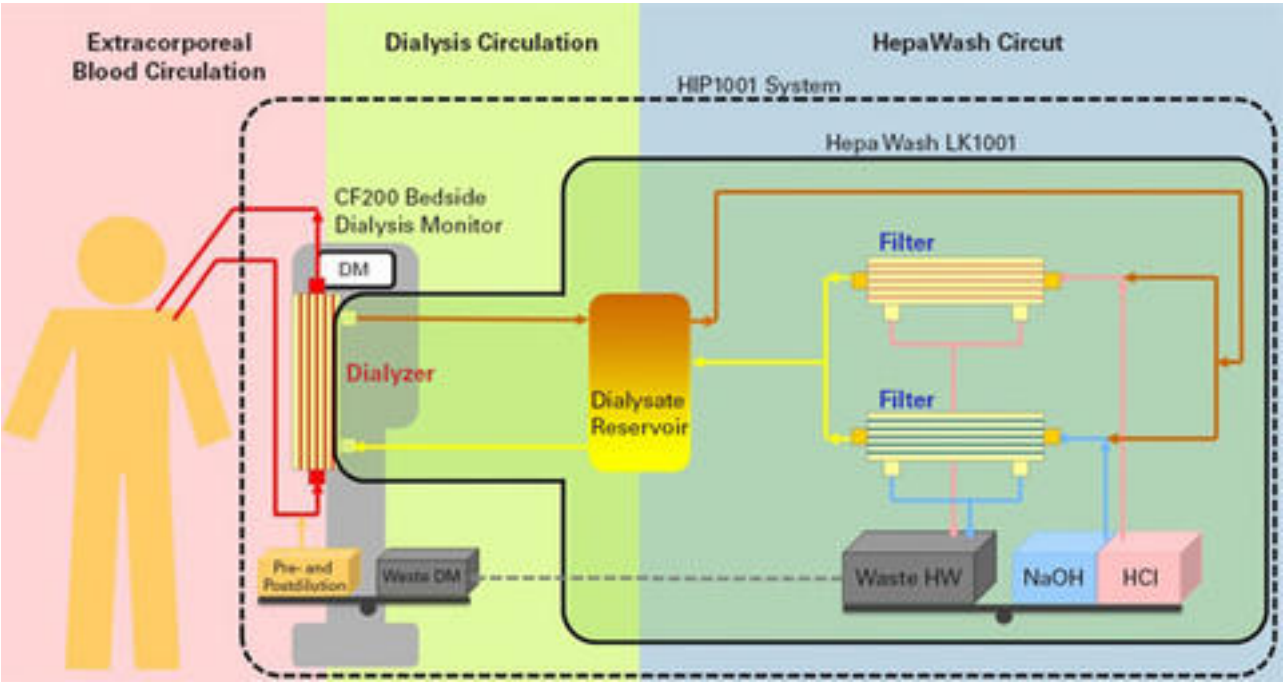
Using LabVIEW and CompactRIO to Create a Liver Dialysis Prototype

National Instruments

Patients with chronic kidney failure can considerably extend their life expectancy with the help of dialysis or a kidney transplant. Patients with liver disease need a comparable therapy to dialysis as an alternative to a liver transplant. Kidney dialysis removes water-based toxins from the patient's blood and liver dialysis needs to eliminate protein-linked toxins.

To lower the high mortality rate of liver disease patients, a dialysis method was developed where the dialysis fluid called dialysate contains the human transporter protein albumin, which binds the toxins from the blood. For repeated use during the course of a dialysis treatment, the albumin-dialysate is continuously purified in a separate fluid circuit by adding a base and an acid to dissolve the toxins and subsequently filtrate the dissolved toxins. This technique offers a cost-efficient therapy with a significantly elevated detoxification capacity compared to other liver dialysis treatments available.



Regulatory Approval Process

According to the European Medical Device Directive 93/42/EEC, the life-sustaining dialysis machine we created is designated a Class IIb medical device. The essential regulatory requirements for device development include mechanical and electrical safety as well as technical and design dossier documentation. We followed the IEC 60601 standards for medical electrical devices and the IEC 62304 standard for

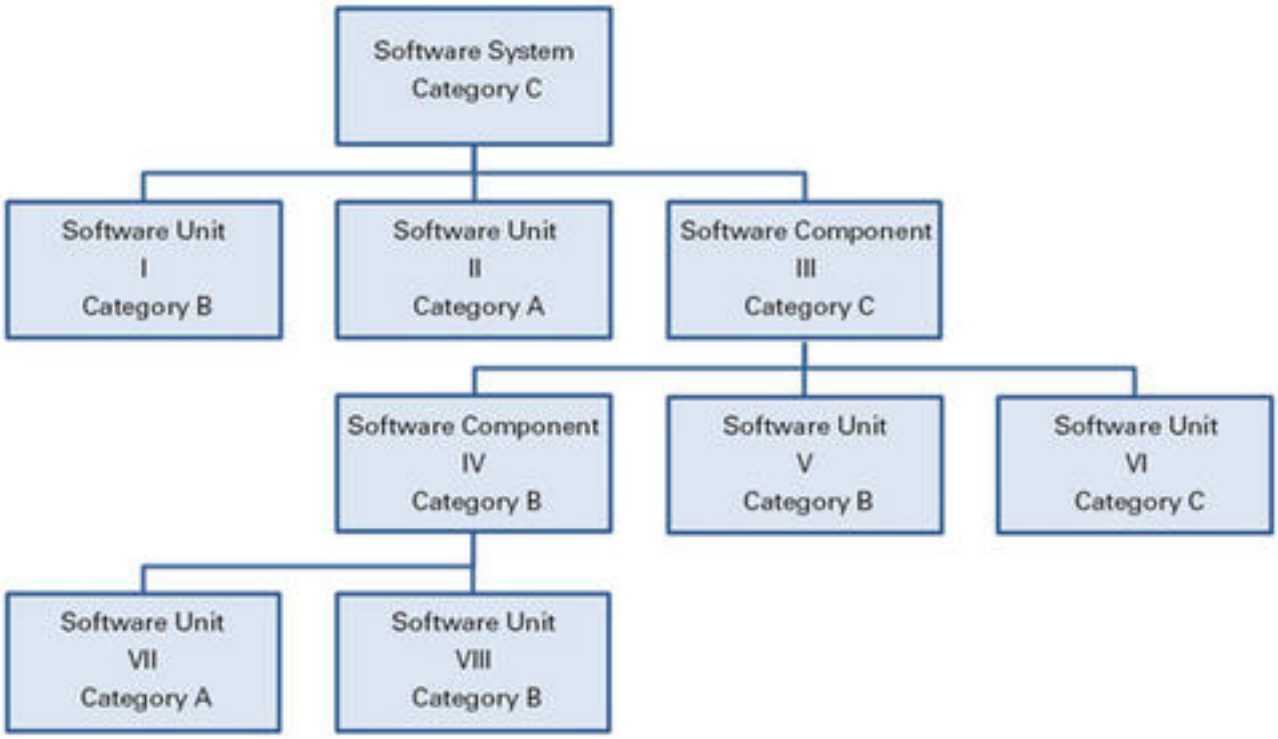
medical device software lifecycle processes.

While the IEC 60601 standard has precise, testable requirements regarding mechanics and hardware such as electromagnetic compatibility or protection against electric shock or radiation, the IEC 62304 standard is a process norm that defines the requirements regarding software development, testing, and maintenance. According to the IEC 62304, for instance, we are required to define and document a detailed software architecture dividing the software system into components and units, each of which is assigned to one of the following three IEC 62304 software safety classes:

A) No injuries or damage to health is possible

B) Non-serious injury is possible

C) Death or serious injury is possible



The granularity of the software architecture is, according to the IEC 62304 standard, is left to the manufacturer. While software classified as safety class A only needs the documentation of requirement specification and test for fulfillment of the requirements, software classified as safety class C also requires detailed design documentation, as well as testing of each individual unit regarding data and control flow, variable initialization, memory overflows, and fault handling. This applies to proprietary developed systems as well as to software of unknown provenance (SOUP) - if used for safety in a class C component.

System Development Using SOUP

SOUP is software that was not developed by the medical device manufacturer, or

for which there is no complete documentation. SOUP can be every component of the software system—even the OS, drivers, and applied library features within the application.

According to IEC 62304, for software components belonging to safety class C, we had three options: a) use either an IEC 62304-certified SOUP, b) use an ISO 61508 SIL2-certified SOUP, or c) design our product software without any SOUP. It is hard to find any SOUP that is certified for IEC 62304 or ISO61508 SIL2. Furthermore, the medical device manufacturers typically do not have access to the SOUP supplier development documentation, or have the ability to perform a supplier quality audit in compliance with ISO 13485.

However, according to IEC 62304, “If the RISK of death or SERIOUS INJURY arising from a software failure is subsequently reduced to an acceptable level (as defined by ISO 14971) by a hardware RISK CONTROL measure…the software safety classification may be reduced from C to B” [1, p.7]. For further safety reasons, the hardware risk control measurements should usually also be independent from the software system in regards to power supply, reference voltage, and the analog/digital converter.

HIP1001 Product Development

After achieving the proof of concept in preclinical studies, we solicited investor money to develop a human prototype within one year. The human prototype, called the HIP1001 System, consists of two subcomponents— a CE-certified dialysis machine (CF200 by Infomed) to regulate and monitor the extracorporeal blood circuit, and a completely new subassembly device, the LK1001, which implements the dialysate circuit and the Hepa Wash circuit. The extracorporeal blood circuit and the dialysate circuit are connected through the dialysis filter (dialyzer).

The LK1001 monitors various pressure, pH, and temperature values, as well as controls valves, pumps, and heating devices using 50 sensors and actuators. Because we were successful using LabVIEW to develop the control software of the animal prototype, we used LabVIEW for the human prototype as well.

LabVIEW is suitable for developing sophisticated embedded systems for measurement, testing, controlling, and regulating. Graphic function blocks are arranged into graphical dataflow diagrams using a drag-and-drop technique. For the LK1001 software development and selection of NI hardware, we collaborated with National Instruments Alliance Partner XOn Software GmbH. XOn is not ISO 13485-certified, which is usually a prerequisite for suppliers to medical device manufacturers in Europe. To solve this issue, we trained all XOn personnel involved in this project in the relevant processes and operating procedures of our in-house quality-management system.

For development flexibility, the team chose CompactRIO hardware. The hardware consists of a reconfigurable chassis with a field-programmable gate array (FPGA); a real-time controller for deterministic communication and processing; and interchangeable analog, digital, and pulse-width modulation (PWM) modules. For

the human prototype, we used the rugged NI cRIO-9024 real-time controller with the NI cRIO-9118 backplane. For the preproduction model – currently under development - we use an NI sbRIO-9612. Software developed for the rugged unit is mostly portable to the high-volume deployment, board-only units.

HIP1001 Regulatory Standards Test and Certification

Even though NI, the controller manufacturer, is not an approved original equipment manufacturer (OEM) according to the ISO 13485 standard, we can use NI products for our medical device because the IEC 60601 testing procedures necessary for approval refer to the whole medical device system. UL 61010 and EN 61326 component certifications of NI components meant we did not need to perform individual unit tests.

The programmed control software consists of the following three main components:

- A Windows application that solely serves as the user interface and cannot interfere with the control (safety class A)
- A LabVIEW Real-Time application that runs within the CompactRIO controller and implements control functions (safety class C)
- LabVIEW FPGA code that adjusts the external ports to any externally used forms of communication, including analog, PWM, and frequency (safety class C)

As outlined above, both safety class C components were eligible to downgrade to safety class B because of supplied independent, hardware-related safety features configured and integrated as hardware risk-control measurements. This safety system subassembly monitors critical system parameters and switches to patient safety mode if critical values are exceeded. The pH and temperature values of the albumin-dialysate are monitored using a Mettler-Toledo M300 probe transmitter. Because the LK1001 is a closed-loop hydraulic system, a component of our safety system monitored the fluid balance using a Mettler-Toledo IND780 weighing terminal. Simple ANSI C++ software code that controls this device (safety class C) was tested using code review and simulation test methods defined in IEC 62304.

The notified body TÜV SÜD (reviewing agency) performed standards testing in parallel with our development. The agency performed the first electrical safety testing procedures three months after development began, and functional safety tests five months after development began. We used the NI Requirements Gateway to create cost-efficient, automatic, specially formatted Microsoft Word documents for traceability between requirements, design, and tests.

Moving From Prototype to Deployment

After TÜV SÜD accepted the system and provided a technical clearance certificate for clinical studies, we tested the human prototype in vitro and in vivo. Ten months after beginning development, we started the clinical pilot study. Due to the successful acceptance for the clinical study and the first promising results at the

clinic, we solicited additional capital to complete the preproduction model and a multicenter study.

A continuous goal for the preproduction model remained the safety system concept to keep control software and safety hardware independent from each other. We also used the NI sbRIO-9612 module, with more than 100 signal connections for analog and digital I/O (Figure 4). The human prototype development met all quality criteria for CE acceptance, meaning we can adopt large parts of the tested prototype software into the final product, and we do not need to completely recreate the existing software. However, as in the preproduction model, we replaced the CF200 with our own extracorporeal blood circuit (including safety features), expanded the control software, and replaced the existing external safety system with an internally developed intelligent safety system. Approximately 90 sensors and actuators are integrated in the real-time control system based on LabVIEW. For the safety system, the following two options meet the IEC 62304 safety class C requirements:

Creating simple LabVIEW FPGA code at the integrated circuit level: For a license fee, NI provides the LabVIEW FPGA environment documentation necessary for acceptance testing to regulatory bodies. Here, we needed to verify not only the LabVIEW graphical source code, but also the generated VHDL code.

Using a microcontroller without an OS: If we do not use library features for the software development (which means no development tool DLLs are in the executable code), the executing code does not fall in the SOUP category. In unit tests we have to examine the computer code created by the compiler. After each new compilation, we have to perform these tests. If the system is updated, we have to completely reexamine it.

Hepa Wash is financed through risk capital. Development time and cost are critical factors since no revenue is generated until market launch. The combination of LabVIEW and a hardware risk measure to monitor system-critical parameters for the new liver dialysis therapy prototype helped us develop the first system within just seven months. Hence, with NI software and hardware, we reduced time to market of our innovative liver assist device. Currently, we are developing the final device using the LabVIEW FPGA and LabVIEW Real-Time modules and NI Single-Board RIO hardware and plan to achieve the European market authorization (CE-mark) in Summer 2012.

A National Instruments Alliance Partner is a business entity independent from National Instruments and has no agency, partnership, or joint-venture relationship with National Instruments.

Source URL (retrieved on *06/18/2013 - 8:29pm*):
http://www.mdtmag.com/news/2012/11/using-labview-and-compactrio-create-liver-dialysis-prototype