Creation and Validation of Long Term Test System for Artificial Heart Mock Loops.

by
Timothy Nolan
Consultant
Data Science Automation, Inc.
USA

Category:
Biotechnology/Life Sciences

Products Used:
NI LabVIEW 8.6
Unit Testing Framework
SCXI-1600, SCXI-1100
SCXI-1581
SCXI-1102
USB-6501

The Challenge:
Create a program for extremely long term testing of artificial heart Simulated Flow Testing Systems (SFTS), making sure that the created VIs are verified and validated for medical device applications

The Solution:
Parallel loop architecture, combined with the National Instruments USB-based DAQ and signal conditioning allowed constant data display with a fast-response user GUI. The Unit Testing Framework allowed validation before, during and after full system integration.

Abstract:
This project had a two part goal:

- Create a monitoring system for artificial heart testing
- Verify and validate the created program to stringent medical program guidelines.

The system cycled through the bank of testing units, measuring data and recording alarm conditions in compliance with user specifications. Communication to the motion control and temperature control systems was handled in parallel, along with a manual user mode to take direct control at any time. After creation, each VI was tested and validated using the Unit Testing Framework, generating required reports and documentation.

Main Acquisition
In the course of testing the artificial hearts, the devices are places in flow systems which simulate arterial fluid flow (the Mock Loop). The hearts are tested for months and years at a time, with a mind to verify their use for months and years in a patient. Up to 18 simultaneous loops are run, all of which need to have data recorded, analyzed and logged on a regular basis. Due to centralized data gathering, as well as proprietary systems for flow probe measurements, a multiplexer system switches the flow sensor wires and serial communication cables each to a single set of data acquisition inputs. Each loop needs to reliably record 10 beats of data three times a day, one for each ‘Mode’ of the system (described in the next section), regardless of alarm conditions or user override or interrupt.

Motion Control

375 Valley Brook Road • Suite 106 • McMurray, PA 15317 • T 724-942-6330 • F 724-942-8390 • DSAutomation.com

During the daily course of testing, the system switches through three 'Modes' of linear motion in the loop, simulating the action of the patients' native heart. Theses modes are defined by the user in a common text file, showing beat rate, duration and duty cycle. These settings need to be communicated to a third party motion control system known as a PMAC, one per each mock loop. Communication is accomplished over RS-232 serial communication, and needs several steps to accomplish. The correct MLTS needs to be selected in the multiplexer (through digital comman), which routes the single computer serial port to the corresponding PMAC. Serial communication compares the desired with actual settings, and updates them as needed.

Alarming
While the data is only gathered three times a day, the system needs to continuously monitor the pressure, flow and temperature of each loop. If any of these values exceed user-defined values, an alarm condition must be activated. In this case, the current automated progression through the loops, and multiplexer, must be overridden, and an extra data file must be gathered and written to the alarm log. In addition, a message is sent to the user, which must be cleared before any other user input is accepted.

Manual Mode
For diagnostic purposes, the user can also manually interrupt the automated monitoring and logging of the system. While in manual mode, any specific MLTS can be examined, and all alarms are delayed until normal operation of the system resumes.

Unit Testing Framework
To meet the specifications for developing software for medical devices, each individual part of the program had to undergo validation testing. As each VI was completed it needed to tested, with an appropriate report generated to demonstrate its validity. While it would be possible to perform this tests manually, the Unit Testing framework saved much time and effort, especially for repeated tests. The Framework allowed the testing and documentation to be produced very quickly, as well as allowing easy return and re-test in the case of a software change during integration. The Unit Test Framework's flexibility and ease of use certainly enabled much faster and reliable completion of this portion of the project.

375 Valley Brook Road • Suite 106 • McMurray, PA 15317 • T 724-942-6330 • F 724-942-8390 • DSAutomation.com