REDUNDANCY FOR EPICS IOCS

Matthias Clausen, Gongfa Liu, Bernd Schoeneburg, DESY, Hamburg

Abstract

High availability is driving the reliability demands for today's control systems. Commercial control systems are tackling these requirements by redundant implementations of major components. Design and implementation of redundant Input Output Controllers (IOCs) for EPICS will open new control regimes also for the EPICS collaboration. The origin of this development is the new XFEL project at DESY. The demands on the availability for the machine uptime are extremely high (99.8%) and can only be achieved if all the utility supplies are permanently available 24/7. This paper will describe the implementation of redundant EPICS IOCs at DESY that shall replace the existing redundant commercial systems for cryogenic controls. Special technical solutions are necessary to synchronize continuous control process databases (e.g., PID). Synchronization of sequence programs demands similar technical solutions. All of these update mechanisms must be supervised by a redundancy monitor task (RMT) that implements a hard-coded expert system that has to fulfill the essential failover criteria: A failover may only occur if the new state is providing more reliable operations than the current state.

OVERVIEW

A redundant IOC system consists of two IOCs. The communication between the IOCs is implemented to support two separate Ethernets, the public Ethernet and the private Ethernet. The redundant pair shares these two Ethernet connections for monitoring the health of the partner and to synchronize the data. A third Ethernet connection, the global Ethernet connection, is established to monitor the availability of higher-ranked network servers, e.g. boot server. The global Ethernet uses the same network device as the public one. An overview of the hardware components is shown in figure 1.

There are three major elements in the software design: The Redundancy Monitor Task (RMT), the Continuous Control Executive (CCE), and the State Notation Language (SNL) Executive. Modifications of the existing applications, like the SNL-Executive, are required to enable synchronization. This includes status information from the drivers that communicate to the hardware, the runtime database and SNL-program state and its internal variable information [1].

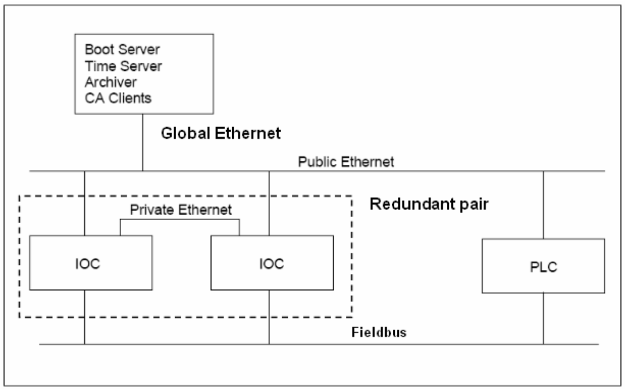

Figure 1: Hardware layout of a redundant IOC system.

RMT

The RMT establishes and maintains communications with the partner IOC. It also controls the drivers that have an impact on the mastership decision. With the information from both of these sources or a command from the operator, it decides when to assume or relinquish control.

To determine the overall condition of the IOC, the RMT examines the status of the important resources. These resources are called Primary Redundancy Resources (PRR), which include the public Ethernet, the private Ethernet, the global Ethernet, device drivers, CA server, scan tasks, CCE, sequencer, SNL executive, etc.

In principle the number of PRRs to be supervised is unlimited. For a flexible and secure solution, a design is chosen wherein each resource has one thread (PRR Controller) instanciated. Each thread performs its check and saves the result in a control table. The threads are triggered by one main thread, which obtains (generates) the overall condition of the IOC by observing the results in the control table.

For a simplification of the internal RMT set-up, an identical interface (Driver IF) is designed. For details, see the section "Driver interface".

For the appraisal of the status of the RMT itself, the RMT triggers a hardware watch dog. This will reboot the system in case of an RMT failure.

The RMT contains a state machine which implements the algorithm of the redundancy transitions. The RMT can be controlled by callable functions from the shell. The configuration is read from a configuration file.

For remote control of the RMT, an XML-task is implemented which provides XML communication over a TCP/IP-port on the public Ethernet [2].

An overview of the software components is shown in figure 2.

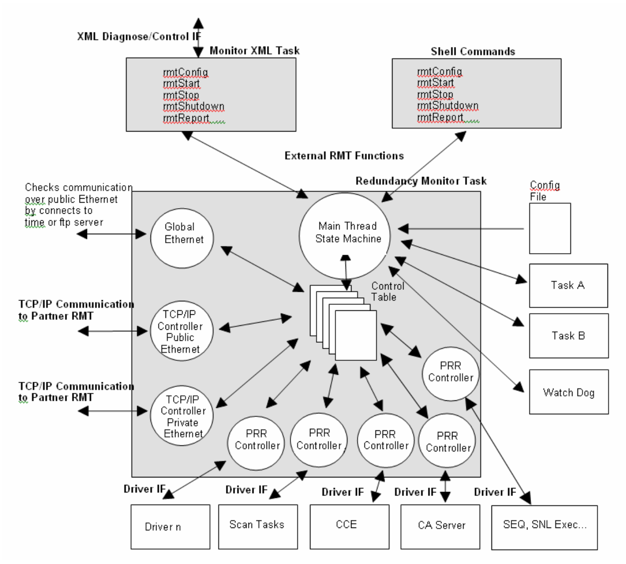

Figure 2: Process and interface design of RMT.

INTERFACE BETWEEN PRRS AND RMT

A PRR is a software component that can be a major part of the EPICS IOC software such as the CCE and the SNL Executive. Other PRRs are the IO-drivers. All kind of components can share the same interface to the RMT. Some parts of the interface are useful for drivers only. Not all parts must be implemented for a particular component. The interface will be implemented as functions defined in the component and callable by the RMT. The addresses of these functions will be in an entry table of the component. In this way all PRRs will have their own methods for a fixed set of commands. During initialization the component first checks if the IOC is redundant. This information is stored in an environment variable. Other OS-independent solutions are in discussion. In case of redundancy the PRR calls rmtRegisterDriver() with the address of the entry table as an argument. If the IOC is not redundant the component works normally (start). In a redundant IOC the component goes to the stopped-state and wait for commands. This allows the use of the same code for redundant and non-redundant IOCs.

A header file "rmtDrvIf.h" defines the interface to the RMT. The following functions can be used by the RMT to send commands to a driver instance or to get information in a format which is defined in the common header file. Since numerous instances of a driver can exist, the functions need a pointer to the driver's internal data to control the desired instance. The RMT stores these pointers during the registration and handles them as void*. The functions can interpret it as a pointer to the driver's private data. Functions are "start", "stop", "testIO", "getStatus", "shutdown", "getUpdate", "startUpdate", "stopUpdate" and "getInfo" [3].

REGISTERED PRRS WITH DRIVER IF

CA Server

There are two types of CA Servers: (1) RSRV is a server for IOCs and Soft IOCs; (2) CAS is a Channel Access Server or Portable server. RSRV is described here. "CAS-TCP", "CAS-beacon" and "CAS-UDP" are 3 tasks spawned at RSRV initialization, while "CAS-client" and "CAS-event" are a pair of tasks spawned when a client connection is set up.

The task "CAS-TCP" is registered. When the IOC is slave, "CAS-TCP", "CAS-beacon" and "CAS-UDP" are frozen by using a flag and all task pairs "CAS-client" and "CAS-event" are deleted. Therefore RSRV does not respond to any client connection request and disconnect all client connections. When the IOC is in the master state, all these tasks work normally. RSRV can accept any client connection requests like in the non redundant case.

Scan tasks

The periodic scan tasks register normally at the RMT during their initialization. There are seven tasks (threads) of this kind. When the IOC is in the slave state, the RMT pauses their activities.

CCE

The main task of the CCE is to keep the IOC database synchronized.

The internal data structures „record blocks" and "field blocks" are constructed at CCE initialization on both IOCs. Record blocks contain a list of pointers to record update structures. The list is sorted by record address when it is created. Each update requires a binary search of the list to find the beginning of the chain of pointer for the field updates for that record. Field Blocks contains the current value of each field and its last sent value. If the field needs a continuous update and the current value differs from the last sent value, the field data is transferred and the current is copied to the last sent. Another internal data structure is "partner record blocks" which is used on the slave IOC. It is an array of pointers ordered by the master IOC's record pointer. The master IOC's record pointer is sent as a handle on every field update for that point.

The CCE attempts to connect to its partner. When a connection is established each unit transitions state to "synching". They stay in this state until the CCE on the master IOC has completed sending a full update to its partner. Then both units transition to "in-sync state". In this state CCE on the master IOC periodically transfers all fields that have been changed [4].

Sequencer

The sequencer provides run-time support for implementing state transition diagrams in an EPICS environment. It is now unbundled from EPICS base.

The task "seqAux" is spawned under vxWorks when the sequencer is started. After the "seqAux" is registered, the sequencer is activated by RMT when the IOC is master, otherwise inactivated.

SNL Executive

The purpose of the SNL Executive is to keep the state program of both IOCs synchronized. This includes variable values and states.

A function of the seq-package is called to construct the internal data structure. This structure is a index of state program, from it all sequence private data structures SPROG (hold all information about a state program), CHAN (hold information about a database channel), SSCB (hold information for a state set), STATE (hold information about a state) can be accessed.

The SNL Executive attempts to connect to its partner via private Ethernet. After a connection is established, the SNL Executive on the master IOC sends the data to its partner periodically, and SNL Executive on the slave IOC updates the corresponding state program data after the match check.

SNL DEBUGGER

A Control System Studio (CSS) plug-in sends an XML stream to RMT for diagnosing the running state programs. A state program can include several separate state sets, in turn, a state set includes several states. Under vxWorks one task is spawned for each state set. Up to now, the following functions are implemented:

(1) query the information of a state program: state sets, their active states, db channels and variables
(2) set the value of a variable when the IOC is master
(3) jump to any state of a state set when the IOC is master and the state set is not suspended
(4) control the run mode of a state set: suspend/resume/single-step when the IOC is master.

A major part of the debugger is based on ideas from the SLAC implementation of their new sequencer version.

TEST

A prototype system is setup, which consists of 2 SMA CompactPCI CPU modules with vxWorks-5.5, EPICS base-3.14.8.2 and seq-2.0.11. Some tests have done and the figure 3 is the DM2K interface.

The triangle waveform of the DM2K interface shows an analog output (ao) record's value, which is controlled by a state program.

Two switch-over events happen and the reconnection time is about 30 seconds. This is a result of the CA timeout management,. The default value of the parameter EPICS_CA_CONN_TMO is 30 seconds. The switch-over event shows that RSRV is controlled by RMT.

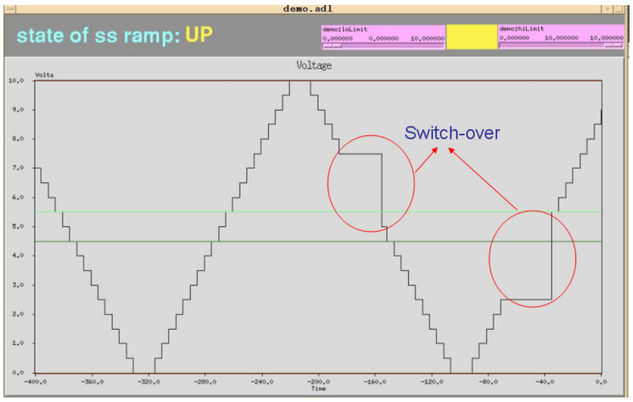

Figure 3: DM2K interface for test.

The value of the ao record is continuous when the switch-over event happens. This shows that the IOC database and the state program data are synchronized, i.e. CCE and SNL Executive do work.

SUMMARY

The support for redundant IOCs opens a new regime of control applications to the EPICS community. High Availability applications like the 24/7 operation of cryogenic plants is no longer only the regime of commercial implementations. Redundant IOCs also play their role in todays facilities where the demands for high availability are reaching 99,8%.

Since the implementation of the Redundancy Monitor Task is independent from the EPICS runtime environment it is possible to use the implementation also for other applications.

Porting the redundancy support to Lunix and Mac-OS opens it's usages to new frontiers.

REFERENCES

[1] John L. Dalesio, Leo R. Dalesio, "IOC Redundancy Design Doc", internal report, Sep. 2005.
[2] Andreas Leymannek, "Redundancy Monitor Task (RMT)", internal report, Sep.25, 2006.
[3] Bernd Schoeneburg, "API for the Redundancy Monitor Task", internal report, Jun.26, 2006.
[4] John L. Dalesio, Leo R. Dalesio. "Continuous Control Exec Implementation Doc", internal report, 2006.