SSCL-N-860

EPICS Database Development Using ECAD Software

C. Cuevas, J. Heefner, G. Melton, and L. Plesea

Superconducting Super Collider Laboratory*
2550 Beckleymeade Ave.
Dallas, TX 75237, USA

May 1994

*Operated by the Universities Research Association, Inc., for the U.S. Department of Energy under Contract No. DE-AC35-89ER40486.

Superconducting Super Collider Laboratory
Accelerator Division
2550 Beckleymeade Avenue, MS-1046
Dallas, TX 75237-3946

Document/Records Submission Form

Title: EPICS Database Development Using ECAD Software

Originating Organization: ASD/Controls Department

Author(s) Title

C. Cuevas
J. Heefner
G. Melton
L. Plesea

Date: April 29, 1994 **Submitted By:** Huan Nguyen

Document/Record Status: **Released By:** **Draft:**

Control Status: **Document/Record Type:**

Document/Record Number:

WBS Number

Alternate Document/Record Number(s): **Keywords:**

1) 1) EPICS

2) 2) CAD

3) 3) CAPFAST

4) 4)

Content Description

This document will discuss a method for using an ECAD software package to develop Experimental Physics and Industrial Control System (EPICS) databases.

Remarks

EPICS Database Development Using ECAD Software

C. Cuevas, J. Heefner, G. Melton, L. Plesea
ASD Controls Department

INTRODUCTION

Experimental Physics and Industrial Control System (EPICS) provides an interface to instrumentation for data acquisition, supervisory control, and state notation. EPICS includes the following features:

- Operator interface to all control system parameters through an interactive display editor.
- Data logging through a table entry archiving file.
- Alarm Management through a table entry alarm file.
- Sequential control through a state definition language with convenient database interface routines.
- Channel access as a protocol for data access.
- Database Configuration Tool (DCT) which is used to generate the EPICS databases.

In its current form, the EPICS development interface displays one database record at a time in table entry format. Because the displayed record is in the text mode, the flow for links, loops, connections, data is not visually represented. Debugging and documentation requires printing the entire database, one page per database record, then manually reviewing each page for the aforementioned information. The CapFast advantages listed below represent enhancements to EPICS database development.

The purpose of this paper is to demonstrate a method for using an ECAD software package (CapFast) to develop EPICS databases. CapFast is a trademark of Phase Three Logic, Inc. CapFast is schematic capture program that can be used to generate EDIF files which are then translated into EPICS short form reports. EDIF tools, consisting of an EDIF symbol reader and an EDIF symbol/schematic/netlist writer, allow the translation of data between CapFast on the PC or Sun workstation and other CAE tools and platforms. The short form reports are used by DCT to generate an EPICS database. Some of the advantages and disadvantages for using CapFast as an EPICS development tool are as follows:

Advantages

- Intuitive
- Reusability
- Error Control
- Easy Debugging
- Self Documenting
- Hierarchical Development
- Multi-tasking Development
- Reduced Development Time
- Object Oriented Development
- Configuration / Version Control
- Parallel Development of hardware and logic databases

Disadvantages

- In most cases it will require an increased number of database records over using DCT alone.

DESCRIPTION

The main goal of using CapFast is to facilitate the development of the complex EPICS databases. From the system engineer's point of view, development is simplified when it is possible to work with replicas of the controlled equipment. For example, the system integrator requirement is, "When the pressure reported by the ion gauge installed on the vacuum foreline exceeds $1e^{-3}$ Pa, a vacuum failure has occurred and valve VLV_1 must be closed to protect the system". The primary idea is to create a mirror image of the physical, controlled equipment and

environment. The mechanisms employed for obtaining this goal are the same as those used in object-oriented programming.

The names generated by CapFast are hierarchical and can match a naming convention. There can also be a fully hierarchical general purpose symbol library, that executes a generic function (logic, arithmetic, data processing). A design is considered to be hierarchical when it contains symbols that represent underlying schematics (non-primitive symbols). In this way, quick prototyping is possible and in addition, improvements in a symbol functionality can be incorporated into schematics developed with the respective symbol by simply regenerating the database.The following figure illustrates the process for using CapFast to generate an EPICS database.

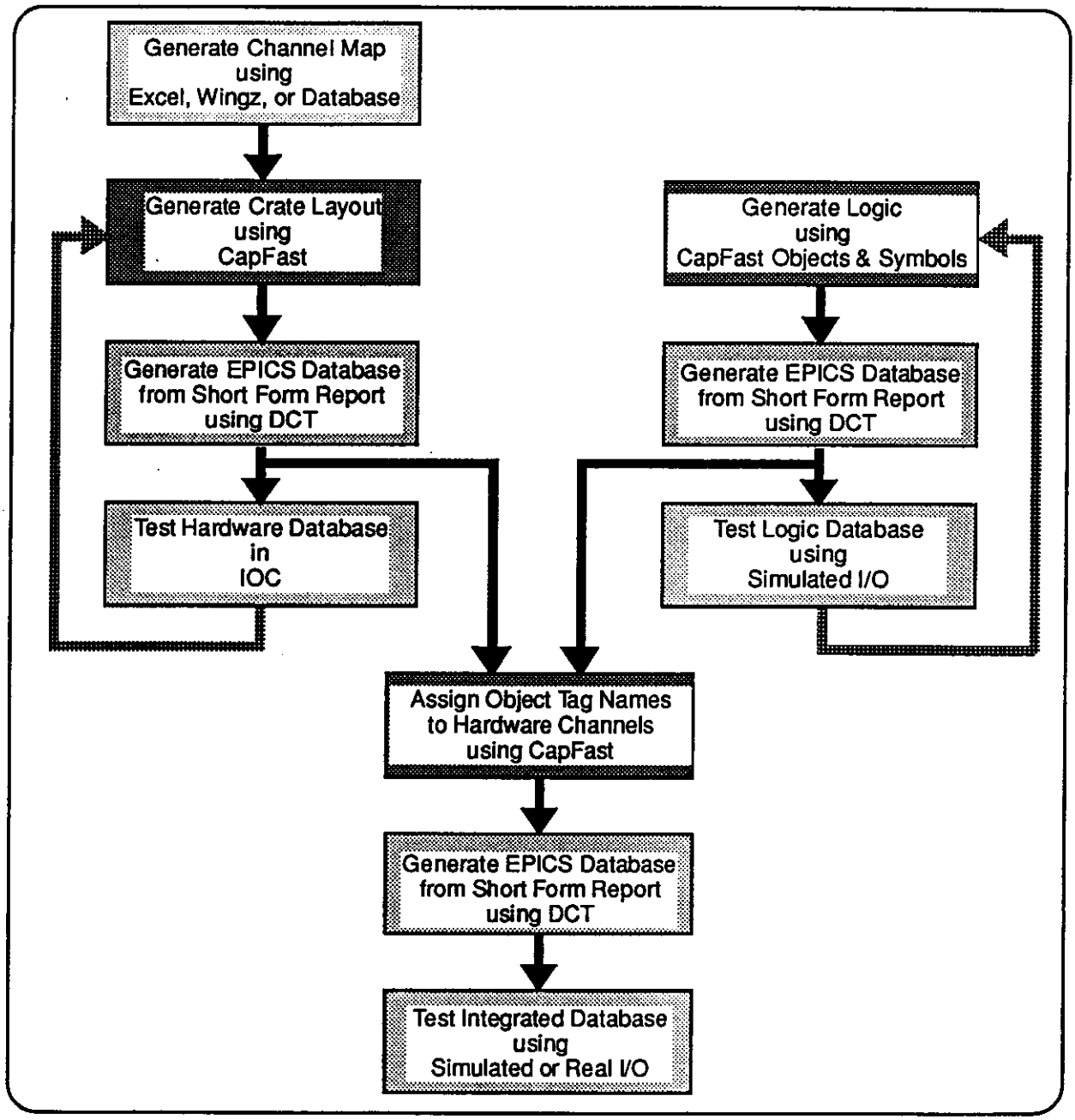

Figure 1 EPICS Database Generation Using CapFast.

THE GENERATION PROCESS

The database generation process using CapFast requires three steps: hardware configuration, logic generation, and integration of the hardware and logic. The hardware configuration process involves assigning modules to the VME crates and assigning the actual I/O tag name to the hardware channels. The logic generation process involves configuring the logic and objects in the required functionality. The integration process is then used to tie together and test the hardware and logic databases.

DATABASES

The computer interface equipment is hidden from the application engineer keeping the database development system uncomplicated. To satisfy this requirement, the database development is divided into two categories. This allows the parallel development of the logic databases by the applications engineer and the computer interface databases by the controls engineer. CapFast drawings, which represent databases, are functionally separated into two levels, hardware and logic. The hardware database represents the computer interface and is structured as a mirror image of the rack/crate/board, and the second database is formed by the application logic and resembles the control/controlled equipment, and all the records connected with it (scaling, calculations, subroutine records, etc.). The following paragraphs describe each database.

Hardware Database

The hardware database hierarchy has several levels. The first (bottom) is the channel level, where an input and/or output record from the EPICS symbol library is preset with parameters matching the schematic conventions. Figure 2 is an example of this level.

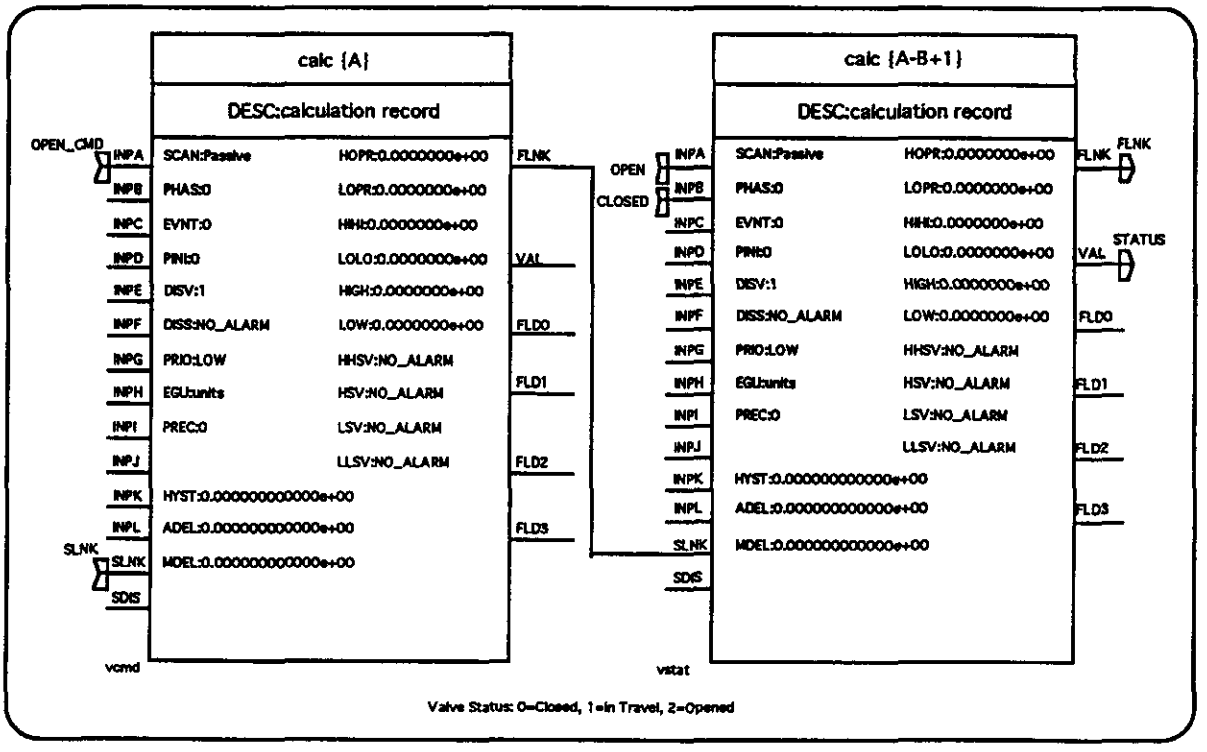

Figure 2 Valve Schematic

This drawing is then connected to a symbol that follows the design rules and defines the particular channel of the control device that is used to control or monitor a particular field device and provide a quick link between the hardware and software.

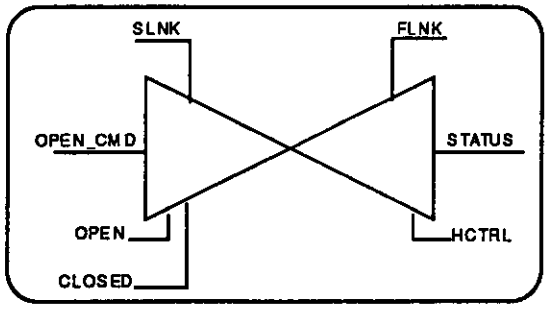

Figure 3 Valve Symbol

This feature provides for simplified field cable troubleshooting. This layer is formed from only two symbols, a hardware input and a hardware output. Each symbol specifies the board number within the mainframe, and the channel number within the board. The symbols can be easily reassigned, a feature which gives the application developer independence from the actual control system wiring. These features also provide a fast method for system reconfiguration in the event of channel failure necessitating channel reassignment. The symbols may be used as a link between the wiring schematics and the control logic schematics.

The second level is the card level, which is a drawing with the necessary number of channels to form a complete circuit board (VME Module). All the channels associated with a particular board are present in this layer. This is the level where the off-page links are used to assign uses for each channel. The tag name includes the channel number. There is a different schematic for each board that is supported. The board and channel numbers are specified on each schematic.

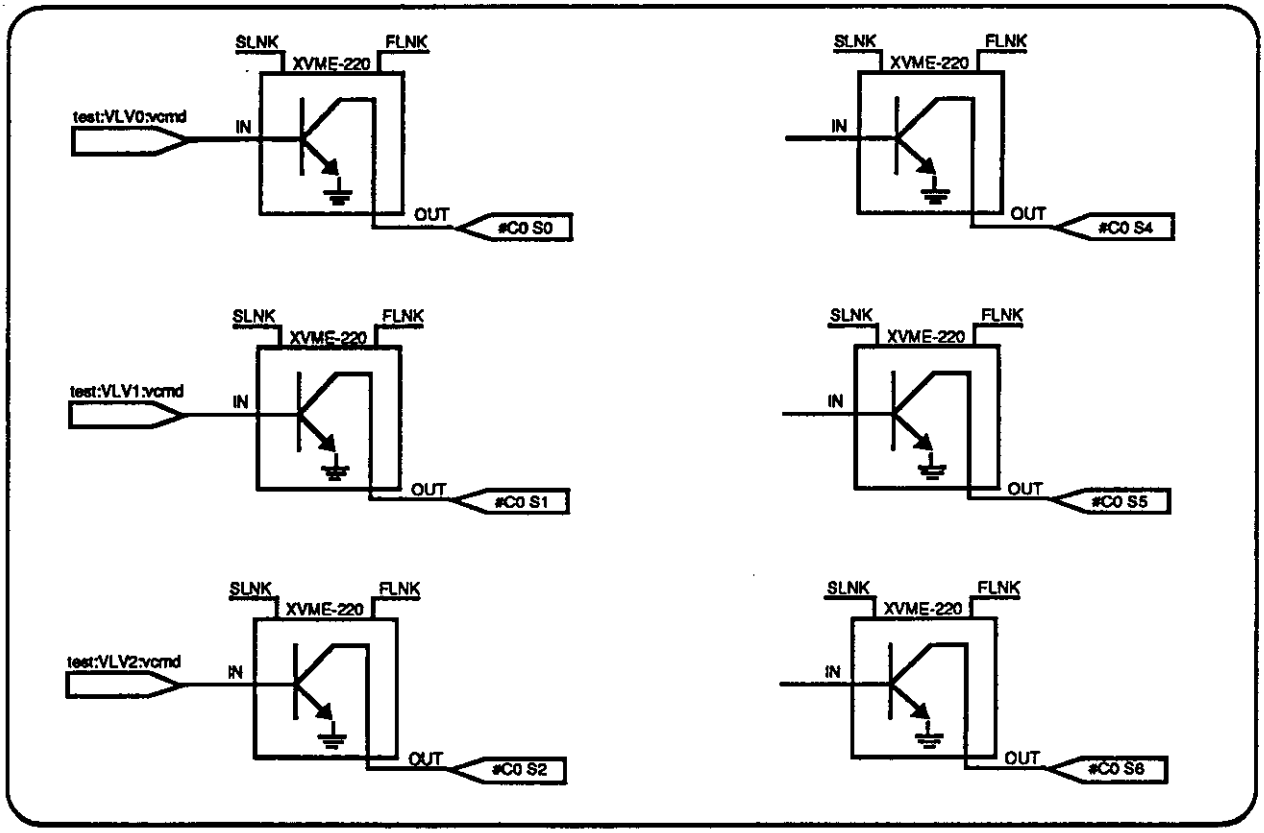

Figure 4 X220 Instance 0 Schematic

The third (top) level of the control interface database is equivalent to the crate layout.

In this way, the names for various hardware database records are generated by CapFast using the following format:

<crate_name>:<board_type_and_number>:<channel_number>:<EPICS_record_type>

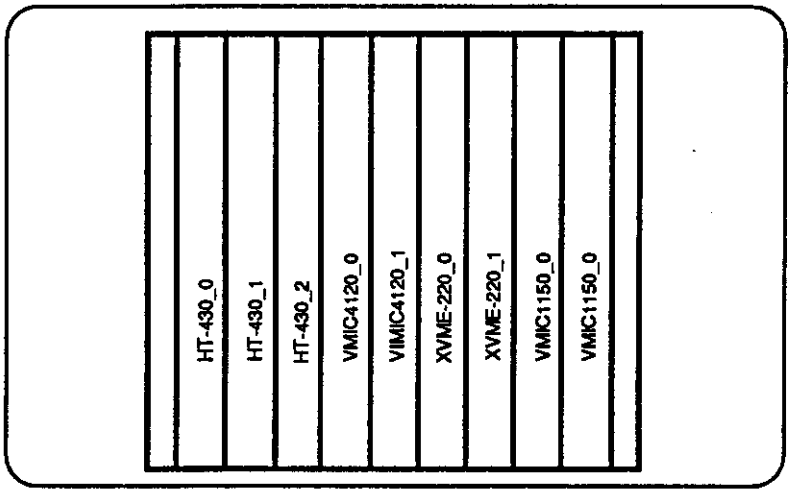

Figure 5 Crate Layout

For example, channel 34 of the second Xycom 220 binary out board from crate RCT02 will be labeled:

RCT02:X_220_1:CH_34:ebo_0

Logic Database

The logic database also has a hierarchical structure. In the first (primitives) level, all of the EPICS records are present. In the subsequent levels, the objects from the controlled space are created as symbols and those symbols include all of the device functions. For example a simple open and close valve (controller) will have a logic input for open and close commands and a logic output with the open/traveling/close states. The physical connections needed to get the status information and control the valve are two binary inputs (open and close limit switches) and one binary output (control line). The logic inside the valve will be one calculation record that passes the command and one calculation record that accepts both sensor data and converts it according to a predefined rule.

At this level, the device type and hardware interface are specified and the basic conversions are performed. This level is intended for system engineers and the parts shall be standardized and thoroughly checked. This minimizes the task of software debugging. This layer is formed from a large number of symbols and provides total hardware independence for the application layer. This layer also makes the hardware testing application independent of system setup.

There can be various degrees of complexity depending on whether a device contains underlying symbols or only EPICS primitive records. For example, a variation of a valve controller can be generated by including a differential pressure mechanism for automatic valve closure if the pressure on either side of the valve rises above a certain limit, and an override mechanism is incorporated that permits keeping the valve open, even above the established limit (Vacuum priming for example). The pressure level can be accepted externally or it can be an internally fixed setting. The symbol can then be reused in any number of schematics.

Once a device is tested and accepted, the whole schematic can be changed to a subroutine record, mapped to the same symbol and the database regenerated, this will then upgrade all the

instances of the given device, with a minimum amount of change. A single subroutine record will provide increased performance over the use of multiple database records.

The third or application (system) level, captures all the high level functionality required by the users. This is the level that describes the controlled system, without any details regarding the control system. This level can be developed before the actual control system is built and it is at this level most of the debugging will take place. The application can be tested, and is perfectly suited for software simulations.

Database Integration

The final database is generated from the hardware and logic databases. Usually there is only one hardware database corresponding to the crate setup and multiple logic databases. Pointers maintain the links between two segments of a completed application. There are two types of links that are necessary to cross the boundary between the hardware and logic databases. They are data transfer (data links) and execution links (forward links). These links represent the mapping of logical control channels to physical I/O channels. The data links are only required on the receiving end, and forward links are necessary only on the transmitting end. Only data links are needed, if reverse order execution is used, .

To keep the documentation complete, the links are marked at both ends, receiver and transmitter. This marking will also aid in the debugging process. Off-page links are implemented using a variation of the <hardware input> and <hardware output> symbols from the EPICS symbol library. Those symbols have the role of generating the name of an external record on a data field or on a link field.

SCHEMATIC CONVENTIONS

All record changes must be made using CapFast. DCT is only used to generate the database from the short form report.

Most of the symbols are passive and provide the functional fields SLNK and FLNK. The data flow is from left to right (input on the left, output on the right). The processing flows, SLNK and FLNK, are on the top of the symbol, left and right respectively. Schematics that represent devices, ion gauges, ion pumps, valves for example, the hardware inputs and outputs are on the bottom, input on the left and outputs on the right. To enhance the transparent nature of the schematics, symbols should be passive with the preprocessing option enabled and they should also provide functional FLNK and SLNK fields. Because of the assumption that this devices are independent of the control hardware, but can receive and send data, some restrictions have to be applied. For example, the analog inputs and outputs are mapped as real numbers, 0-65535, full scale. If an ion gauge controller reports the pressure using voltage, logarithmic mapping, in the 0-10V range, then the ion gauge controller application will consider the 0-10V mapped linearly to 0-65535, and will apply the appropriate conversions to the pressure. The binary in and out symbols, have two values, TRUE (non-zero, active, or unsafe) or FALSE (zero, off, or safe), complying with the active high convention.

Similarly, all of the outputs of measurement devices are converted to SI units. For example, if the ion gauge output from the ion gauge controller is configured to read in Pa. Instructions have to be included in the schematics for proper configuration of the interface equipment (VME boards). For proper object oriented development, all the objects in the field must be modeled as a reusable schematic. Because of the multiple user aspect, different devices that fulfill the same function (i.e., different ion gauge controllers) must have the same design interface. This characteristic provides for the replacement of different types of controlled devices, without having major changes in the software.

For example, there are four main symbols that are used in more than 75% of all the process control applications. They are: analog in, analog out, digital in and digital out. In each class, a class is defined by a symbol, there are many boards which may be used.

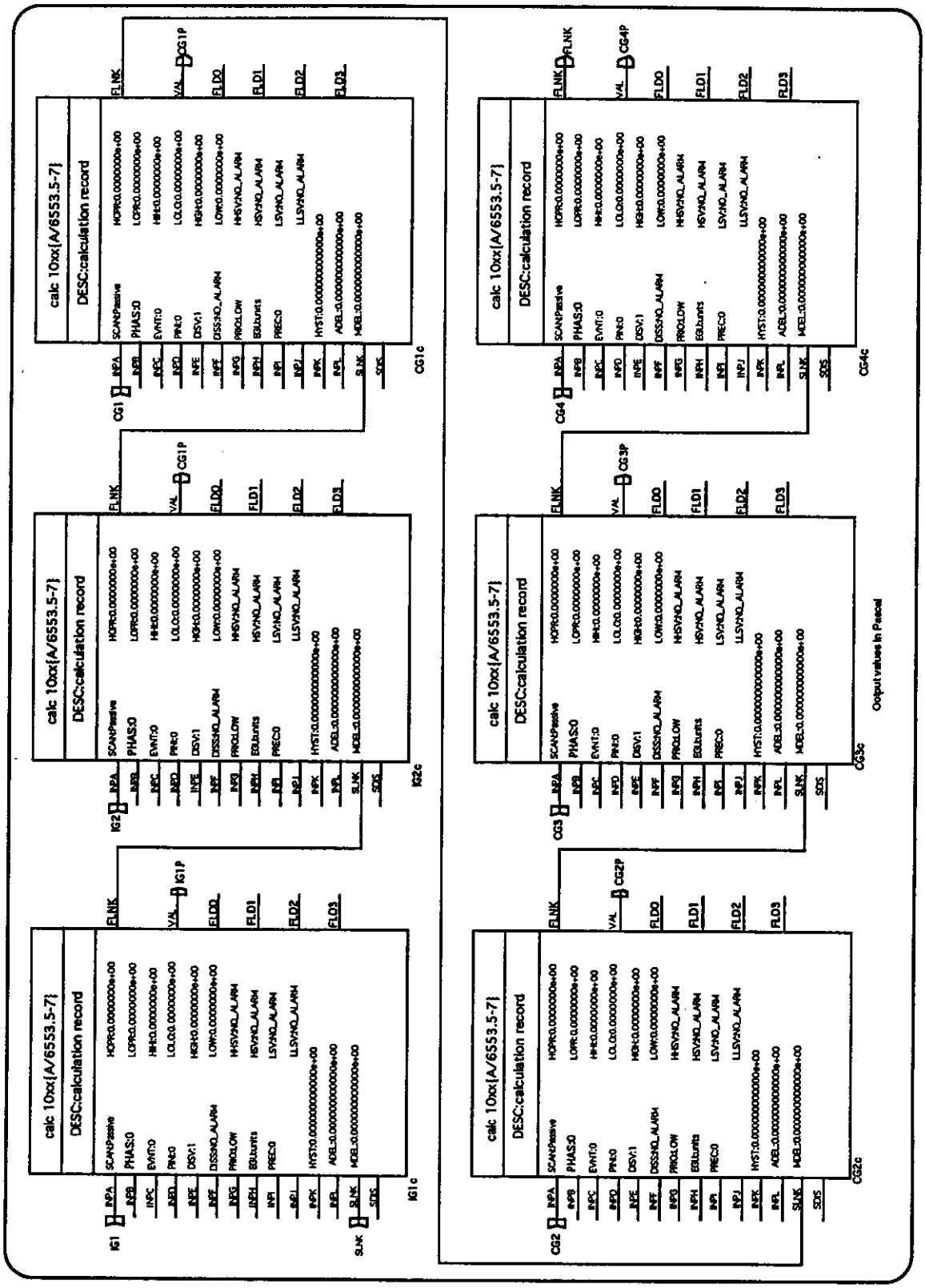

Figure 5 GP-307 Control Schematic

LIMITATIONS

Because a complete database has to be (re)-generated from any top level schematic, off-page connectors are not permitted. This forces a hierarchical development of the applications. This limitation may be circumvented as in the condition of interlocking applications by using pointers which pass the entered value to the specified record field. In applications where a symbol requires different properties from the default, the modified symbol must be graphically different from the original.

If a symbol is developed for a card and utilizes all of the cards functions and channels, and the same type card is used for some other application that only requires 50% of the functions, the symbol must be copied as a new symbol then the extra functions may be disabled on the new symbol. It is important to note that the disabling of unused functions on any board reduces the database overhead.

REFERENCES

CapFast Electronic Circuit Design CAE User's Guide, by Phase Three Logic

EPICS User Manual, developed by The Controls and Automation Group (AT-8), Accelerator Technology Division, Los Alamos National Laboratory, with The Controls and Computing Group, Accelerator Systems Division, Argonne National Laboratory.

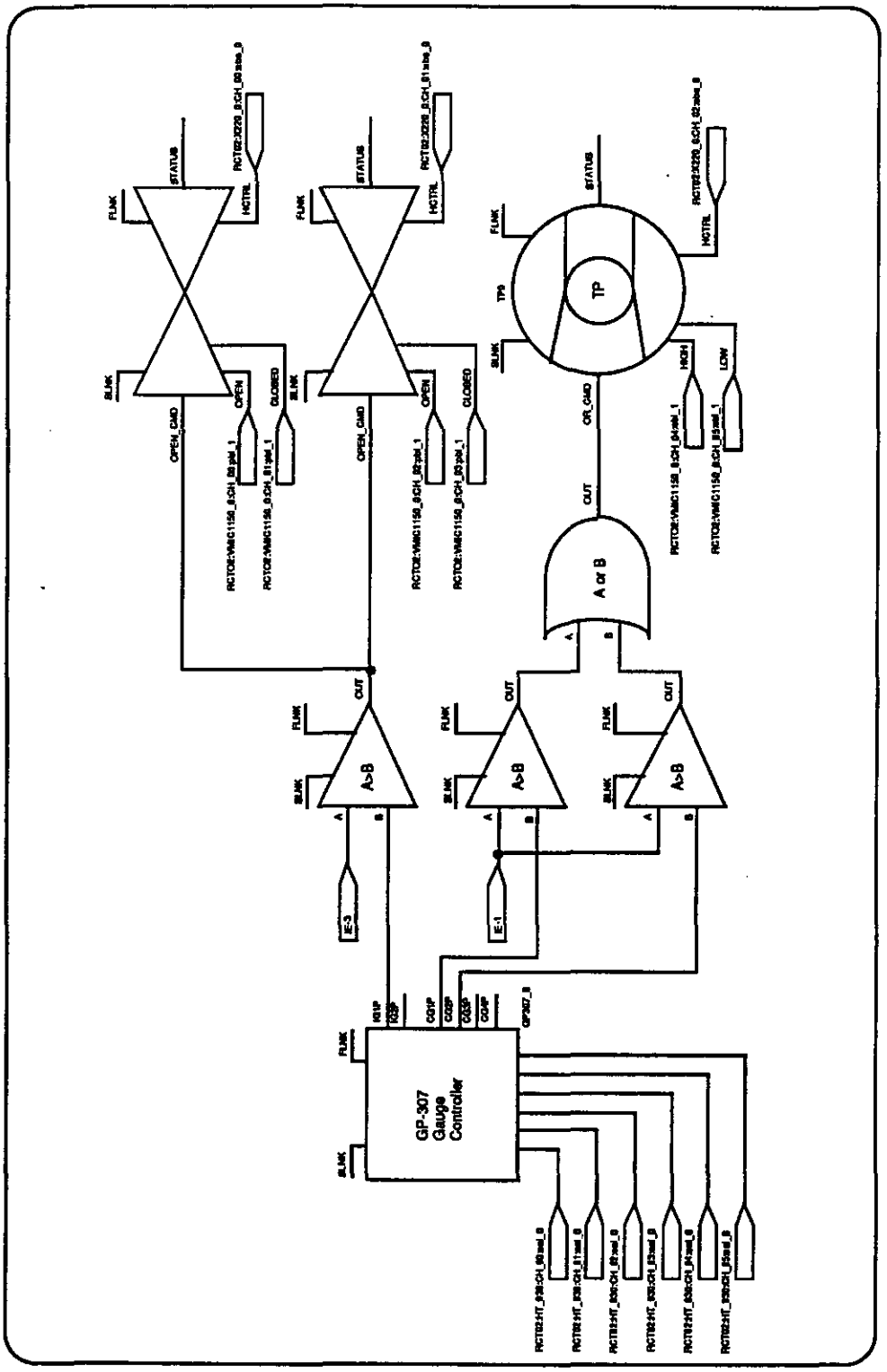

Figure 7 Vacuum Controls Example