LIVE CAPFAST SCHEMATICS IN THE ISAC CONTROL SYSTEM

R. Nussbaumer, TRIUMF, Vancouver, Canada

Abstract

For the EPICS-based control system of the ISAC radioactive beam facility, the Capfast schematic editor is used to design EPICS IOC runtime databases. This graphical tool provides a view of the database with strong visual cues about the functional behavior of the database elements and their inter-relationships, modularity, and hierarchy. The EDM display manager tool is used for the Human-Machine-Interface, providing a graphical view of the accelerator state to machine operators. This paper describes a new tool Sch2Edl, which combines some of the functionality of Capfast and EDM. Sch2Edl creates a view of the runtime database in a format geared toward the system developer/tester/debugger. Sch2Edl is a perl script that translates a set of Capfast schematic files into a corresponding set of EDM screens. The visual representation of the runtime database on the EDM display is nearly identical to the static Capfast views and hierarchies, but incorporates the display of online runtime data. This allows software developers to examine and modify some aspects of a working runtime database in an environment that is rich in visual information.

INTRODUCTION

In the ISAC project, the Capfast[1] schematic editor, schedit, and its related symbol editor, symed, are employed by designers of EPICS runtime databases used by Input-Output Controllers (IOCs). Schematic drawings of symbols that represent the EPICS records or collections of EPICS records and their connections are mechanically translated into runtime database files that EPICS IOCs use to produce the functionality desired by the designer. Schedit is quite rich in interactive and graphical capabilities as a design tool, but lacks any direct connection with the devices or control mechanisms for which a design is targeted. Only after a schematic has been translated to the EPICS '.db' format database, and subsequently loaded into an EPICS IOC can the exact functional behavior be observed. In the translation from the graphical form to the flat-file ASCII database, the graphical and interactive perspective is removed. In order to perform troubleshooting, testing, and diagnostics, the developer is left only with relatively primitive diagnostic tools provided by the EPICS iocShell and other command-line based tools, or with end-user targeted operator interface (OPI) tools, such as the display-list viewer/editor, EDM[2]. While the text-based tools are capable of providing detailed information, they are not well suited to display of database record interaction, are somewhat cumbersome, and do not provide a continuous real-time update of data. Synoptic OPI displays often intentionally conceal information which is not required for operation of the machine, and are optimized for use by machine operators. OPI tools can of course be used to produce diagnostic screens for system developers, but this requires a considerable investment in time. Therefore, the tool sch2edl was developed to help system developers to use the combined strengths of two graphical instruments, Capfast & EDM.

FILES, FORMATS AND FEASIBILITY

The Capfast schematic editor stores its drawing files in a proprietary, undocumented format. The files contain ASCII text that is organized into records delimited by 'newlines', and contains human readable text and numeric content. Through a process of inspection, experimentation, and iterative testing, the meaning of the file contents was discerned. The same reverse-engineering process revealed the nature of the related symbol files, as well as the relationships between the symbol and schematic files. The graphical content of a Capfast drawing consists of lists of primitive line segments representing wires, references to other embedded symbols, & textual content. These elements are mapped into a Cartesian coordinate system defined in integer units.

The EDM display manager tool also contains lists of primitive drawing elements, as well as being mapped onto a coordinate system which describes the on-screen location of those elements. The file format used by EDM is formated as human-readable text, and is fully documented.

There also exists the series of EPICS '.dbd' files, which describe the nature of EPICS records, the fundamental elements of EPICS control system databases. These, too, are stored in human-readable text format, and are easily parsed using the perl programming language.

Finally, there exists a Perl library module, 'EdlBuild.pm'[3], which facilitates the generation of EDM '.edl' format files programatically. The Perl programming language is well suited to the parsing of text-oriented data files and, as such, is a natural choice as a translator of Capfast to EDM files. For most Capfast elements there is a closely corresponding EDM widget or drawing primitive. The translation between differing coordinate systems and display element can be easily accomplished programmatically using perl.

A simple configuration file is read by sch2edl to set configurable items, and is formatted using familiar conventions and style.

LOOK AND FEEL

Both the EDM display manager and the Capfast drawing tool provide an interactive environment for the

user. The nature of the interactions possible and the manner of invoking them differ considerably. Capfast is by nature a 'vector' oriented tool, capable of arbitrary scaling, and allows for panning and scaling over wide ranges of the coordinate system, while EDM displays are fixed in dimension & scaling, and are functionally restricted to the logical size of the hardware screen. Capfast embodies the concept of hierarchical navigation by replacing the present view with another view, and then remembering the hierarchy trail for retracing the steps taken through a hierarchy. EDM, in contrast, provides linkages to other views primarily through *Related Display* call-ups, which simply load a specified display-list file and expand any macros that are passed with the file spec.

Despite these and other differences, a method was developed to map the various capabilities of Capfast onto a set of EDM displays. The look and feel of the resulting displays closely resemble displays that are used in the conventional control system OPI, which was achieved by using the site-specific default configuration of the EdlBuild API. The functional aspects of the EDM displays are intended to provide a dense, graphical, on-line, and somewhat modifiable perspective on the running control system functionality.

USING THE TOOL

The tool is a single perl script, called *sch2edl*, reflecting the translation of the native file formats as named by their parent programs. When run, the script is given the name of one or more Capfast schematic files, along with some optional command-line arguments, and a translation of the specified schematic is performed. In addition, all symbol files reference within the specified Capfast schematic are recursively translated.

USING THE TOOL OUTPUT

The output of the translation tool is a set of EDM screens which contain all of the hierarchical content of the top level schematic. At the top of the EDM screen hierarchy is a menu of call-ups, one for each toplevel schematic given as a command-line argument.

Any element in the schematic which evaluates to a runtime process variable is displayed using a *Text-Monitor* widget. *Related-display* call-ups are used in the EDM screens to provide linkages between the screens in ways that approximate the ability of Capfast to descend and ascend through hierarchically linked schematics.

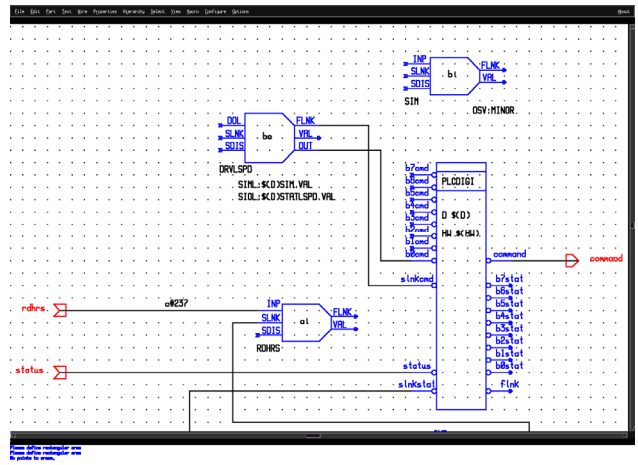

Figure 1: A Capfast drawing fragment containing EPICS primitive and composite symbols (colors modified for reproducibility).

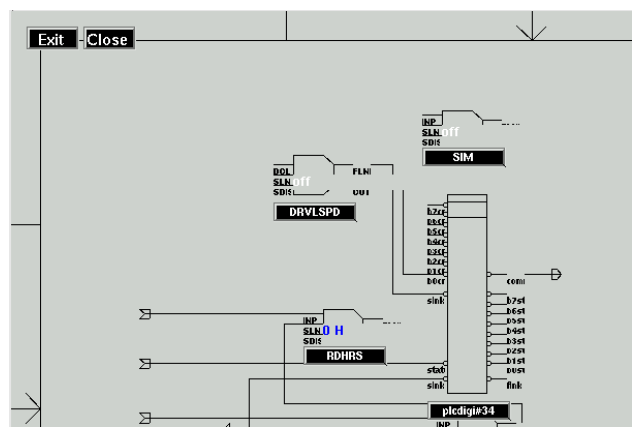

Figure 2: An EDM rendering of the Capfast drawing in Figure 1, above.

The lowest level of descent in the EDM screen set is the EPICS record level. At this level, a single EDM screen displays the contents of all fields in the instance of each record, using *Text Monitor* widgets. This display level is analogous to the 'Item Properties' dialog of Capfast. Since many standard EPICS record fields are capable of producing monitors, i.e. messages that asynchronously pass new data to connected clients, the record displays contain the actual runtime values contained in the respective IOC database. Furthermore, all fields which can be modified at runtime are available for user updates. Fields with enumerated values are displayed with a menu of appropriate values for selection, eliminating the need to remember such values as is necessary when using command-line tools such as 'dbpr'.

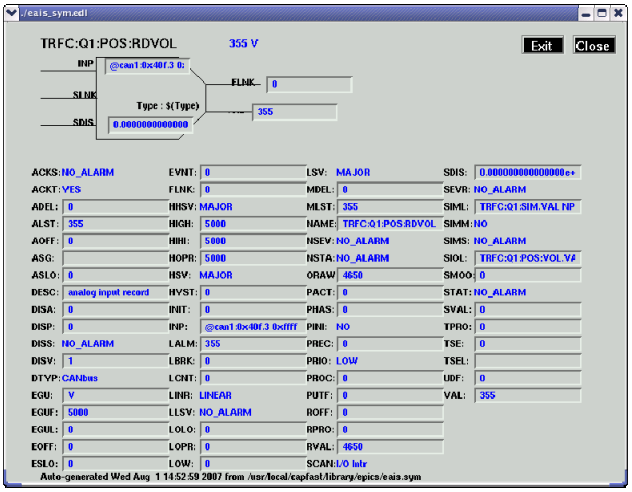

Figure 3: A Capfast analog input record symbol with all of its fields.

LIMITATIONS AND LOCAL CONVENTIONS

The author's intention in the design of the tool was to provide an environment that provides information as concisely and conveniently as possible, and to mimic the behavior of Capfast as closely as possible. Some of the methods which Capfast uses for navigating and browsing can be only roughly approximated. The ability of Capfast to provide nearly continuous levels of zooming and panning can only be represented in the EDM views by creating static displays at generalized zoom levels. The ability of EDM to pass information between screens is limited to the use of macros which are passed between screens. The present limit of 512 bytes as a total macro-list length imposes a restriction in some significant ways. While Capfast maintains an internal memory of traversal of drawing hierarchy, permitting the user to easily ascend back through nested hierarchies, there is no such capability within the EDM representation.

For efficiency in performing coordinate conversions, all Capfast schematics (not symbols) are expected to be bounded by a drawing border. This is used to determine the size of the resulting screen, and relieves the translator from computing a bounding box for each and every Capfast object to determine the overall display boundaries.

In Capfast symbols, generic devices are instantiated in part by assigning values to macros that are defined at the top level symbol representing the device. For instance, a generic symbol representing a turbo pump device type is instantiated as a specific device by assigning a value to a macro defined in the turbo pump device symbol. Sch2edl makes use of local conventions in the use of these macro names, but allows for some flexibility by specifying device-level macro names in the configuration file.

The process of building a full set of screens from a large database, especially where there are deeply nested layers of hierarchy can be slow, and it is not always desirable to perform on a repetitive basis. An improvement to the tool which uses the standard 'make' utility may reduce this problem.

CONCLUSION

The ability to view and interact with a live EPICS database in a way that uses the familiar graphical presentation of Capfast significantly enhances the testing and diagnostics capabilities of EPICS system developers. Of particular value is the capability to make small incremental changes on-line and to immediately see the effects. This is especially useful during the database design phase, where iterations of the 'Draw-Build-Download-Test' procedure can be reduced. When the EDM display and EPICS database are configured to highlight anomalous conditions such as alarm states and uninitialized records, the tool provides a convenient way of browsing databases, visually scanning for unexpected behaviors. Record relationships and interaction is immediately visible, and leads to an enhanced understanding of the database operation.

REFERENCES

[1] Phase 3 Logic, http://www.phase3.com/
[2] J. Sinclair, http://ics-web1.sns.ornl.gov/edm/
[3] R. Keitel, 'EdlBuild – Display Generation for the EPICS EDM Display Manager', ICALEPCS05, Geneva.