Using ezcalDL to connect to EPICS Channel Access from SHADOWVUI for Dynamic X-ray Tracing

Alan Duffy
Canadian Light Source
Introduction

www.lightsource.ca

CLS Layout
Linac - LTB1
BR1 - BTS1
SR1

Shielding

Operating
Construction
Available

www.lightsource.ca
Control system is based on EPICS, RTEMS using GNU GCC, Borland C++ Builder and MKS.

PLCs are based on either MODICON Momentum or Siemens Simatic lines.

VME Equipment is from CAEN, GE-Fanuc/VMIC, Hytec, ICS, OMS, Sensory, and WEINER.

PC Equipment from Dell, Kontrol/PEP Modular, and Tri-M.

Enclosures from LCH Resource, Industrial Computers, and Hammond.

Process Instrumentation: Alltemp, Greystone, Newport, Temco Controls, Wika.

Routine Supplies Gescan
Outline

- Motivation
- Software perquisites (what you need)
- Software description (what it does)
- Simulation model of real-life beamline
- EPICS and ezcaIDL (connections)
- ezcaSHADOWVUI (dynamic ray tracing)
Ray-tracing (in Shadow or ShadowVUI) is typically used during the design stage to optimize beamline performance.

- Model is static and requires user to input positions to match beamline configuration.
- After the beamline is built x-ray tracing is used less frequently.
- Automating process makes life easier.
SHADOW (Fortran and C library of subroutines)
- Ray tracing engine developed at Nanotech Wisconsin (University of Wisconsin)
- Used to study flashlights to x-ray telescopes and microscopes

XOP + SHADOWVUI (written in IDL)
- Visual User Interface for SHADOW

EPICS with extensions: ezca, ezcaIDL
- Provides Channel Access (CA) to process variables
Main program and utilities
- I/O session driven to define system

SHADOW Structure
- Data files (usually binary)
- Parameter files (e.g. START.XX in NAMELIST format)
  - \( X_{\text{ROT}} = -0.500000000 \)
  - \( T_{\text{INCIDENCE}} = 75.0000000 \)
  - \( T_{\text{SOURCE}} = 10.0000000 \)
  - \( T_{\text{IMAGE}} = 0.000000000 \)
- Analysis files (varied)
XOP + SHADOWVUI

Run SHADOW/source

```
cmd.exe /c shadowvui.bat
Calculation completed.
```

Run SHADOW/trace

```
cmd.exe /c shadowvui.bat
Calculation completed.
```
- Interfaces with the SHADOW kernel

- Three main steps:
  - Define source and Run SHADOW/source
  - Define optical system and Run SHADOW/trace
  - Visualize results

- Macros (written in IDL) may be used to automate some tasks (i.e. changing energy, moving mirrors, loops, etc.)
SHADOWVUI Simulation Model

- **SHADOW variables**

<table>
<thead>
<tr>
<th>OFFX</th>
<th>X_ROT</th>
</tr>
</thead>
<tbody>
<tr>
<td>OFFY</td>
<td>Y_ROT</td>
</tr>
<tr>
<td>OFFZ</td>
<td>Z_ROT</td>
</tr>
</tbody>
</table>

The orientation of each element is relative to the previous element, but mirror displacements do not move subsequent components.
In this model
- OFFY corresponds to T2
- OFFZ corresponds to T1

Time to plug and play with EPICS

This is hard to read
EPICS and ezcaIDL

- **EPICS**
  - real-time control system for beamlines etc.
  - process variables indicate positions of optics

- **ezcaIDL**
  - allows access to a set of simplified IDL interface commands to connect to Channel Access

```plaintext
Status = caGet(pvname, value, /string, max=max)
Status = caSetMonitor(pvname)
Status = caWidgetSetMonitor(name, widget_id, time=time)
```
- Initializes ezcaIDL
  
  ```idl
  caInit
  caSetTimeout, 0.001
  caPendIO, time=0.01, list_time=3.
  caPendEvent, time=0.000001
  add_caPendEvent, timer=5.0
  ```

- Accesses SHADOW variables via SHADOWVUI
- Requires user input that defines relationship between model variables and beamline PVs in an IDL structure
### PV_INFO Structure (container)

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pv</td>
<td>string</td>
<td>EPICS process variable string</td>
</tr>
<tr>
<td>desc</td>
<td>string</td>
<td>Text to describe process variable</td>
</tr>
<tr>
<td>pv_min</td>
<td>float</td>
<td>Lower limit</td>
</tr>
<tr>
<td>pv_max</td>
<td>float</td>
<td>Upper limit</td>
</tr>
<tr>
<td>oe_num</td>
<td>int</td>
<td>Optical element number (zero otherwise)</td>
</tr>
<tr>
<td>src_num</td>
<td>int</td>
<td>Screen number (zero otherwise)</td>
</tr>
<tr>
<td>pv_2vui</td>
<td>string</td>
<td>Equation(s) to convert value of PV(s) to SHADOWVUI variable</td>
</tr>
<tr>
<td>vui_2pv</td>
<td>string</td>
<td>To convert value of SHADOWVUI variables(s) to PV value</td>
</tr>
<tr>
<td>vui_val</td>
<td>float</td>
<td>Stores SHADOWVUI variable value</td>
</tr>
</tbody>
</table>

- vui_2pv string is executed on widget start-up
- pv_2vui string is executed on PV events
SHADOWVUI variables and PVs

\[\text{OFFZ} = h \cos(\theta)\]

\[\text{OFFY} = h \sin(\theta)\]

\[\text{pv}_2\text{vui} = \left(\text{(*ptrOE1).OFFY} = \text{data\_struct.h.val} \times \sin(\text{data\_struct.theta.val})\right)\]
& \left(\text{(*ptrOE1).OFFZ} = \text{data\_struct.h.val} \times \cos(\text{data\_struct.theta.val})\right)\]

\[\text{vui}_2\text{pv} = \text{sqrt(((*ptrOE1).OFFY)^2 + ((*ptrOE1).OFFZ)^2)}\]
IDL> resshadowvui, data_struct
SHADOW and XOP + SHADOWVUI
  - Provide ray-tracing engine and user interface
EPICS extensions ezcaIDL/EZCA
  - allow IDL programs to access PVs
ezcaSHADOWVUI
  - takes SHADOWVUI model and user defined relationships between PVs and model parameters
  - live positions may be used for dynamic ray tracing
Research described in this paper was performed at the **Canadian Light Source**, which is supported by:

- Natural Sciences and Engineering Research Council of Canada
- National Research Council Canada
- Canadian Institutes of Health Research
- Province of Saskatchewan
- Western Economic Diversification Canada, and
- University of Saskatchewan.
Appendix - Prerequisites

- EPICS installed with extensions directory setup
  - `/opt/epics/base`
    - `baseR3.14.9.tar.gz`
  - `/opt/epics/extensions`
    - `extensionsTop_20070703.tar.gz`
    - `extensionsConfigure_20070703.tar.gz`
  - `/opt/epics/extensions/src/ (ezca, ezcaIDL, EzcaScan)`
    - `ezca_20070625.tar.gz`
    - `ezcaIDL_20070625.tar.gz`
    - `EzcaScan_20090319.tar.gz`
Install procedure (libezcaIDL.so)

- cd /opt/epics/extensions && make
  - ln -s /usr/local/bin/g++ /usr/bin
  - ln -s libncurses.so libcurses.so
  - yum install mingw32-readline
  - ln -s /usr/i686-pc-mingw32/sys-root/mingw/include/readline /opt/epics/base/readline

- Set environment variable EZCA_IDL_SHARE
  - /opt/epics/extensions/lib/linux-x86_64/libezcaIDL.so

- /etc/ld.so.conf.d/
  - create ezcaIDL.conf with path to libezcaIDL.so