Digital Gammasphere
Software Development at ANL

Timothy Madden
Driver Development

• asynDriver used for hardware drivers.
• asyn eliminates need for custom driver code and custom EPICS DTYP’s.
• All PVs represented as “parameters” in source code, similar to fields in an object.
• C++ based code.
• Simplifies development and support.
• Created Python scripts to convert Gretina EPICS databases to be compatible with asyn.
• Allows easy exposure of raw VME registers as PVs.
Changes to Sender/Receiver

• Changed Send/Receive to accommodate new Digitizer header structure.
• Refactored code to put header-related operations into functions rather than inline code.
• Added switches and PVs to control timing and CPU usage of sender, sorter.
• Various bug fixes, esp. reading digitizer FIFO into “throw-away” buffer on low memory condition, rather than crashing.
• Added PVs to allow sending a random sampling of data, and setting percent of data to send, for UDP monitor.
Changes to GUI

• Migrate from EDM to Control System Studio.
  – See http://sourceforge.net/apps/trac/cs-studio/wiki/BOY

• Created python scripts to convert EDM screens to CSS screens.
Some screens

Global PVs, Edit Mode
Screens
Debug Screens

Raw VME Register Space
Reboot Button

Main Debug Screen
Global Reboot Button

Raw Board Registers
Auto-generated screen.
Auto-generation of Screens and EPICS Databases

• Register specification of VME boards are documented in a spreadsheet.

• Created Python scripts to
  – Read spreadsheet.
  – Generate C++ code for digitizer and trigger driver.
  – Generate EPICS databases
  – Generate CSS GUI screens.
  – Generate st.cmd files for all crates.

• Allows for support of
  – Multiple system configurations such as DFMA, DGS and Clover.
  – Configuration of system for multiple firmware versions and firmware upgrades.
Auto-generation of Screens and EPICS Databases cont.

• New PV names:
  – VME01:DIG4:xxx rather than Cry4xxx

• Python code to parse EPICS databases
  – EPICS PV class, for setting/getting PV fields.
  – Simplifies reading/writing, find/replace, auto-generation of databases.
  – EPICS database represented as Python list of epics_pv objects.
Auto-generation of Screens and EPICS Databases *cont.*

- Python object-oriented code for GUI generation.
  - CSS screen appears as cssScreen object.
  - Widgets appear as cssWidget object.
  - Read/Write, Search/Replace.
  - Allows auto-generation of screens based on EPICS database.
  - Allows the generation of EPICS database from a screen.
  - Can read EDM screens.
VME Address Space Exposed as PVs and Flashing FPGA via EPICS

• Created driver called asynDebugDriver() to represent raw VME address space in Channel Access.

• EPICS PVs represent
  – Board Slot
  – Board Address (offset)
  – Data to read/write
  – PV to perform VME read/write.
  – PV to store FPGA Firmware as EPICS array.

• Allows for hardware development/debugging in-system.

• Allows for flashing FPGA over EPICS.
  – Post FPGA firmware as a large EPICS array.
  – PVs for erasing, writing, verifying FPGA Flash.
  – Linux Command Line to Flash FPGAs.