Proposal for EPICS Interface for IU HV Control System

Hovanes Egiyan

What is EPICS

- Experimental Physics and Industrial Control System.

- Free open source software with large number of users.

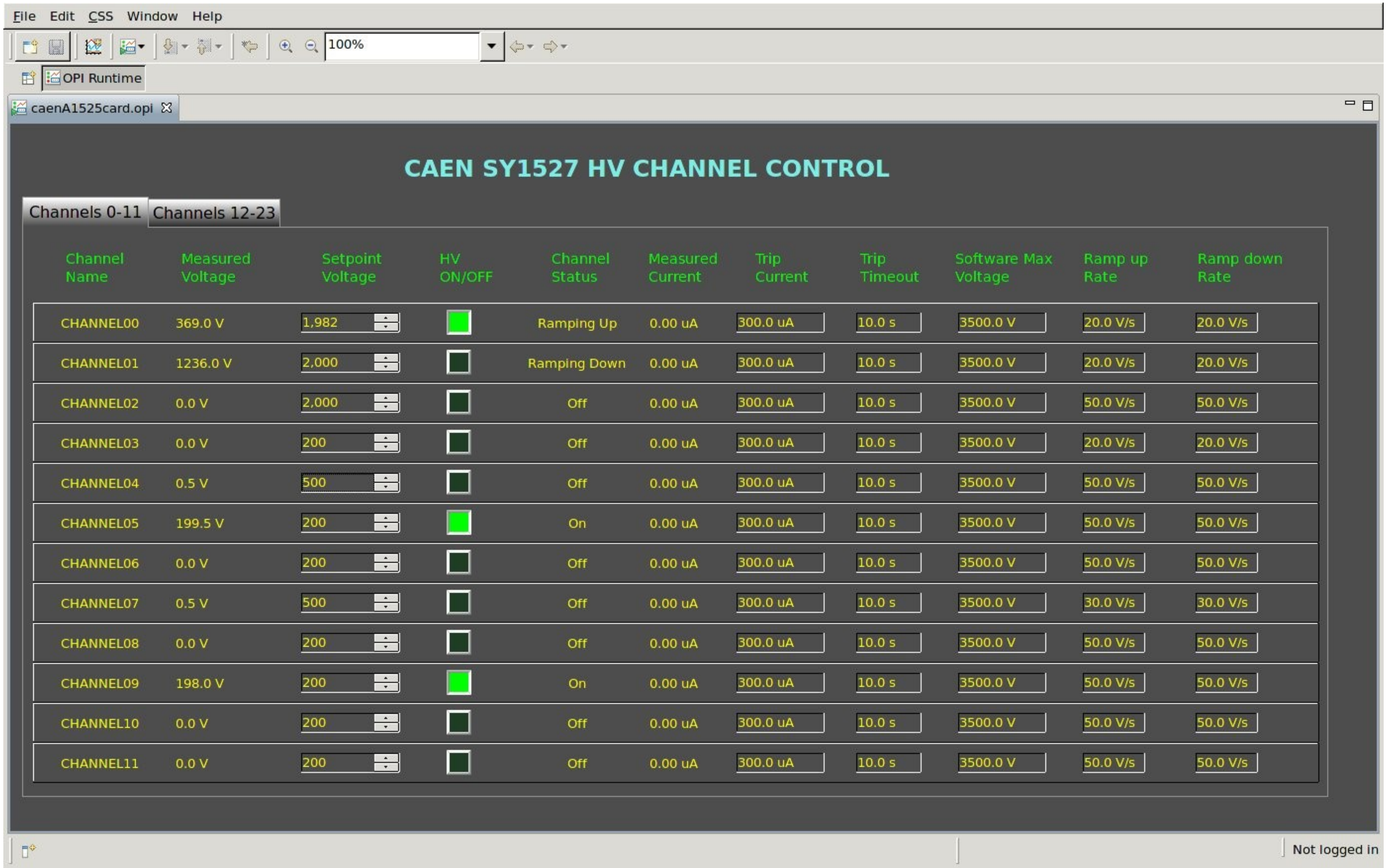
- Utilizes Server/Client architecture:
  - Input/Ouput Controller (IOC) serving variables over Ethernet using Channel Access protocol.
  - Clients on different hosts communicating with the IOC displaying, modifying, archiving values.

- EPICS application use Process Variables (PVs) to control and monitor parameters.
  - PVs can be of different types, called record types: ***ao, ai, bo*** etc.
  - Each PV have different fields: ***pvname.VAL***, ***pvname.SCAN*** etc. The set of the available fields depend on the type of the record the PV is declared.
  - Each PV is connected to a hardware device via "driver" written for a particular record type and that specific hardware.
  - PVs get processed and when they are processed specific action is taken depending on the record type and the hardware type (represented by the Device Support/Driver Support layer).

- Application are written in terms of EPICS record. The Device Support/Driver Support is hidden from the application.

- Provides various tools, like display management, alarm handling, PV archiving

CAEN HV EPICS GUI Example



CAEN SY1527 HV CHANNEL CONTROL

Channel Name	Measured Voltage	Setpoint Voltage	HV ON/OFF	Channel Status	Measured Current	Trip Current	Trip Timeout	Software Max Voltage	Ramp up Rate	Ramp down Rate
CHANNEL00	369.0 V	1,982	🟩	Ramping Up	0.00 uA	300.0 uA	10.0 s	3500.0 V	20.0 V/s	20.0 V/s
CHANNEL01	1236.0 V	2,000	⬛	Ramping Down	0.00 uA	300.0 uA	10.0 s	3500.0 V	20.0 V/s	20.0 V/s
CHANNEL02	0.0 V	2,000	⬛	Off	0.00 uA	300.0 uA	10.0 s	3500.0 V	50.0 V/s	50.0 V/s
CHANNEL03	0.0 V	200	⬛	Off	0.00 uA	300.0 uA	10.0 s	3500.0 V	20.0 V/s	20.0 V/s
CHANNEL04	0.5 V	500	⬛	Off	0.00 uA	300.0 uA	10.0 s	3500.0 V	50.0 V/s	50.0 V/s
CHANNEL05	199.5 V	200	🟩	On	0.00 uA	300.0 uA	10.0 s	3500.0 V	50.0 V/s	50.0 V/s
CHANNEL06	0.0 V	200	⬛	Off	0.00 uA	300.0 uA	10.0 s	3500.0 V	50.0 V/s	50.0 V/s
CHANNEL07	0.5 V	500	⬛	Off	0.00 uA	300.0 uA	10.0 s	3500.0 V	30.0 V/s	30.0 V/s
CHANNEL08	0.0 V	200	⬛	Off	0.00 uA	300.0 uA	10.0 s	3500.0 V	50.0 V/s	50.0 V/s
CHANNEL09	198.0 V	200	🟩	On	0.00 uA	300.0 uA	10.0 s	3500.0 V	50.0 V/s	50.0 V/s
CHANNEL10	0.0 V	200	⬛	Off	0.00 uA	300.0 uA	10.0 s	3500.0 V	50.0 V/s	50.0 V/s
CHANNEL11	0.0 V	200	⬛	Off	0.00 uA	300.0 uA	10.0 s	3500.0 V	50.0 V/s	50.0 V/s

Suggested Principle

- Continuously scan the bus to maintained a synchronized buffer layer. Scan speed will depend on the bandwidth of the system.
- The requests to change hardware could be delayed, or they could be directly send when modified by a “user”.
- Parameters values that EPICS records see may be “old” if requests come within a scan period.

EPICS Support Structure

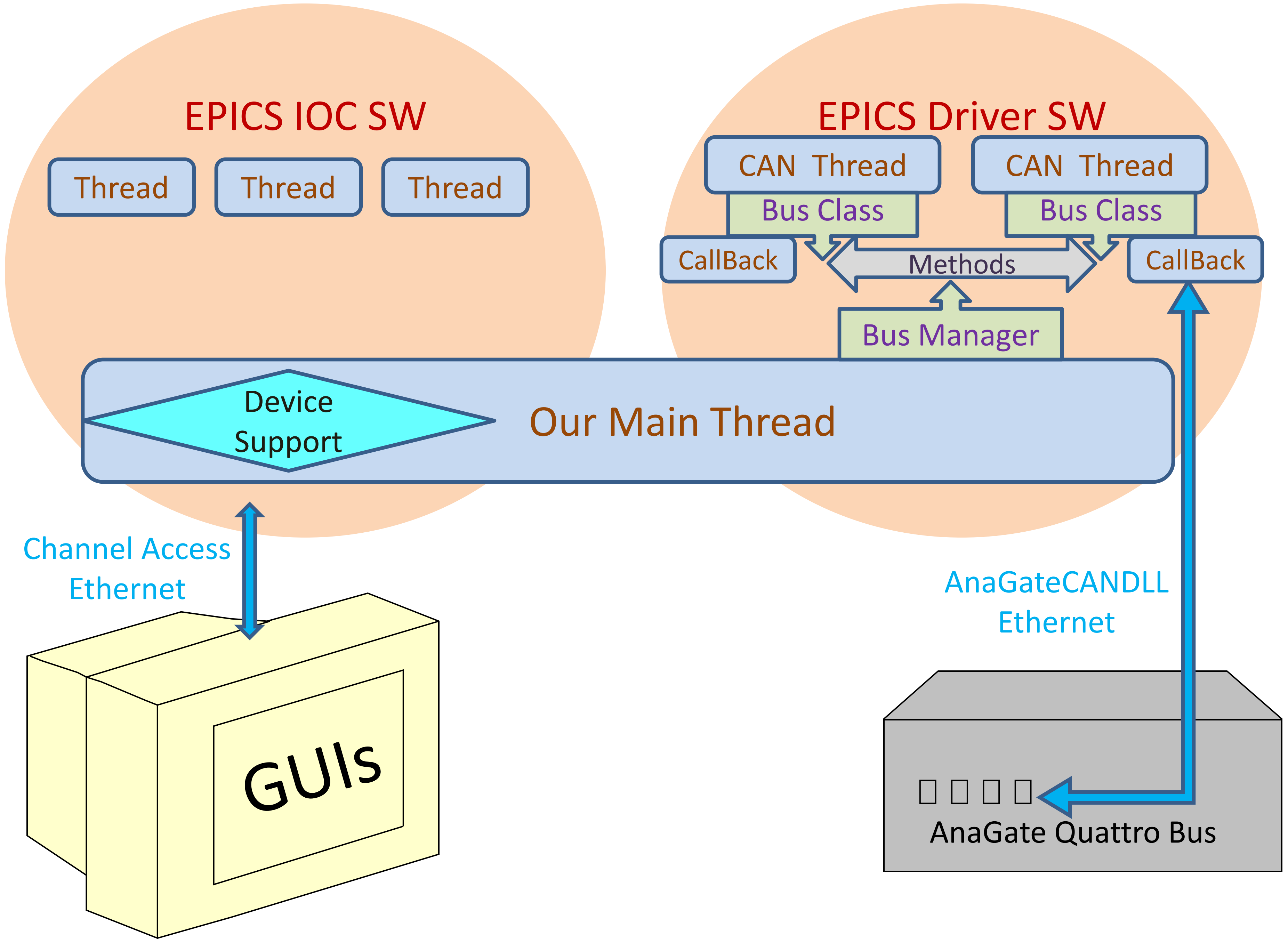
- Our EPICS support would consist of three pieces:

– Driver	-	Synchronizing software/hardware parameters
– Device Support	-	Connecting values to EPICS PVs
– EPICS application	-	Variables, GUIs, Alarms etc

- Driver part can be made independent of EPICS framework

- Device Support requests values from the buffer layer and assigns them to PVs.

- EPICS application is a set of EPICS records, GUIs, Alarm handlers etc, and can be dealt with together with CAEN HV system.


EPICS IOC SW
Thread
Thread
Thread
EPICS Driver SW
CAN Thread
CAN Thread
Bus Class
Bus Class
CallBack
Methods
CallBack
Bus Manager
Device Support
Our Main Thread
Channel Access
Ethernet
GUIs
AnaGateCANDLL
Ethernet
AnaGate Quattro Bus

Bus and Callback Threads

Bus thread periodically requests all IDs on the bus to know which boards are alive.

For each HV parameter on the board:

- *Bus* thread periodically:
    1. Requests the parameter value from the “live” boards on the *bus* using the corresponding to that parameter *command*.
    2. Scans through all [*bus,board,command*] triplet FIFOs for that *bus* and *command*, and synchronizes the buffer layer until responses from all boards are received and processed. On TIMEOUT generate an error or raise an alarm.
    3. Write to boards if the parameter value on the board needs an (or can skip this and write every time an EPICS record is processed/modified by “user”).

- Callback thread:
    - Keeps reading the messages from the bus and fills up the FIFOs for each [*bus,board,command*] triplet.

Questions

- How is the trip current setting control implemented?
- Is there a ramp rate control in the firmware?
- What is the alarm logic for HVs? What is the DAC Voltage to ADC Voltage correspondence?
- What needs to be done with the LED control? The control board seem to be using it already.
- How easy is it to add more HV parameters to the board , like status indication: On, Off, Tripped, Ramping up, Undervoltage etc?
- Can a message from the board FULLY describe what command it is responding to using “extra bytes”?
- Does the board abandon a planned response if a new request is sent to it?
- Can we have a full description of the response messages as well?
- The readback of ADCs and Statuses seems to take 350ms. Can it be improved by a factor of ~3?