SW for main protection functionality and management - IOCs

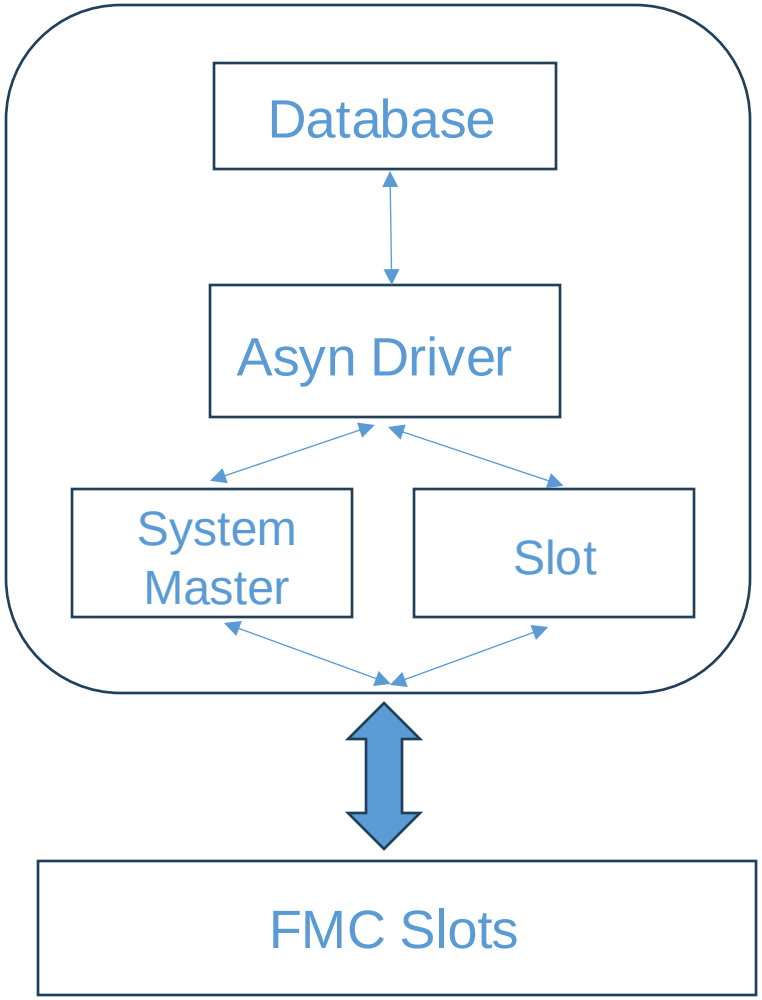
Kacper Kłys

Requirements

- Software Layer must provide interface to fully control system parameters and provide monitoring and archiving of all critical data
- Execution of protection function must be independent of non-critical software components
  - No parts of protection function are running on SW level
  - Software can be started and stopped without disturbance to system functions
- In case of critical software fault, appropriate statuses must be reported and inhibit signals raised

The management IOC architecture

CA/PVA



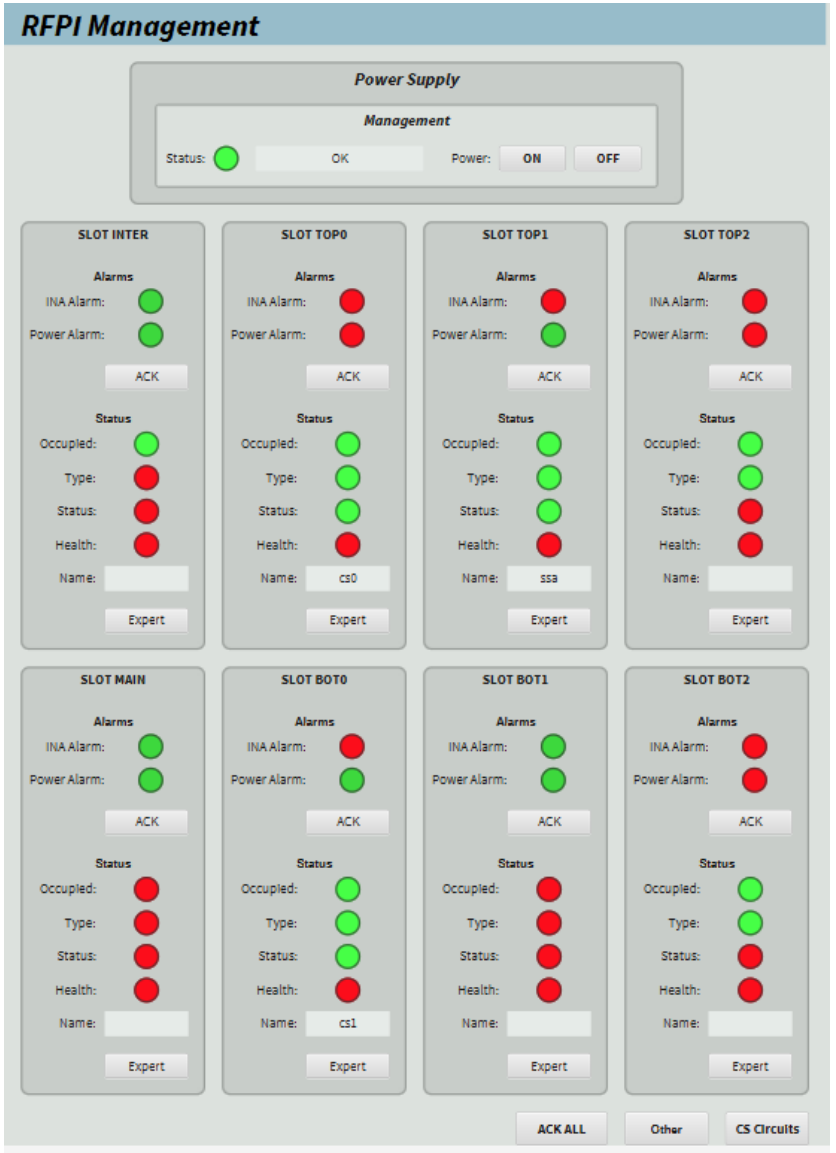
The management IOC architecture

- IOC contains *Asyn* driver that uses *systemMaster* and *Slot* to communicate with FMC slots and boards
- "Per slot approach" - PV database templates per slot
- Periodical data readout at regular, defined time intervals
- Multithread monitoring with callback approach
- No logic inside IOC, it only calls functions

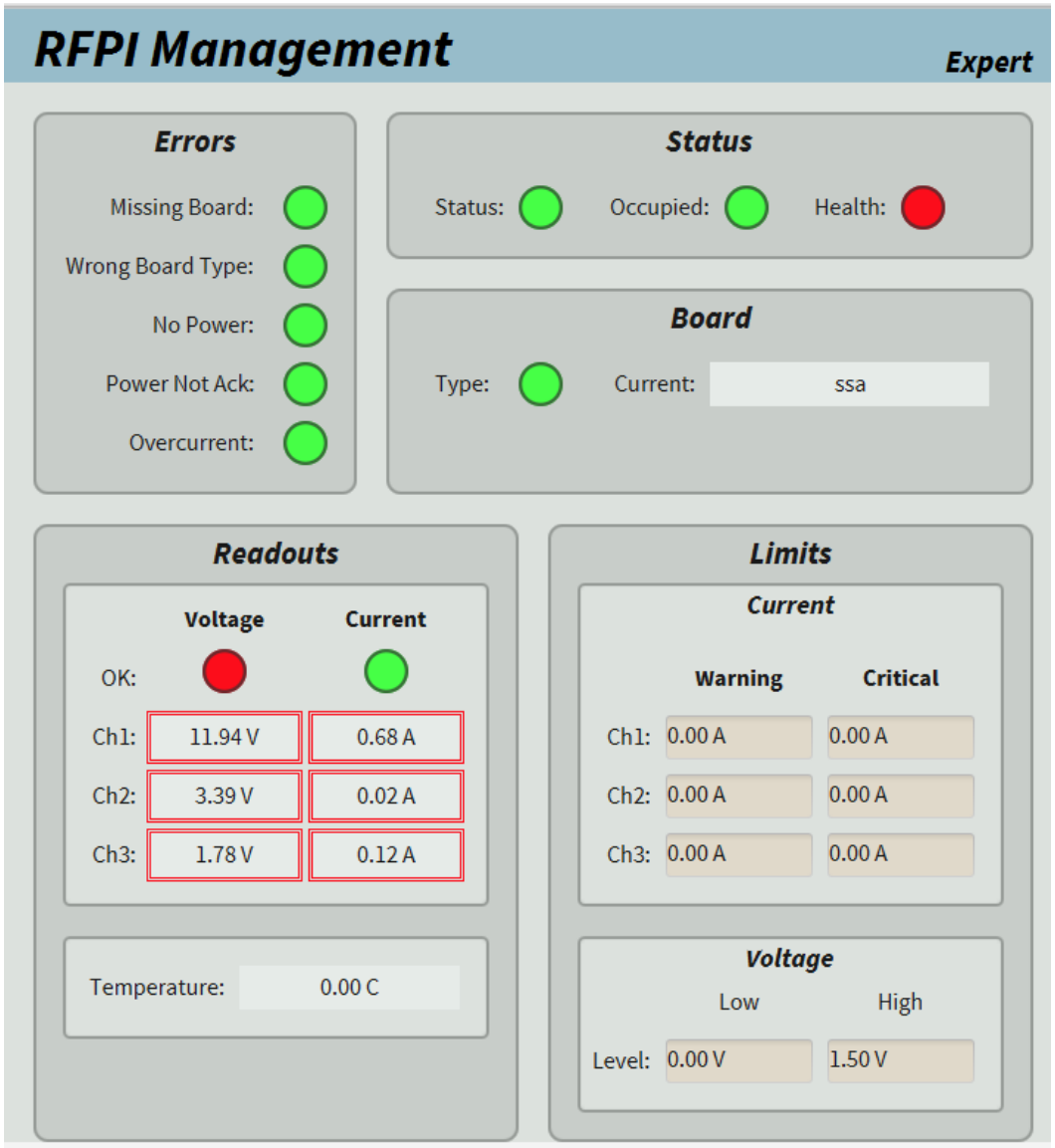
The management IOC OPIs – main window

Main window of the management IOC:

- Overview of each FMS slot – statues and feedback from monitoring.
- Each FMC slot can be reviewed in the export window (Expert button)
- Button to power cycle the whole system (with feedback information about the procedure success/failure



The management IOC OPIs – expert window

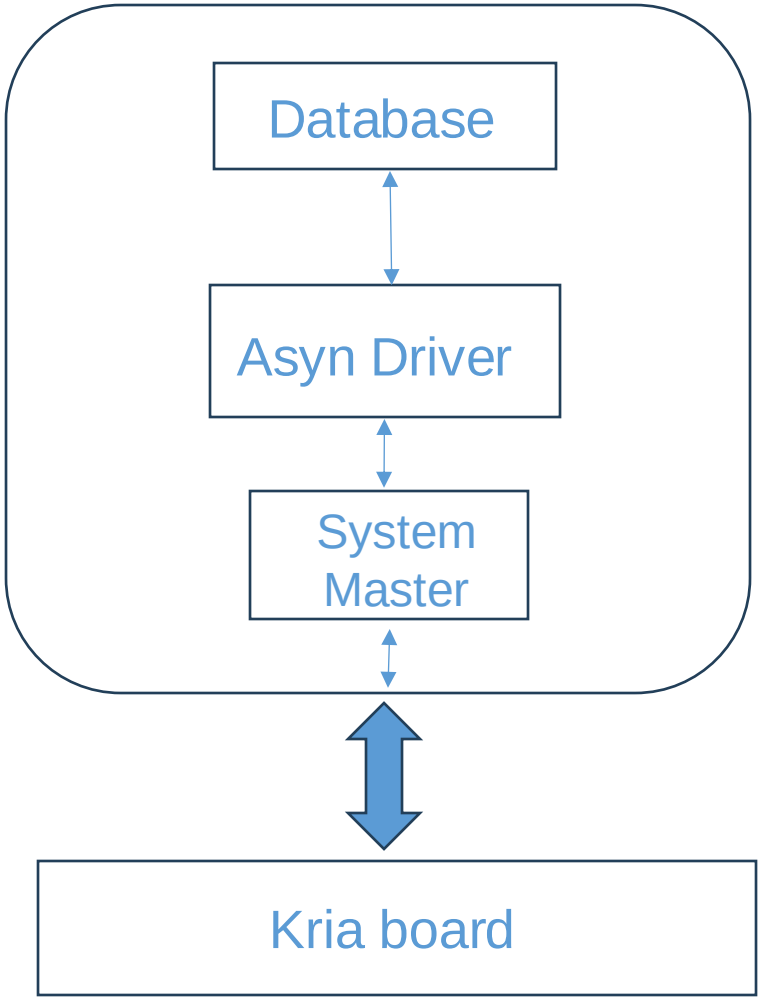


Expert window of the management IOC:

- Detailed information about slot status (occupied, powered, overcurrent etc.)
- Voltage, current and temperature readouts
- User can define voltage and current levels that will trigger alarms when exceeded

The main logic IOC architecture

CA/PVA



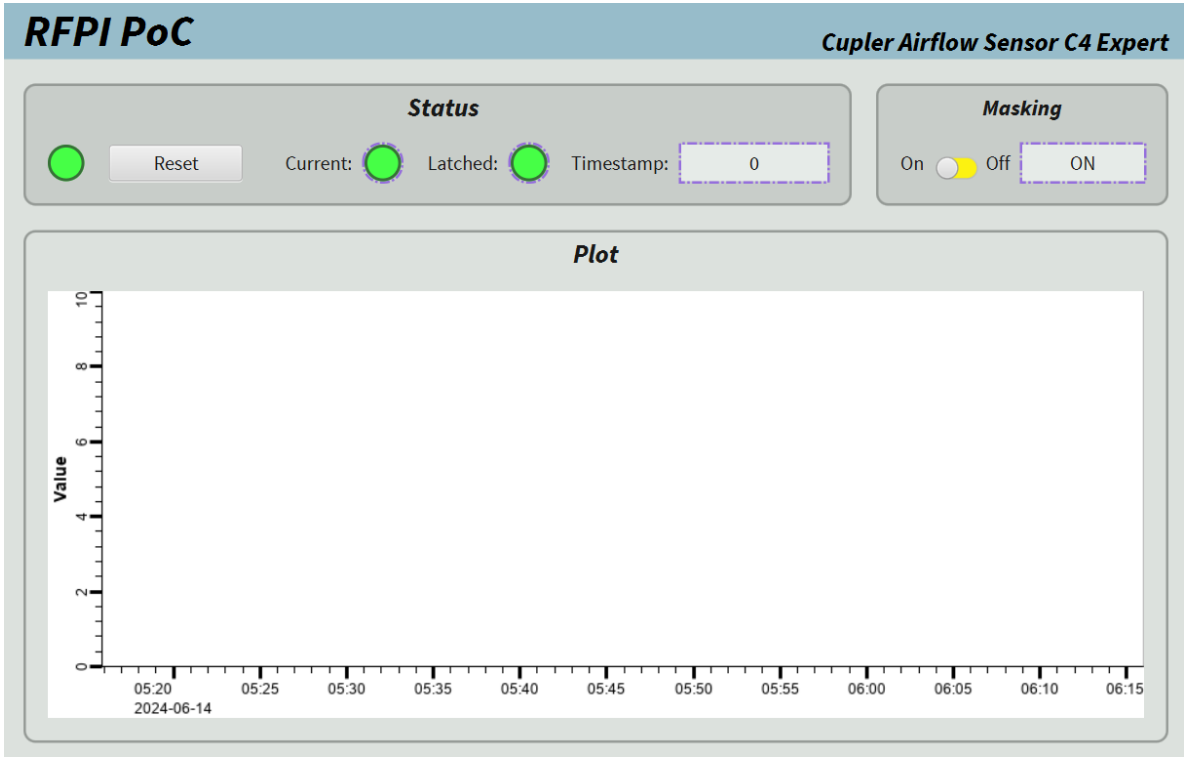
The main logic IOC architecture

- IOC contains asyn driver that uses *systemMaster* class that provides all needed functionalities.
- Periodical data readout at regular, defined time intervals.
- No logic inside IOC, it only calls functions.
- The only algorithm inside IOC is used to sort events by timestamps in case of system failure.

The main logic IOC OPIs – main window

- Statuses of all input and output signals
- Communication al arm status
- Buttons to open "expert" windows for all type of signals

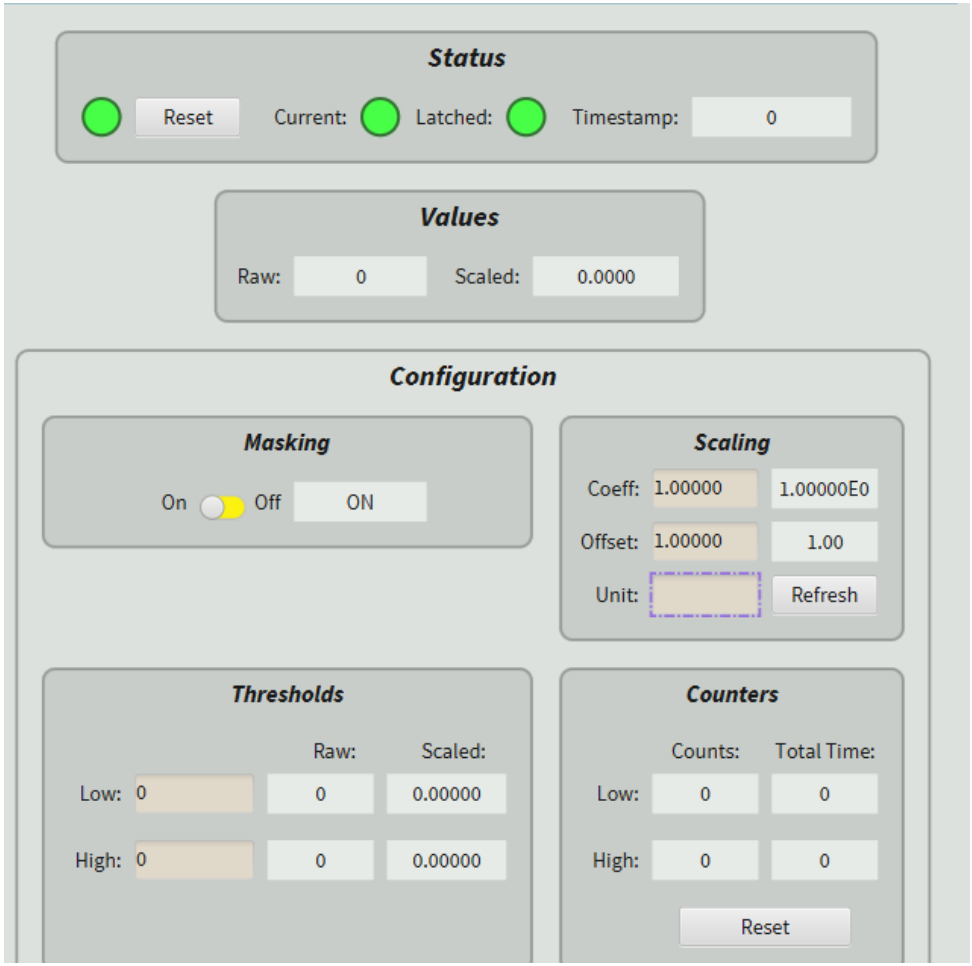
The main logic IOC OPIs – binary input window



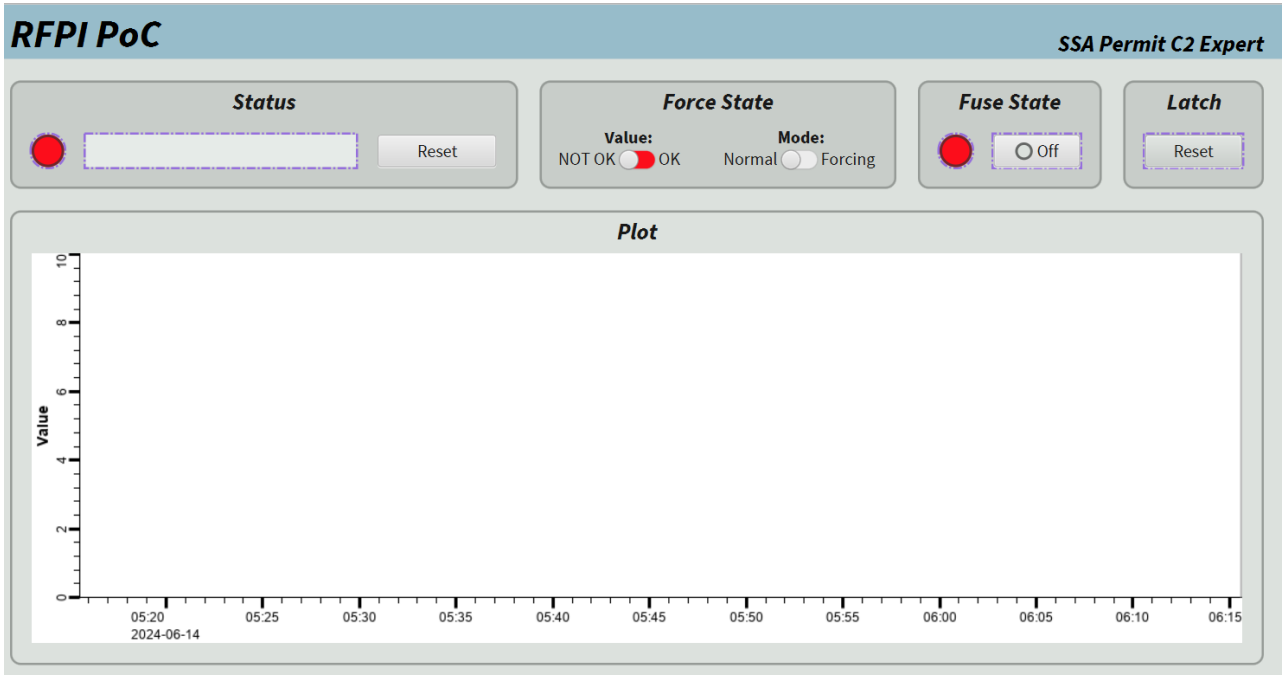
- LEDs with final status, current status and latched status with timestamp
- Masking mechanism
- Signal plot

The main logic IOC OPIs – analog input window

- Provides final, current and latched status LEDs
- Raw and scaled values from the conditioning boards
- Masking mechanism
- Configurable threshold limits
- Counter readouts

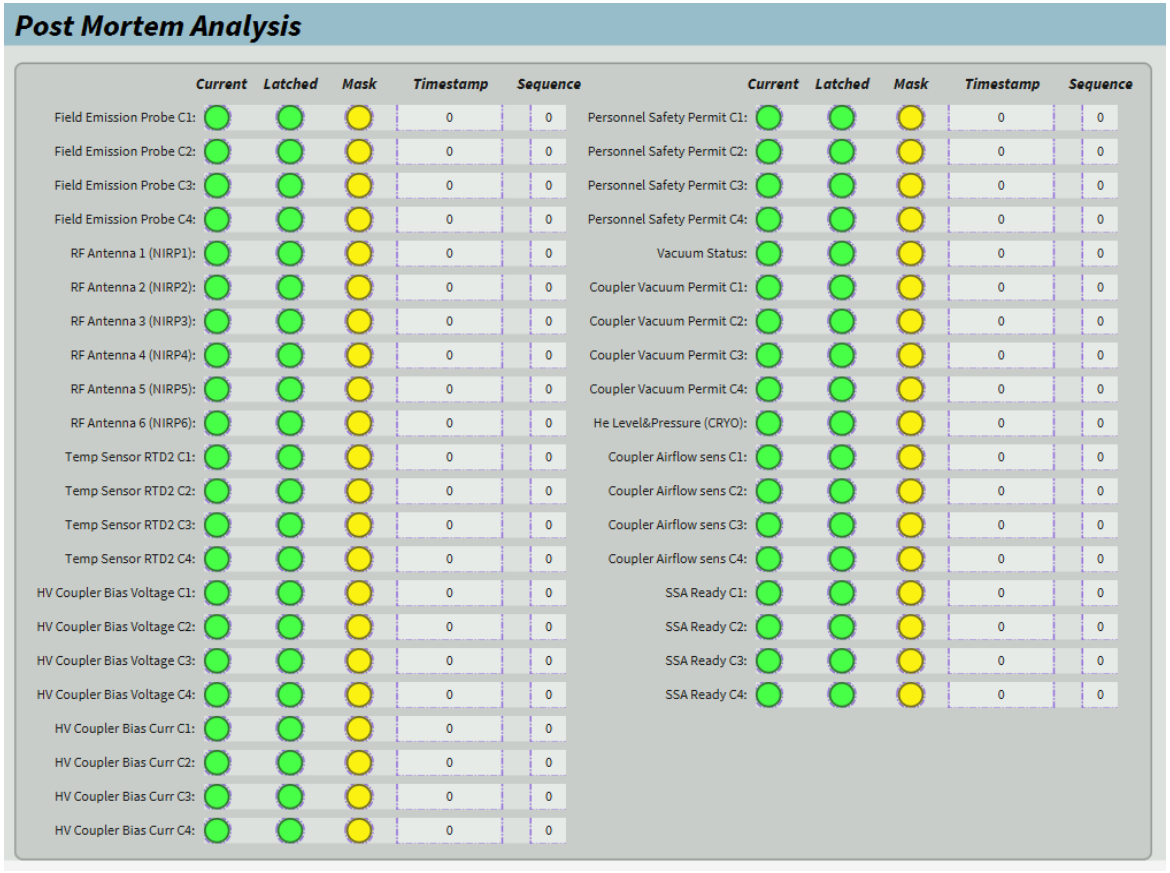


The main logic IOC OPIs – permit window



- LEDs with final, current and latched statuses with timestamps
- Force state mechanism
- Signal plot
- Fuse status

The main logic IOC OPIs – postmortem analysis window



- Summary of all signal events together with timestamps when they have been triggered
- Sorted by timestamps to make postmortem analysis easier

Thank you!