EPICS Development for the ASKAP Design Enhancements Program

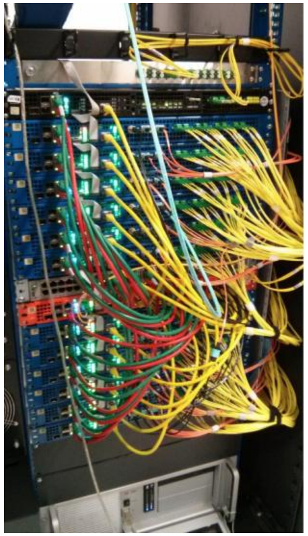
Craig Haskins

EPICS User Meeting – Melbourne 2015
18th October 2015

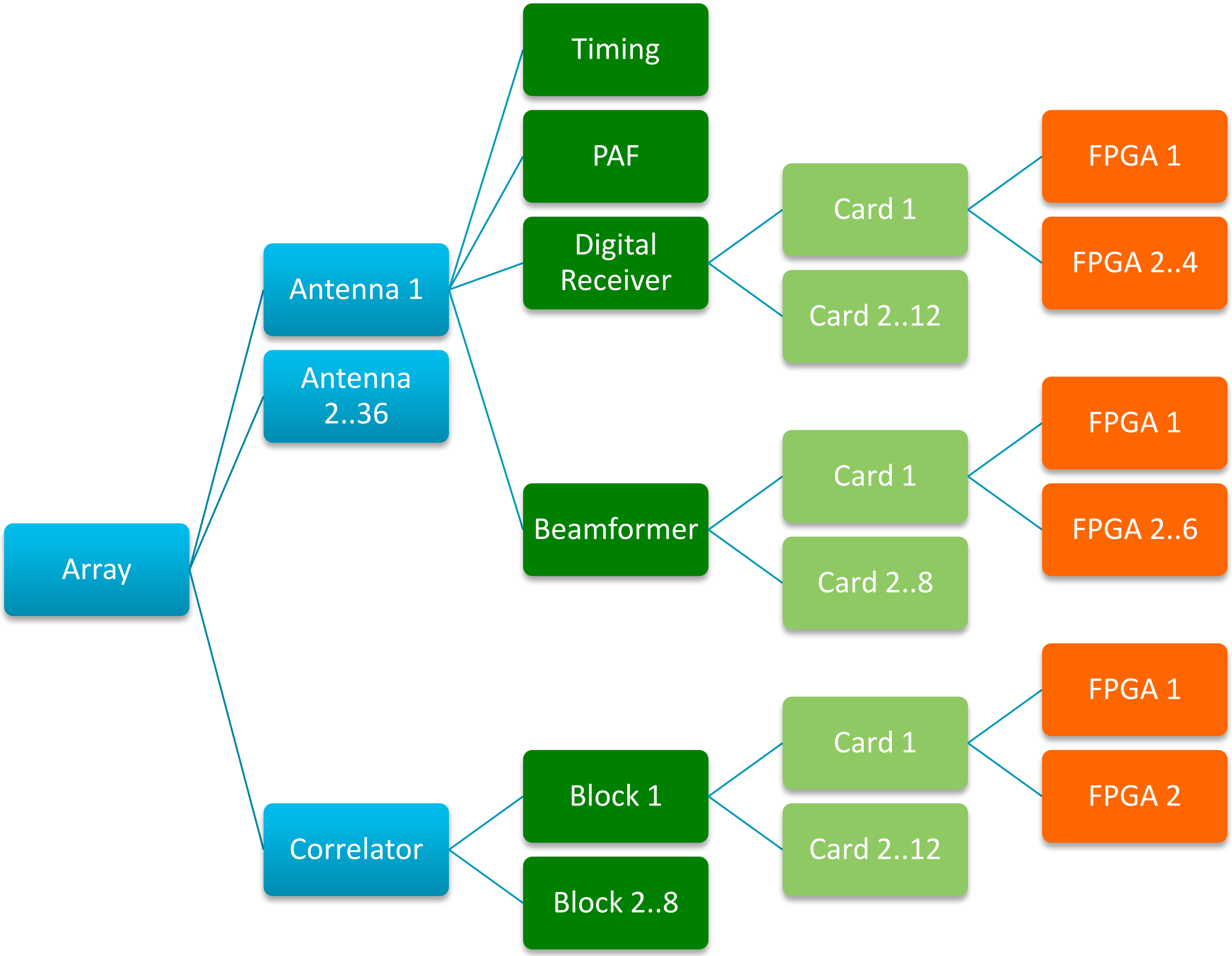
ASKAP Design Enhancements (ADE) Program

- Refresh of electronics hardware
- Mk II Phased Array Feeds (PAF)
- RF over optical fiber
- Digital hardware moved out of pedestals to central site
- Opportunity to revise our EPICS implementation





ASKAP Control & Monitoring Hierarchy



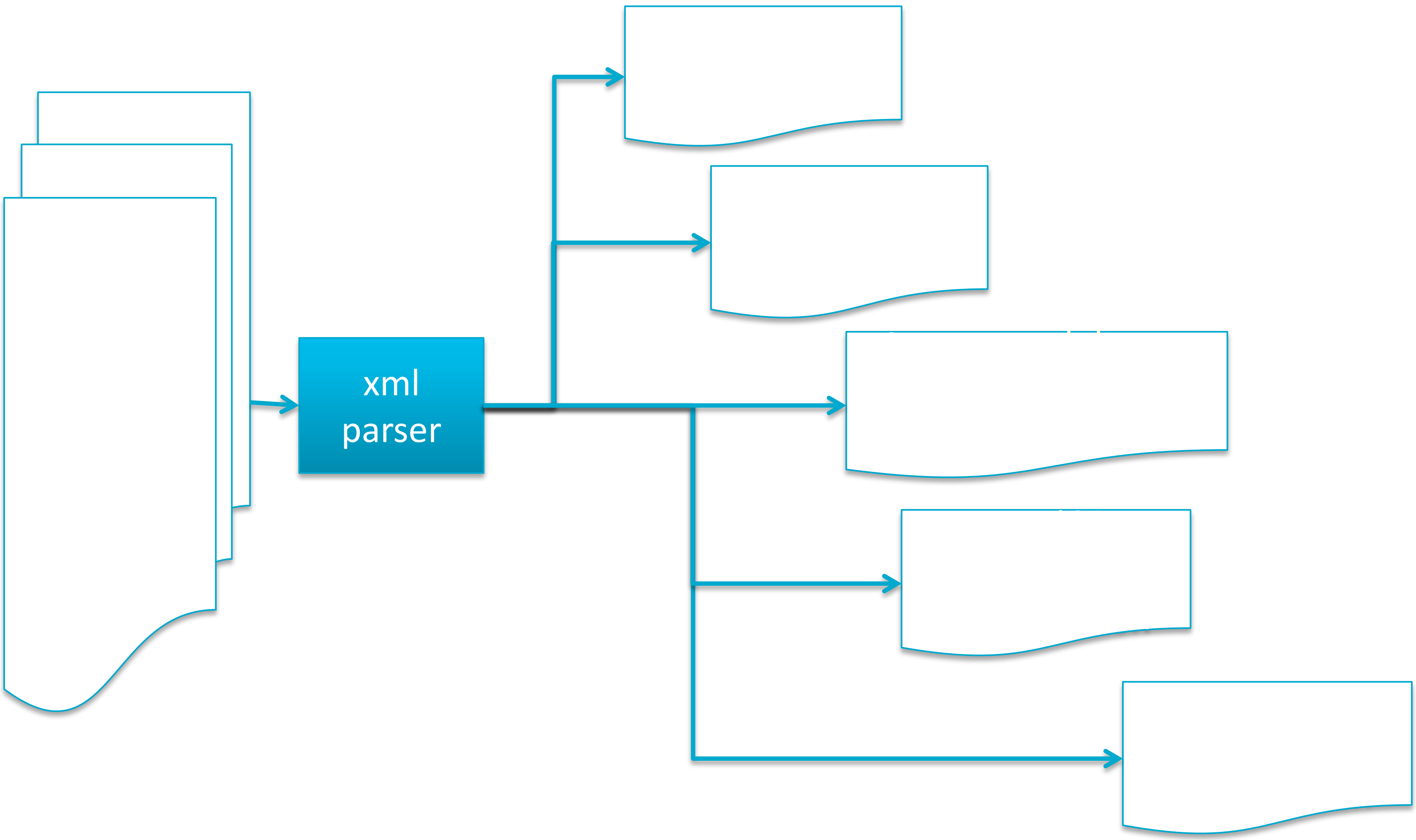
Problem

- Hardware configuration is dynamic
  - number of antennas
  - number of cards
  - features present on cards
- Large number of similar devices to configure and control in parallel
- Large number of hierarchical monitoring points
- Must keep EPICS 3.14

Solution

1. Monitoring points in C structures provided by a Hardware Abstraction Layer (HAL) library annotated with xml tags
2. Build time tools to parse xml and generate IOC configuration files
3. Runtime enhancements to handle disabled, not present or unselected hardware (bigASub, summary & fanout records)
4. Logical control & monitoring groupings via composite IOCs

EPICS DB Generation



EPICS DB Generation Example

```
/// @brief cooling fan info
///
/** @xmlonly <iocStructure name=”FanInfo"> @endxmlonly */
typedef struct FanInfo
{
    AdbeStatus status;  ///< hardware status, e.g. ADBE_NOT_PRESENT (maps to asynDisabled)

    float voltage;  ///< fan voltage
                /** @xmlonly <iocPoint name=”volts" type=”float" egu=”V” comment=”fan volts"/> @endxmlonly */

    float speed;   ///< fan speed
               /** @xmlonly <iocPoint name=”speed" type="float" egu=”rpm” comment=”speed"/> @endxmlonly */
} FanInfo;
/** @xmlonly </iocStructure> @endxmlonly */
```

EPICS DB Generation Example

```
/// @brief a top level structure representing all the monitoring points
///
/** @xmlonly <iocStructure name=”ChassisInfo" type="top"> @endxmlonly */
typedef struct ChassisInfo
{
   AdbeStatus  status;
   float       temp;  ///< temperature
                  /** @xmlonly <iocPoint name=”temp" type=”float" egu=”C" comment=”temp"/> @endxmlonly */

   FanInfo     fan;   ///< chassis fan
                  /** @xmlonly <iocStructure name=”fan" type=”FanInfo" comment=”chassis fan”/>
} ChassisInfo;
/** @xmlonly </iocStructure> @endxmlonly */

/// @brief get current chassis info
ChassisInfo* getChassisInfo();
```

EPICS DB Generation Example

- Parsing the XML will generates EPICS records

```
$(p)ChassisInfo:temp
$(p)ChassisInfo:fan:voltage
$(p)ChassisInfo:fan:speed
```

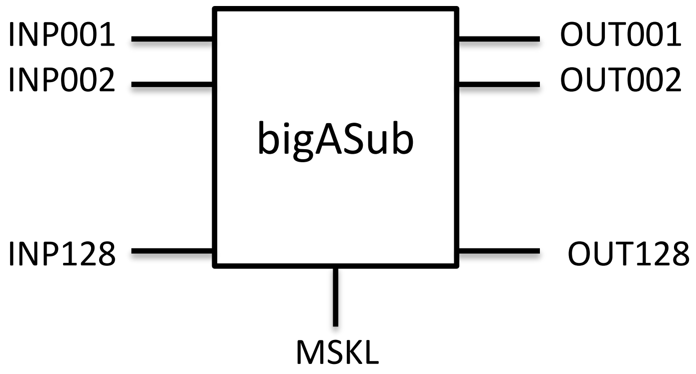
- And to update the records at runtime we call a HAL library method & a generated update function

```
TestInfo* info;
// get structured data from hardware
info = drv->getChassisInfo();
// a single call to update structured EPICS PVs via asyn parameters
Update_ChassisInfo(ASYN_INDEX(ChassisInfo_BEGIN), info);
callParamCallbacks(0);
```

- Can use structure status to set PV alarm status, e.g. asynDisabled
- Other types supported such as arrays, enums
- PV attributes such as alarm states can be specified

BigASub Record

- Forked from aSub record
- More inputs `INP001..INP128`
- More outputs `OUT001..OUT128`
- Numbered I/O for easier partial records
- Mask Input Link `MSKL` for masking out I/O



Summary Records

- Summary records combine an aggregation function with alarm propagation. (.e.g. a summary temp will show max temp plus maximum alarm condition)
- Inputs can be excluded from aggregation & propagation by either being in `DISABLE_ALARM` state or by being masked out
- Supports variable number of inputs via partial records.
- Automatically generated from an XML definition

```
<iocSummary name="temps" type="max" egu="C” comment="max temp">
    <iocSearch name="CtrlMonitorData" egu="C"/>
    <iocSearch name="RedbackMonitorData" egu="C"/>
</iocSummary>
```

Composite IOCs

- ASKAP hardware consists of many instances of identical hardware
- All instances need to be controlled in parallel
- Control commands are fanned out using bigASub
- Monitoring points are aggregated using bigASub
- Handles heterogeneous subsystems
- Handles command sequencing
- IOC load time configuration of components (antennas or subsystems)
- IOC runtime selection of antennas/subsystem via bigASub mask input link

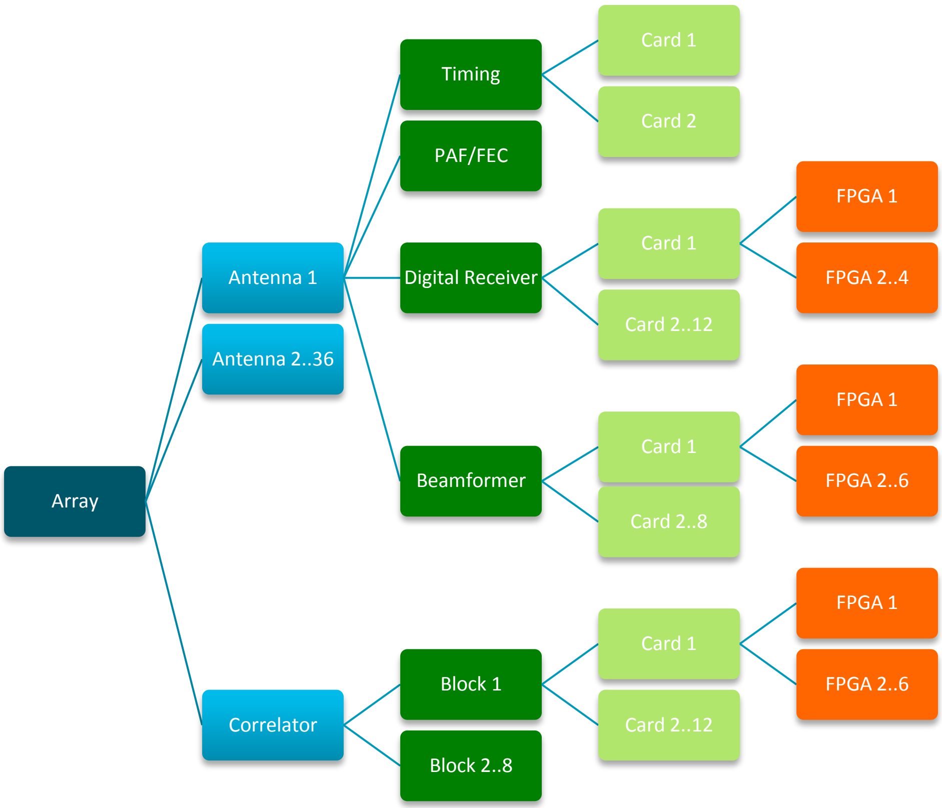
Composite IOCs

•Single set of PV definitions (csv file) for generating Composite IOC database at both antenna and array level. e.g.

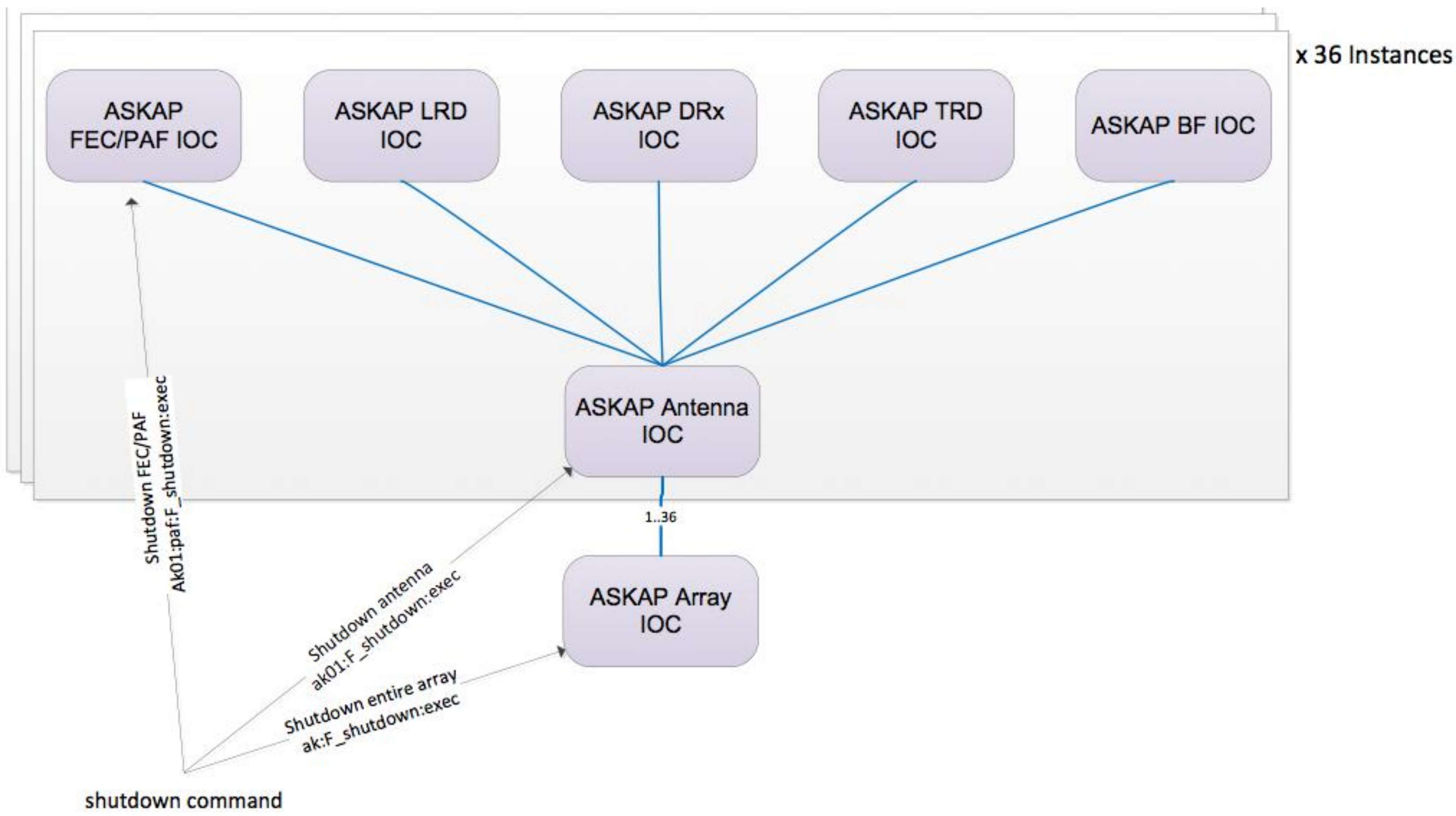
PV	desc	TRD	FEC	DRX	BMF
ctrl1	Control to all subsystems	x	x	x	x
ctrl2	Control to 2 subsystems			x	x
ctrl3	Sequenced control to 3 subsystems	1		3	2

ASKAP Control & Monitoring Hierarchy

Array Composite IOC	x1
Antenna Composite IOC	x36
Subsystem IOC	x152
Asyn port drivers	x888
HAL	x4032



Composite IOC – Control Hierarchy



ASKAP Monitoring Point Hierarchy

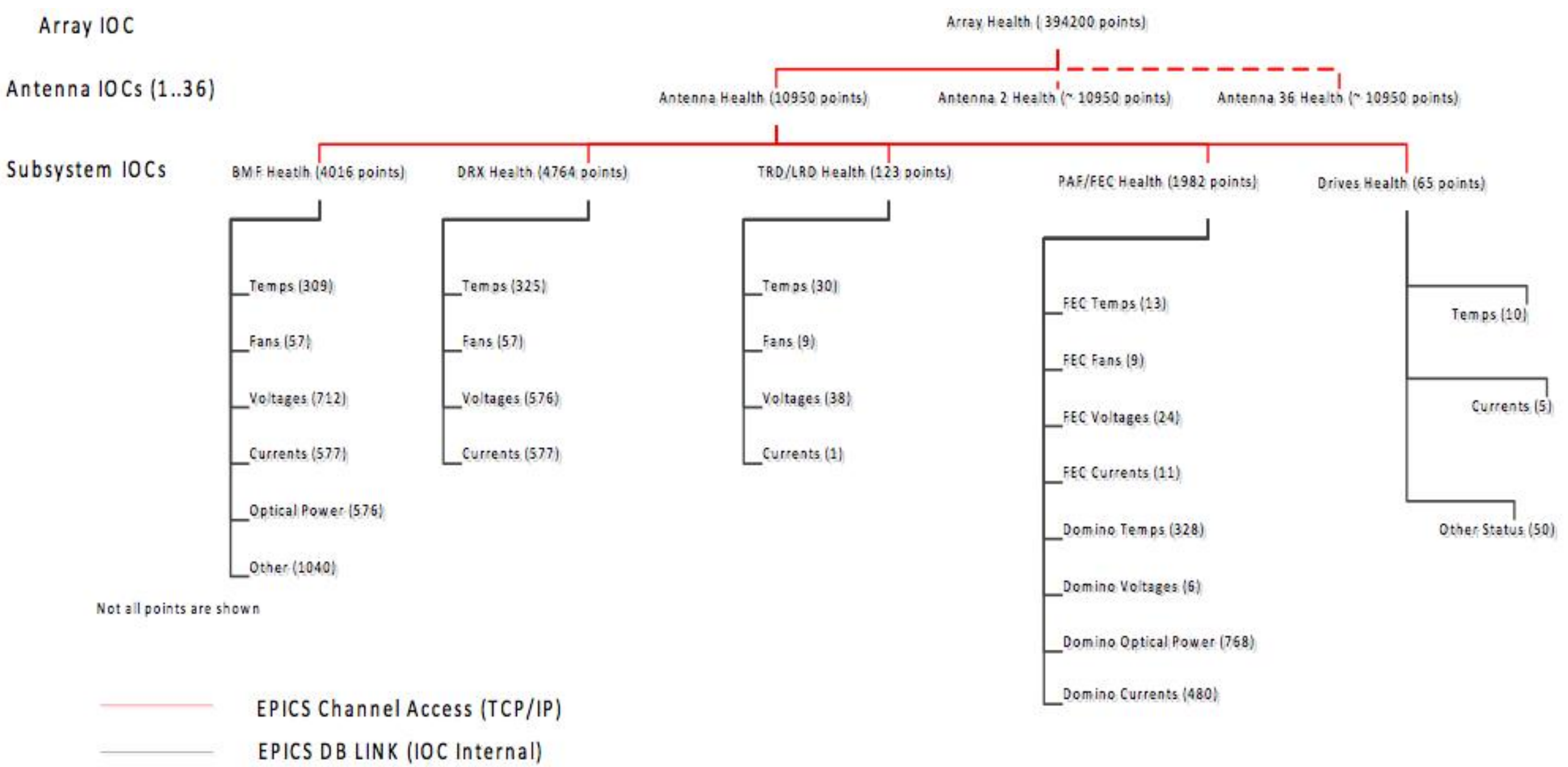


Figure 5 - ASKAP Monitoring Point Hierarchy

Thank you

CSIRO Astronomy and Space Science
Craig Haskins

Software Engineer

t +61 8 6436 8620
e Craig.Haskins@csiro.au
w http://www.atnf.csiro.au