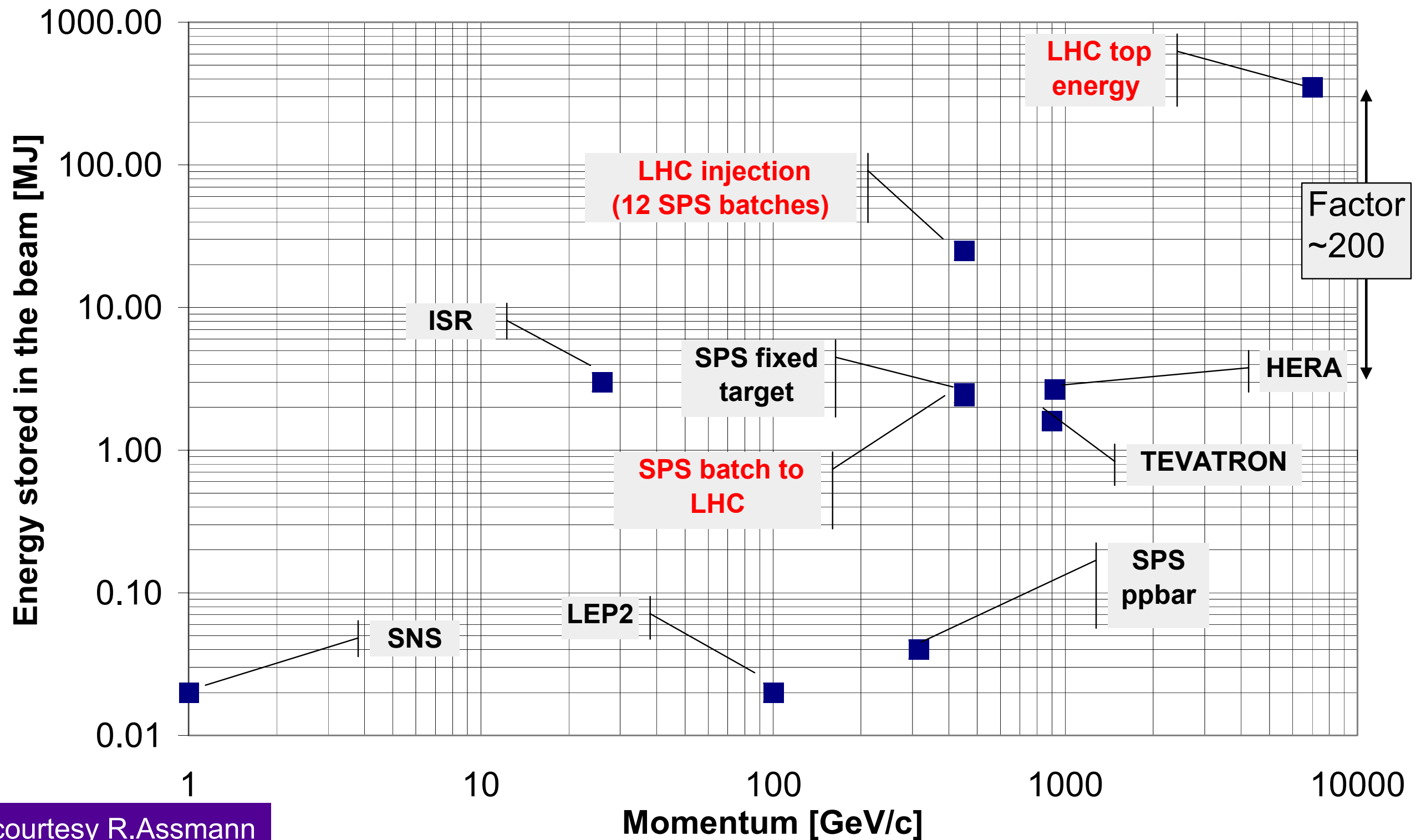


Control System Integration 2

Tom Shea
ORNL

Machine Protection

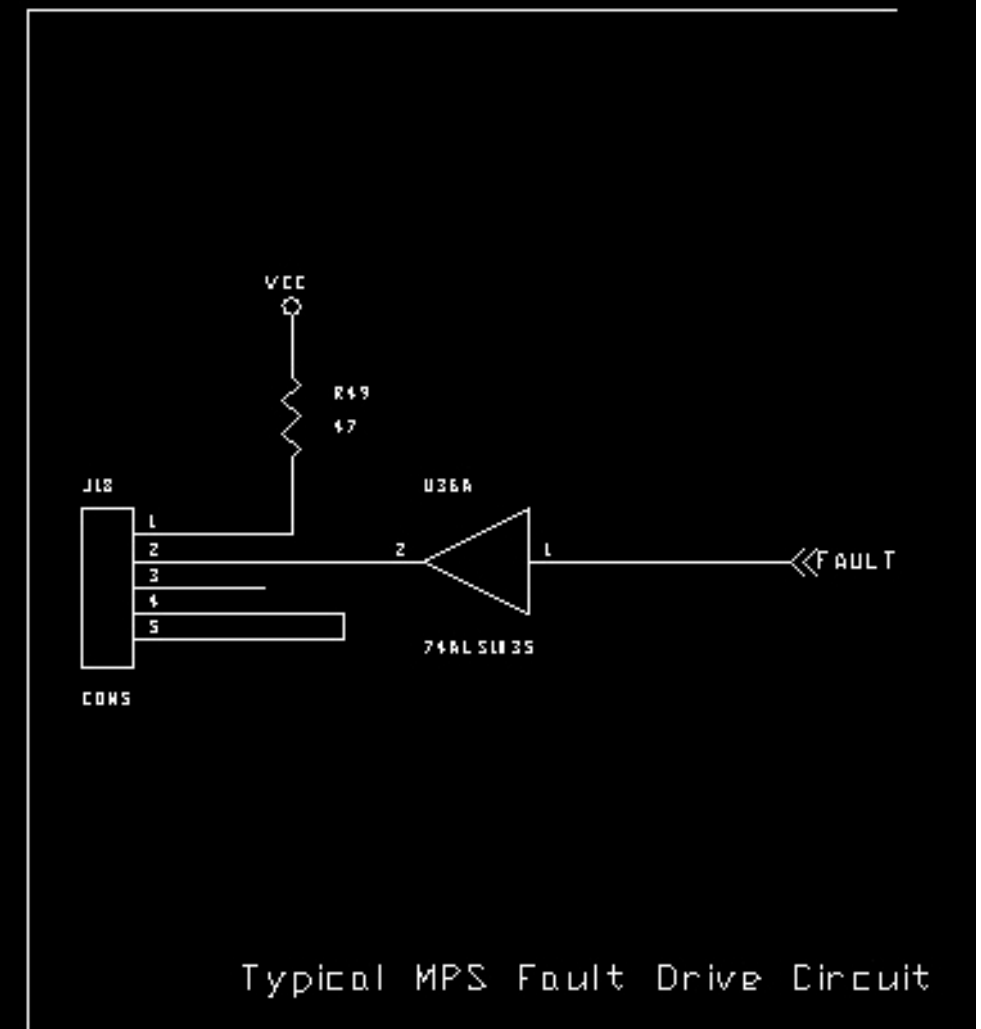
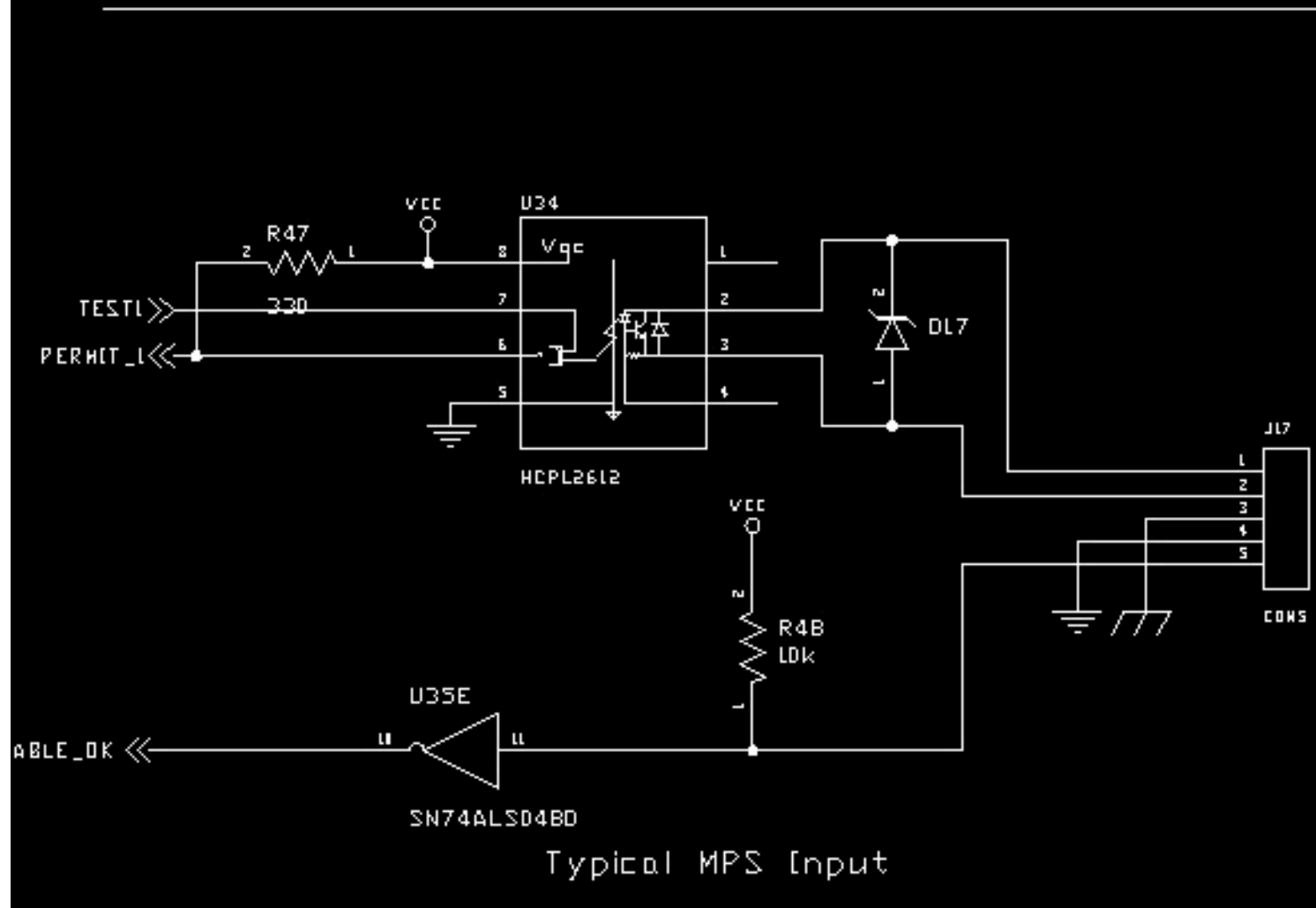
Motivation



Issues

- Subsystem drives MPS input
 - DSP detects loss of control
 - Differential current measurement detects beam loss
- Subsystem responds to MPS trip
 - Communicated on timing or real-time data broadcast system
 - Circular buffers

Interface to MPS

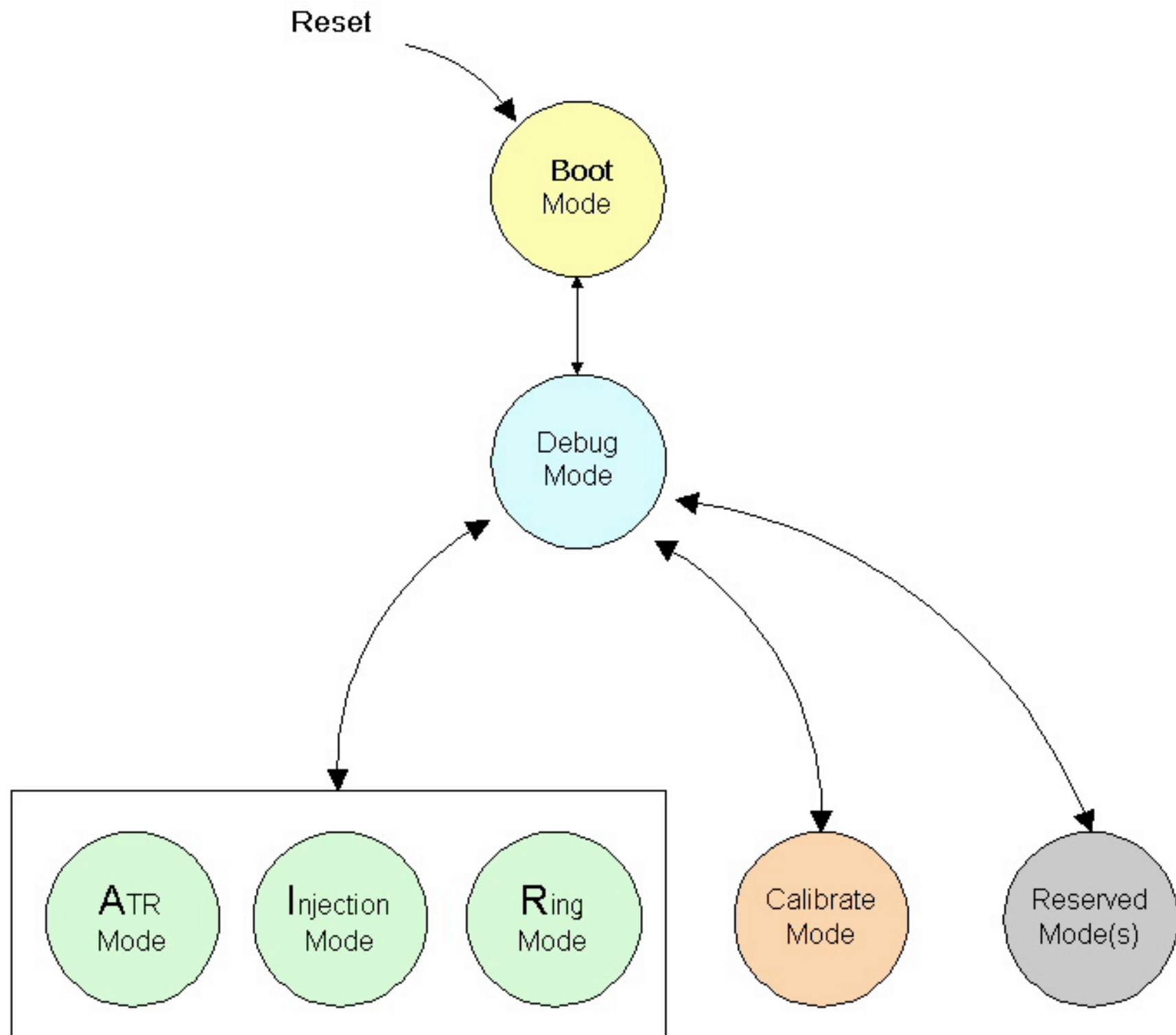


Utility

- Temperature monitor
 - for fault monitoring (filter change time)
 - for correction of thermal effects
- Fan Speed
- Power supply current and voltage
- Heartbeat
 - monitored
 - optional watchdog timer for automatic reset
- Remote Power and Reset
 - Personnel access restrictions
 - Facility Size
 - MTTR

Safe Reconfiguration

- avoiding the “turn antenna away from earth” command.
- protected boot memory containing communication code
- permanent communication subsystem (tiny IOC on a card, hard core on FPGA,...)
- Out of band communication as an option - but may add interconnect or cable in some cases

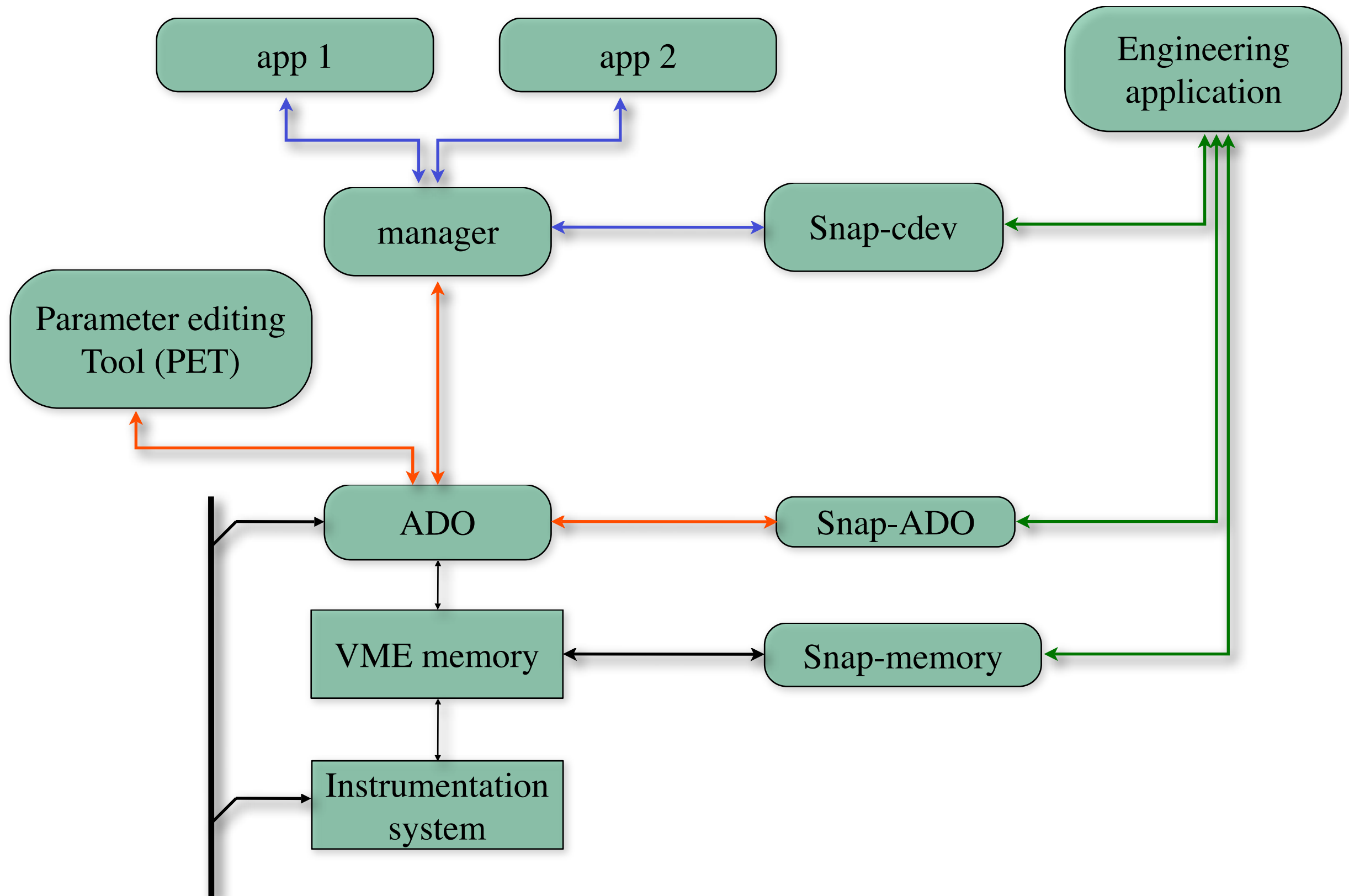


Remote Debugging

by example

- RHIC BPM: Overview
 - Most electronics in radiation area
- SNS LLRF: In depth
 - SNS Equipment gallery put under access restrictions during commissioning

RHIC Debugging Tools



Interface to LLRF Module

- Register based
 - Random read and write access to ease debugging
 - No "don't write during ..." conditions that require tricky timing.
 - No write-only registers that one cannot verify.
- Message based interfaces were avoided
 - Because they are harder to debug.
 - Also harder to implement in an FPGA, requiring parser.
 - One can easily trigger VME/VXI/PCI backplane analyzer on register access. Messages are harder.

Register Documentation

Register† LLRF_DDS_FREQ

The offset frequency for the on-board Direct Digital Synthesizer (DDS). This is a signed 16-bit number that adjusts the frequency of cavity operation, relative to the Master Oscillator, in the range of ± 625 kHz. One bit corresponds to 19.073486328125 Hz exactly (assuming the RF sampling clock is a perfect 40 MHz).

Register LLRF_STATUS

Read-only bit map of module status.

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
-	-	-	-	-	-	-	-	CLP	FLT	LRC2	LRC1	PLL	IL	RF_ON	KCM

CLP: Latched indicator of occurrence of any premature termination of RF due to output magnitude clipping fault. Cleared at beginning of each pulse.

FLT: Latched indicator of occurrence of any premature termination of RF due to faults within the pulse. Cleared at beginning of each pulse.

LRC2,LRC1: FCM-specific readout of the configuration switches that define its position. 0=Left, 1=Center, 2=Right, 3=Invalid. Non-FCM systems read 0.

PLL: Instantaneous readout of the same signal described in the PLL bit of the

"Single Source"

- Perl scripts convert Verilog register definitions into C++ include file, EPICS Database config, ...

The image shows two overlapping code editor windows. The top window, titled 'register-map', displays a list of Verilog registers and their 14-bit binary values. The bottom window, titled 'llrf_defines.h', shows the corresponding C++ preprocessor definitions for these registers. Red arrows highlight the mapping between 'DDS_FREQ' in the register map and 'LLRF_DDS_FREQ' in the defines file.

```
register-map 83
SET_I      14'b000000000100000
SET_Q      14'b000000000100001
DDS_FREQ   14'b000000000100010
CONFIG     14'b000000000100011

llrf_defines.h x
/* automatically generated by intercon.pl ../source/register-map
#define LLRF_CONFIG_ROM      (0x0000)
#define LLRF_ERRORS         (0x0010)
#define LLRF_STATUS         (0x0011)
#define LLRF_UCAPE          (0x0012)
#define LLRF_DCAPE          (0x0013)
#define LLRF_FCAPE          (0x0014)
#define LLRF_INT_STATUS     (0x0015)
#define LLRF_TTLTRG_MON     (0x0016)
#define LLRF_PEAK_ERR       (0x0017)
#define LLRF_PEAK_OUT       (0x0018)
#define LLRF_TOTALIZER      (0x001c)
#define LLRF_SHADOW         (0x0020)
#define LLRF_SET_I          (0x0020)
#define LLRF_SET_Q          (0x0021)
#define LLRF_DDS_FREQ       (0x0022)
#define LLRF_CONFIG         (0x0023)
```

Debugging Registers: Console


- Via serial line or "telnet"

```
test-oper@rftf-test-opi-cow4:~
test-llrf-ioc-vxi> vxiHelp
VXI device support, version 2004-09-23 w/ write checks.

VXI helper routines:

NOTE: Test mode. A24 access actually
uses local RAM when the variable use_A24_TEST is set to 1.
Right now, use_A24_TEST is set to 0

    vxiHelp
    vxiInfo(int la)
    vxiRegister(const char *name, int la)
    vxiReport(int level)
    vxiSearch()
    Word *vxiBoardAddr(int la)
    void *vxiTestA24(int la)
    Bool vxiPeek(int la, int reg)
    Bool vxiPoke(int la, int reg, Word value)
    void vxiMapA24(int la, unsigned long start)
    Word *vxiA24Addr(int la)
    Bool vxiPeekA24(int la, int word)
    Bool vxiPokeA24(int la, int word, Word value)
    worddump(Word *addr, int count)
    value = 83 = 0x53 = 'S'
test-llrf-ioc-vxi>
test-llrf-ioc-vxi> vxiPeekA24(0xD4, 0x22)
LA 0xD4, A24 word 0x22 = 0x0000
value = 1 = 0x1
test-llrf-ioc-vxi> █
```



Debugging Registers: Simple GUI

- **Note:**
Our hardware returns value **0xBAD** for undecoded registers (pullups & downs on data lines in case FPGA doesn't drive them)

VXI Registers

VXI Registers, SCL_LLRF:IOC01a

A16 Read:

LA (hex)	Register(hex, dec)	Value (hex)	(ASCII)	(Bits)
0xd4	0 0	0x4fa0	O_	
0xd4	0x1 1	0x1f4d	_M	

A16 Write:

Register(hex, dec)	Value (hex)	Readback
0 0	0	0x4fa0
0 0	0	0xbad

A24 Read:

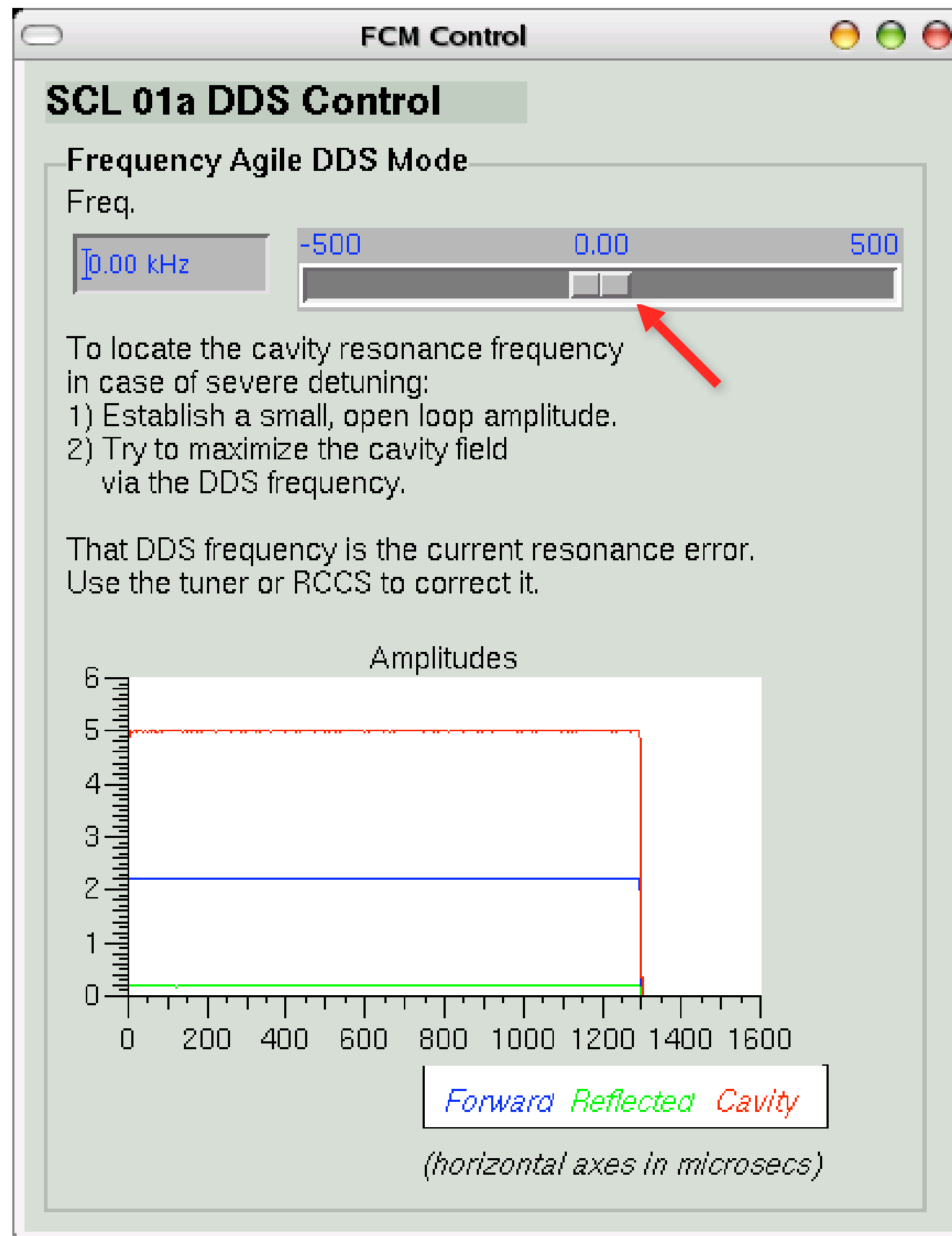
Register(hex, dec)	Value (hex)
0xd4 0x22 34	0

A24 Write:

Register(hex, dec)	Value (hex)
0xd4 0x22 34	0

Use WORD offsets (2-byte) f. Registers, press Return to send.

End-User Register Access



Other Debug Tools

- EPICS Sequencer and Database
 - Online, remote access to internal state
 - View the "raw" value behind some displayed data
 - Timestamps
 - Anything can be displayed on "Strip chart", or added to an archive tool for later analysis - good for infrequent events and unanticipated correlations
- Custom C/C++ Code
 - Access to internal data must be specifically added to the code
 - "Report" methods
 - Debug flags
 - Time stamps

Source Code Control

Version Control Benefits

- Serves as repository
 - Develop some bugfix on laptop, merge that into copy on the main development machine
 - Deploy "latest" version onto linac server
- "Roll back to state of January 18, 2006"
- View history of changes, compare different versions
 - "When did we add this behavior?"
 - "How was this handled 2 years ago?"

CVS - Concurrent Versions System

- A free, open source version control system
 - Available for every operating system
 - Stable
 - Command-line, Emacs, Eclipse, ...
- **Handles text very well**
 - Plain ASCII or LaTeX documents
 - EPICS Sequencer and Database sources
 - C/C++
 - Verilog/VHDL
 - Front-end computer startup files
- For binary files, only date & comments available, no comparisons possible
 - Images, FPGA bitfiles, LabVIEW, ...
- Old
 - "subversion" might be better at handling directories

CVS under Eclipse: Verilog

The screenshot shows the Eclipse IDE interface with the title bar "C/C++ - Compare fdbk_loop.v 1.12 and 1.11 - Eclipse SDK". The menu bar includes File, Edit, Refactor, Navigate, Search, Project, CVS, Window, and Help. The toolbar contains various icons for file operations and development tools.

On the left, the "C/C++ Pr..." and "Navigator" views are visible. The Navigator view shows a list of files in the project, including RAMB4_S8_S8.v 1.5, SRL16E.v 1.5, afterburner.v 1.5, afterdemux.v 1.5, bram.pl 1.4, bram2.pl 1.4, config_cruncher 1.4, config_rom.v 1.5, config_rom_fcm.make 1, config_rom_interim.make, config_rom_meabt.make, cordic.v 1.5, dds.v 1.5, dkcm_bussed.v 1.5, dkcm_controller.v 1.5, dkcm_moving.v 1.5, ds1822_driver.v 1.5, ds2401_driver.v 1.5, fdbk_loop.v 1.12 (ASCII), feedforward.v 1.9, flasher.v 1.5, hist2.v 1.5, histmode.v 1.5, history2e.v 1.8, intercon.pl 1.4, llrf.rules.make 1.4, llrf_config.v 1.5, llrf_fcm.ucf 1.4, llrf_interim.ucf 1.4, llrf_meabt.ucf 1.4, loc.sh 1.4, mult_bussed.v 1.5, mult_controller.v 1.5, and mult_moving.v 1.5.

The main editor area displays a "Text Compare" window titled "Compare fdbk_loop.v 1.12 and 1.11". It shows a side-by-side comparison of the Verilog code for fdbk_loop.v. The left pane is labeled "Repository file: fdbk_loop.v 1.12" and the right pane is labeled "Repository file: fdbk_loop.v 1.11".

The code in the 1.12 version includes a section for "gauging for output magnitude" that is highlighted in blue:

```
// gauging for output magnitude
wire [11:0] peak_out_narrow;
trip clip(
    .clk(clk40), .inval(fdbk_err[11:3]), .trip_thresh(trip_thresh),
    .gate(feedback_on), .reset(trip_reset),
    .tripped(out_clipped), .peak_val(peak_out_narrow)
);
assign peak_out = {peak_out_narrow, 4'b0};
```

The code in the 1.11 version shows a different implementation for the peak error output, using a different set of logic and signals.

At the bottom, the "Problems" and "Console" views are visible. The "Problems" view shows a table of issues, with the following data:

Revision	Tags	Revision Time	Comment
Previous			
*1.12		2/14/07 12:13 PM	firmware with output clipping
1.11		2/14/06 3:42 PM	get new firmware

FPGA Firmware Handling

- Verilog sources are in CVS
 - Full benefit of version comparisons
- Bitfiles also in CVS as 'binary'
 - No insight into changes, but since each "place-and-route" creates different bitfile, it's good to keep a copy of the specific bitfile
- Front-end computer programs FPGA
 - Loads bitfile via network

(For machine protection related FPGA, bitfile is in local EEPROM)

Configuration Control

SNS Device Database

Active configuration file stored here.

Supported by rest of purple tables which include historic.

Diagnostics IOC Active Configuration			
Ioc Config Type Id (FK)	VARCHAR(25)	NOT NULL	
Maj Ver Nbr (FK)	NUMERIC(3,0)	NULL	
Min Ver Nbr (FK)	NUMERIC(4,0)	NULL	
Cmnt	VARCHAR(2000)	NULL	
Modify Date	DATE	NOT NULL	
Mod By Bn	VARCHAR(6)	NOT NULL	

Diagnostics IOC Configuration History			
Ioc Id (FK)	VARCHAR(40)	NOT NULL	
Ioc Config Type Id (FK)	VARCHAR(25)	NOT NULL	
Maj Ver Nbr (FK)	NUMERIC(3,0)	NOT NULL	
Min Ver Nbr	NUMERIC(4,0)	NOT NULL	
Cmnt	VARCHAR(2000)	NULL	
Modify Date	DATE	NOT NULL	
Mod By Bn	VARCHAR(6)	NOT NULL	

Diagnostics IOC Parameter Group			
Ioc Config Type Id (FK)	VARCHAR(25)	NOT NULL	
Maj Ver Nbr (FK)	NUMERIC(3,0)	NOT NULL	
Grp Cnt	NUMERIC(3,0)	NOT NULL	
Paramr Grp Nm	VARCHAR(25)	NULL	
Cmnt	VARCHAR(2000)	NULL	

Diag IOC Config Maj Ver			
Ioc Config Type Id (FK)	VARCHAR(25)	NOT NULL	
Maj Ver Nbr	NUMERIC(3,0)	NOT NULL	
Cmnt	VARCHAR(2000)	NULL	
Modify Date	DATE	NOT NULL	
Mod By Bn	VARCHAR(6)	NOT NULL	

Diagnostics IOC Parameter			
Ioc Config Type Id (FK)	VARCHAR(25)	NOT NULL	
Maj Ver Nbr (FK)	NUMERIC(3,0)	NOT NULL	
Grp Cnt (FK)	NUMERIC(3,0)	NOT NULL	
Paramr Cnt	NUMERIC(4,0)	NOT NULL	
Paramr Nm	VARCHAR(50)	NOT NULL	
Paramr Type Id (FK)	VARCHAR(25)	NOT NULL	

Parameter Datatype	
Paramr Type Id	VARCHAR(25) NOT NULL
Datatype Precs	NUMERIC(3,0) NULL

Diagnostics IOC Parameter Value

```

BEGIN
  SELECT config_file_nm, config_file_loc
  INTO v_config_file_nm, v_config_file_loc
  FROM epics.ioc_config_type
  WHERE ioc_config_type_id = (SELECT ioc_config_type_id
                              FROM epics.ioc_dvc
                              WHERE dvc_id = p_dvc_id);

  EXCEPTION
  WHEN NO_DATA_FOUND
  THEN
    RAISE;
END;

SELECT maj_ver_nbr, min_ver_nbr, ioc_config_type_id
INTO v_maj_ver_nbr, v_min_ver_nbr, v_ioc_config_t
FROM diag_ioc_act_config
WHERE dvc_id = p_dvc_id;
EXCEPTION
WHEN NO_DATA_FOUND
THEN
  ops$oracle.global_utils.v_errornum := SOLCODE;

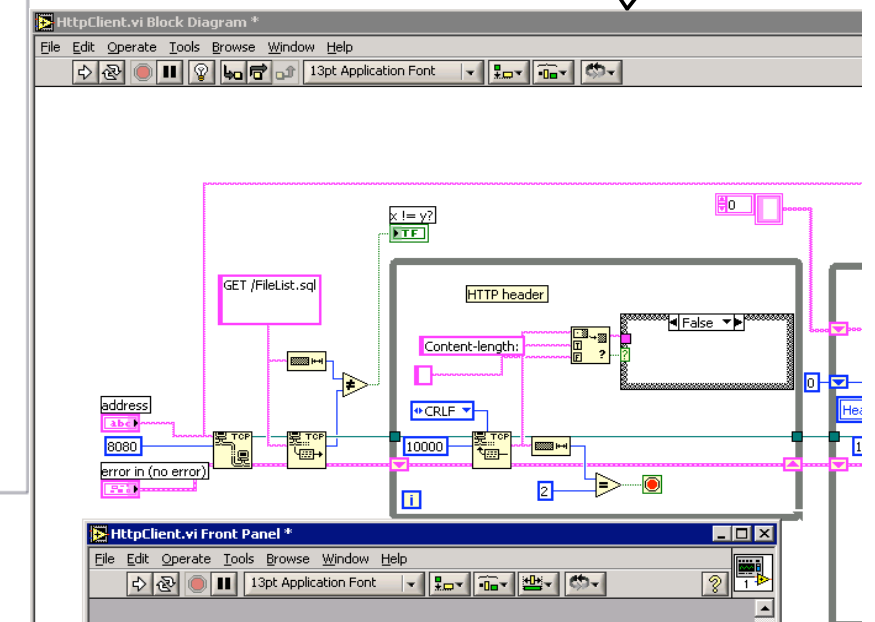
```

Portion of stored procedure used when IOC Configuration File generation UI creates a new configuration file for a given IOC. These selects confirm active versions which then dictate data to be used to create the file stored for use.

Device		
Dvc Id	VARCHAR(40)	NOT NULL
Dvs Id (FK)	VARCHAR(10)	NULL
Subsys Id (FK)	VARCHAR(10)	NULL
Dvc Type Id (FK)	VARCHAR(25)	NULL
Dvc Inst	VARCHAR(25)	NULL
Parent Dvc Id (FK)	VARCHAR(40)	NULL
Dvc Desc	VARCHAR(255)	NULL
BL Dvc Ind	CHAR(1)	NOT NULL
Virtual Dvc Ind	CHAR(1)	NOT NULL
Invalid Id Ind	CHAR(1)	NOT NULL
Dvc Id Alias	VARCHAR(75)	NULL
Act Dvc Ind	CHAR(1)	NULL
MPS Dvc Ind	CHAR(1)	NOT NULL
Dvc Dns Nbr	VARCHAR(50)	NULL
Dns Dvc Id	VARCHAR(40)	NULL
Mod By Uld	VARCHAR(64)	NULL
Mod Dte	DATE	NULL
Owner Cat Id	VARCHAR(10)	NULL

IOC Configuration File		
Dvc Id (FK)	VARCHAR(40)	NOT NULL
Config File Nm	VARCHAR(255)	NOT NULL
Config File Loc	VARCHAR(128)	NULL
Config File Nbr	NUMERIC(50,0)	NULL
Contents	LONG VARCHAR	NULL
Modify Date	DATE	NOT NULL
Mod By Uld	VARCHAR(64)	NOT NULL
Editable Indicator	CHAR(1)	NOT NULL

IOC Config File Hist		
Dvc Id (FK)	VARCHAR(40)	NOT NULL
Config File Nm	VARCHAR(100)	NULL
Modify Date	DATE	NOT NULL
Config File Loc	VARCHAR(128)	NULL
Config File Cont	TEXT	NULL
Mod By Uld	VARCHAR(64)	NOT NULL



IOC Application which pulls Configuration file to IOC from RDB. Uses HTTP Socket Library TCP/IP to connect to Web server Web Server uses standard RDB connectivity: PHP, JSP, ASP

Files pulled from here.

Example: LLRF Multiplicity

- Almost 100 SNS LLRF Systems, handled by ~50 front-ends
 - As different as warm vs. super-conducting cavities
- One source base for all of them
 - Differences handled by configuration settings
 - If possible, startup files and overview displays script-generated from central system info.

NC Linac

RF/MPS/Loop/AFF

RFQ 1
(0)

MEBT 1
(1)

MEBT 2
(2)

MEBT 3
(3)

MEBT 4
(4)

DTL 1
(5)

DTL 2
(6)

DTL 3
(7)

DTL 4
(8)

DTL 5
(9)

DTL 6
(10)

CCL 1
(11)

CCL 2
(12)

CCL 3
(13)

CCL 4
(14)

Medium beta SCL

RF/MPS/Loop/AFF

01a
(15)

01b
(16)

01c
(17)

02a
(18)

02b
(19)

02c
(20)

03a
(21)

03b
(22)

03c
(23)

04a
(24)

04b
(25)

04c
(26)

05a
(27)

05b
(28)

05c
(29)

06a
(30)

06b
(31)

High beta SCL

RF/MPS/Loop/AFF

06c
(32)

07a
(33)

07b
(34)

07c
(35)

08a
(36)

08b
(37)

08c
(38)

09a
(39)

09b
(40)

09c
(41)

10a
(42)

10b
(43)

10c
(44)

11a
(45)

11b
(46)

11c
(47)

12a
(48)

12b
(49)

12c
(50)

12d
(51)

13a
(52)

13b
(53)

13c
(54)

13d
(55)

14a
(56)

14b
(57)

14c
(58)

14d
(59)

15a
(60)

15b
(61)

15c
(62)

15d
(63)

16a
(64)

RF/MPS/Loop/AFF

16b
(65)

16c
(66)

16d
(67)

17a
(68)

17b
(69)

17c
(70)

17d
(71)

18a
(72)

18b
(73)

18c
(74)

18d
(75)

19a
(76)

19b
(77)

19c
(78)

19d
(79)

20a
(80)

20b
(81)

RF/MPS/Loop/AFF

20c
(82)

20d
(83)

21a
(84)

21b
(85)

21c
(86)

21d
(87)

22a
(88)

22b
(89)

22c
(90)

22d
(91)

23a
(92)

23b
(93)

23c
(94)

23d
(95)

Legend:

A! RF Ampl. off Auto-Run goal

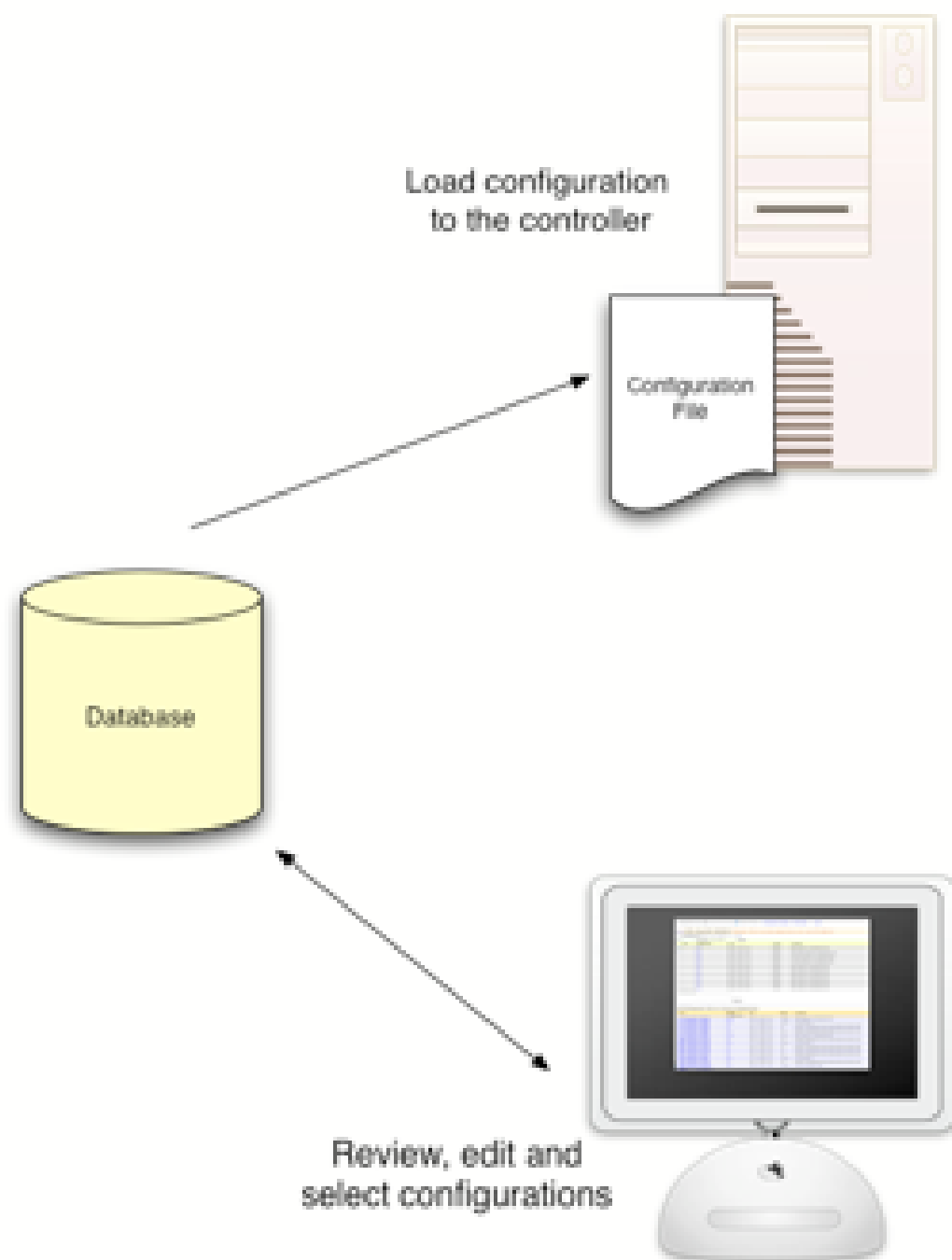
P! RF Phase is shifted

RF System	NC Linac	Medium beta SCL	High beta SCL	RF/MPS/Loop/AFF
RFQ 1 (0)	On	On	On	On
MEBT 1 (1)	On	On	On	On
MEBT 2 (2)	On	On	On	On
MEBT 3 (3)	On	On	On	On
MEBT 4 (4)	On	On	On	On
DTL 1 (5)	On	On	On	On
DTL 2 (6)	On	On	On	On
DTL 3 (7)	On	On	On	On
DTL 4 (8)	On	On	On	On
DTL 5 (9)	On	On	On	On
DTL 6 (10)	On	On	On	On
CCL 1 (11)	On	On	On	On
CCL 2 (12)	On	On	On	On
CCL 3 (13)	On	On	On	On
CCL 4 (14)	On	On	On	On
01a (15)	On	On	On	On
01b (16)	On	On	On	On
01c (17)	On	On	On	On
02a (18)	On	disabled	On	On
02b (19)	On	On	On	On
02c (20)	On	On	On	On
03a (21)	On	On	On	On
03b (22)	On	On	On	On
03c (23)	On	On	On	On
04a (24)	On	On	On	On
04b (25)	On	On	On	On
04c (26)	On	On	On	On
05a (27)	On	On	On	On
05b (28)	On	On	On	On
05c (29)	On	On	On	On
06a (30)	On	On	On	On
06b (31)	On	On	On	On
06c (32)	On	On	On	On
07a (33)	On	On	On	On
07b (34)	On	On	On	On
07c (35)	On	On	On	On
08a (36)	On	On	On	On
08b (37)	On	On	On	On
08c (38)	On	On	On	On
09a (39)	On	On	On	On
09b (40)	On	disabled	On	On
09c (41)	On	On	On	On
10a (42)	On	On	On	On
10b (43)	On	On	On	On
10c (44)	On	On	On	On
11a (45)	On	On	On	On
11b (46)	On	disabled	On	On
11c (47)	On	On	On	On
12a (48)	On	On	On	On
12b (49)	On	On	On	On
12c (50)	On	On	On	On
12d (51)	On	disabled	On	On
13a (52)	On	On	On	On
13b (53)	On	On	On	On
13c (54)	On	On	On	On
13d (55)	On	On	On	On
14a (56)	On	On	On	On
14b (57)	On	On	On	On
14c (58)	On	disabled	On	On
14d (59)	On	On	On	On
15a (60)	On	On	On	On
15b (61)	On	On	On	On
15c (62)	On	On	On	On
15d (63)	On	On	On	On
16a (64)	On	On	On	On
16b (65)	On	On	On	On
16c (66)	On	On	On	On
16d (67)	On	On	On	On
17a (68)	On	On	On	On
17b (69)	On	On	On	On
17c (70)	On	On	On	On
17d (71)	On	On	On	On
18a (72)	On	On	On	On
18b (73)	On	On	On	On
18c (74)	On	On	On	On
18d (75)	On	On	On	On
19a (76)	On	On	On	On
19b (77)	On	disabled	On	On
19c (78)	On	On	On	On
19d (79)	On	On	On	On
20a (80)	On	On	On	On
20b (81)	On	On	On	On
20c (82)	On	On	On	On
20d (83)	On	On	On	On
21a (84)	On	On	On	On
21b (85)	On	On	On	On
21c (86)	On	On	On	On
21d (87)	On	On	On	On
22a (88)	On	On	On	On
22b (89)	On	On	On	On
22c (90)	On	On	On	On
22d (91)	On	On	On	On
23a (92)	On	On	On	On
23b (93)	On	On	On	On
23c (94)	On	On	On	On
23d (95)	On	On	On	Off

Configuration File Strategy

- Track changes to configuration files
 - Who made the change
 - When was the change made
 - Why was the change made
- Restore past configuration files when necessary
- Configuration consists of **structure** and **data**
 - Structure (collection of properties that describe the device) is typically common across devices of a specific type
 - Data typically varies for each device and represents the values for a device's properties
 - Structure is associated with a configuration's major version number and data is associated with the minor version

Configuration File Storage/Retrieval



Configuration Type: BCM Device Overview | [Configuration Templates](#) | [Batch Import](#) | [Logout](#)

[Close](#) DTL_Diag:IOC_BCM622 *Warning: This is not the default device for your IP Address!*

25 Configurations Display 10 items Page 1 of 3

+ Choose File no file selected Upload

Active	Configuration	Date	Author	Comment
<input checked="" type="radio"/>	17.22	May 01, 2007 16:14	900688	Batch upload of configuration files.
<input type="radio"/>	17.21	May 01, 2007 15:48	900688	Demo Batch upload of configuration files.
<input type="radio"/>	17.20	May 01, 2007 11:36	900688	Test batch upload of configuration files.
<input type="radio"/>	17.19	May 01, 2007 11:36	900688	Test batch upload of configuration files.
<input type="radio"/>	17.18	May 01, 2007 10:46	900688	Batch upload of configuration files.
<input type="radio"/>	17.17	May 01, 2007 10:44	900688	Batch upload of configuration files.
<input type="radio"/>	17.16	May 01, 2007 10:43	900688	Batch upload of configuration files.
<input type="radio"/>	17.15	May 01, 2007 10:42	900688	Batch upload of configuration files.
<input type="radio"/>	17.14	May 01, 2007 10:41	900688	Batch upload of configuration files.
<input type="radio"/>	17.13	May 01, 2007 10:40	900688	Batch upload of configuration files.

Activation notes:

Activate

BCM Diagnostic Device Active Configurations:

Device	Active Configuration	Date	Editor	Comment
CCL_Diag:IOC_BCM102	14.22	May 01, 2007 16:14	900688	Batch upload of configuration files.
DTL_Diag:IOC_BCM400	11.1	Nov 13, 2006 14:51	900688	Use the INI file.
DTL_Diag:IOC_BCM428	2.1	May 31, 2006 14:28	9PJ	Original Configuration taken during initial insert May 2006
DTL_Diag:IOC_BCM600	2.1	May 31, 2006 14:28	9PJ	Original Configuration taken during initial insert May 2006
DTL_Diag:IOC_BCM622	17.22	May 01, 2007 16:14	900688	Batch upload of configuration files.
EDmp_Diag:IOC_BCM02	4.1	May 31, 2006 14:28	9PJ	Original Configuration taken during initial insert May 2006
HEBT_Diag:IOC_BCM01	12.1	Nov 13, 2006 14:52	900688	Use the INI file.
HEBT_Diag:IOC_BCM09	2.1	May 31, 2006 14:28	9PJ	Original Configuration taken during initial insert May 2006
HEBT_Diag:IOC_BCM20	2.1	May 31, 2006 14:28	9PJ	Original Configuration taken during initial insert May 2006
HEBT_Diag:IOC_BCM32	2.1	May 31, 2006 14:28	9PJ	Original Configuration taken during initial insert May 2006
IDmp_Diag:IOC_BCM01	11.1	Nov 16, 2006 12:55	900688	Use the INI file.
LDmp_Diag:IOC_BCM05	2.1	May 31, 2006 14:28	9PJ	Original Configuration taken during initial insert May 2006
LEBT_Diag:IOC_BCM1t4	11.1	May 01, 2007 11:44	900688	Test new activation code.

Implementation Choices

Flexibility
Rapid development



Performance

- Human
- Commercial/Scripted High Level
- High Level Application (Multiuser OS)
- Low Level Application (RTOS target)
- Embedded (DSP, limited OS)
- FPGA
- ASIC
- Analog

“Some folk built like this, some folk built like that
But the way I'm built, you shouldn't call me fat
Because I'm built for comfort, I ain't built for speed...”

- Willie Dixon

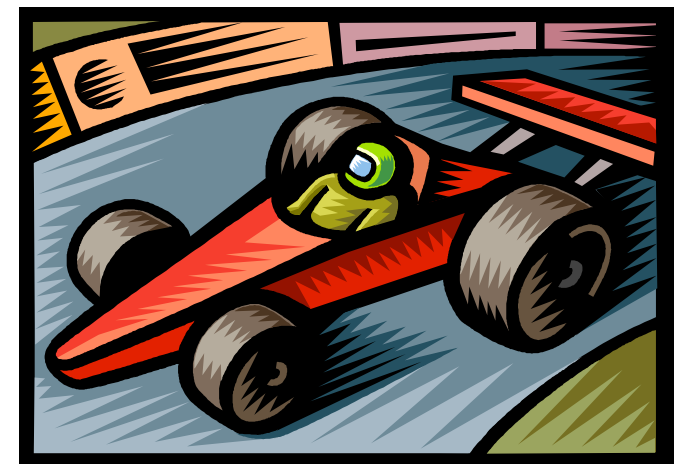
Flexibility or Performance?

- Flexibility in the form of
 - Rapid development, independent testing, remote access, online changes, rich set of debug tools



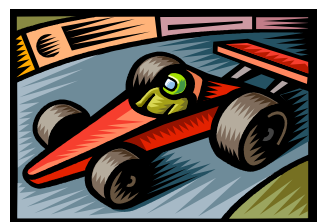
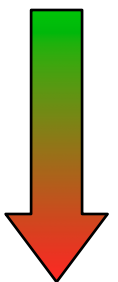
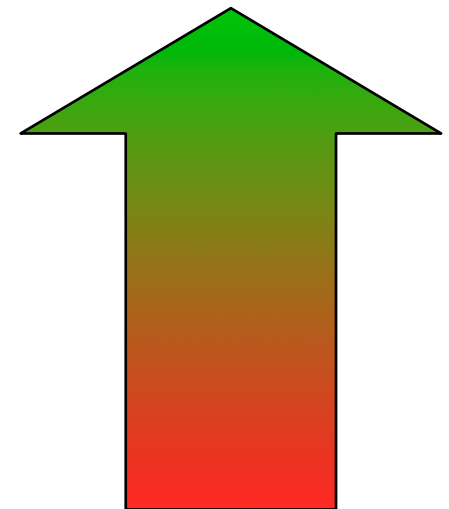
... often differs from ...

- Performance
 - Fast startup times, short response times, deterministic "real time" behavior.



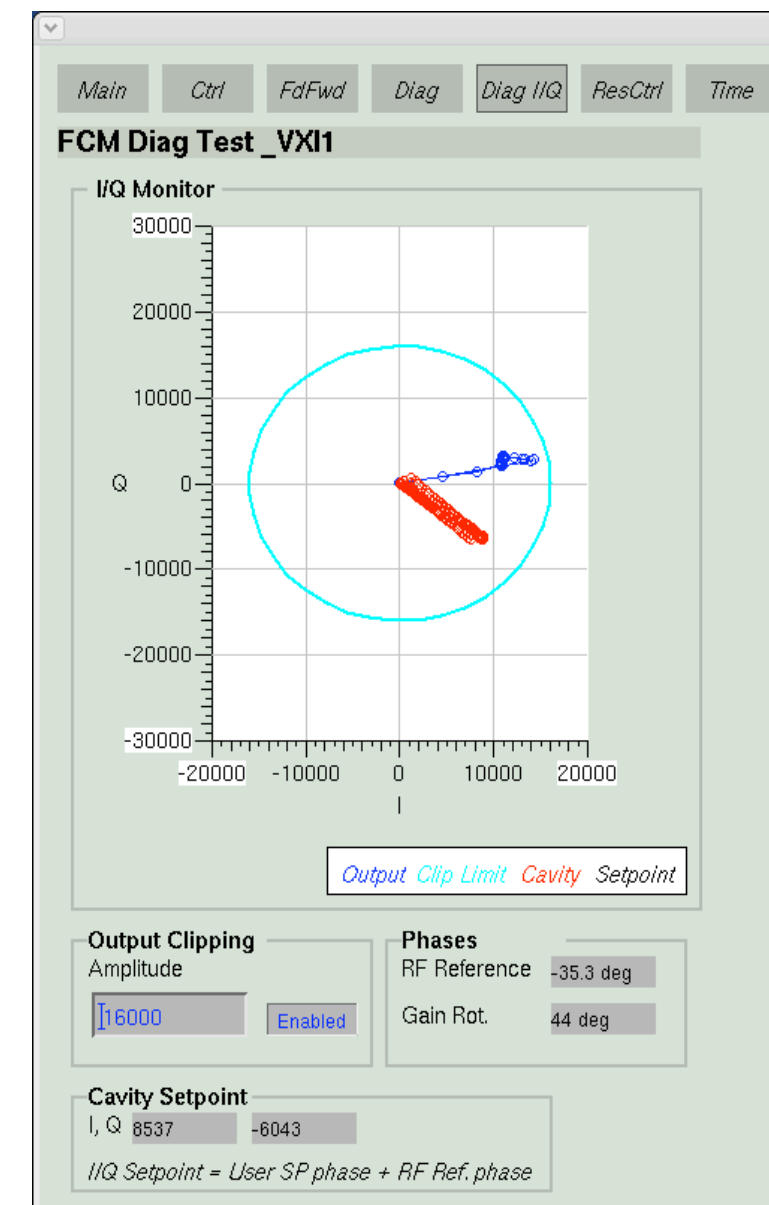
SNS LLRF Choices

- Software based a control system **framework**
(Experimental Physics and Industrial Control System, EPICS)
 - Matlab scripts for test & development of algorithms
 - Front-End computer uses EPICS **State Machine** Tool for automation, and "runtime **Database**" for data flow.
 - C/C++ driver code
-
- **Fast** Feedback (~40MHz) and interlocks in **Verilog**, VHDL, AHDL. Several Iterations
 - Hardware as simple as possible: Analog filtering, ADCs/DACs, then **FPGA**



Operator Interface: Display Manager

- Drawing package for
 - Placing labels, text-monitors, meters, ... on a screen
 - Connecting them to online Process Variables
 - Display panel **Configuration** instead of coding

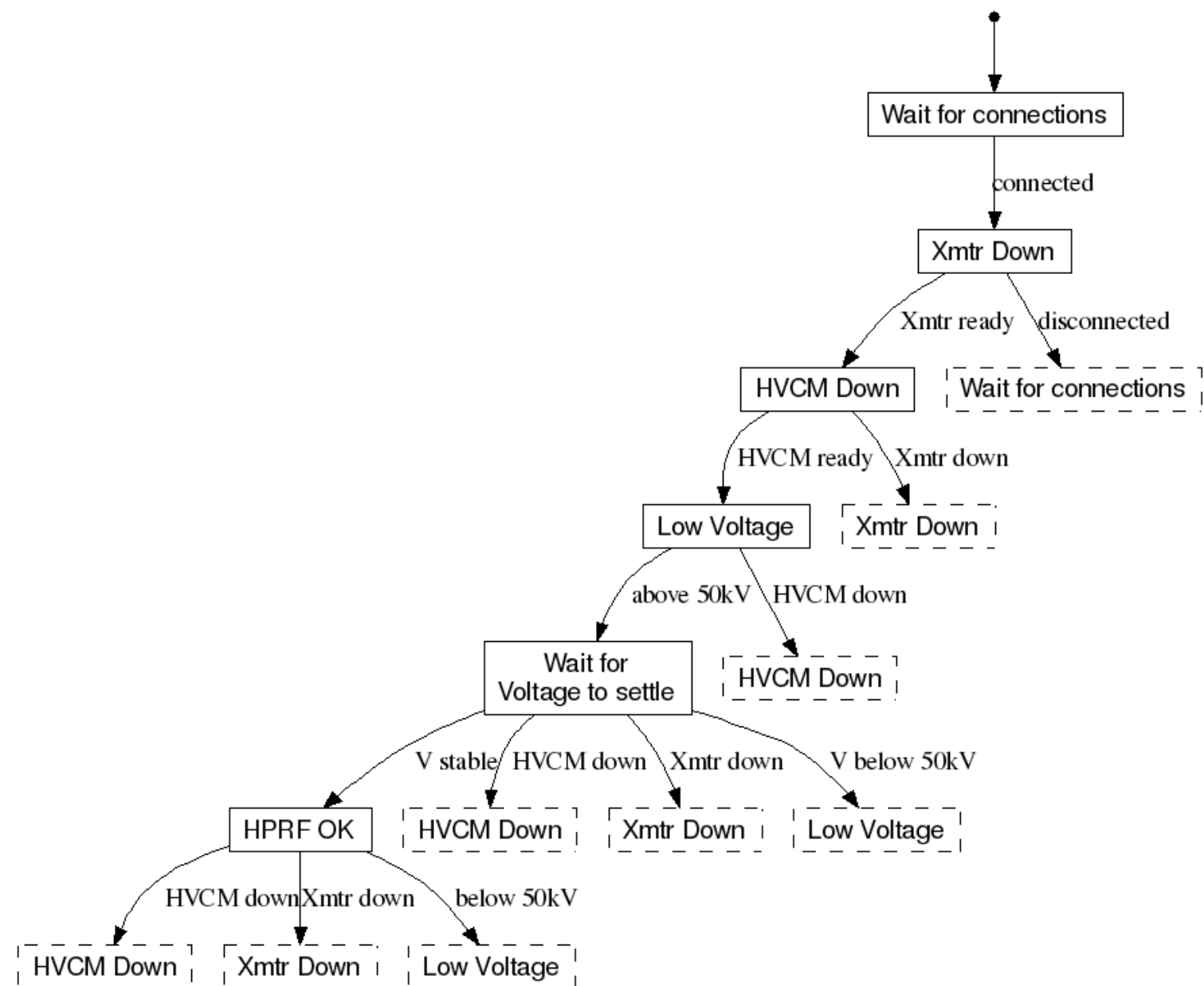


Overall SNS LLRF Strategy

- Requirements change, so **Flexibility** is key.
- Resonance Error computation, feedback loop setup, ...:
 - If possible, first developed in Matlab
 - Then implemented on Front-End as State Machine or EPICS Database
 - If necessary, later moved into custom C++ driver code, or even FPGA

State Machine (EPICS "Sequencer")

- Used for automation whenever possible
- "On Demand" tasks, response times of .1 to 1 sec
- Safest and most flexible tool
 - Runs on host as well as front-end
 - Start/Stop/Update without front-end reboot



EPICS "Database"

- Used for steady-state control, data flow.
- **Full remote access** to any detail.
- Limited online changes.
- "Records", building blocks
 - Read input, computation, write output, ...
- Database Engine handles
 - Periodic or event-driven scanning
 - Time stamps(!)
 - Check of alarm limits
 - Publication of data in "Process Variables"
- Response times of millisecs possible, or more than 10000 records per front-end computer.

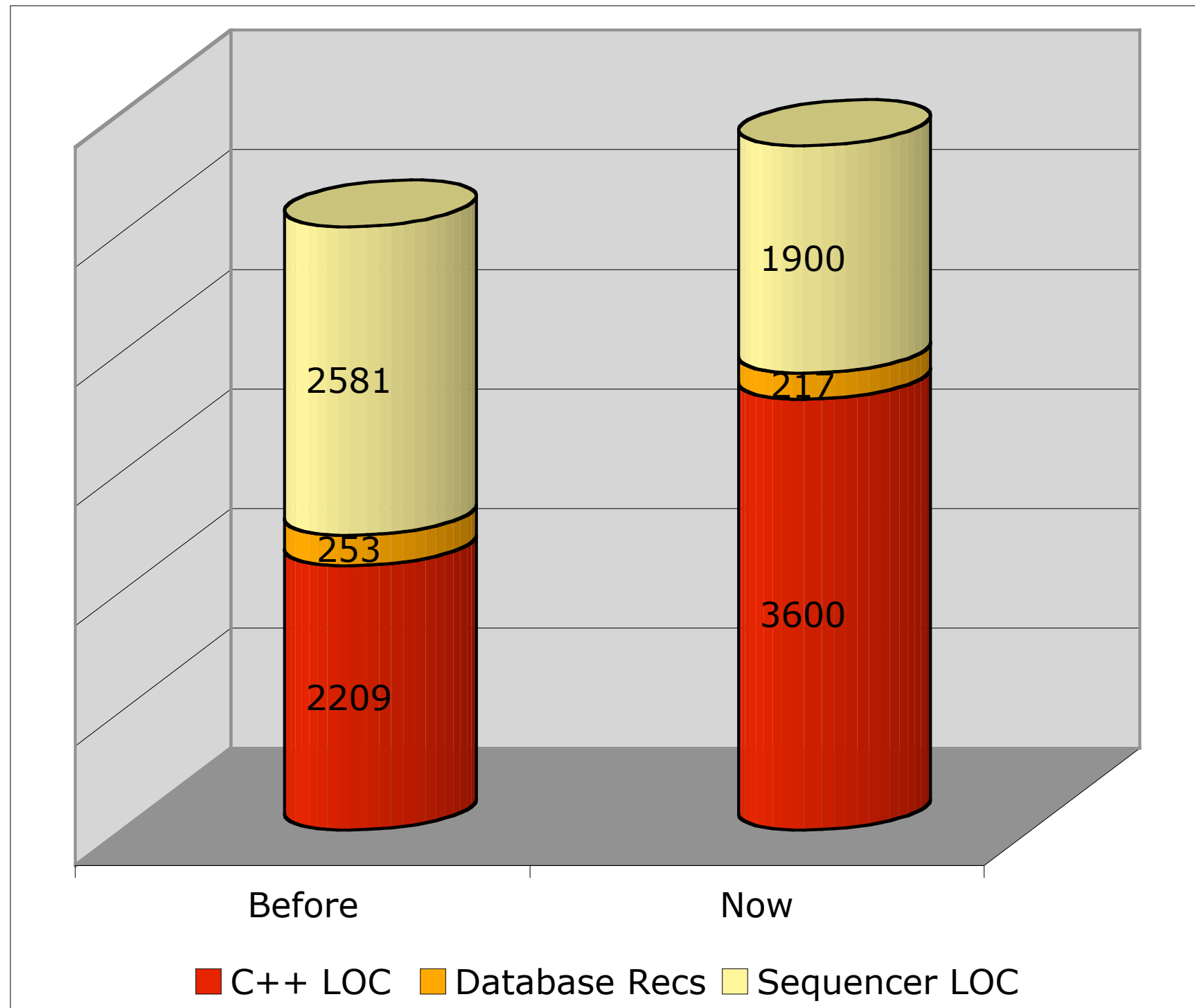
```
record(ai, "temp")
{
    SCAN "1 second"
    INP          "#C2"
    S3"
    EGU          "deg C"
    HIGH         "40.0"
    ...
}
```

Custom Low Level Code

- Custom C/C++
 - When required for higher **performance**, or interrupt service routines, low-level access to custom hardware
 - Usually **no** online changes.
 - It's really **hard** to understand, extend, debug somebody else's custom code
 - Debug tools vary with operating system
 - SNS, using vxWorks5 with Linux hosts has currently no online source-level debugger....
- **FPGA**
 - Same **problems** as custom C/C++ code
 - Probably even more so, since HDL is a "code", but often not implemented by software engineers.
 - Good simulation and offline analysis tools but online debugging limited to scope, signal analyzer.

SNS LLRF Changes in early 2007

- Improve performance of tested algorithms by converting Sequencer (State Machine) code and Database Records to C++



Alternative DSP Implementations

- High level environment
 - Commercial
 - Physics application framework
- Within control system toolkit
- Vertically integrated commercial products

Matlab Scripting

Example

Orbit correction

```
% Get the vertical orbit
```

```
Y = getam('BPMY');
```

```
% Get the Vertical response matrix from the model
```

```
Ry = getrespmat('BPMY', 'VCM');    % 120x70 matrix
```

```
% Computes the SVD of the response matrix
```

```
lvec = 1:48;
```

```
[U, S, V] = svd(Ry, 0);
```

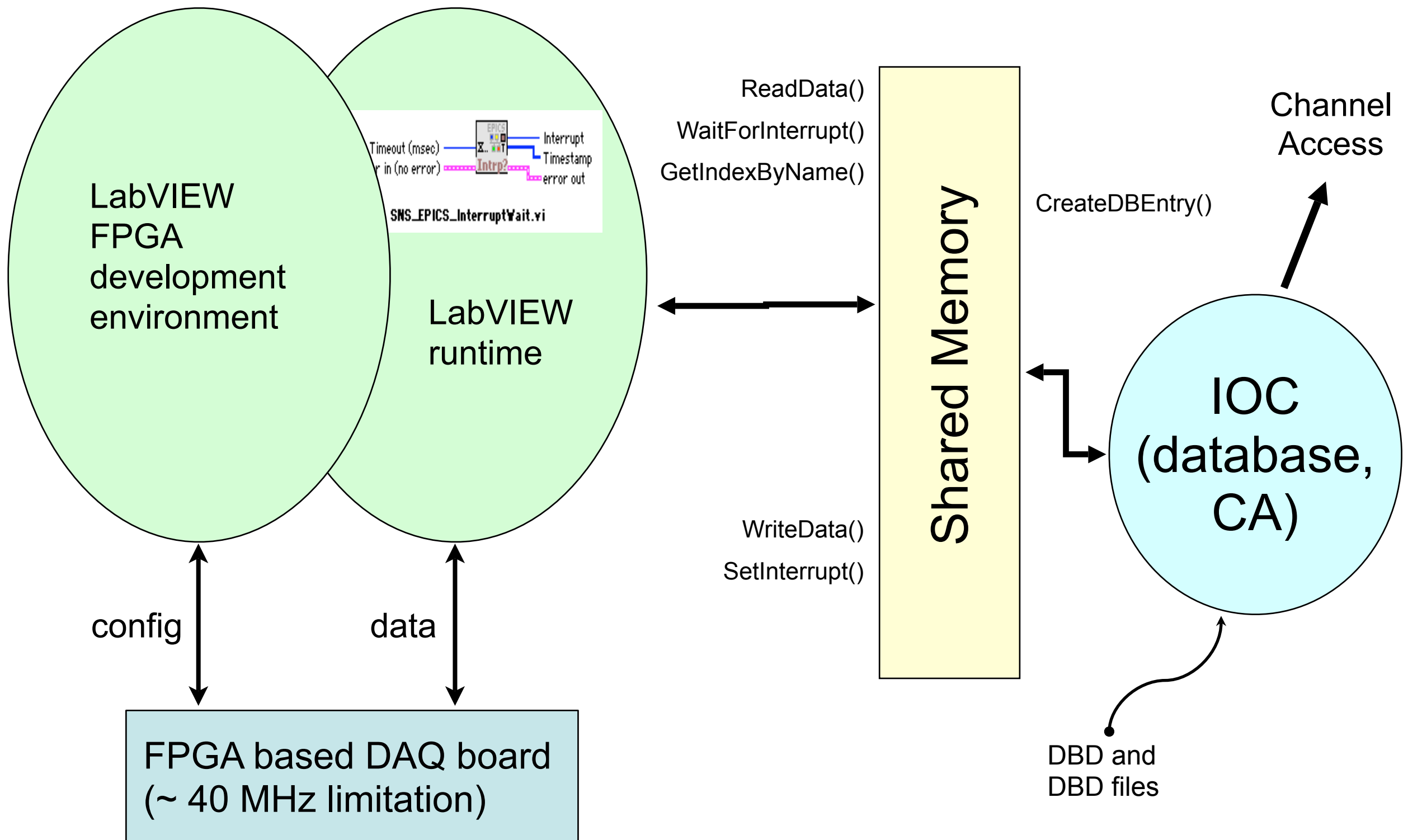
```
% Find the corrector changes use 48 singular values
```

```
DeltaAmps = -V(:,lvec) * S(lvec,lvec)^-1 * U(:,lvec)' * Y;
```

```
% Changes the corrector strengths
```

```
stepsp('VCM', DeltaAmps);
```

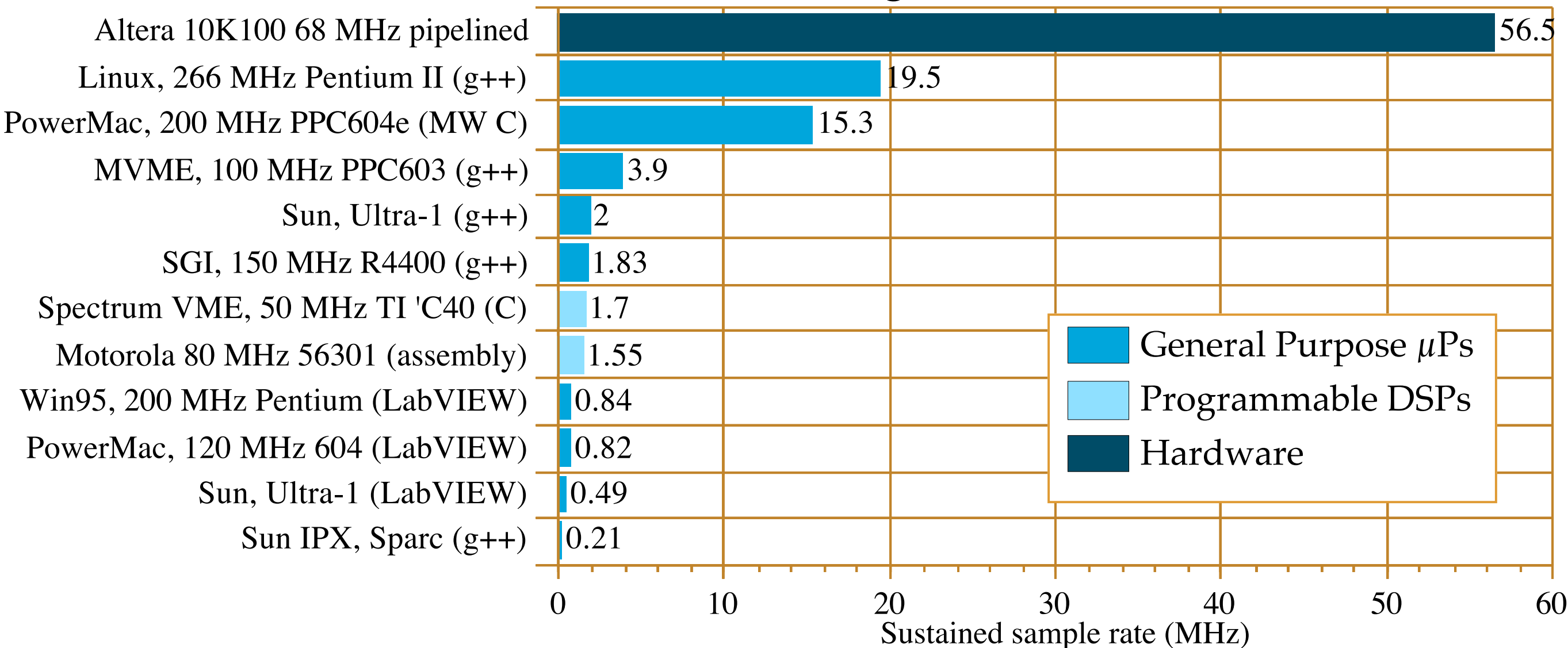
LabVIEW FPGA with EPICS



Ancient Benchmarks

10th Anniversary

RHIC BPM algorithm, circa 1997



Thank you.