LBNL Hardware, Firmware & Software

Shreeharshini Murthy, LBNL
PIP-II LLRF Final Design Review

July 17, 2024

A Partnership of:
- US/DOE
- India/DAE
- Italy/INFN
- UK/UKRI-STFC
- France/CEA, CNRS/IN2P3
- Poland/WUST

Outline

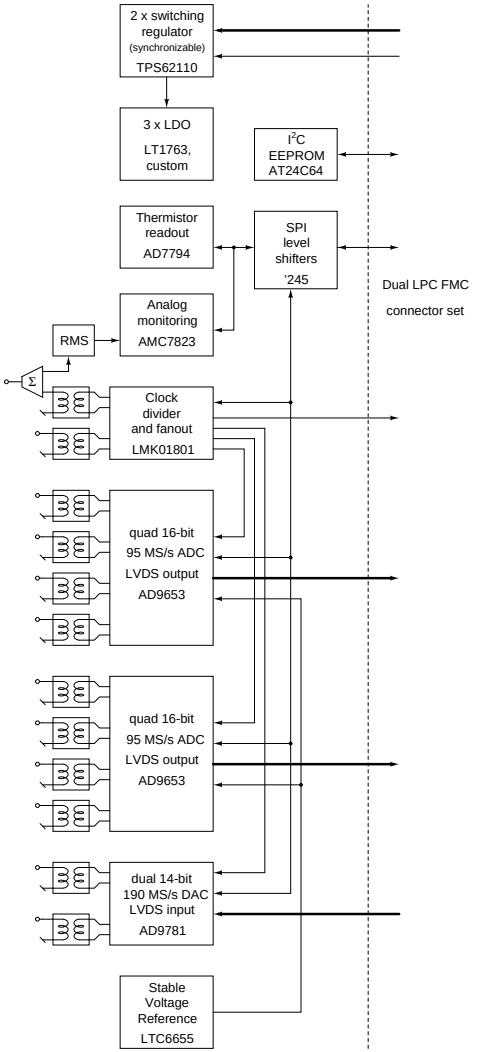
- Introduction
- Hardware
  - Digitizer board
  - FPGA carrier board
- System architecture
- Firmware - Software interface
- FPGA programming
- RFS Controller firmware
- FPGA resource utilization
- System deployment features
- Summary

Introduction

- Hardware
  - PIP-II LLRF designed around Dual FMC FPGA carrier & digitizer boards
  - Low phase noise and low crosstalk digitizer board designed in-house
  - Kintex-7 FPGA carrier board designed for accelerator field deployment
- Firmware & Software codebase
  - Heavily derived from the LCLS-II code base
  - Successfully operating almost 296 multi-cell elliptical high-β SRF cavities at SLAC
  - Real-time feedback in FPGA & system identification
  - Soft IOC (EPICS) communicating with FPGA over GbE
    - Operator Interface (OPI) already available
  - Established design practices and Continuous Integration (CI) framework for a single codebase supporting many projects and carriers
  - Functional logic written in portable synthesizable HDL
    - Vendor-specific primitives only used to access hard-silicon features
  - DSP blocks are Open Source to enable collaboration
  - https://github.com/BerkeleyLab/Bedrock

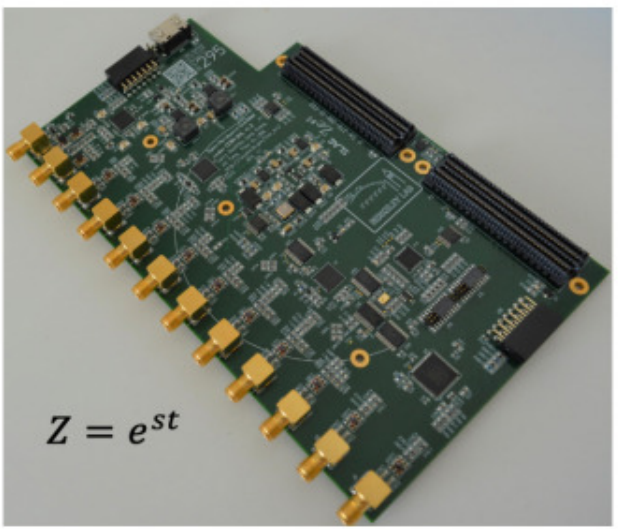
Digitizer board - Zest

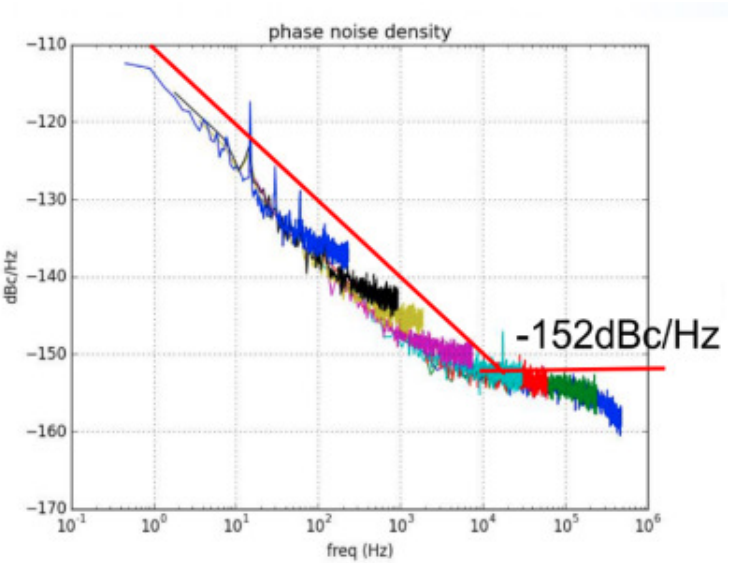
- 8× transformer-coupled inputs (ADCs) sampled at 95 MS/s
- 2× transformer-coupled outputs (DACs) sampled at 190 MS/s
- Input clock source up to 3 GHz (LMK01801, no on-board oscillator)
- Interface to FPGA via dual-LPC-FMC connectors



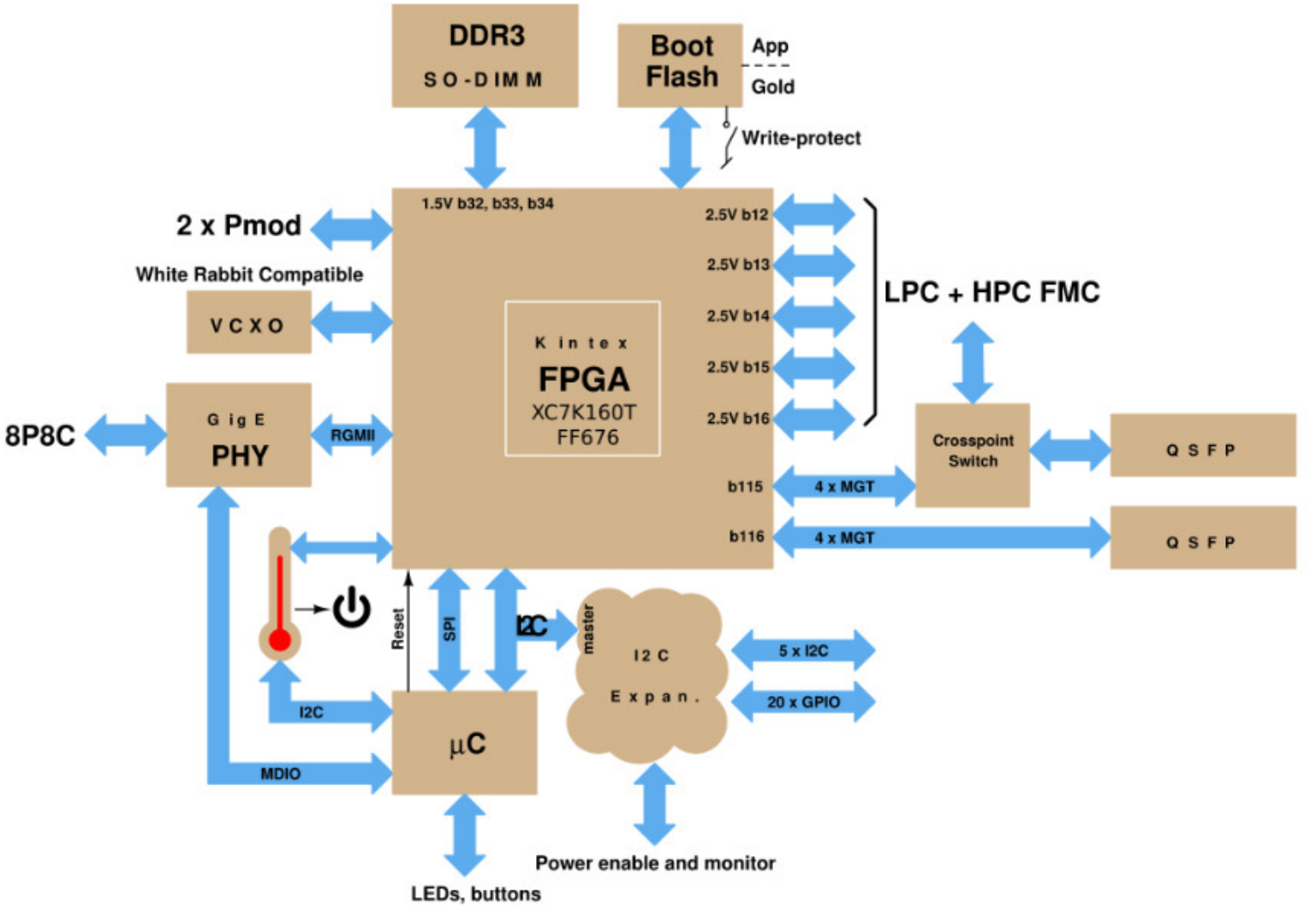
Digitizer board - Zest

- Input and output IF is at 20 MHz
- Measured broadband phase noise <-151 dBc/Hz, and the adjacent channel crosstalk is <-80 dB
- Mature design with >400 units deployed at various accelerator facilities
- Automated testing process with detailed reports enabling thorough hardware verification and quality assurance
- https://github.com/BerkeleyLab/Zest



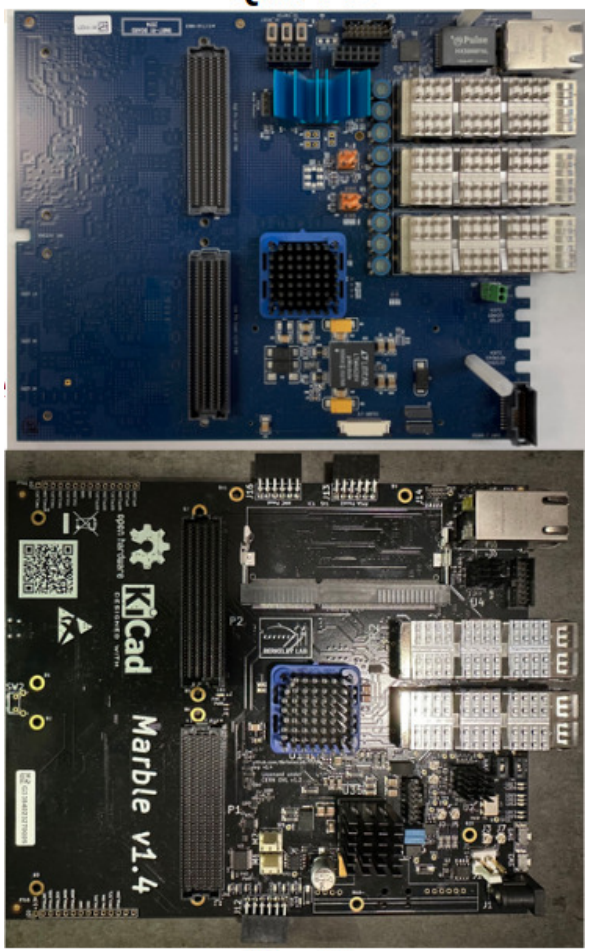


FPGA carrier board - Marble concept



FPGA carrier board

- At test stands currently deployed QF2-Pre board has become unavailable (2015 - 2022)
  - Some parts are now obsolete
  - No more hardware/firmware support from the vendor
- "Drop-in" replacement - Marble board (2020 - present)
  - Developed at Berkeley Lab in collaboration with Warsaw University of Technology
  - Long-term support and maintenance plausible
  - Open source design and open source tools (KiCad) - https://github.com/BerkeleyLab/Marble
  - Plans to deploy it at other facilities: ALS-U, LCLS-II HE, and AWA



QF2-Pre vs. Marble drop-in compatibility

- Similarities
  - Support dual LPC FMC (2 $\times$ 34 pairs), plus one HPC bank (1 $\times$ 24 pairs)
  - Powered from +6 V to +12 V at $\sim$15 W
  - Exactly same FPGA - Xilinx Kintex-7 XC7K160T-2FFG676C
  - Gigabit Ethernet over copper
  - QSFP fiber (MPO) compatible to 10 Gbaud
  - Mechanically 99% compatible, 12-layer 150 mm $\times$ 180 mm PCB
    - RJ45 connector shifts vertically a few mm
    - Some mounting holes are shifted
  - Same reconfigurable SI570 oscillator for serial links

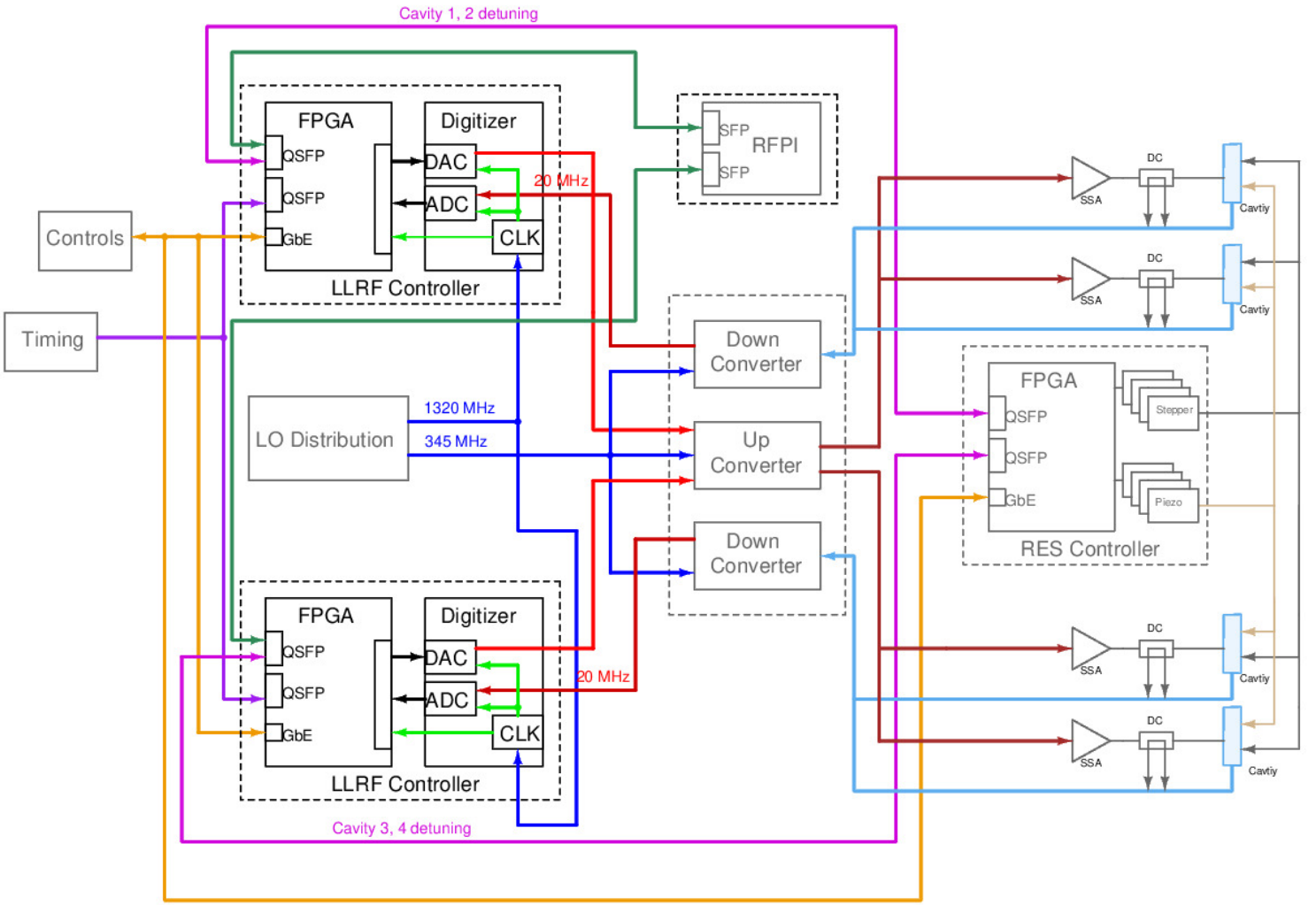
QF2-Pre vs. Marble - key differences

Features	QF2-Pre	Marble
Board management (power management, non-volatile config, etc.)	FPGA - Spartan-6 (closed-source programming)	Microcontroller - STM32 (open-source programming)
QSFP	2 Kintex + 1 Spartan	2 Kintex
System clock (management)	50 MHz	125 MHz high stability ($\pm$ 0.5 ppm) tunable
System recovery	Network boot recovery	Golden image boot from on-board Flash
Management over Ethernet	Spartan (mandatory, always available)	Kintex (using golden image)
Kintex bring-up via JTAG	via Spartan Ethernet (bespoke python tools)	via USB (openocd)
Kintex boot option from on-board Flash	Yes	Yes
DDR3 SO-DIMM	No	Yes (unused for LLRF)

Marble specifics

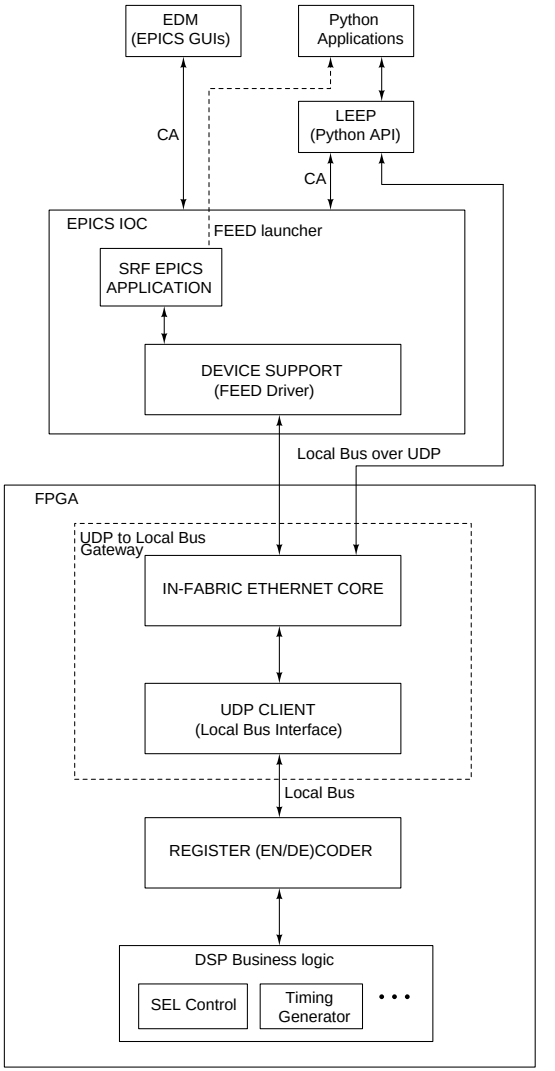
- μC codebase - https://github.com/BerkeleyLab/Marble-MMC
  - Programming features make the board useful
  - Intended to be independent of application
  - On-chip μC flash programmed using hardware debugger J-Link
  - Communication through USB UART console for configuration and testing
  - Non-volatile IP and MAC configuration
  - Communication to FPGA over SPI
  - Configure peripherals (initial settings) - SI570, over-temperature threshold...
  - Unbrickability: based on a network watchdog
- FPGA board support firmware is functional and tested
  - General-purpose I2C gateway for voltage and current monitoring
  - General-purpose Mailbox connects μC features from SPI to network
- Most EPICS support already written and tested; basic board monitoring (temperature, voltage) in-place

System architecture



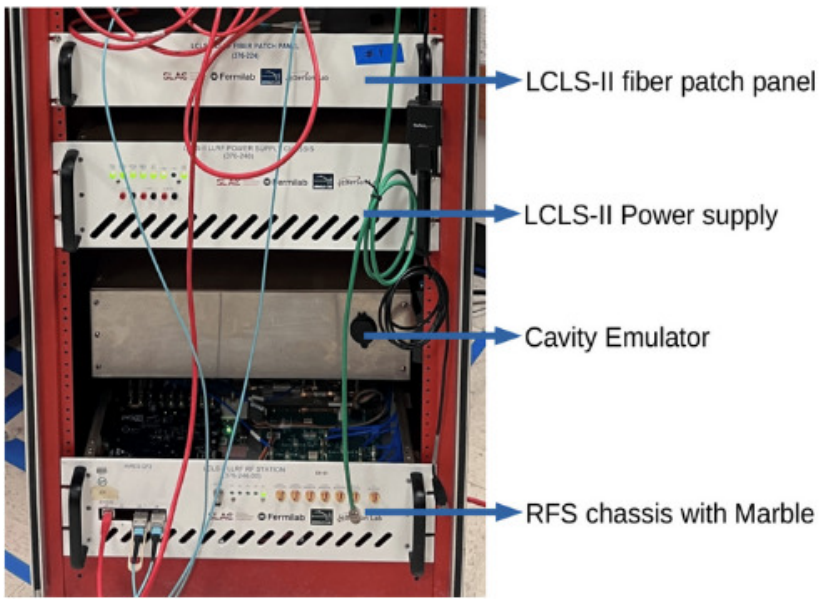
Firmware - Software interface

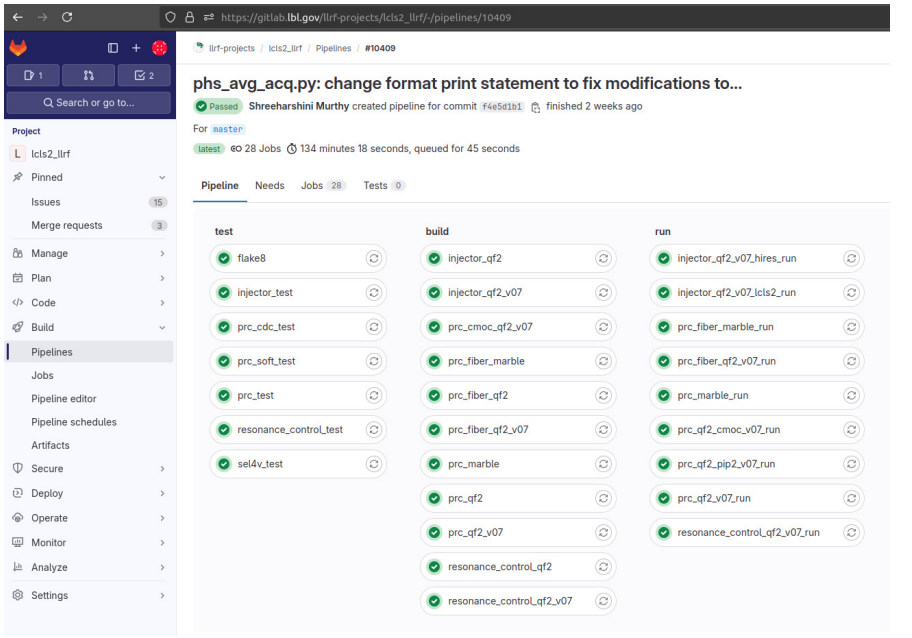
- FPGA Embedded Ethernet Driver (FEED) - C++ library for IOC to communicate FPGA over GbE
- LBNL Embedded Ethernet Protocol (LEEP) - python interface module
  - Used to communicate to the FPGA either directly **or** mediated by an EPICS IOC using the Channel Access (CA) protocol
- Localbus over UDP to access, manipulate registers and capture waveforms
- FEED launcher lets IOC start python jobs



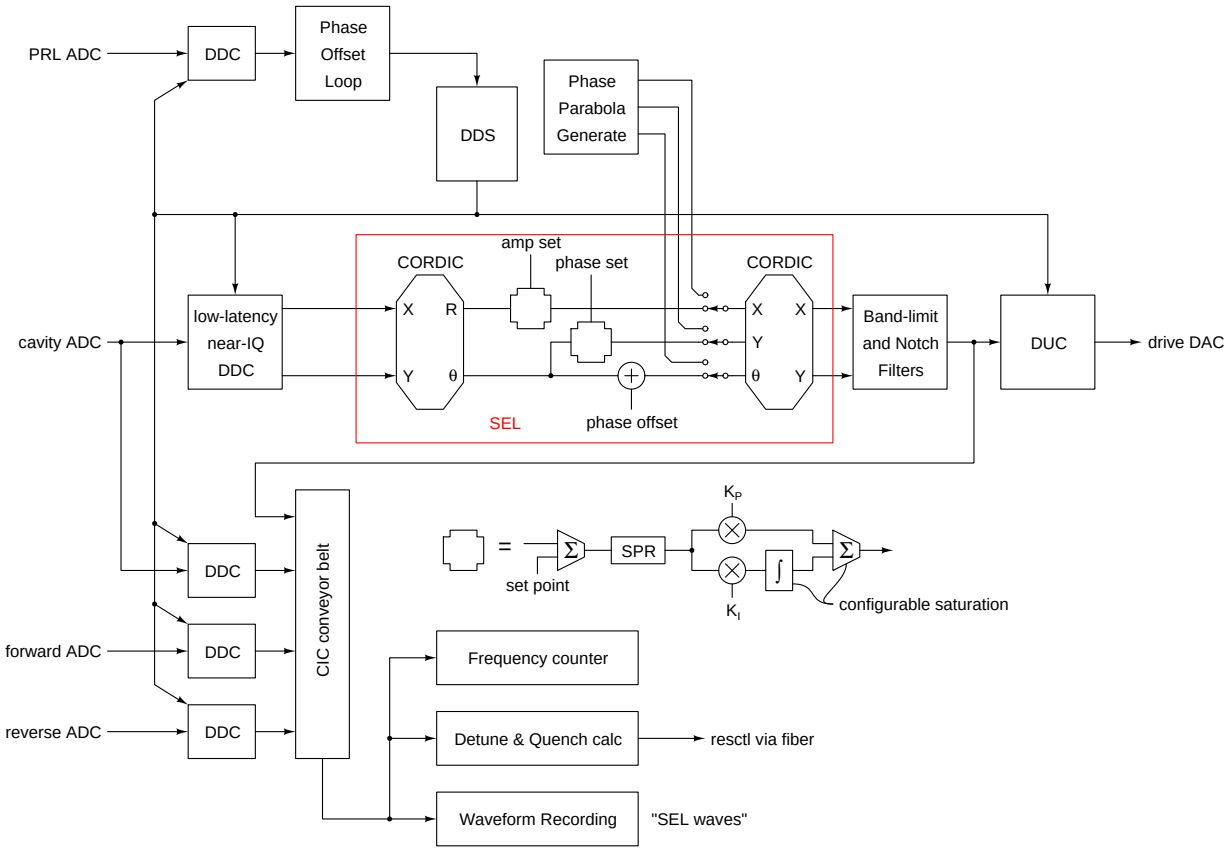
RFS Firmware (FPGA programming)

- Application layer (actual LLRF control) identical to QF2-Pre
- Board support layer supports both Marble & QF2-Pre
- Rack checkout process mostly the same
- Single RFS chassis with Marble and a cavity emulator deployed at LBL for Continuous Integration (CI)





RFS Controller Firmware - SEL

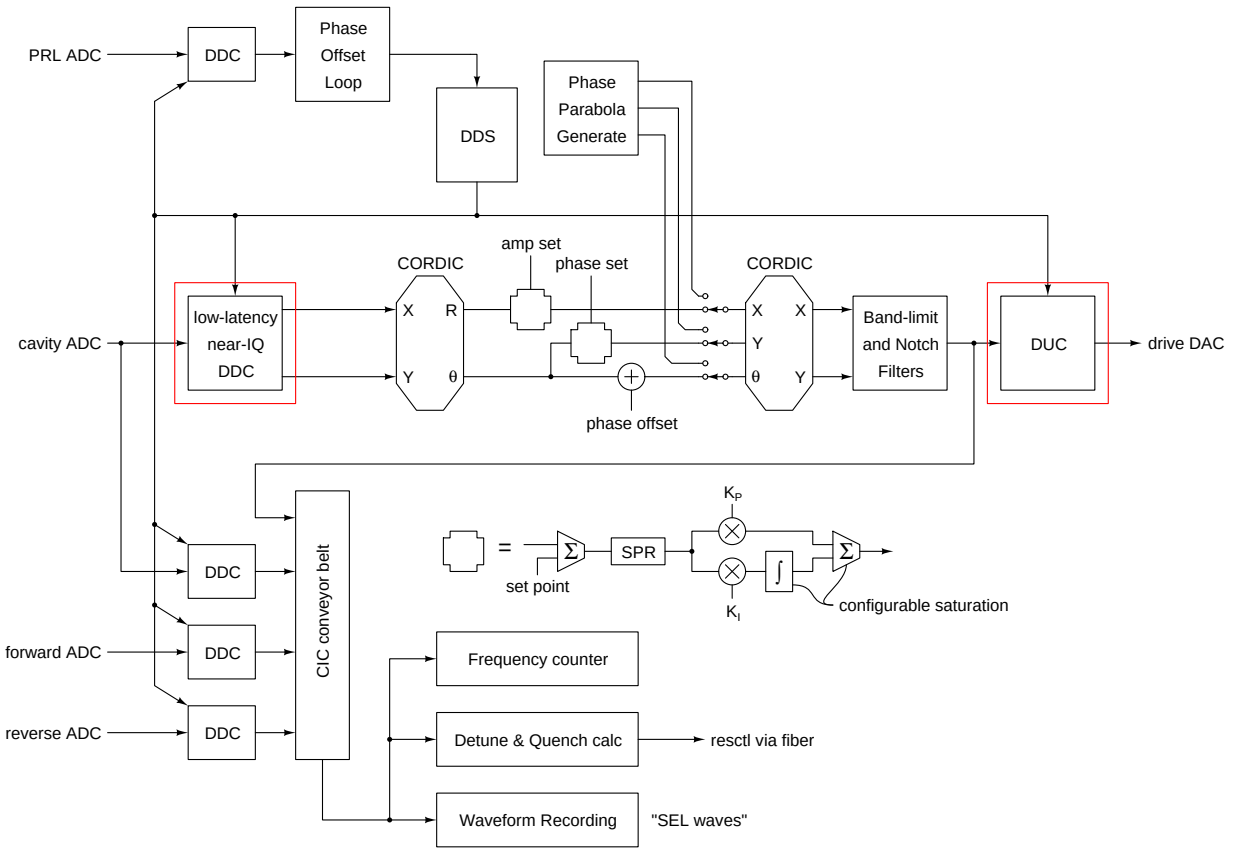


Digital Self-Excited Loop (SEL) implemented in firmware with CORDICs. As presented to BARC-PIP-II collaboration earlier this year

Related to Delayen's 1978 analog cavity controller

Textbook oscillator with amplitude and phase stabilization

RFS Controller Firmware - Digital Converters

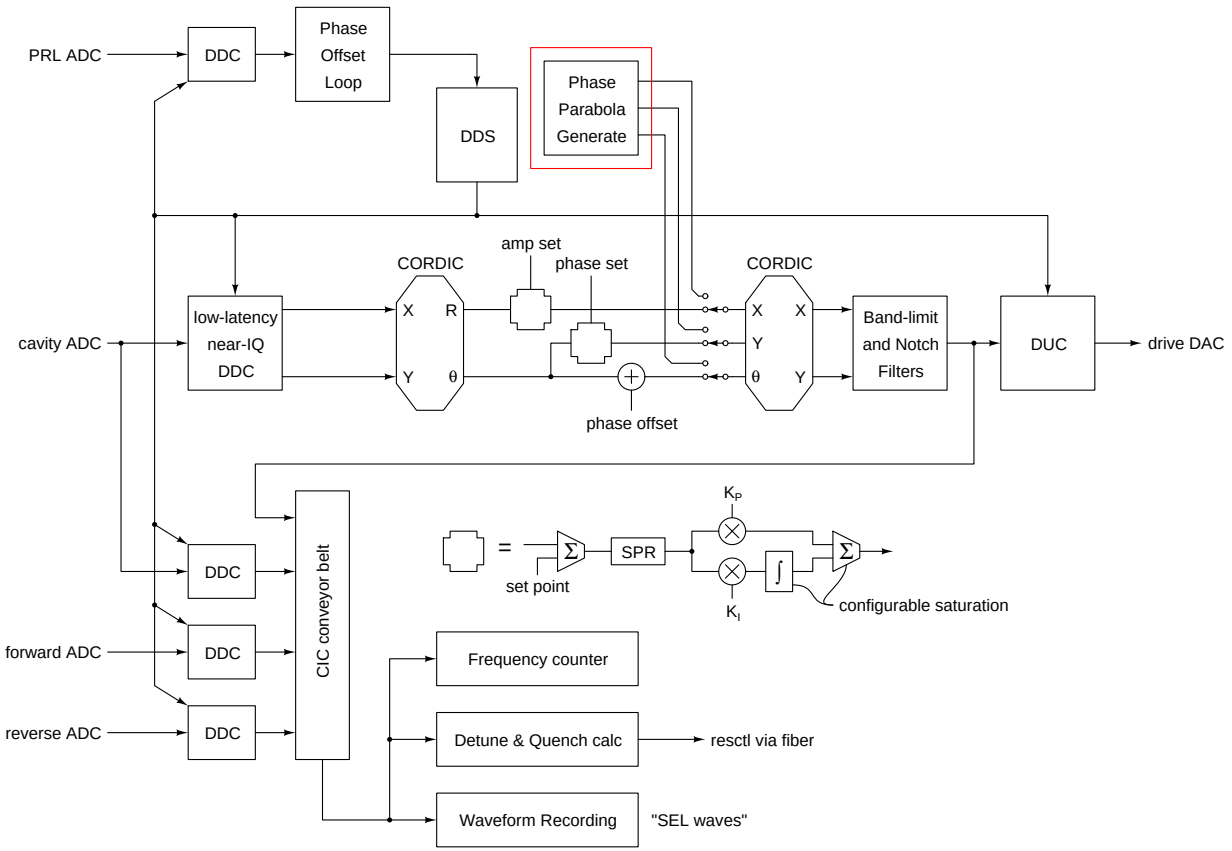


Core conversion of analog SEL to DSP, using separate down and up converters

Near-IQ sampling is used so DSP can reject analog distortion products. A low-latency Digital Down Converter (DDC) converts that to I and Q for further processing

Digital Up Converter (DUC) is used to convert from baseband to IF frequency

RFS Controller Firmware - Chirp

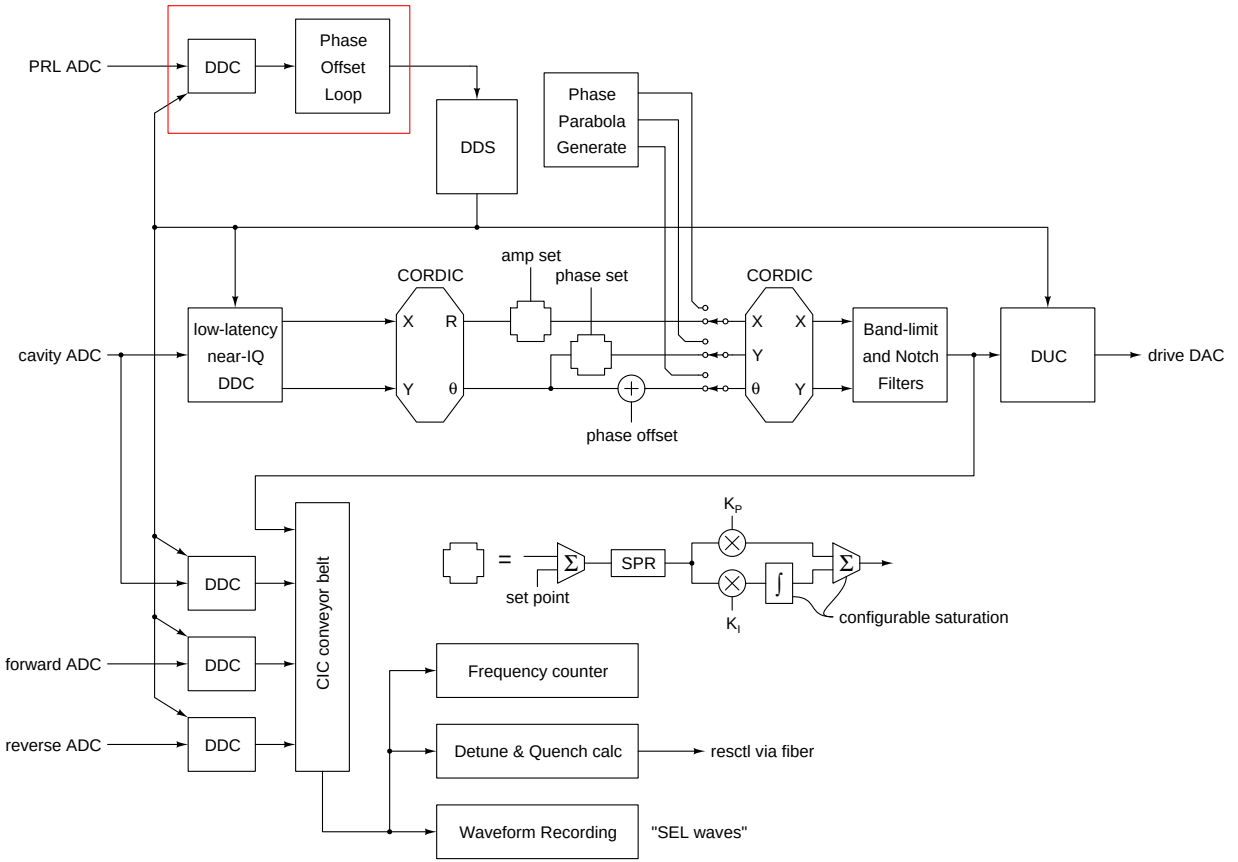


A simple parabola generator with CORDIC is used for generating chirps to find resonances

This feature is used after a cool down to quickly find the π mode, and also to systematically characterize nearby passband mode(s)

These frequency measurements are key to setting up the notch filters that keep the system stable at high proportional gains

RFS Controller Firmware - PRL

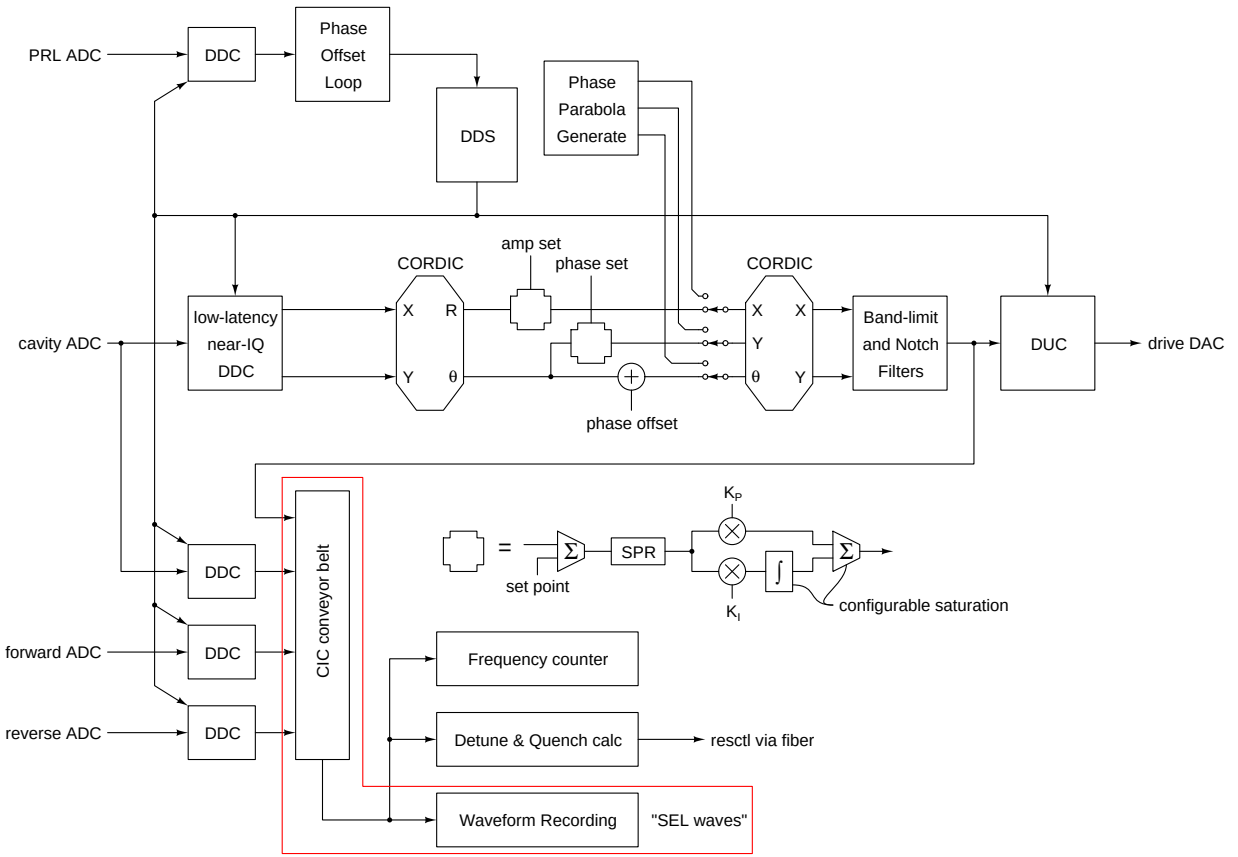


Digitize the phase reference line

Without a phase reference, one can test a cavity but can't accelerate beam

Physically lock the phase of the DDS, so a cavity signal with matching phase measures as zero phase

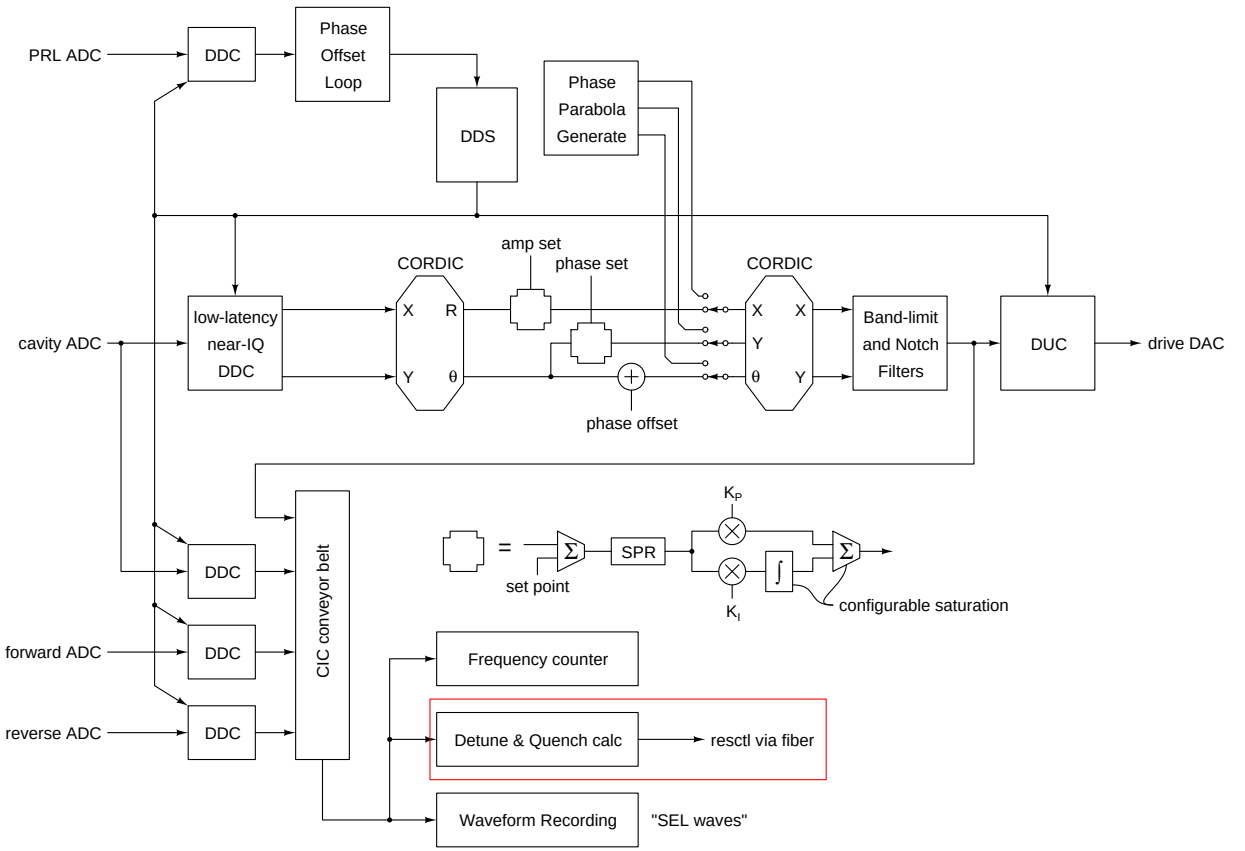
RFS Controller Firmware - Waveforms



A conveyor belt style design is used to capture waveforms. The "I" (Integrator) part of the CIC has one instance per input channel. The "C" (Comb) part and a half-band filter is a shared datapath used by all channels

Includes programmable decimation and channel selection into a fixed-size circular buffer, with fault capture capability

RFS Controller Firmware - Detune



"Detune & Quench calc" is a block that finds

$$a = \frac{1}{\vec{V}} \cdot \left[\frac{d\vec{V}}{dt} - b\vec{K} \right]$$

$$P_{\text{diss}} = |\vec{K}|^2 - |\vec{R}|^2 - \frac{dU}{dt}$$

every 11.2 μs, given measurements of $\vec{V}$, $\vec{K}$, and $\vec{R}$

Its implementation lies in between traditional FPGA DSP design techniques and a general-purpose soft core

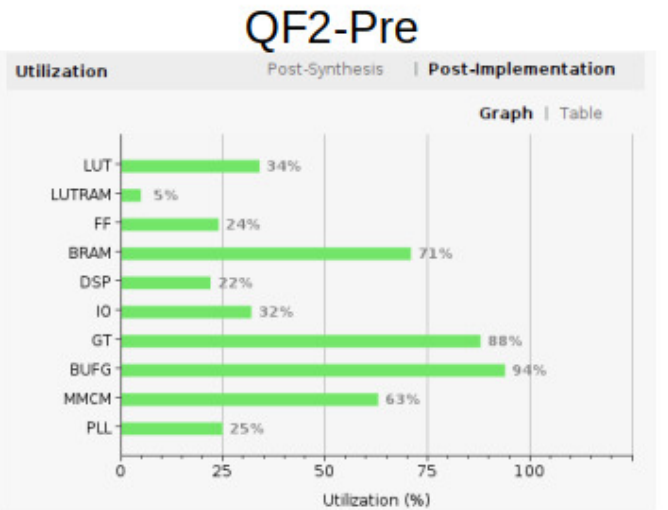
ChitChat - Communication protocol

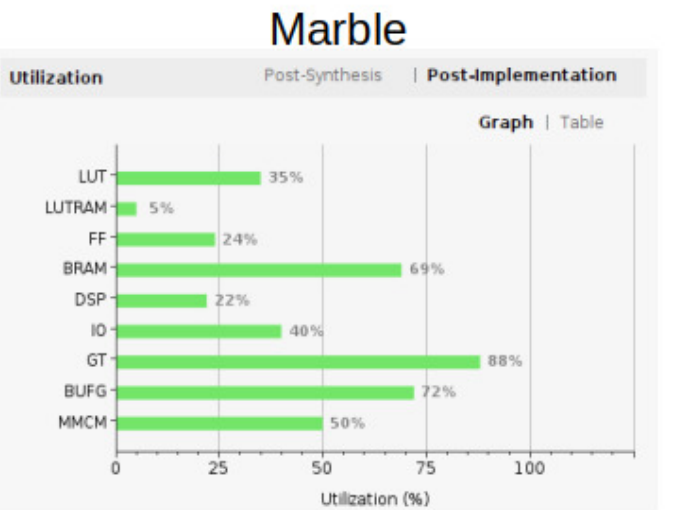
- To transmit real-time data between chassis (RFS, RES, RFPI) over fiber link with a repeat rate of 11.36 MHz
- This includes cavity detune frequency, detune valid status (to RES), and interlock status (proposed: from RFPI)
- Data integrity verified with Cyclic Redundancy Check (CRC)
- Auxiliary information such as frame numbers, protocol identifiers, CRC faults and loop-back latency are also transmitted
- Each packet consists of 11× 16-bit words, 4 of which contain the 64-bit of payload

Word	
0	PROTOCOL_CAT[3:0], PROTOCOL_VER[3:0], COMMA[7:0]
1	GATEWARE_TYPE[2:0], TX_LOCATION[2:0], RESERVED[9:0]
2	REVISION_ID[31:16]
3	REVISION_ID[15:0]
4	TX_DATA0[31:16]
5	TX_DATA0[15:0]
6	TX_DATA1[31:16]
7	TX_DATA1[15:0]
8	TX_FRAME_COUNT[15:0]
9	TX_LOOPBACK_FRAME_COUNT[15:0]
10	CRC_CHECKSUM[15:0]

FPGA resource utilization

- Controller firmware takes 7065 LUTs, 35 DSP blocks, 8 BRAM tiles, and could clock at up to 230 MHz in the (13-year-old, 28 nm) Kintex-7
- That's about 7.4% of the 7K160T on a Marble board
- About a third of the LUT footprint comes from two (DDS and SEL) CORDIC blocks: 22-bit datapath $\times$ 20 stages $\times$ (x, y, θ), for 1320 LUTs
- Resource utilization for both the application (controller) + board support firmware is reasonable and nearly identical to QF2-Pre

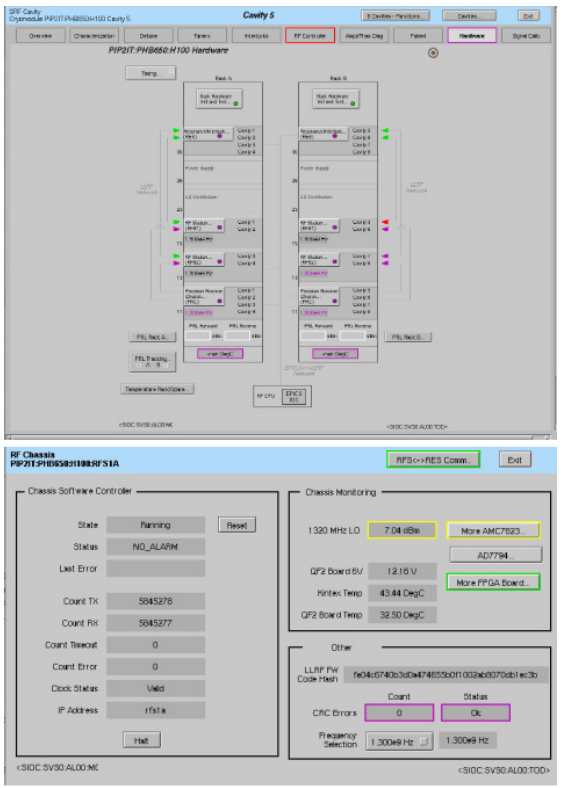




System deployment features

Hardware - automatic rack checkout process

- FPGA bitfile programming
- Marble/Zest version and Serial number reporting
- Digitizer initialization - ADC/DAC data integrity tests
- DSP register initial value setting

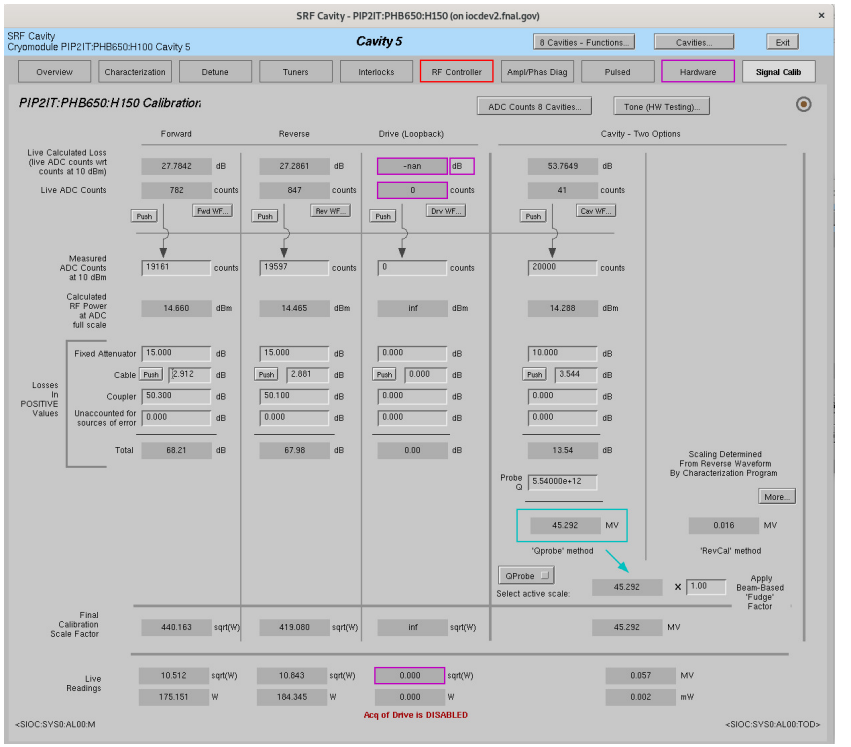


Signal calibration

- Basic RF channel calibration (fwd, rev, cav) based on coupling, cable losses, and chassis cal. Turns raw ADC values to $\sqrt{\mathrm{W}}$ or MV

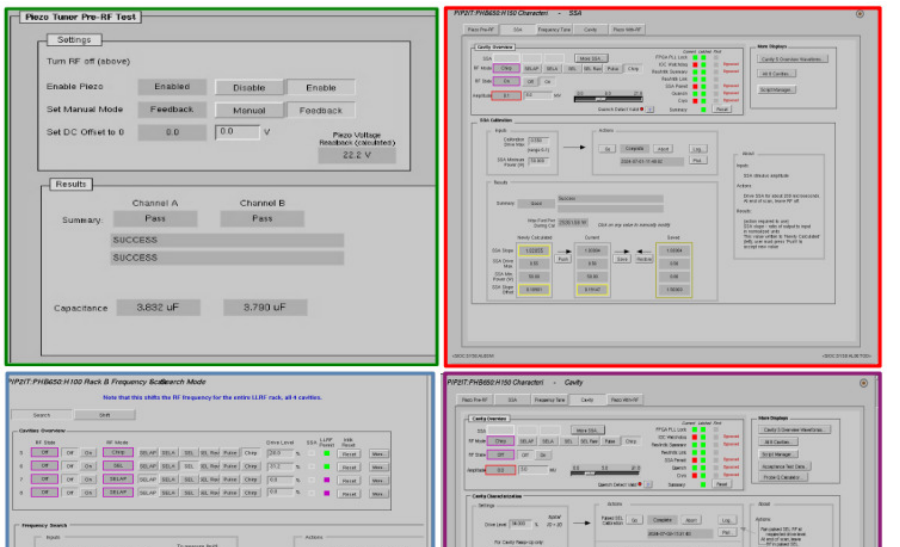
- Cavity Q_{probe} notoriously difficult to set and measure, and can't be re-measured once the cryomodule is assembled

- System re-calibrates the cavity probe channel *in-situ* based on reverse channel decay waveforms (as mentioned in previous talk)



System characterization

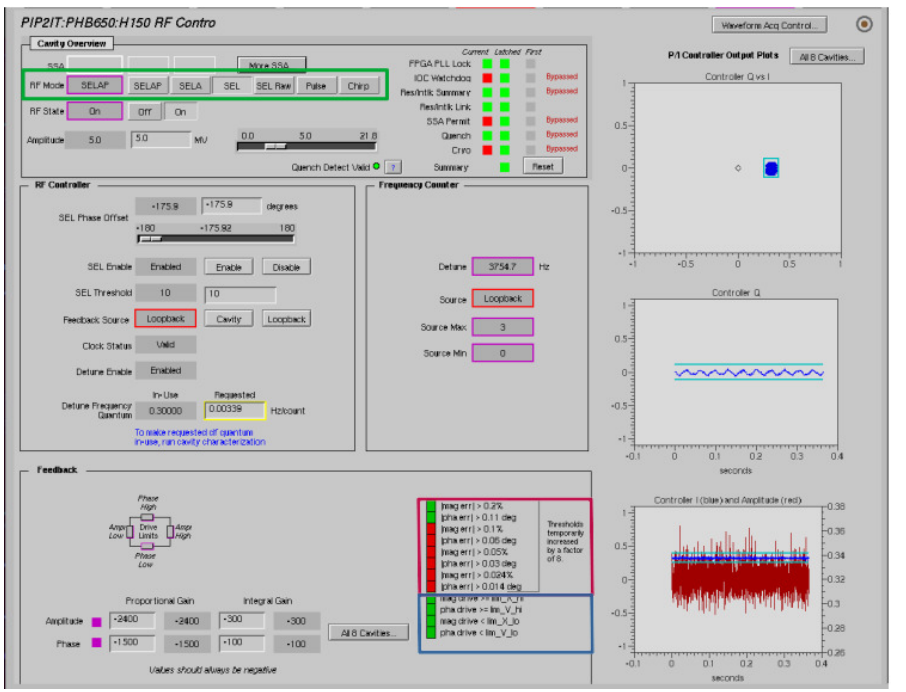
- Piezo Pre-RF - capacitance measurement (RES)
- SSA - slope, minimum power, drive maximum
- Frequency Tune - electrical resonance modes (chirp), detune (tuner)
- Cavity - SEL phase offset, plant gain, detune coefficients, Q_L using pulsed SEL calibration
- Piezo with RF - Hz/V over 2 Hz to 20 Hz (RES)



System deployment features

Field Control

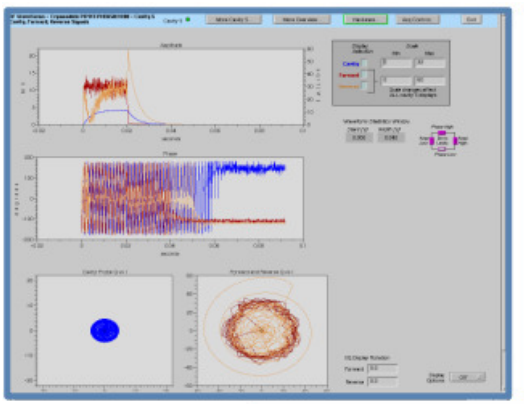
- Chirp, Pulsed, SEL raw, SEL, SELA, SELAP modes
- Operation modes are created in software. Firmware is run-time configurable to support each of these modes, but that's more quantitative than qualitative
- Amplitude and phase errors
- Drive limits
- Automated PI gain tune - based on cavity BW, amplitude, and plant gain

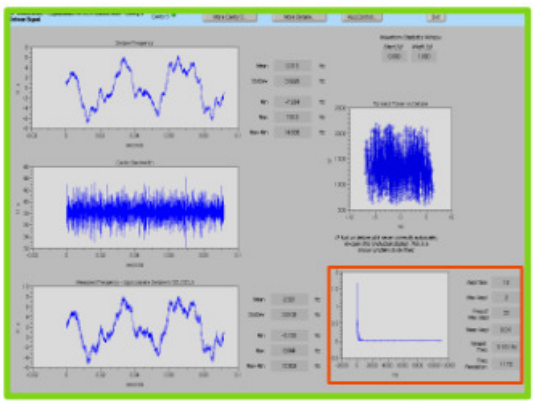


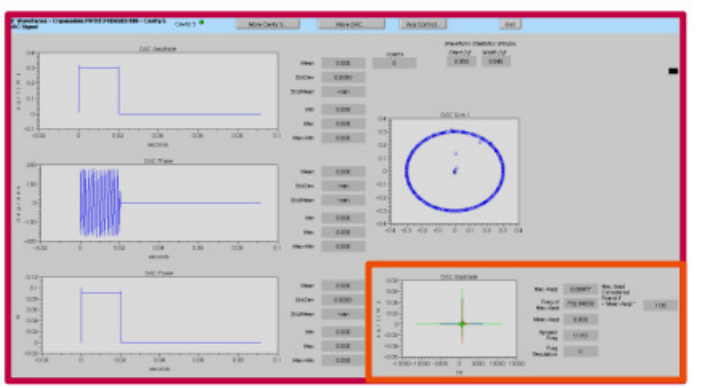
System deployment features

Waveforms

- Cavity, forward, reverse, DAC
- Cavity bandwidth and detune
- FFTs
- PRL (support present but not yet deployed)







Summary

- Digitizer board: Zest
  - Designed to meet stringent LLRF requirements at 20 MHz IF
- FPGA board: Dual FMC carrier Marble board
  - Long-term hardware/firmware/software support plausible
  - Demonstrated to meet DSP and interfacing needs
  - Has been mass produced (∼450)
- PIP-II LLRF firmware/software codebase is mature: heavily derived from the LCLS-II LLRF system
- Features like seamless GDR-SEL transitions, automated system characterization, cavity bring-up and tight firmware/software integration facilitate the operation of a complex system
- Design and development framework (CI) make the codebase reusable/shareable across projects

THANK YOU!