

Low Cost, High-Density Digital Electronics for Nuclear Physics

Wojtek Skulski

Principal Investigator

DE-SC0009543

SBIR Exchange, August 14, 2020, 5:05pm



- The company and its capabilities.
 - Customers.
- Hardware progress.
 - New compact instrument FemtoDAQ+.
- Firmware progress.
 - Improved firmware structure.
 - We organized the FW into *Features*.
- Software progress.
 - Portable remote Web GUI.
 - JavaScript and JSON.
- Plans.
- Highlights.
- Acknowledgements.

- The team: two physicists, a senior software engineer, a part time engineering associate, and a manager. We regularly work with a local EE consultant.
- We worked with several interns listed on the Acknowledgements page.

Our focus:

Digital data acquisition (DAQ) for nuclear physics, high energy physics, DM search, etc.

Our capabilities: Anything, what we need to develop a cutting edge instrument.

- Electronic design.
- Firmware development for Field Programmable Gate Arrays (FPGA).
- Software development for embedded processors, especially Embedded Linux.
- Algorithms for pulse processing.
- Algorithm implementation in the FPGA (VHDL, Verilog) and in embedded processors (Pascal, Python, C).
- Processing data from nuclear detectors of any kind.
- Development of simple detector assemblies using scintillators, PMTs, or SiPMs.



**National Superconducting
Cyclotron Laboratory**



Brown University

Progress

Hardware progress: 2-channel **FemtoDAQ+** is replacing the former FemtoDAQ.

- **More powerful FPGA:** Spartan-6 LX9 replaced with Artix-7.
- **Longer waveforms, faster readout.**
- **Two analog reconstruction channels** can synthesize the signal, possibly after digital filtering, or output an arbitrary analog waveform.
- BeagleBone is **replaced** with MicroBone Single Board Computer, which is also our product.
- We are considering whether the old FemtoDAQ should not become the “last time buy”.

Firmware progress.

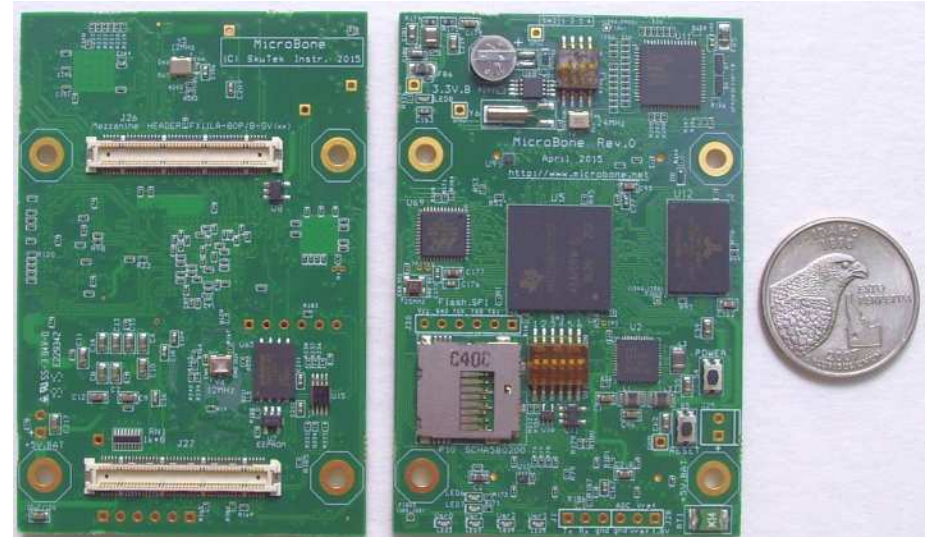
- **Unified interface** between the FPGA and the Linux MicroBone SBC.

Software progress.

- **Web-based GUI** for setup and control works with any operating system using the web browser technology.

- After introducing FemtoDAQ+, **all our products** will now use MicroBone SBC.
- ARM Linux System on Module (SOM) with a low-power 1 GHz ARM processor AM5338.
- It provides embedded Linux with **laptop-class performance** to all our products.
 - Local data logging, remote display for monitoring and diagnostic.
- Hardware:
 - We added an 8-channel ADC/DAC chip with 12 bit resolution and 5 Volts input / output range.
- Software:
 - Adopted Debian-10 for this SBC.
 - Developed both Python and C capabilities: SBC is its own development system.

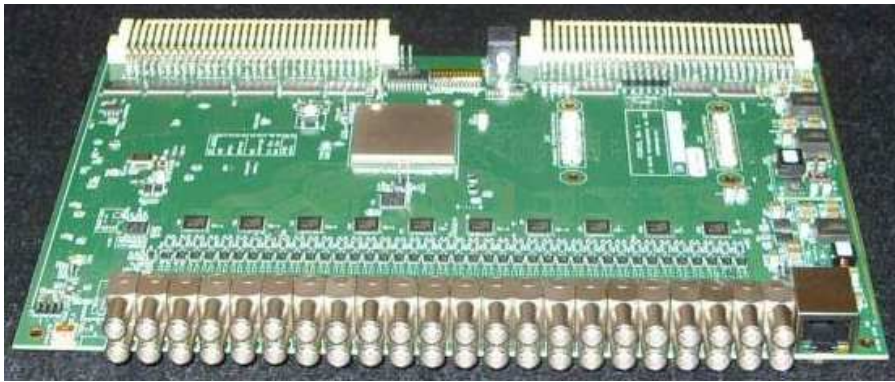
- Easy to embed 2" by 3" size, fits within DAQ module.
- Two 80-pin expansion connectors.
 - Memory interface, USB, GbE, SPI, I²C.
- 512 megabytes of RAM.
- 1 GHz ARM AM5338 running Linux.
- Two Programmable Real Time processor cores.
- μ SD card acting as Solid State Disk, up to 64 GB.
- Eight analog pins: 12-bit ADC or 12-bit DAC each.
- Optional Real Time Clock with battery backup.



Two Markets: Research DAQ & Small DAQ

Our digitizers are using **MicroBone Linux SBC** and **three different FPGA families** to optimize their cost to performance balance. The left column is for research DAQ. The right column is for small applications.

High density digitizer: **40 channels**
Kintex-7
Ethernet readout @ a few MB/s



Mid density digitizer: **10 channels**
Spartan-6, remote Web interface



High density digitizer: **32 channels**
Kintex-7
1G Ethernet at wire speed
10G under development

It was presented yesterday.

Low density digitizer: **2 channels**
Artix-7, remote Web interface



- **Replace** the BeagleBone with the MicroBone developed by SkuTek.
- **Increase** the waveform memory from 64 kB to 2 MB. (Waveforms up to 500 ms @ 100 MSPS.)
- **Upgrade** Spartan-6 to Artix-7 providing more resources and better performance.

Two outputs:
arbitrary wave
generator

Two inputs with termination: 50Ω, 1kΩ, 10kΩ

Logic I/O, 8 single pins
and 6 coaxial MMCX

Artix-7

ADC 14/16 bits
14 or 16 bits @ 100 MHz
14 bits @ 250 MHz

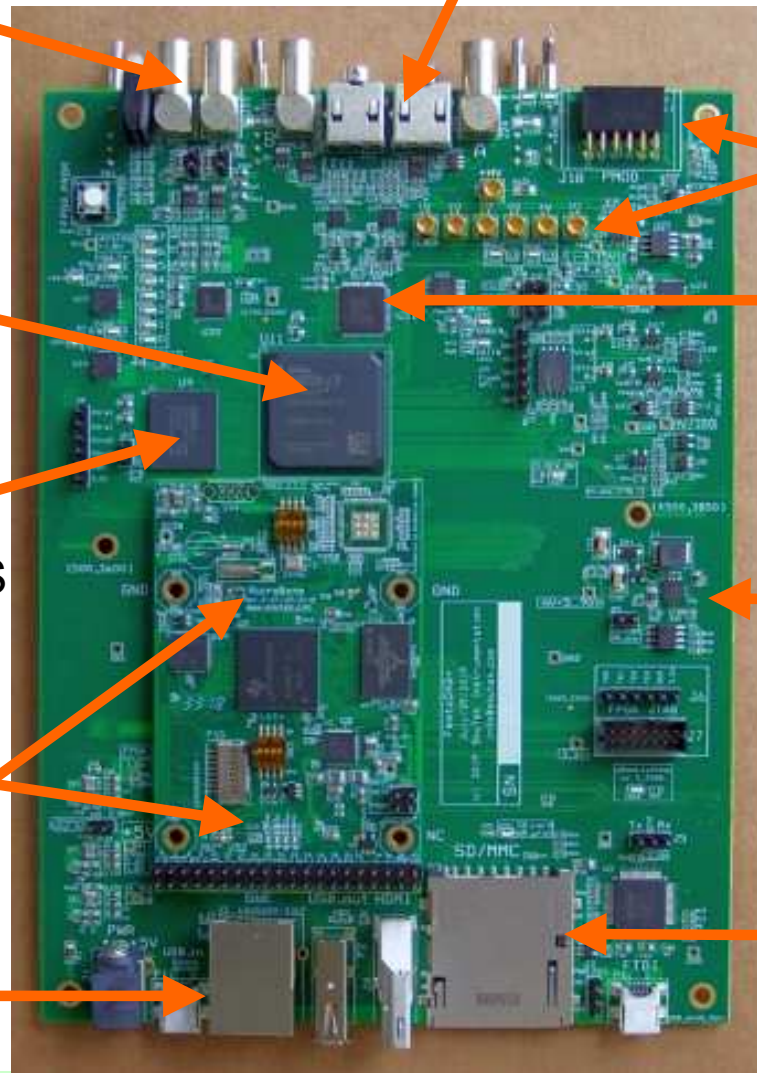
RAM space
for 1 million samples
2 ch * 0.5 s @ 100 MSPS

SiPM bias generator
+5V to +90V
in 1.4 mV steps

MicroBone Linux
Single Board Computer
Running Debian-10

SD card

GbE and USB-2



- The prototypes are ready to go!
- We are now working on the remote Web GUI (next section).

80 μ s waveforms

Half a second waveforms



Previous FemtoDAQ
with BeagleBone



SiPM bias generator
+5V to +90V
in 1.4 mV steps

Logic I/O
coax

Two outputs:
arbitrary wave
generators



Logic I/O, 8 pins
Arranged as PMOD

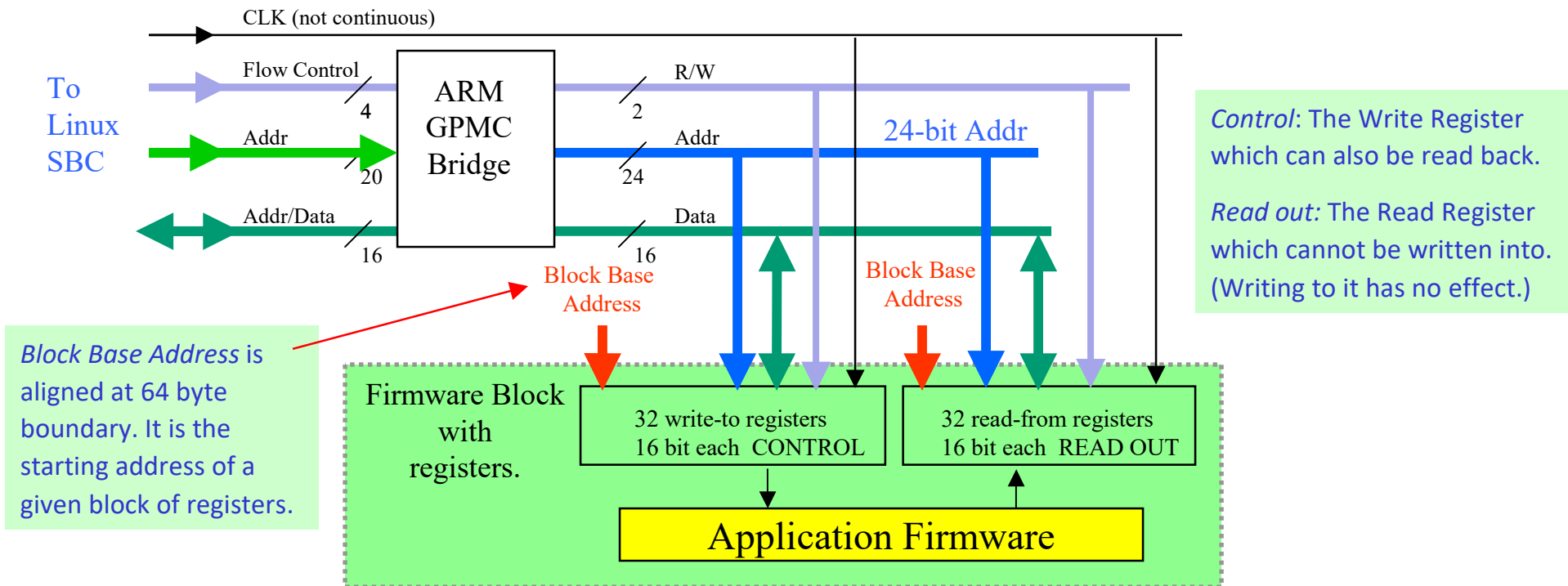
Two inputs: 50 Ω , 1k Ω , 10k Ω
14 or 16 bits, 100 MSPS

Firmware and Software Progress

We organize the FW and SW into the *Features* offered by the instrument to the user.

The *Instrument Feature* consists of Application Logic, the register interface, SBC software, and remote control.

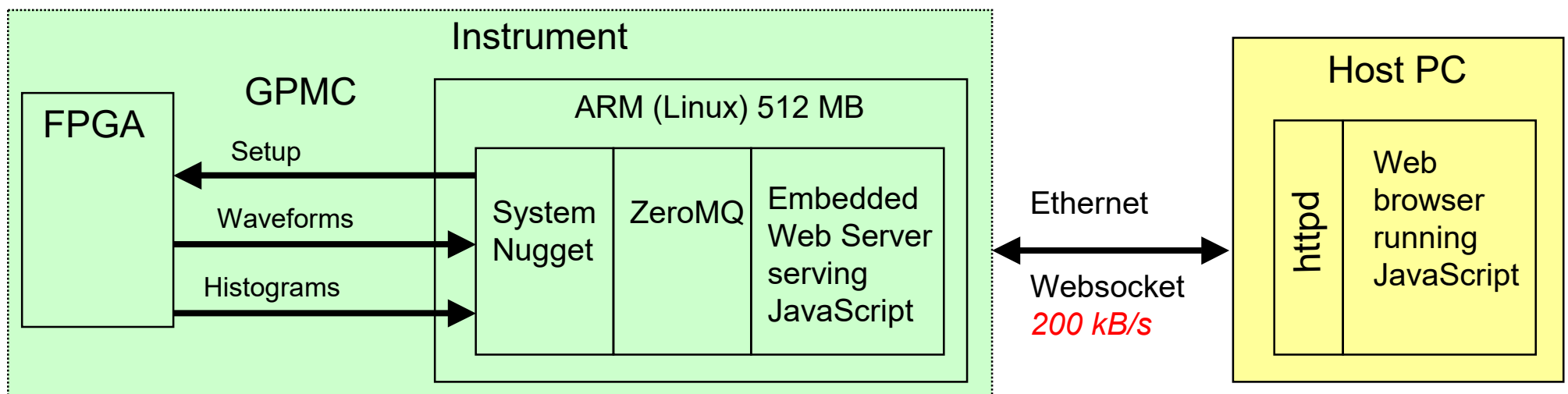
- Internally in the FPGA, the *Feature* is a Firmware Module with Registers for control and read out.
- Registers have *addresses* in the Linux memory space.
- The *Base Address* defines where the registers start in the Linux memory space.
- The *Feature* is interfaced to Linux SBC using the ARM General Purpose Memory Controller (GPMC).
- GPMC is a memory bus between the SBC and the FPGA.



Block Diagram of the Control Software

The software runs **half / half** on the Linux SBC and on the remote host PC.

- **System Nugget** is driving the interface. It is written in C (*about 3,000 lines total*).
- **ZeroMQ** server communicates between the Nugget and the Embedded Web Server.
 - *ZeroMQ provides reliable inter-process point-to-point communication.*
- **Embedded Web Server** is serving the JavaScript to the remote browser (*also about 3000 lines*).
- It also sends out the waveforms and the histograms.
- The browser executes the **JavaScript** and presents the plots to the user.
- The transfer works at ~ 200 kB/s, sufficient for remote display and recording a sample event stream.
- **Higher speed** ~ 10 MB/s is available while writing to the local SD card or to the NFS mounted disk.



Setting up the *Feature* parameters such as trigger thresholds, energy integration, etc.

- The parameters are written into the *Feature* registers.
- The parameters are passed from this remote host GUI to the SBC, using **JSON** protocol.
- The setup can be saved to disk.

The screenshot displays the DDC-10 Setup web interface in a browser window. The address bar shows 'demo.skutek.co/setup.html'. The interface includes a navigation menu with 'Setup', 'Waveform', and 'Histogram' tabs, and a 'Light' indicator. A status bar at the top right shows 'Reserve', 'uBone01', and 'READY TO RUN' buttons.

Global Settings

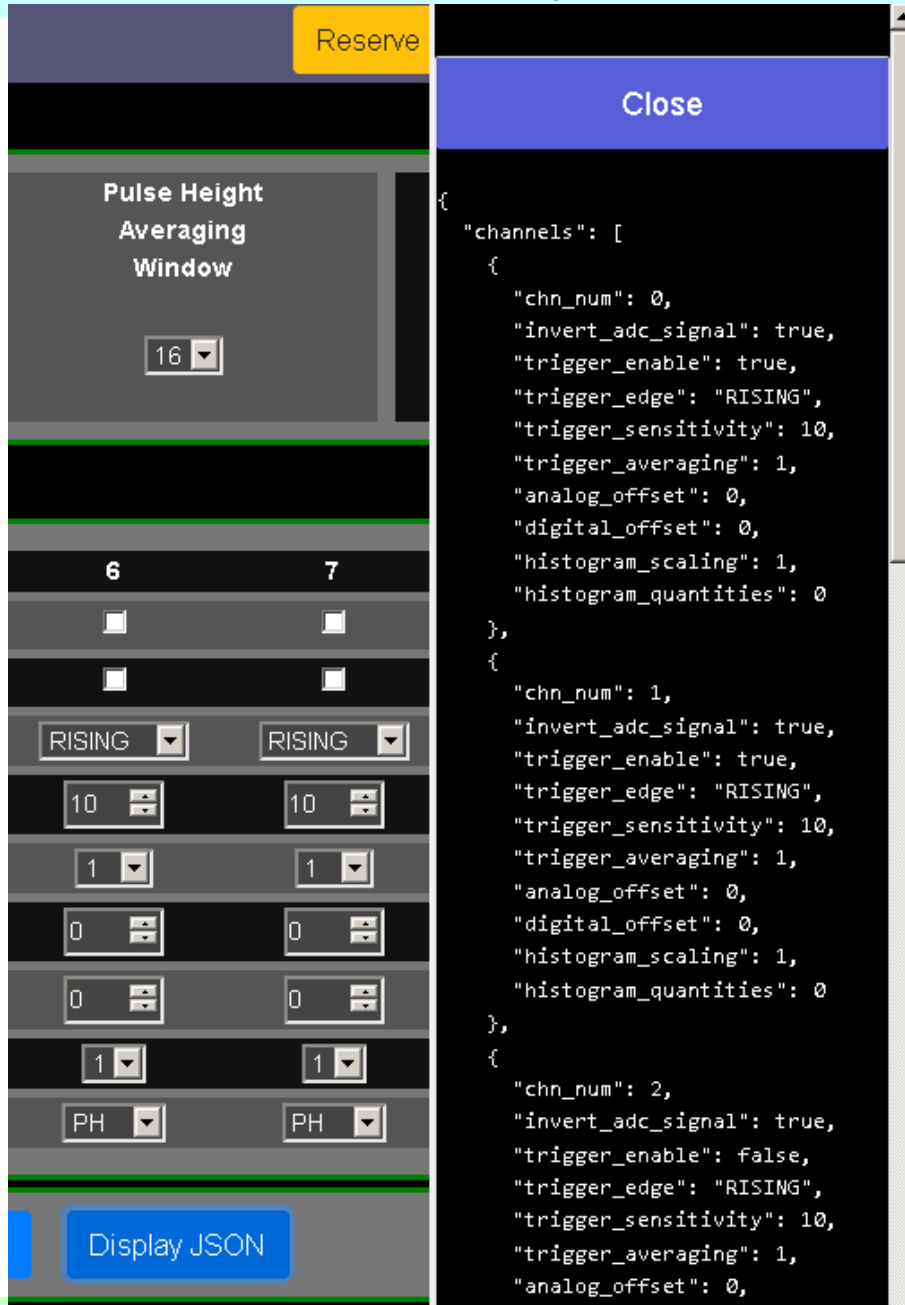
Trigger X Position	Trigger Active Window	Pulse Height Window	Baseline Restore Enable	Baseline Restore Exclusion	Pulse Height Averaging Window
128	100	100	<input checked="" type="checkbox"/>	400	32

Channel Settings

Channel Number	0	1	2	3	4	5	6	7	8	9
Invert ADC Signal	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Trigger Enable	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Trigger Edge	RISING	RISING	RISING	RISING	RISING	RISING	RISING	RISING	RISING	RISING
Trigger Sensitivity	4	4	10	10	10	10	10	10	10	10
Trigger Averaging	32	32	1	1	1	1	1	1	1	1
Histogram Scaling	1	1	1	1	1	1	1	1	1	1
Histogram Quantities	PH	PH	PH	PH	PH	PH	PH	PH	PH	PH

At the bottom of the interface, there are five buttons: 'Save Settings', 'Load Settings', 'Read from the board', 'Send to the board', and 'Display JSON'.

JSON Protocol is used in this scripted web GUI.



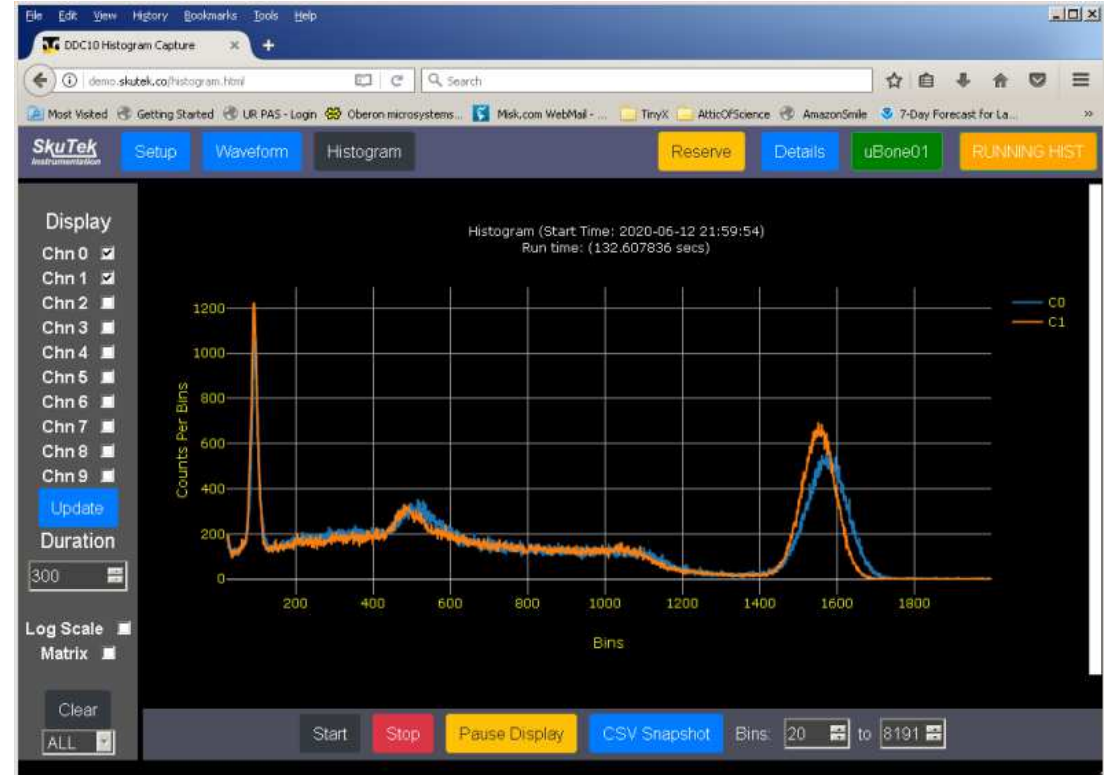
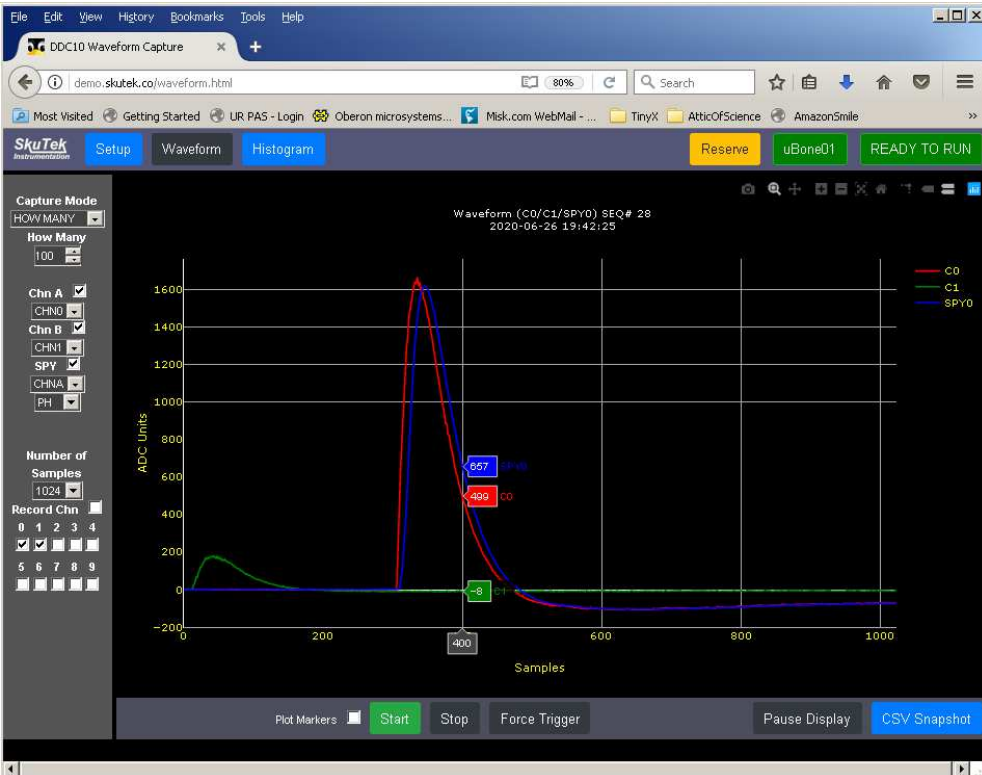
The screenshot shows a web interface with a 'Reserve' button at the top right. Below it is a 'Close' dialog box. The main content area displays a 'Pulse Height Averaging Window' with a dropdown menu set to '16'. Below this are two columns of controls, labeled '6' and '7', each with a checkbox and a dropdown menu. The '6' column has a 'RISING' dropdown, a numeric input '10', a dropdown '1', a numeric input '0', and a dropdown '1'. The '7' column has a 'RISING' dropdown, a numeric input '10', a dropdown '1', a numeric input '0', and a dropdown '1'. At the bottom, there are two 'PH' dropdowns and a 'Display JSON' button.

```
{
  "channels": [
    {
      "chn_num": 0,
      "invert_adc_signal": true,
      "trigger_enable": true,
      "trigger_edge": "RISING",
      "trigger_sensitivity": 10,
      "trigger_averaging": 1,
      "analog_offset": 0,
      "digital_offset": 0,
      "histogram_scaling": 1,
      "histogram_quantities": 0
    },
    {
      "chn_num": 1,
      "invert_adc_signal": true,
      "trigger_enable": true,
      "trigger_edge": "RISING",
      "trigger_sensitivity": 10,
      "trigger_averaging": 1,
      "analog_offset": 0,
      "digital_offset": 0,
      "histogram_scaling": 1,
      "histogram_quantities": 0
    },
    {
      "chn_num": 2,
      "invert_adc_signal": true,
      "trigger_enable": false,
      "trigger_edge": "RISING",
      "trigger_sensitivity": 10,
      "trigger_averaging": 1,
      "analog_offset": 0,

```

- JSON is plain ASCII text which is parsed by the SBC.
- It does not have to come from a web browser.
- Other applications can send JSON.
- JSON is well structured.
- JSON can be understood by humans.
- Other protocols can be implemented as well:
 - Plain HTTP.
 - Direct calls of Python functions under Jupyter.
 - Direct calls of Python functions under RPyC.
 - Python scripts executed at command prompt.
 - Command line calls of C utilities.

- The waveform and histogram displays can be shown in **any browser, any operating system**.
- One can even use **a cell phone**. Not really encouraged, but it works.
- The GUI does **not** require **any software installation** on the remote host. It runs in the browser.
- **Multiple users** can connect to the same instrument.
- We use a “virtual sticky note” to warn against access conflicts.



The multiple histogram display is shown in multiple browsers under Windows 10.

The SSH terminal can be used for diagnostic, copying the files, mounting NFS volumes, etc.

The screenshot displays a Windows 10 desktop environment with a remote web GUI interface for Skutek Instrumentation. The interface is split into several sections:

- Browser Windows:** Two browser windows are open. The left one shows the 'Histogram' page with a control panel on the left and a histogram plot. The right one shows a similar page with a 'Display' control panel on the left and a histogram plot.
- Control Panels:** Both browser windows have a 'Display' panel on the left with a list of channels (Chn 0 to Chn 9) and checkboxes. The right browser window also has a 'Duration' panel with a 'Refresh' button and a 'Log Scale' panel with a 'Matrix' checkbox.
- Histograms:** Both histograms show 'Counts Per Bins' on the y-axis (0 to 1200) and 'Bins' on the x-axis (20 to 160). The left histogram shows two overlapping curves, one blue (labeled 'C0') and one orange (labeled 'C1'). The right histogram shows a single orange curve.
- Terminal Window:** A Windows PowerShell terminal window is open in the center, displaying command-line output for file management and git operations. The output includes:

```
CMakeLists.txt      ChangeLog.txt      build               git_autoversion    ndso
CMakeSettings.json  LICENSE            deploy.sh          html               provisio
COPYING             Vagrantfile        futils            libraries          reformat
root@ubone:/home/skutek/ddc-apps-new# cd ..
root@ubone:/home/skutek# ls
ddc-apps  ddc-apps-new  ddc-apps-orig  ddc-scripts  firmware
root@ubone:/home/skutek# cd ddc-scripts/
root@ubone:/home/skutek/ddc-scripts# ls
Flash.py          mount_share_v21.bash  setupJupyter.sh  setup_proxy.ba
mount_share.bash  zeprogram.py          setup_gpio.bash  show_eoprxoid.
root@ubone:/home/skutek/ddc-scripts# cd ..
root@ubone:/home/skutek/firmware# ls
UboneDDC18-19Apr2019-8.2.8.bin
root@ubone:/home/skutek/firmware# cd ..
root@ubone:/home/skutek# ls
ddc-apps  ddc-Apps-new  ddc-apps-orig  ddc-scripts  Firmware
root@ubone:/home/skutek# cd ddc-apps-new/
root@ubone:/home/skutek/ddc-apps-new# git status
On branch vitkus-devel
Your branch is up-to-date with 'origin/vitkus-devel'.
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   CMakeLists.txt

no changes added to commit (use "git add" and/or "git commit -a")
root@ubone:/home/skutek/ddc-apps-new# git diff
diff --git a/CMakeLists.txt b/CMakeLists.txt
```

Summary

- Hardware:
 - We added a new **FemtoDAQ+** to our digitizer family.
 - FemtoDAQ+ will offer **significantly better performance** than the previous FemtoDAQ.
 - All our products will **uniformly** use the same MicroBone SBC for setup and control.
- Firmware:
 - We developed a unified approach to organizing the firmware around Instrument **Features**.
 - A Feature consists of the Application Firmware, Register Interface, Software Driver, and UI.
 - The Register Interface will use the General Purpose Memory Controller of the AM5338 chip.
 - Adoption of the GPMC was one of the main motivations for developing the FemtoDAQ+.
- Software:
 - We adopted a unified web-based **JavaScript** technology for instrument setup and control.
 - The GUI is portable to **any operating system**. It can even run on a cell phone.
 - Running the GUI **does not require installing any software** on the target host.
 - The GUI is using JavaScript which can run in **any browser**.
 - GUI can setup the instrument and save the configuration.
 - Light weight experiments can be performed under the new GUI with storage of data to disk.
 - Regular event files can be written to the SD card or to NFS mounted disk.

- **Continue** development of firmware and software for our digitizers.
- **Adopt** the new web-based technology in all our products.

- A **family of digitizers**, from 2 up to 40 channels per unit, 14 or 16 bits @ 100 MSPS.
 - The versions with 250 MSPS are under development. The boards are assembled.
 - The digitizers with more than 16 channels will stay at 100 MSPS because of power.
- The 32-channel digitizer will be compatible with the GRETA / GRETINA / DGS environment.
 - This project was presented yesterday.
- **Other digitizers** are targeting small labs, education, and T&M markets.
- **Very low noise**: RMS about 160 microvolts, that is ~ 1.3 LSB @ 14 bits.
- Setup, monitoring, and diagnostic with on-board **Linux** Single Board Computer (SBC).
 - SBC can also perform **readout** at the rate ~ a few megabytes per second.
 - SBC can write **formatted event files** directly to NFS mounted disks.
 - SBC can monitor the detector signals with **low latency** in near **real time**.
 - SBC can show an **interactive display** of waveforms and histograms **in any browser**.
- A **variety of options** for the control software and GUI to be executed by the Linux SBC.
 - SSH, command line, Python, Jupyter, and Remote Python Call (RPyC).
 - JavaScript GUI compatible with any browser, any host platform (**even a cell phone**).

Joanna Klima, Gregory Kick, David Miller, James Vitkus



Consultant: Eryk Druszkiewicz

Interns:

Mandy Nevins, Jeffrey Sylor, Dinesh Anand Bashkaran,
Brian Kroetz, Vedant Karia.

Special thanks to Michelle Shinn and Manouchehr Farkhondeh