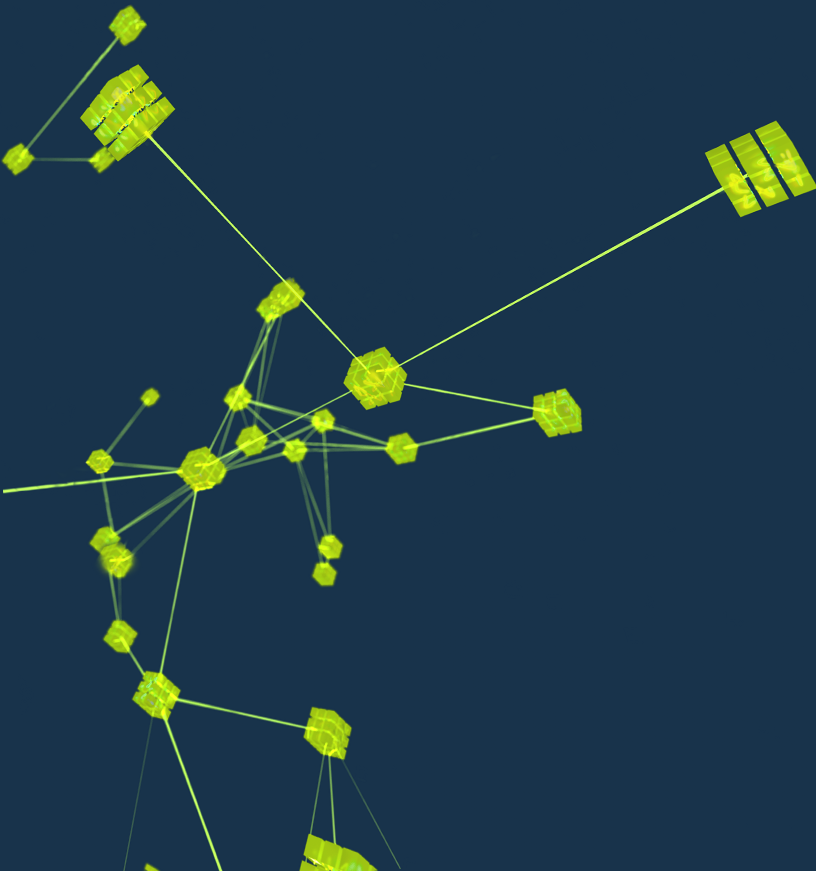




PVAccess on a real-time OS
with ArchiverAppliance as metadata store



EPICS Collaboration Meeting June 2019

Project CRYVISIL(*) (to resolve glass dynamics)

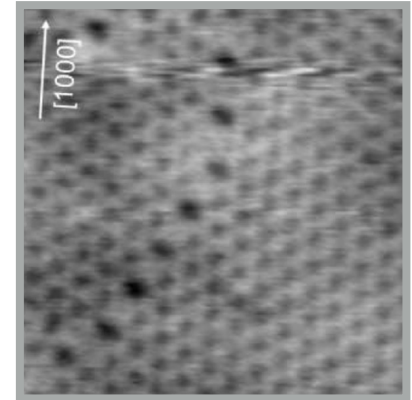
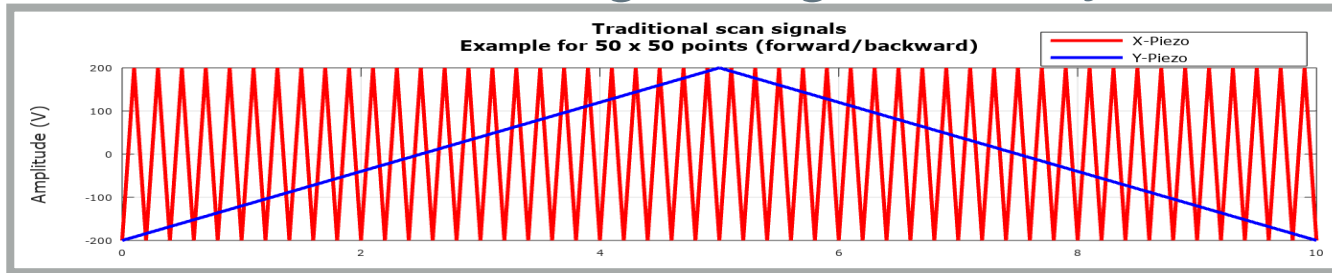
- build up of a very high speed scanning tunneling microscope (VHS-STM)
- High data rate (up to 100 MByte/s for ~5hrs)
- EPICS Control System with VMEbus

(*)This project has received funding from the European Research Council (ERC) under the European Union's Advanced Grant (AdG), 2014, ERC-2014-ADG

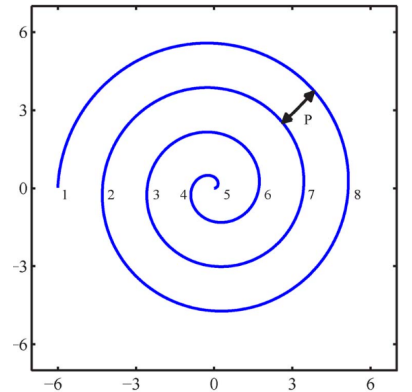
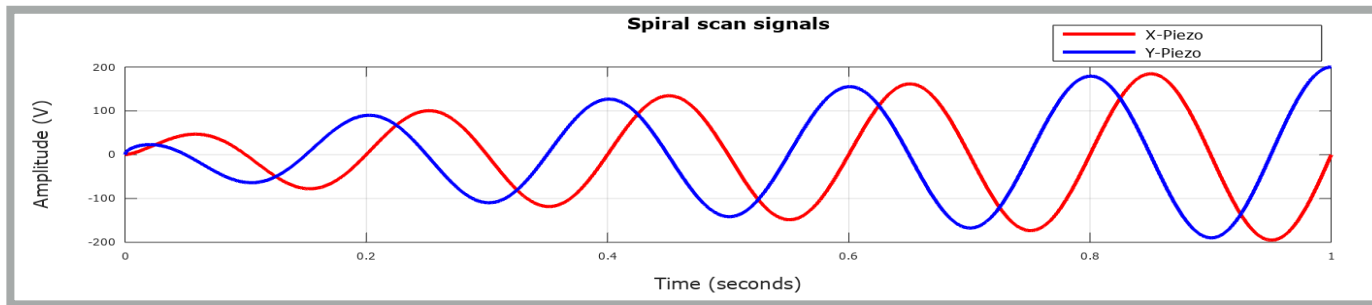


STM tip driven by piezo actuators

- traditional scanning (triangular for x-y)



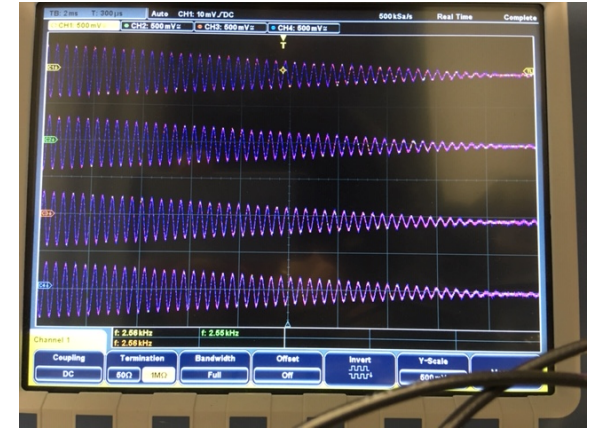
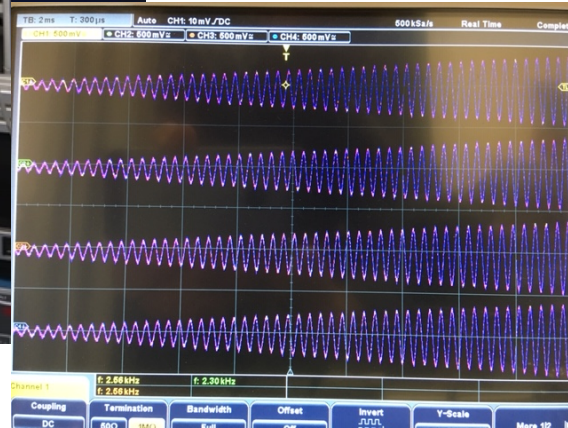
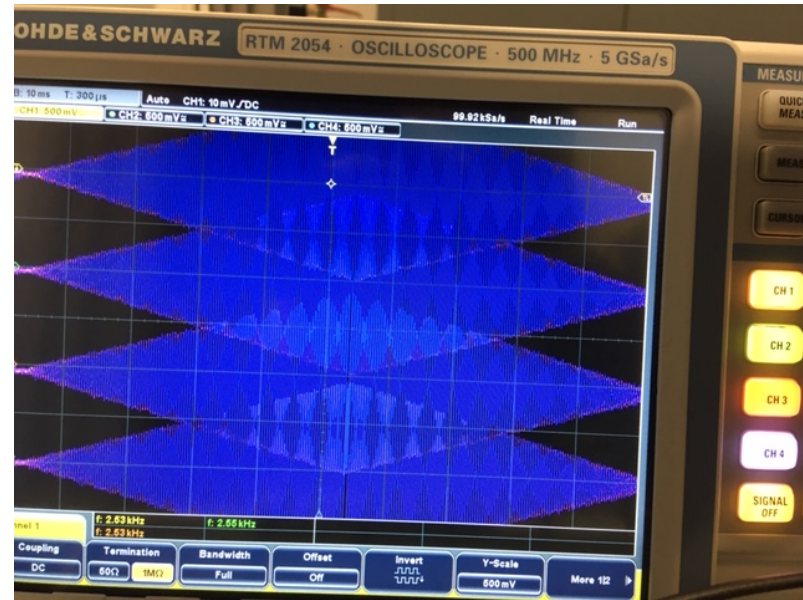
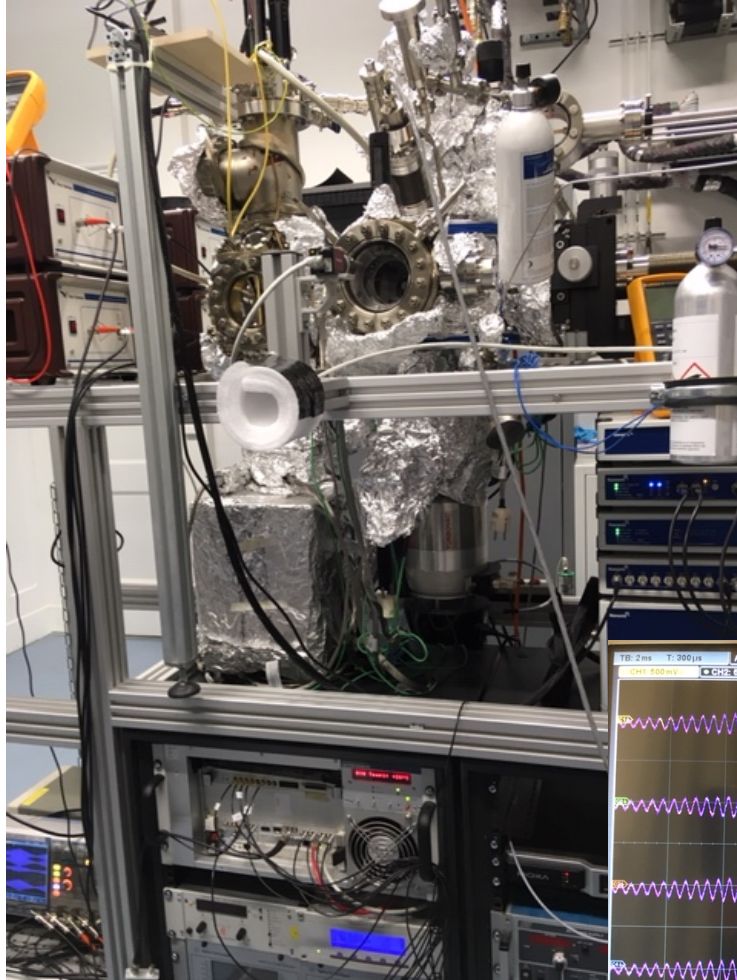
- reverse movement at high speed leads to distortion
- new scanning with spirals



Ref: A New Scanning Method for Fast Atomic Force Microscopy

Iskandar A. Mahmood, S. O. Reza Moheimani, Senior Member, IEEE, and Bharath Bhikkaji in IEEE TRANSACTIONS ON NANOTECHNOLOGY, VOL. 10, NO. 2, MARCH 2011

FAST SCANNING TUNNELING MICROSCOPE








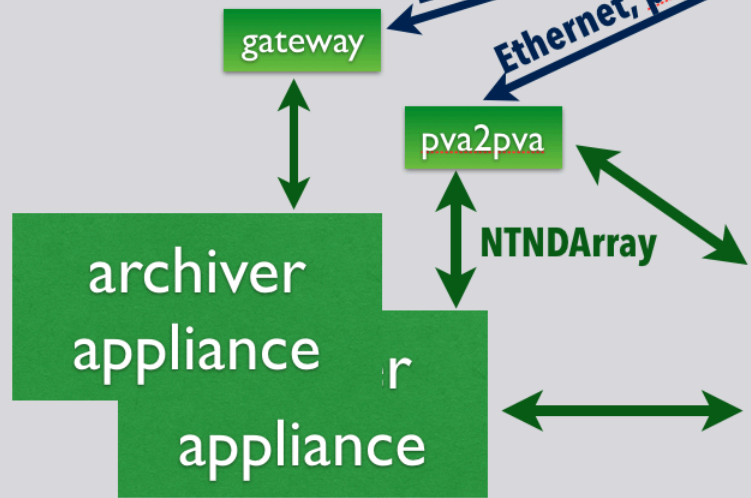
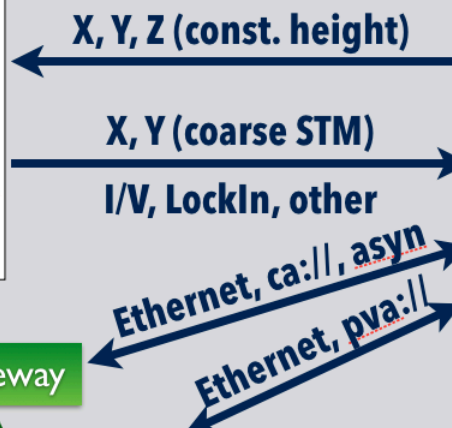
EPICS V4 inside

CRYVISIL



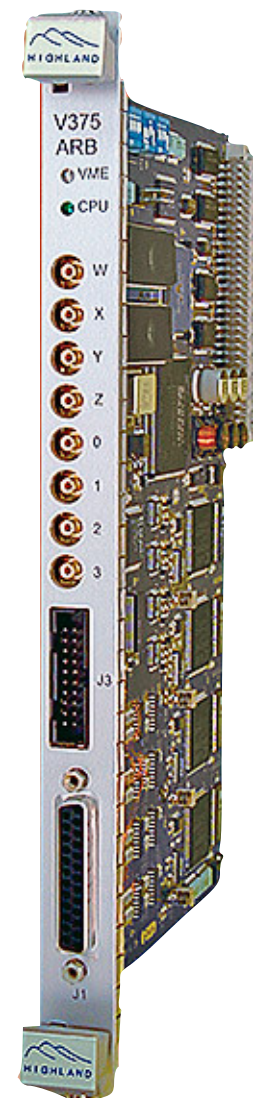
“fast STM” - control

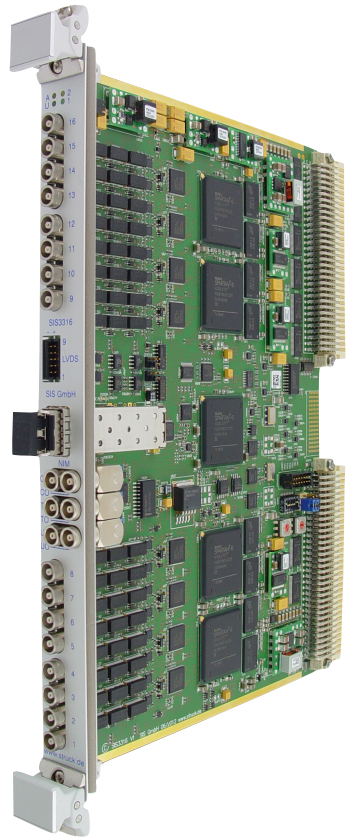
VMEbus	
	Highland V375, 4-channel arbitrary waveform generator, 15 MHz point step rate
	struck SIS3316, 16 channel digitizer, 14bit, 250 MS/s per channel, 64 MSamples memory/ch
	MVME2500, QorIQ, 8GB, 4 Gbit/s Ethernet
	MVME6100, PowerPC, 2GB, 2 Gbit/s Ethernet, 2 PMC-X
	 



Waveform generator Highland V375

- 4 independent direct digital synthesizer (DDS) frequency sources allow smooth variation of waveform scan rates
- 4 versatile, memory-table-driven waveform generators scan up to 65,536 discrete points per waveform at up to 15 MHz point step rate
- 16-bit amplitude resolution; 32-bit frequency resolution
- Output frequency, amplitude, phase, and DC offset are smoothly variable in real time
- Versatile wave memory partitioning allows waveform read/write operations concurrent with wave generation, so multiple waveforms can be loaded and selected in real time



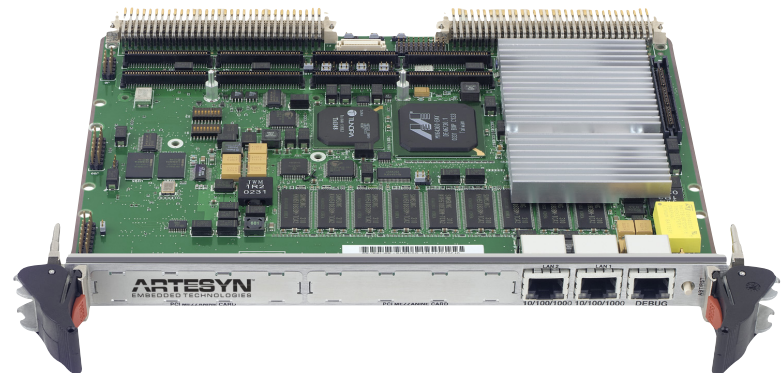


Digitizer Struck SIS3316

- Single width 6U VME card
- 16 channels, 250 MS/s per channel, 14-bit resolution, 64 MSamples memory/channel
- Two programmable input ranges, 50 Ω or high impedance programmable, Offset DACs
- 125 MHz analog bandwidth, Internal/External clock

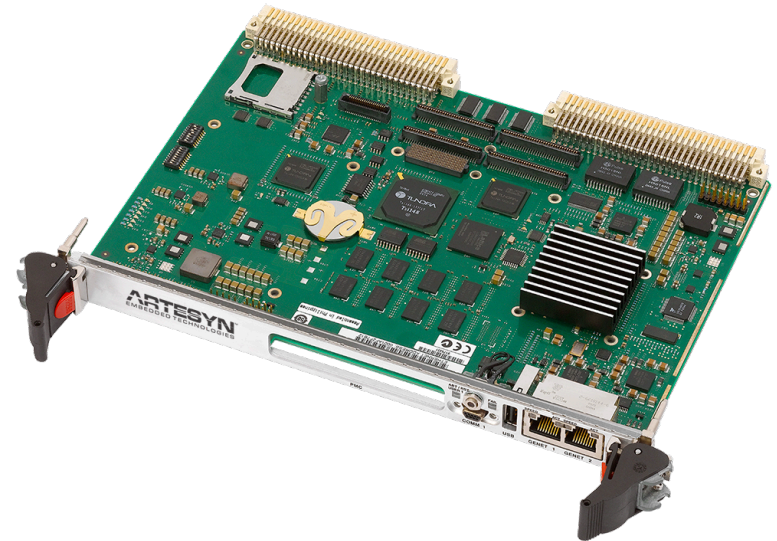
CPU MVME6100 (beatnik)

- 2eSST VMEbus protocol with 320MB/s transfer rate across the VMEbus
- MPC7457 PowerPC[®] processor running at up to 1.267 GHz
- 128-bit AltiVec coprocessor for parallel processing, ideal for data-intensive applications
- Dual Gigabit Ethernet interfaces for high performance networking



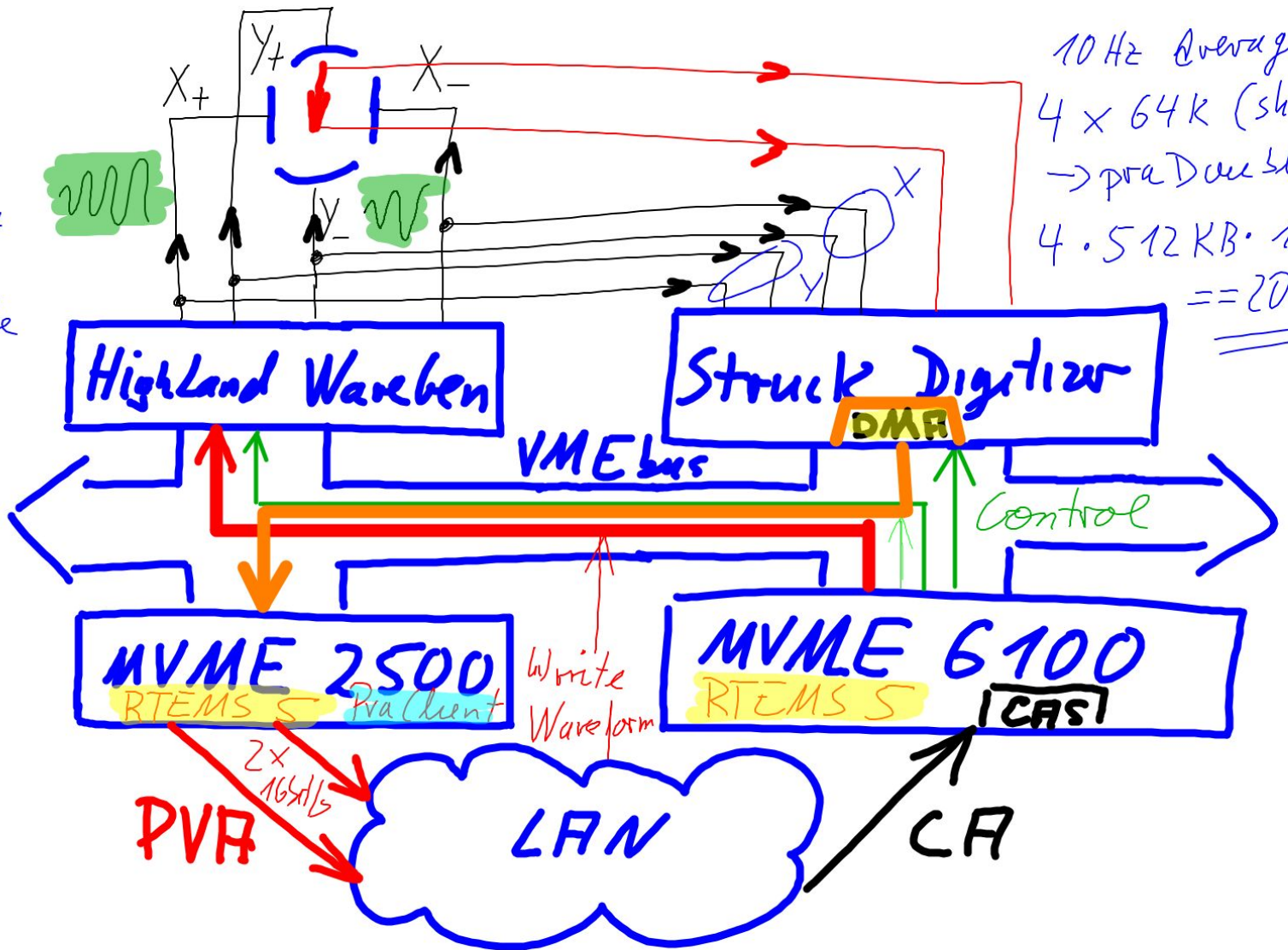
CPU MVME2500

- 800 MHz, 1.0 GHz or 1.2 GHz NXP QorIQ P2010 and P2020 processors
- 1GB or 2GB DDR3-800, soldered down
- Three on-board Gigabit Ethernet interfaces
- Five serial ports
- One USB 2.0 port
- One or two PCM/XMC site



VMEBUS HARDWARE/SOFTWARE

Goal:
 1 kHz / 5TP
 SMP · x4^{x4}
 20MP · 8 Byte
 160 MB/sec



10 Hz Average Mode
 4 x 64K (short)
 → pro Double
 4 · 512 KB · 10 $\frac{1}{5}$
 == 20 MB/s

pva-Pv on unix-python-server

```

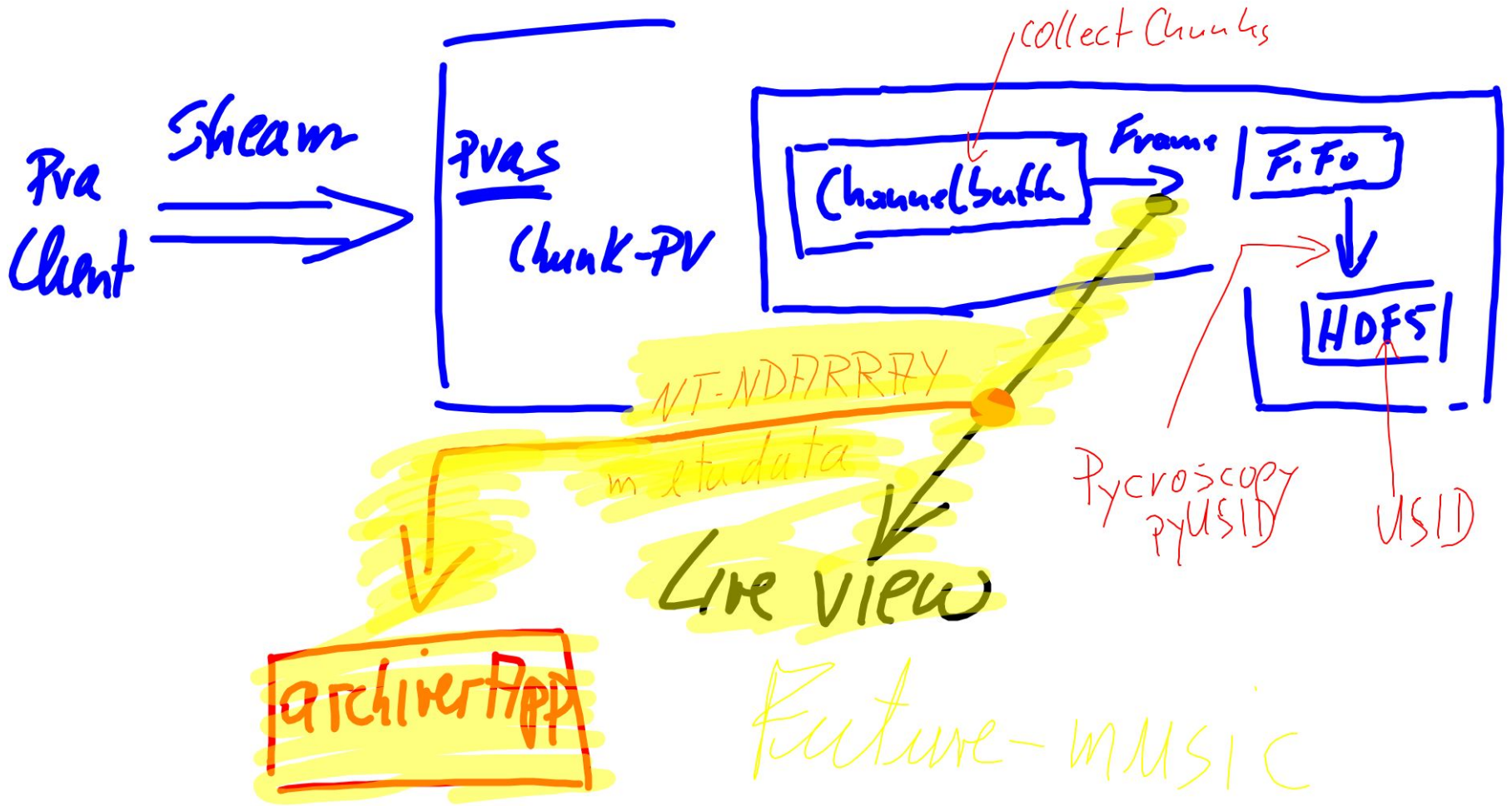
epics@kannsein:~$ pvinfo CRYVISIL:STM:FASTSCAN:IMAGE_CHUNK
CRYVISIL:STM:FASTSCAN:IMAGE_CHUNK
Server: 141.14.133.59:41002
Type:
  structure
    structure value
      double[] column0
      double[] column1
      double[] column2
      double[] column3
    string[] labels
    string[] units
    string descriptor
    long counter
    long uniqueId
    structure[] attribute
      structure
        string name
        any value
        string descriptor
        int sourceType
        int source
    double[] digitizerTimeStamp
    structure timeStamp
      long secondsPastEpoch
      int nanoseconds
      int userTag
    structure alarm
      int severity
      int status
      string message
  
```

ca-Pv's on RTEMS-IOC

```

CRYVISIL:AWG0:Channel0:SetSource
CRYVISIL:AWG0:Channel0:SetClockDivisor
CRYVISIL:AWG0:Channel0:Delay
CRYVISIL:AWG0:DDSB:SetDDSFreq
CRYVISIL:AWG0:Channel1:SetSource
CRYVISIL:AWG0:Channel1:SetClockDivisor
CRYVISIL:AWG0:Channel1:Delay
CRYVISIL:AWG0:DDSC:SetDDSFreq
CRYVISIL:AWG0:Channel2:SetSource
CRYVISIL:AWG0:Channel2:SetClockDivisor
CRYVISIL:AWG0:Channel2:Delay
CRYVISIL:AWG0:DDSD:SetDDSFreq
CRYVISIL:AWG0:Channel3:SetSource
CRYVISIL:AWG0:Channel3:SetClockDivisor
CRYVISIL:AWG0:Channel3:Delay
CRYVISIL:AWG0:Amp0:Amplitude
CRYVISIL:AWG0:AmpW:Offset
CRYVISIL:AWG0:Amp1:Amplitude
CRYVISIL:AWG0:AmpX:Offset
CRYVISIL:AWG0:Amp2:Amplitude
CRYVISIL:AWG0:AmpY:Offset
CRYVISIL:AWG0:Amp3:Amplitude
CRYVISIL:AWG0:AmpZ:Offset
CRYVISIL:SIS0:CH1:Offset
CRYVISIL:SIS0:CH2:Offset
CRYVISIL:SIS0:CH3:Offset
CRYVISIL:SIS0:CH4:Offset
CRYVISIL:SIS0:CH5:Offset
CRYVISIL:SIS0:CH6:Offset
CRYVISIL:SIS0:CH7:Offset
CRYVISIL:SIS0:CH8:Offset
  
```

PROCESS-FLOW



- pvAccess driver receives NTNDArrays over the network, converts to NDArrays and calls plugins
- Can be used to run areaDetector IOC and plugins on another machine or in another process
- High performance:
 - ~1.8 GB/s shown here with interprocess communication

Info's from Mark Rivers, Danke

The screenshot shows the ADPvaDriver.adl@corvette application window with the following sections:

- Setup:**
 - asyn port: PVA
 - EPICS name: 13PVA1:cam1
 - Manufacturer: PVAccess driver
 - Model: Basic PVAccess driver
 - Serial number: No serial number
 - Firmware version: No firmware
 - SDK version: 6.1.2
 - Driver version: 1.6.0
 - ADCore version: 3.5.0
 - Status: **Connected**
 - Connection:
 - Debugging:
- Collect:**
 - # Images: 100
 - # Images complete: 145355
 - Image mode: Continuous
 - Status: **Collecting**
 - Acquire:
 - # Queued arrays: 0
 - Wait for plugins:
 - Acquire busy: **Acquiring**
 - Detector state: **Idle**
 - Status:
 - Image counter: 195392
 - Image rate: 462.00
 - Array callbacks: Enable
- Plugins:**
 -
 - Stats:
- Readout:**

	X	Y
Sensor size	1024	1024
Binning	1	1
Region start	0	0
Region size	1024	1024
Reverse	No	No
Image size	1024	1024
Image size (bytes)	4194304	
Data type	Int32	
Color mode	Mono	
- Buffers:**
 - Buffers used: 3
 - Buffers alloc/free: 36 33
 - Memory max/used (MB): 0.0 144.0
 - Buffer & memory polling:
 - Empty free list:
- pvaDriver:**
 - Overrun counter: 154346
 - 13SIM1:Pval:Image
 - PV Name:
 - PV Connection: **Up**
- Attributes:**
 - File:
 - Macros:
 - Status: **File not found**

Until you get this going, I've attached couple of short python files that should help you figure out if this is a python issue:

server1 generates PVA structures with a few waveforms and exposes these over its PVA channel, and server2 monitors this channel and serves the same data over the second channel. If I understood your use case correctly, this should be similar to what you are trying to do. Simply run server1 in one terminal, and server2 in another on the same machine in order to eliminate any network issues.

On my machine, which is about 2 years old and has Intel Xeon E5-2620 v3 @ 2.40GHz, with pvapy 1.6.0/epics 7.0.2.2/boost 1.70.0 and python 3.7 (installed from conda packages), I get the following results:

Server 1 generates data at over 50MB/s:

Runtime: 136.417 [s], Generated Arrays: 16400, Array Size: 491520 [B], 0.46875 [MB],

Array Rate: 120 [Hz],

Data Rate: 59090338.246 [B/s], **56.353 [MB/s]**

Server 2 keeps up without missing anything:

Runtime: 99.694 [s], Received Arrays: 11900, Missed Arrays: 0, Missed Arrays Since Last:

0, Current Array Rate: 131.458 [Hz], Average Array Rate: 119.366 [Hz]

For the record, if I replicate setup #3, and have C++ IOC with area detector on one machine connected to python pvapy server on another machine over 10Gbps link, with 12MB arrays the pvapy server updates its database record at a rate of over 110 Hz, which means that the server is receiving data at a rate of over **1.2GB/s** (it is keeping up with a fully saturated 10Gbps link).

Info's from Siniša Veseli, Danke

In the meantime I have slightly tweaked my server1 example to use numpy arrays rather than python lists, which boosted its performance considerably. It can now generate/serve **data at a rates that exceed 1GB/s**. Not quite what area detector can do, but also quite usable for quick testing. On my machine, for example, I see this with 4MB arrays:

Runtime: 45.306 [s], Generated Arrays: 14000, Array Size: 4177920 [B],
3.984375 [MB], Array Rate: 309 [Hz], Data Rate: 1291005799.034 [B/s],
1231.199 [MB/s]

Info's from Siniša Veseli, Danke


```
string channelName("CRYVISIL:STM:FASTSCAN:IMAGE_CHUNK");  
string providerName("pva");
```

```
PvaClientPtr pva = PvaClient::get(providerName);
```

```
//pva->setDebug(true);
```

When I turn this on, I always get the message that the channel is connected twice.

```
cout << "Update task -> channelName " << channelName << " providerName " << providerName << endl;
```

```
PvaClientPutPtr put(pva->channel(channelName,providerName, 3.0 )->put("field()"));
```

```
Status status = put->waitConnect();
```

```
if(!status.isOK()) {cout << " createPut failed !!! ???? \n"; return;}
```

```
...
```

The image displays two side-by-side browser windows showing the EPICS Archiver Appliance Viewer. Both windows show a time-series plot of a signal labeled 'CRYVISIL:AWG0:Channel0:wform' and 'Time'. The x-axis represents time from 14:10 to 15:00 on May 29, 2019. The y-axis represents signal amplitude, ranging from -8000 to 10. The signal shows a clear upward trend in amplitude over time, with a high-frequency oscillation superimposed on a lower-frequency envelope.

Below the plots is a browser window showing the 'FHI Archiver Appliance (aa1)' management console. The console includes a navigation menu (Home, Reports, Metrics, Storage, Appliances, Integration, Help) and a main content area with the following text:

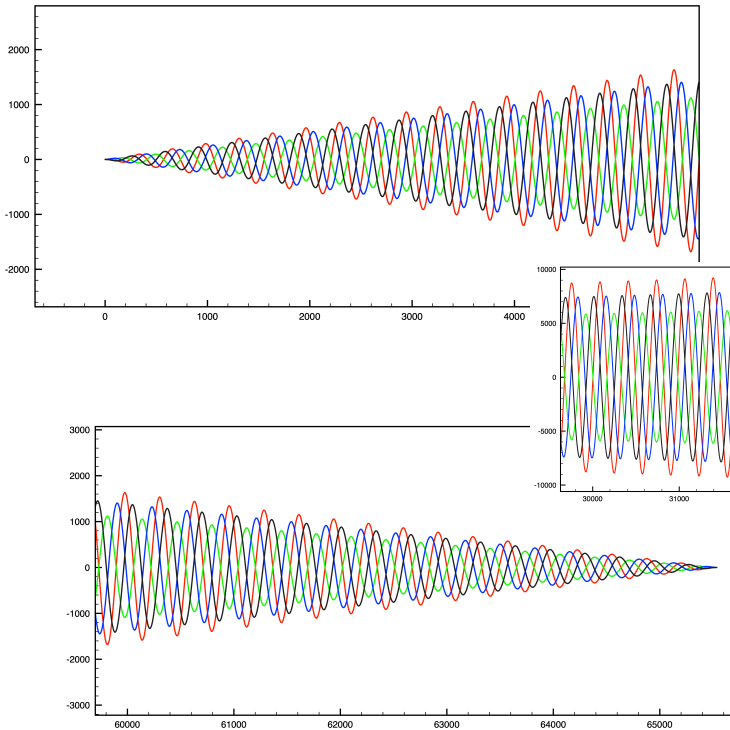
This is the EPICS archiver appliance management console for FHI including the FEL facility. Please contact Heinz Junkes at 4270 if you have any issues. To check the status of or to archive some PV's, please type in some PV names here.

Below this text is a text input field containing the PV names: CRYVISIL:AWG0:Channel0:wform and CRYVISIL:AWG0:Channel1:wform. Below the input field are buttons for 'Check Status', 'Archive', 'Archive (specify sampling period)', 'Lookup', 'Pause', and 'Resume'.

At the bottom of the console is a table with the following data:

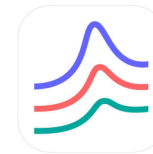
PV Name	Status	Appliance	Connected?	Monitored?	Sampling period	Last event	Details	Quick chart
CRYVISIL:AWG0:Channel0:wform	Being archived	archappl0	true	true	1.0	May/29/2019 14:58:56 +02:00		
CRYVISIL:AWG0:Channel1:wform	Being archived	archappl0	true	true	1.0	May/29/2019 14:58:56 +02:00		

- To visualize data on iPhone/iPad Daviz is used



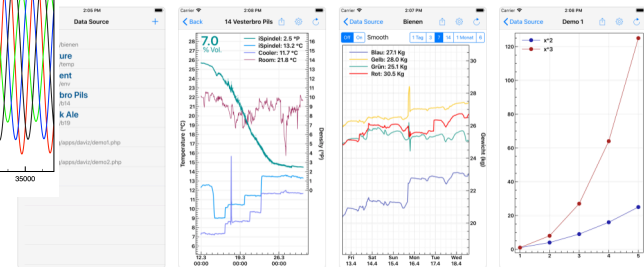
App Store Preview

This app is only available on the App Store for iOS devices.



Daviz 4+
Mike Wesemann
Free

shots iPhone iPad



Description

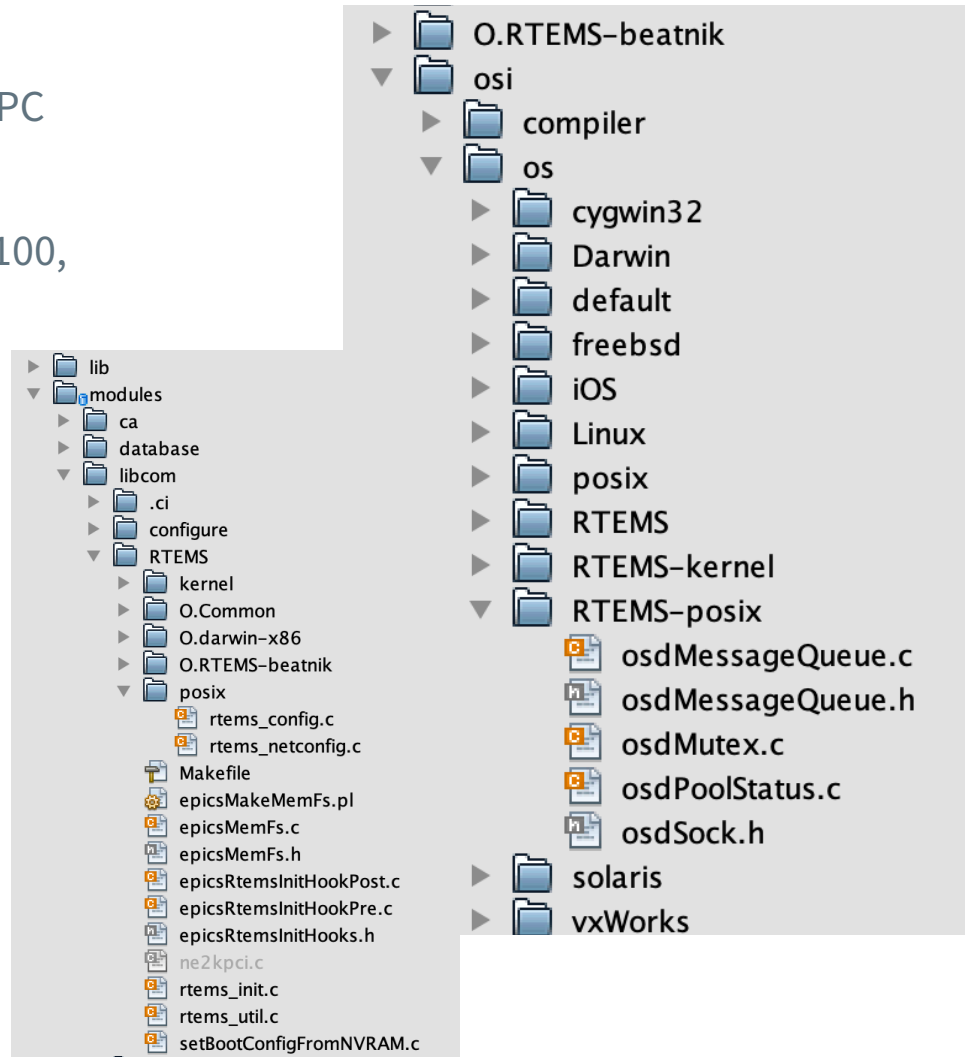
Daviz is a data visualisation tool for IOS. The App works both on iPhone and iPad. The macro language which is used by Daviz is very similar to the one you can find in Plot2

In the App you have to enter a data source which is just an URL. In the simplest case this can be a text file which provides data and formatting information (See documentation at <https://davizdoc.micw.org>)....

[more](#)

- data read from the archiver on demand by JSON like
 $\$url = "http://aa1.fhi-berlin.mpg.de:17668/retrieval/data/getData.json?pv=CRYVISIL\%3AAWG0\%3AChannel0\%3Awform";$
 $\$json = file_get_contents(\$url);$

- Still waiting for a release ...
- Epics 7 support available, used mainly with PPC (arm is working as well)
- funding of BSP - support (MVME6100, MVME2100, MVME3100 and MVME2500) by FHI
- funding of dynamic linker by HZB
- For lack of time :
 - line editing missing
 - missing test on e.g. telnet support (and other network stuff)
 - missing tests on intel hardware
 - not well tested boot config from NVRAM



Some improvements (Danke Michael Davidsaver)

```
cry-test> rt help
```

```
help: The topics are
```

```
  all, help, misc, files, mem, rtems, monitor
```

```
cry-test> rt malloc
```

```
C Program Heap and RTEMS Workspace are the same.
```

```
Number of free blocks:           141
Largest free block:              164206288
Total bytes free:                164941976
Number of used blocks:          20338
Largest used block:             20000008
Total bytes used:               363718176
Size of the allocatable area in bytes: 528660152
Minimum free size ever in bytes: 162844968
Maximum number of free blocks ever: 179
Maximum number of blocks searched ever: 98
```

```
...
```