RADCELF User Guide

Rev	Date	Description
1	20160826	init

Document created using OpenOffice.Org www.openoffice.org.

This document and D-TACQ Software comprising platform Linux port, Linux kernel modules and most applications are released under GNU GPL/FDL:

Document:

Software:

Table of Contents

1 Introduction

1.1 Scope

Covers software control of RADCELF.

RADCELF has the following programmable devices:

1) AD9854 DDS : ddsA, ddsB, ddsC
2) AD9512 CLKD chips clkdA, clkdB
3) AD7417 temperature and slow ADC
4) 8 bit slow DIO buffer.

1.2 References

1) RADCELF Block Diagram
2) AD9854 Data Sheet
3) AD9854 Quick Ref
4) AD512 Data Sheet
5) 4GUG

1.3 Software concept

Each device is represented by a device driver that presents every register as a virtual file or “knob”. The knobs have meaningful names and are set with simple ascii strings. Where possible, knobs are read/write. These features make for easy local scripting. Scripting languages available on the ACQ400 series include : shell, TCL, python.

In addition, it's ACQ400 convention to present every user-controllable knob on a “site service” where it's accessible by connecting a TCP socket to a well-known port. This makes writing a remote automation script extremely easy. The site service may also be accessed locally using the **`set.site`** command.

Then, to examine and optionally set the complete state of the device, virtual files are provided to dump / load all the registers in the device. This information is also presented in a dynamic web page for easy monitoring.

An emulation layer is also provided supporting the low level byte-address access command “**`Baadd`**” for the AD9854. This is for back compatibility with existing software, but we really do not recommend it, from both the point of view of code readability, and because the serial interface does not support byte addressing, it's actually really inefficient. The clue is in the name ..

2 AD9854 DDS

2.1 Virtual Devices

2.1.1 Top level directory for RADCELF

```
ls /dev/radcelf
clkdA  clkdB  ddsA   ddsB   ddsC
```

2.1.2 One virtual file per register

```
cd /dev/radcelf/ddsA
ls [A-Z][A-Z]*
CR     DFR    FTW1   FTW2   IPDMR  POTW1  POTW2  QDACR  QPDMR  RRCR   SKRR   UCR
```

2.1.3 Knobs accessed from site service

Chip	Site	Port	Local Command	Remote Command
ddsA	4	4224	set.site 4	nc ACQ 4224
ddsB	5	4225	set.site 5	nc ACQ 4225
ddsC	6	4226	set.site 6	nc ACQ 4226

2.1.4 Example Site Service Usage

Shows a get followed by a set. It's a good idea to check that the set worked, at least in debug by following it up with another get. Or refer to the web page.

```
acq1001_068> set.site 4
FTW1
100000000000
FTW1=110000000000
FTW1
110000000000

Baadd=B0409
FTW1
090000000000
```

So either way you can set FTW1, however, apart from being less legible, “Baadd” is <u>really</u> inefficient, because not only are 6 commands required to complete the word, each command has to write the full 7 byte command..

2.1.5 Socket with complete low level emulation

Ports 4244, 4245, 4346 emulate the Baadd protocol directly, this is provided so that customer's existing client software may run directly.

Configuration:

```
acq1001_068> cat /etc/inetd.radcelf.conf
4244 stream tcp nowait root set.Baadd set.Baadd 4
4245 stream tcp nowait root set.Baadd set.Baadd 5
4346 stream tcp nowait root set.Baadd set.Baadd 6
```

Example:

```
a-host> nc acq1001_068 4244
B0411
B0500
B0600
B0700
B0800
B0900

Equivalent to

nc acq1001_068 4224
FTW1=110000000000
```

2.1.6 Recommended Scripting Style

Convenience commands are provided to easy local scripting:

```
set.ddsA    FTW1  0123456789abc

set.clkdB   UPDATE      01
```

2.1.7 Help

```
acq1001_068> set.site 4 help2
BPSK                    : rw
      0 FPGA Enable BPSK
CR                      : rw
      [7,4] Control Register
DFR                     : rw
      [4,6] Delta Frequency Register
FTW1                    : rw
      [2,6] Frequency Tuning Word #1
FTW2                    : rw
      [3,6] Frequency Tuning Word #2
IPDMR                   : rw
      [8,2] I Path Digital Multiplier Register
OSK                     : rw
      0 FPGA Enable OSK
POTW1                   : rw
      [0,2] Phase Offset Tuning Word Register #1
POTW2                   : rw
      [1,2] Phase Offset Tuning Word Register #2
QDACR                   : rw
      [B,2] Q DAC Register 2 Bytes
QPDMR                   : rw
      [9,2] Q Path Digital Multiplier Register
RRCR                    : rw
      [6,3] Ramp Rate Clock Register
SKRR                    : rw
      [A,1] Shaped On/Off Keying Ramp Rate Register
UCR                     : rw
      [5,4] Update Clock Rate Register
strobe                  : rw
      0 manually strobes update if required
strobe_mode             : rw
      0   0:manual 1:self 2:group (hit strobe to assert group)
help                    :
      help
help2                   :
      /usr/share/doc/acq400_help0:help2
```

2.1.8 Register Dumps

```
acq1001_068> cat /proc/driver/ad9854/ddsA/rare
0000
0000
110000000000
000000000000
000000000000
00000000
000000
000f0041
0000
0000
00
```

```
0000
```

2.1.9 Load/Restore state

```
cat /proc/driver/ad9854/ddsA/rare >/mnt/local/ddsA_saved_state

… later

cat /mnt/local/ddsA_saved_state > /proc/driver/ad9854/ddsA/rare
```

2.1.10 Strobe

AD9854 requires the IOUD signal to clock register settings through to the device.

Each DDS driver provides the **strobe_mode** knob to control this operation.

Values for **strobe_mode** are:

0 : SPI_STROBE_NONE

no action, a separate write to the **strobe** knob pulses IOUD

1: SPI_STROBE_SELF

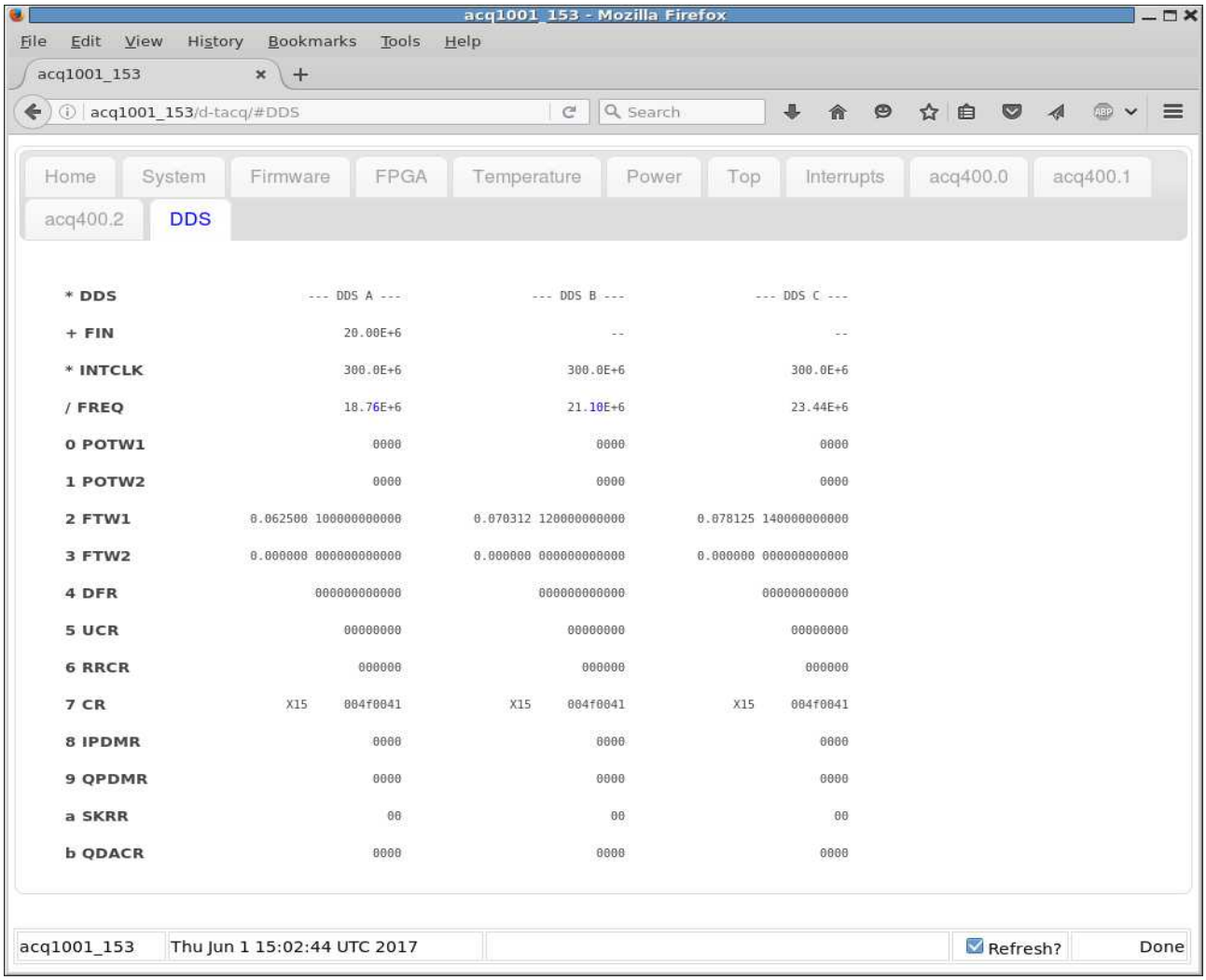
automatic IOUDD pulse after each command [default]

2: SPI_STROBE_GROUP

ddsA, ddsB constitute a group, in this mode, a write to the **strobe** knob pulses IOUD on both dds devices to enable synchronous update

2.2 Web page.

Shows complete register state of all 3 DDS, + measured output frequency



3 AD9512 CLKD

3.1 Virtual Devices

3.1.1 Top level directory for RADCELF

```
ls /dev/radcelf
clkdA  clkdB  ddsA   ddsB   ddsC
```

3.1.2 One virtual file per register

```
acq1001_068> cd /dev/radcelf/clkdA/
acq1001_068> ls [A-Z][A-Z]*
CSPD     DFA4     DIV0     DIV2     DIV4     LVDS3    LVPECL0  LVPECL2  UPDATE
DBP4     DFS4     DIV1     DIV3     FUNPS    LVDS4    LVPECL1  SCPC
```

3.1.3 Knobs accessed from site service

Chip	Site	Port	Local Command	Remote Command
clkdA	7	4227	set.site 7	nc ACQ 4227
clkdB	8	4228	set.site 8	nc ACQ 4228

3.1.4 Example Site Service Usage

Shows a get followed by a set. It's a good idea to check that the set worked, at least in debug by following it up with another get. Or refer to the web page.

```
acq1001_068> set.site 4
FTW1
100000000000
FTW1=110000000000
FTW1
110000000000
```

3.1.5 Help

```
acq1001_068> set.site 7
help2
CSPD                : rw
     [45 1] Clocks Select Power Down
DBP4                : rw
```

```
     [34 1] Delay Bypass 4
DFA4                 : rw
     [36 1] Delay Fine Adjust 4
DFS4                 : rw
     [35 1] Delay Full Scale 4
DIV0                 : rw
     [4A 2] Divider 0  8000 => Bypass
DIV1                 : rw
     [4C 2] Divider 1  8000 => Bypass
DIV2                 : rw
     [4E 2] Divider 2  8000 => Bypass
DIV3                 : rw
     [50 2] Divider 3  8000 => Bypass
DIV4                 : rw
     [52 2] Divider 4  8000 => Bypass
FUNPS                : rw
     [58 1] FUNCTION Pin and Sync
LVDS3                : rw
     [40 1] LVDS CMOS Out 3
LVDS4                : rw
     [41 1] LVDS CMOS Out 4
LVPECL0              : rw
     [3D 1] LVPECL Out 0
LVPECL1              : rw
     [3E 1] LVPECL Out 1
LVPECL2              : rw
     [3F 1] LVPECL Out 2
SCPC                 : rw
     [0 1] Serial Port Control Configuration
UPDATE               : rw
     [5A 1] Update Registers
```

4 Voltage Monitor and DIO

Knobs available on site 3.

d? : dio bit

in? : analog inputs

temp?: temperature.

```
acq1001_153> get.site 3
help
ai1
ai2
ai3
ai4
ai5
ai6
ai7
ai8
d0
d1
d2
d3
d4
d5
d6
d7
temp
temp2
```

4.1 DIO direction

Default: out, set by modifying file on boot as follows (script in /mnt/local/rc.user).

```
acq1001_153> cat /dev/gpio/CELF/.direction.d0
out
acq1001_153> echo in > /dev/gpio/CELF/.direction.d0
```

5 Boot Time Initialisation

`/usr/local/init/RAD-CELF-init`

Enables the DDS devices, configures a 25MHz source clock and sets the DDS to run internally at 300MHz.

6 Test routines

6.1 Sets some default clocks

/usr/local/CARE/radcelf-init-123

Configures ddsA , ddsB, ddsC with test frequencies. Use Web page to confirm it's working.

6.2 Set a single chirp

shell script runs on ACQ1001, sets single chirp

```
#!/bin/sh
#
# SETTING KAKA'AKOS CHIRP
#
# Set AD9854 clock remap to 25 MHz
export PATH=$PATH:/usr/local/bin
set.ddsC        CR      004C0041
set.ddsC        FTW1    1AAAAAAAAAAA
# Program AD9512 secondary clock to choose 25 MHz from the AD9854 remap
set.clkdB       CSPD    02
set.clkdB       UPDATE  01
# Program the chirp using Kaka'ako parameters
set.site 2 ddsA_upd_clk_fpga 1
set.ddsA        CR      004F0061
set.ddsA        FTW1    172B020C49BA
set.ddsA        DFR     0000000021D1
set.ddsA        UCR     01F01FD0
set.ddsA        RRCR    000001
set.ddsA        IPDMR   0FFF
set.ddsA        QPDMR   0FFF
set.ddsA        CR      004C8761
# Set the trigger
set.site 2 ddsA_upd_clk_fpga 0
# lera_acq_setup
# we assume a 25MHz from ddsC
# trigger from site 3 ddsA
set.site 1 trg 1,3,1
set.site 1 clk 1,3,1
set.site 1 hi_res_mode 1
# 25 MHz/4 = 6.25MHz / 512 = SR 12207
set.site 1 CLKDIV 4
```

7 Remote Scripting

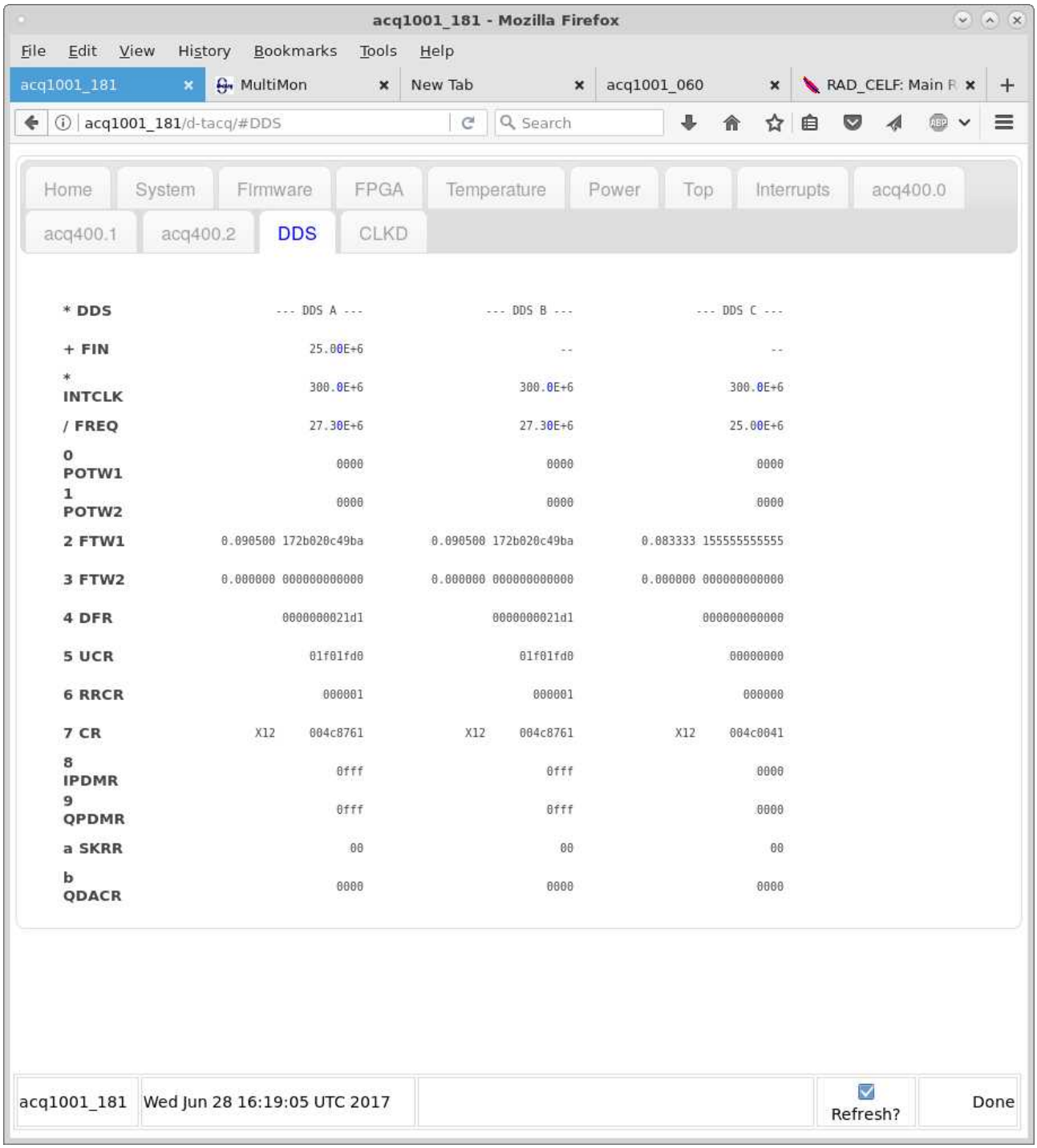
D-TACQ offers a host side api “HAPI”. This brings all the site services together in one consistent object API. https://github.com/petermilne/acq400_hapi_tests

Example below “radcelf-chirp-init.py” sets up a chirp on either ddsA or ddsB

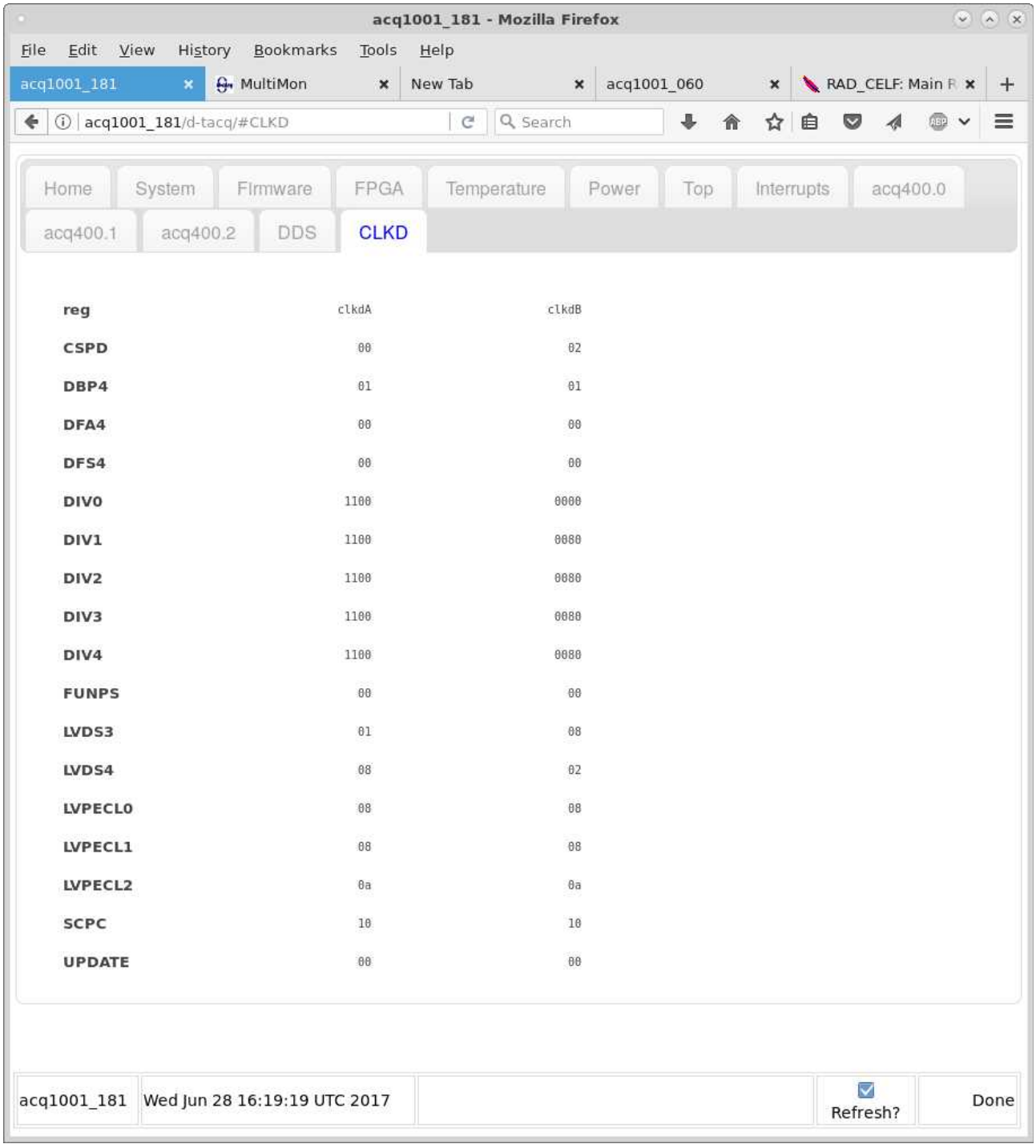
Note, it's basically identical to the shell script example, but with a richer capability.

```
def init_chirp(uut, idds):
# SETTING KAKA'AKOS CHIRP
#
# Set AD9854 clock remap to 25 MHz
    dds = uut.ddsA if idds == 0 else uut.ddsB

    uut.ddsC.CR     = '004C0041'
    uut.ddsC.FTW1   = FTW1(1.0/12.0)
# Program AD9512 secondary clock to choose 25 MHz from the AD9854 remap
    uut.clkdB.CSPD  = '02'
    uut.clkdB.UPDATE = '01'
# Program the chirp using Kaka'ako parameters
    set_upd_clk_fpga(uut, idds, '1')
    dds.CR     = '004F0061'
    dds.FTW1   = '172B020C49BA'
    dds.DFR    = '0000000021D1'
    dds.UCR    = '01F01FD0'
    dds.RRCR   = '000001'
    dds.IPDMR  = '0FFF'
    dds.QPDMR  = '0FFF'
    dds.CR     = '004C8761'
    set_upd_clk_fpga(uut, idds, '0')

# Set the trigger
# lera_acq_setup
# we assume a 25MHz from ddsC
# trigger from site 3 ddsA
    uut.s1.trg  = '1,3,1'
    uut.s1.clk  = '1,3,1'
    uut.s1.hi_res_mode = '1'
# 25 MHz/4 = 6.25MHz / 512 = SR 12207
    uut.s1.CLKDIV = '4'

# create an object instance and init_chirp
init_chirp(acq400_hapi.RAD3DDS("acq1001_181"), 0)
```

7.1 Monitoring After Chirp : Web pages:

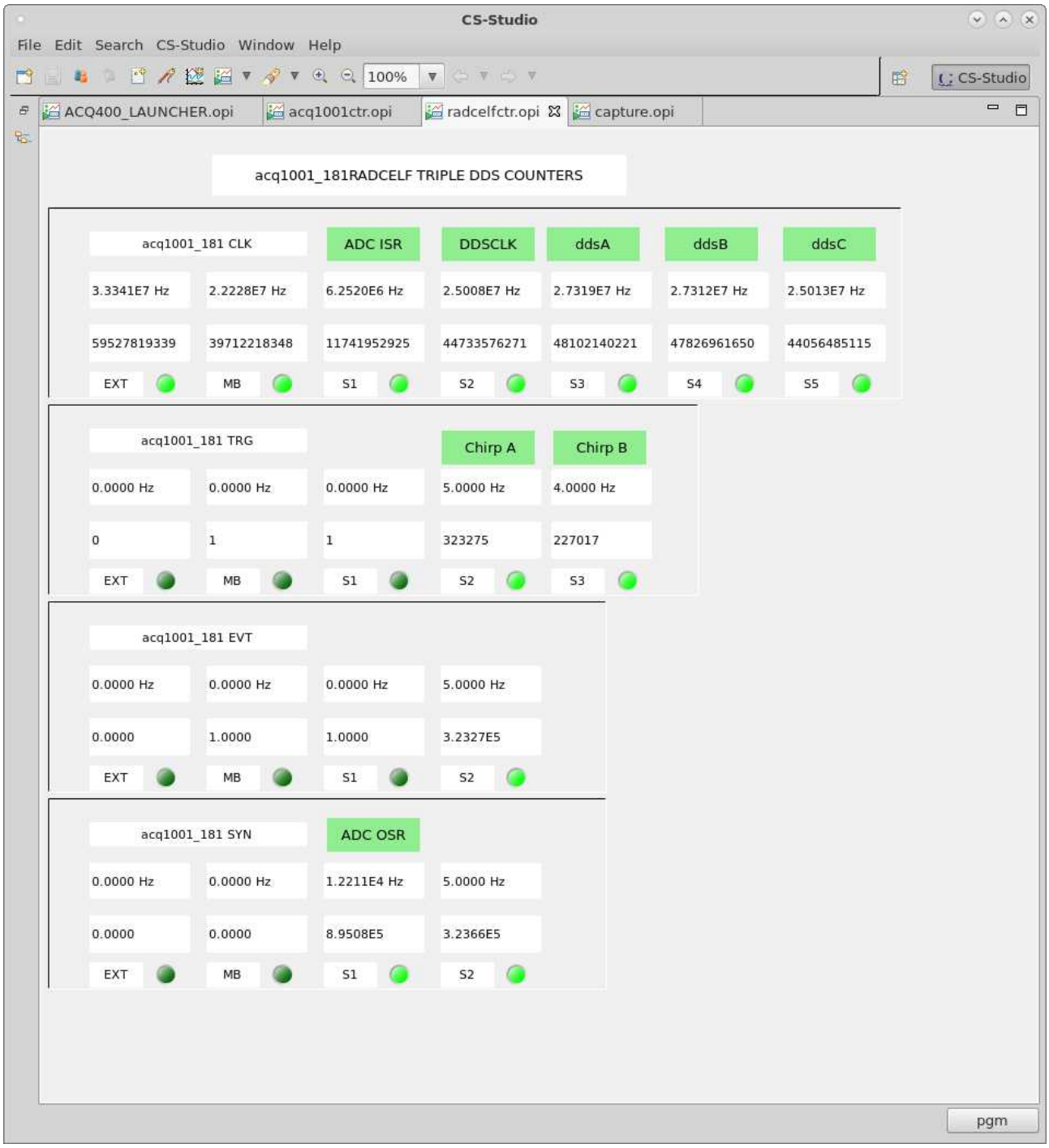


7.2 Monitoring CLKD



Web page shows current CLKD register settings.

7.3 Monitoring Chirp : OPI



7.4 Remote monitoring

The frequency counters are aliased on site 2 with meaningful names:

```
[pgm@hoy4 acq400_hapi_tests]$ nc acq1001_181 4222
*SIG*
SIG:DDS:A:CHIRP:COUNT 595852
SIG:DDS:A:CHIRP:FREQ 0
SIG:DDS:A:FREQ 0
SIG:DDS:B:CHIRP:COUNT 77397
SIG:DDS:B:CHIRP:FREQ 0
SIG:DDS:B:FREQ 0
SIG:DDS:C:FREQ 0
SIG:DDS:INP:FREQ 25004591
```

The counters may be cleared from this knob:

```
clear_counts 1
```

Clearing the counts is useful to track “number of chirp cycles from start”.

The DDS and CLKD system may be restored to power up state:

```
RADCELF_init 1
```

8 Appendix

8.1 Software Install

8.1.1 Install Fresh Release

acq4xx-477-20160826145633.tgz or newer

Customize after install as follows:

```
mv /mnt/packages.opt/21-python-* /mnt/packages
mv /mnt/packages.opt/35-radcelf* /mnt/packages
```

8.1.2 U-Boot settings.

```
Connect serial console
Press <SPACE> within 3s of power up
from the u-boot prompt:
u-boot> setenv devicetree_image dtb.d/acq1002r.dtb
u-boot> saveenv
u-boot> boot
```

8.1.3 Boot log

```
Press the spacebar to stop ACQ2006 autoboot:  0
Copying Linux from SD to RAM...
Device: zynq_sdhci
Manufacturer ID: 3
OEM: 5344
Name: SS04G
Tran Speed: 50000000
Rd Block Len: 512
SD version 3.0
High Capacity: Yes
Capacity: 3.7 GiB
Bus Width: 4-bit
reading uImage
3084880 bytes read in 484 ms (6.1 MiB/s)
reading dtb.d/acq1002r.dtb
11050 bytes read in 27 ms (399.4 KiB/s)
reading uramdisk.image.gz
6864673 bytes read in 1057 ms (6.2 MiB/s)
## Booting kernel from Legacy Image at 03000000 ...
```

```
   Image Name:   Linux-3.14.2-pgm-xilinx-00048-g5
   Image Type:   ARM Linux Kernel Image (uncompressed)
   Data Size:    3084816 Bytes = 2.9 MiB
   Load Address: 00008000
   Entry Point:  00008000
   Verifying Checksum ... OK
## Loading init Ramdisk from Legacy Image at 02000000 ...
   Image Name:   D-TACQ ACQ400 RAMDISK
   Image Type:   ARM Linux RAMDisk Image (gzip compressed)
   Data Size:    6864609 Bytes = 6.5 MiB
   Load Address: 00000000
   Entry Point:  00000000
   Verifying Checksum ... OK
## Flattened Device Tree blob at 02a00000
   Booting using the fdt blob at 0x02a00000
   Loading Kernel Image ... OK
OK
   Loading Ramdisk to 1f974000, end 1ffffee1 ... OK
   Loading Device Tree to 1f96e000, end 1f973b29 ... OK


Starting kernel ...


Uncompressing Linux... done, booting the kernel.
[    0.000000] Booting Linux on physical CPU 0x0
[    0.000000] Linux version 3.14.2-pgm-xilinx-00048-g547b3e2-dirty (pgm@hoy4) (gcc version 4.7.2
(Sourcery CodeBench Lite 2012.09-104) ) #102 SMP PREEMPT Fri Aug 19 08:28:30 BST 2016
[    0.000000] CPU: ARMv7 Processor [413fc090] revision 0 (ARMv7), cr=18c5387d
[    0.000000] CPU: PIPT / VIPT nonaliasing data cache, VIPT aliasing instruction cache
[    0.000000] Machine model: Xilinx Zynq
[    0.000000] bootconsole [earlycon0] enabled
[    0.000000] Memory policy: Data cache writealloc
[    0.000000] PERCPU: Embedded 8 pages/cpu @eefdd000 s9088 r8192 d15488 u32768
[    0.000000] Built 1 zonelists in Zone order, mobility grouping on.  Total pages: 260624
[    0.000000] Kernel command line: console=ttyPS0,115200 root=/dev/ram rw earlyprintk
hostname=acq1001_068
...


[    0.554967] TCP: reno registered
[    0.558229] UDP hash table entries: 512 (order: 2, 16384 bytes)
[    0.564198] UDP-Lite hash table entries: 512 (order: 2, 16384 bytes)
[    0.571146] NET: Registered protocol family 1
[    0.575727] RPC: Registered named UNIX socket transport module.
[    0.581539] RPC: Registered udp transport module.
[    0.586367] RPC: Registered tcp transport module.
[    0.591054] RPC: Registered tcp NFSv4.1 backchannel transport module.
```

```
[    0.597788] Trying to unpack rootfs image as initramfs...


Starting rcS...
++ Mounting filesystem
[    2.032716] FAT-fs (mmcblk0p1): Volume was not properly unmounted. Some data may be corrupt.
Please run fsck.
++ Setting up mdev
++ Setting hostname
++ Start Lo
++ Networking .. assigning serial console, use CTRL-C to break
+++ Starting dhcp daemon [default]
udhcpc (v1.22.0.git) started
Sending discover...
Sending discover...
[    7.206243] xemacps e000b000.ps7-ethernet: Set clk to 124999998 Hz
[    7.212347] xemacps e000b000.ps7-ethernet: link up (1000/FULL)
Sending discover...
Sending select for 10.12.196.240...
Lease of 10.12.196.240 obtained, lease time 21600
deleting routers
route: SIOCDELRT: No such process
adding dns 10.12.196.10
+++ dhcp good, remove failsafe 621
udhcpc (v1.22.0.git) started
Sending discover...
Sending select for 10.12.196.240...
Lease of 10.12.196.240 obtained, lease time 21600
deleting routers
route: SIOCDELRT: No such process
adding dns 10.12.196.10
++ Starting http daemon
++ Starting ssh daemon
[   10.751110] random: sshd urandom read with 25 bits of entropy available
++ Loading packages from /mnt/packages
++ Loading Package 02-tcl-1311231158.tgz
++ Loading Package 03-acq400_common-160408172105.tgz
acq400_common.init
++ Loading Package 04-custom_pmod-1504271257.tgz
+++ custom_pmod.init 01
+++ custom_pmod.init SITE:1 FRU /mnt/local/CUSTOM_PMOD_1_E43510999.fru
+++ custom_pmod.init 99
+++ model acq1002 load support
++ Loading Package 05-acq1002-160227231458.tgz
++ acq400_init_gpio_common begin
ZCLK DISABLE
```

```
HDMI sync setting /dev/gpio/CLK/SYNC_SET_IN as INPUT 0
HDMI sync setting /dev/gpio/CLK/SYNC_SET_OUT as OUTPUT 1
++ acq400_init_gpio_common end 01
++ lamp test 01
++ lamp test 99
++ leds all clear now
++ acq400_init_gpio_common end 99
++ acq1002_init_gpio done
+++ fmc-scan using /mnt/local/fmc-scan.conf
++ decode FRU EEPROM site 1 OK
++ decode FRU EEPROM site 2 OK
..
myspec ACQ1001TOP_02_69 filespec ACQ1001TOP_02_69
+++ identical bitfile
load FPGA /mnt/fpga.d/ACQ1001_TOP_02_69.bit.gz
++ Sites populated: 2 ALL GOOD
set.sys /dev/gpio/CLK/FP_CLK_OE 0
set.sys /dev/gpio/CLK/FP_OE 0
set.sys /dev/gpio/CLK/OSC_X_CLK_OE 0
set.sys /dev/gpio/CLK/OSC_X_OE 0
set.sys /dev/gpio/CLK/ZCLK_OE 0
++ Loading Package 06-procServ-1505022121.tgz
++ Loading Package 10-acq420-160824230114.tgz
acq420.init B1010
changing loglevel to debug
[   20.612402] random: nonblocking pool is initialized
[   20.643010] dma-pl330 f8003000.ps7-dma: pl330 driver hacked by PGM 1203
[   20.649768] dma-pl330 f8003000.ps7-dma: mcode_cpu:0xf0508000 bus:0x2d69a000 c:8 sz:512
totsize:4096
[   20.680144] dma-pl330 f8003000.ps7-dma: Loaded driver for PL330 DMAC-2364208
[   20.687322] dma-pl330 f8003000.ps7-dma:   DBUFF-128x8bytes Num_Chans-8 Num_Peri-4 Num_Events-16
for debug:
echo file acq400_drv.c +p > /sys/kernel/debug/dynamic_debug/control
echo file acq400_sysfs.c +p > /sys/kernel/debug/dynamic_debug/control
insmod /usr/local/lib/modules/acq420fmc.ko bufferlen=1048576 nbuffers=512 good_sit[   20.714249] D-
TACQ ACQ400 FMC Driver 3.051
es=1,2
[   20.722094] acq420 40000000.acq2006sc: site:0 GOOD
[   20.727230] acq420 40000000.acq2006sc: failed to find 3 IRQ values
[   20.733383] acq420 40000000.acq2006sc: About to read MODID from f0560000
[   20.740109] acq420 40000000.acq2006sc: Device MODID 81000022
[   22.198666] acq420 40000000.acq2006sc: setting nbuffers 512
[   22.204164] acq420 40000000.acq2006sc: acq400_createSysfs()
[   22.210708] acq420 40010000.acq400fmc: site:1 GOOD
[   22.215438] acq420 40010000.acq400fmc: About to read MODID from f0580000
```

```
[   22.222129] acq420 40010000.acq400fmc: Device MODID 02000029
[   22.227786] acq420 40010000.acq400fmc: ACQ435 device init
[   22.233140] acq420 40010000.acq400fmc: acq400_createSysfs()
[   22.239153] acq420 40020000.acq400fmc: site:2 GOOD
[   22.243879] acq420 40020000.acq400fmc: About to read MODID from f05a0000
[   22.250581] acq420 40020000.acq400fmc: Device MODID 69000001
[   22.256229] acq420 40020000.acq400fmc: rad_celf_init_defaults()
[   22.262092] acq420 40020000.acq400fmc: acq400_createSysfs()
+++ build_site0 model acq1002r
Calibration default installed site:1 /usr/local/cal/ACQ435ELF-defcal.xml
build_knobs_device site:1 mtype:2
WORKAROUND: clear FIFERR
set clkdiv 8 get.sys /dev/acq400.1.knobs/clkdiv 8
WARNING: no calibration or default found for site:2 type:RAD
build_knobs_device site:2 mtype:69
/usr/local/bin/set.fanspeed created
[   23.225878] acq420 40000000.acq2006sc: store_fan_percent:10 write 00000a03
+++ build nodes site 2
+++ build nodes site 1
+++ build nodes site 0
++ Enable analog power..
/usr/local/bin/procServ -p /var/run/knobs0.pid --restrict -q 4820 /usr/local/bin/acq400_knobs -s 0
/usr/local/bin/procServ -p /var/run/knobs1.pid --restrict -q 4821 /usr/local/bin/acq400_knobs -s 1
/usr/local/bin/procServ -p /var/run/knobs2.pid --restrict -q 4822 /usr/local/bin/acq400_knobs -s 2
/usr/local/bin/procServ -p /var/run/knobs3.pid --restrict -q 4823 /usr/local/bin/acq400_knobs -s 3
0: waiting for /var/run/acq400_knobs.0.pid
10: waiting for /var/run/acq400_knobs.0.pid
lrwxrwxrwx    1 root     root            38 Aug 26 14:23 /etc/acq400/0/sync_out_cable_det ->
/dev/acq400.0.knobs/sync_out_cable_det
++ Loading Package 20-httpd-1604070921.tgz
++ Loading Package 21-python-1509272010.tgz
++ Loading Package 22-inotifytools-130926090935.tgz
++ Loading Package 30-ai_monitor-1507161950.tgz
/usr/local/bin/procServ: spawning daemon process: 3497
++ Loading Package 33-libute-160711235303.tgz
++ Loading Package 34-at-1502112135.tgz
++ Loading Package 35-radcelf-1608261455.tgz
[   30.178135] D-TACQ AD9512 spi driver 2
[   30.202097] D-TACQ AD9854 DDS serial driver 0.9.2
[   30.216484] D-TACQ RADCELF Driver 1
[   30.219901] zynq_spi_goslow_usec_kludge -10acq480_hook_spi() ZYNQ SPI workaround
DEBUGS OFF
[   30.697370] ad9512 spi1.3: data: 00 450000
[   30.702184] ad9512 spi1.3: data: 00 5a0100
monitor_dds waiting for ioc
```

```
[   30.707450] ad9512 spi1.3: data: 00 3f0a00
[   30.713647] ad9512 spi1.3: data: 00 400100
.
[   30.830780] ad9512 spi1.4: data: 00 5a0100
now hit the FPGA knobs
++ Loading Package 39-transient-1606191315.tgz
++ transient runs at 39 BEFORE ioc to use knobs restart
[   30.897147] acq420 40000000.acq2006sc: a400fs_add_site() 01 site:0 sb:ed61a800
[   30.904316] acq420 40000000.acq2006sc: a400fs_add_site() 99 site:0
[   30.911241] acq420 40000000.acq2006sc: a400fs_add_site() 99 site:0
[   30.917404] acq420 40010000.acq400fmc: a400fs_add_site() 01 site:1 sb:ed61a800
[   30.924575] acq420 40010000.acq400fmc: a400fs_add_site() 99 site:1
[   30.930893] acq420 40010000.acq400fmc: a400fs_add_site() 99 site:1
sitelist: 1
sites: 1
setting NCHAN=32 data32=1
/usr/local/bin/procServ: spawning daemon process: 3791
+++ transient set /dev/acq400/data/.control COOKED=0 NSAMPLES=100000 NCHAN=4 TYPE=SHORT
/usr/local/bin/procServ: spawning daemon process: 3821
++ Loading Package 40-acq400ioc-1608102100.tgz
monitor_dds waiting for ioc
++ create EPICS db /tmp/st.cmd
load.records:load_400 site:1 model:2 name:acq435
HAS_ACQ43X Detected: 1
is_adc ds
HAS_ACQ43X Detected: 1
load.records:load_400 site:2 model:69 name:rad-celf
load.records:load_mb
load.records:load_mb_sites /dev/acq400.0.knobs 1 acq1001
HAS_ACQ43X 1
load.records: control COOKED=0 NSAMPLES=100000 NCHAN=4 TYPE=SHORT
load.records 99
/usr/local/bin/procServ: spawning daemon process: 4319
/usr/local/bin/procServ -p /var/run/knobs0.pid --restrict -q 4820 /usr/local/bin/acq400_knobs -s 0
/usr/local/bin/procServ -p /var/run/knobs1.pid --restrict -q 4821 /usr/local/bin/acq400_knobs -s 1
/usr/local/bin/procServ -p /var/run/knobs2.pid --restrict -q 4822 /usr/local/bin/acq400_knobs -s 2
/usr/local/bin/procServ -p /var/run/knobs3.pid --restrict -q 4823 /usr/local/bin/acq400_knobs -s 3
/usr/local/bin/procServ -p /var/run/knobs4.pid --restrict -q 4824 /usr/local/bin/acq400_knobs -s 4
/usr/local/bin/procServ -p /var/run/knobs5.pid --restrict -q 4825 /usr/local/bin/acq400_knobs -s 5
/usr/local/bin/procServ -p /var/run/knobs6.pid --restrict -q 4826 /usr/local/bin/acq400_knobs -s 6
/usr/local/bin/procServ -p /var/run/knobs7.pid --restrict -q 4827 /usr/local/bin/acq400_knobs -s 7
/usr/local/bin/procServ -p /var/run/knobs8.pid --restrict -q 4828 /usr/local/bin/acq400_knobs -s 8
0: waiting for /var/run/acq400_knobs.0.pid
10: waiting for /var/run/acq400_knobs.0.pid
```

```
/usr/local/bin/procServ: spawning daemon process: 5911
++ Loading Package 44-curl-140609223146.tgz
++ Loading Package 95-ttytools-1411212053.tgz
[   39.369439] usbcore: registered new interface driver usbserial
[   39.393958] usbcore: registered new interface driver ftdi_sio
[   39.407054] usbserial: USB Serial support registered for FTDI USB Serial Device
[   39.433316] usbcore: registered new interface driver pl2303
[   39.445327] usbserial: USB Serial support registered for pl2303
++ Loading Package 96-nettools-1411030925.tgz
++ calling /mnt/local/rc.user
++ rcS complete acq1001_068 14:23:29 up 0 min, 0 users, load average: 0.48, 0.11, 0.03
```

9 Appendix: Production Self-Tests

9.1 DDS

- run this script: /usr/local/CARE/radcelf-init-123
- Check this result:
- 20MHz becomes 18.75, 21.09 and 23.4 MHz on ddsA, ddsB, ddsC, respectively.

See Web Page 2.2

9.2 Aux DIO

- connect a 16 way header cable from J1 to J2.
- run this script:

```
acq1001_153> /usr/local/CARE/rad_aux_test
loopback J1 to J2
make all dX outputs, active high
OUT:0 0 0 0 0 0 0 0 BACK:0 0 0 0 0 0 0 0 PASS 0 0 0 0 0 0 0 0
OUT:1 0 0 0 0 0 0 0 BACK:1 0 0 0 0 0 0 0 PASS 2476 0 0 0 0 0 0 0
OUT:0 1 0 0 0 0 0 0 BACK:0 1 0 0 0 0 0 0 PASS 0 2478 0 0 0 0 0 0
OUT:0 0 1 0 0 0 0 0 BACK:0 0 1 0 0 0 0 0 PASS 0 0 2476 0 0 0 0 0
OUT:0 0 0 1 0 0 0 0 BACK:0 0 0 1 0 0 0 0 PASS 0 0 0 2476 0 0 0 0
OUT:0 0 0 0 1 0 0 0 BACK:0 0 0 0 1 0 0 0 PASS 0 0 0 0 2485 0 0 0
OUT:0 0 0 0 0 1 0 0 BACK:0 0 0 0 0 1 0 0 PASS 0 0 0 0 0 2493 0 0
OUT:0 0 0 0 0 0 1 0 BACK:0 0 0 0 0 0 1 0 PASS 0 0 0 0 0 0 2480 0
OUT:0 0 0 0 0 0 0 0 BACK:0 0 0 0 0 0 0 0 PASS 0 0 0 0 0 0 0 0
PASS: 9/9 PASS
```

9.3 USB

Connect USB stick and confirm it's available.

It's necessary to set bConfigurationValue to force power up.

```
echo 1 > /sys/bus/usb/devices/1-1.1/bConfigurationValue
mkdir /sd
mount /dev/sda1 /sd
```

9.4 Check all signal outputs

Run the dual chirp and use a scope to monitor (not covered in this guide)