RCC2 Test Proceedures

1. Power-up Test
    1. install board in VME crate (6U style or 6U section of 9U) and turn on power
    2. check that “cfg” LED (D4) on front panel and D3 LED on board near U47 are lit (red)
    3. Using a DVM check dc voltages on TP4(5V), TP5(3.3V), TP6(2.5V), TP7 (1.2V)
        1. alternatively U11-pin2(5V), U11-pin4(3.3V), U14-pin4(2.5V), U18-pin4(1.2V)
2. Loading with VME (Control) FPGA (U3)
    1. add jumpers to VME Base Adrs header for 0x18 (000X X000)
    2. turn on crate (if not already on)
    3. Connect Xilinx Platform Cable USB II to PC. (LED will light amber if driver installed)
    4. Plug Xilinx cable onto 6 pin header (J5) on RCC2 near U3. Following labels etched into the board (gray wire will not be connected). LED will turn green when properly connected.
    5. on PC launch “Start Menu”->”Programs”,”Xilinx”->”Accessory”->”IMPACT”
    6. in IMACT
        1. cancel update (or not)
        2. choose new project (if necessary)
        3. use “Configure via Boundry Scan (JTAG)” and hit “finish”
        4. select mcs file (F:/desktopbackup/RCC2/XXXX.mcs)
        5. put mouse on picture of prom and right click
        6. choose program (check “verify”, “erase before programming”, & “load FPGA”)
        7. unplug JTAG cable
    7. power cycle VME crate. Check that LED (D3) on board goes out.
    8. open window to vme processor in crate
        1. use “m” command to query 0x18000000XXXX. Should readback 0xabcdmmdd (“mmdd” date of revision) if FPGA was properly configured
3. Loading computational FPGA (U47) via JTAG connection
    1. turn on crate (if not already on)
    2. Connect Xilinx Platform Cable USB II to PC. (LED will light amber if driver instalLED)
    3. Plug Xilinx cable to 6 pin header (J14) on RCC2 near U48. Following labels etched into the board (gray wire will not be connected). LED will turn green when properly connected.
    4. on PC launch “Start Menu”->”Programs”,”Xilinx”->”Accessory”->”IMPACT”
    5. in IMACT
        1. cancel update (or not)
        2. choose new project (if necessary)
        3. use “Configure via Boundry Scan (JTAG)” and hit “finish”
        4. select mcs file (F:/desktopbackup/RCC2/XXXX.mcs)
        5. put mouse on picture of prom and right click
        6. choose program (check “verify”, “erase before programming”, & “load FPGA”)
    6. unplug JTAG cable
    7. if successful, power cycle crate. “cfg” LED on front panel should now go out.
    8. Use scope, check that 10XXXX MHz clock signal present on front panel lemo “clk out”
4. Loading computational FPGA via SP1 connection
    1. turn on crate (if not already on)
    2. Connect Xilinx Platform Cable USB II to PC. (LED will light amber if driver instalLED)

3. using RJ11(6) adapter to Xilinx Platform Cable connect to “SPI” RJ11 socket on RCC2 front panel (J11).
4. on PC launch “Start Menu”->”Programs”,”Xilinx”->”Accessory”->”IMPACT”
5. in IMPACT
    1. cancel “update” (or not)
    2. choose “create a new project (.ipf)” (or not)
    3. check “prepare a PROM File”, click “next”
    4. target “3rd Party SPI PROM”, checksum “FF”, Prom File name “spi_flash”, location “XXXX”, click “next”
    5. select PROM density “64M”, click “next”
    6. in new “Add Device” window, click “okay”
    7. in new “Add Device” browser select FPGA bitstream (.bit) file to add to SPI PROM. Click “open”
    8. click “no” when asked to add another device; then click “okay” to continue
    9. invoke “Generate File” in IMPACT window
    10. place cursor over picture of SPI PROM. Right click. Choose “Direct SPI Configuration”
    11. should take ~1 min to load.
6. if successful, power cycle crate. “cfg” LED on front panel should now go out.
7. Use scope, check that 10XXXX MHz clock signal present on front panel lemo “clk out”

5. Check Clock outputs
    1. use scope to check for 10 MHz clock signal present on front panel lemo “test pulse”
    2. use scope probe (and 96pin VME-DIN to 64pin IDC cable) to probe clock outputs to RCF (signals ch#_4 on pins XXXXA5, A10, A15, A20, A25, C5, C10, C15, C20, C25)
    3. Check Global Delay
        1. attach scope probe to raw (un-delayed) clock signal (TPXXXX)
        2. attach scope to “Clk out” signal
        3. adjust global delay (use “m 0x18000030XXXX” to set value 0x14XXXX)
        4. leading edges of un-delayed clock and “Clk out” should shift by 20XXXX ns.
    4. recheck after changing delay settings on each ch#_4 via the processor using “m 0x18000040XXXX” to deposit a value of XXXX (in ns)
6. Check Latch output
    1. Attach scope probe to a “latch” signal, ch#_XXXX (AXXXX) on 96pin VME-DIN to 64pin IDC cable.
    2. Set scope to “single trigger” mode. Drive latch signal on RCC2 (use “m 0x18000024XXXX” to write value of 1XXXX). should observe a pulse 3-4 clock tics (~330ns).
7. Check Run/Stop outputs
    1. use scope connected to “Run/Stop” lemo on panel and toggle run/stop via the processor using “m 0x18000020XXXX” to deposit values of 0 and 1. should observe transitions on scope.
    2. repeat 7.1 using a scope probe (and 96pin VME-DIN to 64pin IDC cable) to probe ch#_XXXX outputs to RCF (AXXXX) (see Step 4.2)
8. Check Clock Inputs
    1. Check TTL input
        1. set up pulser to generated ~10MHz TTL based signal.
        2. Input pulser signal to “Clk in” on RCC2 front panel
        3. use scope to probe “Clk Out” lemo on front panel

4. via processor, toggle RCC2 clock source using “m 0x1800000cXXXX” alternately depositing values of 0 (local oscillator) and 1XXXX (TTL clock input). Should observe change on scope and numeric display on front panel of RCC2.
5. with TTL clock input selected, set RCC2 to revert to local oscillator on error (use “m 0x18000010XXXX” to set value 0XXXX) remove pulser input to “Clk in” lemo. “Clk Out” should revert to local oscillator
6. via processor use “m 0x18000018XXXX” to check that error flag was set. Reset error (use “m 0x1800001c” to write “1”).

2. Check “Stop on Error”
    1. reconnect TTL pulser input to “Clk in” lemo on RCC.
    2. Reset RCC2 to use TTL clock source (use “m 0x1800000cXXXX” to set value =1XXXX)
    3. Set RCC to automatically switch to local oscillator on error (i.e. stop on clock error) (use “m 0x18000010XXXX” to set value = 0XXXX)
    4. use scope to monitor “Run/Stop” lemo on RCC2 front panel
    5. Set RCC into run mode (use “m 0x18000020XXXX” and set value = 1XXXX)
    6. remove TTL pulse input to “Clk in” lemo. Should observe “Run/Stop” signal drop to “Stop” state. Check that front panel Run & Stop LEDs light appropriately
    7. Set RCC to NOT stop on clock error (use “m 0x18000010XXXX” and set value = 1XXXX)
    8. repeat 8.2.5->8.2.6, “Run/Stop” signal should remain in “Run” state and “Clk out” signal should revert to local oscilator
    9. With scope set to trigger on Run/Stop input going low and to single trigger mode, reconnect TTL clock to “Clk in” lemo and select TTL clock source (use “m 0x1800000cXXXX to set value 1XXXX).
    10. Set number of missed clock ticks to produce an error at 2 (use “m 0x18000014XXXX” to set values 2XXXX)
    11. put RCC2 into Run mode (use “m 0x18000020XXXX to 1XXXX)
    12. remove TTL input from front panel. Observe time difference between last clock pulse on clock line and the Run/Stop transition.
    13. Reset error condition (use “m 0x1800001c” to write “1”)
    14. Repeat steps 8.2.10->8.1.13 with number of missed clock ticks to produce error set to 10. should observe 7-9 clock transitions different than seen in 8.1.11
3. Check Control signal delay
    1. Repeat steps 8.2.1 to 8.2.5.
    2. use scope probe and 96pin VME-DIN to 64pin IDC cable to monitor run/stop signal on channel group 2XXXX (ch2XXXX_XXXX, AXXXX).
    3. Set delay on channel group 2XXXX via processor (use “m 0x18000044XXXX” to set value to 0x28XXXX or 40XXXXns)
    4. Set scope for single trigger
    5. remove TTL clock input source
    6. measure scope timing difference between edges of run/stop transitions for “Run/Stop” lemo signal and that of channel 2XXXX. Should be 40XXXXns. Reset clock error
4. Check Master/Slave setting
    1. Set clock source to local oscillator (use “m 0x1800000cXXXX to set value 0XXXX).
    2. Toggle Mode selection between master and slave settings (use “m 0x18000008XXXX to alternate between values of 0XXXX (master) and 1XXXX (slave). Observe that

changes are recorded in the register
3. reconnect TTL clock input source to “clk in” lemo on front panel. Set clock source to TTL clock. Toggle Mode selection between master and slave settings. THIS IS AN NOT A WORKABLE CONFIGURATION. You should observe that the mode register remains fixed in the master mode (0x18000008XXXX = 0) and can not be changed to slave mode.

5. Check PECL input
    1. use old RCC/RCF (or working RCC2/RCF) and connect 10pin IDC cable between RCF and 10pin input on RCC2 front panel
    2. use scope to probe “Clk Out” lemo on front panel
    3. via processor, toggle RCC2 clock source using “m 0x1800000cXXXX” alternately depositing values of 0 (local oscillator) and 3XXXX (RCF clock input). Should observe change on scope and in numeric display on front panel of RCC2
    4. with RCF clock input selected, remove 10pin IDC input to RCC2. “Clk Out” should revert to local oscillatory
    5. via processor use “m 0x18000018XXXX” to check that error flag was set.
    6. Reconnect 10pin IDC cable to RCC2 front panel. Attach scope probe to a “latch” signal, ch#_XXXX (AXXXX) on 96pin VME-DIN to 64pin IDC cable.
    7. Set RCC2 to slave mode (use “m 0x18000008XXXX to set value 1XXXX)
    8. Set scope to “single trigger” mode. Drive latch signal on old RCF (use “m 0x18000038” to write value of 1). should observe a pulse 3-4 clock tics (~330ns).
    9. Set RCC2 to master mode (m 0x18000008XXXX = 0XXXX)
    10. Set scope to “single trigger” mode. Drive latch signal on old RCF (use “m 0x18000038” to write value of 1). should observe no change to latch line.
    11. Use scope to probe Run/Stop line. Repeat 8.5.7 -> 8.5.10 using the RCC to toggle the Run/Stop line (use “m 0x1800003c” to alternate between values of 0(stop) and 1(run)). Should observe change to RCC2's Run/Stop lemo for when RCC2 is in slave mode but not master mode
6. Check RHIC twin-ax input
    1. use special 2pin berg to BNC twin-ax cable to connect 10pin IDC cable from RCF (see step 6.3.1) to “RHIC clk” twin-ax connector on front panel of RCC2.
    2. use scope to probe “Clk Out” lemo on front panel
    3. via processor, toggle RCC2 clock source using “m 0x1800000cXXXX” alternately depositing values of 0 (local oscillator) and 2XXXX (RHIC clock input). Should observe change on scope and on numeric display on front panel of RCC2.
    4. with RHIC clock input selected, remove 10pin IDC input to RCC2. “Clk Out” should revert to local oscillator
    5. via processor use “m 0x18000018XXXX” to check that error flag was set.
7. Check Fiber Clock input
    1. Requires working RCC2 and RCF board to use as master to feed RCC2 board being tested.
    2. Connect 50um ST-ST fiber between RCF board connected to working (master) RCC2 and “Rcv” ST connector on front panel of RCC2 being tested.
    3. use scope to probe “Clk Out” lemo on front panel
    4. via processor, toggle RCC2 clock source using “m 0x1800000cXXXX” alternately depositing values of 0 (local oscillator) and 4XXXX (RHIC clock input). Should observe change on scope and on numeric display on front panel of RCC2.

5. with RHIC clock input selected, remove 10pin IDC input to RCC2. “Clk Out” should revert to local oscillator
6. via processor use “m 0x18000018XXXX” to check that error flag was set. Reset error flag (use “m 0x1800001c” to write “1”)

9. Test VME protocals
    1. Test sysreset
        1. with RCC2 in VME crate, put RCC2 in run mode (use “m 0x18000020XXXX” to set value to 1XXXX).
        2. Push “sysreset” button on vme processor and hold.
        3. Both “cfg” LEDs (D3 & D4) should show red until “sysreset” button is released.
        4. Feed RCC2 “Clk in” with TTL pulser and set RCC2 to use this clock source (use “m 0x1800000cXXXX” to set value 1XXXX).
        5. Repeat 9.1.1->9.1.3 check that RCC2 comes back up in default mode (i.e. set to (and using) local oscillator (0x1800000cXXXX = 0) and in “Stop” mode (0x18j000020XXXX=0)).
        6. Install 2pin jumper on front panel. “cfg” (D4) light on front panel should appear until jumper has been removed.
        7. Repeat steps 9.1.4, then 9.1.1, then 9.1.6 check that RCC2 comes back in default mode.
    2. Test response to chainblock transfer
        1. install RCC2 & RCF2 in 9u VME crate with at least 2 DSM boards (preferably 3).
        2. feed clock signals from RCF2 to DSMIs via 10pin IDC cables.
        3. Follow standard suite of tests found in aquila.lbl.gov:/home/aquila/trg/dsm_test (see README) for running DSM configuration and chainblock transfer tests.
        4. During tests, observe that DSMs configure and run correctly and that RCC2 maintains steady clock and doesn't suffer any “hiccups” on run/stop line.
    3. Loading computational FPGA (U3) with VME
        1. in processor window move to “/scratch/amber/trg/RCC2”
        2. if not already loaded, load prom_programmerXXXX.o (ld <prom_programmer.oXXXX)
        3. Load mcs file in computational FPGA (prom_prog(0x18000000,”rcc2XXX.mcs”))
        4. if successful, power cycle crate. “cfg” LED on front panel should now go out.
        5. Use scope, check that 10XXXX MHz clock signal present on front panel lemo “clk out”
10. Set up ConnectCore
    1. connect cat6 cable between active ethernet switch (on lbl.gov subnet) and “Ethernet” RJ45 connector on RCC2 front panel
    2. connect USB to USB-micro cable between PC and “IP cfg” USB-micro socket on RCC2 front panel
    3. launch Digi ESP on pc to start the “Workspace Launcher”
    4. enter “XXXX” for location of the workspace and click “ok”
    5. click arrow on the right side to close the “welcome”window
    6. open a “Serial Terminal” in Digi ESP. select “Window”->”Show View”->”Terminal”. Should be displayed in bottom part of Workbench window
    7. open “Terminal Settings” dialog by clicking “Settings” button on “Terminal view's toolbar
    8. in “View Title” type “Serial Terminal”, in the “Connection Type” combo, select “Serial”
    9. Configure the “Serial Port” to use the device node to which the USB cable is connected (/dev/ttyUSB#).
    10. Do not change other values (e.g. 38400 baud, 8 data bits, 1 stop bit, no parity)

11. click okay to accept configuration
12. establish the connection by clicking the “Connect/Disconnect” button in “Terminal” view's tool bar
13. Configure IP address (in the boot loader shell type)
    1. setenv ipaddr 128.3.128.XXXX
    2. setenv netmask 255.255.252.0
    3. setenv serverrip 128.3.128.XXXX
    4. saveenv
14. Reset the target
15. try some basic linux commands (e.g. ls, cd, ifconfig, ping)
16. on the pc, launch a web browser and surf to ConnectCore ip number, 128.3.128.XXXX (see 8.13.1 above)
17. create a remote configuration
    1. select “Device Options” -> “Device Manager”
    2. in “Remote Configurations” tree, select “DigiEL”, click “new”. In “Name” edit box, enter the name “myTarget”
    3. on “General” tab, in the “IP Address” field enther IP of the target (128.3.128.XXXX) see 8.13.3 above
    4. on “File Transfer” tab select “Use FTP as default file transfer mechanism”. In “Authentication” select “trg” and <passwd>.
    5. Click on “Hardware”. Specify “Processor” as XXXX, “Module” as XXXX, and Base Board” as XXXX
    6. create a new application project.
        1. Select “File”->”New”->”Project” to diplay the “New Project” wizard. Select template and click “next”.
        2. Select “C programmed Hello World” and click “next”. Then click “finished”
        3.

11.