RCC2 Manual

July 2018 J.Engelage, E.G.Judd

General

The RCC2 is a 6U VME module (see Figure 1) that has been created to provide a centralized clock source for the STAR experiment. Its primary function is to receive the RHIC clock from CAD and distribute it, with the correct phases, to all electronic clients at STAR. Alternately, if no clock signal is available from CAD, a local oscillator on the RCC2 can provide the heartbeat for the experiment. To accomplish these tasks, the RCC2 has been designed with a Xilinx FPGA to control the assignment of inputs, output delays and other internal functions. The user controls these assignments by depositing desired values in specific registers that are defined through VHDL firmware coded for the RCC2 by Eleanor Judd. The firmware version running on the FPGA can be read from register 0 (0x0). The RCC2 also contains a standalone Linux system with a fully functional Ethernet interface and Web server. As such, it is possible to use these boards in a standalone application needing only a +5V (1.5A) supply without any supporting VME infrastructure required.

Inputs

The RCC2 has five (5) possible clock sources, and three (3) sources of control signals. The five clock sources are shown in Table 1. The clock sources are user selectable by writing the values to the Input Selection register as shown in Table 1.

Clock Source	location	0x0C register value
Local Oscillator (10MHz)	On board	0
External TTL (TTL clk in)	Front panel Lemo	1
RHIC clock (from V124)	Front panel twin-ax	2
10-pin PECL (from RCF2)	Front Panel IDC	3
Fiber (Not implemented yet)	Front Panel ST	4

Table 1: Available RCC2 clock sources

Figure 1

Control Sources

Control signals can be supplied over VME or Ethernet, via the 10-pin PECL input, or via the front panel fiber input. Only the 10-pin PECL (3) and fiber (4), which is not yet implemented, are available for running the RCC2 in slave mode. If the RCC2 is in slave mode and the 10-pin PECL clock is selected then the control signals that come with that **selected clock** will be passed on to all clients. If the RCC2 is in slave mode and the Fiber clock is selected then the control signals that come over the fiber with that **selected clock** will be passed on to all clients. In Master mode the user must supply the control signals over VME or Ethernet. Master or Slave mode is selected by writing to the Mode Selection register at address 0x08.

Control signals include Run/Stop, Address Latch, Halt, and Test. The Run/Stop control signal is used to place trigger electronics into Run or Load mode. Configuration of some clients (e.g. DSMs) can only be performed by the user when those clients are in Stop Mode. To assure that the Run/Stop signal is latched properly to the RCC2 client, it trails the rising clock edge by ~8ns. The clock delay was included in the design in case it was necessary to deal with setup-and-hold timing violations between the selected clock and the control signals. That feature turned out to be unnecessary so the clock delay is never used. The clock delay register should ALWAYS be set to zero (0).[1] Caveat: in practice this means that the signals output from the RCF can be phased by 60ns (an RCC2 delay of 0xB2) or more and have the Run/Stop signal latch at the client before it would if that delay was set to 0. Activating the Address Latch causes the current working address to be stored in a register on each of the client boards (e.g. 0xYY60001C for DSMs and 0xYY000158 for the TCU). The Address Latch is used to check address alignment on all trigger clients, and for other diagnostic tests of the DSMs and TCUs. The Halt signal was used in the past to stop data taking and freeze the trigger system if the readout time had exceeded 7ms and the data buffers were in jeopardy of being overwritten. The Halt signals function has since been superseded by the address counter (see the New Features section). The Test pulse is a new feature that can be used to strobe test pulse output from the RCC2. It expected to be used to strobe calibration LEDs on trigger detectors and for other diagnostic tests.

Outputs

The RCC2 distributes information in three (3) ways: from the front panel, via the VME backplane, and through Ethernet. There are two types of output information served from the front panel: visual and electronic. Visual output information displayed by LEDs on the front panel is listed in Table 2:

[1] The implementation of a clock distribution to TOF tray circuit in the TOF-DSMI boards reduced this hold time to zero. A special cable between RCF2 and TOF-DSMI was required to re-establish the ~8ns difference between clock and control signals.

Stop/Run	Indicates whether RCC2 broadcasting STOP (amber) or RUN(green) level to clients
CFG	Lights (red) if Control FPGA on RCC2 is not configured
Fuse	Lights (red) if power fuse is blown
Clock Src	Indicates which clock source was selected and being used (see Table 1 for Definitions)

Table 2: Front panel LEDs

Electronic output exiting the front panel on lemo connectors includes:

Clk Out	Clock signal in TTL format capable of driving 50ohms
Run/Stop	Run/Stop level in TTL format
test pulse	Emulated PMT-like pulse

Table 3: Front panel lemo outputs

The RCC2 front panel also contains a micro-B USB connector for RS-232 communication to set up the ConnectCore Linux board, an RJ-45 connector for 10/100 Base-T Ethernet communication to the ConnectCore Linux board, and an RJ-11 connector for externally configuring the Control FPGA and its SPI Memory using Xilinx Impact protocols.

Registers

Control information passed via the VME interface includes the delay settings for the 10 independent output channels shipped to the P2/J2 interface for RCF2 distribution (see Table 4). The current RCF2 makes identical A&B outputs for each of these 10 RCC2 channels (with delay settings) and fans them out onto 10 conductor IDC connectors in PECL format.

Global delay	0xYY000030	Delays selected clock & control signals in ½ns steps
Clock delay	0xYY000034	Delays the selected clock with respect to the selected control signals in 1/2 ns steps (not used. Set to 0)
Test pulse delay	0xYY000038	Delays output PMT-like pulse in ½ns steps
Group 0 delay	0xYY00003C	Delays output RCF2 channel 0 in ½ns steps
Group 1 delay	0xYY000040	Delays output RCF2 channel 1 in ½ns steps
Group 2 delay	0xYY000044	Delays output RCF2 channel 2 in ½ns steps
Group 3 delay	0xYY000048	Delays output RCF2 channel 3 in ½ns steps
Group 4 delay	0xYY00004C	Delays output RCF2 channel 4 in ½ns steps
Group 5 delay	0xYY000050	Delays output RCF2 channel 5 in ½ns steps
Group 6 delay	0xYY000054	Delays output RCF2 channel 6 in ½ns steps
Group 7 delay	0xYY000058	Delays output RCF2 channel 7 in ½ns steps
Group 8 delay	0xYY00005C	Delays output RCF2 channel 8 in ½ns steps
Group 9 delay	0xYY000060	Delays output RCF2 channel 9 in ½ns steps

Table 4: the thirteen RCC2 Delay Registers

Configuration information is also available via VME registers as described in Table 5. Note, the Run/Stop register (0xYY000020) reflects the state as specified by the onboard FPGA. It may not reflect the actual Run/Stop state while the RCC2 is in Slave mode with Input Selection of either 3 or 4.

ID: Name	Address	Functionality	Access
0: Firmware revision	0xYY000000	0xFFFFMMnn-Major(MM) & Minor(nn) revision	R
1: Firmware date	0xYY000004	0xFFFFmmdd-Month(mm) & Day(dd) of revision	R
2: Mode selection	0xYY000008	0 = Slave mode: 1= Master mode	R/W
3: Input Selection	0xYY00000C	Clock source (See Table 1)	R/W
4: Clock error behavior	0xYY000010	0= auto-switch on error: 1= no auto switching	R/W
5: Clock error period	0xYY000014	D(0:15) Clock period that produced an error	R
6: Clock error status	0xYY000018	D0= Clock period above high threshold D1= Clock period below low threshold	R
7:Reset clock error	0xYY00001C	D0: Reset	W
8: Run/Stop	0xYY000020	0= Stop (load) mode: 1= Run mode	R/W
9: Address Latch	0xYY000024	D0: Generate address latch pulse	W
11: Test Pulse Trigger	0xYY00002C	D0: Generate test pulse	W

Table 5: RCC2 Configuration Registers

Power up defaults

At power up the default clock source is the 10MHz local oscillator. Other default conditions include:

- Mode selection set to master (0x08 = 1)
- Clock error behavior set to automatically switch to local oscillator on error (0x10 = 0)
- Clock error definition set to produce an error if clock falls outside defined frequency bounds
- Run/Stop set to Stop (0x20 =0)
- Group Delays All 13 delay values described in Table 6 are set to 0 on power up.
- Clock Period limits are set to 0 and 10MHz

Before changing Mode Selection to the "slave" mode, the Input Selection (or clock source) must be set to a source that can also supply control signals (e.g. 10-pin PECL or Fiber). Any other setting will result in errors in slave mode and cause the unit to revert to "master" mode. If the PECL (#3) or Fiber (#4) clock source is not specified before Slave mode is selected, the control signals from the Master WILL NOT be distributed through the Slave. Failure to follow the proper sequence will result in the slave RCC2 distributing control signals from its own FPGA.

Errors

If an external clock is selected which is not present or is considered unstable then the Clock Error LED on the RCC2 front panel will light and the Clock error status register (0x18) will be set to a value of 1. The RCC2 also will revert to using its local oscillator if the Clock error behavior register (0x10) is set to 0. Note: in the event of an external clock error causing the unit to revert to running with the local oscillator, the digit display will be unchanged. An external clock is considered unstable if it falls outside user settable frequency limits. This is a departure from the old RCC (and earlier RCC2's v5.3 VHDL coding) where a clock was considered unstable only if it missed sending a specified number of clock pulses. A more complete description of this change is given in the New Features section below. The external clock source that the user selected will continue to be displayed, regardless of whether the RCC2 has been set to automatically switch to its local oscillator or not. The digit display will not change until the user actively requests a different clock input. To recover from this condition, a valid input clock must first be selected via the Input Selection register (0x0C) and then the Clock error status reset by writing a "1" to the Reset clock error status register (0x1C).

Procedure for starting run

A static binary file internal to the trigger system contains the low-level programming of the trigger hardware. At run start the information contained in this "Tier1" file is downloaded into the VME based boards on a crate by crate basis. Since the new RCC2 boards will be hosted in existing VME crates, the Tier1 file configuration information has been amended to include RCC2 parameters. The crate configuration files that make up the Tier1 file now contain code similar :

```
RCC_Number 1
RCC_BASE_ADDRESS   0x25000000
RCC_CLOCK_SRC      2
RCC_MASTER         1
RCC_STOP_RUN       1
RCC_NUM_TICKS      20
RCC_GLOBAL_DELAY   0
RCC_CLOCK_DELAY    0
RCC_DELAY 0 0 0 0 0 0 0 0 0 0
RCC_End
```

This RCC2 configuration information MUST appear at the beginning of those files.

Table 6a: table of RCC2s vs output clients

CHNL	L1(mast)	BBC	BBQ	BC1	BCW	BCE	EPQ|EQ3	MIX	MXQ
0A	BC1 RCC	QT1 RCC						TF102	
B	FMS RCC								
1A	BCE RCC	EQ3 RCC	BBQ BOC	EE005	BW013	BE013	EPQ BOC	EM202	MXQBOC
B	BCW RCC	EQ2 RCC		EE006	BW014	BE014		MT201	
2A		EP001	VPD TAC	EE001	BW009	BE009	EPQ TAC		
B	TCUR	EP002		EE002	BW010	BE010			
3A	BX201	ZD101		BC105	BW005	BE005		EQ101	EPQ RCC
B	ST201	VP101		BC106	BW006	BE006			
4A	TF201	BB101		BC101	BW001	BE001		MT101	
B	VT201	BB102		BC102	BW002	BE002		TF101	
5A	DQ301			EE009					P2P TAC
B	RAT								
6A	BBC RCC	EQ1 RCC	BBC TAC	EE007	BW015	BE015		FQ1 RCC	MTD TAC
B	BBQ RCC			EE008					
7A	MIX RCC	EP101	ZDC TAC	EE003	BW011	BE011		TF005	VPD TAC
B	MXQ RCC			EE004	BW012	BE012		TF006	
8A	BX202	EP003		EE101	BW007	BE007		TF003	
B	TCU			EE102	BW008	BE008		TF004	
9A	EM201			BC103	BW003	BE003		TF001	
B	FM201			BC104	BW004	BE004		TF002	

Table 6b: table of RCC2s versus client (continued)

CHNL	FMS	QT1	QT2	QT3	QT4	FPre	FPst	EQ1	EQ2
0A	QT1 BOC	QT3 RCC					Fpre BOC		
B	QT2 BOC								
1A	FM101								
B	FM103	QT2 RCC		QT4 RCC					
2A	FM009								
B	FM010								
3A	FM005								
B	FM006								
4A									
B									
5A	QT3 BOC	Fpst BOC							
B	QT4 BOC								
6A	FM104								
B	FM102								
7A	FM011								
B	FM012								
8A	FM007								
B	FM008								
9A	FpreBOC								
B									

Choose the name in the CFG.RCC_TAC column that corresponds to the detector whose TAC setting wants to be changed. Adjust the "phase" parameter to obtain the desired RCC2 Delay setting. The phase value is specified in 0.5ns units. While not specifically required the convention of setting the RCC2 VME addresses to 0x25 has been adopted. At present there are RCC2s installed in the L1, BBC, BBQ, BC1, BCW, BCE, MIX, MXQ and FMS crates. Table 6 contains the RCF2 output channel assignments for those crates.

New Features

Test Pulse

The RCC2 has several new features and improvements from the original RCC board. A TTL clock output on the front panel of the new RCC2 allows easy access to the experiment clock pulse for timing in inputs, triggering scopes and driving other equipment. A new "test pulse" circuit has been added to mock up a photomultiplier signal in sync with the experiment clock. This mock pulse can be input to QT boards to simulate trigger detector pulses so as to test the entire chain of trigger electronics from QT input to TCU output. The mock pulse can be fired as a single shot (0x6C=2) or set to "free running" (0x6C=1). Both the magnitude (12bits) and frequency (16bits) is user settable from -50mV to -1V and 1Hz to 588KHz (see Tables 7 & 8). Default condition is for the mock pulse to be turned off (0x6C=0).

Pulse Height (mV)	0x64 Register Value
-45mV	0x8F (lower limit)
-100mV	0x13F
-250mV	0x2DF
-500mV	0x54F
-1120mV	0xCFF(upper limit)

Table 7: Test pulse height vs Register value

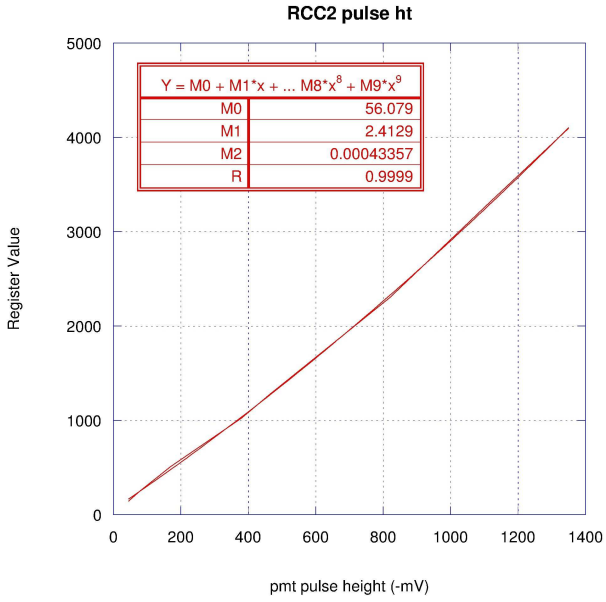

Figure 3: pulse height as a function of reg value

If a specific pulse height not given in the table is required use the plot in figure 3 or the following equation to obtain the register value to produce the desired pulse height:

Register Value = 433.6E-6(pulse ht)**2 + 2.413(pulse ht) + 56

Pulse Frequency	0x68 Register Value
1kHz	0xFD80
5kHz	0xFF82
20kHz	0xFFE0
100KHz	0xFFF9
588KHz	0xFFFE(upper limit)

Table 8: Test pulse frequency vs Register Value

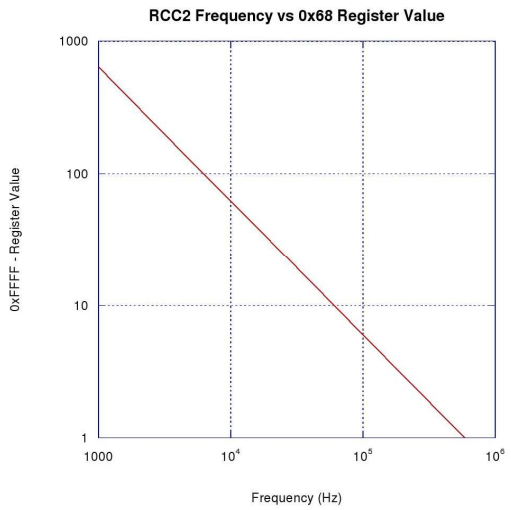

Figure 4: pulse frequency vs register value

If the frequency desired is not given in the table it can be estimated from the plot in figure 4 or obtained using the following equation:

log(0xFFFF-RegVal) = 5.8468 - 1.0135*log(frequency)
Register Value = 0xFFFF - 10**[5.8468 - 1.0135*log(frequency)]

Address Monitor

An Address monitoring feature was added in version 6.01 of the RCC2 control code to help identify any data overwrites. This monitoring features a 32 bit counter, composed of two registers, 0x78 and 0x7C, that contain the LSB 16 bits and MSB 16 bits, respectively. The monitor also contains a third register, 0x74, to control resetting the counter and recording instantaneous counter values. At configuration, both RCC2 local address and registers are cleared by writing a "1" to 0x74. When the run starts, the count will begin to increment by one every tick of the RHIC clock. When an event readout is initiated, the DSM or QT boards are read in each crate. At the end of readout, for each crate that contains an RCC2, a "2" is written to the RCC2 Address Control register (0x74) to save the instantaneous value of that counter. The values in the Local Address registers (0x78 & 0x7C) for each crate are then read and placed in the localClock variable of the trigger data block before being shipped to L2 over STP. The L2 machine receives all counter values from those trigger crates as well as the TCU counter from the L1 crate. The 32 bit TCU counter readout marks when the event was triggered. An L2 analyzer takes the difference between the TCU counter value and the crate counter values (accounting for possible rollovers) and places those values in the LocalClocks array. Any LocalClocks element greater than 7ms, the depth of the DSM memory buffer, will cause the L2 analyzer to log a notice of possible event corruption. A similar message will appear on the DAQ monitor to alert the shift crew of a possible problem. A graph (see figure 5) of each crate's counter values for each run can be viewed at http://online.star.bnl.gov/rcc

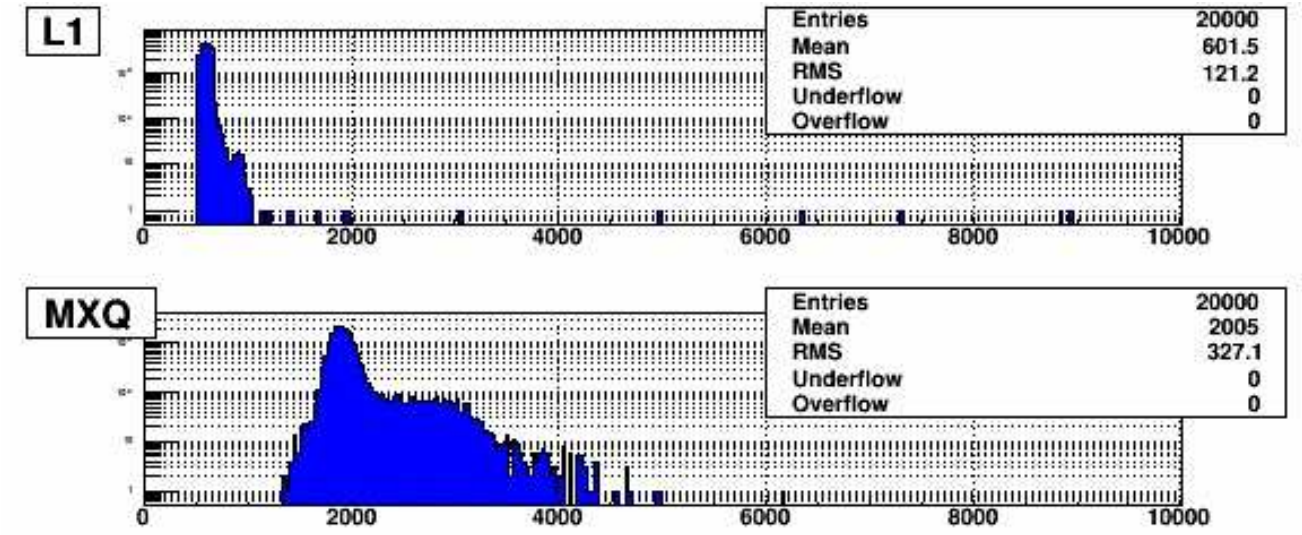

Figure 5: Example plot of L1 and MXQ crate readout times from run 17171001

Clock Monitoring

In 2015 an electronic "Clock Cleaner" module was added to the external RHIC clock (0x0C = 2) circuit. This unit successfully reduced the noise in the clock signals supplied by the BNL CAD group from 200-300ps to just above 100ps. However, the PLL circuit employed to achieve this goal had the side effect continuously generating output even in the absence of a RHIC clock signal at its input. This rendered the RCC's method of determining an external clock error solely on the absence of an input clock signal invalid. To correct for this change and better identify clock anomalies the addition of period monitoring logic was added to the RCC2's VHDL coding. The new monitoring logic takes the incoming clock and divides it down by 4096 to get a slow clock. The logic checks the period of the slow clock using the faster 66MHz local oscillator (period = ~15.15ns). For a 9.4MHz clock input this slow clock value would be 0x705A ns (i.e. 4096/9.4MHz/15.15). This value is stored at each input clock tick in the RCC2's memory at address 0x88 and then compared to the user set limits. For example: setting frequency bounds to a minimum of 9.2Mhz and a maximum of 9.6MHz would entail setting 0x80 to 0x6E03 ns and 0x84 to 0x72CB. If the momentary clock value should fall outside these limits, the clock error LED and flag will be set and the offending value will be written in the Clock error period register (0x14)

ConnectCore

The RCC2 incorporates a ConnectCore 9P-9215 network-enabled ARM9 module (http://www.digi.com/products/model?mid=3187) that allows the board to be operated independent of a VME processor. The 9P-9215 contains an embedded linux system that can be configured via a micro usb connector on the RCC2 front panel using RS-232. To setup the Ethernet connection set your RS-232 device for 38400 baud, 8 data bits, 1 stop bit, and no parity.
Once communication has been established, configure IP address (in the boot loader shell type)

```
nvram set network ip1= 128.3.128.129
nvram set network netmask1=255.255.252.0
nvram set network gateway=128.3.128.1
nvram set network dns1=128.3.34.186
nvram save
```

Issue a "reboot" to reset the ConnectCore to the new values. Once the ConnectCore Ethernet is configured and cabled the RCC2 is accessible directly via web or ssh from the Ethernet. C based code can be cross compiled using the DigiEl-4.2 cross compiler on either of the startrg machines.

1. 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"

Control of TAC stop on TAC2002 and TAC2015 implementations

For the convenience of the TAC operators, the RCC_DELAY values (i.e. the phase of the clock signals assigned in the Tier1 files) can be overwritten by the Run Control operator. To accomplish this, from the RunControl window select “edit configuration”. From the “edit configuration” GUI choose “TCD_setup_name” “details”. A screen like that shown in Figure 2 should appear. This feature has been superseded in the newer QT32C boards with onboard TACs, where the RCC2 delays are manipulated via the RunControl trigger labels table.

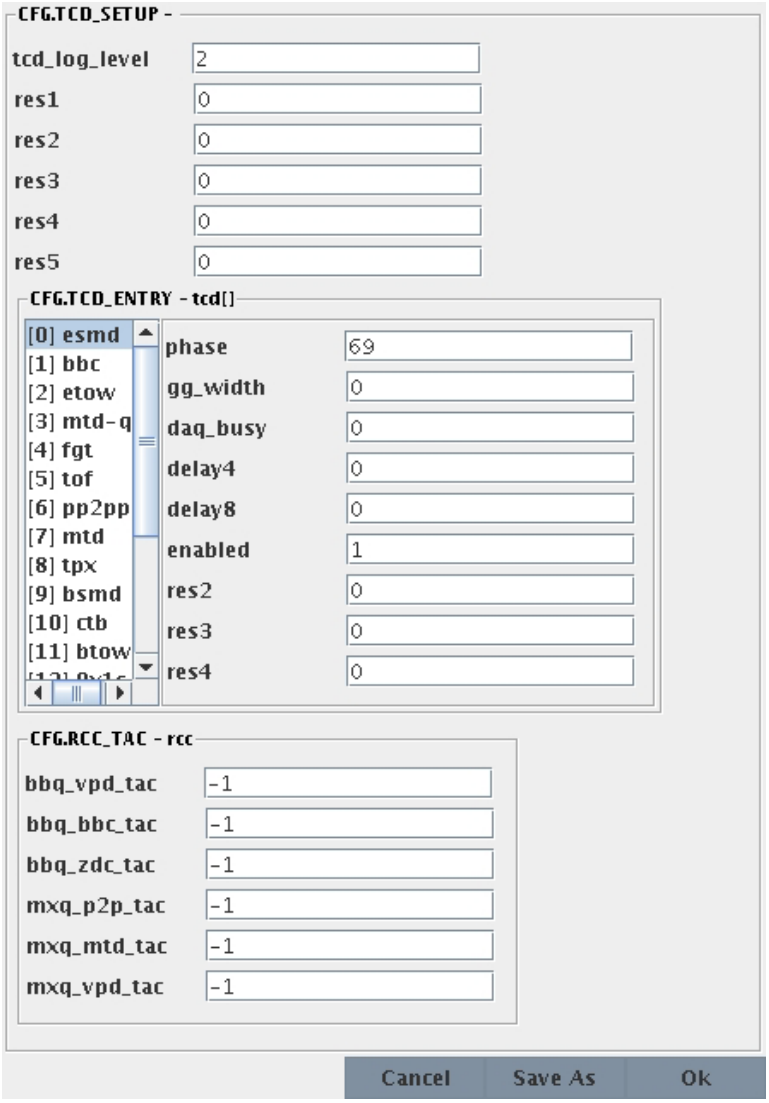

Fig 2: RunControl GUI for changing RCC2 delays