Physics and Astronomy Dept.
University of British Columbia
MCE Project

08 July 2008
SC2-ELE-S582-501

MCE Readout Card Technical Description

Revision History

Rev.	*Date*	*Description of change*
2.5	20080708	Added sample_dly for the timing diagrams, raw-mode and preliminary continous-coadd description. Updated data_mode table, section 3.1, readout control loop and the timing of the feedback loop, added Fig. 1 PID servo diagram.
2.4	20080421	fb_dly > 7 in normal operation, and fb_dly>18 when flux-jumping on, clarified captr_raw description.
2.3	20080129	Added flux-jump description, timing diagram, under Firmware section
2.2	20080109	Added firmware description, updated Abstract.
2.1	20070716	MA: added background/General Description
2.0	20070709	WH: modified after
1.11	20040111	BNB/WH: Initial release

Abstract

The Read-out Card (RC) of the the Multi-Channel Electronics (MCE) provides the read-out warm electronics for array of detectors based on transition-edge sensor (TES) bolometers and SQUID amplifiers. An RC has 8 readout channels that each controls a column of the array of detectors. Each readout circuitry consists of a preamp stage followed by a 14-bit 50MS/s video analog-to-digital converter (ADC) to sample an input signal, a 14-bit DAC to output the calculated SQ1 feedback, two 16-bit serial DACs to output the series-array (SA) bias and cable offset, respectively. The on-board FPGA firmware provides digital filtering of the signal. The input signal, the calculated feedback, the low-pass filtered feedback signal, or a combination of these data, are transmitted over the MCE backplane (2 point-to-point LVDS lines) to the clock-card (CC) where they are rearranged and transmitted to the PC that controls the MCE.

1. General Description

1.1 Background

In a SCUBA2-like array of detectors, the radiation sensitive elements are TES bolometers. When voltage-biased at their superconducting transition temperature, an incoming radiation produces a change in their resistance and consequently a change of current. Each of them is inductively coupled to a first-stage of SQUID amplifiers used as low-noise amplifiers. The SQUIDs are non-linear devices that their response to the increasing input signal is approximately a sine wave. Therefore, a feedback signal is calculated based on the measured output, i.e., the change in magnetic flux in the SQUID ammeters is countered by a first-stage feedback bias arriving from the RCs of the MCE [1]. The feedback loop is provided by a digital proportional-integral-differential (PID) flux-locked loop that calculates the correct flux to keep the whole amplification chain in a linear regime. [Delete?? The acquired signal will thus be represented by the first-stage feedback signal while the output signal, amplified by two further stages of SQUID amplification is actively kept to zero by the PID loop itself. Delete??]

1.2 Functional Description

The RCs account for 4 of the 10 cards in each MCE subrack. The RCs are sub-array data acquisition cards. They are identical in manufacturing and duties and read signals from columns 0-7, 8-15, 16-23, and 24-32 respectively for RC1, RC2, RC3 and RC4. Each card will synchronously:

(a) acquire SQUID data using 8 PID loop for a maximum of 32 columns per MCE,
(b) filter the data,
(c) transmit the data as frame fragments to the Clock Card (CC).

Each RC PID loop will be coupled to a column of up-to-41 first-stage SQUIDs (SQ1), but will only digitize the output of one SQ1 at a time. This will be achieved by using the Address Card (AC) to null the contributions from all SQUIDs in the column, except for the one that is to be sampled. The sampled SQUID will be actively biased by the AC, and multiple readings of its analog current, amplified by two further stages of SQUID amplification, will be digitized and co-added before the AC activates the next SQUID in each column. To read out the next SQUID, the AC will nullify the output of the present SQUID, and actively bias the next one.

Each PID loop will be switched from pixel-to-pixel (or more precisely row-to-row) in a column, at a line rate of approximately 800kHz set by the L/R of the cryogenic cables. During each row dwell-time, the PID loop will:

(a) digitize the SQUID output (SSA_SIG),
(b) calculate the necessary PID loop feedback value,
(c) filter the result,
(d) apply the necessary (non-filtered) first-stage feedback

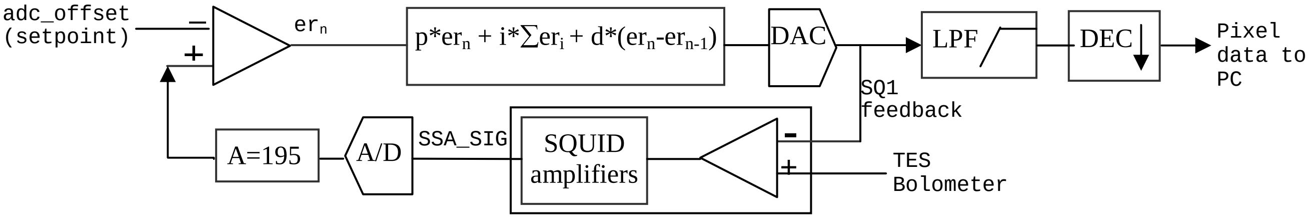

Figure 1: SQUID Readout PID Servo Loop

For SQUID output digitization, 14-bit video ADCs are used. The readout value effectively determines the value that will be applied to the DAC driving the first-stage feedback coil during the next frame. Each PID loop has a pixel-dwell time of about 1.2µs. There will be several 50MHz pixel read-outs during a dwell-time, but only one bias correction per dwell-time. The RCs are synchronized with the rest of the system in such a manner that SQUIDs are only sampled when their outputs have stabilized. The read-outs taken during part of a pixel dwell-time are co-added and then the results are assembled into ‘unfiltered’ frame fragments, at a rate of about 20kHz.

In order to reduce in-band aliasing, a low-pass filter is required with a cut-off frequency of half the sampling frequency. A 4-pole Butterworth low-pass filter is thus implemented by the RC firmware.

Different data modes allow to output different signals like the amplified output signal (*error*), the processed feedback signal (*fb*), the filtered feedback (*filt_fb*) as well as mixture of different signals, as required by the user, all in a 32-bit format. In science mode, 32 bits of filtered feedback signal will be output allowing to stay away from digitization problems and to have enough signal-to-noise ratio (S/N) for any of the expected signal which can be as high as $S/N \cong 10^9$. The latter statement derives from the consideration that the digitization noise becomes important to a level of 1% for a noise signal whose standard deviation drops below 2.

2. Hardware

This section describes the circuit implementation of the Readout Card by going through the schematics sheet by sheet.[101].

2.1 Sheet 1

This is the top level Protel sheet.

2.2 Sheet 2: Bus Backplane and Instrument Backplane

Some signals have slightly different names on Bus Backplane (BB) schematics than on the RC schematic, and are referred to by their BB names with their RC names in parentheses.

2.2.1 Instrumentation Bus Connector (J1)

The RC has 8 analog inputs (S1_FBv_CH0 to S1_FBv_CH7) and 8 analog outputs (SSA_SIG_CH0 to SSA_SIG_CH7); all are single-ended and are brought to the IB through J1. Each signal has a separate return line, routed on the circuit board so that it forms a 50-ohm transmission line with its corresponding signal line, back to AGND at the circuitry the signal comes from.

2.2.2 Power-Supply Voltages (on J2)

The RC uses several supply voltages from the BB for its digital and analog circuitry. The RC drops the BB supply voltages through linear regulators to ensure correct voltage and noise requirements are met.

The +Vlvd (4VD5+) and +Vcore (2VD5+) supplies are used to power the digital I/O circuits and FPGA core, respectively. They feed on-board regulators on Sheet 5. The +Vah (15VA+), +Va (7VA+) and -Va (7VA-) supplies are used for the analog circuits on Sheets 6 and 7.

U1A and associated components form a voltage comparator with a threshold of approximately 11.34 V and a small amount of hysteresis. This circuit is intended to pull n15VOK low when the +Vah supply is deemed sufficient. However, the +Vah output for the latest Power-Supply Unit design for MCE, is only about +10 V, so this circuit requires component-value changes to work properly.

U1B and associated components form a voltage comparator with a threshold of approximately -6.46V and a small amount of hysteresis. This circuit is intended to set MINUS7VOK high when the -Va supply is deemed sufficient. However, the -Va output for the latest Power-Supply Unit design for MCE, is only about -6.2 to -6.5V, so this circuit requires component-value changes to work properly.

U2A and associated components form a voltage comparator with a threshold of approximately 5.93V and a small amount of hysteresis. This circuit is intended to pull n7VOK low when the +Va supply is deemed sufficient.

The on-board supply-voltage monitor circuitry is **not** compatible with the latest Power-Supply Unit design and therefore, it is not used by the FPGA firmware to monitor voltages. Instead, the CC firmware monitors the backplane rail voltages by issuing commands to the power supply unit in a timely manner.

2.2.3 LVDS Receivers and Transmitters (on J2)

U3 and U5 are the LVDS receivers for the bussed BB signals from the CC. These devices are located very close to J2 on the BC circuit board in order minimise stub length on the LVDS busses. U4 is the LVDS transmitter for the dedicated communication channels back to the CC.

2.2.4 Other BB Interfaces

U2B and associated components form a threshold detector for the BB BRst signal. R109, R111 and C219 create a filtered reference of approximately 1.03 V. This reference, together with R108 and R110 cause U2B's output to pull low when BRst is above approximately 2.58 V. The damping provided by C218 and the small amount of hysteresis provided by R112 prevent oscillations due to noise. D10A prevents the reference voltage created by R109 and R111 from affecting the divider formed by R67 and R36 on Sheet 5.

Q1A and Q1B generate an open-drain signal on the BB SpTTL4 (ConfRdy) line; the CC uses this signal to tell if any card in the subrack has an unconfigured FPGA. Q2A illuminates D2 only if the BC's FPGA is unconfigured.

U8 buffers the SID*x* (SID*x*_IN) and nExtnd (EXTEND_IN) signals, with ESD protection from R120, R122 and R124-R126; these resistors are large, 1206, packages to discourage arc-over of the ESD charge.

U9 and U7 create the interface to the bi-directional SpTTL*x* and BClr (BCLR) BB signals. Note that BClr (BCLR) was originally a bi-directional SpTTL*x* signal, and was reassigned to be an output from the CC and an input on all the other cards (except for the PSU) in the subrack.

The BRst (BRST) and JTAG (TCK, TDI, TDO, TMS) signals are protected against ESD and other damage by R115, R116, R118, R123 and U6, before continuing to Sheets 3 and 5.

2.3 Sheet 3: Configuration and Extra Features

2.3.1 Debugging Headers

P9 is a general-purpose debugging header. P1 is an impedance-controlled, Mictor type header which can be used for high-speed logic analysis. P8 is a dedicated interface for an external RS-232 serial line interface for debugging purposes.

2.3.2 SRAM

U11 is an optional SRAM which, when installed, may be used by the FPGA firmware to implement more complex filtering and data-buffering algorithms than the FPGA’s internal memory permits. The circuit board layout places the series matching resistors for the address bus near the FPGA to minimise ringing. The data bus, being bi-directional, has split series-matching resistors, with each half of the resistance at either end of the bus lines.

This SRAM is not currently used by the firmware.

2.3.3 Configuration Device

U33, an Altera flash-based Enhanced Configuration Device (EPC16), and associated components are the configuration memory circuit for the FPGA, connected in the standard Altera Fast Passive Parallel mode [2]. Under certain circumstances, U33 will hold its nINIT_CONF signal low; D11A prevents this condition from affecting the rest of the RC’s circuitry (e.g. holding the DACs in reset via nDAC_CLR). The BTDI, CRDTDO, BTMS and BTCK signals form a chain with the FPGA JTAG signals on U10I (Sheet 5) to insert these two devices into the overall subrack JTAG chain.

2.3.4 Temperature Sensors, EEPROM, LEDs

U34 is the combined silicon identifier and temperature sensor device. It is not installed right on the circuit board pattern, but extended with wires to locate it near the hottest part of the circuit board [102].

U35 is an EEPROM with SPI interface that enables the FPGA firmware to save data and parameters that survive resets and power-cycles. D15-D17 are the front-panel LEDs. Note that D15 is wired so that it is illuminated (albeit at a lower intensity) if the FPGA is not driving the output pin to D15; an illuminated red LED indicates that a card is powered, but has a fault.

P4 allows debugging of the SMBus interface to the helper chip, on Sheet 5, that reads the FPGA’s on-die temperature sensor.

2.4 Sheet 4 : FPGA interface to DACs/ADCs

2.4.1 Mixed-Signal Interfaces

This Sheet contains all of the digital connections between the FPGA and the signal-conversion (ADC, DAC) circuitry. P5 is an impedance-controlled debugging header that is connected to all the high-speed interface signals for the first channel; it was used extensively during development of the FPGA firmware. P6 provides debugging access to the low-speed signals for the first channel.

U10F provides the LVDS clocks signals to the ADCs for each channel. The I/O pins on this bank of the FPGA are controlled together, so the other I/O pins on this bank cannot be used as standard single-ended digital signals.

2.4.2 FPGA Clear Button

Push-button SW1 and associated circuitry provide an ESD-protected manual register-reset input for the RC’s FPGA. Note that the FPGA firmware must be programmed to use the FPGA’s DEV_CLRn input as a register-reset input in order for SW1 to have any effect.

2.5 Sheet 5: FPGA Power, Clock Distribution, Die Sensor, Reset circuitry

2.5.1 Clock Distribution

The RC requires the reference clock from the BB to be supplied to many FPGA clock inputs, in order to implement the functionality required of it. U19 and U43 are special low-skew clock driver devices used to fan-out the BB LVDS clock source to all the FPGA clock inputs. Two clock driver devices are cascaded in order to maintain a single-load on the CLK signal from the LVDS receiver (U5B on Sheet 2). Impedance-controlled debugging header P7 provides access to the FPGA's PLL outputs and other clock inputs, during development.

2.5.2 FPGA I/O Power, Configuration Signals, Die Temperature Sensor

An elaborate power-decoupling scheme is shown on U10I. The capacitors are distributed in a carefully-planned pattern on the circuit board to reduce power-rail bounce and I/O ringing. While the scheme shown does not completely eliminate simultaneous-switching noise (SSN) when a large number of FPGA I/O pins are switching at the same time, it is sufficient for the firmware the RC runs in the MCE.

Also on U10I is the FPGA's dedicated pins for selecting the configuration mode, and the FPGA's hardware JTAG signals. SW3 controls the FPGA's configuration mode selection signals, and also provides two general-purpose input switches (signals DIP1 and DIP2).

The FPGA contains an on-die, diode-connected, transistor for use as a temperature sensor. U36 is a helper chip specifically designed to convert the silicon-junction forward-bias voltage to an SMBus data stream. The FPGA firmware reads the die temperature to ensure overheating damage does not occur in case of subrack ventilation failure.

2.5.3 FPGA Core and PLL Power

U10J's VCCINT pins are connected to an elaborate power-distribution "mesh" similar to the FPGA's VCCIO*x* pins; similar care was taken in the circuit board layout to ensure proper FPGA core operation.

Note the additional inductive filtering for the FPGA PLL +1.5 V power pins. This technique is recommended by Altera to provide additional isolation between the PLLs, and between the PLLs and the FPGA core. Although no PLL outputs are used except for connecting to the debugging header P7, the PLL output power pins must still be connected to +3.3 V; the filters formed by L500-L503 and associated capacitors are overkill, and sharing the VCCIO*x* distribution network would likely have been sufficient.

Pins U12 and U18 are ground pins in the EP1S30F780, but have been left unconnected to allow use of the EP1S40F780. The final version of the RC was populated with an EP1S40F780, due to the resource demands of the SCUBA-2 application.

2.5.4 FPGA I/O and Core Voltage Regulators, Chassis Grounding, Reset Circuitry

U37 and U42 are low-dropout, high-current, regulators providing the +3.3 V and +1.5 V power rails, respectively. They are protected against input transients and noise by D18, L16, D19, D20, L17, D21 and have multiple, large, input and output capacitors to ensure stability. Aluminium-organic and/or tantalum-organic capacitors are used for low-ESR without the short-circuit failure mechanism of regular tantalum capacitors. Additionally, the +3.3 V rail is prevented from being lifted above approximately 3.6 V by D24, as could happen if any of the BB SpTTL*x* (and other BB) signals are driven by 5-volt logic. Note that these regulators have sufficient output-current capability for the normal firmware run by the RC; however, tests have shown that the FPGA, especially if populated with an EP1S40F780, can easily demand more current than U37 and/or U42 can supply, if the firmware is not designed with consideration for power consumption.

R72-R74, distributed around the RC's card-edge contact rails, permit controlled ESD charge equalisation between the subrack chassis and the BC circuitry during card insertion and removal. D22 and D23 limit the chassis-to-circuit voltage differential to a safe level; for low voltage differentials, the resistors provide a soft equalisation while the transient suppressor diodes ensure a fast discharge of larger voltages.

U38, U40 and associated components form a voltage-detecting reset circuit to reset/reconfigure the FPGA once both I/O and core voltages are stable. R66 directly connects the 1VD5+ rail to U38's

RST IN pin to determine the reset threshold at 1.22 V. C169 programs U38 to release its /RST output approximately 30 ms after the 1VD5+ rail has stabilised above 1.22 V. U38's /RST output is cascaded to U40, which is programmed by R67, R70 and C170 to release the RC's nRESET signal approximately 3 seconds after the 3VD3+ rail has stabilised above 2.81 V. (Note: The RT and WT times shown on the schematic are outdated.)

U38 and U40 also provide a watchdog function for the FPGA. While U38's watchdog is disabled by grounding the SWT pin, U40's watchdog timeout is programmed to 1 second by C190. U38 and U40's reset timeout (SRT) capacitors were chosen to satisfy the FPGA's requirements to ensure reconfiguration regardless of I/O and core voltage sequencing; the timeout values were obtained after a large amount of research, as the FPGAs do not behave exactly as Altera claims. JP2 is normally set to allow CONF_DONE to disable the watchdog, by allowing U40's WDI input to float and self-reset, when the FPGA has not been configured. Once the FPGA has been configured, CONF_DONE goes high and the FPGA must toggle the WDI signal to avoid being reset by the watchdog. JP2 in the grounded position forces the watchdog off during debugging, while R71 ensures the watchdog is enabled if JP2 is not installed.

Push-button SW2 (not accessible from front panel) shunts the voltage from R66 to ground, providing a manual, debounced, reset trigger. R69, D2B and D2A provide protection against ESD from the operator's finger. R69 is a 0805 package, but should have been a large, 1206, package to discourage arc-over of the ESD charge. The nBRST signal from Sheet x pulls down on U40's input when BB BRst signal is asserted.

2.6 Sheet 6: FPGA Interface to DACs/ADCs, Regulators

2.6.1 Analog Circuitry Voltage Regulators and Reference

U18 and U24 provide the supply voltage and reference to the serial-input bias and offset DACs for each readout input stage. 15VA+ is used for both the supply and reference, as the serial-input DACs must have supply voltage when the reference is applied.

U25 and U27 provide the analog and digital +3.3 V supplies to the high-speed feedback DACs for each readout channel. Previously, when the 7VA+ rail was 7.0 to 7.5 V, U25 would reach a fairly high temperature; however, the new PSU voltage of 6.2 to 6.5 V lowers U25's dissipation considerably.

U21 and U28 provide the ±5 V power rails for the serial-input bias and offset DAC's buffer amplifiers, and for each input stage's preamplifier chain. D26 and D27 ensure that the regulator outputs do not get reverse-biased by more than one Schottky diode-drop. U28, in particular, will latch in a fault mode if its output pin is lifted positive due to return current from the powered circuitry; this condition can present itself if U21's output comes up faster than U28's output.

U26 generates the digital supply rail for the video ADCs in each readout channel; one regulator can comfortably supply the digital requirements for all 8 channels. The analog supply for the AD6644 video ADC (U31 on Sheet 7), however, draws so much current that only two channels may be supported by a single LT1129 regulator. Thus, four regulators (U17, U20, U22 and U23) are required to supply 4 separate 5VA_ADC+ rails.

Note the use of ferrite isolation components (L*x*) at the input of each regulator, to ensure noise isolation between on-board regulators and between the RC and other cards in the subrack.

2.6.2 Readout Channel Sub-Circuits

The 8 blocks labelled "RC_CHANNEL.SCHDOC" each represent one instance of the circuitry on Sheet 7. Channels 1 through 8 (as seen in the digital signal names) are labelled as blocks A through H on this schematic sheet, and as "Analog 1" through "Analog 8" on the circuit board.

2.7 Sheet 7: Readout Circuitry for One Column

This schematic sheet is the implementation of one readout circuit, which is replicated 8 times by the blocks on Sheet 6. The readout signal (about 10 mV peak to peak) from the SQUID array arrives at IN, and is clamped to approximately ±0.6 V by low-leakage diodes D13 and D14. A serial-input, 16-bit, MAX5443 DAC (U29), low-noise buffer (U15B) and low-noise bias resistor R9 generates the SQUID bias current. U29's 6.2 kilohm output impedance combined with R6/C49 form an 11.5 kHz lowpass filter to limit the total noise energy injected into the SQUID output. Resistors R2, R5 and R7 provide noise isolation between the digital and analog circuitry.

Low-noise, high-speed amplifier U12 provides a voltage gain of 4, with C74 providing roll-off above 64 MHz. C74's effect is primarily to prevent any high-frequency oscillations from occurring. The gain-setting resistors are low-noise types, and very low values are used in order to control Boltzmann noise. The AD797 is specially designed to drive very low feedback impedances, in low-noise applications. Empty circuit-board locations C80 and C82 are provided in case external compensation is required for a specific application. Another serial-input DAC (U30) and buffer circuit (U15) is used to counter the offset generated by the SQUID bias current passing through the array cable resistance. R3 and R1 provide for some additional gain, should it be required, in the offset circuit, while C46 provides high-frequency roll-off in case R1 is populated. The offset DAC works together with the bias DAC in a closed-loop fashion to ensure that U12 and the next stage (U13) do not saturate.

U12 is followed by a 10 MHz filter network (R20 and C78) and another, identical, gain stage (U13 and associated components). After the 16-times gain, the signal is large enough that a high slew-rate amplifier must be used for additional gain. U14, an AD848, has a higher noise figure than the AD797, but the speed to handle the signal which will be almost 1.0 V peak-to-peak at its output. U16, an AD8138 differential amplifier specifically designed to drive the AD6644 video ADC, provides another 2 times gain for a 2.0 V peak-to-peak nominal signal at the ADC (U31). As U31's full-scale differential input range is 2.2 V peak-to-peak, its resolution is well-utilised. The 2.4 volt reference from U31 is provided to U16 as Analog Devices intended for these two parts to work together. C73, C77 and C225 provide high-frequency stability, while R18 and R23 provide impedance matching for U16 to U31. In-depth noise analysis of the RC's readout front-end may be found in [103].

U31's conversion clock is obtained differentially via termination and capacitive-coupling network R25, C119 and C120. LVDS drivers in the FPGA provide the clock via a 100-ohm transmission line, to keep the digital noise injection to a minimum. Resistor packs RN1-RN4 provide some output matching and isolation between the ADC and the digital half of the circuit board. Being a very high speed circuit, U31 requires very careful circuit layout and power decoupling, as might be suggested by the multiplicity of bypass capacitors, power pins and ground pins.

The high-speed DAC, U32, completes the hardware for implementing the flux-locked loop methodology of reading TES bolometer detectors. R36 sets the full-scale output of the 14-bit AD9744 to 19.2 mA. Output termination resistor R33 yields a full-scale voltage of 0.958 V, to be converted to a feedback current by bias resistors on the IB. R33, R28 and C103 yield a roll-off above 48 MHz, with additional capacitances in the IB, filter boxes, cryogenic cable and array further lowering the frequency response. Series matching resistors RN5-RN7 and RN30 control ringing and provide some noise isolation between the digital and analog halves of the RC circuit board.

3. Firmware

The readout card firmware running on the EP1S40 Stratix FPGA controls the following on-board devices: 8 readout circuitries and some auxiliary parts. Each readout circuit includes:

- 14-bit 50MS/s video ADCs
- 14-bit parallel DACs
- 16-bit serial (SPI) DACs
- 16-bit serial (SPI) DACs

The auxiliary parts include the temperature sensors, slot-id and silicon-id, and LVDS transceivers:

- A temperature/silicon id chip (DS8020)
- Backplane transceivers (1 receiver and 2 transmitter)
- 3 front-panel LEDs
- FPGA temperature-read helper chip (MAX1618)

The FPGA runs at 50MHz with the exception of backplane transmit/receive modules running at 100MHz and SPI DACs running at 12.5MHz. All the clocks are generated using Stratix enhanced PLL. For more information about timing and synchronization refer to [xx].

3.1 Readout Control Loop

The main task of the firmware is to provide the right feedback flux to the SQUID readout circuitry to keep the non-linear SQUIDs in their linear regime. The feedback is tuned using a PID servo loop, shown in Figure 1, that samples the SQUID response with an ADC and interprets the difference between the SQUID response and a target locking point as the error entering the PID loop.

The firmware runs independent PID loops in a time-multiplexing fashion for all the rows in a given column. Each time slice amounts to a row visit that is typically 100×20ns (*row_len* × 50MHz clock cycles). During each row visit, the calculated feedback is applied as soon as possible (*fb_dly*) and the ADCs are sampled after the transients are settled towards the end of the row visit. **Therefore, the feedback applied during each visit is calculated based on ADC readings of the previous visit.** This is described in more details later on.

The firmware, similar to the hardware, is modular and runs 8 instances of a *flux_loop_ctrl* block, one per channel. The *flux_loop_ctrl* block controls the data path between the sampling ADCs and the feedback DAC of all the SQUIDs in the corresponding channel in a time-multiplexing fashion. For each time slice (pixel visit), it is in charge of sampling the input signal *N* (*sample_num*) times and calculating a new feedback (*fb*) to be applied to the SQ1 feedback DAC during the next iteration (frame). Figure 2 shows the block diagram for one readout channel and how it communicates with the rest of the overall system.

Each readout channel can independently be in one of the modes determined by *servo_mode* parameter (see Table 1: *servo_mode* settings). When in constant mode, a constant value determined by *fb_const* parameter of the RC is applied to the parallel feedback DAC. When in ramp mode, a ramp signal with parameterized number of steps, amplitude and frequency (*ramp_amp, ramp_step*, *ramp_dly* parameters of the RC), is applied to the feedback DAC. *Ramp_dly* determines the number of address-return-to-zero (ARZ)s between two ramp steps. (See Frame_timing document for definition of a frame and other timing information.)

The constant mode and ramp mode are mainly used during tuning of the 3-stage SQUID amplifier chain. The PID mode is used during normal operation of the MCE. As described earlier, the change in magnetic flux in the SQUID ammeters is countered by a first-stage feedback bias calculated by the firmware and applied by the 14-bit parallel DAC of the RC. The feedback loop is provided by a digital PID loop that calculates the correct flux to keep the whole amplification chain in a linear regime or in "lock". The time around the servo loop is very short compared to the time constant of the detectors. The feedback loop runs at about 10kHz while the detector time constants are about few milliseconds.

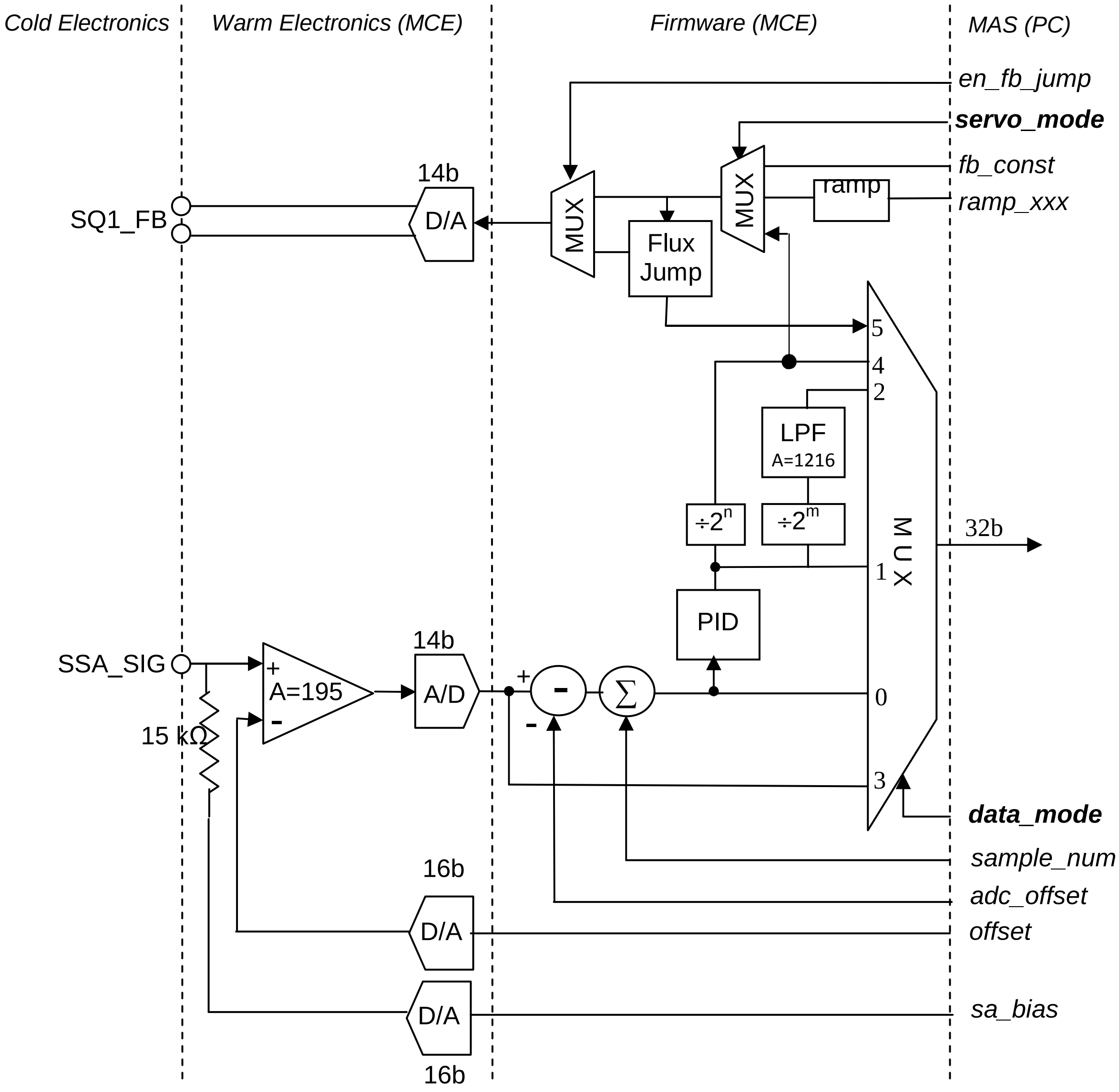

Figure 2: Readout-Card Firmware Block Diagram (n=12, m=12)

value	**mode**	**Description**
0	constant	*fb_const* value is applied to the feedback DAC. This mode is aware of flux-jump (*en_fb_jump*) settings, i.e., if *fb_const* is set out-of-range and flux jump is enabled; it will try to bring it back within DAC range.
1	constant	Same as 0
2	ramp	A ramp is applied to feedback DAC. Ramp settings are determined by *ramp_dly*, *ramp_amp*, *ramp_step*. A new value is applied at each new row visit.
3	PID loop	A new feedback value is calculated as in Eq. 1.

Table 1: *servo_mode* settings

The PID loop calculation of Figure 1 is shown in Eq. 1.

$$fb_1 = P \times coadd_0 + I \times \sum coadd_i + D \times (coadd_0 - coadd_{-1}) \qquad \text{(Eq 1)}$$

where $coadd_i = \sum er_j = \sum (adc\_sample_j - adc\_offset)$ $\quad j = 0, ..., sample\_num - 1$

The value applied to the feedback DAC:

$fb_{DAC} = fb_1/k$ where $k=2^{12}$

For detailed analysis of the PID loop and coefficient settings see [106].

Low-Pass Filtering

Only in PID mode, fb_1 is low-pass filtered through a digital 4-pole Butterworth filter implemented as 2 cascaded biquads. Each biquad has the form of:

$$H(z) = \frac{1 + 2z^{-1} + z^{-2}}{1 + b_1 z^{-1} + b_2 z^{-2}} \qquad \text{(Eq 2)}$$

The key features of the filter characteristics are:

- DC amplification=1217.9148
- f_{3dB}=122.226 Hz (the frequency at which h(n) = $\sqrt{h_{max}(n)}$)
- Gain @ 200Hz=0.14189148 (with respect to the DC gain)

Timing of the Feedback Loop

Each *flux_loop_ctrl* runs as a time-division multiplexer where each time slice corresponds to the time the Address Card of the MCE biases a given row (SQUID1) in that column (channel). This can be interpreted as running independent servos at each time slice and storing the results in different memory locations. The time interval is determined by a common parameter to all cards called *row_len* specifying number of 50MHz clock cycles per row visit. The number of rows is specified by *num_rows* parameter that is also common to all cards. Hence, a frame period (or address-return-to-zero (ARZ) period) is defined as the time it takes to visit all rows:

$t_{ARZ} = num\_rows \times row\_len \times 1/f_{max}$

Considering that MCE can support up to 41 rows and assuming 128 clock cycles per row to be greater than detector settling time:

t_{row} = 128 × 20ns = 2.56µs or f_{row} = 390.625kHz

t_{ARZ}= 41 × 128 × 20ns = 104.96µs or f_{ARZ}= 9.527kHz

This implies that the feedback servo loop for each row (pixel) is running at 9.527kHz.

Note that in each time slice or row visit, after SQ1 is turned on, the SQ1 feedback is applied as soon as possible (*fb_dly*) while the system waits for the transients to settle (*sample_dly*), before reading the ADC samples and co-adding them. The feedback applied during each row visit is calculated based on ADC readings of the previous visit which happened during the previous frame. The timing diagram shown in Figure 3 indicates the timing of operation during each row visit (*row_len* clock cycles). Note that the row number used for calculating the feedback is one behind compare to the row number used for applying the feedback and sampling the ADCs.

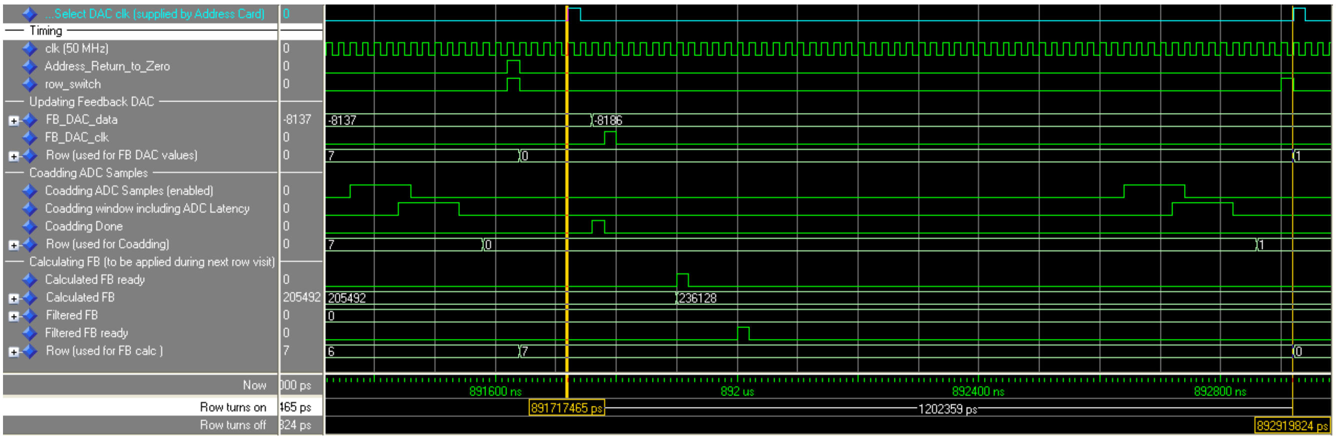

Figure 3: Feedback Calculation, coadding, and feedback assertion timing with

num_rows=8, *row_len*=64, *sample_num*=5, *sample_dly*=50.

RAM Storage

In order to track the results of each row visit (time-slice) independently, various storage RAMs are used. The index of all these memory blocks are reset to 0 upon ARZ assertion and the memory address advances upon every new row visit:

- 2 x 64 x 32b RAM blocks: coadded error storage
- 2 x 64 x 40b RAM blocks: first-stage feedback (fsfb) queue storage (the MSb is used only in ramp mode to indicate the slope of the ramp)
- 2 x 64 x 8b RAM blocks for flux-jump-count storage.
- 1 x 64 x 32b RAM block for storing filter results.
- 4 x 64 x 29b RAM block for storing intermediate filter calculation terms or so-called w_n terms. There are 4 w_n terms needed for the 4-pole filter.

Note that some of the memory banks are double buffered (coadd, fsfb, flux_jump_count) to accommodate simultaneous accesses of previous and current-frame results without arbitration.

Flux Jumping Algorithm

The flux-jumping is performed as follows:

Assume:

n = number of Flux jumps , $n_{max}= 2^7$
fb = calculated feedback value by the PID servo loop
fb_adj = adjusted (flux-jumped) feedback value
Q_{ch} = Flux Quanta for channel ch
$FB_{max} = 2^{13}- 1$ (DAC range)
$J_{max} = 0.95 \times FB_{max}$

Then:

```
$j_i = fb - Q \times n_{i-1}$                (Eq. 1)

if |$j_i$| > $J_{max}$ {
   if |ni| = $2^7$ {
      $fb\_adj_i = FB_{max} \times |j_i| / j_i$
   else
      $n_i = n_{i-1} + |j_i| / j_i$
      $fb\_adj_i = fb_i - Q \times n_i$      (Eq. 2)
   }
}
```

There are two multipliers (mult, mult2) and two subtractors (sub1, sub2) instantiated to handle the arithmetic. Mult1 and sub1 are used to evaluate Eq. 1 while mult2 and sub2 are used to evaluate Eq. 2.

In the following table, the multiplier and subtractor operands are noted by *a* and *b*. The arithmetic for all 8 channels are pipelined. The pipeline operations are scheduled as follows:

State	Calculate			Load mult1 a	Load mult1 b	Load sub1 b	Load mult2 a	Load mult2 b	Load sub2 a
Idle	-	-	-	Q_7	$n_{(i-1)7}$	fb_{i7}	Q_7	$n_{(i-1)7}$	fb_{i7}
Calca0	j_{i0}	-	-	Q_0	$n_{(i-1)0}$	fb_{i0}	Q_7	$n_{(i-1)7}$	fb_{i7}
Calca1	j_{i1}	n_{i0}	-	Q_1	$n_{(i-1)1}$	fb_{i1}	Q_7	$n_{(i-1)7}$	fb_{i7}
Calca2	j_{i2}	n_{i1}	fb_adj_{i0}	Q_2	$n_{(i-1)2}$	fb_{i2}	Q_0	$n_{(i-1)0}$	fb_{i0}
Calca3	j_{i3}	n_{i2}	fb_adj_{i1}	Q_3	$n_{(i-1)3}$	fb_{i3}	Q_1	$n_{(i-1)1}$	fb_{i1}
Calca4	j_{i4}	n_{i3}	fb_adj_{i2}	Q_4	$n_{(i-1)4}$	fb_{i4}	Q_2	$n_{(i-1)2}$	fb_{i2}
Calca5	j_{i5}	n_{i4}	fb_adj_{i3}	Q_5	$n_{(i-1)5}$	fb_{i5}	Q_3	$n_{(i-1)3}$	fb_{i3}
Calca6	j_{i6}	n_{i5}	fb_adj_{i4}	Q_6	$n_{(i-1)6}$	fb_{i6}	Q_4	$n_{(i-1)4}$	fb_{i4}
Calca7	j_{i7}	n_{i6}	fb_adj_{i5}	Q_7	$n_{(i-1)7}$	fb_{i7}	Q_5	$n_{(i-1)5}$	fb_{i5}
Pause1	-	n_{i7}	fb_adj_{i6}	Q_7	$n_{(i-1)7}$	fb_{i7}	Q_6	$n_{(i-1)6}$	fb_{i6}
Pause2	-	-	fb_adj_{i7}	Q_7	$n_{(i-1)7}$	fb_{i7}	Q_7	$n_{(i-1)7}$	fb_{i7}
Pause3	-	-	-	Q_7	$n_{(i-1)7}$	fb_{i7}	Q_7	$n_{(i-1)7}$	fb_{i7}

Note that turning on the flux-jumping option adds additional delays (12 clock cycles or 12×20ns = 240ns) to when the feedback is applied to the DAC in respect to the row being turned on.

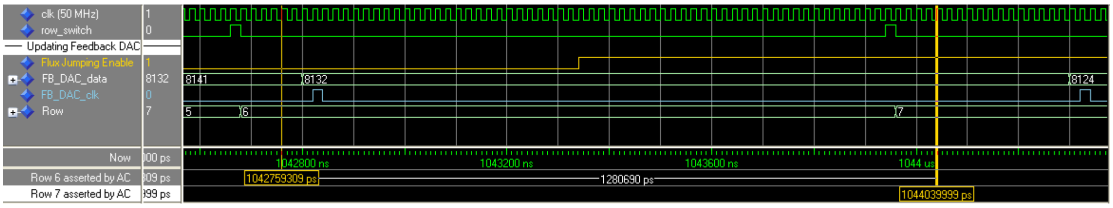

Figure 4: Timing Diagram for Updating the Feedback DAC. Note that when flux-jumping is turned on, it takes 11 clock cycles (220ns) longer to apply the feedback.

3.2 Readout Modes

Separately from the SQ1 feedback loop, a collection of data is transmitted over the backplane and then to the PC. A frame of data is defined as series of 32b numbers that each correspond to a pixel. The 32b of data may contain the pixel-data, the coadded error, the filtered data, the number of flux jumps or a combination of these depending on the *data_mode* settings as listed in Table 2.

fb = calculated SQ1 feedback (i.e. constant/ramp/pid mode)

fb_{DAC} = feedback applied to the DAC ($fb_{DAC}=fb/k$ where $k=2^{12}$ when *servo_mode* = 3 and 1 otherwise)

fb_{filt} = low-pass filtered SQ1 feedback

adc_sample = ADC sample (signed 14b)

coadd = $\sum$ *(adc_sample*$_i$ *– adc_offset)* where j=0, 1, .., *sample_num* -1

n = number of flux jumps

data = pixel-data read-out

mode	Description	pixel-data (*data*)	Firmware Rev. support
0	Coadded error	*data [31:0] = Coadd[31:0]*	All
1	feedback	*data [31:0] = fb[31:0]*	All
2	Filtered feedback	*data [31:0] =* fb_{filt}*[31:0]*	2.0.5 and later
3	Raw samples	*data [31:0] = sign-extend (adc_sample[13:i])* where i=0 in firmware 4.3.7 and i = 6 otherwise.	Special build: 3.0.7, 3.0.16, 3.0.25, 4.1.7, 4.2.7, 4.3.7
4	mixed: fb & error	*data [31] =* fb_{DAC}*[31] , data[30:14] =* fb_{DAC} *[28:12]* *data[13] = coadd [31], data[12:0] = coadd[12:0]*	2.0.9 and later
5	mixed: fb & n	*data [31: 8] = fb [31: 8] , data[7:0] = n*	All
6	(obsolete) mixed: fb_{filt} & error	*data [31]=* fb_{filt}*[31] , data[30: 14] =* fb_{filt}*[27:11]* *data [13] = coadd[31], data[12:0]= coadd[12:0]*	3.0.30 till 4.0.6
7	mixed: fb_{filt} & error	*data [31]=* fb_{filt}*[31] , data[30: 10] =* fb_{filt}*[27:7]* *data [9] = coadd[31], data[8:0]= coadd[12:4]*	4.0.2 and later
8	(obsolete) mixed: fb_{filt} & n	*data[31: 8] =* fb_{filt}*[31:8]* *data[7:0] = n*	4.0.4 only
9	mixed: fb_{filt} & n	*data [31]=* fb_{filt}*[31] , data[30: 8] =* fb_{filt}*[23:1]* *data[7:0] = n*	4.0.5 and later

Table 2 : *data_mode* settings

Raw Mode Readout

Raw data refers to 50MHz ADC samples. The memory available in MCE to store such data is 8192×8 bits where the 8 lower bits of the 14-bit ADC samples are stored. In the latest raw-mode firmware (rev. 4.3.7) we could fit 8192×14 bits where the full-resolution ADC samples are stored.

Raw data acquisition is only available in certain build of readout-card firmware. The latest firmware to support raw mode is: 0x04030007.

Raw data acquisition and 4-pole low-pass filter are both memory-intensive features of the MCE readout-card firmware. As a general rule, a given firmware has only one of these features available, i.e., the firmware that has raw mode enabled will not include the filter and vice versa. In some versions that support raw mode, read back of gaini0/gainp0/gaind0 is also not supported.

There are two MCE parameters that have to be set for raw data acquisition:

- *captr_raw* stores a snapshot of *row_len* samples of *num_rows* rows for **2** consecutive frames up to 8192 samples for each channel and stores them in MCE memory buffer for readout in raw mode.
- *data_mode* has to be set to 3 for raw-mode readout.

A typical sequence of commands for data acquisition is:

```
 wb cc ret_dat_s 1 1
 wb rc1 data_mode 3
 wb rc1 captr_raw 1
 acq_config filename rc1
 acq_go row_len*2
```

1) Once the raw-buffer of 8192 samples is full, no more samples are stored till a new *captr_raw* is issued.

2) The raw-buffer is sent over the backplane in a regular frame-format chunks (num_rows_reported*num_columns) till the buffer is emptied.

For rc1 only, for example, time indices t = [0, num_rows_reported-1] are reported in the first frame, with the offset of column c and time t being:

```
index = t*8 + c
```

Subsequent frames contain subsequent data samples. Since the number of time samples, per column, in a full multiplexing cycle of the MCE is *row_len*num_rows*, the number of readout frames required to record an entire MCE internal frame's-worth of raw data is equal to *row_len* (assuming that *num_rows* = *num_rows_reported*).

The readout frame and index of a given time t, column c, is thus

```
frame = floor[ (t*8 + c) / num_rows_reported ]
index = (t*8 + c) mod num_rows_reported
```

Since the number of clock ticks spent on each row is *row_len*, the row associated with time t is:

```
row = floor [ t / row_len ]
```

It is possible to get raw data from multiple readout cards; in that case the data will be packaged in the usual way, with all columns reporting at each time.

Schematically, the organization of raw data for *num_rows* = *num_rows_reported* = 33 and *row_len* = 100 can be represented as follows (here sNNNNNcXX represents a word containing the data for time sample NNNNN, column XX):

```
(header of frame 0, 43 words)
s0000c00 s0000c01 s0000c02 s0000c03 s0000c04 s0000c05 s0000c06
s0000c07
s0001c00 s0001c01 s0001c02 s0001c03 s0001c04 s0001c05 s0001c06
s0001c07
s0002c00 s0002c01 s0002c02 s0002c03 s0002c04 s0002c05 s0002c06
s0002c07
...
s0032c00 s0032c01 s0032c02 s0032c03 s0032c04 s0032c05 s0032c06
s0032c07
(checksum of frame 0, 1 word)

(header of frame 1, 43 words)
s0033c00 s0033c01 s0033c02 s0033c03 s0033c04 s0033c05 s0033c06
s0033c07
...
s0065c00 s0065c01 s0065c02 s0065c03 s0065c04 s0065c05 s0065c06
s0065c07
```

```
(checksum of frame 1, 1 word)

...

(header of frame 199, 43 words)
s6567c67 s6567c01 s6567c02 s6567c03 s6567c04 s6567c05 s6567c06
s6567c07
...
s6599c00 s6599c01 s6599c02 s6599c03 s6599c04 s6599c05 s6599c06
s6599c07
(checksum of frame 199, 1 word)
```

3) Unfortunately, collecting data for up to **2** internal MCE frames is hard-coded in firmware. Even if you reduce number of rows, you will be restricted by 2 frames. If you really want to look at more samples, then you may want to increase *row_len* instead and using *row_order* parameter of the address card, move your particular row of interest to the beginning of the frame.

4) *readout_row_index* parameter is ignored when data_mode is set to 3 or raw mode.

5) If *num_rows_reported* < *num_rows*, it takes more frames (to be exact: n = *num_rows***row_len**2/*num_rows_reported* frames) to read all the data out.

Continous Co-add Readout

Not implemented yet. Continuous coadd data refers to reading the co-added error data in a frame rate.

3.3 SA_Bias and Offset

The 16b serial SPI DACs are controlled by *sa_bias_ctrl* and *offset_ctrl* blocks instantiated in firmware. When a new value is set through the *sa_bias* and *offset* parameters, then on the next ARZ, these values are loaded to the DACs. The SPI interface runs at 12.5MHz. Therefore, it takes about 17 clock cycles to update all DACs. Up to Rev. 4.0.7, these values were updated on every ARZ, but the latest firmware only updates the DACs once a new command is issued. All DACs are set to 0 at startup, but resetting the MCE will not automatically set them to 0.

3.4 Auxiliary Interface

Refer to all_cards.doc to find out about firmware related to controlling temperature sensors, LEDs, ID chips, etc. These parts are common to all cards.

3.5 Other

The Wishbone bus protocol (refer to Wishbone Specification) is adopted as inter-block communication standard between the different blocks in readout_card firmware. One block, in this case *dispatch*, acts as Wishbone Master and the other blocks are Wishbone slaves. Commands are received along the backplane LVDS lines and then decoded by the Wishbone Master and communicated to the relevant Wishbone Slave through the Wishbone protocol.

The Wishbone interface and timing is better described in “dispatch block description (SC2-ELE-S580-202)” document. Parameters decoded by the top-level (as of rc firmware 4.0.6) are listed in Table .

3.6 Readout–Card Parameters

Parameter	**Description**
servo_mode	8 values, one per column. Default is 0 for all columns. See Table 1.
sa_bias	8 values, 1 per column, to set 16b sa_bias serial DACs.
offset	8 values, 1 per column, to set 16b cable-offset serial DACs.

adc_offset0 adc_offset1 adc_offset2 adc_offset3 adc_offset4 adc_offset5 adc_offset6 adc_offset7	Each of the 8 parameters, one per column, takes 41 values, one per row, to set the *adc_offset* used in PID servo calculation as per Eq. 1.
fb_const	8 values, one per column, for feedback DACs when *servo_mode* is set to constant mode.
gainp0 gainp1 gainp2 gainp3 gainp4 gainp5 gainp6 gainp7	*P* coefficient of the PID servo calculation in Eq. 1. Each of the 8 parameters, one per column, takes 41 values, one per row. Total of 41×8 *P* values are specified through these parameters. *Note1 : gaini are specified as "signed 12-bit" values as of revision 4.0.9. (These were 10-bit values in 4.0.2 to 4.0.8, 8-bit values prior to revision 4.0.2.)* *Note 2: In firmware revisions with raw-mode readout feature, the read-back of gaini is disabled.*
gaini0 gaini1 gaini2 gaini3 gaini4 gaini5 gaini6 gaini7	*I* coefficient of the PID servo calculation in Eq. 1. (Similar to gainp0.)
gaind0 gaind1 gaind2 gaind3 gaind4 gaind5 gaind6 gaind7	*D* coefficient of the PID servo calculation in Eq. 1. (Similar to gainp0.)
en_fb_jump	enable flux jumping
flx_quanta0 flx_quanta1 flx_quanta2 flx_quanta3 flx_quanta4 flx_quanta5 flx_quanta6 flx_quanta7	Each of the 8 parameters, one per column, takes 41 values. Each value represents the size of the phi-0 in number of DAC units.
ramp_step	The step size of the ramp signal in terms of DAC units (14b DAC).
ramp_amp	The amplitude of the ramp signal in terms of DAC units (14b DAC).
ramp_dly	The delay between two successive values of the ramp in terms of number of ARZs. If this is set to be the same as *data_rate* parameter in CC, then each readout corresponds to one ramp value.
filt_coeff	Not implemented
fltr_rst	Resets the filter and internal filter registers (w_n RAM storage blocks) to 0
data_mode	Determines the pixel-data readout mode. See Table 2.
readout_row_index	The row index sets the starting row in a reported data block.
ret_dat	Requests a data frame

captr_raw	Capture a snapshot of *row_len* samples of *num_rows* for 2 frames up to 8192 samples for each channel and stores them in MCE memory buffer for readout in raw data mode.
flx_lp_init	Reinitializes the PID servo loop.
sample_num	Number of ADC samples to coadd
fb_dly	Number of 50MHz clock cycles since a row switch before a new value is applied to the feedback DAC. (*fb_dly* >= 7, when flux jumping is on *fb_dly*>=18)
row_dly	Number of 50MHz clock cycles between the de-assertion of the previous row at the new-row boundary, and the assertion of the current row on the Address Card (multiplexer). This is only used to recreate the AC timing and is irrelevant for readout card itself.
fpga_temp	Retrieves the FPGA silicon temperature through the on-board helper chip. The result is in ºC.
card_temp	Retrieves the card temperature from on-board DS8020. The result is in ºC.
card_id	Retrieve the unique silicon ID from on-board DS8020 (ID chip).
Led	Controls 3 on-board LEDs. The specified value is XORed with current status of the LEDs.
fw_rev	Returns a firmware revision where the format in hex is RRrrBBBB • RR is the major revision number • rr is the minor revision number • BBBB is the build number
slot_id	Returns the 4b ID of the particular backplane slot the card is plugged into.
card_type	Returns the card_type which is 2 for readout cards.
scratch	8 user-configurable read/write registers.
row_len	number of 50MHz clock cycles per row
num_rows	number of rows multiplexed

Table 3: List of readout-card parameters

4. Future improvements

An alternative to the PID-loop way of reading data is represented by the modulation scheme that uses the 1st stage SQUID V-phi curves to modulate the incoming signal. A continuous 1st-stage SQUID feedback ramp would be implemented instead of the PID feedback loop allowing a signal modulation with the SQ1 V-phi curve periodicity. Typically the SQ1_fb could ramp through 100 phi_0 at 10Hz (we could also implement a triangular wave). This would create a 1kHz modulation on the output signal. This frequency should be between the detector time constant and the frame rate. Then we demodulate by measuring the phase change (induced by the TES) on every V-phi curve with respect to a template. With this method we remove pick-up in the summing coil (apparently the dominant source of field sensitivity reduction). Another advantage is that, in this scheme, we don't need 2 locking points if we want to modulate the TES bias. The implementation of this new read-out scheme requires the necessity of a firmware that allows the demodulation of these signals from the 1st stage SQUID V-phi curves at the row rate to extract the incoming signal.

5. References

[1] M. Halpern, "Functional Description of Multi-Channel Electronics", SC2-ELE-S585-502

[2] "Configuring Altera FPGAs", Altera Configuration Handbook, Chapter 1, Aug. 2005

[101] “Readout Card Rev. B Issue 9 Schematic”, SC2-ELE-S582-101

[102] *** Need a reference to where U34 is now located on the circuit board. ***

[103] M. Halpern, “Analysis of Readout Amplifier voltage and current noise” (amplifier_note.pdf)

[104] “Low-pass Filtering”, SC2-ELE-S582-211, v1.2

[105] B.D. Kelly, “SCUBA-2 Servo Loop Simulation”, SC2-SOF-S200-049, v1

[106] B. Burger, “Frame Timing”, SC2-ELE-Sxxx-xxx,

6. List of Tables and Figures

7. List of Acronyms

8. Table of Contents