SAMPA V3
Specification

Revision 0.2

Arild Velure[1] Bruno Sanches[2]

September 20, 2018

[1] e-mail: Arild.Velure@cern.ch
[2] e-mail: Bruno.Cavalcante.De.Souza.Sanches@cern.ch

Revision

0.1 Initial draft

0.2 Functional description, Interfaces and protocols and Circuit description sections modified. Reviewer: Raul Acosta Hernandez

Contents

1 Functional description 6

1.1 Analog front-end 6

1.2 Analog to digital converter 6

1.3 Pre-trigger samples delay 7

1.4 Digital signal conditioning 7

1.4.1 First baseline correction 8

1.4.2 Digital shaper 10

1.4.3 Baseline correction and subtraction II 14

1.4.4 Baseline correction III 15

1.5 Compression 17

1.5.1 Huffman compression 17

1.5.2 Zero suppression 17

1.5.3 Lossy data compression 19

1.6 Channel ordering 20

1.7 Data formatting unit and Ring buffer 20

1.7.1 Data formatting unit (DFU) 20

1.7.2 Ring buffer 21

1.8 Serializing 23

1.8.1 Standard Modes 23

1.8.2 Test Modes 23

1.9 Event managing 23

1.10 Daisy chaining 24

1.10.1 Neighbor module 24

1.11 Direct ADC serialization mode 25

2 Interfaces and protocols 27

2.1 Analog front-end 27

2.2 Slow Control 27

2.3 Serial interface 28

2.3.1 Protocol 29

2.3.2 Heartbeat packet 29

2.3.3 Sync packet 30

2.3.4 Types of packets 31

2.4 Daisy chaining 32

2.5 Direct ADC serialization 34

2.6 Test interfaces 35

2.6.1 JTAG 35

2.6.2 Scanchain 37

2.6.3 Memory built in tester 38

2.7 Hardware address 39

2.8 Sync and trigger 39

2.9 Clock and reset 40

2.9.1 Clock 40

2.9.2 Reset 41

3 Registers 45

3.1 Global registers 45

3.1.1 Pin status 45

3.1.2 Event management 45

3.1.3 Channel register access 47

3.1.4 Channel ordering 48

3.1.5 ADC configuration 49

3.1.6 Serial link configuration 49

3.1.7 Data compression 50

3.1.8 Power saving features 50

3.1.9 Test functionality . . . 51
3.2 Channel specific registers . . . 57
3.2.1 Data path configuration . . . 58
3.2.2 Digital shaper . . . 58
3.2.3 BC1 . . . 59
3.2.4 BC2 . . . 60
3.2.5 BC3 . . . 62
3.2.6 Zero suppression . . . 62

4 Circuit description 64
4.1 Analogue front-end . . . 64
4.2 ADC . . . 64
4.3 Configurable Reference Voltage Source . . . 64
4.4 Triple Modular Redundancy . . . 65
4.5 Filters Data-path . . . 65
4.6 Pre-trigger samples delay . . . 67
4.7 Baseline Correction I . . . 67
4.8 Digital shaper . . . 70
4.9 Baseline Correction II . . . 72
4.10 Baseline Correction III - Slope based . . . 74
4.11 Huffman . . . 74
4.12 Zero suppression . . . 74
4.13 Data formatting . . . 78
4.14 Serializing . . . 78
4.15 Direct ADC serialization mode . . . 79

5 Datasheet 81
5.1 Electrical specifications . . . 81
5.1.1 Power supplies . . . 81
5.1.2 ADC voltage reference . . . 81
5.1.3 Decoupling . . . 82
5.1.4 Absolute maximum ratings . . . 82

5.1.5 Digital IO pad characteristics . . . 82
5.1.6 Power consumption . . . 82
5.1.7 SLVS termination . . . 82
5.1.8 Internal pull-up/pull-down . . . 83
5.1.9 Reset generation . . . 83
5.1.10 Configuration pins pull-up/pull-down . . . 83
5.1.11 Floating/unused inputs . . . 84
5.1.12 I2C . . . 84
5.2 Package description . . . 85
5.2.1 Mechanical characteristics . . . 85
5.2.2 Pinout . . . 85
5.3 Related documentation . . . 92
5.4 Known issues . . . 93
5.4.1 V2 . . . 93
5.4.2 V3/v4 . . . 95

Chapter 1

Functional description

This section describes the functional operation of the main blocks of the SAMPA ASIC.

1.1 Analog front-end

The front-end block is composed of a positive/negative polarity Charge Sensitive Amplifier (CSA) with a capacitive feedback C_f and a resistive feedback R_f connected in parallel, a Pole-Zero Cancellation (PZC) network, a high pass filter, two bridged-T second order low pass filters, a non-inverting stage, as shown in figure 1.1.

The first shaper is a scaled-down version of the CSA and generates the two first poles and one zero. A copy of the first shaper connected in unity gain configuration is implemented in order to provide a differential mode input to the next stage.

The second stage of the shaper is a fully differential second order bridged-T filter and it includes a Common-Mode Feed-Back network (CMFB). The non-inverting stage adapts the DC voltage level of the shaper output to use the full dynamic range of the ADC. It consists of a parallel connection of two equally designed Miller compensated amplifier.

The gain, polarity and shaping time of the front-end is programmable through external pins.

1.2 Analog to digital converter

The ADC is based on a split capacitor fully differential successive approximation (SAR) topology. The SAR topology allows for low power with reasonable sample rates and resolution. The ADC has a resolution of 10-bit and a sample rate up to 20 Msamples/s. The block diagram of the ADC is shown in figure 1.2. The main parts of the circuit are: capacitive array, switches, comparator and the SAR control logic. The capacitor array is used to perform sample & hold and the digital to analog converter functions.

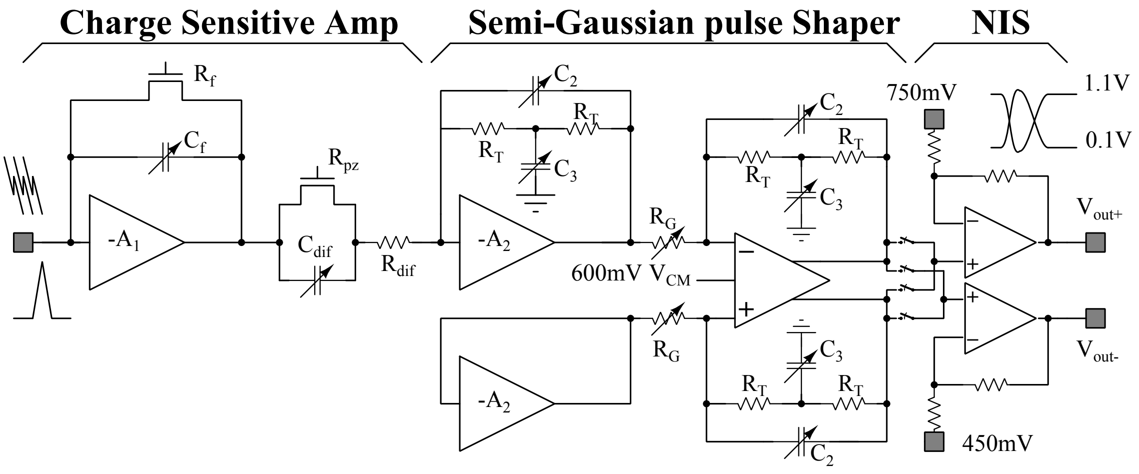


Figure 1.1: *Block diagram of the analog front-end.*

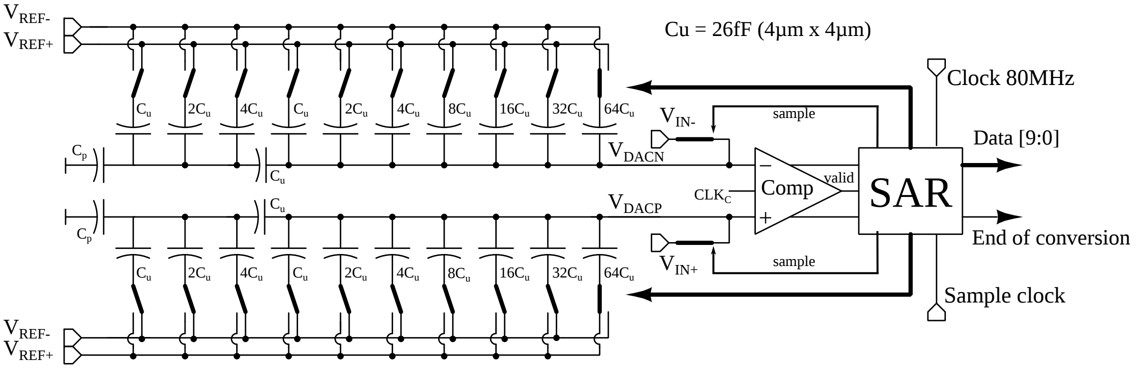


Figure 1.2: *Block diagram of the ADC.*

1.3 Pre-trigger samples delay

When running with an externally provided trigger signal, it is often needed to compensate for the delay between the triggering event and the reception of the trigger signal on the device. This compensation is done by delaying the incoming data by the same amount as the trigger signal. If one also wants to look at data prior to the triggering event, it is also possible to add more delay. It is possible to delay the data by up to 192 ADC cycles, corresponding to 19.2 µs at 10 MHz.

The configuration does not alter the length of the processing time window, it merely moves the trigger point in the sample stream earlier in time then when the trigger was received.

1.4 Digital signal conditioning

The objective of the Digital Signal Condition (DSC) is to increase the efficiency of the data compression algorithms (e. g. zero suppression). The DSC implements, in several subsequent stages (pipeline), different algorithms to condition and shape the signal. The DSC is comprised of four main building blocks

Baseline Correction I (BCI) It is the first stage of the data processor. It's main task is to remove the low frequency perturbations and systematic effects. This filter has different modes of operation depending on the application.

Digital Shaper (DS) It can be use for two different applications; tail cancellation or peaking time correction. A fourth order IIR filter is implemented. The choice of the filter parameters configure the system for one of the applications:

- Tail cancellation: an accurate cancellation of the signal tail is required in order to perform the zero suppression efficiently. Since the filter coefficients for each channel are fully programmable and re-configurable, the circuit is able to cancel a wide range of signal tail shapes.
- Peaking time correction: some applications need the modification of the peaking time to

Baseline Correction II (BCII) This unit applies a baseline correction based on a moving average filter. This scheme removes non-systematic perturbations of the baseline that are superimposed to the signal. At the output of this block, the signal baseline is constant with an accuracy of 1 Least Significant Bit (LSB).

Baseline Correction III (BCIII) This unit applies a baseline correction through a slope based filter. It is provided as an alternative to the BCII filter as the BCII filter can potentially get stuck outside its thresholds in cases of very large perturbations.

1.4.1 First baseline correction

The perturbations affecting the signal from a gas chamber can be:

- Low-frequency spurious signals (in the range of less than one kilohertz). These perturb the detector signal by shifting its baseline by an amount that is almost constant (less than one ADC count) inside the processing time window. This type of signal perturbation could for instance be caused by
  - Environmental changes e.g. drifts in temperature and voltage.
  - Environmental variations due to differences in placement in the detector.
  - Manufacturing/process variations.

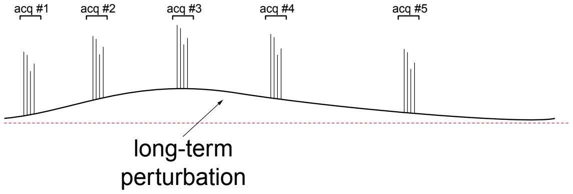


Figure 1.3: *Example of long term perturbations.*

- Signal perturbations created by systematic effects, like those related to triggering of the detector, which affect the signal in terms of a superimposed noise pattern.

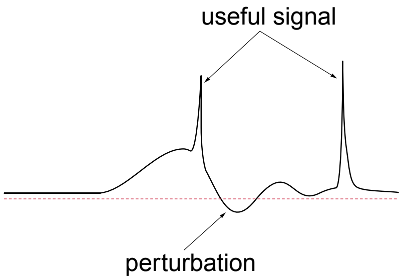


Figure 1.4: *Example of gating perturbations.*

To cope with the first effect, an IIR filter is implemented right at the output of the ADC. It has a programmable response time of $\tau = 2^n$ sampling periods, where n is a value between zero and eight which needs to be adjusted to fit the (high frequency) noise of the environment. As the signals we are interested in also have a low frequency component, which should not be disturbed or taken into account for the baseline correction, the IIR filter has to be protected against these. There are two ways foreseen to do so to interrupt the IIR update (both maskable):

ALTRO description, not updated

- during data acquisition
- If a signal is to far away from the actual calculated baseline

As the different modes of operation do not always allow to specify the acquisition window before the data is processed the first option is not enough. Furthermore in typical environments it is likely to have physical pulses even if there is no trigger. The second option on the other hand does have a start-up problem, as the initial baseline is not known. To deal with that the baseline thresholds are disabled for a fixed period of 5 time the programmed τ to make sure, the baseline being reasonable good calculated.

ALTRO description, not updated

To remove systematic effects, a pattern memory (pedestal memory) is used. If the memory is programmed beforehand with the shape of the systematic perturbation, then the shape can be subtracted from the input signal for each triggered acquisition. Alternatively, the memory can be used as a Look-Up Table (LUT) to perform non-linear conversion or to equalize the response across different channels. As a test feature, this memory can inject a pattern in the processing chain to allow the testing of all the logic downstream without the need of an external analogue signal.

The two aforesaid circuits allow for 3 different modes of operation: subtraction mode, conversion mode and test mode. Some of these modes of operation can be combined allowing numerous configurations of the BC1 circuit. The most relevant configurations have been summarized in table 1.1 while the complete list is reported in table 3.19. Hereafter we describe the main modes of operation.

Subtraction mode In this mode of operation, the BC1 performs the subtraction of spurious signals from the input-signal values. The subtracted signal can be fixed (fixed subtraction mode), time-dependent (time-dependent subtraction mode) or self-calibrated (self-calibrated subtraction mode).

Fixed-subtraction mode The value to be subtracted from the input signal is constant and stored in a configuration register.

Time-dependent subtraction mode The time-dependent pedestal values which are to be subtracted from the incoming signal are stored in a memory (pedestal memory) that, in this configuration, is addressed by a time counter started by the trigger signal. The 10-bit word values in the memory can be chosen to have an accuracy of 2, 1, 0.5 or 0.25 ADC counts.

Self-calibrated subtraction mode The value to be subtracted is computed as cumulative average of a programmable number of samples (n) of the input signal outside the processing time window.

$$Out = ADC - baseline \tag{1.1}$$

ALTRO formula, not updated

$$baseline' = \frac{(2^n - 1) \cdot baseline + ADC}{2^n} \tag{1.2}$$

While the fixed-mode and time-dependent-mode are exclusive, any of them can be combined with the self-calibrated mode as shown in table 1.1.

Conversion mode The circuit can perform a memory (static) conversion of the input signal of the type $y_n = F(x_n)$. At any cycle n, the output y_n depends at most on the input sample x_n at the same time, but not on past or future samples of the input. The output values y_n are stored in the pedestal memory addressed, in this case, by the input values x_n. The conversion mode can work concurrently to the self-calibrated subtraction mode and to the fixed subtraction mode.

Test mode The LUT can be used to generate a pattern to be injected into the processing chain for test purposes. It also supports subtraction a constant value.

Finally, the BC1 circuit provide also the possibility of inverting the input signal polarity (1's complement). The pedestal memory is accessible, in write and read mode.

Modes of operation		Main configurations						
		din - FPD	din - f(t)	din - VPD - FPD	din - VPD - f(t)	f(din) - FPD	f(din - VPD) - FPD	f(t) - FPD
Subtraction mode	Fixed	x		x		x	x	x
	Time-dependent		x		x			
	Variable			x	x		x	
Conversion mode						x	x	
Test mode								x

Table 1.1: *BC1 baseline correction and subtraction modes. Legend: din : data input (samples); f(t) : LUT data; FPD : fixed pedestal data value; VPD : variable pedestal data value; f(din) : converted data.*

1.4.2 Digital shaper

Merge descriptions

ALTRO description

ALTRO description, not updated

The SAMPA can be suited for a wide class of applications. One of its applications is the readout of the cathode pad plane of a conventional multi-wire proportional chamber. In this detector, the necessary signal amplification is provided by an ionization avalanche created in the vicinity of the anode wires. Moving from the anode wire towards the surrounding electrodes, positive ions, created in the avalanche, induce a positive current signal on the pad plane. This current signal is characterized by a fast rise time (less than 1 ns) and a long tail with a rather complex shape, which depend on the details of the wires and pad geometry.

The signal tail increases the superimposition of subsequent pulses (pile-up) rendering the zero suppression quite inefficient. In order to minimize such effect, the SAMPA incorporates a filter for the cancellation of the signal tail. The algorithm used for the tail cancellation is the same that it was used in ALTRO chip. It is explained hereafter. The signal is approximated by the sum of 4 exponential functions:

$$is(t) = I_0 \times \sum_{i=1}^{4} A_i \times e^{-\frac{t}{\alpha \cdot \tau_i}} + r(t) \begin{cases} \tau_1 \ll \tau_2 \ll \tau_3 \ll \tau_4 \\ \sum_{i=1}^{4} A_i = 1 \end{cases} \tag{1.3}$$

Where r(t) is a residual function due to the approximation error. The sum of the gains A_i should be equal to 1 so that input and output have the same amplitude. The time function equation (1.3) can be expressed in the Z domain as:

$$Is(z) = I_0 \times \sum_{i=1}^{4} \frac{A_i}{1 - exp(\frac{T}{\alpha \cdot \tau_i} \cdot z^{-1})} + R(z) \tag{1.4}$$

The signal is passed through a linear network that cancels all but the fastest of the exponential terms. The n-1 pole-zero network has a transfer function that expressed in the Z domain is:

$$H(z) = \frac{(1 - exp(\frac{T}{\alpha \cdot \tau_2}) \cdot z^{-1})(1 - exp(\frac{T}{\alpha \cdot \tau_3}) \cdot z^{-1})}{1 - L_1 z^{-1} + L_2 z^{-2} + L_3 z^{-3}}) \tag{1.5}$$

The numerator of F(z) will perfectly cancel all the poles of Is(z) except one. The constants L1, L2 and L3 are chosen such that the numerator of the expanded form of Is(z) disappears. The response of this linear network to the incoming signal is the convolution in the time of the impulse response function of the filter and the signal itself:

$$is(t) * f(t) = I_0 e^{-\frac{t}{\alpha \cdot \tau_0}} + r(t) * f(t) \tag{1.6}$$

One can easily observe from this expression that the performance of the tail cancellation is strongly related to r(t). The remaining fast exponential is a constraint of the system and can be chosen such that:

$$e^{-\frac{t}{\alpha \cdot \tau_i}} < 0.1\% \qquad t \geq 1\mu s \tag{1.7}$$

The filter considered is an IIR filter of order 4. The filter is composed of 4 first order filters in cascade, although just 3 first order filters are used in this application. The filter is flexible in the configuration of the digital signal processing operation by changing 8 programmable and accessible coefficients, K1, K2, K3, L1, L2 and L3, for each filter. In this case, K4=L4=0.

The processing performed is shown in figure 1.5.

S-ALTRO description

S-ALTRO description, not updated

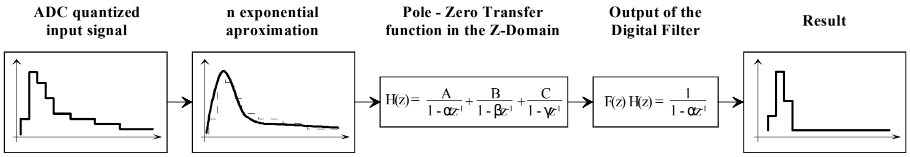

ADC quantized input signal
n exponential aproximation
Pole - Zero Transfer function in the Z-Domain
Output of the Digital Filter
Result
$H(z)=\frac{A}{1-\alpha z^{-1}}+\frac{B}{1-\beta z^{-1}}+\frac{C}{1-\gamma z^{-1}}$
$F(z)\,H(z)=\frac{1}{1-\alpha z^{-1}}$

Figure 1.5: *Tail cancellation scheme.*

There are different architectures possible for the IIR filter. The possibility to implement complex-conjugate poles/zeros (second order) and the advantage of lower sensitivity to quantification error (cascade form) make optimal the cascade combination of two second order filters.

The different architectures for IIR filters have been compared in terms of hardware resources, overflow and errors due to round-off and coefficient quantification. Being the transposed form the optimal architecture in this application, mainly because the overflow in the internal nodes is not a problem for fast impulses in the input.

Quantization analysis

When the parameters of the system are quantized, the poles and zeros move to a new position in the z-plane. Obviously, the frequency response is perturbed from its original value. This effect could modify the specifications of the original filter, or an IIR filter might even become unstable. Coefficient quantization errors are more sensitive to high order sections. There are two ways to reduce the order through the transfer functions decomposition in sections of less order: cascade-form and parallel-form. The cascade-form is the less sensitive to coefficient quantization [1??]. These effects were studied via the implementation of the transfer functions of the filters in Matlab. The outputs show that 13bits second order filters in cascade form do not have coefficient quantization effects in the output.

Round-off analysis

The simple linear-noise model presented (Figure **??**) allows to characterize the noise generated in the system (using Matlab) by averages such as the mean and variance and to determine how these averages are modified by the system.

$$\sigma_{ADC}^2 = \frac{2^{2\cdot bits}}{12}(V_{max} - V_{min})^2 \tag{1.8}$$

$$\sigma_{er}^2 = \frac{2^{2\cdot bits}}{12}(V_{max} - V_{min})^2; \qquad \sigma_{ei}^2 = \sum_{i=0}^{n} \sigma_{er}^2 \tag{1.9}$$

$$\sigma_{y_noise}^2 = \sigma_{ADC}^2 \cdot \sum_{n=-\infty}^{\infty} |h[n]|^2 + \sigma_{ei}^2 \cdot \sum_{n=-\infty}^{\infty} |h_e[n]|^2 \tag{1.10}$$

$$\sigma_x^2 = \frac{1}{12}(V_{max} - V_{min})^2 \tag{1.11}$$

resolution	SNR
ADC (10 bits)	66.22 db
ADC (10 bits) + Shaper (10 bits)	53.99 db
ADC (10 bits) + Shaper (11 bits)	59.30 db
ADC (10 bits) + Shaper (12 bits)	63.25 db
ADC (10 bits) + Shaper (13 bits)	65.27 db

Table 1.2: *Signal to noise ratio of the filter*

$$\sigma_y^2 = \sigma_x^2 \cdot \sum_{n=-\infty}^{\infty} |h[n]|^2 \tag{1.12}$$

$$SNR(db) = 10 \cdot \log \left(\frac{\sigma_y^2}{\sigma_{y_noise}^2} \right) \tag{1.13}$$

A common objective in round-off analysis is to choose the digital word length such that the digital system is a sufficiently accurate realization of the desired linear system and at the same time requires a minimum of hardware or software complexity. The digital word length can be changed only in steps of 1 bit, the addition of 1 bit to the data-path reduces the size of quantification error analysis by factor of 2. Table **??** shows the signal to noise ratio of the system. Following a commitment between accuracy system and hardware complexity, it is added two extra bits (LBS) in the data path.

Overflow analysis

The possibility of overflow is an important consideration in the implementation of IIR systems using fixed-point arithmetic. In this architecture the internal values are calculated from a subtraction of the input and the output value (Figure **??**), the internal values does not increase as much as the other architectures. The transposed form is the only configuration with no overflow error for this application.

Tail cancellation parameters

The eight parameters for the Digital Shaper can be set individually on each input of the channel in order to maximize its effect. The optimal parameters for the tail cancellation can be found with the following algorithms.

- Mota method (real poles/zeros): in this method is proposed to divide the complex ion tail into its various effects and consider them separately. In order to suppress a given effect, a correction function with an opposite impulse response behavior should be chosen. The parameters are calculated via recursive formulas proposed in [2??]. Next equations show the relation between Mota method parameters and S-ALTRO parameters.

$$b_{12} = L_2 + L_3 \tag{1.14a}$$

$$b_{13} = L_2 \cdot L_3 \tag{1.14b}$$

$$a_{12} = K_2 + K_3 \tag{1.14c}$$

$$a_{13} = K_2 \cdot K_3 \tag{1.14d}$$

b_{22}, b_{23}, a_{22}, a_{23} can be used for height normalization [2??]

- Riegler method (real poles/zeros): this method approximates the input signal by a sum of exponentials. The model function to calculate the parameters is determinate by the convolution of the IRF and the signal [3??] Following equations show the relation between Riegler method parameters and S-ALTRO parameters.

$$b_{12} = e^{-1/\tau_2} + e^{-1/\tau_3} \tag{1.15a}$$

$$b_{22} = e^{-1/\tau_2} \cdot e^{-1/\tau_3} \tag{1.15b}$$

$$a_{12} = e^{-1/\tau_a} + e^{-1/\tau_b} \tag{1.15c}$$

$$a_{22} = e^{-1/\tau_a} \cdot e^{-1/\tau_b} \tag{1.15d}$$

b_{22}, b_{23}, a_{22}, a_{23} can be used for height normalization [3??]

- New method (complex poles/zeros): still it's being studied$\cdots$

1.4.3 Baseline correction and subtraction II

A second level of baseline correction can be applied to the signal to correct for signal perturbations created by non-systematic effects. This level of correction is based on a moving average filter. This functionality is performed in two different levels, one is the generation of the window to perform the average of the baseline (acceptance window), and the other is the correction itself. The correction of the baseline is based on a moving average filter.

The acceptance window is based on a double threshold scheme, one above and one below, that follows the slow variations of the signal (figure 1.6). Inside the acceptance window, the baseline is updated according to the following equation:

$$y(n) = \frac{1}{M+1} \sum_{k=0}^{M} x(n-k) \qquad M = 1,3,5 or 7 \tag{1.16}$$

This value is the result of the moving average of a signal $x(n)$. In case all previous values were inside the threshold, then it is the average of this sample and the previous 1, 3, 5 or 7 (depending on the configuration) and the current sample value will be corrected with this value.

When there is a fast variation in the signal, like a pulse, the samples would pass out of the acceptance window, and therefore excluded from the baseline calculation. In this case the value of the samples are corrected with the value calculated by the moving average filter for the last sample that was inside the window.

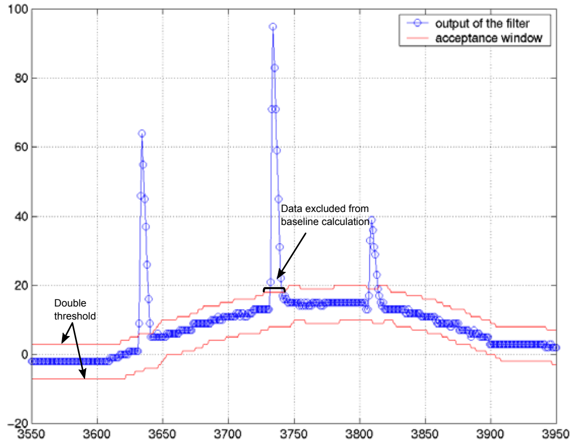


Figure 1.6: *Moving average principle.*

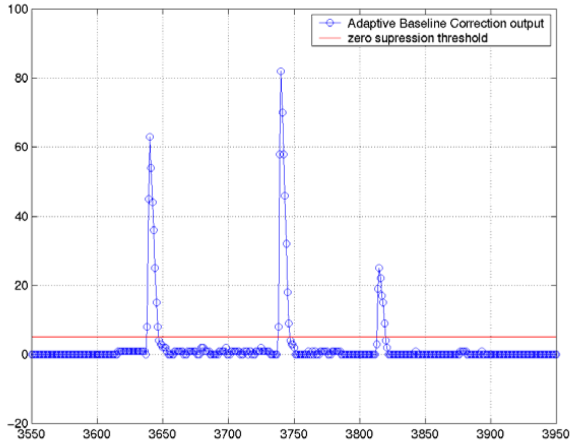


Figure 1.7: *Data after adaptive baseline correction.*

A setting to additionally exclude a programmable number of samples before and after any pulse that passes out of the exclusion thresholds is available. This could be helpful if the baseline is generally very stable.

To combat some of the problems that were experienced previously with the ALTRO BC2 filter, where the filter was sometimes seen to get stuck outside the threshold ranges, a configurable auto reset feature have been implemented to solve this.

1.4.4 Baseline correction III

Conventional linear filters like the BC1 and BC2 have been used in the past in particle detector experiments and are useful for the task, but in some exceptional cases they can end up in a failure mode in where the baseline of the signal shifts to outside the thresholds of the filter and the filter stops to operate.

In general the filter has a max slope that can be handled before it needs to be reset or the thresholds need to be changed.

A new nonlinear filter called BC3 has been included which does not have any thresholds. This makes the baseline tracking free of dead areas and the need for fine tuning of configurations. The main idea behind this filter is to always follow the baseline, but with a limited slope configurable per channel. The slope parameter can be set individually for the up and down slope.

The output signal $h(t)$ is expressed as a function of the input $f(t)$ as follow:

$$h(t) = f(t) - g(t) \tag{1.17}$$

Where $g(t)$ is the tracked slope baseline value and can be expressed as:

$$g(t) = g(t-1) + \begin{cases} slope\uparrow & \text{if} f(t) > g(t-1) \\ 0 & \text{if} f(t) = g(t-1) \\ -slope\downarrow & \text{if} f(t) < g(t-1) \end{cases} \tag{1.18a}$$

$$g(0) = f(0) \quad \text{(arbitrary)} \tag{1.18b}$$

The filter has also a maximal slope as BC2, but exceeding it causes only a time-limited deviation. As the filter does not have acceptance thresholds it also does not stay constant during an input pulse so the output pulse will be slightly affected.

Figure 1.8 demonstrates the standard behavior of the filter when changes of the baseline and different input pulses are applied.

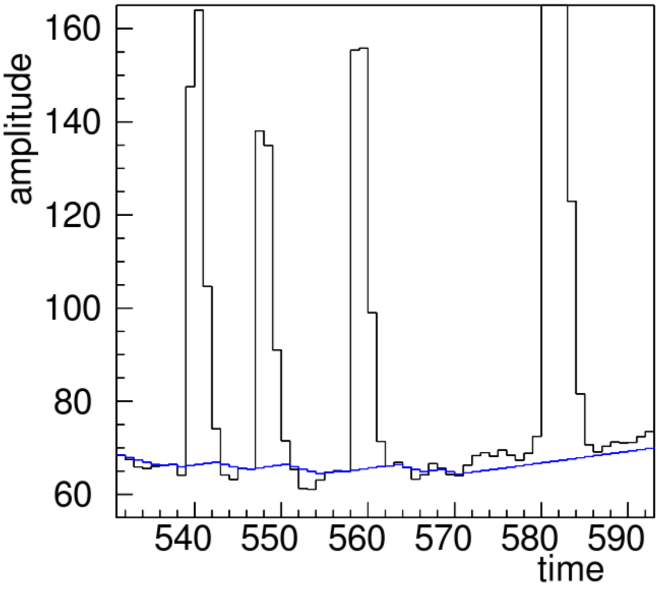


Figure 1.8: *BC3 filtering principle.*

1.5 Compression

Even though the SAMPA has enough output data links to handle reading out raw data, it is often not practical or economically feasible to do a raw readout. Different compression methods are provided to reduce the data amount enough so that fewer serial data links can be used. Both lossless and lossy compressions are available, depending on the detectors needs and resources.

Huffman This compression method is a lossless compression, ie. no information is lost. It uses differential encoding of the data combined with a lookup table to reduce the length of each word. This method is useful for detectors that want all data, but has limited bandwidth available. Depending on the detector data, it has a compression factor close to or better than zero suppression.

Zero suppression This method removes all data below a given threshold, leaving only cluster data. As the data which is in between clusters are lost, it is a lossy encoding. This method is highly dependent on having a stable baseline to achieve good compression and minimal loss of information, so digital signal conditioning might be needed.

Zero suppression with cluster sum This method operates on the same principle as zero suppression, but it additionally integrates the sample values in the cluster into one value. This compression method is suitable for detectors that have very clean signals and where only the time for the start of the cluster and the area is of interest.

1.5.1 Huffman compression

1.5.2 Zero suppression

The basic pulse detection scheme is fixed thresholding: samples of value smaller than a constant decision level (threshold) are rejected. When a sample is found above the threshold, it is considered the start of a pulse (figure 1.9).

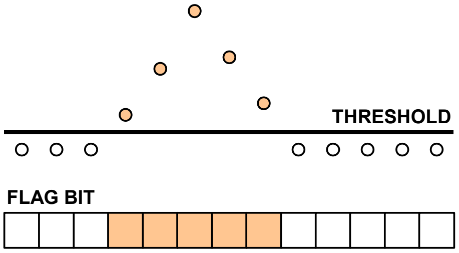


Figure 1.9: *Basic detection scheme.*

In order to keep enough information for further feature extraction, the complete pulse shape must be recorded. Therefore, a sequence of samples (pre-samples) before the signal overcome the threshold and a sequence of samples (post-samples) after the signal returns below the threshold are also recorded (figure 1.10). The number of presamples and the number of post-samples can vary independently in the range between 0 and 3 for pre-samples and 0 to 7 for post-samples.

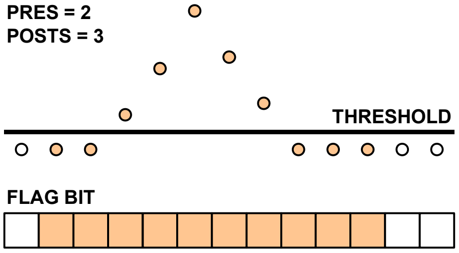


Figure 1.10: *Feature extraction with two extra samples before pulse and three after.*

To reduce the impulsive noise sensitivity, a glitch filter checks for a consecutive number of samples above threshold, confirming the existence of a real pulse (figure 1.9). The minimum sequence of samples above the threshold (glitch filter) which defines a pulse can vary from 1 to 3, not including any pre- or post-samples.

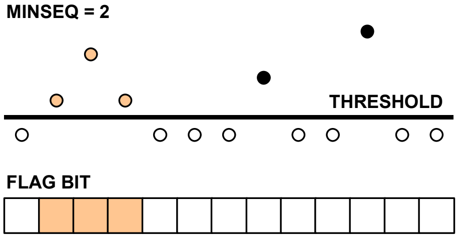


Figure 1.11: *Glitch filtering with minimum samples above threshold of 2. Samples in solid black are treated as if they were below the threshold.*

The pulse thus identified and isolated must be tagged with a time stamp, in order to be synchronized with the trigger decision for validation. Otherwise the timing information would be lost by the removal of a variable number of samples between accepted pulses. This requires the addition of a time data to the set of sample data. Besides that, in a data format where the addition of flag bits is not allowed, a further word is needed to distinguish the sample data from the time data. This extra word represents the number of words in the set. Since for each new set of data we have two extra words, the merging of two consecutive sets, which are closer than 3 samples, is performed (figure 1.12).

In case of the cluster sum mode, the data in one cluster is summed together into a 20 bit word and the requirement for the minimum distance between clusters is then changed to 3.

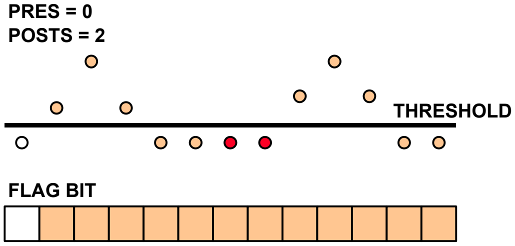


Figure 1.12: *Merging of close clusters. Samples in red are included to make one complete cluster.*

1.5.3 Lossy data compression

Due to the removal of a variable number of samples between accepted clusters in the lossy encodings (zero suppression, cluster sum), the timing information would be lost in the process. This requires the addition of a time-data to each accepted set of samples. Since 1023 is the maximum length of the data stream that can be processed by the SAMPA chip, the time information can be encoded in a 10-bit word. The principle is to label each sample with a time-stamp that defines the time distance from the trigger signal. The time information added to each cluster during the formatting phase corresponds to the time-stamp of the first sample in the cluster.

The SAMPA data format does not make use of extra flag bits to distinguish the samples data from the time-data, but introduces a further word for each accepted cluster, which represents the number of samples in the cluster. In the ALTRO, the cluster size data included also the time information data and cluster information data in the count of the total number of words in the cluster, the SAMPA does not count them into the total.

These new 10-bit words, time data and number of samples per cluster, are introduced at the beginning of the cluster (figure 1.13) in contrast to how it operated in the ALTRO chip where the information words were located after the cluster data.

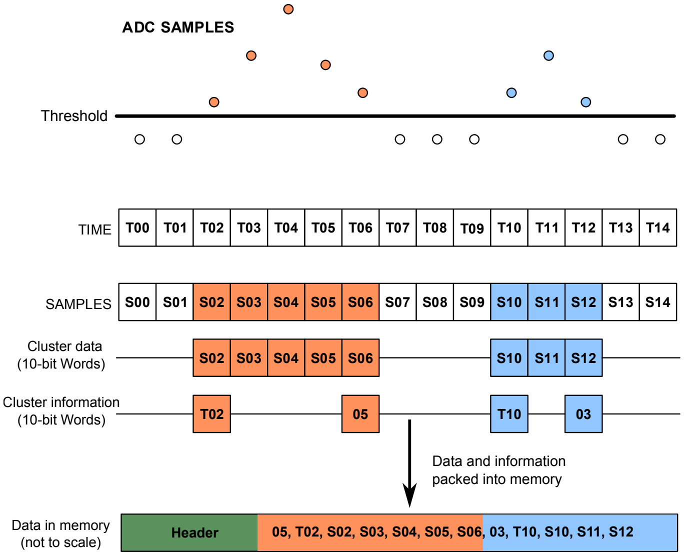


Figure 1.13: *The SAMPA data format for zero suppression encodings.*

A header of five 10-bit words (50 bits in total) are inserted at the start of a processing time window, where among other things the total length of the compressed data stream is recorded. See section §2.3.1 for more information on the header format.

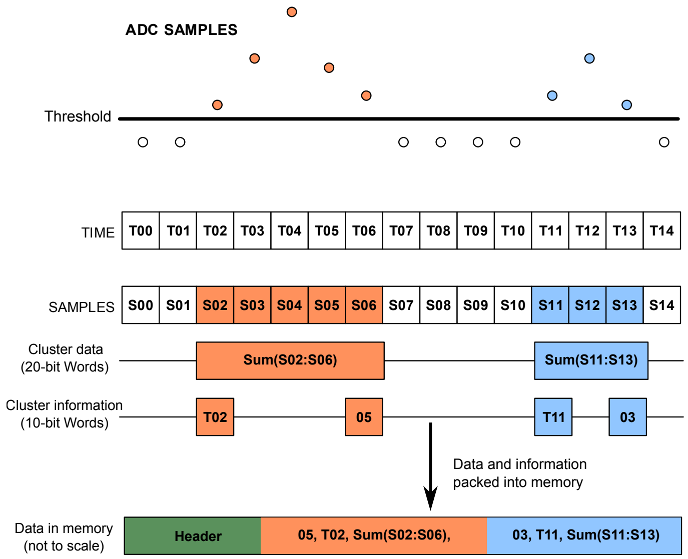


Figure 1.14: *The SAMPA data format for zero suppression encoding with cluster sum. The sum part is 2 words.*

1.6 Channel ordering

1.7 Data formatting unit and Ring buffer

The Data Formatting Unit (DFU) and the Ring buffer are the last two blocks of each channel as shown in figure 2.9. The DFU generates the packet format for serial transmission and the Ring buffer provides a temporary storage for the data and headers.

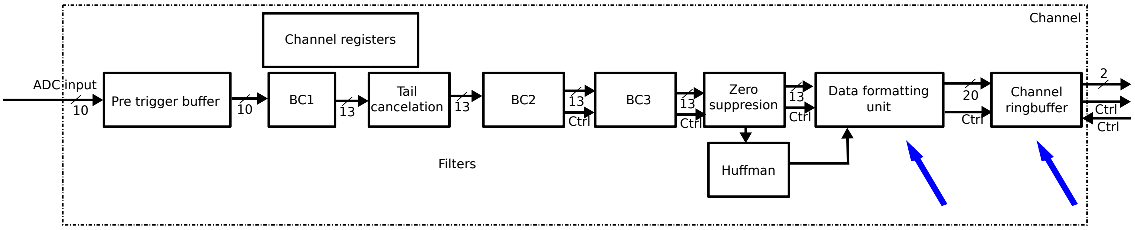


Figure 1.15: *DFU + RB in channel circuit diagram.*

1.7.1 Data formatting unit (DFU)

Data arriving from Zero Suppression block or Huffman compressor is analyzed in the DFU block and cluster information as number of samples and timing is inserted for the generation of the payload part of packets for serial transmission (see section §1.5.3). Optionally, payloads can be generated from raw data. It can also generate data compression as the sum of all the cluster samples, denominated cluster sum mode. These options can be configured through VACFG global register.

The DFU receives information of start and end of time window (tw_start and tw_end) and a time count that is restarted at the beginning of each time window, see figure 1.16. It also receives signals to determine the start and end of each cluster (cluster_start and cluster_end). The number of samples of each cluster is counted and transmitted to the Ring buffer along with the value of the last sample in the 20 bits dout signal. The time count of the sample that arrives one cycle early is added at the beginning of the cluster. See these details in figure 1.17.

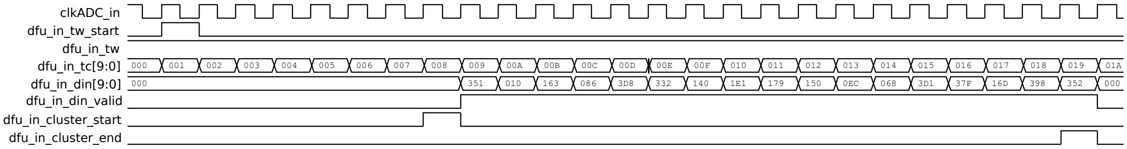


Figure 1.16: *Waveforms of time window, cluster and data DFU inputs.*

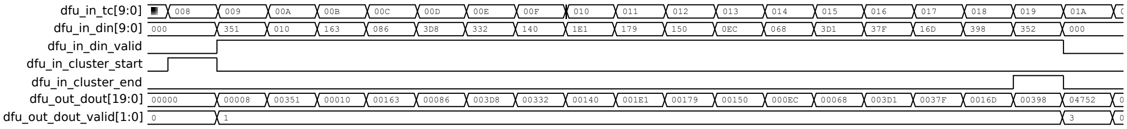


Figure 1.17: *Control signals of time window and cluster and data DFU input/output.*

If a cluster starts in one time window and ends in the following, for example in continuous mode, at the output it is divided in two clusters as in figure 1.18.

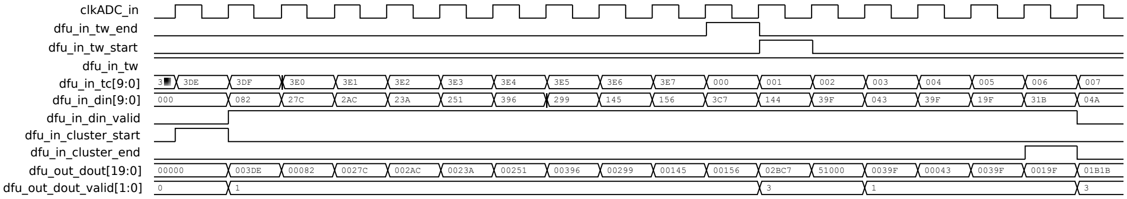


Figure 1.18: *Cluster starting in one time window and ending in the next one.*

In cluster sum mode, the DFU transmit to the ring buffer the time count of the first sample of the cluster when the cluster starts. When the cluster ends the first word at the output of the DFU is the cluster size, then the sum of values of the samples in the cluster. See figure 1.19 for details.

1.7.2 Ring buffer

The Ring Buffer contains two temporary memories one for payloads, data memory of 10 bits X 6144 words, and another for headers, header memory of 10 bits X 256 words. These memories allow simultaneous reading and writing operations at different rates through two ports in each one. IP memory manufacturing defects can be detected through a built in test block in ring buffer block. Also, if an error occurs in header memory it can be corrected using Hamming code. Two or more errors per header can be detected but not corrected. Being the last block of each channel, it interfaces with the serial output block. As the reading can be slower than the writing of data or headers, memory overflow can occur. For the data memory overflow header packets are generated indicating number of words of payload zero (data truncated

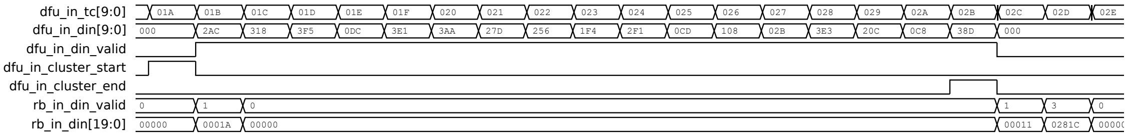


Figure 1.19: *Waveforms of DFU and Ring Buffer signals in cluster sum mode*

packet type). If the header memory overflows the packets are discarded. But in both cases the Ring Buffer recovers and continue to write to and read packets from memory correctly. Empty packets generation and transmission, when there is no any sample value above the zero suppression threshold in a time window, can be configured through the VACFG global register.

The Ring Buffer generates the headers of the packets adding bunch crossing count for event counting, packet type, payload parity information, hardware address, channel address, number of words in the associated payload, hamming code for header error detection and correction and header parity information. For more details about headers see section §2.3.

The data is received from the DFU including time count and cluster size. The tw_start_ADC and tw_end_ADC signals establish the limits of the time window and they arrive to the ring buffer one cycle later than those for the DFU. The clusters in a time window are included in a payload and the header generated. When a packet is ready to be transmitted, the DR_SOD2 signal is set to '1'. The figure 1.20 shows details.

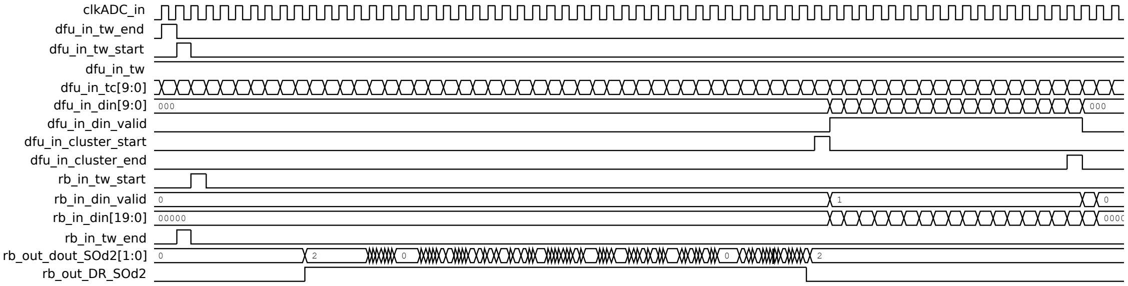


Figure 1.20: *Waveforms of input/output signals of DFU and Ring Buffer*

When in trigger mode if a trigger pulse is received before time window end a trigger too early packet type is generated. At the trg_overlap input a pulse is generated indicating that this type of packet is to be generated by the Ring buffer. Also, at the end of the time window a new packet is then initiated. A trg_overlap pulse generation example is in figure 1.21.

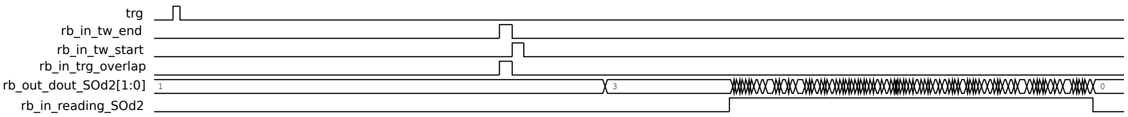


Figure 1.21: *Waveforms of ring buffer input/output signals when trigger early*

1.8 Serializing

The SAMPA allows the configuration of the functionality of the 11 data output pins. Configurations are for the standard modes, i.e. modes intended for normal usage, and test modes.

1.8.1 Standard Modes

The SAMPA read-out may be performed in DSP mode or in Direct ADC Serialization mode (DAS mode). When in DSP mode all the processing capabilities of the SAMPA and up to eleven serial links at 80, 160 or 320 Mbps are available. To this mode be enabled the clk_config[6] and the sme input pins must to be in '0' and the BYPASS global register also in 0x0 (default value). By default the number of enabled serial links is four, but it may be modified through the SOCFG[3:0] global register. See section §2.3 for more details about the serial interface and protocol. In daisy chaining NBflowstop_out_SO5 output pin is used to halt the transmission of data from an upstream device. See section §2.4. This is its default functionality. However, it can be used as the fifth serial link when enabling six or more links through SOCFG[3:0].

When in Direct ADC Serialization mode, the samples of the 32 ADCs and the end of conversion signal are multiplexed to the 11 data output pins and sequentially transmitted at the serial output clock frequency. To enable the DAS mode the clk_config[6] and the trigger pin (trg) must be in '1' and the heartbeat trigger (hb_trg) input pin must be in '0'. The BYPASS global register must also be in 0x0 (default value). See section §2.5 and section §4.15.

1.8.2 Test Modes

The 11 data outputs, SerialOut[4:0], NBflowstop_out_SO5 and sdo[4:0], may also be set to work in ADC combo data mode for testing with clk_config[6] and hb_trg in '1'. In this mode, the output data and the end of conversion of a selected ADC is directly connected to the output pins. The ADC selection is performed using the sme and the hardware address (hadd) input pins.

The SerialOut[0] output may be used for internal memory test indicating errors through pulses. This mode is enabled setting the input pin sme. SerialOut[0] may also be configured to bypass one of 14 signals according to the BYPASS global register. See table 3.13 for specifics.

Other test functionality is the bypass in daisy chaining. In this case NBflowstop_out_SO5 output pin is NBflowstop_in enabled through BYPASS and SOCFG[5] registers.

1.9 Event managing

SAMPA supports two read-out modes: continuous mode and external triggered. In continuous mode a new time window is started when the previous one has ended. The length of the time window can be set up to 1024 samples long. In triggered mode, upon reception of the external trigger, a new time window is started. At the end of the time window, the SAMPA will go back to being idle until the next trigger arrives. The trigger is sent either via an external pin with a maximum latency of 192 times the sample rate. All 32 channels of the SAMPA uses the same time frame. If an external trigger signal is received in continuous mode, the current time window is ended and a new one is started immediately without loss of information. In this way it is possible to align the time frames across different SAMPA chips.

Normally there will be sent a packet containing a header and the compressed data pertaining to the completed time window, one packet per channel, each time a time window is completed. Though an option is available to disable sending of packets that contain no data, where there was no data above the zero suppression threshold.

The SAMPA has an internal 20 bit counter, that runs on the LHC 40 MHz clock, called the bunch crossing counter. At the start of each new time window the value of this counter is captured and added to the header of the packet for that time window as a bunch crossing id. If the ADC sampling clock is derived from the same source clock as the 40 MHz clock it is possible to use this value as an event number id by calculating :

$$event\,num = \frac{bx\,counter}{time\,window\,length \cdot \frac{40MHz}{F_s}} \tag{1.19}$$

On reception of a bunch crossing sync trigger the SAMPA will reset the counter. In this way it is possible to align the bunch crossing id across different SAMPA chips.

If the 40 MHz or the ADC sampling clock is set up to be internally derived from another clock, then the phase difference between these clocks for one SAMPA versus another might not be the same after a cold start. This would present it self as differences in a few counts of the bunch crossing id for devices that are synchronously triggered. To synchronize the internal clocks the devices need to be also reset at the same time or the reset signal must be applied to one device so that it lines up with the others.

For off-site synchronization the SAMPA provides the generation of a specially crafted heart beat packet once a heart beat trigger signal is provided. The packet contains among other things the bunch crossing ID when the trigger arrived and the chip address.

1.10 Daisy chaining

To simplify wiring and for detectors with very low data rates a daisy chaining option has been implemented which lets multiple SAMPAs share a single serial link. Each SAMPA uses a data input port, a data output port and busy in and busy out signals. When a device receives a busy signal from a device downstream it will pause its transmission until the downstream device is ready for receiving data again.

1.10.1 Neighbor module

The serial link of the device upstream (neighbor) is connected to the dinN input pin of the device that shares the single serial link (master). The packets arriving at the master are then received by the neighbor block. This module is composed by three main sub blocks as seen in figure 1.22.

The delay compensation block is to avoid metastability caused by the same serial output frequency of neighbor and master devices. A delay can be set through the NBCFG[5:0] global register. This register also allows the configuration of device priorities. See 3.1.6 for more details. The delayed input signal is also

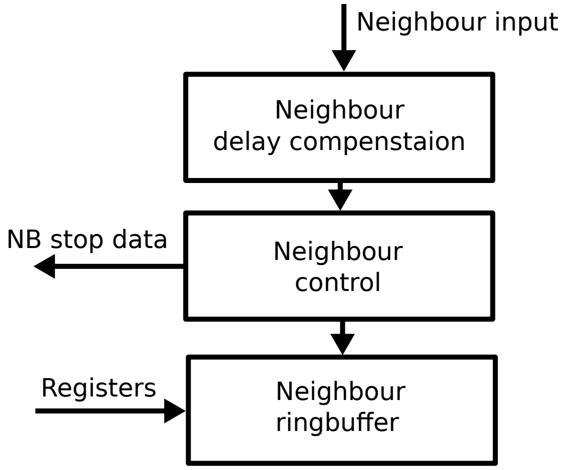


Figure 1.22: *Neighbor block diagram.*

available at the output of the neighbor block for the bypass option at the serial output block for testing.

After the delay is applied to the input signal, the hamming code of headers is decoded and the header can be corrected. If the header is not correctable then the packet is discarded.

The received packets are then stored in the neighbor ring buffer block using a data memory of 2 KW and a header memory of 256 W. To avoid the loss of packets due to memory overflow, the neighbor control generates a NBflowStop_SO5 signal to the upstream device to halt packet transmission. The internal signal h_full indicates that the arriving of a new packet can produce header memory overflow. Similarly, the d_full is an indication that the arriving of a new packet can produce data memory overflow. The activation of any of these signals activates the NBflowStop_SO5 and the neighbor SAMPA waits the NBflowStop_SO5 deactivation to transmit a new packet. In the example of the figure 1.22 the data is received by the dinN input. When the data memory is full the d_full signal is activated and then the NBflowStop_SO5 signal, when the current packet is finished, at dinN sync packets are received until NBflowStop_SO5 is deactivated and a new packet transmission is initiated.

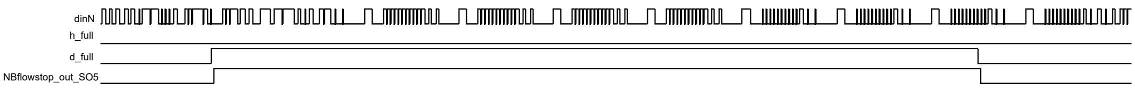


Figure 1.23: *Neighbor block diagram.*

1.11 Direct ADC serialization mode

For detectors that would not prefer to use the data handling capabilities of the SAMPA, a mode is available where the raw ADC samples are directly serialized and the rest of the digital circuitry is powered down. This mode operates with a serialization speed of 32 times the ADC sampling speed. Two data transmission modes are available:

Normal mode The 10 bit data for channel 0 will be put on the serial link 9-0 in the first cycle, in the consecutive cycle the data for channel 1 will be put out and so on.

Split mode The serial link 4-0 will be used by channel 0-15 and link 5-9 will be used by channel 16-31. In the first cycle the 5 lower bits for channel 0 will go on link 4-0 and the 5 lower bits for channel 16 will go on link 9-4. On the consecutive cycle the 5 upper bits for channel 0 will go on link 4-0 and the 5 upper bits for channel 16 will go on link 9-4 and so on. This mode enables the data for channel 0-15 and 16-31 to be sent to two different upstream receivers.

A 32 cycle sync pattern is provided for synchronization to the stream when the direct ADC serialization mode is enabled. To be able to monitor for phase shifts in the ADC clock caused by single event upsets in the internal clock divider one can monitor the 11th serial link which provides the internally generated clock directly outputted.

Chapter 2

Interfaces and protocols

This section describes the protocols and properties of the various interfaces for the SAMPA.

2.1 Analog front-end

The analog front-end can be configured to operate at two shaping time settings controlled by the CTS pin, three sensitivity settings controlled by the CG[1:0] pins and positive/negative input charge polarity controlled by the POL pin. Table 2.1 summarize the possible configurations.

Polarity	POL
Positive	low
Negative	high

(a) *Pulse polarity configuration*

Gain	Shaping time	CTS	CG0	CG1
30 mV/fC	160 ns	low	high	high
20 mV/fC	160 ns	low	low	high
4 mV/fC	300 ns	high	low	low
Unsupported modes				
9 mV/fC	160 ns	low	low	low
33 mV/fC	160 ns	low	high	low
16 mV/fC	300 ns	high	low	high
11 mV/fC	300 ns	high	high	low
17 mV/fC	300 ns	high	high	high

(b) *Gain and shaping time configuration*

Table 2.1: *Analog Front-end configurations*

2.2 Slow Control

The slow control of the SAMPA is done with I2C. Pin positioning is given in table 2.2.

The protocol follows the I2C standard [1] and uses the 10-bit addressing scheme and address increment on continued read/write operations. It can operate from 100 kHz to 5 MHz. A watchdog is implemented that

Pin	Name	Dir	Type	Description
D21	SDA	I/O	LVCMOS (1.2 V)	I2C data
D22	SCL	I	LVCMOS (1.2 V)	I2C clock

Table 2.2: *Slow control interface pins.*

will time out if the time between two falling edges of SCL is more than 512 BX clock cycles (generally 12.5 µs). Clock stretching is not in use. The protocol is shown in table 2.1 and table 2.2 with the description in table 2.3. Boxes marked in gray are bits sent by the SAMPA.

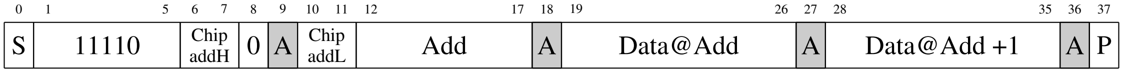


Figure 2.1: *Format for writing to the SAMPA.*

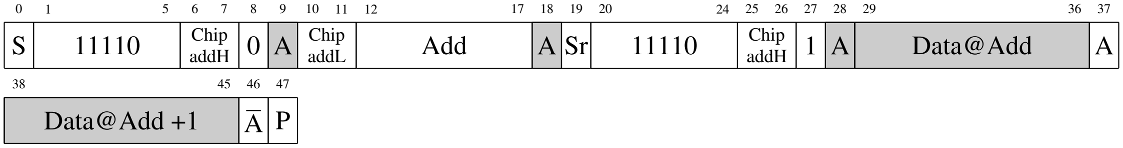


Figure 2.2: *Format for reading from to the SAMPA.*

Name	Bits	Description
S	1	I2C start
Sr	1	I2C start repeat
A	1	I2C acknowledge
$\bar{A}$	1	I2C not acknowledge
P	1	I2C stop
11110	5	Fixed preamble address for 10 bit addressing
Chip addH	2	Chip address [3:2]
Chip addL	2	Chip address [1:0]
Add	6	Register address
Data	8	Register data to be read/written

Table 2.3: *Protocol bit field descriptions of I2C.*

2.3 Serial interface

The SAMPA is equipped with eleven SLVS [2] serial links clocked in relation to the serial clock provided by the interfacing device. The relevant pins for the serial interface is shown in table 2.4.

Pin	Name	Dir	Type	Description
L19	SerialOut+[0]	O	SLVS	Serial link 0 p
L18	SerialOut-[0]	O	SLVS	Serial link 0 n
M22	SerialOut+[1]	O	SLVS	Serial link 1 p
M21	SerialOut-[1]	O	SLVS	Serial link 1 n
K22	SerialOut+[2]	O	SLVS	Serial link 2 p
K21	SerialOut-[2]	O	SLVS	Serial link 2 n
H22	SerialOut+[3]	O	SLVS	Serial link 3 p
H21	SerialOut-[3]	O	SLVS	Serial link 3 n
J19	SerialOut+[4]	O	SLVS	Serial link 4 p
J18	SerialOut-[4]	O	SLVS	Serial link 4 n
G19	NBflowstop_out_SO5+	O	SLVS	Serial link 5 p
G18	NBflowstop_out_SO5-	O	SLVS	Serial link 5 n
C22	sdo+[0]	O	SLVS	Serial link 6 p
C21	sdo-[0]	O	SLVS	Serial link 6 n
Y22	sdo+[1]	O	SLVS	Serial link 7 p
Y21	sdo-[1]	O	SLVS	Serial link 7 n
B22	sdo+[2]	O	SLVS	Serial link 8 p
B21	sdo-[2]	O	SLVS	Serial link 8 n
AB20	sdo+[3]	O	SLVS	Serial link 9 p
AA20	sdo-[3]	O	SLVS	Serial link 9 n
B20	sdo+[4]	O	SLVS	Serial link 10 p
A20	sdo-[4]	O	SLVS	Serial link 10 n

Table 2.4: *Serial interface pins.*

2.3.1 Protocol

The data sent from the SAMPA consists of a fixed length header and a variable length data payload. The combined header and payload is here on referred to as a packet. The data is written LSB first on rising edge of the clock. The data payload contains zero suppressed and run length encoded samples in a forward linked list in a similar way as for the previous ALTRO chip, though the ALTRO uses a back linked list instead.

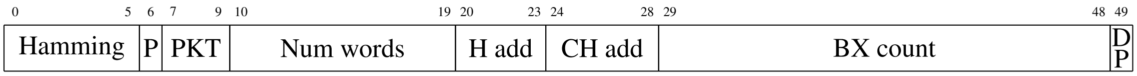


Figure 2.3: *Format of serial data header.*

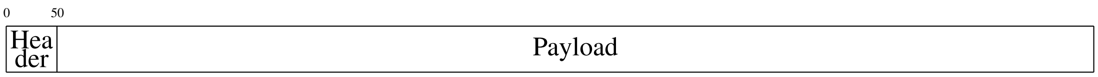


Figure 2.4: *Format of serial data. Length of payload is variable and given in the header information.*

2.3.2 Heartbeat packet

Heartbeat packets are specially crafted packets that have the same format as a header (50 bits) with the packet type marked as heartbeat type. They are initiated by a single pulse on the heartbeat trigger pin

Name	Bits	Description
Hamming	6	Hamming code
P	1	Parity (odd) of header including hamming
PKT	3	Packet type, see table 2.6
Num words	10	Number of 10 bit words in data payload
H add	4	Hardware address of chip
CH add	5	Channel address
BX count	20	Bunch-crossing counter (40MHz counter)
DP	1	Parity (odd) of data payload

Table 2.5: *Protocol bit field descriptions of data sent from the SAMPA.*

Data	1024 words	1025 words	Heartbeat	Sync	Trigger too early	Data truncated	Num words [0]	PKT [2]	PKT [1]	PKT [0]
X							0	1	0	0
X	X						0	1	0	1
X		X					1	1	0	1
X					X		0	1	1	0
X	X				X		0	1	1	1
X		X			X		1	1	1	1
			X				0	0	0	0
X						X	0	0	0	1
				X			0	0	1	0
X					X	X	0	0	1	1

Table 2.6: *Packet type coding (PKT).*

(hb_trg) and the bunch crossing counter is saved the moment the trigger is detected.

The heartbeat packets are only sent on serial link 0 and has the highest priority, it will be sent after the transmission of the current packet has been completed.

0 – 5	6	7 – 9	10 – 19	20 – 23	24 – 28	29 – 48	49
Hamming	P	0b000	0x0	H add	0b10101	BX count	0

Figure 2.5: *Format of heartbeat packet.*

Heartbeat trigger

This is the trigger signal for the heartbeat packet. It is sampled on the rising edge of the 40 MHz bunch crossing clock. See section §2.8

2.3.3 Sync packet

A sync packet is a specially crafted packet that has the same general format as a header (50 bits), but with the packet type marked as sync type and the other fields having a fixed value independent of the state of the chip. As the sync packets have a fixed pattern it is possible for the receiving system to use the pattern to synchronize against the incoming data stream. Sync packets are transmitted, to keep the link active, whenever there is no data available in the channel buffers to send.

It is possible to force the transmission of a sync packet by sending a command through the slow control, this will create a sync packet with a higher priority than the channel packets so the sync would be sent after the current channel packet is sent on all links. Two consecutive sync packets are always sent in this case.

0 5	6	7 9	10 19	20 23	24 28	29 48	49
0x13	0	0b010	0x000	0xF	0x0	0xAAAAA	0

Figure 2.6: *Format of sync packet.*

2.3.4 Types of packets

The packet type codes are in table 2.6. They are classified as:

data: [Num words [0], PKT] = 0x4. Data packets are those that contain input signal information. Empty packets are also considered as data type. The number of words of the payload is zero in empty packets, and they are generated when there are no pulses above the zero suppression threshold. No other kind of singularity is notified in this type of packet.

payload of 1024 words: [Num words [0], PKT] = 0x5. It is a particular type of packet whose payload is of 1024 words.

payload of 1024 words and trigger early: [Num words [0], PKT] = 0x7. It is a particular type of packet whose payload is of 1024 words, and its time window was generated by a trigger pulse that arrived before the previous time window had finished.

payload of 1025 words: [Num words [0], PKT] = 0xD. It is a particular type of packet whose payload is of 1025 words.

payload of 1025 words and trigger early: [Num words [0], PKT] = 0xF. It is a particular type of packet whose payload is of 1025 words, and its time window was generated by a trigger pulse that arrived before the previous time window had finished.

data packet and trigger early: [Num words [0], PKT] = 0x6. It is a normal data type packet, and its time window was generated by a trigger pulse that arrived before the previous time window had finished.

heartbeat: [Num words [0], PKT] = 0x0. It is a heartbeat type packet. That is a packet that indicates that the SAMPA chip responds correctly.

data truncated : [Num words [0], PKT] = 0x1. It is a data type packet without payload due to the data memory overflow in ring buffer.

sync : [Num words [0], PKT] = 0x2. It is a sync packet. It is a packet for the receiver synchronization.

data truncated and trigger early : [Num words [0], PKT] = 0x3. It is a data type packet without payload due to the data memory overflow in ring buffer and its time window was generated by a trigger pulse that arrived before the previous time window had finished.

2.4 Daisy chaining

To adapt the SAMPA for detectors with very low data rates a daisy chaining option has been implemented which lets multiple SAMPAs share a single serial link. Each SAMPA has a data input port (DinN) and two data-control signals for this purpose. A connection diagram can be seen in figure 2.7.

The upstream device is set up to run with one serial downlink and the Nbflowstop_out_SO5 signal from the downstream device is connected to the Nbflowstop_in of the upstream device. The NBflowstop signals acts as a busy signal and tells the upstream device to halt its transmission of data after the current packet is done. While waiting to send data again the upstream device will send sync packets to keep the communication in sync.

The downstream device has the serial out of the upstream device connected to its DinN connection. Sync packets arriving on the link will be filtered and only heartbeat and data packets will be retransmitted. The downstream device selects to send packets from either internally or from the upstream link in an equally shared fashion, depending on how many devices in total there are upstream. The Nbflowstop_in signal of the last device downstream in the chain should be tied low.

In the circuit diagram of figure 2.8 two SAMPAS are interconnected and their inputs are specified. The diagram illustrates an example of daisy chaining mode for clkSO of 80MHz, clkBX and clkADC internally generated and ADC sampling rate of 10 MSPS. The master SAMPA hardware address is configured as 0x1 and the neighbor SAMPA as 0x0. The polarity is configured as positive and the gain and peaking time as 4 mV/fC and 300ns. These inputs can be configured for different values if required.

Pin	Name	Dir	Type	Description
G19	Nbflowstop_out_SO5+	O	SLVS	Data flow-stop signal to upstream device p
G18	Nbflowstop_out_SO5-	O	SLVS	Data flow-stop signal to upstream device n
E19	Nbflowstop_in+	I	SLVS	Data flow-stop signal p
E18	Nbflowstop_in-	I	SLVS	Data flow-stop signal n
F22	DinN+	I	SLVS	Data from upstream device p
F21	DinN-	I	SLVS	Data from upstream device n
L19	serialOut0+	O	SLVS	Data output p
L18	serialOut0-	O	SLVS	Data output n

Table 2.7: *Pin list for daisy chaining.*

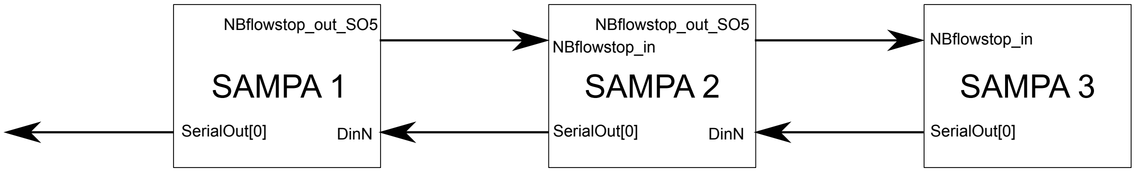


Figure 2.7: *Connection setup for daisy chaining.*

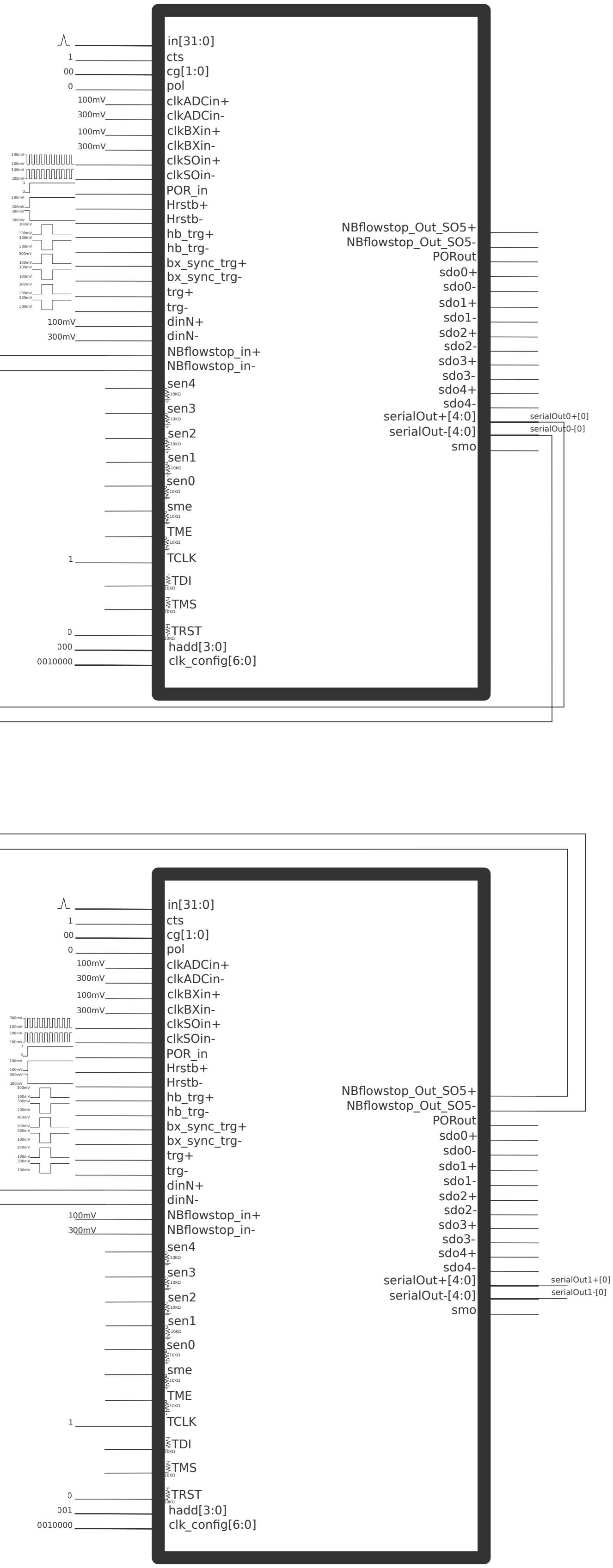


Figure 2.8: *Circuit diagram of a Daisy Chaining mode example.*

2.5 Direct ADC serialization

Direct ADC serialization mode enables serialization of raw ADC samples directly while the rest of the digital circuitry is shut down. The mode only functions properly with the serial clock being 32 times the ADC clock, so the valid settings of clk_config are only those in table 2.8. Running with an external ADC clock is not recommended as the serial link clock and ADC clock need to be synchronous.

clk_config	external clkSO (MHz)	clkADC (MHz)
1000001	160	5
1010011	320	10

Table 2.8: *clk_config settings in DAS mode.*

Pin	Name	Dir	Type	Description
L19	SerialOut+[0]	O	SLVS	Serial link 0 p
L18	SerialOut-[0]	O	SLVS	Serial link 0 n
M22	SerialOut+[1]	O	SLVS	Serial link 1 p
M21	SerialOut-[1]	O	SLVS	Serial link 1 n
K22	SerialOut+[2]	O	SLVS	Serial link 2 p
K21	SerialOut-[2]	O	SLVS	Serial link 2 n
H22	SerialOut+[3]	O	SLVS	Serial link 3 p
H21	SerialOut-[3]	O	SLVS	Serial link 3 n
J19	SerialOut+[4]	O	SLVS	Serial link 4 p
J18	SerialOut-[4]	O	SLVS	Serial link 4 n
G19	NBflowstop_out_SO5+	O	SLVS	Serial link 5 p
G18	NBflowstop_out_SO5-	O	SLVS	Serial link 5 n
C22	sdo+[0]	O	SLVS	Serial link 6 p
C21	sdo-[0]	O	SLVS	Serial link 6 n
Y22	sdo+[1]	O	SLVS	Serial link 7 p
Y21	sdo-[1]	O	SLVS	Serial link 7 n
B22	sdo+[2]	O	SLVS	Serial link 8 p
B21	sdo-[2]	O	SLVS	Serial link 8 n
AB20	sdo+[3]	O	SLVS	Serial link 9 p
AA20	sdo-[3]	O	SLVS	Serial link 9 n
B20	sdo+[4]	O	SLVS	Serial link 10 p
A20	sdo-[4]	O	SLVS	Serial link 10 n
T22	trg+	I	SLVS	Serialization enable p
T21	trg-	I	SLVS	Serialization enable n
W19	bx_sync_trg+	I	SLVS	Serialization mode p
W18	bx_sync_trg-	I	SLVS	Serialization mode n
B14	clk_config[6]	I	Static (1.2 V)	Clock configuration bit for gating

Table 2.9: *Pin list for direct ADC serialization.*

Setting bx_sync_trg to '1' enables the split output mode where ch 0-15 is on link 0-4 and ch 16-31 is on link 5-9 with low nibble first and high nibble second. Setting it to '0' will make it send the 10 bits of ch 0-31 consecutively.

Set trg to '0' disables output, set it to '1' to start the serialization up with the sync sequence first. To avoid the SAMPA chip entering in bypass test mode the hb_trg pin must be set to '0'. The sync pattern is a

32 cycle long pattern that switches between the two complimentary values 0x2B5 and 0x14A. If we define 0x2B5 as 0 and 0x14A as 1 the pattern observed on the serial links [9:0] per serial link clock cycle in the non-split mode would be
01010101001100110101010100110011
and for the split mode it would be slightly different
00110011001100110000111100001111
as we then only send 5 bits per channel per cycle. De-interleaving the sync pattern in the split mode will give the same data as the non-split mode.

The 11th link is the direct output of the ADC clock from the internal clock divider and should be used for monitoring that the SAMPAs are in sync and that no upsets occur in the clock division, which would present itself as a phase shift. Upset detection could be done by counting the number of high cycles and the number of low cycles and then if they differ then do a reset of the SAMPA. The relationship between the clock on the 11th link and the data is not guaranteed so it should not be used as a marker for a specific channel in the stream.

The circuit diagram of figure 2.9 illustrates SAMPA's inputs and outputs signals in DAS mode. In the example, the clkSO is configured for 160MHz, clkBX and clkADC internally generated and the ADC sampling rate is configured for 5 MSPS. The SAMPA hardware address is configured as 0x0. The polarity is configured as negative and the gain and peaking time as 20 mV/fC and 160ns. The output mode is set to normal trough bx_sync_trg in '0'. These inputs can be configured for different values if required.

2.6 Test interfaces

Various test interfaces are available to ease both functional and production testing of the SAMPA.

2.6.1 JTAG

SAMPA supports boundaryscan according to IEEE/ANSI Standard 1149.1-2001[3]. It is intended for testing the connectivity between digital inputs/outputs in the chip and on the board.

Pin	Name	Dir	Type	Function
AA17	TDI	I	LVCMOS (1.2 V)	Test Data In
AB16	TMS	I	LVCMOS (1.2 V)	Test Mode Select
AB17	TCLK	I	LVCMOS (1.2 V)	Test Clock
AA16	TRST	I	LVCMOS (1.2 V)	Test Reset (active low)
AA18	TDO	O	LVCMOS (1.2 V)	Test Data Out

Table 2.10: *JTAG test interface pins.*

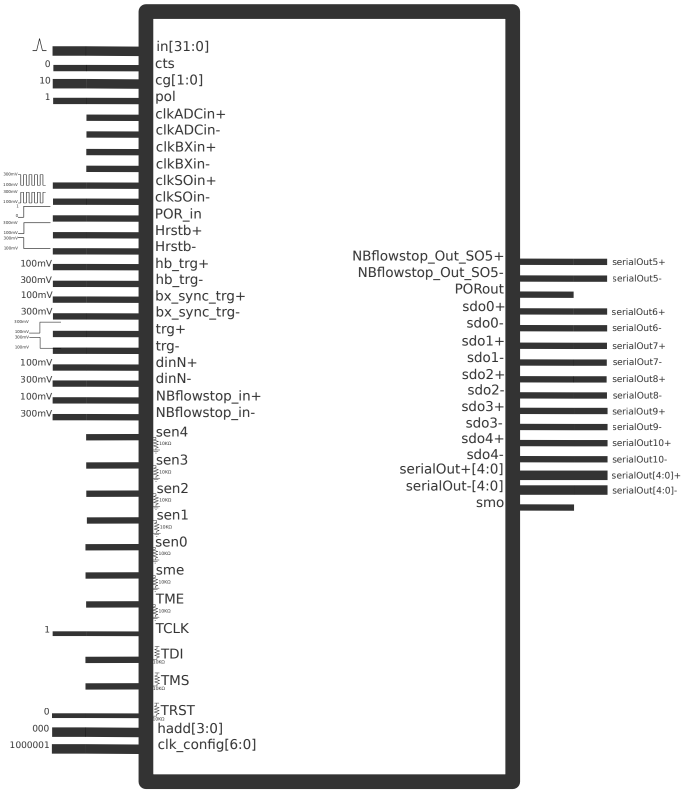


Figure 2.9: *Circuit diagram of a DAS mode example.*

JTAG instructions

The SAMPA has an 3 bit instruction register and supports the mandatory four instructions BYPASS, SAMPLE, PRELOAD and EXTEST.

Instruction	Name	DR Length	Function
0b000	NA	1	Unused (defaults to BYPASS)
0b001	NOP	1	No operation (defaults to BYPASS, reset instruction)
0b010	NA	1	Unused (defaults to BYPASS)
0b011	NA	1	Unused (defaults to BYPASS)
0b100	INTEST	1	Internal test (not implemented, defaults to BYPASS)
0b101	SAMPLE/PRELOAD	44	Sample and Preload, inputs and outputs can be read
0b110	EXTEST	44	External test, outputs can be controlled
0b111	BYPASS	1	Data register bypassed

Table 2.11: *JTAG instruction operation codes.*

BYPASS This instruction puts a 1bit register between TDI and TDO. It is used for shortening the shift path through a component to avoid having to clock the data to the next device through the full data register of the current device.

SAMPLE Reads pin values into the boundary scan register. Used for verifying board connection for input pins. Combined with PRELOAD instruction.

PRELOAD Loads pin values into the boundary scan register that will later be used for EXTEST.

EXTEST Set pins with value loaded with PRELOAD. Used for verifying board connection for output pins.

JTAG boundary register

The boundary register contains the registers that are connected to each input and output pin and it is placed between TDI and TDO in the SAMPLE/PRELOAD or EXTEST mode. In SAMPLE/PRELOAD the values that were present on input pins and output pins can be clocked out, additionally the value that should be loaded in EXTEST mode can be clocked in. Since the SAMPA has a shutdown feature for unused differential outputs it is required to set bit 0 to 1 to enable all output drivers before entering the EXTEST mode.

Bit position	Pin	Function
37	clkADCin	Observe only input
36	clkBXin	Observe only input
35	clkSOin	Observe only input
34	Hrstb	Observe only input
33	bx_sync_trg	Observe only input
32	trg	Observe only input
31	hb_trg	Observe only input
30	hadd[3]	Observe only input
29	hadd[2]	Observe only input
28	hadd[1]	Observe only input
27	hadd[0]	Observe only input
26	clk_config[6]	Observe only input
25	clk_config[5]	Observe only input
24	clk_config[4]	Observe only input
23	clk_config[3]	Observe only input
22	clk_config[2]	Observe only input
21	clk_config[1]	Observe only input
20	clk_config[0]	Observe only input
19	sda_i	Observe only input
18	scl	Observe only input
17	dinN	Observe only input
16	NBflowstop_in	Observe only input
15	sme	Observe only input
14	PORin	Observe only input
13	sda_o	Output (High-Z, 0)
12	NBflowstop_out_SO5	Output with output enable at bit 0
11	serialOut[4]	Output with output enable at bit 0
10	serialOut[3]	Output with output enable at bit 0
9	serialOut[2]	Output with output enable at bit 0
8	serialOut[1]	Output with output enable at bit 0
7	serialOut[0]	Output with output enable at bit 0
6	sdo4	Output with output enable at bit 0
5	sdo3	Output with output enable at bit 0
4	sdo2	Output with output enable at bit 0
3	sdo1	Output with output enable at bit 0
2	sdo0	Output with output enable at bit 0
1	smo	Output with output enable at bit 0
0	-	Control for outputs at bit 1-12 (active high)

Table 2.12: *JTAG boundary register.*

2.6.2 Scanchain

The SAMPA has all flipflops (except memory IPs) wired up in 5 scan chains four of them of 24595 flip flops and one of 24596 flip flops. As some of the flip flops in the design is used with a negative output then there will also be inversions of a signal that travels through the chain. The sen pins enable its corresponding scan chains, while TME enables various override modes internally that is needed for running the scan chain reliably. The sdo pins act as serial output pins in normal operation mode and are therefore differential.

Pin	Name	Dir	Type	Function
B15	TME	I	LVCMOS (1.2 V)	Test mode enable (active high)
A17	sen[0]	I	LVCMOS (1.2 V)	Scan chain enable for chain 0 (active high)
AA14	sen[1]	I	LVCMOS (1.2 V)	Scan chain enable for chain 1 (active high)
A16	sen[2]	I	LVCMOS (1.2 V)	Scan chain enable for chain 2 (active high)
AB12	sen[3]	I	LVCMOS (1.2 V)	Scan chain enable for chain 3 (active high)
A14	sen[4]	I	LVCMOS (1.2 V)	Scan chain enable for chain 4 (active high)
B18	sdi[0]	I	LVCMOS (1.2 V)	Scan chain data in for chain 0
AB13	sdi[1]	I	LVCMOS (1.2 V)	Scan chain data in for chain 1
B17	sdi[2]	I	LVCMOS (1.2 V)	Scan chain data in for chain 2
AA13	sdi[3]	I	LVCMOS (1.2 V)	Scan chain data in for chain 3
B16	sdi[4]	I	LVCMOS (1.2 V)	Scan chain data in for chain 4
C22	sdo+[0]	O	SLVS	Scan chain data output for chain 0 p
C21	sdo-[0]	O	SLVS	Scan chain data output for chain 0 n
Y22	sdo+[1]	O	SLVS	Scan chain data output for chain 1 p
Y21	sdo-[1]	O	SLVS	Scan chain data output for chain 1 n
B22	sdo+[2]	O	SLVS	Scan chain data output for chain 2 p
B21	sdo-[2]	O	SLVS	Scan chain data output for chain 2 n
AB20	sdo+[3]	O	SLVS	Scan chain data output for chain 3 p
AA20	sdo-[3]	O	SLVS	Scan chain data output for chain 3 n
B20	sdo+[4]	O	SLVS	Scan chain data output for chain 4 p
A20	sdo-[4]	O	SLVS	Scan chain data output for chain 4 n

Table 2.13: *Scan chain test interface pins.*

2.6.3 Memory built in tester

The built in memory tester is designed to facilitate testing for deficiencies in the memory IP cores of the SAMPA as these are not covered by the scan chain. The memories tested are the data memory (32x10x6144 at SO/2 clock), the header memories (32x10x256 at ADC clock), the neighbor memory (1x10x2048 at SO/2 clock), the pedestal memories (32x10x1024 at ADC clock) and the pre-sample memories (32x10x192 at ADC clock). The test is based on writing a fixed pattern of 0x2AA to the memory in the first run through, then in the next it will read and check the 0x2AA pattern and write back a pattern of 0x155 (the inverse pattern) and in the next it will check the 0x155 and write 0x2AA and so on.

The testing is enabled by setting the sme pin to 1. There is two outputs available for the test, a latched and a pulsed output. The latched output will stay high until an error is found and will then stay low. This output is intended for production testing. The pulsed output will give one pulse per address that an error is found on. This output can potentially be used for radiation testing. The minimum running time for a test is determined by the largest memory that runs on the slowest clock which is the pedestal memory. The minimum time is 5x1024xADC clock period where 5 is given by the need to do 3 iterations to cover the read-back of both 0x2AA and 0x155 as the first iteration only initializes the memory (W|RW|RW).

Some drawbacks of the pulsed mode should be noted:

- Multiple bit upsets in the same address is only detected as one error.
- The tests are all run in parallel with the test outputs ORed so if two memory testers find an error at the same time they will mask each other.
- The pulse length for a found error is one clock cycle of the clock domain the memory tester is running at.

Pin	Name	Dir	Type	Function
AA12	sme	I	LVCMOS (1.2 V)	Memory test enable (active high)
AA15	smo	O	LVCMOS (1.2 V)	Latched memory test failure output (active low)
L19	serialOut+[0]	O	SLVS	Pulsed memory test failure output p (active high)
L18	serialOut-[0]	O	SLVS	Pulsed memory test failure output n (active high)

Table 2.14: *MBIST test interfaces pins.*

2.7 Hardware address

The hardware/chip address for the device can be set with hadd[3:0] (see table 2.15) between 0 and 15. This is used both as the I2C address for the slow control and to mark the packets transmitted on the serial interface.

Pin	Name	Dir	Type	Description
AA11	hadd[0]	I	Static (1.2 V)	Chip address
AB11	hadd[1]	I	Static (1.2 V)	Chip address
AA10	hadd[2]	I	Static (1.2 V)	Chip address
AB10	hadd[3]	I	Static (1.2 V)	Chip address

Table 2.15: *Hardware address pins*

2.8 Sync and trigger

Three inputs are provided for sync and triggering as listed in table 2.16. They are all sampled on the rising edge of the serial clock and the pulse only needs to be one cycle long. Consecutive pulses need to be spaced by 2 ADC cycles for the event trigger, 2 BX cycles for the bunch crossing trigger and 4 serial cycles for the heartbeat trigger for the trigger to be registered.

hb_trg Upon a pulse, the current BX counter value is captured and a heartbeat packet is crafted.

trg This is the event trigger for use when running in triggered mode. It can also be used to synchronize multiple devices running in continuous readout mode.

bx_sync_trg This is used for synchronizing the internal bunch crossing counter across multiple devices. A trigger will reset the counter.

Pin	Name	Dir	Type	Description
V22	hb_trg+	I	SLVS	Heartbeat trigger p
V21	hb_trg-	I	SLVS	Heartbeat trigger n
T22	trg+	I	SLVS	Event trigger p
T21	trg-	I	SLVS	Event trigger n
W19	bx_sync_trg+	I	SLVS	Bunch crossing sync p
W18	bx_sync_trg-	I	SLVS	Bunch crossing sync n

Table 2.16: *BX sync, event- and heartbeat trigger pins*

2.9 Clock and reset

The pins clock and reset pins are given in table 2.17.

Pin	Name	Dir	Type	Description
U19	clkADCin+	I	SLVS	ADC clock (up to 20 MHz) p
U18	clkADCin-	I	SLVS	ADC clock (up to 20 MHz) n
R19	clkBXin+	I	SLVS	Bunch crossing clock (40 MHz) p
R18	clkBXin-	I	SLVS	Bunch crossing clock (40 MHz) n
N19	clkSOin+	I	SLVS	Serial link clock (up to 320 MHz) p
N18	clkSOin-	I	SLVS	Serial link clock (up to 320 MHz) n
P22	Hrstb+	I	SLVS	Hard reset p
P21	Hrstb-	I	SLVS	Hard reset n
AB14	PORin	I	LVCMOS (1.2 V)	Power on reset input (active low)
AB15	PORout	I	LVCMOS (1.2 V)	Power on reset input (active low)
B10	clk_config[0]	I	Static (1.2 V)	Clock configuration
B11	clk_config[1]	I	Static (1.2 V)	Clock configuration
B12	clk_config[2]	I	Static (1.2 V)	Clock configuration
A10	clk_config[3]	I	Static (1.2 V)	Clock configuration
A11	clk_config[4]	I	Static (1.2 V)	Clock configuration
A12	clk_config[5]	I	Static (1.2 V)	Clock configuration
B14	clk_config[6]	I	Static (1.2 V)	Clock configuration

Table 2.17: *Clock and reset pins*

2.9.1 Clock

There are four clock domains in the SAMPA;

- The ADC clock controls the acquisition of data from the ADC and acts as the main clock for the DSP channels from the input to the ringbuffer.
- Between the ringbuffer and the serial out is the internally generated serial link half clock, which is the serial link clock divided by 2.

- In the final part of the serializer the serial link clock is used.
- A bunch crossing counter is implemented which operates on the LHC bunch crossing clock. Additionally the I2C module operates at this frequency.

The clock divider is implemented as described in figure 2.10. The design of the clock manager has been made so that it is possible to run with a fixed 40 MHz clock output and an ADC speed of 10 or 20 MHz independent of the input serial link speed being 80/160/320 MHz. Either the ADC clock, bunch crossing clock or both can be configured to be supplied externally. An option to derive the ADC clock from the bunch crossing clock input is also available.

Configurations outside of what is listed in table 2.18 is possible with some constraints, but are not guaranteed.

- The ckoutADCSAR clock must always be 8x the ADC clock or more for correct operation of the ADC. 20 MHz ADC with 80 MHz serial link clock is thus not supported.
- Operating the ckoutBX on other than 40 MHz would alter counting rate of the bunch crossing counter and lower the maximum operating speed of the I2C.

Bit number [6] is only for use in the direct ADC serialization mode. Setting it to '1' will disable the clock for the rest of the digital circuitry that is not needed by the direct ADC serialization mode. This has the benefit of saving power and reducing any potentially generated noise. See section §2.5 for more information about the direct ADC serialization mode.

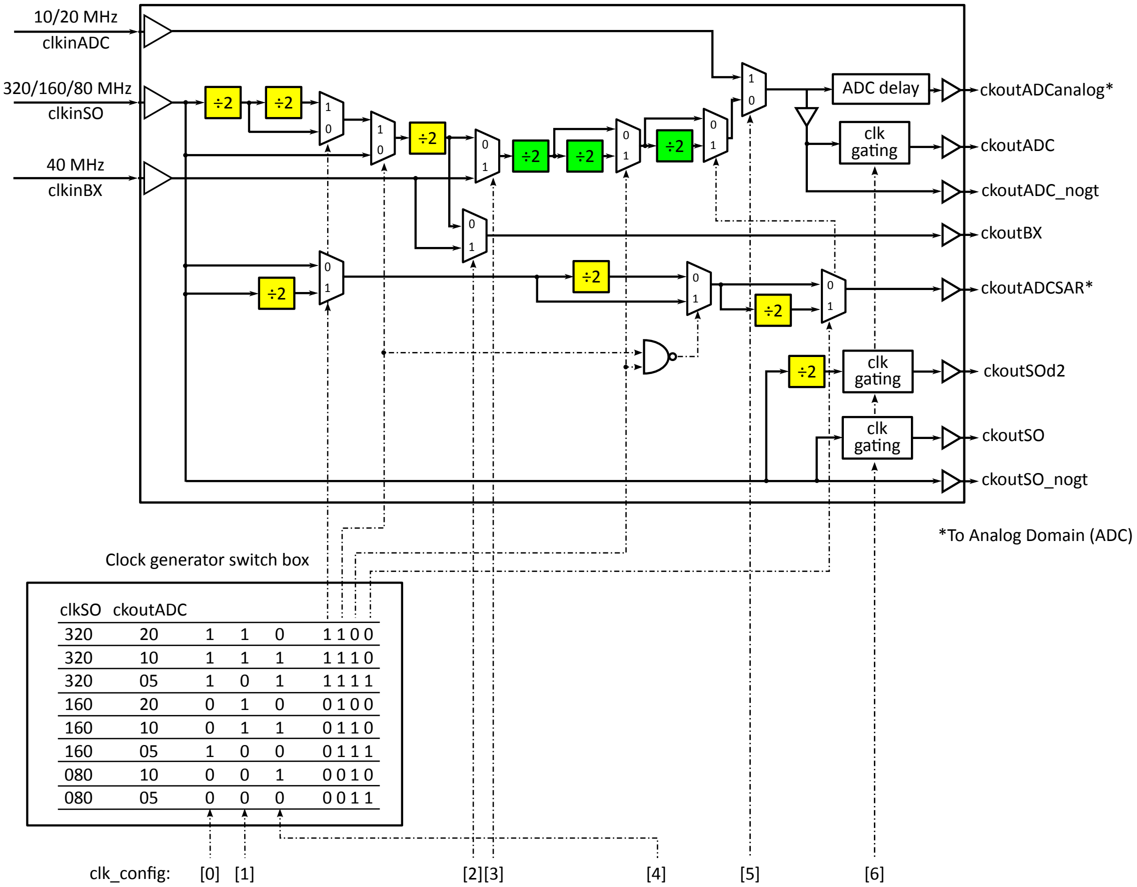
2.9.2 Reset

The reset tree is implemented as shown in figure 2.12. The design uses synchronous reset, but has an asynchronous assert, synchronous de-assert reset synchronizer on the input to avoid metastability problems and guarantee a long enough reset pulse. The hard reset input is an active low differential input and the pulse needs to be longer than one serial link clock cycle. The release from reset takes 2x serial link clock cycle + 2x ADC clock cycle in a worst case condition depending on the phase of the different clocks.

A single ended power on reset signal is also provided (PORin) which can be strapped externally to the chip to the PORout pin to take advantage of an internally generated power on reset circuitry, or it could be connected to an external power on reset circuit.

A software reset is available through the instruction interface. The signal is synchronized to each clock domain before being combined with the hardware reset signal.

A reset signal from either the hard reset pin or the power on reset pin will reset the complete design. A soft reset will reset everything except the configurable registers values (global, channel, channel ordering) and the clock manager.



clkSO	ckoutADC				
320	20	1	1	0	1 1 0 0
320	10	1	1	1	1 1 1 0
320	05	1	0	1	1 1 1 1
160	20	0	1	0	0 1 0 0
160	10	0	1	1	0 1 1 0
160	05	1	0	0	0 1 1 1
080	10	0	0	1	0 0 1 0
080	05	0	0	0	0 0 1 1

Figure 2.10: *Block diagram of the clock generation tree. All flip-flops have asynchronous clear. The flip-flops marked in yellow uses the primary reset signal from figure 2.11, while the green uses the ADC reset signal from the same figure.*

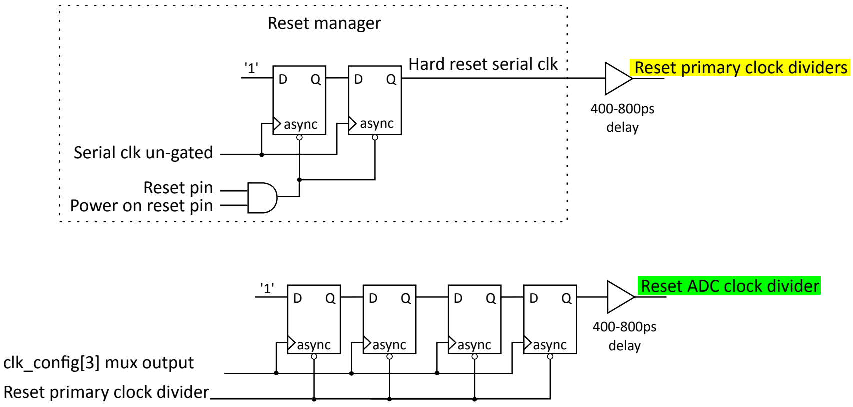


Figure 2.11: *Schematic diagram of the reset for the clock generation tree.*

Inputs			Outputs					clk_config
clkSO	clkBX	clkADC	ckoutSO	ckoutSOd2	ckoutADCSAR	ckoutBX	ckoutADC	[6:0]
All internal								
320	na	na	320	160	160	40 (i)	20 (i so)	0000011
320	na	na	320	160	80	40 (i)	10 (i so)	0010011
320	na	na	320	160	40	40 (i)	5 (i so)	0010001
160	na	na	160	80	160	40 (i)	20 (i so)	0000010
160	na	na	160	80	80	40 (i)	10 (i so)	0010010
160	na	na	160	80	40	40 (i)	5 (i so)	0000001
80	na	na	80	40	80	40 (i)	10 (i so)	0010000
80	na	na	80	40	40	40 (i)	5 (i so)	0000000
BX external								
320	40	na	320	160	160	40 (e)	20 (i so)	0000111
320	40	na	320	160	80	40 (e)	10 (i so)	0010111
320	40	na	320	160	40	40 (e)	5 (i so)	0010101
160	40	na	160	80	160	40 (e)	20 (i so)	0000110
160	40	na	160	80	80	40 (e)	10 (i so)	0010110
160	40	na	160	80	40	40 (e)	5 (i so)	0000101
80	40	na	80	40	80	40 (e)	10 (i so)	0010100
80	40	na	80	40	40	40 (e)	5 (i so)	0000100
320	40	na	320	160	160	40 (e)	20 (i bx)	0001111
320	40	na	320	160	80	40 (e)	10 (i bx)	0011111
320	40	na	320	160	40	40 (e)	5 (i bx)	0011101
160	40	na	160	80	160	40 (e)	20 (i bx)	0001110
160	40	na	160	80	80	40 (e)	10 (i bx)	0011110
160	40	na	160	80	40	40 (e)	5 (i bx)	0001101
80	40	na	80	40	80	40 (e)	10 (i bx)	0011100
80	40	na	80	40	40	40 (e)	5 (i bx)	0001100
ADC external								
320	na	20	320	160	160	40 (i)	20 (e)	0100011
320	na	10	320	160	80	40 (i)	10 (e)	0110011
320	na	10	320	160	40	40 (i)	5 (e)	0110001
160	na	20	160	80	160	40 (i)	20 (e)	0100010
160	na	10	160	80	80	40 (i)	10 (e)	0110010
160	na	10	160	80	40	40 (i)	5 (e)	0100001
80	na	10	80	40	80	40 (i)	10 (e)	0110000
80	na	10	80	40	40	40 (i)	5 (e)	0100000
ADC and BX external								
320	40	20	320	160	160	40 (e)	20 (e)	0100111
320	40	10	320	160	80	40 (e)	10 (e)	0110111
320	40	10	320	160	40	40 (e)	5 (e)	0110101
160	40	20	160	80	160	40 (e)	20 (e)	0100110
160	40	10	160	80	80	40 (e)	10 (e)	0110110
160	40	10	160	80	40	40 (e)	5 (e)	0100101
80	40	10	80	40	80	40 (e)	10 (e)	0110100
80	40	10	80	40	40	40 (e)	5 (e)	0100100

Table 2.18: *Supported clock configurations. If other external ADC frequencies than 10 or 20 MHz is required, then choose the configuration for the closest higher.*

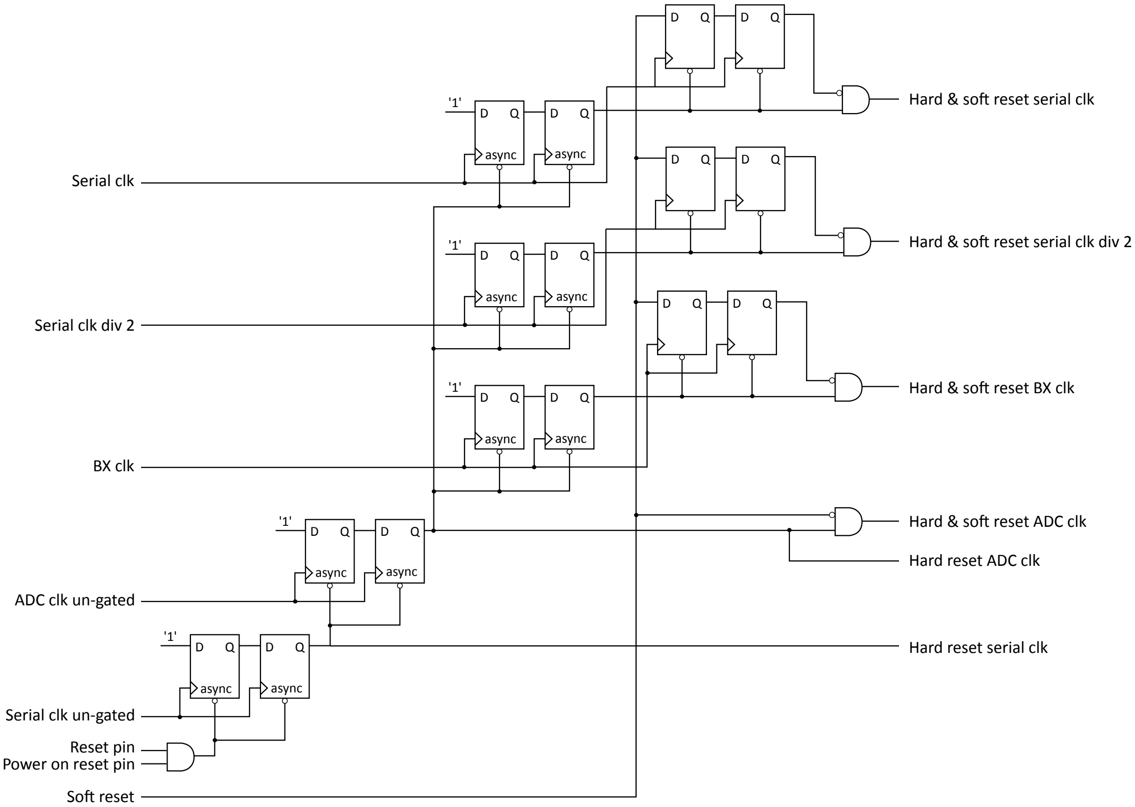


Figure 2.12: *Block diagram of the reset tree.*

Chapter 3

Registers

The SAMPA contains two types of registers; global and channel. The global registers are directly accessible through the I2C interface and contains configurations that concern the main operation of the device or configurations that are common to all channels. The channel registers exist in each channel and contains configurations that can be individually set for each channel. The channel registers are only accessible by going through the global register as further explained in section §3.1.3.

3.1 Global registers

The full list of global registers are listed in table 3.2. Following is the extended description of the registers.

3.1.1 Pin status

The registers in table 3.1 shows the status of input pins on the SAMPA.

HWADD HardWare ADDress is the device address. It is used as the device's I2C address as well as in the header of the data packets leaving the device to indicate to the readout unit where the packets originated.

CLKCONF CLocK CONFiguration is the status of the clk_config pins used for controlling the internal clock division. Refer to table 2.18 for configuration options.

BOUNDARY gives the status of the trigger inputs and the daisy chaining pins.

3.1.2 Event management

The registers in table 3.4 lists the configuration options for setting up events.

Register name		Address	Type	Usable gated	Default	Description	
HWADD	[3:0]	0x00	R	Y	0x00	[3:0]	Chip address (hardware address)
CLKCONF	[6:0]	0x22	R	Y	0x00	[6:0]	Clock configuration pin status
BOUNDARY	[4:0]	0x23	R	Y	0x00		Status of differential input pins
			R	Y	0x00	[0]	NBflowstop_in
			R	Y	0x00	[1]	DinN
			R	Y	0x00	[2]	hb_trg
			R	Y	0x00	[3]	trg
			R	Y	0x00	[4]	bx_sync_trg

Table 3.1: *Global registers*

TRCNTL, TRCNTH TRigger CouNTer is the number of triggers that have been processed by the device. This is regardless if the source of the trigger is the trg pin, soft trigger or from running in continuous mode. The counter can be reset with the TRCLR soft command, see table 3.3, if not it will automatically roll around after 2^{16} triggers. The value has no internal operational purpose.

BXCNTLL, BXCNTLH, BXCNTHL BuncXrossing CouNTer is the current value of the bunch crossing counter. The counter runs on the LHC 40 MHz BX clock and its value is captured at the start of each processing time window and added to the header of the packet for that processing time window. It is also added to the heartbeat packet when one is requested through pulsing of the hb_trg pin. The counter can be reset with the BXCLR soft command, see table 3.3, or by pulsing the bx_sync_trg pin.

PRETRG PRE TRiGger sets the delay of the input data in relation to the trigger signal. A one sample-delay is equivalent to the trigger signal arriving one ADC clock cycle earlier. The configuration is intended to compensate for delays in the trigger signal or to give more time for trigger decision. The maximum configurable delay is 192 samples which corresponds to 19.2 µs when running at an ADC clock of 10 MHz. The delay is inserted at the entry of the digital design, before any data handling or compression happens. The configuration does not alter the length of the time window it merely moves the trigger point in the sample stream earlier in time than when the trigger was received.

TWLENL, TWLENH Time Window LENgth is the number cycles +1 that a time window will last from a trigger start. In continuous mode this will impact the average length of packets.

add image

ACQSTARTL, ACQSTARTH ACQuisition START is the number of cycles from the start of a trigger that will be suppressed in the output data. It is useful for instance in cases where it is know that there always will be an unwanted signal perturbation directly after the trigger which could badly influence the filters. Set the value to 0 to disable.

ACQENDL, ACQENDH ACQuisition END is the number of cycles from the trigger start to when sample acquisition should end. The remaining samples after this point, until the end of the time window, will be suppressed. If the value is set higher or equal to TWLEN then it has no effect. The boolean equation is *acquisition_enable = (TWLEN >= ACQSTART) & (TWLEN <= ACQEND)*.

VACFG[0] Continuous mode enable (VArious ConFiGurations) This enables automatic generation of a new trigger at the end of a time window. A manual soft or pin trigger would prematurely end the current time window and start a new one. Manual pin triggering can be used to align time windows across multiple SAMPAs since they will all start at the same time.

CMD:TRCLR TRigger CLeaR, clears the trigger counter TRCNT.

CMD:SWTRG SoftWare TRiGger provides a single soft trigger, mainly for test purposes.

CMD:BXCLR BuncXrossing CLeaR, clears the bunch crossing counter BXCNT.

CHENx CHannel ENable provides the possibility to shut down specific channels to avoid that they send data. It works by disabling all the filters and suppressing the data readout for the channel in addition to disabling the clock for the ADC.

3.1.3 Channel register access

The register layout has been designed to minimize the amount of writes needed to update the channel registers and pedestal memories of each channel. By using the I2C automatic address increment feature, it is possible to complete all writes needed to update one channel or pedestal address in one continuous I2C command. An additional broadcast bit can be set to write the same data to all channels so that individual access is not needed. When filling the pedestal memory it is possible to set the pedestal memory to increment on each write avoiding the need to update the two registers each time. Table 3.5 lists the register needed to access the channel registers.

Write

1. Set CHRGADD (CHannel ReGister ADDress) to the address of the channel register that you wish to write to.
2. Set the data to write at CHRGWDATH:CHRGWDATL (CHannel ReGister Write DATa).
3. Set CHRGCTL[6] (CHannel ReGister ConTroL) high for write, set CHRGCTL[5] high if you wish to write the same value to all channels (broadcasting), set CHRGCTL[4:0] to the channel number that you wish to write the data to. When broadcasting the channel number is ignored.

Read

1. Set CHRGADD to the address of the channel register that you wish to read from.
2. Set CHRGCTL[6] low for read, set CHRGCTL[4:0] to the channel number that you wish to read from. Broadcast (CHRGCTL[5]) is ignored for reads.
3. The data will appear at CHRGRDATH:CHRGRDATL (CHannel ReGister Read DATa).

Pedestal memory write

1. Make sure the data path configuration for the channel to be written to (DPCFG, see table 3.19) is not using a lookup function f( ), as the lookup function utilizes the pedestal memory and will cause corrupted writes.
2. Set PMADDH:PMADDL (Pedestal Memory ADDress) to the address in the pedestal memory that you wish to write to.
3. Set CHRGADD to 0x10 which is the address for PMDATA (Pedestal Memory DATA) in the channel register.

4. Set the data to write at CHRGWDATH:CHRGWDATL.
5. Set CHRGCTL[7] high to automatically increment the currently set pedestal memory address (increment before write), set CHRGCTL[6] high for write, set CHRGCTL[5] high if you wish to write the same value to all channels (broadcasting), set CHRGCTL[4:0] to the channel number that you wish to write to. When broadcasting the channel number is ignored.

Pedestal memory read

1. Make sure the data path configuration for the channel to be read from (DPCFG, see table 3.19) is not using a lookup function f( ), as the lookup function utilizes the pedestal memory and will cause corrupted reads.
2. Set the address in the pedestal memory that you wish to read from at PMADDH:PMADDL.
3. Put the register address for the channel register PMDATA (0x10) at CHRGADD.
4. Set CHRGCTL[7] high to automatically increment the currently set pedestal memory address (increment before read), set CHRGCTL[6] low for read, set CHRGCTL[4:0] to the channel that you wish to read from. Broadcast (CHRGCTL[5]) is ignored for reads.
5. The data will appear at CHRGRDATH:CHRGRDATL.

3.1.4 Channel ordering

The channel ordering is kept as an ordered list of channels. The order of the channels in the list defines the order that the channels will be read out in. Subsequently it also defines the serial link that a channel will be read out on, see table 3.9. Note that reading and writing to CHORDAT does not necessarily talk to the same register. When you read from CHORDAT, you read directly from the register that CHORDCTL[4:0] is pointing to. Writing to CHORDAT writes to a temporary register and the data in this register is only moved to the address that CHORDCTL[4:0] is pointing to when the write command CHORDCTL[5] is set high. Table 3.6 lists the register needed to access change the channel ordering.

Write

1. Set the channel number that you want to place in the list at CHORDAT (CHannel ORDering DATa).
2. Set CHORDCTL[5] (CHannel ORDering ConTroL) high for write and set CHORDCTL[4:0] to the position in the list that you want to place the channel in.

Read

1. Set CHORDCTL[4:0] to the position in the list that you want to read from.
2. The data will appear at CHORDAT.

3.1.5 ADC configuration

The registers in table 3.7 enables configuration of some aspects of the ADC operation.

ADCDEL (ADC DELay) As shown in figure 2.10 there is available a possibility to delay the clock supplied to the ADCs (ckoutADCanalog) in relation to the clock for the digital part (ckoutADC). This feature could be used to reduce the influence of the digital part switching noise on the ADC sampling. The phase can be changed by ca. 1.5 ns per bit for a total of ca. 94.5 ns depending on process variations, which amounts to almost a full cycle at the minimum designed for ADC clock speed of 10 MHz. If bit [6] is enabled the clock is also inverted, which gives a less process dependent phase shift.

ADCTRIM ADC TRIMing sets the configuration for the Configurable Reference Voltage Source for the analog front end which adjusts the $v450$, $v600$ and $v750$. See section §4.3 for more information.

3.1.6 Serial link configuration

The registers in table 3.10 enables configuration of the serial link driver and receivers.

NBCFG NeighBour ConFiGuration

5:0 As the serial link speed from the neighbouring slave device in a daisy chain runs at the same speed as the serial link clock on the master device, there is no faster clock available which can be used for oversampling the input. So to avoid metastability on the input a delay chain is provided to encompass this. The delay is process dependent, but in most cases only an approximate delay is needed which leaves room for the process variations. The delay is ca. 0.2 ns per bit for a total of ca. 12.5 ns, which amounts to a full cycle at the minimum designed for serial link clock speed of 80 MHz.

7:6 This sets the priority of the neighbour input data in relation to the internal data. It should be set to the number of device following the current device in the chain. If it is the first device in the chain and there are two following, it should be set to two. If it is the second device in the chain and there is one following, it should be set to one, etc. If the value is set to two, then the device will take two packets from the neighbouring link for each packet it takes from any of the internal channels. Only serial link 0 can output data from the neighbouring link.

SOCFG Serial Out ConFiGuration

3:0 This sets the number of active serial link outputs. The links are activated in sequential order, setting the value to 0 disables all links. Disabling a link also disables its corresponding driver to save power. See table 3.9 for which channel would be connected to which link when this setting is changed.

4 The physical connection to the input SLVS links are assumed to be differential transmission lines with a differential impedance of 100 Ω. The receivers have a common built in 100 Ω termination, which can be enabled if required. See section §5.1.7 for further information.

5 The NBflowstop_in pin should be pulled low if not in use. Setting this bit to zero will set the pin to zero internally so that serial outputs will work normally in case the pin externally is stuck at one.

SODRVST (Serial Out DRiVeSTrength) The serial link drivers have a programmable output current. This enables power saving when transmitting over short distances or it provides the possibility to compensate for loss in long cables. The configurations are listed in table 3.8.

CMD:LINKSYNC This command forces sending of a sync packet on all active links after the current transmission on the link has completed.

3.1.7 Data compression

Several methods of compressing the data stream before transmission is available as listed in table 3.11, see section §1.5 for further details on the operation of each method. If no method is selected it defaults to zero suppression. In case more than one method is selected, the priority is raw > huffman > cluster sum > zero suppression.

VACFG[1] Raw mode This enables sending of raw data samples. It is equivalent to running zero suppression with a threshold of 0, but it removes the two extra control words at the beginning of the data payload (time stamp and cluster size).

VACFG[2] Cluster sum This enables the cluster sum compression method.

VACFG[3] Huffman This enables Huffman compression.

VACFG[4] Send empty packets In case there are no pulses above the zero suppression threshold for a complete time window, which means there is no data to be transmitted in the payload of the packet, there will still be sent a header with an empty payload. If this is unwanted due to bandwidth restrictions or other causes, then this bit can be set to 0 to prevent them from being sent.

3.1.8 Power saving features

Some power saving features have been implemented as listed in table 3.12. As power saving features sometimes might have unintended consequences which can render the design unusable, it is possible to disable them.

VACFG[5] Power save enable This bit makes sure that the data pins of the pedestal memory are not toggling when the device is in a mode where the pedestal memory is not in use. When the pins are not toggling there will be no activity in the memory and it will consume less power.

VACFG[6] I2C gate The state machine of the I2C module can be automatically powered down when there is no activity on the I2C lines to save power. This bit enables this feature.

VACFG[7] Neighbour block power down The neighbour block, including its ringbuffer and receiver circuit, can be disabled automatically to save power when the number of neighbours (NBCFG[7:6]) is set to 0, this feature enables this functionality.

3.1.9 Test functionality

To aid in debugging and test of the chip, some extended functionality has been added and is available from the registers listed in table 3.14.

ERRORS The packet headers are equipped with a Hamming error correction code which enables correction of one error and detection of two errors in the header. The Hamming code is added to the header before it is written to the ringbuffer memory and it is checked again when it is read out of the ringbuffer, before it is sent to the serial links. The ERRORS register contains a counter for the number of errors found that could be corrected, and the number of errors that were not correctable and where the packet was dropped. The counters collect errors from all channel ringbuffer outputs, in addition to the neighbour ringbuffer output and the neighbour input. The ERRORS register can be cleared by sending the ERCLR (0x3) command, see table 3.3.

BYPASS The BYPASS register controls a multiplexer situated directly at the serialOut[0] output. See table 3.13 for specifics.

0x1-0x3 Feed-throughs from trigger inputs can be used for determining round-trip delay of the system in a production environment. It can also be used for qualifying the SLVS driver/receiver pair.

0x4 Feed-through from neighbour input DinN can be used for bypassing the device in the chain. Also feeds NBflowstop_in to NBflowstop_out for full bypass in daisy chain.

0x5 Feed-through from delayed neighbour input after the neighbour input has passed through the delay specified by NBCFG[5:0]. This can be used for qualifying the neighbour input delay chain. Also feeds NBflowstop_in to NBflowstop_out for full bypass in daisy chain.

0x6 LFSR generator is intended for use in qualifying the SLVS driver. When selected it enables a 31 bit LFSR generator which feeds back the XNOR of output 31 and 28. It runs at the serial link clock.

0x7 The ADC test serializer selects the channel configured in the SERCHSEL register and serializes it at the serial link half clock with a preamble of b010011.

0x8-0xC Internal clock output can be used for qualifying the quality of the clocks arriving from the clock divider. As there is a configurable delay chain (ADCDEL) between the digital ADC clock and analog ADC clock it is possible to qualify the delay chain by providing an external ADC clock and outputting the analog ADC clock through this BYPASS mode.

0xD A ring oscillator has been implemented to determine process variations and thermal/voltage effects. The oscillation is only operative after the RINGOSCTST (table 3.3) command has been sent and then only for 256 ADC clock cycles or 512 oscillator cycles, whichever is the shortest. It should oscillate at about 100.9 MHz for worst case, 220 MHz for best case and 160.9 MHz for typical case.

SERCHSEL (SERializer CHannel SELect) When BYPASS is set to 0x7, then this selects the ADC channel used for the test serializer.

RINGCNT (RINGCouNTer) In a non-test environment it is possible to determine the process speed of a device by comparing the ring oscillator clock against the ADC clock. The ring oscillator clock is divided 16 times and is used to run a counter which is compared to a counter running at the ADC clock speed. The RINGCNT value is a signed byte and the oscillation period can be calculated as $OSC = \frac{255 \cdot ADC}{(255 - RINGCNT) \cdot 16}$ where ADC is the ADC clock period.

Register name		Address	Type	Usable gated	Default	Description	
HWADD	[3:0]	0x00	R	Y	0x00	[3:0]	Chip address (hardware address)
TRCNTL	[7:0]	0x01	R	N	0x00	[7:0]	Trigger count, lower byte
TRCNTH	[7:0]	0x02	R	N	0x00	[7:0]	Trigger count, upper byte
BXCNTLL	[7:0]	0x03	R	N	0x00	[7:0]	Bunch crossing count, lower byte
BXCNTLH	[7:0]	0x04	R	N	0x00	[7:0]	Bunch crossing count, mid byte
BXCNTHL	[3:0]	0x05	R	N	0x00	[3:0]	Bunch crossing count, upper byte
PRETRG	[7:0]	0x06	RW	N	0x00	[7:0]	Number of pre-samples (Pre-trigger delay), max 192
TWLENL	[7:0]	0x07	RW	N	0xE7	[7:0]	Number of cycles for time window +1, lower byte
TWLENH	[1:0]	0x08	RW	N	0x03	[1:0]	Number of cycles for time window +1, upper byte
ACQSTARTL	[7:0]	0x09	RW	N	0x00	[7:0]	Number of cycles to wait before acquisition starts, lower byte
ACQSTARTH	[1:0]	0x0A	RW	N	0x00	[1:0]	Number of cycles to wait before acquisition starts, upper byte
ACQENDL	[7:0]	0x0B	RW	N	0xFF	[7:0]	Number of cycles elapsed from trigger to acquisition end +1, lower byte
ACQENDH	[1:0]	0x0C	RW	N	0x03	[1:0]	Number of cycles elapsed from trigger to acquisition end +1, upper byte
VACFG	[7:0]	0x0D	RW	Y	0x30		Various configuration settings
			RW	N	0x00	[0]	Continuous mode enabled
			RW	N	0x00	[1]	Raw data enable
			RW	N	0x00	[2]	Cluster sum enable
			RW	N	0x00	[3]	Huffman enable
			RW	N	0x01	[4]	Enable header generation for empty channels
			RW	N	0x01	[5]	Power save enable
			RW	Y	0x00	[6]	Enable automatic clock gating on I2C block
			RW	N	0x00	[7]	Enable clock gating on neighbour block when number of neighbour is 0
CMD	[2:0]	0x0E	RW	Y	0x00	[2:0]	Commands, see table 3.3
NBCFG	[7:0]	0x0F	RW	N	0x40		Neighbor configuration settings
			RW	N	0x00	[5:0]	Neighbor input delay, ca. 0.2 ns per bit for a total of ca. 12.5 ns
			RW	N	0x01	[7:6]	Number of neighbors
ADCDEL	[6:0]	0x10	RW	Y	0x00		ADC sampling clock delay
			RW	Y	0x00	[5:0]	ADC sampling clock delay, ca. 1.5 ns per bit for a total of ca. 94.5 ns
			RW	Y	0x00	[6]	Invert ADC sampling clock
ADCTRIM	[2:0]	0x11	RW	Y	0x04	[2:0]	Voltage reference trimming
SOCFG	[5:0]	0x12	RW	Y	0x34		Serial link configuration
			RW	N	0x04	[3:0]	Number of serial out, 0-11
			RW	Y	0x01	[4]	Disable internal termination of input differential links
			RW	Y	0x01	[5]	Enable NBflowstop_in pin
SODRVST	[7:0]	0x13	RW	Y	0x55		Serial link drive strength configuration, see table 3.8
			RW	Y	0x01	[1:0]	Drive strength of serial out 4-0
			RW	Y	0x01	[3:2]	Drive strength of neighbor flow stop out/serial out 5
			RW	Y	0x01	[5:4]	Drive strength of serial out 6,8,10
			RW	Y	0x01	[7:6]	Drive strength of serial out 7,9
ERRORS	[7:0]	0x14	R	N	0x00		Errors accumulated
			R	N	0x00	[4:0]	Correctable header hamming errors
			R	N	0x00	[7:5]	Uncorrectable header hamming errors
PMADDL	[7:0]	0x15	RW	N	0x00	[7:0]	Pedestal memory address, lower byte
PMADDH	[1:0]	0x16	RW	N	0x00	[1:0]	Pedestal memory address, upper byte
CHRGADD	[4:0]	0x17	RW	N	0x00	[4:0]	Channel register address
CHRGWDATL	[7:0]	0x18	RW	N	0x00	[7:0]	Channel register write data, lower byte
CHRGWDATH	[4:0]	0x19	RW	N	0x00	[4:0]	Channel register write data, upper byte
CHRGCTL	[7:0]	0x1A	RW	N	0x00		Channel register control
			RW	N	0x00	[4:0]	Channel number
			RW	N	0x00	[5]	Broadcast to all channels (channel number ignored)
			RW	N	0x00	[6]	Write, not read from register address (returns to read after write)
			RW	N	0x00	[7]	Increment PMADD (returns automatically to zero)
CHRGRDATL	[7:0]	0x1B	R	N	0x00	[7:0]	Channel register read data, lower byte
CHRGRDATH	[4:0]	0x1C	R	N	0x00	[4:0]	Channel register read data, upper byte
CHORDAT	[4:0]	0x1D	RW	N	0x00	[4:0]	Channel readout order data
CHORDCTL	[5:0]	0x1E	RW	N	0x00		Channel readout order control
			RW	N	0x00	[4:0]	Position in order
			RW	N	0x00	[5]	Write enable
BYPASS	[3:0]	0x1F	RW	Y	0x00	[3:0]	Bypass inputs to serial 0, see table 3.13
SERCHSEL	[4:0]	0x20	RW	N	0x00	[4:0]	Channel select for ADC test serializer mode in bypass
RINGCNT	[7:0]	0x21	R	Y	0x00	[7:0]	Ring oscillator counter difference from reference ADC clock
CLKCONF	[6:0]	0x22	R	Y	0x00	[6:0]	Clock configuration pin status
BOUNDARY	[4:0]	0x23	R	Y	0x00		Status of differential input pins
			R	Y	0x00	[0]	NBflowstop_in
			R	Y	0x00	[1]	DinN
			R	Y	0x00	[2]	hb_trg
			R	Y	0x00	[3]	trg
			R	Y	0x00	[4]	bx_sync_trg
CHEN0	[7:0]	0x24	RW	Y	0xFF	[7:0]	Channel enable 7-0
CHEN1	[7:0]	0x25	RW	Y	0xFF	[7:0]	Channel enable 15-8
CHEN2	[7:0]	0x26	RW	Y	0xFF	[7:0]	Channel enable 23-16
CHEN3	[7:0]	0x27	RW	Y	0xFF	[7:0]	Channel enable 31-24

Table 3.2: *Global registers. Usable gated indicates which registers have any useful function in the Direct ADC Serialization mode.*

Command name	CMD[2:0]	Description
NOP	0x0	No operation
SWTRG	0x1	Software trigger
TRCLR	0x2	Clear trigger counter
ERCLR	0x3	Clear errors
BXCLR	0x4	Clear bunch crossing counter
SOFTRST	0x5	Software reset
LNKSYNC	0x6	Generates sync packet on serial links
RINGOSCTST	0x7	Run ring oscillator test

Table 3.3: *Command register, register returns to 0 after command is executed*

Register name		Address	Type	Usable gated	Default	Description	
TRCNTL	[7:0]	0x01	R	N	0x00	[7:0]	Trigger count, lower byte
TRCNTH	[7:0]	0x02	R	N	0x00	[7:0]	Trigger count, upper byte
BXCNTLL	[7:0]	0x03	R	N	0x00	[7:0]	Bunch crossing count, lower byte
BXCNTLH	[7:0]	0x04	R	N	0x00	[7:0]	Bunch crossing count, mid byte
BXCNTHL	[3:0]	0x05	R	N	0x00	[3:0]	Bunch crossing count, upper byte
PRETRG	[7:0]	0x06	RW	N	0x00	[7:0]	Number of pre-samples (Pre-trigger delay), max 192
TWLENL	[7:0]	0x07	RW	N	0xE7	[7:0]	Number of cycles for time window +1, lower byte
TWLENH	[1:0]	0x08	RW	N	0x03	[1:0]	Number of cycles for time window +1, upper byte
ACQSTARTL	[7:0]	0x09	RW	N	0x00	[7:0]	Number of cycles to wait before acquisition starts, lower byte
ACQSTARTH	[1:0]	0x0A	RW	N	0x00	[1:0]	Number of cycles to wait before acquisition starts, upper byte
ACQENDL	[7:0]	0x0B	RW	N	0xFF	[7:0]	Number of cycles elapsed from trigger to acquisition end +1, lower byte
ACQENDH	[1:0]	0x0C	RW	N	0x03	[1:0]	Number of cycles elapsed from trigger to acquisition end +1, upper byte
VACFG	[7:0]	0x0D	RW	Y	0x31		Various configuration settings
			RW	N	0x01	[0]	Continuous mode enabled
CMD	[2:0]	0x0E	RW	Y	0x00	[2:0]	Commands, see table 3.3
CHEN0	[7:0]	0x24	RW	Y	0xFF	[7:0]	Channel enable 7-0
CHEN1	[7:0]	0x25	RW	Y	0xFF	[7:0]	Channel enable 15-8
CHEN2	[7:0]	0x26	RW	Y	0xFF	[7:0]	Channel enable 23-16
CHEN3	[7:0]	0x27	RW	Y	0xFF	[7:0]	Channel enable 31-24

Table 3.4: *Event management registers*

Register name		Address	Type	Usable gated	Default	Description	
PMADDL	[7:0]	0x15	RW	N	0x00	[7:0]	Pedestal memory address, lower byte
PMADDH	[1:0]	0x16	RW	N	0x00	[1:0]	Pedestal memory address, upper byte
CHRGADD	[4:0]	0x17	RW	N	0x00	[4:0]	Channel register address
CHRGWDATL	[7:0]	0x18	RW	N	0x00	[7:0]	Channel register write data, lower byte
CHRGWDATH	[4:0]	0x19	RW	N	0x00	[4:0]	Channel register write data, upper byte
CHRGCTL	[7:0]	0x1A	RW	N	0x00		Channel register control
			RW	N	0x00	[4:0]	Channel number
			RW	N	0x00	[5]	Broadcast to all channels (channel number ignored)
			RW	N	0x00	[6]	Write, not read from register address (returns to read after write)
			RW	N	0x00	[7]	Increment PMADD (returns automatically to zero)
CHRGRDATL	[7:0]	0x1B	R	N	0x00	[7:0]	Channel register read data, lower byte
CHRGRDATH	[4:0]	0x1C	R	N	0x00	[4:0]	Channel register read data, upper byte

Table 3.5: *Channel access registers*

Register name		Address	Type	Usable gated	Default	Description	
CHORDAT	[4:0]	0x1D	RW	N	0x00	[4:0]	Channel readout order data
CHORDCTL	[5:0]	0x1E	RW	N	0x00		Channel readout order control
			RW	N	0x00	[4:0]	Position in order
			RW	N	0x00	[5]	Write enable

Table 3.6: *Channel order access registers*

Register name		Address	Type	Usable gated	Default	Description	
ADCDEL	[6:0]	0x10	RW	Y	0x00	[5:0]	ADC sampling clock delay, ca. 1.5 ns per bit for a total of ca. 94.5 ns
			RW	Y	0x00	[6]	Invert ADC sampling clock
ADCTRIM	[2:0]	0x11	RW	Y	0x04	[2:0]	Voltage reference trimming

Table 3.7: *ADC configuration registers*

Drive strength [1:0]	Iout mean (mA)	Vdiff (mV)	Description
00	1.95	438	Normal mode
01	1.61	348	Low power mode
10	2.87	610	High drive strength mode

Table 3.8: *Serial link drive strength configuration*

Channel	Number of links enabled										
	1	2	3	4	5	6	7	8	9	10	11
0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	1
4	0	0	0	0	0	0	0	1	1	1	1
5	0	0	0	0	0	0	1	1	1	1	1
6	0	0	0	0	0	1	1	1	1	1	2
7	0	0	0	0	1	1	1	1	1	1	2
8	0	0	0	1	1	1	1	2	2	2	2
9	0	0	0	1	1	1	1	2	2	2	3
10	0	0	0	1	1	1	2	2	2	2	3
11	0	0	1	1	1	1	2	2	2	3	3
12	0	0	1	1	1	2	2	3	3	3	4
13	0	0	1	1	1	2	2	3	3	3	4
14	0	0	1	1	2	2	2	3	3	4	4
15	0	0	1	1	2	2	3	3	3	4	5
16	0	1	1	2	2	2	3	4	4	4	5
17	0	1	1	2	2	3	3	4	4	5	5
18	0	1	1	2	2	3	3	4	4	5	6
19	0	1	1	2	2	3	3	4	4	5	6
20	0	1	1	2	3	3	4	5	5	6	6
21	0	1	1	2	3	3	4	5	5	6	7
22	0	1	2	2	3	4	4	5	5	6	7
23	0	1	2	2	3	4	4	5	6	7	7
24	0	1	2	3	3	4	5	6	6	7	8
25	0	1	2	3	3	4	5	6	6	7	8
26	0	1	2	3	4	4	5	6	7	8	8
27	0	1	2	3	4	5	5	6	7	8	9
28	0	1	2	3	4	5	6	7	7	8	9
29	0	1	2	3	4	5	6	7	8	9	9
30	0	1	2	3	4	5	6	7	8	9	10
31	0	1	2	3	4	5	6	7	8	9	10

Table 3.9: *This table lists which channel is connected to which serial link when the specified number of links is selected. In case the channel ordering is in use, then the list position can be substituted for the channel.*

Register name	Address	Type	Usable gated	Default	Description	
NBCFG [7:0]	0x0F	RW	N	0x40		Neighbor configuration settings
		RW	N	0x00	[5:0]	Neighbor input delay, ca. 0.2 ns per bit for a total of ca. 12.5 ns
		RW	N	0x01	[7:6]	Number of neighbors
SOCFG [4:0]	0x12	RW	Y	0x34		Serial link configuration
		RW	N	0x04	[3:0]	Number of serial out, 0-11
		RW	Y	0x01	[4]	Disable internal termination of input differential links
		RW	Y	0x01	[5]	Enable NBflowstop_in pin
SODRVST [7:0]	0x13	RW	Y	0x55		Serial link drive strength configuration, see table 3.8
		RW	Y	0x01	[1:0]	Drive strength of serial out 4-0
		RW	Y	0x01	[3:2]	Drive strength of neighbor flow stop out/serial out 5
		RW	Y	0x01	[5:4]	Drive strength of serial out 6,8,10
		RW	Y	0x01	[7:6]	Drive strength of serial out 7,9

Table 3.10: *Serial link driver and receiver registers*

Register name	Address	Type	Usable gated	Default	Description	
VACFG [7:0]	0x0D	RW	Y	0x31		Various configuration settings
		RW	N	0x00	[1]	Raw data enable
		RW	N	0x00	[2]	Cluster sum enable
		RW	N	0x00	[3]	Huffman enable
		RW	N	0x01	[4]	Enable header generation for empty channels

Table 3.11: *Data compression registers*

Register name	Address	Type	Usable gated	Default	Description
VACFG [7:0]	0x0D	RW	Y	0x31	Various configuration settings
		RW	N	0x01	[5] Power save enable
		RW	Y	0x00	[6] Enable automatic clock gating on I2C block
		RW	N	0x00	[7] Enable clock gating on neighbour block when number of neighbour is 0

Table 3.12: *Power saving feature registers*

BYPASS[3:0]	Description
0x0	Serializer 0
0x1	Feed-through from bx_sync input
0x2	Feed-through from trg input
0x3	Feed-through from hb_trg input
0x4	Feed-through from neighbour input (dinN)
0x5	Feed-through from delayed neighbour input (dinN_del)
0x6	Output of 31 bits lfsr generator
0x7	Output of ADC test serializer
0x8	Internal ADC clock for digital part 10/20MHz
0x9	Internal serial out clock divided by 2
0xA	Internal bunch crossing clock 40MHz
0xB	Internal ADC clock for analog part 10/20MHz
0xC	Internal SAR ADC statemachine clock for analog part 80/160MHz
0xD	Clock from ring oscillator (only when triggered to run) 100-220 MHz

Table 3.13: *Bypass signals for serial out 0 output*

Register name	Address	Type	Usable gated	Default	Description
ERRORS [7:0]	0x14	R	N	0x00	Errors accumulated
		R	N	0x00	[4:0] Correctable header hamming errors
		R	N	0x00	[7:5] Uncorrectable header hamming errors
BYPASS [3:0]	0x1F	RW	Y	0x00	[3:0] Bypass inputs to serial 0, see table 3.13
SERCHSEL [4:0]	0x20	RW	N	0x00	[4:0] Channel select for ADC test serializer mode in bypass
RINGCNT [7:0]	0x21	R	Y	0x00	[7:0] Ring oscillator counter difference from reference ADC clock

Table 3.14: *Test functionality registers*

3.2 Channel specific registers

The registers listed in table 3.15 are registers specific to each channel. By using a broadcast command when writing, all channels can be written at the same time. Refer to section §3.1.3 for information on how to access the channel registers.

Register name		Address	Type	Default	Description	
K1	[12:0]	0x00	RW	0x000	[12:0]	First pole of the TCFU
K2	[12:0]	0x01	RW	0x000	[12:0]	Second pole of the TCFU
K3	[12:0]	0x02	RW	0x000	[12:0]	Third pole of the TCFU
K4	[12:0]	0x03	RW	0x000	[12:0]	Fourth pole of the TCFU
L1	[12:0]	0x04	RW	0x000	[12:0]	First zero of the TCFU
L2	[12:0]	0x05	RW	0x000	[12:0]	Second zero of the TCFU
L3	[12:0]	0x06	RW	0x000	[12:0]	Third zero of the TCFU
L4	[12:0]	0x07	RW	0x000	[12:0]	Fourth zero of the TCFU
L30	[12:0]	0x08	RW	0x000	[12:0]	TCFU IIR SOS first zero(L3) gain
ZSTHR	[11:0]	0x09	RW	0x000	[11:0]	Zero suppression threshold, 2 bit precision
ZSOFF	[12:0]	0x0A	RW	0x000	[12:0]	Offset added before truncation, 2's compliment, 2 bit precision
ZSCFG	[8:0]	0x0B	RW	0x000		Zero suppression configuration
			RW	0x000	[1:0]	Glitch filter, minimum accepted pulse, all, >1, >2, >2
			RW	0x000	[4:2]	Post-samples
			RW	0x000	[6:5]	Pre-samples
			RW	0x000	[7]	Change position of BC3 in pipeline (BC3 after BC2)
			RW	0x000	[8]	Enable Raw data output of ZSU
FPD	[12:0]	0x0C	RW	0x000	[12:0]	BC1 Fixed pedestal (offset subtracted), 2's compliment, 2 bit precision
VPD	[12:0]	0x0D	R	0x000	[12:0]	BC1 variable pedestal, 2's compliment
BC2BSL	[12:0]	0x0E	R	0x000	[12:0]	BC2 Computed Baseline, 2's compliment
BC3BSL	[12:0]	0x0F	R	0x000	[12:0]	BC3 Computed Baseline, 2's compliment
PMDATA	[9:0]	0x10	RW	0x000	[9:0]	Data to be stored or read from the pedestal memory
BC2LTHRREL	[9:0]	0x11	RW	0x003	[9:0]	BC2 lower relative threshold
BC2HTHRREL	[9:0]	0x12	RW	0x003	[9:0]	BC2 higher relative threshold
BC2LTHRBSL	[10:0]	0x13	RW	0x400	[10:0]	BC2 lower saturation level for baseline
BC2HTHRBSL	[10:0]	0x14	RW	0x3FF	[10:0]	BC2 higher saturation level for baseline
BC2CFG	[10:0]	0x15	RW	0x000		BC2 configuration
			RW	0x000	[1:0]	Number of taps in moving average filter
			RW	0x000	[3:2]	BC2 pre-samples
			RW	0x000	[7:4]	BC2 post-samples
			RW	0x000	[8]	BC2 glitch removal
			RW	0x000	[10:9]	Auto reset configuration
BC2RSTVAL	[7:0]	0x16	RW	0x032	[7:0]	Reset value for maf baseline when auto reset is enabled
BC2RSTCNT	[7:0]	0x17	RW	0x0FF	[7:0]	Number of samples outside of thresholds before resetting maf filter (divided by 4)
DPCFG	[11:0]	0x18	RW	0x000		Data path configuration
			RW	0x000	[3:0]	BC1 mode, see table 3.19
			RW	0x000	[4]	BC1 data input polarity
			RW	0x000	[5]	BC1 pedestal memory polarity
			RW	0x000	[6]	BC1 pedestal memory record from input
			RW	0x000	[7]	TCFU enabled
			RW	0x000	[8]	BC2 moving average filter enable
			RW	0x000	[9]	BC3 filter enable
			RW	0x000	[10]	TCFU SOS Architecture enable
			RW	0x000	[11]	TCFU signed Pole/Zero enable
BC1THRL	[10:0]	0x19	RW	0x7FD	[10:0]	Lower threshold of variable pedestal filter, 2's compliment
BC1THRH	[10:0]	0x1A	RW	0x003	[10:0]	Higher threshold of variable pedestal filter, 2's compliment
BC1CFG	[9:0]	0x1B	RW	0x114		BC1 configuration
			RW	0x004	[3:0]	Number of taps in variable pedestal filter
			RW	0x001	[4]	Define open threshold time of 31(high)/15(low) samples after (auto)reset
			RW	0x000	[5]	Force enable IIR also inside time window
			RW	0x000	[6]	High if BC1THR should be considered absolute, else relative
			RW	0x002	[8:7]	Shift output data of pedestal memory
			RW	0x000	[9]	BC1 negative clipping enabled
BC1RSTCNT	[7:0]	0x1C	RW	0x000	[7:0]	Number of samples outside of thresholds before resetting vpd filter (divided by 4)
			RW	0x000		0 disables the auto reset
BC3SLD	[7:0]	0x1D	RW	0x020	[7:0]	Rate of the BC3 baseline down counter
BC3SLU	[7:0]	0x1E	RW	0x010	[7:0]	Rate of the BC3 baseline up counter

Table 3.15: *Channel specific registers*

3.2.1 Data path configuration

All the filters can be bypassed so only the filters that suits the detectors needs can be enabled.

ZSCFG Zero-Suppression ConFiGuration

7 Enabling this moves the BC3 filter before the BC2 filter. The BC2 filter can then help reduce some of the noise in the baseline generated by the BC3 filter. The BC2 filter will then also not be affected by large changes in the baseline which can make it get stuck outside its thresholds so it should be safe to use.

DPCFG (Data-Path ConFiGuration)

3:0 This switches the data path in the Baseline Correction 1 filter between the different operation modes. See table 3.19 for the different modes and section §4.7 for the description of the modes.

6 This enables writing the data from the input directly to the pedestal memory. Addressing happens from 0 to TWLEN from trigger start.

7 This enables the Tail Cancellation filter correction.

8 This enables the Baseline Correction 2 moving average based filter correction.

9 This enables the Baseline Correction 3 slope based filter correction.

Register name		Address	Type	Default	Description	
ZSCFG	[8:0]	0x0B	RW	0x000		Zero suppression configuration
			RW	0x000	[7]	Change position of BC3 in pipeline (BC3 after BC2)
DPCFG	[11:0]	0x18	RW	0x000		Data path configuration
			RW	0x000	[3:0]	BC1 mode, see table 3.19
			RW	0x000	[6]	BC1 pedestal memory record from input
			RW	0x000	[7]	TCFU enabled
			RW	0x000	[8]	BC2 moving average filter enable
			RW	0x000	[9]	BC3 filter enable

Table 3.16: *Data path configuration registers*

3.2.2 Digital shaper

K1:K4 Poles of the IIR filter

L1:L4 Zeros of the IIR filter

L30 **bruno fix**

DPCFG (Data-Path ConFiGuration)

10 **bruno fix**

11 **bruno fix**

Register name	Address	Type	Default	Description
K1 [12:0]	0x00	RW	0x000	[12:0] First pole of the TCFU
K2 [12:0]	0x01	RW	0x000	[12:0] Second pole of the TCFU
K3 [12:0]	0x02	RW	0x000	[12:0] Third pole of the TCFU
K4 [12:0]	0x03	RW	0x000	[12:0] Fourth pole of the TCFU
L1 [12:0]	0x04	RW	0x000	[12:0] First zero of the TCFU
L2 [12:0]	0x05	RW	0x000	[12:0] Second zero of the TCFU
L3 [12:0]	0x06	RW	0x000	[12:0] Third zero of the TCFU
L4 [12:0]	0x07	RW	0x000	[12:0] Fourth zero of the TCFU
L30 [12:0]	0x08	RW	0x000	[12:0] TCFU IIR SOS first zero(L3) gain
DPCFG [11:0]	0x18	RW	0x000	Data path configuration
		RW	0x000	[10] TCFU SOS Architecture enable
		RW	0x000	[11] TCFU signed Pole/Zero enable

Table 3.17: *Digital shaper registers*

3.2.3 BC1

FPD (Fixed PeDestal) Each channel has a baseline value which usually is desirable to remove so that the signal is sitting at zero instead of at an offset from zero. The value is two's complement with two extra bits of precision.

VPD (Variable PeDestal) This is the current baseline calculated by the variable pedestal filter.

DPCFG (Data-Path ConFiGuration)

4 Enabling this will negate the input to the Baseline Correction 1 filter.

5 Enabling this will negate the output value of the pedestal memory.

BC1THRL (Baseline Correction 1 THReshold Low) This sets the threshold below the current value of the VPD for when the VPD should not update its value. If BC1CFG[6] is enabled then this threshold is an absolute threshold instead of being relative to the current VPD value.

BC1THRH (Baseline Correction 1 THReshold High) This sets the threshold above the current value of the VPD for when the VPD should not update its value. If BC1CFG[6] is enabled then this threshold is an absolute threshold instead of being relative to the current VPD value.

BC1CFG (Baseline Correction 1 ConFiGuration)

3:0 This sets the number of taps of the VPD filter.

4 When the signal has been outside the acceptance thresholds for a specified time given by BC1RSTCNT the filter will reset and open up the thresholds so that it can regain the correct baseline and avoid getting stuck. If this bit is set high it will open the thresholds for 31 cycles, and if it is low it will open them for 15 cycles.

5 The VPD filter can be set to only run when outside of a time window or it can be force to run all the time if this is enabled. If running in continuous mode and one wants to use the VPD then this needs to be enabled.

6 Enabling this changes BC1THRL and BC1THRH to being absolute thresholds instead of relative.

8:7 This shifts the output value from the pedestal memory in the specified amount of steps. The final output value is 13 bits, two's complement with two bit precision. With a value of 0 the pedestal memory value is 8 bits with 2 bits of extra precision, with a value of 1 the pedestal memory value is 9 bits with 1 bit of extra precision, with a value of 3 the pedestal memory value is 10 bits and with a value of 4 the pedestal memory value is multiplied by 2 and is 10 bits two's complement.

9 If this is set, then negative value on the output of the BC1 filter will be clipped to zero.

BC1RSTCFG (Baseline Correction 1 ReSeT ConFiGuration) This specifies the number of samples that the signal can be outside the thresholds before the auto resetting steps in and the thresholds are opened. It is given in steps of four so the required number of samples must be divided by four. If the value is set to zero then the auto-resetting feature is disabled.

Register name		Address	Type	Default	Description	
FPD	[12:0]	0x0C	RW	0x000	[12:0]	BC1 Fixed pedestal (offset subtracted), 2's compliment, 2 bit precision
VPD	[12:0]	0x0D	R	0x000	[12:0]	BC1 variable pedestal, 2's compliment
DPCFG	[11:0]	0x18	RW	0x000		Data path configuration
			RW	0x000	[4]	BC1 data input polarity
			RW	0x000	[5]	BC1 pedestal memory polarity
BC1THRL	[10:0]	0x19	RW	0x7FD	[10:0]	Lower threshold of variable pedestal filter, 2's compliment
BC1THRH	[10:0]	0x1A	RW	0x003	[10:0]	Higher threshold of variable pedestal filter, 2's compliment
BC1CFG	[9:0]	0x1B	RW	0x114		BC1 configuration
			RW	0x004	[3:0]	Number of taps in variable pedestal filter
			RW	0x001	[4]	Define open threshold time of 31(high)/15(low) samples after (auto)reset
			RW	0x000	[5]	Force enable IIR also inside time window
			RW	0x000	[6]	High if BC1THR should be considered absolute, else relative
			RW	0x002	[8:7]	Shift output data of pedestal memory
			RW	0x000	[9]	BC1 negative clipping enabled
BC1RSTCNT	[7:0]	0x1C	RW	0x000	[7:0]	Number of samples outside of thresholds before resetting vpd filter (divided by 4)
			RW	0x000		0 disables the auto reset

Table 3.18: *BC1 registers*

DPCFG[3:0]	Effect
0x0	din - FPD
0x1	din - f(t)
0x2	din - f(din)
0x3	din - f(din - VPD)
0x4	din - VPD - FPD
0x5	din - VPD - f(t)
0x6	din - VPD - f(din)
0x7	din - VPD - f(din - VPD)
0x8	f(din) - FPD
0x9	f(din - VPD) - FPD
0xA	f(t) - FPD
0xB	f(t) - f(t)
0xC	f(din) - f(din)
0xD	f(din - VPD) - f(din - VPD)
0xE	din - FPD
0xF	din - FPD

Table 3.19: *Operating modes of the first Baseline Correction.*
The lookup function f( ) *is the pedestal memory with the argument as the address.*

3.2.4 BC2

BC2BSL (Baseline Correction 2 BaSeLine) This is the baseline as calculated by the BC2 filter.

BC2LTHRREL (Baseline Correction 2 Low THReshold RELative) This sets the threshold below the current value of the calculated baseline for when the filter should not update its baseline value.

BC2HTHRREL (Baseline Correction 2 High THReshold RELative) This sets the threshold above the current value of the calculated baseline for when the filter should not update its baseline value.

BC2LTHRBSL (Baseline Correction 2 Low THReshold BaSeLine) This sets the minimum absolute value that the baseline can have before auto-reset countdown starts.

BC2HTHRBSL (Baseline Correction 2 High THReshold BaSeLine) This sets the maximum absolute value that the baseline can have before auto-reset countdown starts.

BC2CFG (Baseline Correction 2 ConFiGuration)

- **1:0** Select the number of taps in the moving average filter, 3 = 8 taps, 2 and 1 = 4 taps, 0 = 2 taps.
- **3:2** Sets the number of samples before (pre-samples) the signal crosses out of the acceptance threshold that should be excluded from the averaging.
- **7:4** Sets the number of samples after (post-samples) the signal crosses in to the acceptance threshold that should be excluded from the averaging.
- **8** If this is enabled, then single samples that passes out of or into the thresholds do not trigger the pre- and post-samples.
- **10:9** This selects the method of auto-resetting the BC2 filter. A value of 3 selects the open threshold method of reset. Selecting 2, enable the autoreset of the baseline to the current BC3 baseline, a value of 1 resets the baseline to the value given in BC2RSTVAL, 0 will disable the auto-resetting functionality.

BC2RSTVAL (Baseline Correction 2 ReSeT VALue) When BC2CFG[10:9] is set to 1, then this value is used when the BC2 filter needs to be reset (the value BC2RSTVAL is multiplied by 4 when loaded in to a 13 bit signed datapath register).

BC2RSTCNT (Baseline Correction 2 ReSeT CouNT) When auto-reset is enabled, then this specifies the number of samples outside the threshold before the auto-resetting steps in. It is given in steps of four so the required number of samples must be divided by four.

Register name		Address	Type	Default	Description	
BC2BSL	[12:0]	0x0E	R	0x000	[12:0]	BC2 Computed Baseline, 2's compliment
BC2LTHRREL	[9:0]	0x11	RW	0x003	[9:0]	BC2 lower relative threshold
BC2HTHRREL	[9:0]	0x12	RW	0x003	[9:0]	BC2 higher relative threshold
BC2LTHRBSL	[10:0]	0x13	RW	0x400	[10:0]	BC2 lower saturation level for baseline
BC2HTHRBSL	[10:0]	0x14	RW	0x3FF	[10:0]	BC2 higher saturation level for baseline
BC2CFG	[10:0]	0x15	RW	0x000		BC2 configuration
			RW	0x000	[1:0]	Number of taps in moving average filter
			RW	0x000	[3:2]	BC2 pre-samples
			RW	0x000	[7:4]	BC2 post-samples
			RW	0x000	[8]	BC2 glitch removal
			RW	0x000	[10:9]	Auto reset configuration
BC2RSTVAL	[7:0]	0x16	RW	0x032	[7:0]	Reset value for maf baseline when auto reset is enabled
BC2RSTCNT	[7:0]	0x17	RW	0x0FF	[7:0]	Number of samples outside of thresholds before resetting maf filter (divided by 4)

Table 3.20: *BC2 registers*

3.2.5 BC3

BC3BSL (Baseline Correction 3 BaSeLine) This is the baseline as calculated by the BC3 filter.

BC3SLD (Baseline Correction 2 SLope Down) This is the rate at witch the average changes in the downward direction, when the signal is below the current average. The value is given with 0.25 bits precision.

BC3SLU (Baseline Correction 2 SLope Up) This is the rate at witch the average changes in the upward direction, when the signal is above the current average. The value is given with 0.25 bits precision.

Register name		Address	Type	Default	Description	
BC3BSL	[12:0]	0x0F	R	0x000	[12:0]	BC3 Computed Baseline, 2's compliment
BC3SLD	[7:0]	0x1D	RW	0x020	[7:0]	Rate of the BC3 baseline down counter
BC3SLU	[7:0]	0x1E	RW	0x010	[7:0]	Rate of the BC3 baseline up counter

Table 3.21: *BC3 registers*

3.2.6 Zero suppression

ZSTHR (Zero-Suppression THReshold) Signals below this value will be suppressed and not included in the data stream. The value is given with 0.25 bits precision. Setting this value to zero effectively disables zero suppression as long as no filters are enabled which can bring the sample value below zero.

ZSOFF (Zero-Suppression OFFset) Since data is transmitted as 10 bits unsigned, there is a necessity to truncate the data to only positive values. To avoid losing information, like the tale of pulses that can pass below the average, it is possible to add an offset to all values before truncation. As this happens before the zero-suppression threshold decision, any offset added here needs to be taken into account when setting the threshold.

ZSCFG (Zero-Suppression ConFiGuration)

1:0 Glitch filtering setting to remove spurious signals above zero-suppression threshold. With this set to 4 , then pulses that are 3 samples or shorter will be remove. If it is 3, then pulses that are 2 samples or shorter will be removed. With this set to 2, then single samples above threshold will be removed. Setting it to 0 will accept all pulses.

4:2 This sets the number of samples before the signal passes above the threshold that should be included. (Samples before pulse starts)

6:5 This sets the number of samples after the signal has passed below the threshold that should also be included. (Samples after pulse ends)

8 This enables sending of samples that have not been modified by the BC2 or BC3 filter, but the zero-suppression threshold decision is still done on the filtered baseline.

Register name	Address	Type	Default	Description	
ZSTHR [11:0]	0x09	RW	0x000	[11:0]	Zero suppression threshold, 2 bit precision
ZSOFF [12:0]	0x0A	RW	0x000	[12:0]	Offset added before truncation, 2's compliment, 2 bit precision
ZSCFG [8:0]	0x0B	RW	0x000		Zero suppression configuration
		RW	0x000	[1:0]	Glitch filter, minimum accepted pulse, all, >1, >2, >2
		RW	0x000	[4:2]	Post-samples
		RW	0x000	[6:5]	Pre-samples
		RW	0x000	[7]	Change position of BC3 in pipeline (BC3 after BC2)
		RW	0x000	[8]	Enable Raw data output of ZSU

Table 3.22: *Zero suppression registers*

Chapter 4

Circuit description

This section gives a description of the various circuit blocks shown in figure 4.1.

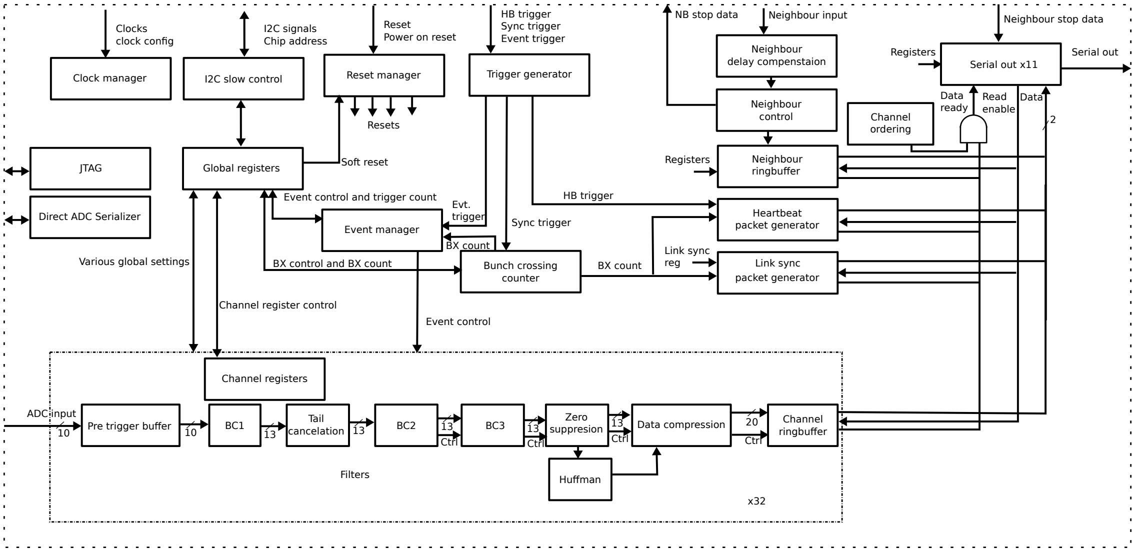


Figure 4.1: *SAMPA block diagram.*

4.1 Analogue front-end

4.2 ADC

4.3 Configurable Reference Voltage Source

The Configurable Reference Voltage Source circuit is based on a selectable resistor array on the output of the chip Bandgap circuitry.

The circuit provide three main voltage levels that are critical for the proper functionality of the analog section of the SAMPA chip.

Then main reference voltages are named according to its nominal expected values respectively 450 mV, 600 mV, 750 mV for the signals $v450$, $v600$ and $v750$.

The reference voltage source is configured using three I2C configurable bits available on the global register unit at ADCTRIM[2:0] configuration value.

The expected levels for each possible configuration are shown in the table table 4.1. It is important to note that these values are subject to process variations.

Setting	Bit 2	Bit 1	Bit 0	$v600$ [mV]	$v750$ [mV]	$v450$ [mV]
0	0	0	0	700.9	850.4	551.3
1	0	0	1	677.3	827.0	527.6
2	0	1	0	652.5	802.3	502.7
3	0	1	1	627.9	777.8	478.0
4	1	0	0	601.8	751.7	451.8
5	1	0	1	577.9	727.9	427.8
6	1	1	0	551.8	701.9	401.6
7	1	1	1	527.0	677.2	376.8

Table 4.1: *Voltage Reference Settings*

The default operational voltages for this block should be adequate with the default startup value (0x04) of the ADCTRIM register. This will provide voltages near the voltage required by the analog block.

If the reference can not be adequately trimmed to the correct value, it is possible to provide externally 750 mV on the $v750$ pin and the other voltages should adjust automatically.

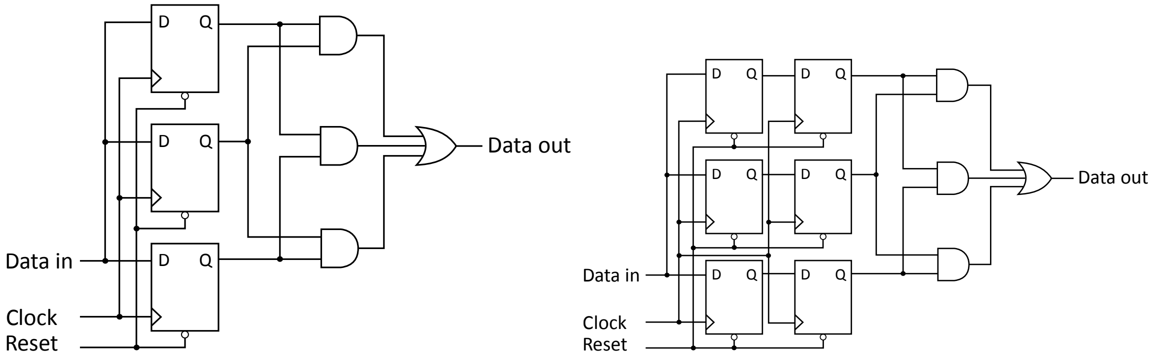
4.4 Triple Modular Redundancy

Most of the registers in the design has been protected through Triple Modular Redundancy (TMR). Figure 4.2a shows the schematic drawing for a normal TMR registers and figure 4.2b shows the schematic for a TMR protected synchronizer. Flip-flops that are implemented as registers or synchronizers and are TMR protected are not shown in this manual with the full TMR circuitry to simplify the circuitry drawings, instead the registers will be marked with "TMR" to indicate that they are TMR protected.

4.5 Filters Data-path

The Filters Module encapsulate the DSP filters in the SAMPA ASIC, providing the configuration values to these blocks and selecting the data path of the signals passing through the SAMPA DSP channel. The diagram 4.3 shows the schematic of the interconnections between the filter modules of the DSP chain.

In the figure 4.3 it is possible to see four different kinds of signals. The main data path signals, the raw data path signals, the configuration signals and the BC3BSL feedback.



(a) *TMR protected register* **(b)** *TMR protected synchronizer*

Figure 4.2: *TMR protection.*

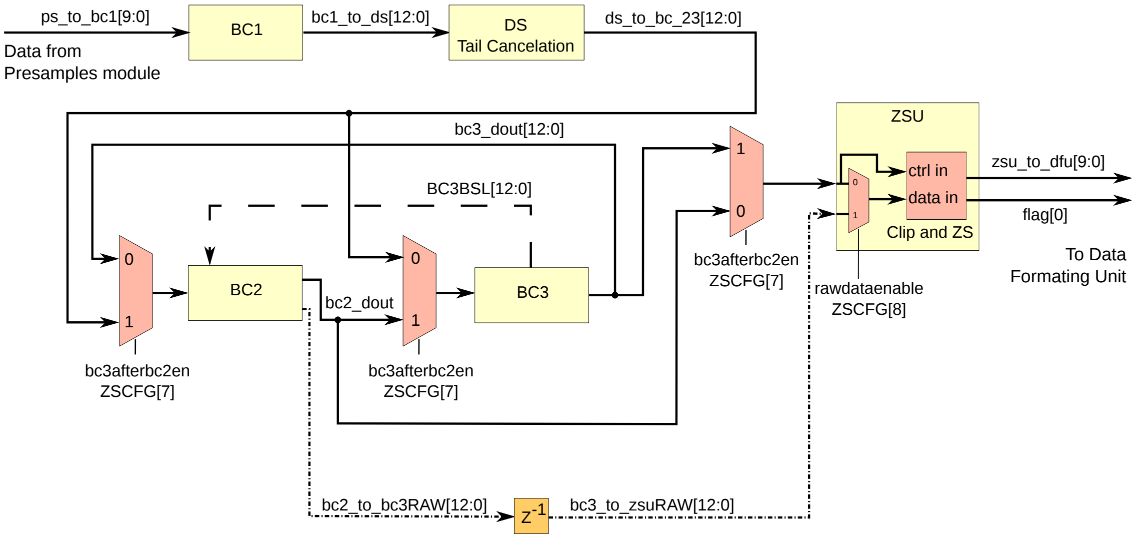


Figure 4.3: *Filters module block diagram.*

Is important to cite that there are the clock and reset signals connected to all the blocks in the diagram 4.3, and they run with the DSP *clkADC* domain, which is the slowest one and works in the same frequency as the digital to analog converter but with a different phase.

The main data path signals are basically the data from the ADC being processed by the chain of filters, it goes straightforward up to the output of DS filter, then the user can configure the order of the filters BC2 (Baseline Correction) and BC3. The filters BC1, BC2, BC3 and DS (Digital Shaper/Tail Cancellation) will be described better in the next sections, but by now is important to note that all the filters have a turnoff/bypass option, so they can be removed from the data path.

The order of the BC2 and BC3 filters can be controlled by the register ZSCFG[7] which is part of the CHRGU (Channel Register Unit), so the filters can be configured and addressed in a per channel basis.

Another function available in the filters module is the ability to use the filtered data to perform zero suppression, and using the raw data path, send out the data from the output of DS, this means that if DS is disabled, and BC1 is in simple FPD (Fixed Pedestal Mode) the user can get unprocessed data in the output. This configuration can be enabled by the ZSCFG[8] bit in the CHRGU.

Finally there is a feedback from BC3 to BC2 named BC3BSL, this can be used as an auto-reset value source to the BC2 filter if it get stuck out of its thresholds. This is useful when running just with BC2 and BC3 is in bypass mode, but still calculating the baseline BC3BSL. To enable this configuration refer to the BC2 section.

4.6 Pre-trigger samples delay

The pre-trigger samples delay module uses a 10x192 bit dualport memory to provide a programmable delay chain for the samples to compensate for any delay in the trigger signal. The delay is set through the PRETRG programmable register with a maximum programmable delay of 192 samples. The top left circuit in figure 4.4 normalizes this programmable delay so it can be used for addressing the memory. The upper limit for the normalized value is set by the size of the memory and the lower is dependent on restriction in the memory core. It is not possible to read and write to the same address so the minimum distance between the read and write pointer needs to be one, which due to intrinsic delay in the memory gives a delay of 2 so one delay needs to be subtracted from the programmable delay value when the memory is used for delaying.

The top right circuitry in figure 4.4 generates the write pointer. It increments on each clock cycle and rolls over at the maximum address of 191.

The middle left circuitry of figure 4.4 calculates the current read pointer from the write pointer. The read pointer is calculated by subtracting the programmable delay value from the current write pointer. If the write pointer is at an address that is larger or equal to the delay value it can be subtracted directly as it won't go negative. If it is less, then a value of 192 is added to it.

The bottom circuitry of figure 4.4 shows the main data path of the module. If the programmable delay is set to zero, the data from the input is passed directly to the output. If the value is one, it is passed through a register due to the memory being unable to read and write at the same address. When the programmable delay is larger than one, the data is passed through the memory.

When the memory is not in use or when the channel has been disabled, then the memory is also disabled with its chip enable signal. The memory will still draw some power beside the static power as long as the input pins are toggling, so the input to the write pointer register is muxed to its previous value, preventing the read and write pointer to change, additionally the data input port to the memory is kept at zero.

4.7 Baseline Correction I

ALTRO description, not updated

As it is shown in figure 4.5, the baseline subtraction circuit is based on a LUT (Pedestal Memory) of 1kx10 bits wide, the IIR filter circuit, a set of multiplexers, which control the modes of operation (described in table 1.1) and a 10-bit adder.

A set of bits controls the circuit:

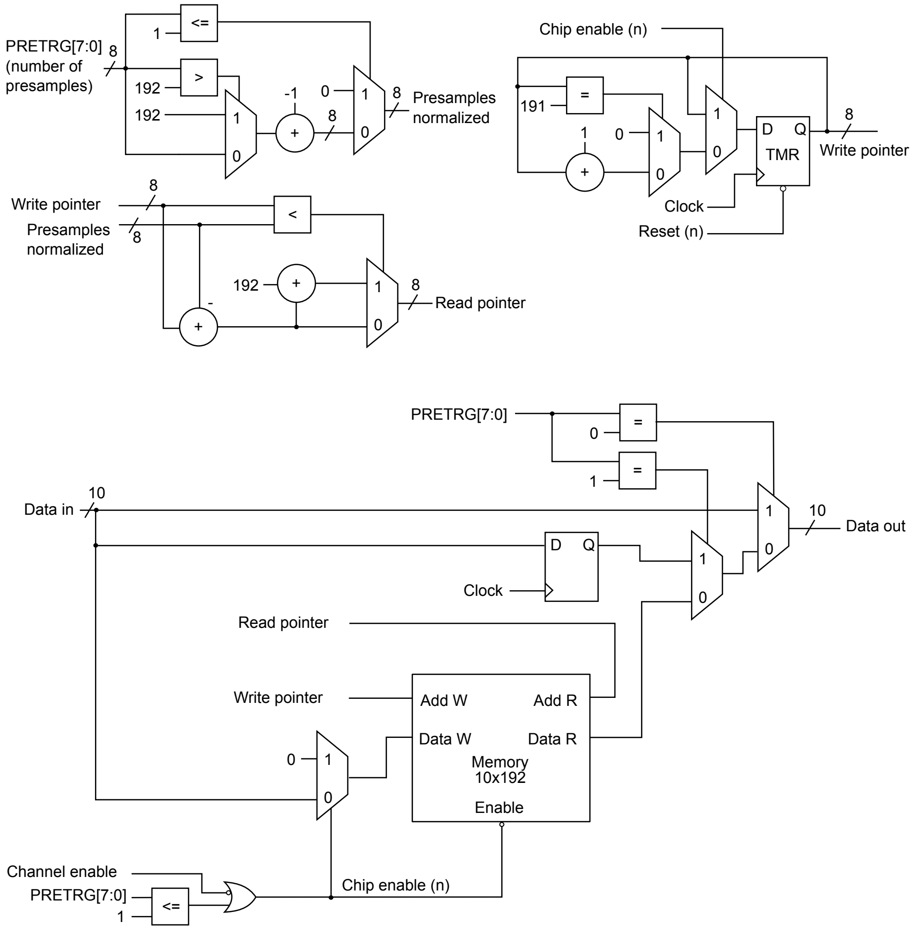


Figure 4.4: *Pre-trigger samples delay schematic.*

bsc0-bsc4 Controls the modes of operation (control of the multiplexers). These bits are decoded from a Configuration Register (BSU Mode, DPCFG Register).

bsc5 Allows the user to control the polarity (1's complement) of the signal (Polarity, DPCFG Register).

bsc6 Allows the user to set the data path to zero in between events to avoid high activity in the Digital Shaper (Power Save, DPCFG2 register).

The pedestal memory is addressed either by the input data (sample) in Conversion mode or by an internal counter in Subtraction and Test mode (time). The IIR filter block self-calibrates subtraction mode. This circuit calculates the cumulative average of a programmable number of samples and subtracts the value to the input samples as it is shown in fig 3.4. The Acqn signal is set when the system is in processing mode (gate open), and therefore controls the window time to calculate the baseline.

ALTRO description, not updated

The IIR formulas are:

$$\text{Baseline}: \quad baseline' = \frac{(2^n - 1) \cdot baseline + ADC}{2^n} \tag{4.1}$$

ALTRO formula, not updated

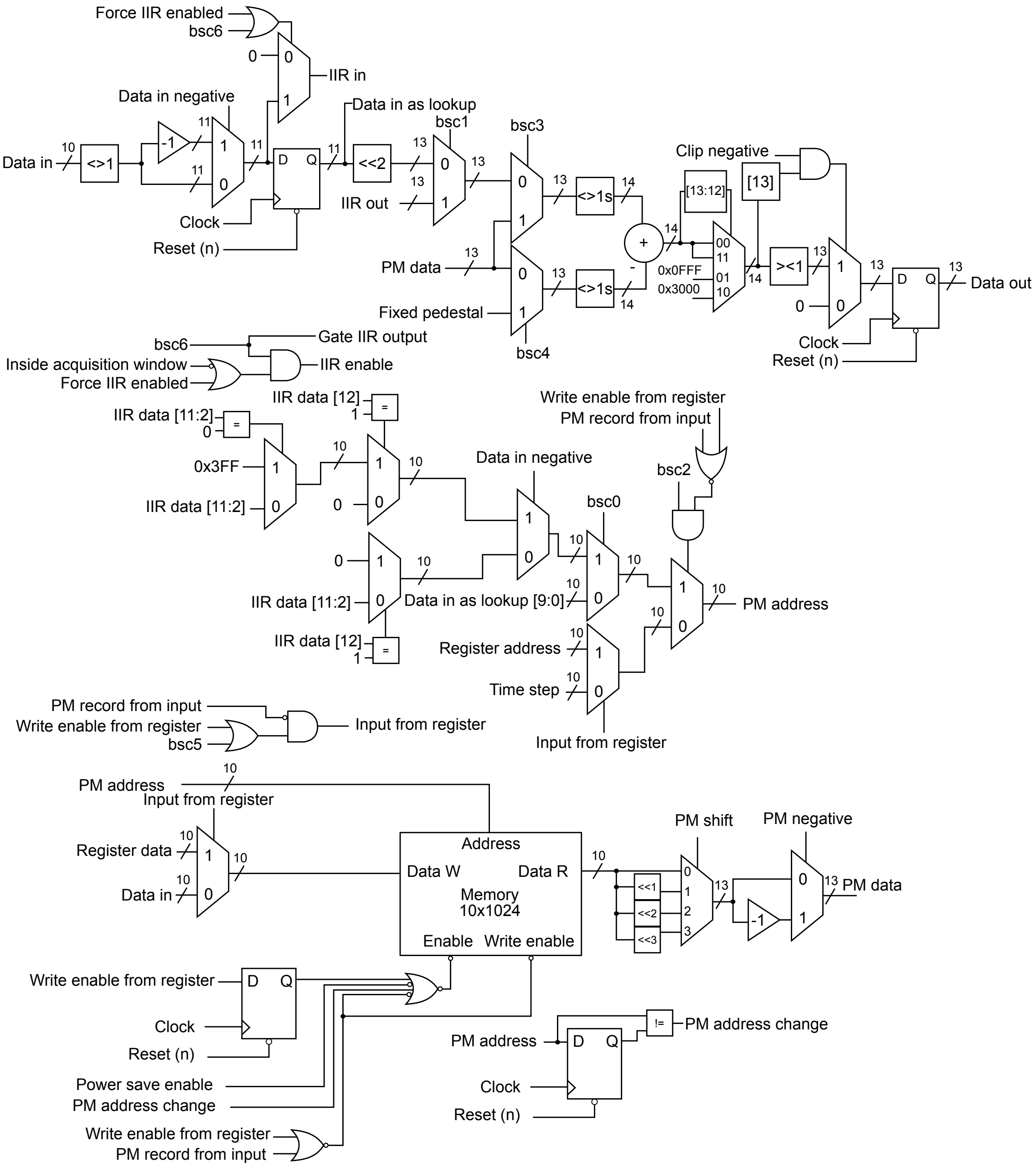


Figure 4.5: *Baseline correction 1 circuit.*

$$\text{Output}: \quad out = ADC - baseline \tag{4.2}$$

$$\text{Implementation}: \quad 2^n \cdot baseline' = 2^n \cdot baseline + out \tag{4.3}$$

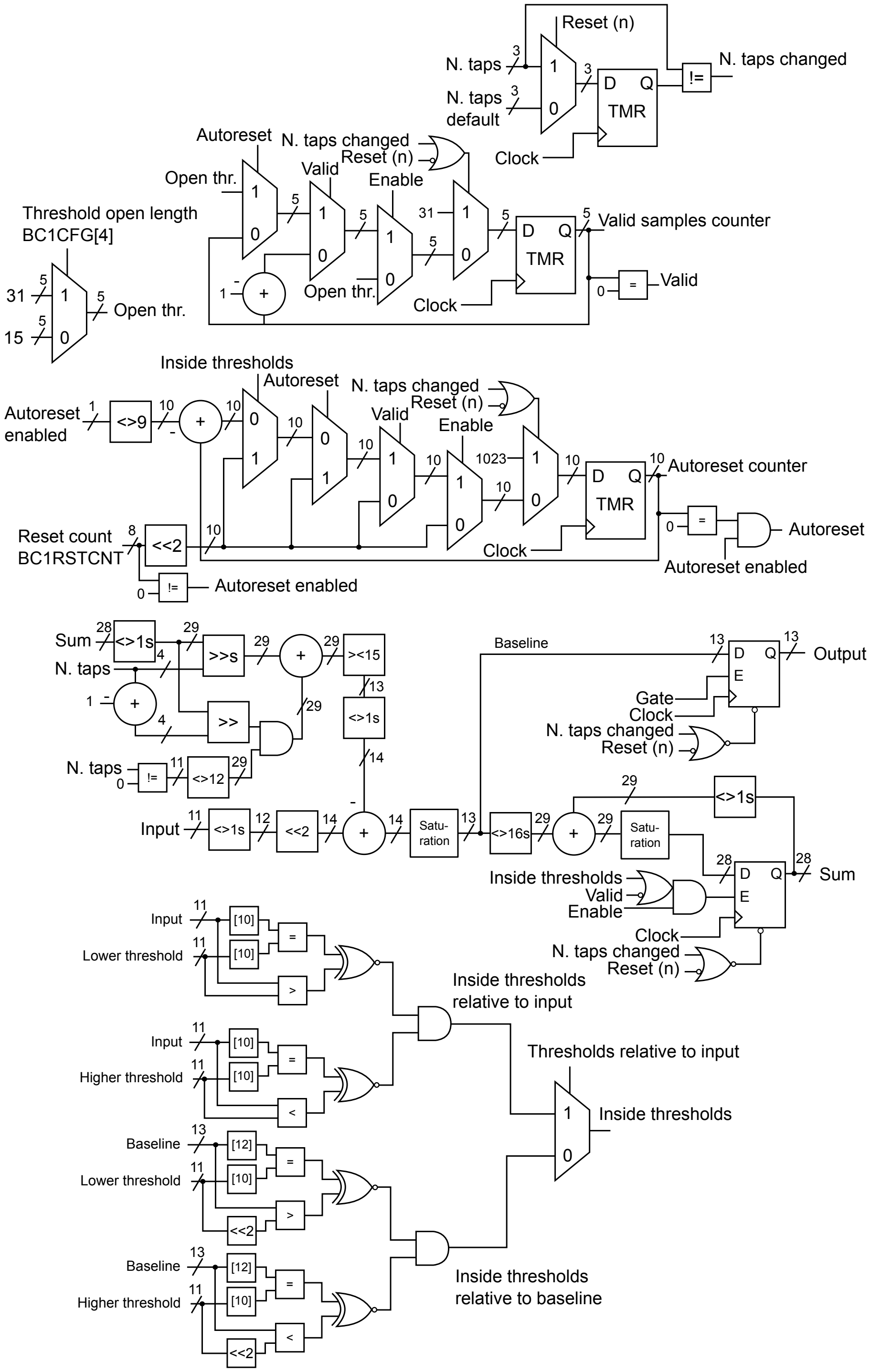


Figure 4.6: *Variable pedestal IIR circuit.*

4.8 Digital shaper

The architecture of the Digital Shaper is implemented as cascade of 4 first order IIR filters as it is shown in the picture above. Each stage of the filter is controlled by means of 2 coefficients (L_i and K_i), which are

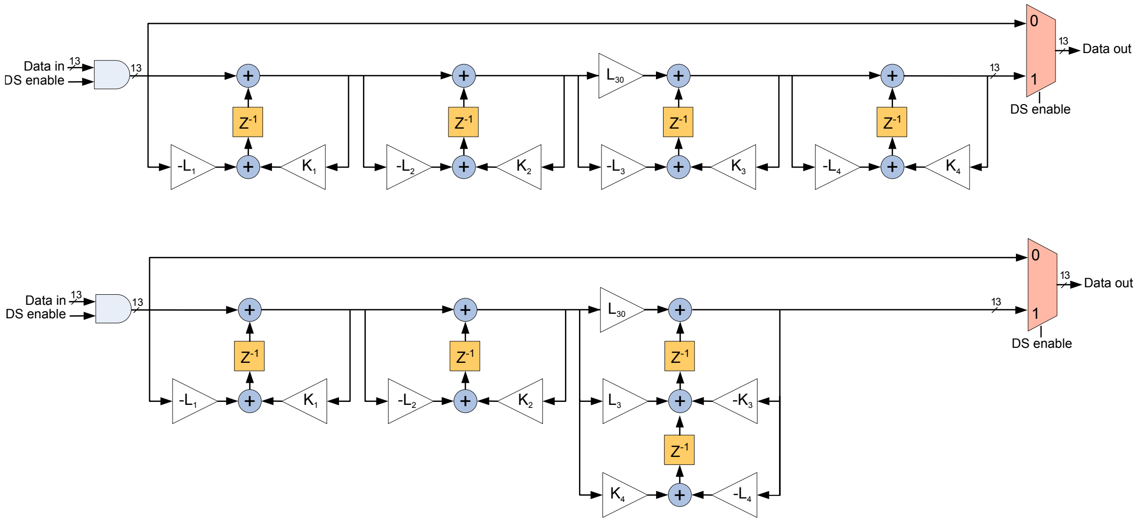


Figure 4.7: *Simplified digital shaper circuit.*

programmed independently. This implementation corresponds to the function in the Z domain:

$$H(z) = \frac{1-L_1z^{-1}}{1-K_1z^{-1}} \cdot \frac{1-L_2z^{-1}}{1-K_2z^{-1}} \cdot \frac{L_{30}-L_3z^{-1}}{1-K_3z^{-1}} \cdot \frac{1-L_4z^{-1}}{1-K_4z^{-1}} \qquad \begin{cases} TCFUSCM = 0 & \forall \quad 0 \le K_i, L_i < 1 \\ TCFUSCM = 1 & \forall -1 \le K_i, L_i < 1 \end{cases} \tag{4.4}$$

$$H(z) = \frac{1-L_1z^{-1}}{1-K_1z^{-1}} \cdot \frac{1-L_2z^{-1}}{1-K_2z^{-1}} \cdot \frac{L_{30}+L_3z^{-1}+K_4z^{-2}}{1+K_3z^{-1}+L_4z^{-1}} \qquad \begin{cases} TCFUSCM = 0 & \forall \quad 0 \le K_i, L_i < 1 \\ TCFUSCM = 1 & \forall -1 \le K_i, L_i < 1 \end{cases} \tag{4.5}$$

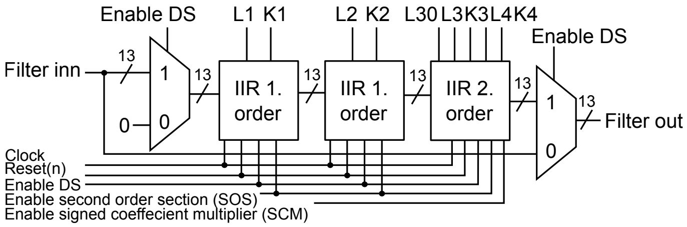


Figure 4.8: *Digital shaper top level circuit.*

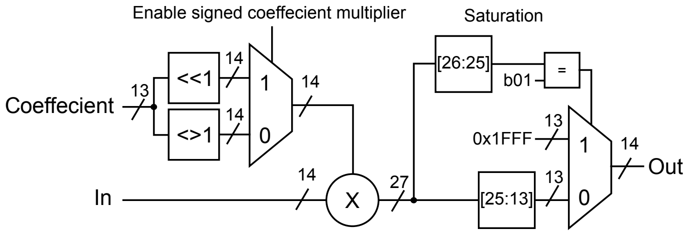


Figure 4.9: *Fixed point two's complement multiplier of digital shaper circuit.*

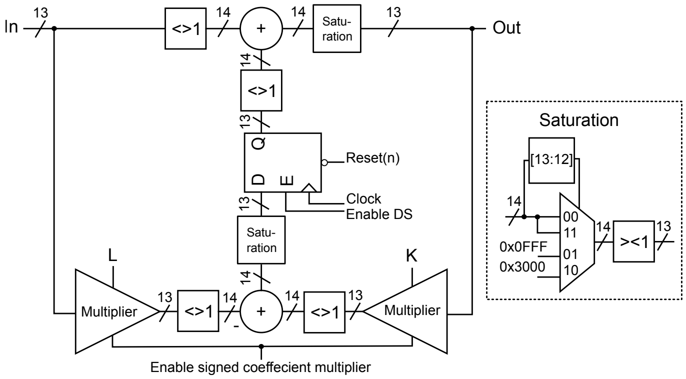


Figure 4.10: *1^{st} order filter of digital shaper circuit.*

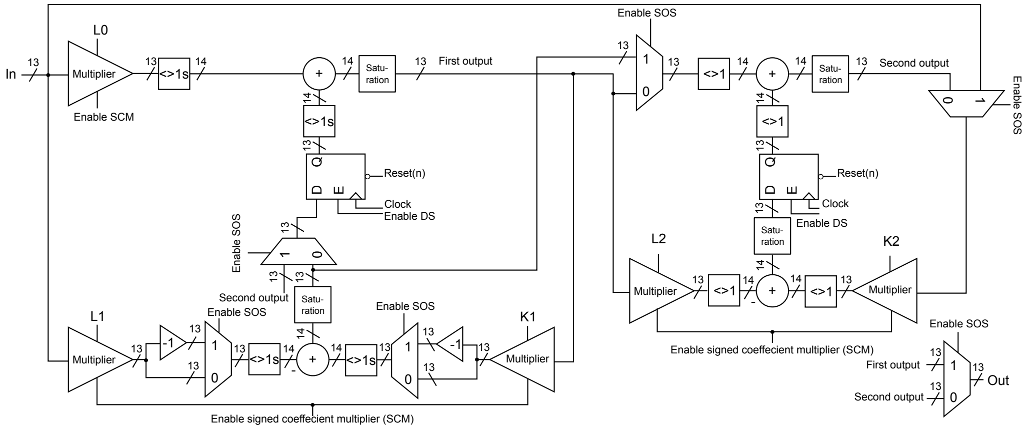


Figure 4.11: *2^{nd} order filter of digital shaper circuit.*

4.9 Baseline Correction II

The Active Baseline Correction circuit is integrated in the data path. This circuit has two main blocks, a double threshold scheme and a moving average filter (figure 4.12).

The double threshold scheme is composed of two comparators and two adders. The value of the thresholds follows the baseline by adding the output value of the moving average filter. It is important to remark that this added value corresponds to the one calculated for a sample 4 cycles before, this allows to have post-samples and presamples in the generation of the exclusion window (fig 1.5). This circuit enables the moving average filter, and therefore determines the window for the adaptive baseline correction.

The input signal is converted to an unsigned signal by adding 4096, this simplifies the architecture of the moving average filter.

At the output, the signal is clipped to a range between 0 and 1023. The values above zero are set to 0. The offset can be useful to keep the information above 0, which is lost when clipping.

When the input signal is out of the margin given by the double threshold, the value given for the moving

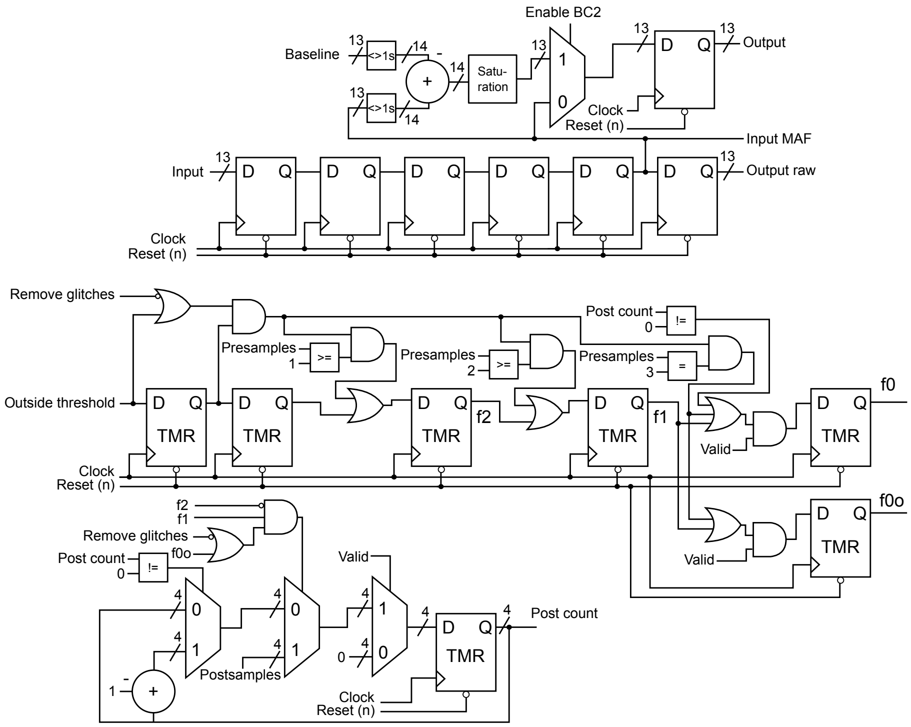


Figure 4.12: *Baseline correction II circuit.*

average filter is frozen.

The Moving Average Filter is based on a FIR system. The function of the filter, for N following samples, in the Z domain is given by the following formula:

$$H(z) = z^{-1}[1 - \frac{1}{N}(1 + z^{-1} + z^{-2} + \dots z^{-(N-1)})] \tag{4.6}$$

The circuit implemented (figure 4.14) is a recursive realization of the FIR system described above for N = 2, 4 or 8. TapsEn configures the number of taps (N).

The block »1, »2 and »3 performs a 1, 2 and 3 bit right shift respectively, which is equivalent to 1/N term in the equation. The maximum round off error in the circuit is 1/2 LSB.

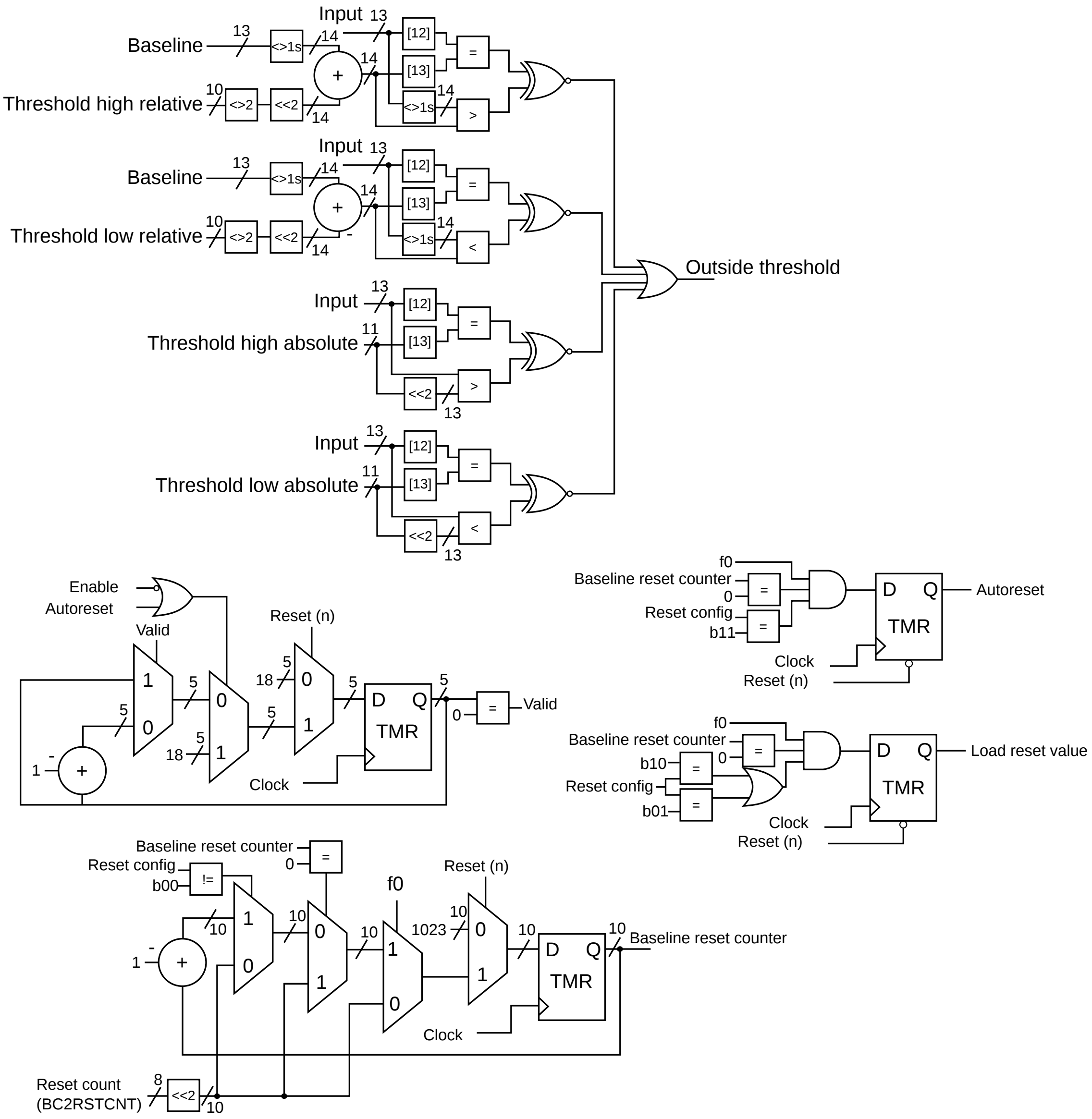


Figure 4.13: *Double threshold scheme for moving average circuit.*

4.10 Baseline Correction III - Slope based

4.11 Huffman

4.12 Zero suppression

The Zero Suppression circuit is based on a fix threshold to generate a flag signal, which is aligned with the data by using a pipeline of 11 clock cycles. This is the same number of delay cycles introduced by the blocks implicated in the generation of the flag, the glitch filter, the pre-sample and post-samples circuit and the cluster merger (figure 4.16).

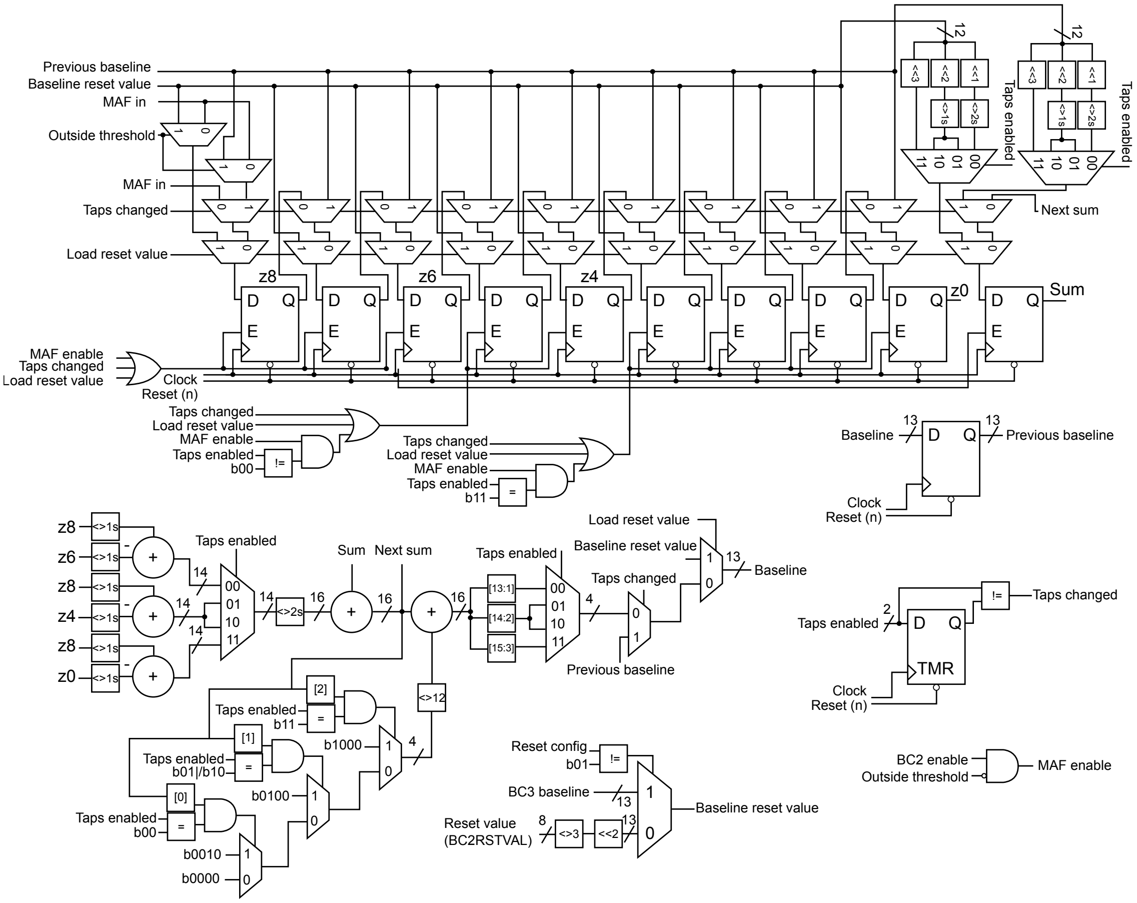


Figure 4.14: *Moving average circuit.*

update

The data is in a 10-bit unsigned format.

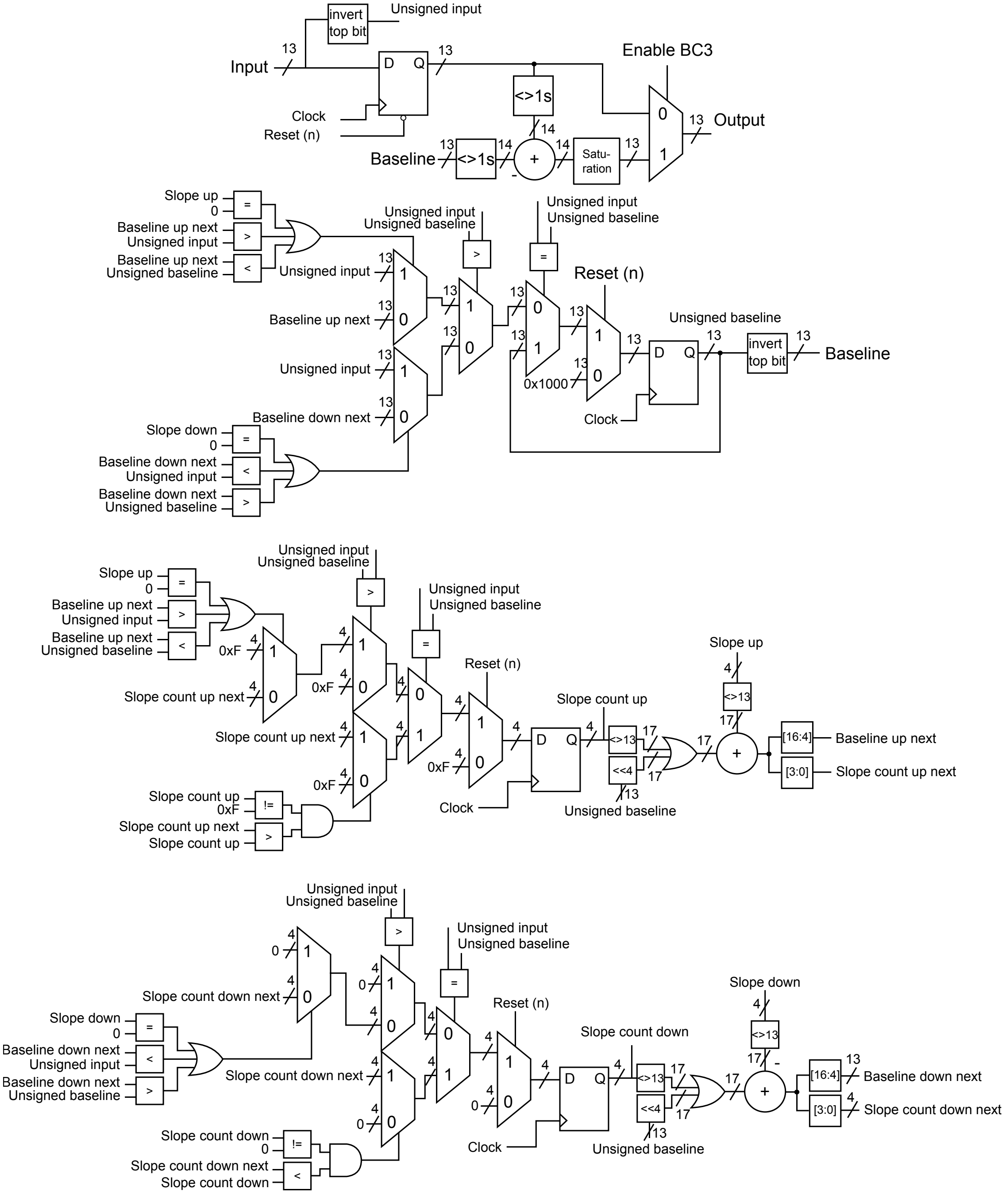


Figure 4.15: *Baseline correction III circuit.*

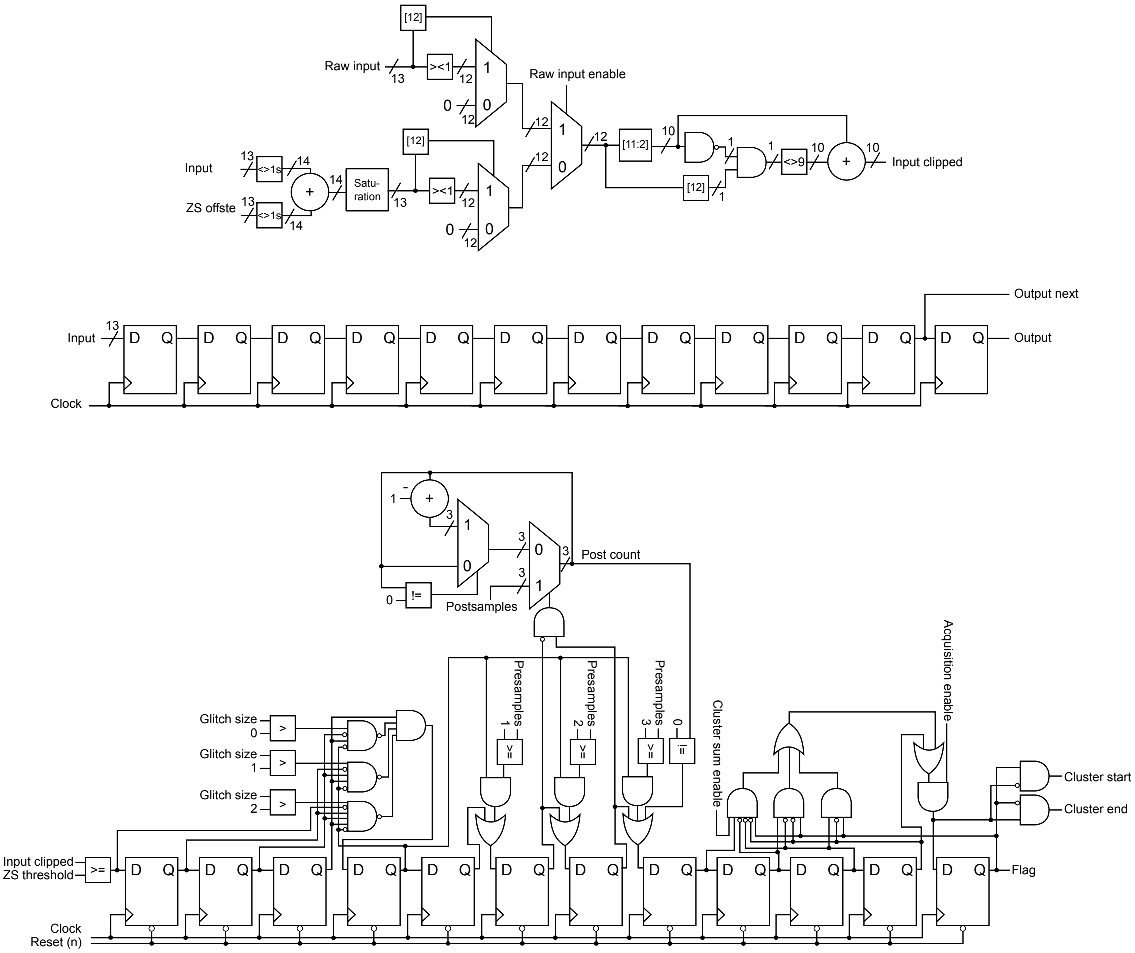


Figure 4.16: *Zero suppression circuit.*

4.13 Data formatting

4.14 Serializing

The functionality of the output interface SerialOut[4:0], NBflowstop_out_SO5 and sdo[4:0] pins is determined by clk_config[6], hb_trg and sme input pins and the BYPASS register. The sme pin and the BYPASS register are used for test modes as their default values are for the standard modes, DSP and DAS.

If $clk\_config[6]\ \&\ \overline{hb\_trg}$ is '1' then the output interface pins are in Direct ADC Serialization mode (DAS mode). if not, but $clk\_config[6]\ \&\ hb\_trg$ is '1' then SerialOut[4:0], NBflowstop_out_SO5 and sdo[3:0] are directly set to one of the 32 ADC outputs chosen by the five bit input {sme, hadd} for test purposes. In this case, sdo[4] is the end of conversion signal (eoc) from the selected ADC. If none of the previous conditions are true (clk_config[6] set to '0') and sme is '1', the memory test is enabled and SerialOut[0] is set to the pulsed memory test output. If sme is '0' SerialOut[0] is set to the internal signal bypass_int that depends on the register BYPASS[3:0]. If this register is 0x0 (default value) SerialOut[0] is set to the serial data link 0.

Setting clk_config[6] to '0' , i.e. the first two previous conditions being false, the NBflowstop_out_SO5 pin functionality depends on the BYPASS register and the internal signal serialOut_en_int[5]. If BYPASS is 0x4 or 0x5 then NBflowstop_out_SO5 is set to $NBflowStop\_in\ \&\ SOCFG[5]$. Where SOCFG[5] enables the NBflowStop_in pin. For any other BYPASS value the NBflowstop_out_SO5 pin functionality depends on the serialOut_en_int[5]. SerialOut_en_int[5] is '1' if the number of enabled serial data links is greater than or equal to five according to the SOCFG[3:0] configuration. In this case NBflowstop_out_SO5 pin is set to the serial data link 5. If SerialOut_en_int[5] is '0', NBflowstop_out_SO5 works as neighbor flow stop of serial output in daisy chaining as its name implies.

With clk_config[6] in '0' serialOut[4:1] is configured to serial data links 4 down to 0 and sdo[4:0] to serial data links 10 down to 6. In figure 4.17 the circuit diagram of the output interface is depicted. In the JTAG block the TRST (JTAG reset) must be in '0' to set the the SerialOut_jtag[4:0], NBflowstop_out_SO5_jtag and sdo_jtag[4:0] to SerialOut[4:0], NBflowstop_out_SO5 and sdo[4:0] pins.

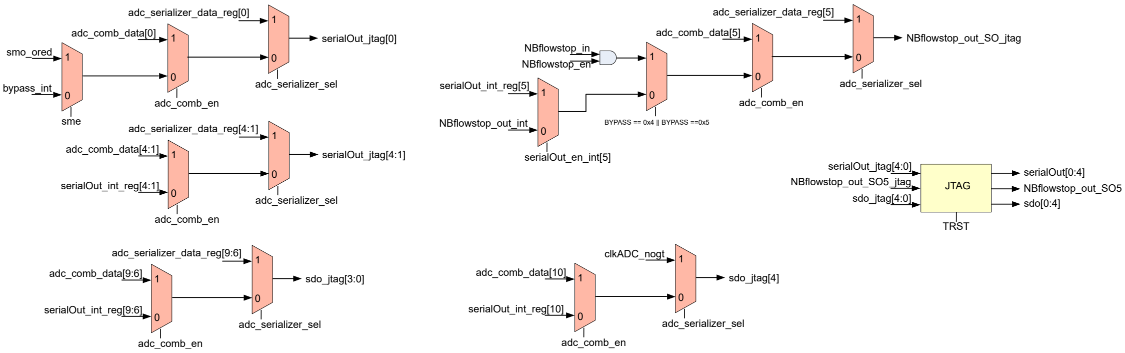


Figure 4.17: *Circuit diagram of the output interface.*

4.15 Direct ADC serialization mode

The ADC serialization module consists of a counter that counts from 0 to 31 running at serial link clock speed, which in turn directly controls a 320-to-10 bit multiplexer for the output. Before the counter rolls over to 0, the data from the ADC is captured and held in a register until the next time the counter reaches the rollover point. The enable signal (trg) is taken from the input and synchronized to the ADC clock (sent through 2 flip-flops running on ADC clock) as seen in figure 4.18 bottom right. It has then an edge which is synchronous to the ADC clock. After that the enable is synchronized to the serial link clock (2 flip-flops) and it will then have a rising edge which is 2 serial link clock-cycles after the rising edge of the ADC clock. The enable signal is used to start the capturing and serialization. As it is always later than the rising edge of the ADC clock it is safe to capture the data without synchronizing all the ADC data. When the enable signal is low, the counter is held at value 1 and the data stored in the ADC capture register is then the synchronization pattern. When the enable goes high it starts counting and sends first the sync pattern and then continues with data. Setting the mode (bx_sync_trg) to '1' enables the split output mode where ch 0-15 is on link 0-4 and ch 16-31 is on link 5-9 with low nibble first and high nibble second. Setting it to '0' will make it send the 10 bits of ch 0-31 consecutively. The direct ADC serialization mode is only activated when the chip has the clk_config[6] clock gating pin high as then it doesn't interfere with the rest of the chip operation.

The start value (synchronization pattern) is the same for both modes and is defined as

```
localparam [32*10-1:0] start_val =
{2{{2{{2{10'b0101001010}},{2{10'b1010110101}}}},{4{10'b0101001010, 10'b1010110101}}}};
```

Which means the serial link output will switch between the two complimentary values 0x2B5 and 0x14A. If we define 0x2B5 as 0 and 0x14A as 1, the pattern observed on the serial links [9:0] per serial link clock cycle in the non-split mode would be
01010101001100110101010100110011
while for the split 5+5 mode it would appear slightly different
00110011001100110000111100001111
as we only send 5 bits per channel per cycle.

Make sure the the enable signal (trg) is low before executing a reset, if not there will be a short sync pattern transmitted after the release of the external reset, which will last until the internal ADC clock has started up again and has reset the enable signal for the ADC domain.

The ADC clock is directly outputted from the internal clock divider module so that it can be used for monitoring of single event upsets in the clock divider. It can also be used for verifying ADC clock synchronicity between different SAMPAs.

The ADC in the SAMPA is a differential ADC. It takes the difference between the two outputs of the analog front-end and converts it to a two's complement value. To get the samples back to bipolar offset binary values [**?**], the top bit is inverted by XORing the ADC value with a constant 10 bit value where the top bit is one.

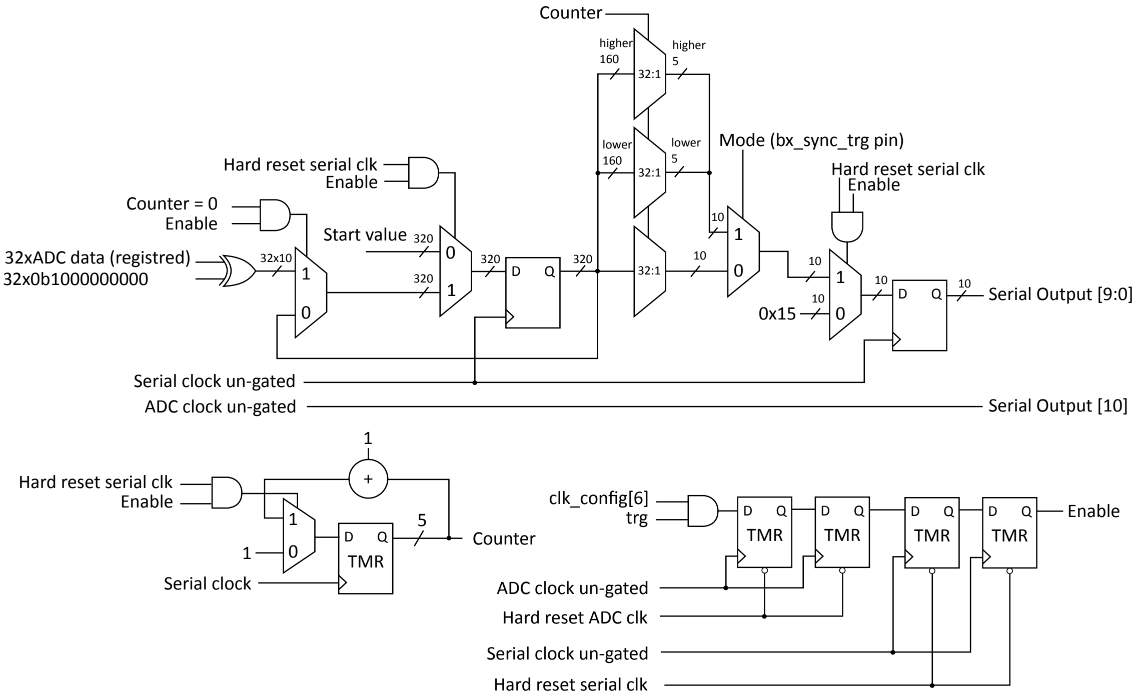


Figure 4.18: *Direct ADC serialization schematic.*

Chapter 5

Datasheet

5.1 Electrical specifications

5.1.1 Power supplies

The SAMPA ASIC has five power-domains

FE Analogue front-end - First Shaper and Charge Sensitive Amplifier

FE2 Analogue Front-end - Second Shaper and Output Buffer

AD ADC

DG Core digital logic

DR SLVS IO drivers

Each power domain has its corresponding VDD_xxx and VSS_xxx net. Additionally there is a separate IO padring ground for the IO pad ESD diodes named just VSS.

The nominal supply voltage of the chip is 1.25 V with variation range of 5%: 1.1875 V <1.25 V <1.325 V

A common PCB ground plane can be used. Depending on noise requirements it is possible to split the power nets on the pcb into [FE + FE2, AD and DG + DR] or [FE + FE2 + AD and DG + DR] or join them all.

5.1.2 ADC voltage reference

The ADC reference voltage should be provided on the VREFP pin and it should be 1.1 V.

5.1.3 Decoupling

For best performance groups of four pins should be decoupled with 100 nF as close as possible to the group. For each fourth group there should be a 1 µF capacitor for bulk decoupling. VREFP decoupling? V750, 600, 450 decoupling? **update**

5.1.4 Absolute maximum ratings

5.1.5 Digital IO pad characteristics

Table 5.1 shows the electrical specifications for the JTAG, I2C, and scanchain control pins

Parameter	Description	Condition	Min	Max
V_{IL}	Input Low Voltage			0.465 V
V_{IH}	Input High Voltage		0.625 V	
V_{OL}	Output Low Voltage		xxx	zzz
V_{OH}	Output High Voltage		xxx	zzz

Table 5.1: *Digital IO pad electrical specifications*

5.1.6 Power consumption

5.1.7 SLVS termination

The SLVS differential input pairs need a 100 ohm differential termination (odd-mode transmission-line impedance between 50 and 62 ohms (56 Ω ± 10 %)) [2]. For non critical signals an option to enable internal termination is available (default off), see table 5.2 for a list of the affected pins. The option is common for all signals and can not be set individually.

Pin name	Termination
clkADCin	External only
clkBXin	External only
clkSOin	External only
Hrstb	External only
bx_sync_trg	Internal/external
trg	Internal/external
hb_trg	Internal/external
NBflowstop_in	Internal/external
dinN	Internal/external

Table 5.2: *Differential input terminations*

5.1.8 Internal pull-up/pull-down

To provide a higher yield in packaging some test pins have been tied high or low internally so that the chip will start up in normal operation mode if the pins are left unconnected. These pins should not have external pull up/pull downs, except possibly TRST as described in section §5.1.9.

- Internal pull-down resistors (10 kΩ): TME, sme, sen0, sen1, sen2, sen3, sen4
- Internal pull-up resistors (10 kΩ) TMS, TRST, TDI, PORin

5.1.9 Reset generation

The digital circuitry has two primary external reset inputs, the hard reset differential input (Hrstb) and the power-on-reset input (PORin), both of which are active low. The inputs are ANDed internally and thus provide the same functionality. The hard reset signal should be connected to the master and the power-on-reset input should either be connected

- To the output of the on-chip power-on-reset circuit (PORout).
- To an on-board power-on-reset sequencer chip.
- To a capacitor, providing a delayed reset after power is applied.
- Left floating, effectively disabling the input.

JTAG reset

The JTAG boundary scan needs to be externally reset to avoid that it interferes with the normal operation. Figure 5.1 shows three examples on how this can be done on the board. The chip has an internal pull-up on TRST and so can be pulled to ground through a resistor less than 4.7 kΩ on the board as shown in figure 5.1A, in the dashed box there is an optional buffer to avoid the current path through the internal pull-up and the external pull-down. Figure 5.1B shows the TRST connected to the power on reset circuitry directly, this would though prevent a JTAG controller from exercising the reset, though this is generally not needed. Figure 5.1C enables both power on reset and JTAG master control.

The TRST pin should neither be left unconnected or pulled high externally as this leaves the JTAG circuitry in an undefined state at startup. It should also not be connected directly to ground as this would continuously reset and disable the JTAG, preventing any operation of the JTAG if needed.

5.1.10 Configuration pins pull-up/pull-down

Clock configuration, chip address and analog front end configuration is set via

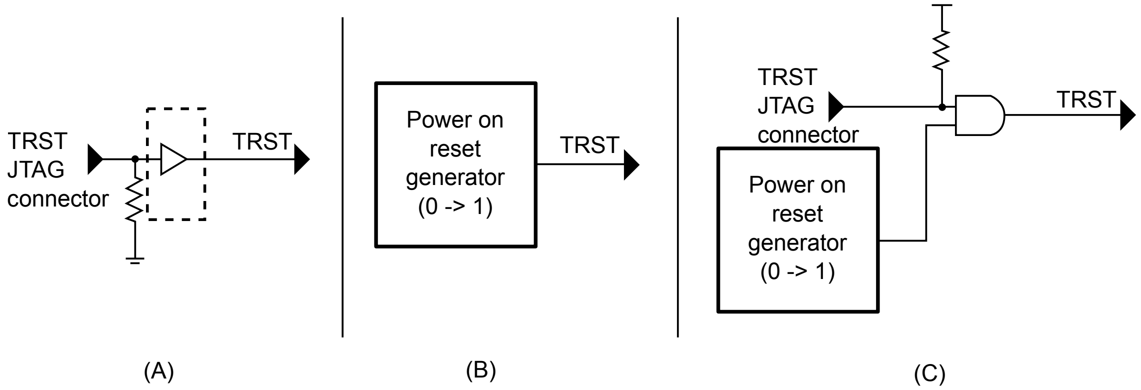


Figure 5.1: *Three examples of board level reset generation for JTAG [3].*

5.1.11 Floating/unused inputs

Unused inputs, except those listed in section §5.1.8, should not be left floating. The unused differential inputs need to maintain a differential voltage of more than 200 mV, taking into account any termination resistor between the differential inputs.

5.1.12 I2C

The SDA pin is open drain and need external pull-up resistors to create a high level. For the default configuration, the size of the pull-up resistor depends primarily on the bus capacitance.

Minimum resistance is given by

$$\begin{aligned} R_p &\geq \frac{Vdd - 0.2Vdd}{I_{ol}} \\ R_p &\geq \frac{0.8 \cdot 1.2}{20 \times 10^{-3}} \\ R_p &\geq 48\Omega \end{aligned}$$

Where I_{ol} is the sinking current of the I2C driver and Vdd is the pull-up voltage.

Maximum resistance is given by

$$\begin{aligned} R_p &\leq \frac{t_r}{0.847298 \cdot C_{BUS}} \\ R_p &\leq \frac{120 \times 10^{-9}}{0.847298 \cdot C_{BUS}} \end{aligned}$$

Where t_r is the maximum rise time for running at 1 MHz given by the standard[1]. C_{BUS} is the capacitance of the bus including the trace capacitance, pin capacitance etc.

Maximum bus capacitance at 1 MHz can then be found to be

$$\begin{aligned} C_{BUS} &\leq \frac{120 \times 10^{-9}}{0.847298 \cdot 48} \\ C_{BUS} &\leq 3\,\mathrm{nF} \end{aligned}$$

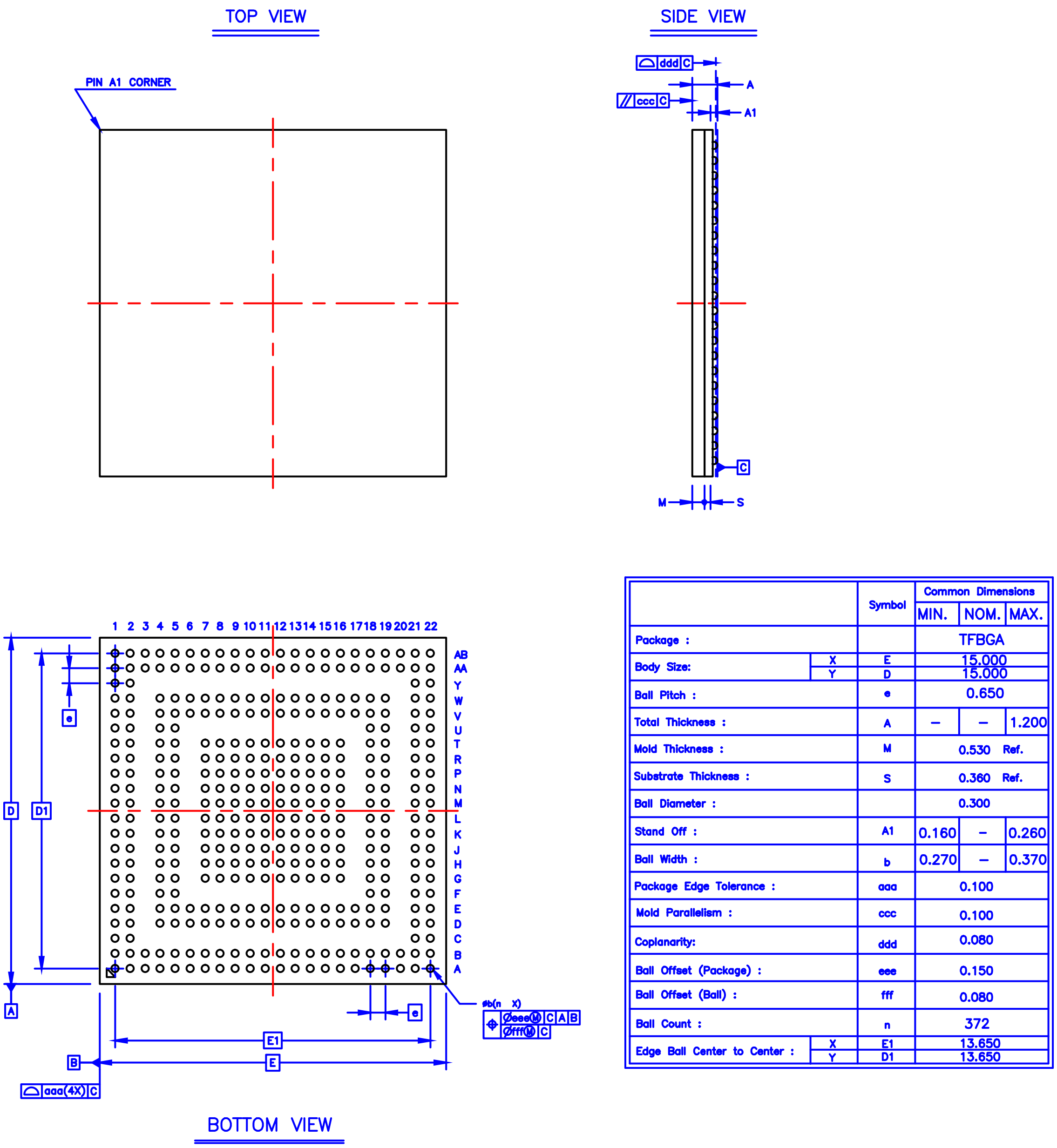
As the SAMPA is an I2C slave only, the SCL pin is a regular CMOS input and so the requirement for a pull-up resistor on this pin depends on the I2C master.

5.2 Package description

5.2.1 Mechanical characteristics

5.2.2 Pinout

All input pins must be driven by a valid logic level: 0 < "logic zero" < 0.2 V or 1.1 < "logic one" < 1.3 V. Unused single-ended input pins should be tie high or low through a resistor (the same resistor can serve many pins). Differential inputs must be set to "0" or "1" by tying the non-inverting input low/high and the inverting input high/low respectively. The method to tie low/high each input of a differential signal is the same as for the single-ended inputs. Output pins can be left floating. Pins marked N/C are unconnected in the package.



	Symbol	Common Dimensions		
		MIN.	NOM.	MAX.
Package :			TFBGA	
Body Size: X	E		15.000	
Body Size: Y	D		15.000	
Ball Pitch :	e		0.650	
Total Thickness :	A	–	–	1.200
Mold Thickness :	M		0.530 Ref.	
Substrate Thickness :	S		0.360 Ref.	
Ball Diameter :			0.300	
Stand Off :	A1	0.160	–	0.260
Ball Width :	b	0.270	–	0.370
Package Edge Tolerance :	aaa		0.100	
Mold Parallelism :	ccc		0.100	
Coplanarity:	ddd		0.080	
Ball Offset (Package) :	eee		0.150	
Ball Offset (Ball) :	fff		0.080	
Ball Count :	n		372	
Edge Ball Center to Center : X	E1		13.650	
Edge Ball Center to Center : Y	D1		13.650	

Figure 5.2: *Packaging details.*

Ball	Netname	Domain	IO type	Direction	Speed	Description
A1	IN[1]	Analog	Analog	in	0	Analog channel 1
A2	IN[0]	Analog	Analog	in	0	Analog channel 0
A3	N/C	-	-	-	-	Not connected
A4	V450	Analog	Power	in/out	0	Bandgap voltage reference (450mV)
A5	V600	Analog	Power	in/out	0	Bandgap voltage reference (600mV)
A6	V750	Analog	Power	in/out	0	Bandgap voltage reference (750mV)
A7	VREFP	Analog	Power	in	0	Voltage reference p
A8	VREFP	Analog	Power	in	0	Voltage reference p
A9	VREFP	Analog	Power	in	0	Voltage reference p
A10	clk_config[3]	Digital	Static	in	Static	Clock configuration
A11	clk_config[4]	Digital	Static	in	Static	Clock configuration
A12	clk_config[5]	Digital	Static	in	Static	Clock configuration
A13	VSS_DG	Digital	Power	in	0	Digital ground
A14	sclk	Digital	CMOS	in	20MHz	Scanchain test clock
A15	sen4	Digital	CMOS	in	20MHz	Scanchain test Scan Enable
A16	sen2	Digital	CMOS	in	20MHz	Scanchain test Scan Enable
A17	sen0	Digital	CMOS	in	20MHz	Scanchain test Scan Enable
A18	N/C	-	-	-	-	Not connected
A19	N/C	-	-	-	-	Not connected
A20	sdo4-	Digital	SLVS	out	160MHz	Scanchain test Scan Out SLVS
A21	N/C	-	-	-	-	Not connected
A22	sdo2-	Digital	SLVS	out	160MHz	Scanchain test Scan Out SLVS
B1	IN[2]	Analog	Analog	in	0	Analog channel 2
B2	N/C	-	-	-	-	Not connected
B3	VDD_FE2	Analog	Power	in	0	Analog core voltage SS and output buffer
B4	VDD_FE2	Analog	Power	in	0	Analog core voltage SS and output buffer
B5	VDD_AD	Analog	Power	in	0	ADC supply voltage
B6	VDD_AD	Analog	Power	in	0	ADC supply voltage
B7	VDD_AD	Analog	Power	in	0	ADC supply voltage
B8	VREFP	Analog	Power	in	0	Voltage reference p
B9	VREFP	Analog	Power	in	0	Voltage reference p
B10	clk_config[0]	Digital	Static	in	Static	Clock configuration
B11	clk_config[1]	Digital	Static	in	Static	Clock configuration
B12	clk_config[2]	Digital	Static	in	Static	Clock configuration
B13	VSS_DG	Digital	Power	in	0	Digital ground
B14	clk_config[6]	Digital	Static	in	Static	Clock configuration (Internal DSP Clock Gatting)
B15	TME	Digital	CMOS	in	Static	Scanchain test âĂŞ TEST MODE EN
B16	sdi4	Digital	CMOS	in	20MHz	Scanchain test Scan In
B17	sdi2	Digital	CMOS	in	20MHz	Scanchain test Scan In
B18	sdi0	Digital	CMOS	in	20MHz	Scanchain test Scan In
B19	N/C	-	-	-	-	Not connected
B20	sdo4+	Digital	SLVS	out	160MHz	Scanchain test Scan Out SLVS
B21	N/C	-	-	-	-	Not connected
B22	sdo2+	Digital	SLVS	out	160MHz	Scanchain test Scan Out SLVS
C1	IN[3]	Analog	Analog	in	0	Analog channel 3
C2	VSS_FE	Analog	Power	in	0	Analog ground FS and CSA
C21	sdo0-	Digital	SLVS	out	160MHz	Scanchain test Scan Out SLVS
C22	sdo0+	Digital	SLVS	out	160MHz	Scanchain test Scan Out SLVS
D1	IN[4]	Analog	Analog	in	0	Analog channel 4
D2	N/C	-	-	-	-	Not connected
D4	VSS	Analog/Digital	Power	in	0	PADRING ground
D5	VSS_AD	Analog	Power	in	0	ADC supply ground
D6	VSS_AD	Analog	Power	in	0	ADC supply ground
D7	VDD_AD	Analog	Power	in	0	ADC supply voltage
D8	VDD_AD	Analog	Power	in	0	ADC supply voltage
D9	VDD_AD	Analog	Power	in	0	ADC supply voltage
D10	VDD_DG	Digital	Power	in	0	Digital core voltage
D11	VDD_DG	Digital	Power	in	0	Digital core voltage
D12	VDD_DG	Digital	Power	in	0	Digital core voltage
D13	VDD_DG	Digital	Power	in	0	Digital core voltage
D14	VDD_DG	Digital	Power	in	0	Digital core voltage
D15	VDD_DG	Digital	Power	in	0	Digital core voltage
D16	VDD_DG	Digital	Power	in	0	Digital core voltage
D17	VDD_DG	Digital	Power	in	0	Digital core voltage
D18	VSS_DG	Digital	Power	in	0	Digital ground
D19	VSS_DG	Digital	Power	in	0	Digital ground
D21	sda_o	Digital	I2C	in/out	1MHz	I2C data
D22	scl	Digital	I2C	in	1MHz	I2C clock
E1	IN[5]	Analog	Analog	in	0	Analog channel 5
E2	VSS_FE	Analog	Power	in	0	Analog ground FS and CSA
E4	VDD_FE	Analog	Power	in	0	Analog core voltage FS and CSA
E5	VSS	Analog/Digital	Power	in	0	PADRING ground
E6	VSS_AD	Analog	Power	in	0	ADC supply ground
E7	VDD_AD	Analog	Power	in	0	ADC supply voltage
E8	VDD_AD	Analog	Power	in	0	ADC supply voltage

Table 5.3: *Pinout part 1*

Ball	Netname	Domain	IO type	Direction	Speed	Description
E9	VDD_AD	Analog	Power	in	0	ADC supply voltage
E10	VDD_DG	Digital	Power	in	0	Digital core voltage
E11	VDD_DG	Digital	Power	in	0	Digital core voltage
E12	VDD_DG	Digital	Power	in	0	Digital core voltage
E13	VDD_DG	Digital	Power	in	0	Digital core voltage
E14	VDD_DG	Digital	Power	in	0	Digital core voltage
E15	VDD_DR	Digital	Power	in	0	Digital SLVS drivers voltage
E16	VDD_DR	Digital	Power	in	0	Digital SLVS drivers voltage
E17	VDD_DG	Digital	Power	in	0	Digital core voltage
E18	NBflowstop_in-	Digital	SLVS	in	160MHz	Stop neighbour data in n
E19	NBflowstop_in+	Digital	SLVS	in	160MHz	Stop neighbour data in p
E21	VSS_DG	Digital	Power	in	0	Digital ground
E22	VSS_DG	Digital	Power	in	0	Digital ground
F1	IN[6]	Analog	Analog	in	0	Analog channel 6
F2	N/C	-	-	-	-	Not connected
F4	VDD_FE	Analog	Power	in	0	Analog core voltage FS and CSA
F5	VSS_FE2	Analog	Power	in	0	Analog ground SS and output buffer
F18	VSS_DG	Digital	Power	in	0	Digital ground
F19	VSS_DG	Digital	Power	in	0	Digital ground
F21	dinN-	Digital	SLVS	in	160MHz	Neigbour chip data n
F22	dinN+	Digital	SLVS	in	160MHz	Neigbour chip data p
G1	IN[7]	Analog	Analog	in	0	Analog channel 7
G2	VSS_FE	Analog	Power	in	0	Analog ground FS and CSA
G4	VDD_FE	Analog	Power	in	0	Analog core voltage FS and CSA
G5	VSS_FE2	Analog	Power	in	0	Analog ground SS and output buffer
G7	VSS_AD	Analog	Power	in	0	ADC supply ground
G8	VSS_DG	Digital	Power	in	0	Digital ground
G9	VSS_DG	Digital	Power	in	0	Digital ground
G10	VSS_DG	Digital	Power	in	0	Digital ground
G11	VSS_DG	Digital	Power	in	0	Digital ground
G12	VSS_DG	Digital	Power	in	0	Digital ground
G13	VSS_DG	Digital	Power	in	0	Digital ground
G14	VSS_DR	Digital	Power	in	0	Digital SLVS drivers ground
G15	VSS_DG	Digital	Power	in	0	Digital ground
G16	VDD_DG	Digital	Power	in	0	Digital core voltage
G18	NBflowstop_out_SO5-	Digital	SLVS	out	160MHz	Stop neighbour data out or serial data link 5 n
G19	NBflowstop_out_SO5+	Digital	SLVS	out	160MHz	Stop neighbour data out or serial data link 5 p
G21	VSS_DG	Digital	Power	in	0	Digital ground
G22	VSS_DG	Digital	Power	in	0	Digital ground
H1	IN[8]	Analog	Analog	in	0	Analog channel 8
H2	IN[9]	Analog	Analog	in	0	Analog channel 9
H4	VDD_FE	Analog	Power	in	0	Analog core voltage FS and CSA
H5	VSS_FE2	Analog	Power	in	0	Analog ground SS and output buffer
H7	VSS_AD	Analog	Power	in	0	ADC supply ground
H8	VSS_DG	Digital	Power	in	0	Digital ground
H9	VSS_DG	Digital	Power	in	0	Digital ground
H10	VSS_DG	Digital	Power	in	0	Digital ground
H11	VSS	Analog/Digital	Power	in	0	PADRING ground
H12	VSS_DG	Digital	Power	in	0	Digital ground
H13	VSS	Analog/Digital	Power	in	0	PADRING ground
H14	VSS	Analog/Digital	Power	in	0	PADRING ground
H15	VSS	Analog/Digital	Power	in	0	PADRING ground
H16	VDD_DG	Digital	Power	in	0	Digital core voltage
H18	VSS_DG	Digital	Power	in	0	Digital ground
H19	VSS_DG	Digital	Power	in	0	Digital ground
H21	serialOut-[3]	Digital	SLVS	out	160MHz	Serial data link 3 n
H22	serialOut+[3]	Digital	SLVS	out	160MHz	Serial data link 3 p
J1	IN[10]	Analog	Analog	in	0	Analog channel 10
J2	IN[11]	Analog	Analog	in	0	Analog channel 11
J4	VDD_FE	Analog	Power	in	0	Analog core voltage FS and CSA
J5	VSS_FE	Analog	Power	in	0	Analog ground FS and CSA
J7	VSS_AD	Analog	Power	in	0	ADC supply ground
J8	VSS_DG	Digital	Power	in	0	Digital ground
J9	VSS_DG	Digital	Power	in	0	Digital ground
J10	VSS_DG	Digital	Power	in	0	Digital ground
J11	VSS_DG	Digital	Power	in	0	Digital ground
J12	VSS_DG	Digital	Power	in	0	Digital ground
J13	VSS_DG	Digital	Power	in	0	Digital ground
J14	VSS_DG	Digital	Power	in	0	Digital ground
J15	VSS_DR	Digital	Power	in	0	Digital SLVS drivers ground
J16	VDD_DR	Digital	Power	in	0	Digital SLVS drivers voltage
J18	serialOut-[4]	Digital	SLVS	out	160MHz	Serial data link 4 n
J19	serialOut+[4]	Digital	SLVS	out	160MHz	Serial data link 4 p
J21	VSS_DG	Digital	Power	in	0	Digital ground
J22	VSS_DG	Digital	Power	in	0	Digital ground

Table 5.4: *Pinout part 2*

Ball	Netname	Domain	IO type	Direction	Speed	Description
K1	IN[12]	Analog	Analog	in	0	Analog channel 12
K2	IN[13]	Analog	Analog	in	0	Analog channel 13
K4	VDD_FE	Analog	Power	in	0	Analog core voltage FS and CSA
K5	VSS_FE	Analog	Power	in	0	Analog ground FS and CSA
K7	VSS_AD	Analog	Power	in	0	ADC supply ground
K8	VSS_DG	Digital	Power	in	0	Digital ground
K9	VSS_DG	Digital	Power	in	0	Digital ground
K10	VSS_DG	Digital	Power	in	0	Digital ground
K11	VSS	Analog/Digital	Power	in	0	PADRING ground
K12	VSS_DG	Digital	Power	in	0	Digital ground
K13	VSS	Analog/Digital	Power	in	0	PADRING ground
K14	VSS_DR	Digital	Power	in	0	Digital SLVS drivers ground
K15	VSS_DG	Digital	Power	in	0	Digital ground
K16	VDD_DG	Digital	Power	in	0	Digital core voltage
K18	VSS_DG	Digital	Power	in	0	Digital ground
K19	VSS_DG	Digital	Power	in	0	Digital ground
K21	serialOut-[2]	Digital	SLVS	out	160MHz	Serial data link 2 n
K22	serialOut+[2]	Digital	SLVS	out	160MHz	Serial data link 2 p
L1	IN[14]	Analog	Analog	in	0	Analog channel 14
L2	IN[15]	Analog	Analog	in	0	Analog channel 15
L4	VDD_FE	Analog	Power	in	0	Analog core voltage FS and CSA
L5	VSS_FE	Analog	Power	in	0	Analog ground FS and CSA
L7	VSS_AD	Analog	Power	in	0	ADC supply ground
L8	VSS_DG	Digital	Power	in	0	Digital ground
L9	VSS_DG	Digital	Power	in	0	Digital ground
L10	VSS_DG	Digital	Power	in	0	Digital ground
L11	VSS_DG	Digital	Power	in	0	Digital ground
L12	VSS_DG	Digital	Power	in	0	Digital ground
L13	VSS_DG	Digital	Power	in	0	Digital ground
L14	VSS	Analog/Digital	Power	in	0	PADRING ground
L15	VSS	Analog/Digital	Power	in	0	PADRING ground
L16	VDD_DG	Digital	Power	in	0	Digital core voltage
L18	serialOut-[0]	Digital	SLVS	out	160MHz	Serial data link 0 n
L19	serialOut+[0]	Digital	SLVS	out	160MHz	Serial data link 0 p
L21	VSS_DG	Digital	Power	in	0	Digital ground
L22	VSS_DG	Digital	Power	in	0	Digital ground
M1	IN[16]	Analog	Analog	in	0	Analog channel 16
M2	IN[17]	Analog	Analog	in	0	Analog channel 17
M4	VDD_FE	Analog	Power	in	0	Analog core voltage FS and CSA
M5	VSS_FE	Analog	Power	in	0	Analog ground FS and CSA
M7	VSS_AD	Analog	Power	in	0	ADC supply ground
M8	VSS_DG	Digital	Power	in	0	Digital ground
M9	VSS_DG	Digital	Power	in	0	Digital ground
M10	VSS_DG	Digital	Power	in	0	Digital ground
M11	VSS_DG	Digital	Power	in	0	Digital ground
M12	VSS_DG	Digital	Power	in	0	Digital ground
M13	VSS_DG	Digital	Power	in	0	Digital ground
M14	VSS_DR	Digital	Power	in	0	Digital SLVS drivers ground
M15	VSS_DG	Digital	Power	in	0	Digital ground
M16	VDD_DG	Digital	Power	in	0	Digital core voltage
M18	VSS_DG	Digital	Power	in	0	Digital ground
M19	VSS_DG	Digital	Power	in	0	Digital ground
M21	serialOut-[1]	Digital	SLVS	out	160MHz	Serial data link 1 n
M22	serialOut+[1]	Digital	SLVS	out	160MHz	Serial data link 1 p
N1	IN[18]	Analog	Analog	in	0	Analog channel 18
N2	IN[19]	Analog	Analog	in	0	Analog channel 19
N4	VDD_FE	Analog	Power	in	0	Analog core voltage FS and CSA
N5	VSS_FE	Analog	Power	in	0	Analog ground FS and CSA
N7	VSS_AD	Analog	Power	in	0	ADC supply ground
N8	VSS_DG	Digital	Power	in	0	Digital ground
N9	VSS_DG	Digital	Power	in	0	Digital ground
N10	VSS_DG	Digital	Power	in	0	Digital ground
N11	VSS	Analog/Digital	Power	in	0	PADRING ground
N12	VSS_DG	Digital	Power	in	0	Digital ground
N13	VSS	Analog/Digital	Power	in	0	PADRING ground
N14	VSS_DG	Digital	Power	in	0	Digital ground
N15	VSS_DR	Digital	Power	in	0	Digital SLVS drivers ground
N16	VDD_DG	Digital	Power	in	0	Digital core voltage
N18	clkSOin-	Digital	SLVS	in	320MHz	Serial out clock n
N19	clkSOin+	Digital	SLVS	in	320MHz	Serial out clock p
N21	VSS_DG	Digital	Power	in	0	Digital ground
N22	VSS_DG	Digital	Power	in	0	Digital ground
P1	IN[20]	Analog	Analog	in	0	Analog channel 20
P2	IN[21]	Analog	Analog	in	0	Analog channel 21
P4	VDD_FE	Analog	Power	in	0	Analog core voltage FS and CSA

Table 5.5: *Pinout part 3*

Ball	Netname	Domain	IO type	Direction	Speed	Description
P5	VSS_FE	Analog	Power	in	0	Analog ground FS and CSA
P7	VSS_AD	Analog	Power	in	0	ADC supply ground
P8	VSS_DG	Digital	Power	in	0	Digital ground
P9	VSS_DG	Digital	Power	in	0	Digital ground
P10	VSS_DG	Digital	Power	in	0	Digital ground
P11	VSS_DG	Digital	Power	in	0	Digital ground
P12	VSS_DG	Digital	Power	in	0	Digital ground
P13	VSS_DG	Digital	Power	in	0	Digital ground
P14	VSS_DG	Digital	Power	in	0	Digital ground
P15	VSS_DG	Digital	Power	in	0	Digital ground
P16	VDD_DR	Digital	Power	in	0	Digital SLVS drivers voltage
P18	VSS_DG	Digital	Power	in	0	Digital ground
P19	VSS_DG	Digital	Power	in	0	Digital ground
P21	Hrstb-	Digital	SLVS	in	160MHz	Reset n
P22	Hrstb+	Digital	SLVS	in	160MHz	Reset p
R1	IN[22]	Analog	Analog	in	0	Analog channel 22
R2	IN[23]	Analog	Analog	in	0	Analog channel 23
R4	VDD_FE	Analog	Power	in	0	Analog core voltage FS and CSA
R5	VSS_FE	Analog	Power	in	0	Analog ground FS and CSA
R7	VSS_AD	Analog	Power	in	0	ADC supply ground
R8	VSS_DG	Digital	Power	in	0	Digital ground
R9	VSS_DG	Digital	Power	in	0	Digital ground
R10	VSS_DG	Digital	Power	in	0	Digital ground
R11	VSS	Analog/Digital	Power	in	0	PADRING ground
R12	VSS_DG	Digital	Power	in	0	Digital ground
R13	VSS	Analog/Digital	Power	in	0	PADRING ground
R14	VSS	Analog/Digital	Power	in	0	PADRING ground
R15	VSS	Analog/Digital	Power	in	0	PADRING ground
R16	VDD_DG	Digital	Power	in	0	Digital core voltage
R18	clkBXin-	Digital	SLVS	in	40MHz	Bunchcrossing clock n
R19	clkBXin+	Digital	SLVS	in	40MHz	Bunchcrossing clock p
R21	VSS_DG	Digital	Power	in	0	Digital ground
R22	VSS_DG	Digital	Power	in	0	Digital ground
T1	IN[24]	Analog	Analog	in	0	Analog channel 24
T2	VSS_FE	Analog	Power	in	0	Analog ground FS and CSA
T4	VDD_FE	Analog	Power	in	0	Analog core voltage FS and CSA
T5	VSS_FE2	Analog	Power	in	0	Analog ground SS and output buffer
T7	VSS_AD	Analog	Power	in	0	ADC supply ground
T8	VSS_DG	Digital	Power	in	0	Digital ground
T9	VSS_DG	Digital	Power	in	0	Digital ground
T10	VSS_DG	Digital	Power	in	0	Digital ground
T11	VSS_DG	Digital	Power	in	0	Digital ground
T12	VSS_DG	Digital	Power	in	0	Digital ground
T13	VSS_DG	Digital	Power	in	0	Digital ground
T14	VSS_DR	Digital	Power	in	0	Digital SLVS drivers ground
T15	VSS_DG	Digital	Power	in	0	Digital ground
T16	VDD_DG	Digital	Power	in	0	Digital core voltage
T18	VSS_DG	Digital	Power	in	0	Digital ground
T19	VSS_DG	Digital	Power	in	0	Digital ground
T21	trg-	Digital	SLVS	in	160MHz	Event trigger n
T22	trg+	Digital	SLVS	in	160MHz	Event trigger p
U1	IN[25]	Analog	Analog	in	0	Analog channel 25
U2	N/C	-	-	-	-	Not connected
U4	VDD_FE	Analog	Power	in	0	Analog core voltage FS and CSA
U5	VSS_FE2	Analog	Power	in	0	Analog ground SS and output buffer
U18	clkADCin-	Digital	SLVS	in	20MHz	ADC clock n
U19	clkADCin+	Digital	SLVS	in	20MHz	ADC clock p
U21	VSS_DG	Digital	Power	in	0	Digital ground
U22	VSS_DG	Digital	Power	in	0	Digital ground
V1	IN[26]	Analog	Analog	in	0	Analog channel 26
V2	VSS_FE	Analog	Power	in	0	Analog ground FS and CSA
V4	VDD_FE	Analog	Power	in	0	Analog core voltage FS and CSA
V5	VSS	Analog/Digital	Power	in	0	PADRING ground
V6	VSS_AD	Analog	Power	in	0	ADC supply ground
V7	VDD_AD	Analog	Power	in	0	ADC supply voltage
V8	VDD_AD	Analog	Power	in	0	ADC supply voltage
V9	VDD_AD	Analog	Power	in	0	ADC supply voltage
V10	VDD_DG	Digital	Power	in	0	Digital core voltage
V11	VDD_DG	Digital	Power	in	0	Digital core voltage
V12	VDD_DG	Digital	Power	in	0	Digital core voltage
V13	VDD_DG	Digital	Power	in	0	Digital core voltage
V14	VDD_DG	Digital	Power	in	0	Digital core voltage
V15	VDD_DR	Digital	Power	in	0	Digital SLVS drivers voltage
V16	VDD_DR	Digital	Power	in	0	Digital SLVS drivers voltage
V17	VDD_DG	Digital	Power	in	0	Digital core voltage

Table 5.6: *Pinout part 4*

Ball	Netname	Domain	IO type	Direction	Speed	Description
V18	VSS_DG	Digital	Power	in	0	Digital ground
V19	VSS_DG	Digital	Power	in	0	Digital ground
V21	hb_trg-	Digital	SLVS	in	160MHz	Heartbeat trigger n
V22	hb_trg+	Digital	SLVS	in	160MHz	Heartbeat trigger p
W1	IN[27]	Analog	Analog	in	0	Analog channel 27
W2	N/C	-	-	-	-	Not connected
W4	VSS	Analog/Digital	Power	in	0	PADRING ground
W5	VSS_AD	Analog	Power	in	0	ADC supply ground
W6	VSS_AD	Analog	Power	in	0	ADC supply ground
W7	VDD_AD	Analog	Power	in	0	ADC supply voltage
W8	VDD_AD	Analog	Power	in	0	ADC supply voltage
W9	VDD_AD	Analog	Power	in	0	ADC supply voltage
W10	VDD_DG	Digital	Power	in	0	Digital core voltage
W11	VDD_DG	Digital	Power	in	0	Digital core voltage
W12	VDD_DG	Digital	Power	in	0	Digital core voltage
W13	VDD_DG	Digital	Power	in	0	Digital core voltage
W14	VDD_DG	Digital	Power	in	0	Digital core voltage
W15	VDD_DG	Digital	Power	in	0	Digital core voltage
W16	VDD_DG	Digital	Power	in	0	Digital core voltage
W17	VDD_DG	Digital	Power	in	0	Digital core voltage
W18	bx_sync_trg-	Digital	SLVS	in	160MHz	Bunchcrossing counter sync n
W19	bx_sync_trg+	Digital	SLVS	in	160MHz	Bunchcrossing counter sync p
W21	VSS_DG	Digital	Power	in	0	Digital ground
W22	VSS_DG	Digital	Power	in	0	Digital ground
Y1	IN[28]	Analog	Analog	in	0	Analog channel 28
Y2	VSS_FE	Analog	Power	in	0	Analog ground FS and CSA
Y21	sdo1-	Digital	SLVS	out	160MHz	Scanchain test Scan Out SLVS
Y22	sdo1+	Digital	SLVS	out	160MHz	Scanchain test Scan Out SLVS
AA1	IN[29]	Analog	Analog	in	0	Analog channel 29
AA2	N/C	-	-	-	-	Not connected
AA3	CTS	Analog	Static	in	Static	Shaping configuration
AA4	CG[0]	Analog	Static	in	Static	Gain configuration
AA5	VDD_FE2	Analog	Power	in	0	Analog core voltage SS and output buffer
AA6	VDD_FE2	Analog	Power	in	0	Analog core voltage SS and output buffer
AA7	VDD_FE2	Analog	Power	in	0	Analog core voltage SS and output buffer
AA8	VREFP	Analog	Power	in	0	Voltage reference p
AA9	VREFP	Analog	Power	in	0	Voltage reference p
AA10	hadd[2]	Digital	Static	in	Static	Chip address
AA11	hadd[0]	Digital	Static	in	Static	Chip address
AA12	sme	Digital	CMOS	in	Static	MEM Test En
AA13	sdi3	Digital	CMOS	in	20MHz	Scanchain test Scan In
AA14	sen1	Digital	CMOS	in	20MHz	Scanchain test Scan Enable
AA15	smo	Digital	CMOS	out	20MHz	MEM Test Out
AA16	TRST	Digital	CMOS	in	20MHz	JTAG
AA17	TDI	Digital	CMOS	in	20MHz	JTAG
AA18	TDO	Digital	CMOS	out	20MHz	JTAG
AA19	N/C	-	-	-	-	Not connected
AA20	sdo3-	Digital	SLVS	out	160MHz	Scanchain test Scan Out SLVS
AA21	N/C	-	-	-	-	Not connected
AA22	VSS_DG	Digital	Power	in	0	Digital ground
AB1	IN[30]	Analog	Analog	in	0	Analog channel 30
AB2	IN[31]	Analog	Analog	in	0	Analog channel 31
AB3	N/C	-	-	-	-	Not connected
AB4	CG[1]	Analog	Static	in	Static	Gain configuration
AB5	N/C	-	-	-	-	Not connected
AB6	POL	Analog	Static	in	Static	Polarity configuration
AB7	VREFP	Analog	Power	in	0	Voltage reference p
AB8	VREFP	Analog	Power	in	0	Voltage reference p
AB9	VREFP	Analog	Power	in	0	Voltage reference p
AB10	hadd[3]	Digital	Static	in	Static	Chip address
AB11	hadd[1]	Digital	Static	in	Static	Chip address
AB12	sen3	Digital	CMOS	in	20MHz	Scanchain test Scan Enable
AB13	sdi1	Digital	CMOS	in	20MHz	Scanchain test Scan In
AB14	PORin	Digital	CMOS	in	Static	Power on reset input
AB15	PORout	Digital	CMOS	out	Static	Power on reset output
AB16	TMS	Digital	CMOS	in	20MHz	JTAG
AB17	TCLK	Digital	CMOS	in	20MHz	JTAG
AB18	N/C	-	-	-	-	Not connected
AB19	VSS_DG	Digital	Power	in	0	Digital ground
AB20	sdo3+	Digital	SLVS	out	160MHz	Scanchain test Scan Out SLVS
AB21	N/C	-	-	-	-	Not connected
AB22	N/C	-	-	-	-	Not connected

Table 5.7: *Pinout part 5*

5.3 Related documentation

The latest version of this document can be located at
https://svnweb.cern.ch/cern/wsvn/SAMPA/docs/SAMPA%20Specification%20MPW3/sampa.pdf
Other documentation for the project can be located at
https://twiki.cern.ch/twiki/bin/viewauth/ALICE/SAMPA
Source code is available from SVN at
https://svnweb.cern.ch/cern/wsvn/SAMPA/ for browsing or
https://svn.cern.ch/reps/SAMPA/ from a client.
The source code for the digital can be located at digital/branches/MPW3

5.4 Known issues

5.4.1 V2

1. POR circuit is stuck low, if PORout is connected to PORin it will keep the chip in reset.
   Workaround : Leave PORout and PORin unconnected Status: Solution implemented for V3.

2. Inserting or removing injection board on front-end pins can corrupt the digital registers due to injected charge.

3. Touching the analogue reference test points (v400, v650, v700) with a multimeter can corrupt the digital registers due to injected charge.

4. Reading BC2BSL or BC3BSL when the baseline is around 64 or 128 etc could cause incorrect data to be retrieved as the top byte and bottom byte is not read at the same time.
   Workaround: Check multiple times.
   Workaround: A second option is to temporarily disable the BC2 filter and read it when it is off.
   Status: Fixed for V3 (rev766).

5. Sending a soft reset through slow control does not return an ACK as the I2C slave gets reset before it is sent.
   Workaround: Make exception on master.
   Status: Fixed for V3 (rev881).

6. Changing number of serial links without soft resetting could make the channel ordering of the packets to not be in the expected sequence.
   Workaround: Always soft reset after configuration change.
   Status: Won't fix

7. Reference voltage problem (if it is not possible to trim the reference voltage to nominal values).
   Workaround:

   - Monitor the V650 - V450 voltage with a multimeter.
   - Connect a power supply, or adjustable voltage regulator, on V750 pin. A low noise regulator, if possible.
   - Adjust V750 until V650 - V450 = 150 mV
   - Hard reset the chip as the registers can have been corrupted.

   Status: Solution implemented for V3.

8. A generated trigger will transmit data for the previous event while the header is for the current event.
   Workaround: Set ACQEND one value lower than TWLEN when using zero suppression or two lower when using cluster sum.
   Status: Fixed for tentative MPW3 (rev766), verified.

9. Pedestal memory readout mode (f(t)-FPD) starts with reading the last address it should be reading from before it continues from address zero. (If TWLEN is 999, it starts with address 999 and then 0).

Workaround: Put first value in last address.
Status: Fixed for V3 (rev886).

10. ADC values in DAS mode are two's complement.
    Workaround: To get the samples back to unsigned values, just add 512 or invert the top bit.
    Status: Fixed for V3 (rev766).

11. First 10 bit output of sync pattern in DAS mode is repeated twice.
    Workaround: Adapt receiver.
    Status: Fixed for V3 (rev766).

12. When the bandwidth occupancy through the daisy chained link is very high and the payload of the packets are very short (<10) a packet might be lost once in a while.
    Workaround: Lower occupancy.
    Status: Partly fixed for V3 (rev775).

13. With high bandwidth occupancy and short payloads, the header memory might overflow. An incorrect implementation causes the payload to still be written to memory causing incorrect payload to header matching.
    Workaround: Lower occupancy.
    Status: Fixed for V3 (rev786).

14. 11th serial link (ADC clock) needs to be enabled through I2C in DAS mode.
    Workaround: Set SOCFG to 0x1B.
    Status: Fixed for V3 (rev872).

15. If NBflowstop_in is left floating, the serial out 0 might not work depending on the value read from the floating input.
    Workaround: Pull the pin low.
    Status: Fixed for V3 (rev786).

16. The device is sensitive to single event latchups.
    Workaround: External supervision circuitry to reset device in case of latchup.
    Status: Fixed for V3 (replaced SP pedestal mem with DP).

17. Increasing number of neighbours register to more than one forces a sync packet to be sent in between each packet from the neighbour, possibly decreasing efficiency.
    Workaround: None.
    Status: Not fixed

18. A single event upset in the accumulator register (fx) for the BC2 will introduce a permanent offset in the baseline that is subtracted from the signal, until the device is reset.
    Workaround: Enable the autoreset feature of the BC2.
    Status: Not fixed

19. Heart beat packets are not buffered and if a new hearbeat trigger is received before a heartbeat packet has been transmitted it will be discarded.
    Workaround: Keep spacing between heartbeat triggers over 3.5 us.
    Status: Not fixed

5.4.2 V3/v4

1. Inserting or removing injection board on front-end pins can corrupt the digital registers due to injected charge.

2. Touching the analogue reference test points (v400, v650, v700) with a multimeter can corrupt the digital registers due to injected charge.

3. Changing number of serial links without soft resetting could make the channel ordering of the packets to not be in the expected sequence.
   Workaround: Always soft reset after configuration change.
   Status: Won't fix

4. When the bandwidth occupancy through the daisy chained link is very high and the payload of the packets are very short (<10) a packet might be lost once in a while.
   Workaround: Lower occupancy.
   Status: Not fixed

5. Increasing number of neighbours register to more than one forces a sync packet to be sent in between each packet from the neighbour, possibly decreasing efficiency.
   Workaround: None.
   Status: Not fixed

6. Too high occupancy on neighbour link could cause a packet to be lost.
   Workaround: Keep occupancy below 10 percent.
   Status: Not fixed

7. A single event upset in the accumulator register (fx) for the BC2 will introduce a permanent offset in the baseline that is subtracted from the signal, until the device is reset.
   Workaround: Enable the autoreset feature of the BC2.
   Status: Not fixed

8. Heart beat packets are not buffered and if a new hearbeat trigger is received before a heartbeat packet has been transmitted it will be discarded.
   Workaround: Keep spacing between heartbeat triggers over 3.5 us.
   Status: Not fixed

Bibliography

[1] UM10204 - I2C-bus specification and user manual, 6th ed., NXP Semiconductors, April 2014.

[2] JESD8-13 - Scalable Low-Voltage Signaling for 400 mV (SLVS-400), JEDEC, October 2001.

[3] "IEEE Standard for Test Access Port and Boundary-Scan Architecture," IEEE Std 1149.1-2013 (Revision of IEEE Std 1149.1-2001), pp. 1–444, May 2013.