ETROC

ETROC2 Reference Manual

FERMILAB-TM-2797-CMS-PPD

ETROC Design Team

Revision 0.41

May 22, 2023

Table 1. Document Revision History

Revision	Date	Reason for Revision Change(s)
Number	Date	Add explanation
0.3	03/23/2023	Re-organized the sections
0.4	05/19/2023	Update the default I2C setting description to match with actual chip default settings

Table of Contents

List of Tables

List of Figures

List of Listings

List of Acronyms

ADC Analog-to-Digital Converter. 69, 83–85

AFC Auto Frequency Calibration. 27

CDR Clock and Data Recovery. 27

DAC Digital-to-Analog Converter. 9, 14, 16, 18, 19

DLL Delay-locked Loop. 28, 30

EBS Event Buffer Status. 36, 37

ETL Endcap Timing Layer. 9

ETROC ETl ReadOut Chip. 9, 14, 79

GRO Gated Ring Oscillator. 9, 14, 23, 26

L1A Level-1 Accept. 9, 14

LGAD Low Gain Avalanche Diodes. 9, 12, 14, 15, 67, 79

LSB Least Significant Bit. 19

PFD Phase and Frequency Detector. 28

PLL Phase-Locked Loop. 9, 14, 26–28

SEE Single Event Effect. 35

TDC Time-to-Digital Converter. 9, 14, 23, 25, 26, 30, 35

TID Total Ionizing Dose. 18

TOA Time Of Arrival. 9, 14, 23, 25, 39

TOT Time Over Threshold. 9, 14, 23, 25, 39

UVM Universal Verification Methodology. 69

VCDLs Voltage-Controlled Delay Lines. 28

VCO Voltage-Controlled Oscillator. 27, 28

WS Waveform Sampler. 79

1 ETROC2 Overview

ETl ReadOut Chip (ETROC) project kicked off in 2018 to develop readout chip for Endcap Timing Layer (ETL) of MTD [1]. Five prototype chips have been developed since then with a progressive development approach. ETROC0 was firstly developed and it prototypes an analog front-end of ETROC [2]. A dedicated Time-to-Digital Converter (TDC) [3] was developed in 2019 and then integrated in ETROC1 which includes a 4×4 pixel matrix with simple readout logic circuits. Some important building blocks were prototyped with two test chips, ETROC_PLL and ET2_Test during 2020 and 2022. ETROC2 is the first full-size full-functionality prototype which includes all the functionalities of the future production version of ETROC.

ETROC2 has a 16×16 pixel matrix to pair with a 16×16 Low Gain Avalanche Diodes (LGAD) sensor with pixel size of $1.3 \times 1.3\ mm^2$. It aims to provide 50 ps per hit time resolution with LGAD. Figure 1 illustrates a simplified block diagram of ETROC2.

In each pixel, the signal current from LGAD is conditioned by a preamplifier to generate an analog pulse on top of the baseline voltage of the preamplifier. The analog pulse is then converted into a digital pulse by a discriminator, which is followed by a low power TDC specifically designed for ETROC [3]. A charge injector is included in each pixel to generate an emulated LGAD signal to facilitate the testing of the full signal process chain. A 10-bit Digital-to-Analog Converter (DAC) is used to generated the threshold voltage of the discriminator. The input of the DAC could be obtained with an optional threshold calibration integrated in each pixel [4]. Users can choose to bypass the calibration and write the DAC directly with the threshold obtained using their own calibration method.

The TDC measures the Time Of Arrival (TOA) and Time Over Threshold (TOT) of the digital pulse, and generates calibration information for each measurement along with the measured TOA and TOT. For each hit, the TOA is measured twice using double time stamping techniques and the difference between the two measurements is used as calibration information for each hit. The output data of TDC is saved continuously into a circular buffer awaiting for Level-1 Accept (L1A). Upon L1A, the TDC data will be readout for each pixel with hit. A coarse map of hits can be made available for every bunch crossing, and the users can define the hits using TOA/TOT/CAL information. This is potentially useful for online occupancy monitoring, luminosity monitoring, as well as potential trigger application. The clocks used in pixels are delivered from peripheral by shielded clock trees based on H-tree structure.

Both the coarse hit map data and the L1A data are processed and formatted in the peripheral global readout stage, and the data can be transmitted off chip with up to two serial link ports, each with programmable data rate at 320/640/1280 Mbps. The ETROC2 is controlled by an I2C-based slow control and a 320 Mbps fast control(also known as fast command). The peripheral includes a Phase-Locked Loop (PLL) and an analog phase shifter for clocks generation. A 2.56 Gsps waveform sampler is integrated to monitor the preamplifier output of one of the pixels. An eFuse is included to store a 17-bit chip ID which is used to keep track of each chip from the wafer to the end of the life. Other supporting functional blocks include a voltage reference generator, an analog temperature sensor and a Gated Ring Oscillator (GRO).

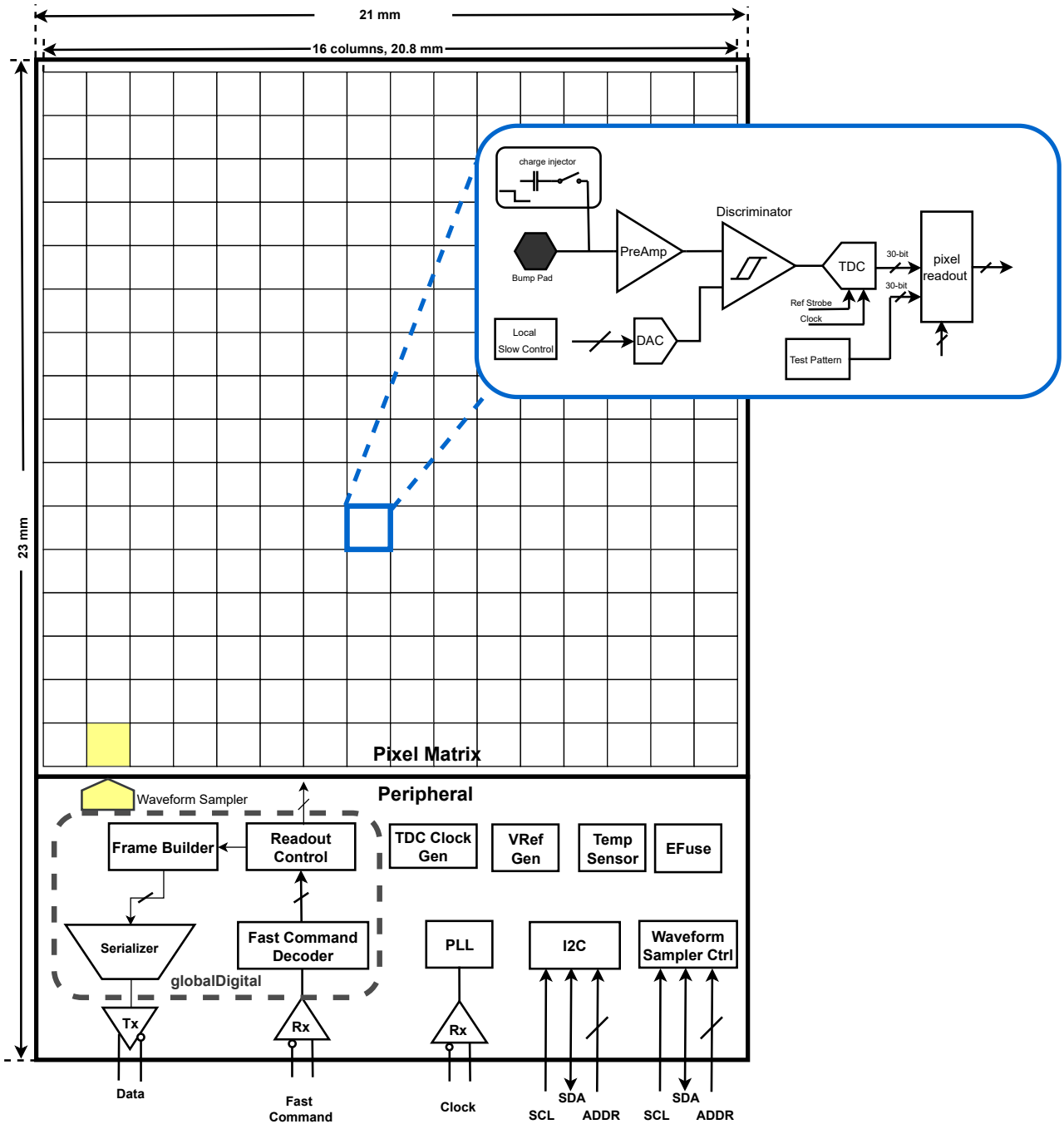


Figure 1. ETROC2 diagram.

ETROC2 is developed with TSMC 65 nm CMOS process (CMN65, RF) and with the metallization of 1p9m-6x1z1u. The dimensions of an ETROC2 die is $21 \times 23\ mm^2$. The thickness of a die is thinned down to around 280 μm.

ETROC2 was taped-out in an engineering run in October 2022. A total of 20 wafers were fabricated in two engineering lots, N60R80 and N60R91. Lot N60R80 includes 14 typical-corner wafers, 8 of them are processed and rest 6 wafers are held before back end processes. Lot N60R91 consists of 6 skewed wafers, comprising of 3 slow-corner wafers and 3 fast-corner wafers. An additional 25 dummy wafers have been booked for module assembly practice.

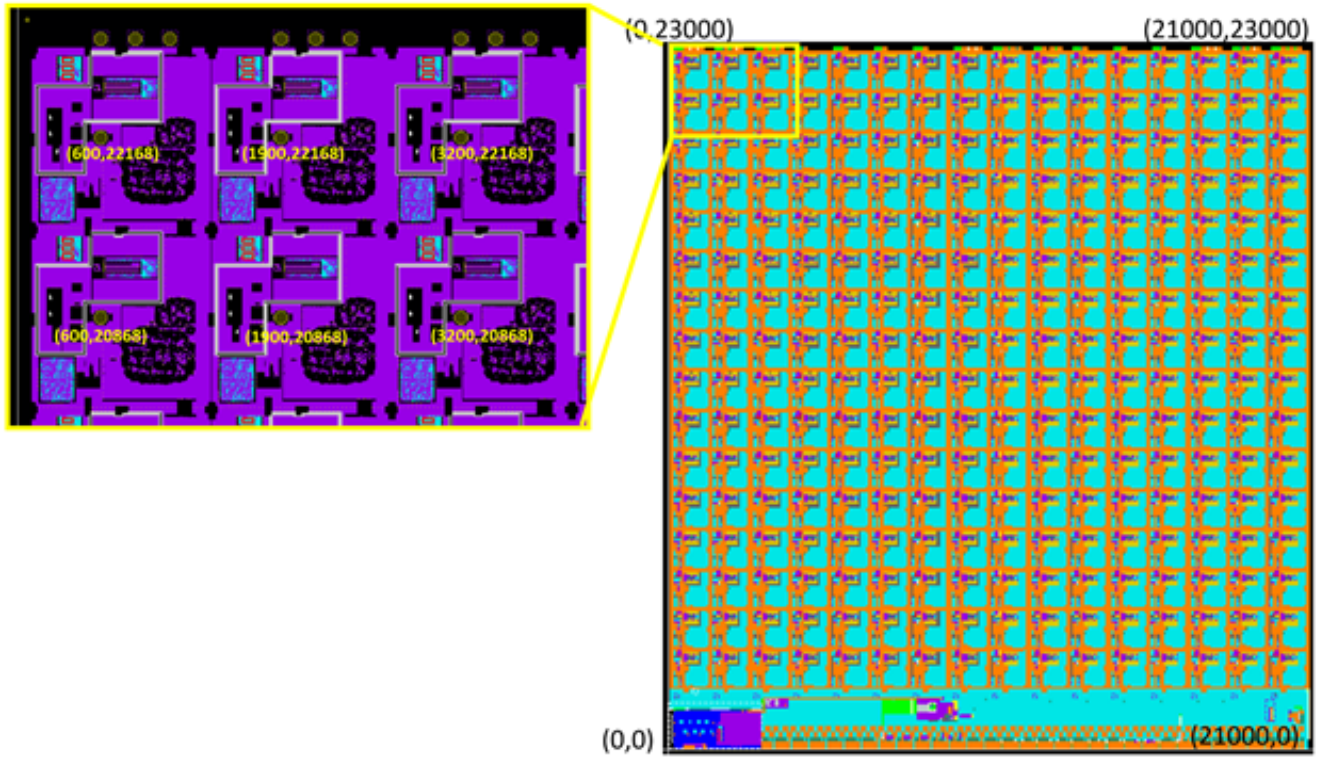


Figure 2. Layout of ETROC2. The unit of coordinates is micron.

2 Sensor Interface

An LGAD sensor can be bump-bonded on the top of the ETROC2 chip. Two types of electrical connections between the ASIC and the sensor, one signal connection for each of the pixels and guard ring connections for the entire sensor, are required for proper operation. In addition, the sensor needs a negative bias voltage to operate. The bias voltage should be isolated carefully to prevent noise pickup.

2.1 Signal Connection

Each pixel in ETROC2 has a bump pad for LGAD signal connection. The bump pad is offset 150 μm towards left relative to the center of the pixel to allow optimized positioning. Figure 2 shows the layout of ETROC2 with bump pad coordinates of some pixels located in the top-left corner. The bump pad,as shown in figure 3,is aluminum octagon with 90 μm opening.

2.2 Guard Ring Connection

There are two rows of guard ring pads in ETROC2, located on the top and bottom of pixel matrix, respectively. It is recommended to use only the top row of guarding pads for initial testing. The guard ring pads at the bottom of the pixel matrix are above the global digital block. Since there might be active cells located underneath these guard ring pads, using these guard ring pads

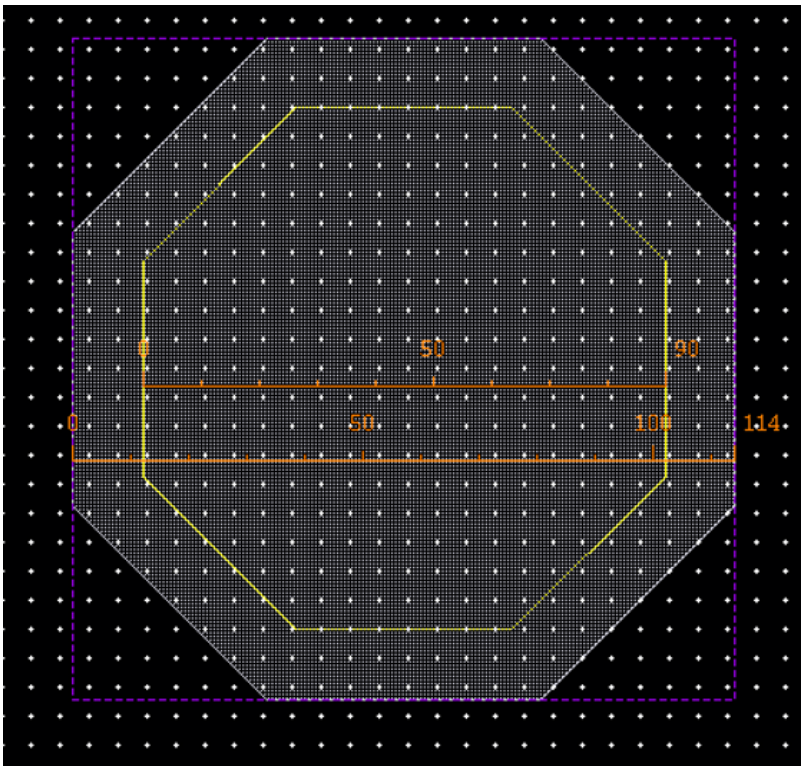


Figure 3. Bump pad used in ETROC2 pixels

may degrade the timing margin of ETROC2 readout.

Electrically, the guard ring pads are connected to the ground of the preamplifier.

3 ETROC2 Operation

ETROC is the ASIC designed for precision timing of in CMS ETL. ETROC2 is the first full-size full-functinality prototype of ETROC, which aims to achive 50 ps per hit time resolution together with LGAD. An ETROC2 includes a 16×16 pixel matrix and a chip peripheral. Figure 1 shows a simplified block diagram of ETROC2.

In each pixel, the signal current from LGAD is conditioned by the preamplifier to generate an analog pules on top of the baseline voltage of the preamplifier. The analog pulse is then converted into a digital pulse by a discriminator, which is followed by a low power TDC specifically design for ETROC [3]. A charge injector is included in each pixel to generate the emulated LGAD signal to facilitate the testing of the full signal process chain. The threshold voltage of the discriminator is generated with a 10-bit DAC. The input of the DAC could be obtained with an optional threshold calibration integrated in each pixel [4]. The calibration can be bypassed in case users want to write the DAC directly with the threshold obtained by their own calibration method.

The TDC measures the TOA and TOT of the digital pulse, and generates calibration information for each measurement along with the measured TOA and TOT. The output data of TDC (data) is read out by the readout logics which is triggered by L1A commands. A coarse map of delayed hits (trigger) are also provided for L1 trigger application. The clocks used in pixels are delivered from peripheral by shielded clock trees based on H-tree structure.

The data and the trigger from the pixel matrix are organized in the peripheral. and then delivered with two serial link ports, each with programmable data rate up to 1.28 Gbps. The ETROC2 is controlled by an I2C-based slow control and a 320 Mbps fast control(fast command). The peripheral includes a PLL and an analog phase shifter for clocks generation. A 2.56 Gsps waveform sampler is integrated to monitor the preamplifier output of one of the pixels(pixel 224). An EFuse is included to store a 17-bit chip ID which is used to keep track of each die from the wafer to the end of the life. Other supporting functional blocks include a voltage reference generator, an analog temperature sensor and a GRO.

The preamp and the discrimantor have been tested in the ETROC0 prototype chip. The details are documented in this paper [2]. The design is good enough that it is used in both ETROC1 and ETROC2 as it is described in the paper.

The preamp, the discmrinator, the TDC and 4×4 clock tree have been prototyped in ETROC1, and the details of the TDC is documented in paper [3]. The TDC design is also directly used in ETROC2 as it is described in the paper (without any change).

The functions of the ETROC2 are elaborated below.

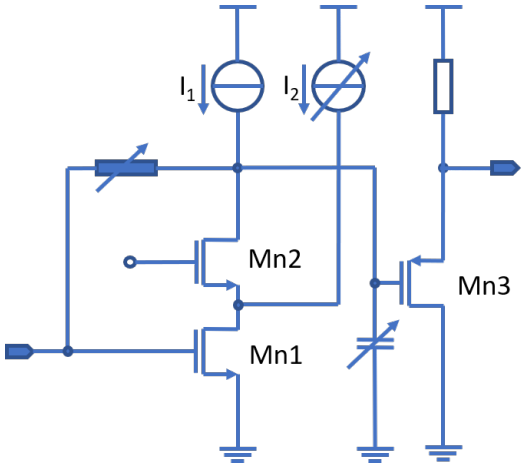


Figure 4. Simplified preamplifier schematic.

RfSel[1:0]	Expected feedback resistance
2'b00	20 kOhm
2'b01	10 kOHm
2'b10	5.7 kOHm
2'b11	4.4 kOHm

Table 2. Preamplifier feedback resistance programming

3.1 Analog Front End

3.1.1 Preamplifier

The preamplifier is a two-stage amplifier, as shown in figure 4 [2]. A cascode amplifier with resistive feedback acts as the first stage, and a source follower as the second stage. The size of each transistor in the preamplifier has been optimized by using input signal from an LGAD simulation. The feedback resistance is programmable to allow adjustment of the fall time, while the bias current is also programmable to allow different trade-offs between power consumption and performance. The load capacitance of the first stage is also programmable to allow optimization of the bandwidth.

An in-pixel I2C register, RfSel[1:0], controls the feedback resistance, as summarized in table 2. A smaller feedback resistance results in smaller gain and smaller preamplifier output noise, which may suggest better timing performance with a large input signal. When signal is small, a large feedback resistance is recommended to get optimized timing performance.

The programmable bias current is controlled by an 3-bit in-pixel I2C register, IBSel[2:0]. The bias current determines the power consumption of the preamplifier, the greater the bias current, the higher the preamplifier power consumption. A larger bias current suggests faster signal and hence a better timing performance.

The load capacitance of the first stage provides the possibility of optimizing timing perfor-

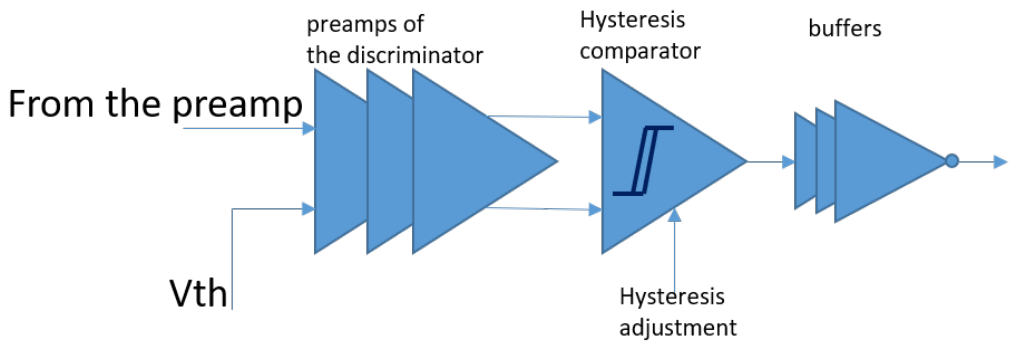


Figure 5. Simplified discriminator diagram.

mance with trade-off between noise and signal speed, which can be programmed with a 2-bit in-pixel I2C register, CLSel[1:0]. However, the test of ETROC0 showed that the best timing performance was obtained at the smallest capacitance. The default setting(smallest capacitance) is recommended.

3.1.2 Discriminator

The discriminator consists of a three-stage preamplifier, a comparator with programmable hysteresis and a buffer, as illustrated in figure 5 [2]. The input pulse seen by the discriminator could be very low in some extreme cases. Therefore, the three-stage preamplifier in the discriminator is employed to amplify these small input pulses to a level where the succeeding comparator can fire. The comparator digitizes the differential input at the crossing point with an adjustable hysteresis ranging from 0 to 1 mV. Finally,the internal buffer is used to relieve the loading pressure from the following circuits.

The hysteresis is controlled with a 4-bit in-pixel I2C register, HysSel[3:0]. It is rare to see multiple pulses trigged by the noise in the ETROC based on the testing experience with ETROC0 an ETROC1. But if it happens, a larger hysteresis can be applied to alleviate it.

3.1.3 Threshold Generation and Calibration

The 10-bit DAC is included in the pixel to generate the threshold voltage of the discriminator. Figure 6 shows a simplified schematic of the DAC. As the baseline of the preamplifier varies with temperature and bias setting, the DAC is designed to cover a range from 0.6 V to 1 V with a step size of 0.4 mV. The DAC employs a resistor-string structure to achieve monotone over its operation range. A noise filter is used to limit the noise of the threshold voltage below 0.1 mV, to minimize the DAC noise contribution to the timing performance.

Figure 7 illustrates a conceptual waveform of the preamplifier and the discriminator, with a small signal and a larger signal. The threshold voltage is usually placed a couple of millivolt above the preamplifier baseline to get the best trade-off between the timing performance and the efficiency [4], as shown in figure 8.

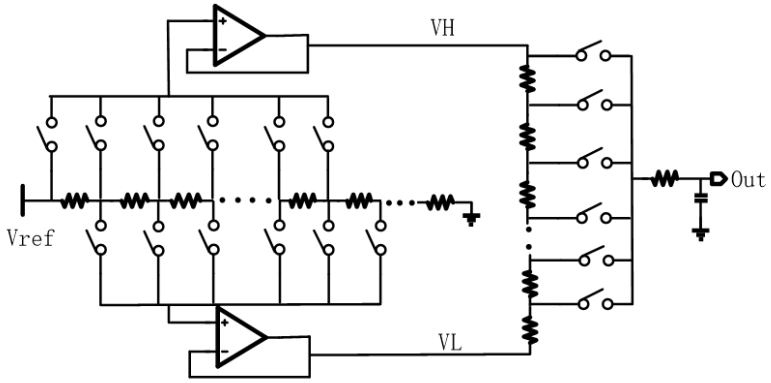


Figure 6. Simplified DAC schematic.

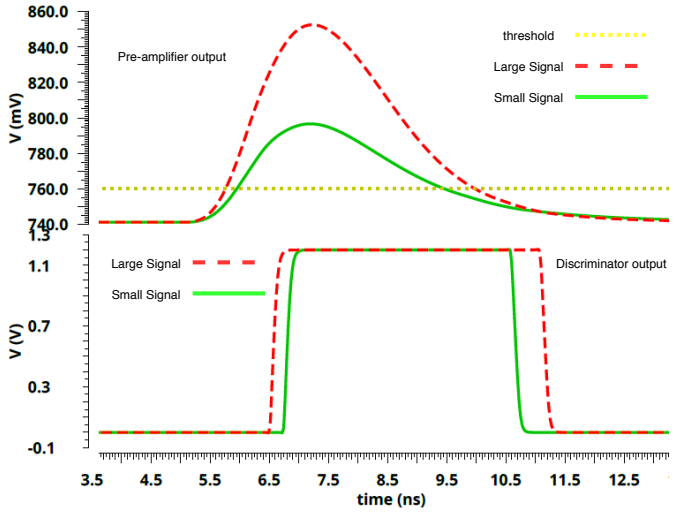


Figure 7. Conceptual waveform of the preamplifier and the discriminator.

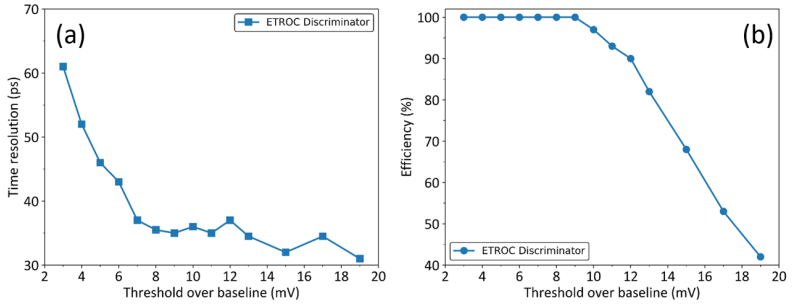


Figure 8. The dependence of discriminator threshold on time resolution (a) and efficiency (b) from simulation.

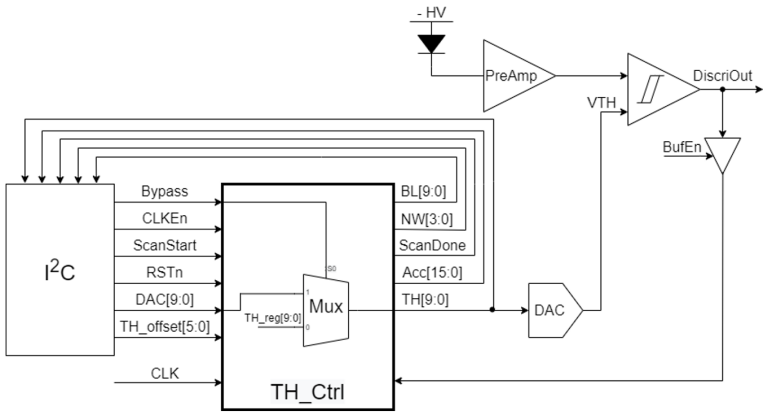


Figure 9. Threshold calibration block diagram.

A dedicated procedure called threshold calibration is needed to define a proper threshold for correct functionality and best performance of the ETROC2. The threshold of the discriminator should be calibrated pixel by pixel due to mismatch of the devices. And the calibration should be performed on a regular basis to alleviate the parameters shift due to operational condition change and Total Ionizing Dose (TID).

The threshold calibration is typically done manually by users using S-curve scan techniques (see for example, ETROC0 paper). This is usually tedious and time consuming, and for that reason, an automatic threshold calibration circuit is designed and integrated inside each pixel with the hope to save users the time and effort for calibration. This is optional and can be disabled if not used. The details of how this automatic calibration work are described in this paper [4]. An digital block, Th_Cal, is integrated in each pixel to perform the in-pixel threshold calibration with minimum external control.

The in-pixel threshold calibration works together with the analog front-end (the preamplifier, the discriminator and the DAC), and the in-pixel I2C block, as illustrated in the block diagram of figure 9.

There are two working modes in TH_Cal, auto threshold calibration mode and the bypass mode.

In the auto threshold calibration mode, a rising edge of ScanStart initializes the threshold calibration procedure. Once the threshold calibration is done, ScanDone become high. The internal clock is enabled at the rising edge of ScanStart and disabled at the rising edge of ScanDone, resulting in a very low the dynamic power when TH_Cal is not operating.

The basic idea of the auto threshold calibration is to measure the average output of the discriminator at the different threshold voltages generated by the DAC, and find out the transition point or the equivalent baseline [5], as illustrated in an example in figure 10. The average output is measured by sampling the discriminator output under the control the 40 MHz clock, and accumulating the samples in a given time window of 32768 clock period (0.8192 ms). The sum of the samples represents the average output of the discriminator.

■ The sum is maximum when the threshold voltage (VTH) is at lower end and the discrimi-

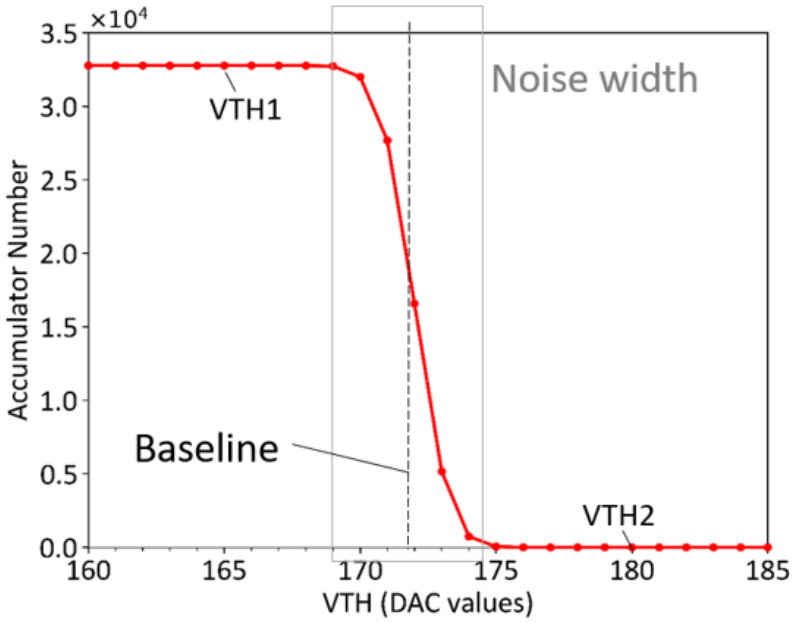


Figure 10. An illustration of the auto threshold calibration.

nator output high.

- The sum is zero when VTH is at high end and the discriminator output low.
- Transition region width represents the noise amplitude, and the equivalent baseline locates the middle of the transition region.

To get an efficient auto threshold calibration, a 10-bit binary successive approximation is first performed to find the equivalent baseline. Then a 25-step linear sweep is used to identify the noise width which is defined as width of the transition region ($0 < sum < 32768$) in number of DAC Least Significant Bit (LSB).

After the auto threshold calibration is done, the DAC input is set to:

$$TH[9:0] = BL[9:0] + TH\_offset[5:0], \tag{1}$$

where TH[9:0] is the DAC input, BL[9:0] is the equivalent baseline from auto threshold calibration procedure, and TH_Offset[5:0] is a relative threshold applied on top of the equivalent baseline. The users can specify TH_offset[5:0] as needed to trade-off between timing performance and efficiency.

The other mode of Th_Cal is the bypass mode in which Th_Cal passes DAC[9:0] to TH[9:0] without any manipulation. The users can use this mode when the threshold is obtained through an off-chip threshold calibration. It is recommended that for the initial testing of ETROC2, users use this mode first.

Note that the users can measure the average of the discriminator output at the applied DAC input in the bypass mode. This feature is useful when diagnosing the ETROC2 or performing "a semi-auto threshold calibration".

The procedure of auto threshold calibration is summarized below.

1. set "Bypass" low.
2. set "BufEn_THCal" high.
3. set "TH_offset" a proper value.
4. reset "Th_Cal":
   (a) set "RSTn" low.
   (b) enable clock by issuing a rising edge of ScanStart.
   (c) set "RSTn" high.
5. launch auto threshold calibration by issuing a rising edge of ScanStart.

Note that a proper TH_offset might be obtained with trails. A recommendation is that TH_offset should be larger than $NW[3:0]/2$ to avoid noise triggering.

A snippet of Verilog code for in-pixel threshold calibration is provided in list 1 as an example.

Listing 1: A Verilog example for in-pixel threshold calibration.

```
/*
three actions are combined
1. reset the in-pixel threshold calibration block
2. issue a ScanStart edge at the same time with broadcast
3. set BufEn_THCal high
*/
#1_000
pixel_x = 4'hf;
pixel_y = 4'hf;
pixelConfig = 8'h12;
pixelConfigAddrIn = 5'h03;
pixel_bc(pixel_x, pixel_y, pixelConfig, pixelConfigAddrIn);

// release the reset and set the ScanStart to low with broadcast
#1_000
pixel_x = 4'hf;
pixel_y = 4'hf;
pixelConfig = 8'h03;
pixelConfigAddrIn = 5'h03;
pixel_bc(pixel_x, pixel_y, pixelConfig, pixelConfigAddrIn);

// initialize the in-pixel threshold calibration process with broadcast
#1_000
pixel_x = 4'hf;
pixel_y = 4'hf;
pixelConfig = 8'h13;
pixelConfigAddrIn = 5'h03;
pixel_bc(pixel_x, pixel_y, pixelConfig, pixelConfigAddrIn);
```

Table 3 and table 4 summarize the configuration of TH_Cal block for different working mode.

Input Signals	Values
Bypass	1'b0
CLKEn	1'bx
DAC[9:0]	10'dx
TH_offset[5:0]	user specified relative threshold
RSTn	active low
ScanStart	a rising edge

Table 3. Auto threshold calibration mode configuration.

Input Signals	Values	
	Passing DAC	Average output meas.
Bypass	1'b1	
CLKEn	1'b0	1'b1
DAC[9:0]	DAC value	
TH_offset[5:0]	6'bx	
RSTn	1'bx	active low
ScanStart	1'bx	a rising edge

Table 4. Bypass mode configuration.

3.1.4 Charge Injection

The charge injector is employed to facilitate ETROC2 testing. The charge is generated by applying a known voltage step(QV) to a terminal of a capacitor(Cq) connected to the input of the preamplifier on the other terminal, as shown in figure 11. The details are described in this paper [5].

The charge amplitude is defined by a 5-bit in-pixel I2C register, QSel[4:0], as QSel[4:0]+1 fC. The error of the generated charge is up to $\pm 15\%$ due to process variation and circuits non ideal effects, but it is good enough as a test facility. The charge injector is enabled when QInjEn is high by closing the switch between the capacitor and the preamplifier.

The charge injector's input pulse is generated in ETROC2 peripheral and delivered to each pixel with a H-tree based clock tree. Figure 12 shows an example of charge injection waveform. When a charge injection command (69) is decoded, a charge injection pulse is generated from the 40 MHz PLL clock, with a delay of t_{dQInj} with regard to the rising edge of the 40 MHz PLL clock, where t_{dQInj} is a delay determined by an I2C register, chargeInjectionDelay[4:0], as given in equation 2.

$$t_{dQInj} = chargeInjectionDelay[4:0] \cdot 25ns/32, \tag{2}$$

, assuming 25 ns period of the 40 MHz PLL clock.

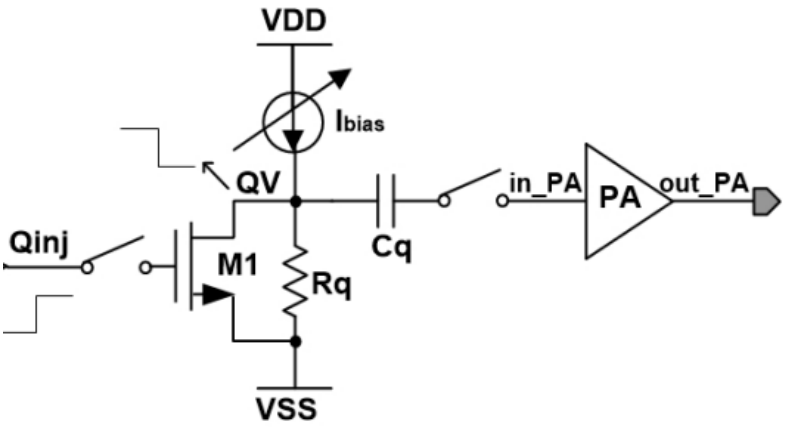


Figure 11. Simplified schematic of charge injection.

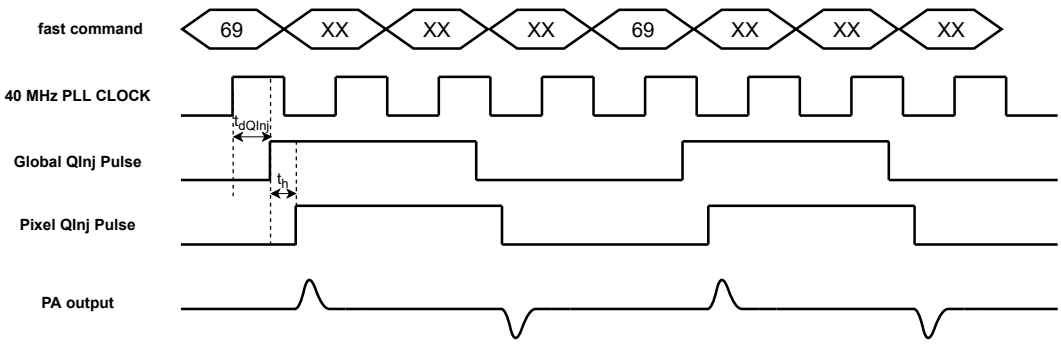


Figure 12. Charge injection waveforms.

3.2 TDC

3.2.1 TDC

The TDC is based on a free-running GRO. This structure is originally developed in FPGAs, and ETROCs employ it to solve our challenges including lower power consumption, consistency and issues related to large die size.

Figure 13 shows a simplified schematic of the TDC core [3]. The GRO consists of 63 NAND-based delay cells. One of the delay cells is used to enable or disable the oscillation at Start port. Figure 14 shows the waveforms of the TDC core key signals. Here are explanation of signals in figure 14.

1. CLK40M: the 40 MHz TDC clock, in sync with the 40 MHz readout clock.
2. CLK320M: the TDC reference strobe which are two pulses derived from a 320 MHz clock.
3. PULSE: the TDC input or the discriminator output
4. START: a TDC internal signal used to control the GRO, related to PULSE and the the reference strobe (CLK320M)
5. TOACLK: a TDC internal signal used to latch the GRO status for TOA measurement, related to the first rising edge of the reference strobe (CLK320M)
6. TOALATCH: a TDC internal signal used to latch the GRO status that is previously latched with TOACLK, related to TOACLK. see reference [3] for details
7. TOTCLK: a TDC internal signal used to latch the GRO status for TOT measurement, related to the trailing edge of PULSE

The TDC core works as follows.

The GRO is enabled and start to oscillate at rising edge of the discriminator output pulse. The period of the oscillation is expressed as:

$$T_{GRO} = 2 \cdot 63 \cdot t_{bin} \tag{3}$$

, where t_{bin} is the bin size of the TDC (delay of the NAND).

The ripple counters in figure 13 record the number of turns the signal goes in the GRO. The status of the GRO and the ripple counters are latched three times, at the trailing edge of the discriminator output pulse as $R_{tailing}$, at the first rising edge of the reference strobe as R_{RS1}, and at the second rising edge of the reference strobe as R_{RS2}.

$R_{tailing}$, R_{RS1} and R_{RS2} are then encoded to get TOT code, TOA code and CAL (calibration) code.

The encoded TDC data will show up in a data word in the ETROC2 data frame, as shown in figure 15. More details of the ETROC2 data frame can be found in a later section.

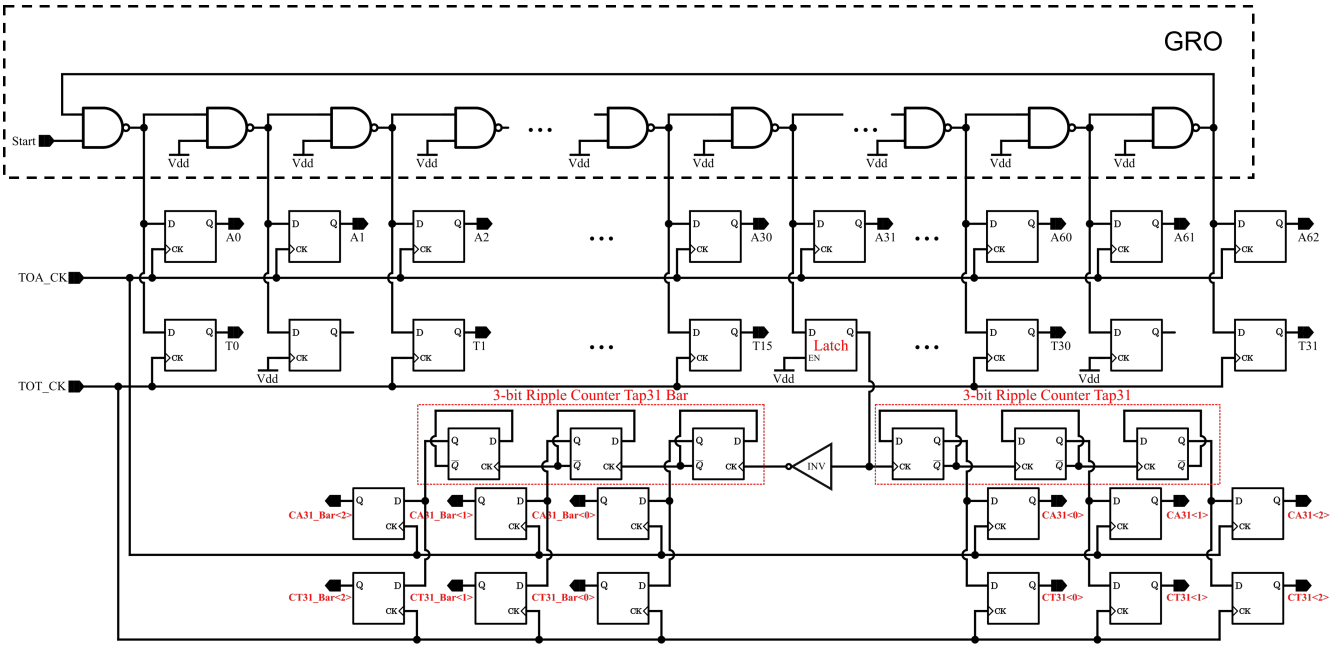


Figure 13. Simplified schematic of the TDC core.

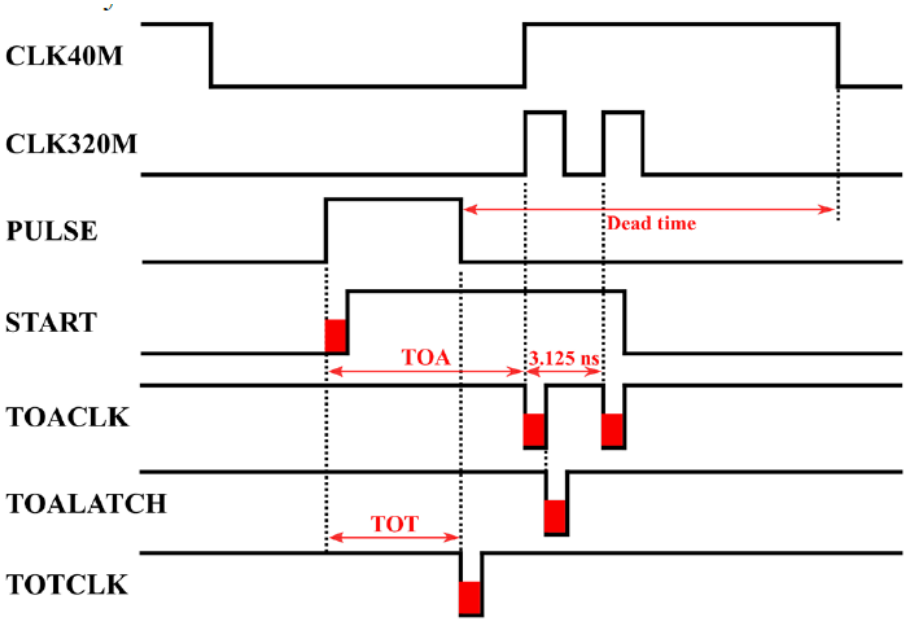


Figure 14. Waveforms of the TDC core key signals.

Figure 15. Data word format.

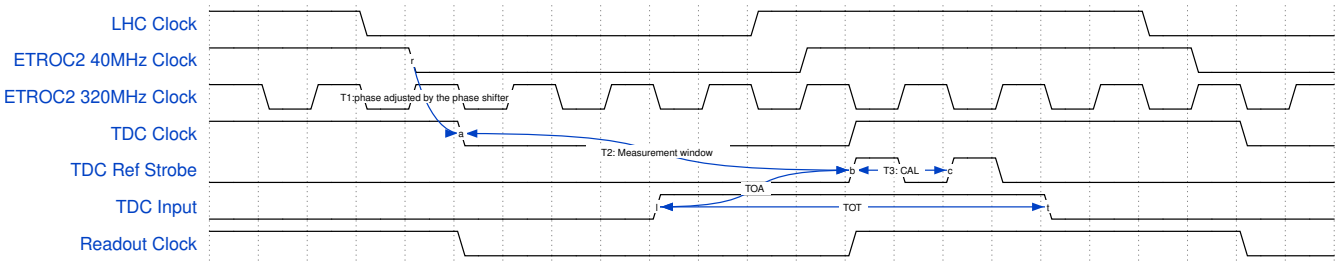


Figure 16. waveforms of the TDC.

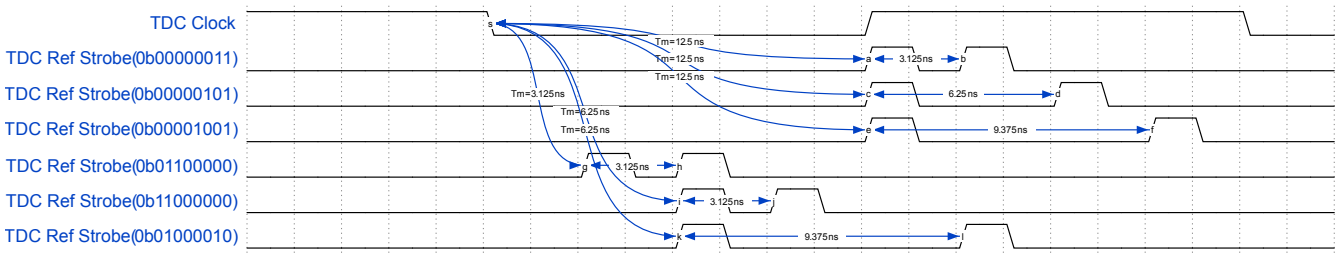


Figure 17. Examples of the TDC reference strobe programming.

Figure 16 shows the TDC waveforms from the users point of view. The TDC measures TOA, TOT, and gives a calibration code along with each measurement. The true time of the measurement can be constructed with equation 4, 5 and 6.

$$t_{bin} = T3/Cal_{CODE} \tag{4}$$

$$TOA = t_{bin} \cdot TOA_{CODE} \tag{5}$$

$$TOT = (2 \cdot TOT_{CODE} - floor(TOT_{CODE}/32)) \cdot t_{bin} \tag{6}$$

$T3$ is time difference of two rising edges of the reference strobe.

Note that the TDC has a measurement window in which the discriminator output pulses can be measured. The discriminator output pulses locating outside of the measurement windows will be ignored.

The measurement window starts from the falling edge of the TDC clock and ends at the first rising edge of the reference strobe. The users can program the reference strobe with a peripheral I2C register, RefStrSel[7:0], to adjust the measurement window. Figure 17 illustrates some examples of the TDC reference strobe programming.

T3 in equation 4 can be set by RefStrSel[7:0] as well.

Other features implemented in the TDC are listed below.

1. auto reset: automatically reset controller for every clock period, a feature for debug only. The related I2C register is autoReset_TDC.

2. enable or disable the TDC conversions. The related I2C register is enable_TDC.

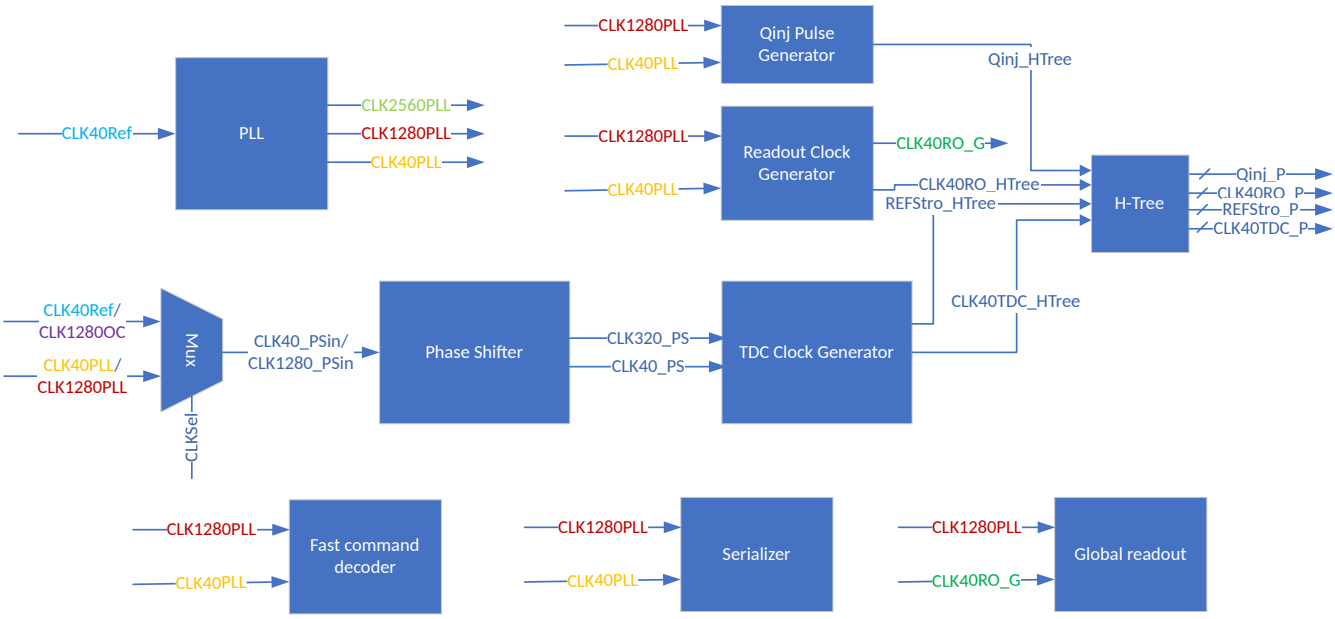


Figure 18. ETROC2 clock generation and delievery diagram.

3. bubble tolerance: A bubble is discontinuities is the raw TDC code from the GRO generated by noise or metastability. The TDC encode is designed in such a way that some bubbles can be tolerated. The related I2C register is level_TDC. This is a feature for debugging only.

4. reset the TDC encoder: These is no need to reset the TDC encoder usually. The reset is for debugging only. The related I2C register is resetn_TDC.

5. test mode: The TDC has a built-in test mode which allows self test. In the test mode, a fixed test pulse is applied to the TDC input for every 25ns. The related I2C register is testMode_TDC.

3.2.2 TDC GRO

The ETROC2 includes an individual GRO (TDC core, 13) in the peripheral for process monitoring purpose. The GRO can be enabled by setting high to the I2C register bit GRO_Start.

The output of the GRO is followed by a frequency divider of 1/256. The users will expect a clock of about 1.72 MHz on the GRO's output pad. The GRO's output pad is a bump pad at the bottom right of the die, which is accessible through a probe.

3.3 Clock Generation and Delivery

Figure 18 shows a block diagram of the ETROC2 clock generation and delivery system. The ETROC2 takes a 40 MHz clock as the reference clock of a PLL clock generator. Three main clocks generated by the PLL are CLK2560PLL, CLK1280PLL and CLK40PLL.

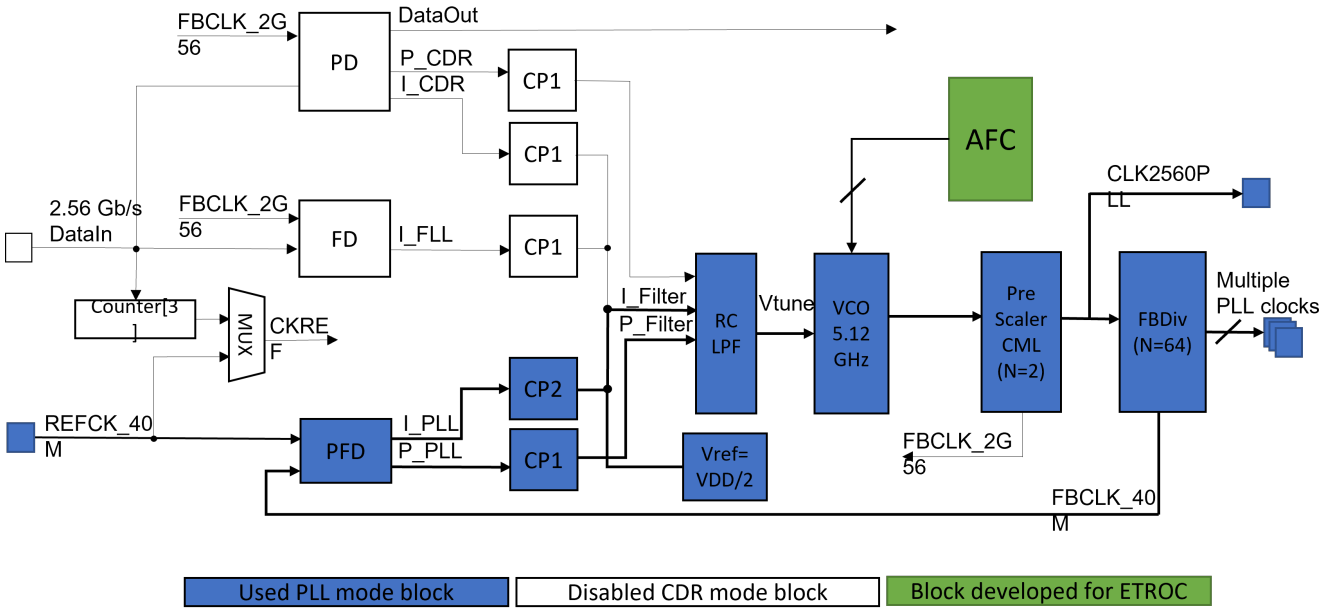


Figure 19. Simplified PLL clock generator block diagram.

As shown in Figure 18, the signal CLK2560PLL is a 2.56 GHz clock used by the waveform sampler. The clock signals CLK1280PLL and CLK40PLL are 1.28 GHz and 40 MHz, respectively, which are used in the TDC clock generation, the charge injection pulses generation, the readout clock generation, the fast command decoding, and the data serialization.

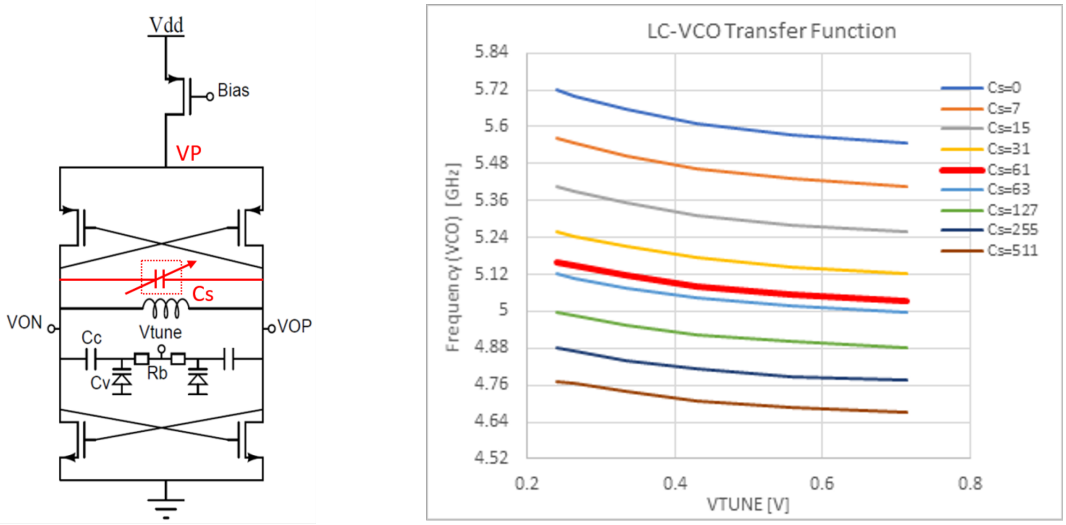
3.3.1 PLL

The PLL in the ETROC2 is migrated from the clock generator block of lpGBT [6]. Involving metal stack change and some new blocks for calibration, the PLL has been validated in a dedicated test chip [7] before being integrated into the ETROC2.

Figure 19 shows a simplified block diagram of the PLL. As illustrated in the diagram, the Clock and Data Recovery (CDR) function of the original design is disabled.

The PLL includes a Voltage-Controlled Oscillator (VCO) which features low jitter and low irradiation sensitivity. Nominally working at 5.12 GHz, the LC based VCO has a narrow operating range which may shift significantly due to process variation. To make sure that the VCO works at 5.12 GHz under any process corner and operating conditions, a procedure named Auto Frequency Calibration (AFC) is needed to calibrate its operating range before being used in the PLL. An AFC block is designed to perform the auto frequency calibration.

The VCO includes a programmable capacitor so that the operating range can be shifted digitally, as shown in figure 20. During the calibration, two counters driven by the reference clock and the feedback clock(the VCO clock divided by 128), respectively, are racing under the control of a binary search algorithm, and the results determine the setting of the programmable capacitor. The calibration is needed only one time at power up. Once the calibration is done, the PLL will lock to the reference clock in tens of micro-seconds.



(a) (b)

Figure 20. A simplified schematic of the LC-VCO (a), and the transfer function of the LC-VCO (b).

An instant locking signal is reported by the Phase and Frequency Detector (PFD) to indicate the instant locking status of the PLL. A locking detector is employed to filter the instant locking signal.

The locking detector is a parameterizable state machine which is depicted in figure 21 [6]. The locking threshold value of the instant locking low pass filter can be set by an I2C register, lfLockThrCounter[3:0]. The number of cycles is set to $2^{lfLockThrCounter[3:0]}$. Similarly, the relocking and unlocking threshold values are set by two registers, lfReLockThrCounter[3:0] and lfUnLockThrCounter[3:0], respectively.

There is an I2C register, pllUnlockCount, which is used to monitor the number of unlocking have happened since power up.

The VCO calibration and locking detecting are part of power up sequence and under control of a state machine at power up. The details are elaborated in the relevant sections.

3.3.2 Phase Shifter

The phase shifter block generates two clocks at frequency of 40 MHz and 320 MHz, respectively, with programmable phase. The time resolution of the phase shifter is about 97.6 ps (1000/8/1.28 ps), and the phase rotation can be up to 360 degree.

As shown in figure 22, the phase shifter is implemented as two functional sub-blocks: a digital coarse phase shifter, and a fine phase shifter, based on a Delay-locked Loop (DLL). Two Voltage-Controlled Delay Lines (VCDLs) are implemented based on one DLL to save the area and power consumption.

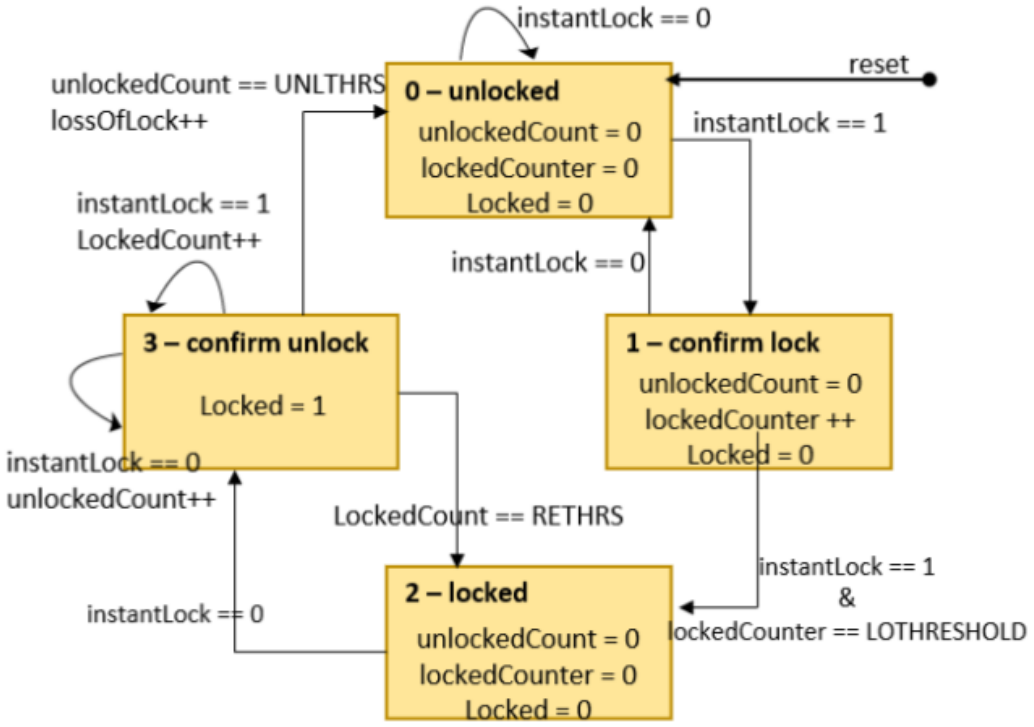


Figure 21. State diagram of the locking detector.

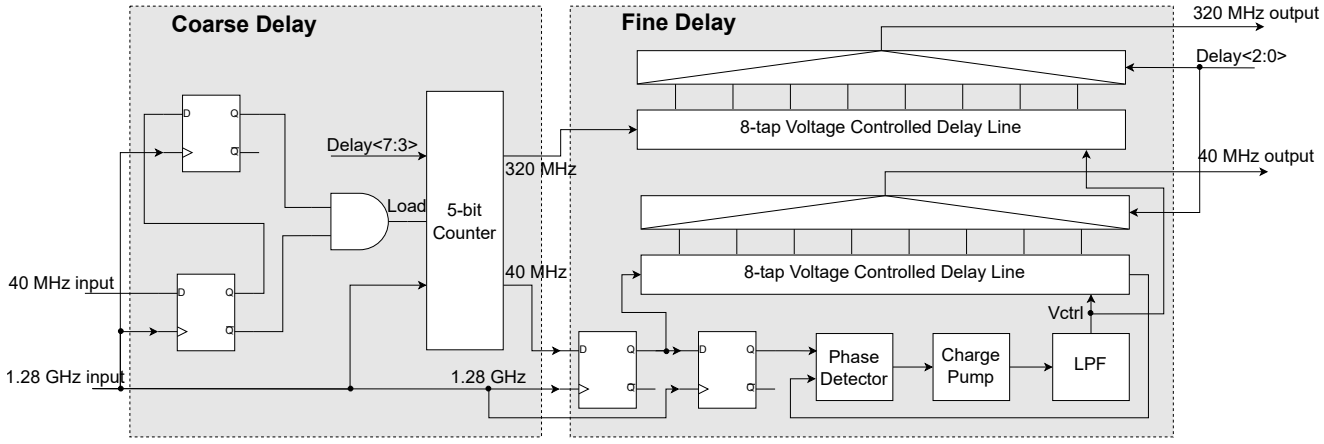


Figure 22. Block diagram of the phase shifter.

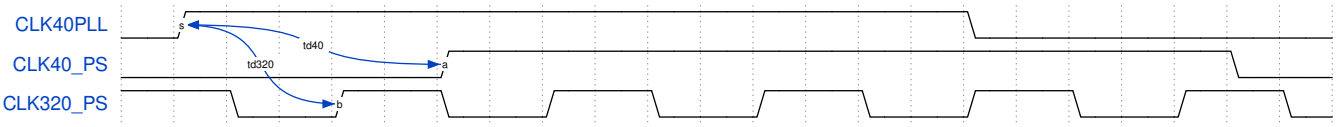


Figure 23. Timing diagram of the phase shifter.

Figure 23 illustrates the functionality of the phase shifter in a timing diagram. Two output clocks have programmable delays with regard to the 40 MHz input clock, CLK40PLL(40 MHz PLL clock) by default. The programmable delay of two output clocks are represented in equation 7 and equation 8, respectively.

$$t_{d40} = (19 + (1 + PS\_PhaseAdj[7:0])/8) \cdot T_{1280} + t_{mux} \tag{7}$$

$$t_{d320} = (19 + (1 + PS\_PhaseAdj[2:0])/8) \cdot T_{1280} + t_{mux} \tag{8}$$

PS_PhaseAdj is an I2C register which controls the programmable delay. t_{d40} and t_{d320} are the delay of 40 MHz clock and 320 MHz clock, respectively. T_{1280} is the period of the 1.28 GHz clock in pico-second, typically 781.25 ps. And t_{mux} is a constant time related to an internal multiplexer, typically 150 ps.

After power up, the DLL in the phase shifter should be locking automatically. If the phase shifter doesn't work as expected, the users can reset the DLL manually by setting PS_CapRst to high and setting it back to low after 1 us.

3.3.3 TDC Clock Generator

The two output clocks of the phase shifter are used to generate the 40 MHz TDC clock and the TDC reference by the TDC clock generator.

The 40 MHz TDC clock is simply a copy of the 40 MHz phase shifter clock. The TDC reference strobe is described in TDC section.

3.3.4 Readout Clock Generator

The readout clock generator is based on two digital phase shifters. Two output clocks are both 40 MHz and used for global readout and pixel readout. The phase and duty-cycle of the clocks are programmable independently.

Figure 24 illustrates the functionality of the readout clock generator. The parameters in figure 24 are defined in the equations below:

$$t_{dg} = readoutClockDelayGlobal[4:0] \cdot T_{1280} \tag{9}$$

$$t_{dp} = readoutClockDelayPixel[4:0] \cdot T_{1280} \tag{10}$$

$$t_{wg} = readoutClockWidthGlobal[4:0] \cdot T_{1280} \tag{11}$$

$$t_{wp} = readoutClockWidthPixel[4:0] \cdot T_{1280} \tag{12}$$

, where T_{1280} is the period of the 1.28 GHz clock in pico-second, typically 781.25 ps.

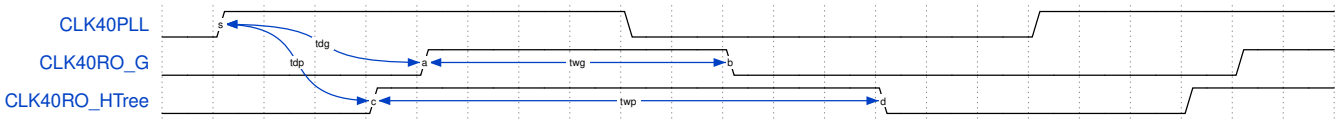


Figure 24. Timing diagram of the readout clock generator.

Figure 25. Cross section of H-tree in ETROC2.

3.3.5 H-tree

The ETROC2 employs H-tree structures to deliver some critical clocks/signals to the pixel matrix.

A unshielded H-tree is used in ETROC1 to delivery clock from periphery to the 4×4 pixel matrix. In ETROC2, the H-tree structure used ETROC1 is scaled up to the 16×16 pixel matrix, and shielding is applied to alleviate interference.

Figure 25 shows the cross section of the H-tree in ETROC2. As shown, M5 is the routing layer in the H-tree, while M3 and M7 are used to provide shielding.

Various inverter types and layer choices have been studied to optimized the H-tree. Figure 26 summaries the simulation results with different choices.

Four clocks/signals are delivered from the peripheral to each pixels by H-trees, including the pixel readout clock, the TDC 40 MHz clock, the TDC reference strobe and the charge injection pulse.

Figure 27 illustrates the structure of an H-tree in the ETROC2, which is implemented based on inverter chains. The red circles in the figure 27 highlight the location of the inverters associ-

Buffer size	INVD4_LVT_ELT	INVD4_LVT_ELT	INVD2_LVT_ELT	INVD2_LVT_ELT
Trace Layer	M6	M6	M6	M5
Trace Width (um)	1	1	1	1
Shielded Layer	-	M5/7	M5/7	M3/7
Delay (ns)	1.2	5.1	9.0	3.7
Skew (ps)	6.7	28.3	27.1	10.4
Jitter (ps)	0.75	0.25	-	-
Duty Cycle (%)	50.1	49.3	50.4	50.2
Rise Time (ps)	-	92	160.88	88.75
Fall Time (ps)	-	92	183.98	87.33
Power (mW)	7.4	41.5	32.5	13.5

Figure 26. ETROC2 H-tree simulation summary.

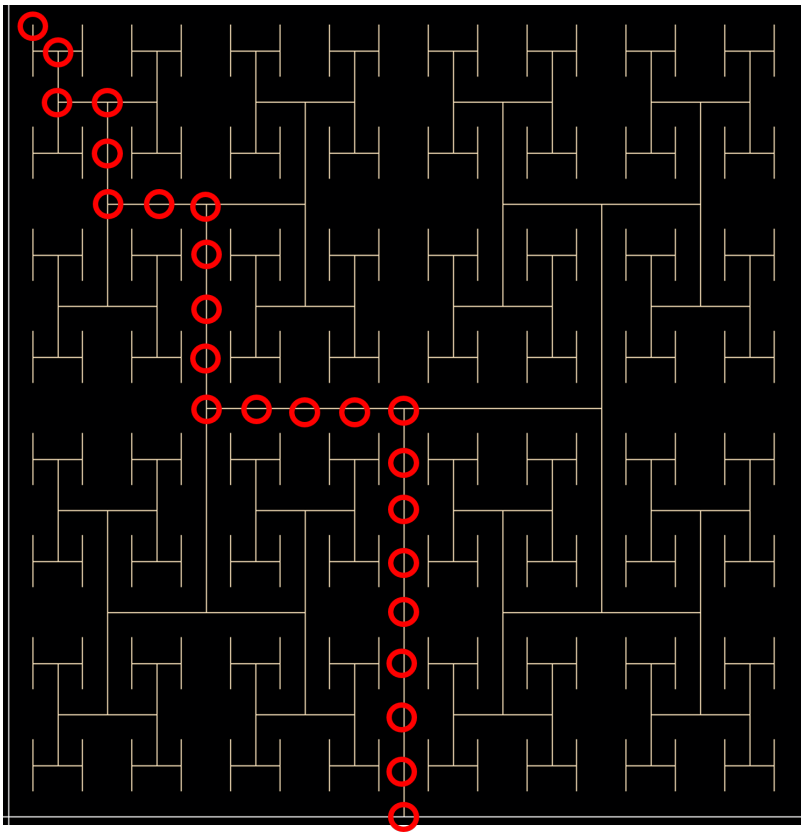


Figure 27. H-tree structure

ated with pixel 255. The delays introduced by the H-Trees are designed in such a way that (1) for all the pixels, they are almost the same for the readout clock and the charge injection pulse, and (2) there are natural offsets in a four-pixel cluster for the TDC clock and the TDC reference strobe. Figure 28 shows the typical delay of the H-trees for the four clocks/signal. Note that these numbers vary with process and operation conditions.

Figure 29 shows an example of ETROC2 timing diagram which highlights some clocks/signals in the peripheral and pixel. Note that the output of the TDC will be update at rising edge of the TDC clock for the input in the preceding cycle. The TDC will hold its output if the preceding cycle has no input pulse.

3.4 Readout

The readout packages L1 selected TDC data from hitted pixels into a data frame for each bunching crossing and transmits it off-chip. The readout design is new for ETROC2 and not silicon proven before. However, it has been extensively studied and verified with ETROC2 emulator [8]. The packaged data frames are sent out via two serial ports. As can be seen in figure 30, due to a dedicated switching network connecting a global readout module and pixels, the TDC data are pop out to the global readout orderly and packaged into data frames. Figure 31 shows that the data is continuously written into a circular buffer and awaits the L1A signal.

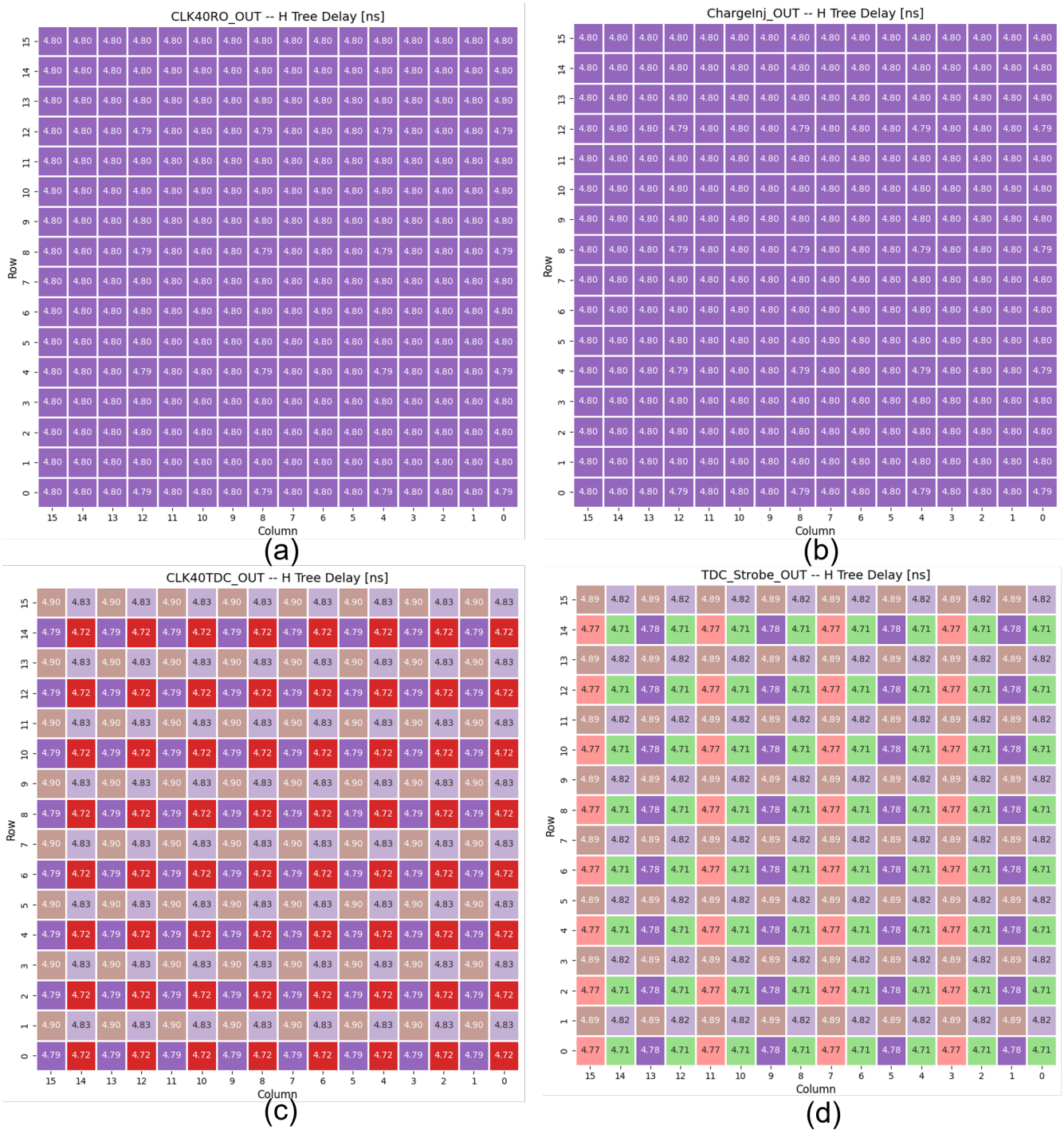


Figure 28. Color map of the H-Trees' typical delay: (a)pixel readout clock, (b) charge injection pulse, (c) TDC clock, (d) TDC reference strobe.

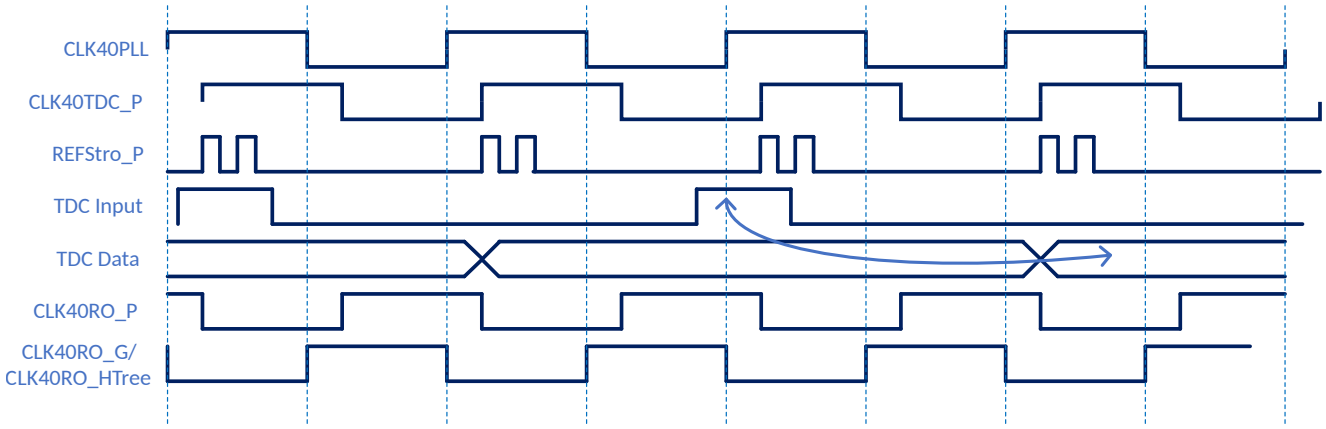


Figure 29. An example of ETROC2 timing diagram.

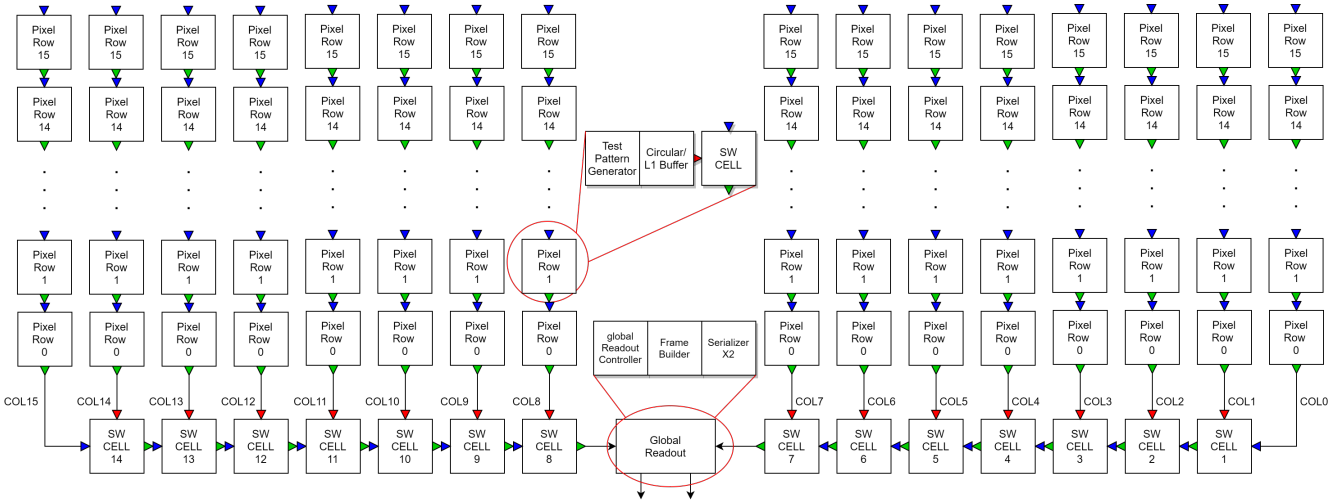


Figure 30. The overall readout structure.

When a L1A signal occurs, the read pointer goes back to the corresponding memory address for that bunch crossing (BC) and reads out the data. The offset of the memory address is programmable. In CMS application, the CMS L1A latency is about 12.5 us, which corresponds to about 500 bunch crossings. The data readout from circular buffer are written into L1 event buffer and then output to global readout. In the test mode, the TDC data is replaced with an on-pixel data pattern so that user is able to verify the data readout without real TDC data input. The TDC hit flag of the pixels are directly output to the global readout, bypassing the circular buffer and L1 event buffer. These flags are then merged into a coarse hit map for user trigger or monitoring purpose.

3.4.1 Data frame definition

If there is no L1A signal, the output block will send filler continuously. When L1A is true, the selected data are packaged into data frame including header, data word and trailer. The filler, header, trailer and data word are all 40 bits length and each data word corresponds to one TDC

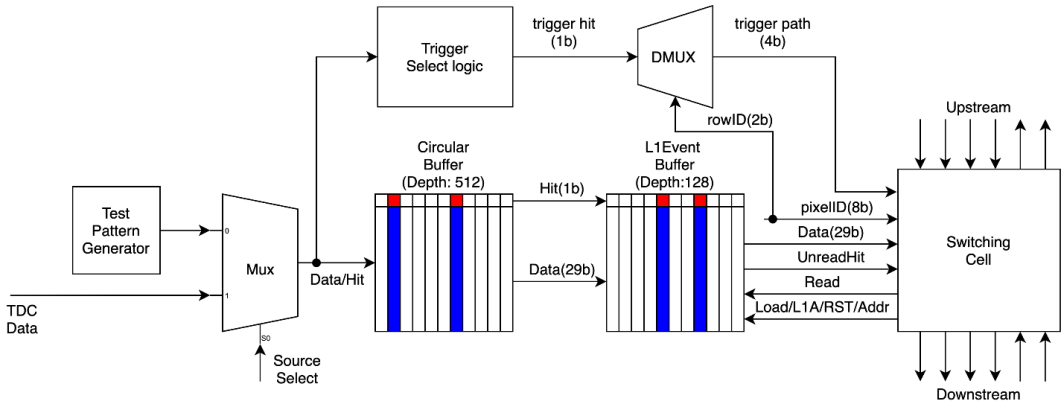


Figure 31. The on-pixel readout design.

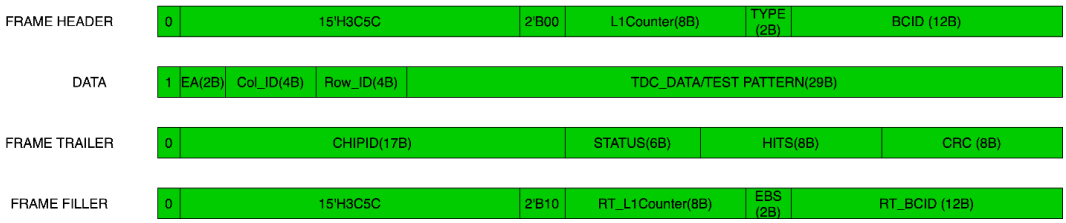


Figure 32. Raw data frame.

data of a hitted pixel in one bunching crossing, as illustrated in figure 32. If the first bit of the word is '1', it is a data word. Otherwise, it is a header,trailer or filler. A regular data frame with a hit comprises a header, data, and a trailer. With a total of 256 pixels in an ETROC chip, the number of data words in a data frame can range from 0 to 256. If there is no hit for a given L1A signal, the data frame will consist only of a header and trailer.

Each data word has a 4-bit column id and 4-bit row id to determine its source pixel. Beside the 29 bits of TDC data and 8 bits of pixel id, there is a two bits of EA for potential Single Event Effect (SEE) or memory error detection. As can be seen in figure 33, when TDC data (or test pattern) is written in circular buffer with Hamming code and is decoded in the frame builder. The two bits of EA is the error status of Hamming code decoder in the frame builder. If EA is '00', it means there is no error detected. If EA is '01', it means there is one bit error has been detected and corrected. If EA is '10', it means there is 2 or more bits error and the Hamming code can not correct it.

The header and filler are always started with 16 bits of code 0x3C5C. If the 16 bits of code "0x3C5C" is followed by '00', it is a header. If it is followed by '10', it is a filler. The header has a two-bit data type to describe the type of the data frame. The detailed definition is in table 5. The

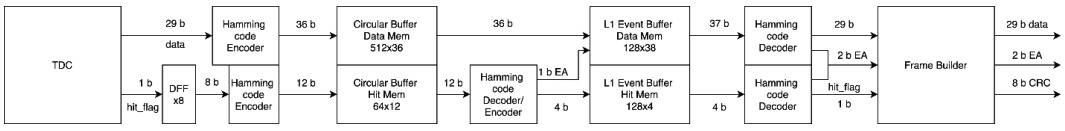


Figure 33. Hamming code in the readout.

Type	Meaning
00	Regular data format
01	Random test pattern
10	Counter test pattern
10	Reserved

Table 5. The type of data frame definition

Status Bit	Define	Meaning
0	L1 Event buffer overflow	L1 event buffer is full, event may lose
1	L1 Event buffer almost full	L1 Events >= 96 (can be user defined)
2	L1 Event buffer half full	L1 Events >= 64
3	Data error occur but corrected	error occurred, corrected
4	Data error occur but not corrected	2 or more bits error occurred, not corrected
5	Reserve	−−

Table 6. The definition of status in a trailer

header includes a 12-bit BCID to identify the event BCID. The L1 counter is an 8-bit counter to count the L1A signals when the event is accepted.

A full data frame is always terminated with a trailer. The trailer is always started with '0' and 17 bit chip id. We will avoid using chip id 01111001011100XX. By avoiding to use these 4 chip id number, the trailer is not started with 16 bits of code 0x3C5C, different from the filler and header. The 6-bit of status in the trailer describe the status when this event is accepted. The status definition is shown in table 6. It is not expected to receive a broken data frame without a trailer even if the bandwidth of the data is beyond the data taking bandwidth. When the occupancy is too high and the L1 event buffer is almost full, the readout switches to quick read mode in which the data words are not included, and this data frame is tagged as an overflow event in the status field. The hits count presents 8-bit count of the data words in this data frame. The 8-bit CRC code is to verify the data integrity of the whole data frame from header to the trailer.

It is a common practice to inject filler words in the data output stream when there is no trigger selected data. These fillers are used to maintain the data transmission rate and to provide a continuous data flow to the downstream systems. The filler starts with a 16-bit code of 0x3C5C and is followed by '10' to indicate that it is a filler word. The filler dumps the status of readout when it is sending out for diagnostic purpose. A 2-bit Event Buffer Status (EBS) is the 2 MSB of number of hits left in event buffer. The definition of the EBS bits are shown in table 7.

The readout can be configured into the test mode to read out test pattern. There are two type of the test patterns are defined. The event type in the frame header describes if the current data frame is data or one of the test patterns (see table 8). The detailed definition the test pattern and regular data are shown in figure 34.

EBS	Meaning
00	Hits in buffer is less than 32
01	Hits in buffer is between 32 to 63
10	Hits in buffer is between 64 to 95
10	Hits in buffer is more than 96

Table 7. The definition of EBS.

Type	Meaning
00	Regular data format
01	Random test pattern
10	Counter test pattern
10	Reserved

Table 8. The definition of event type in the header.

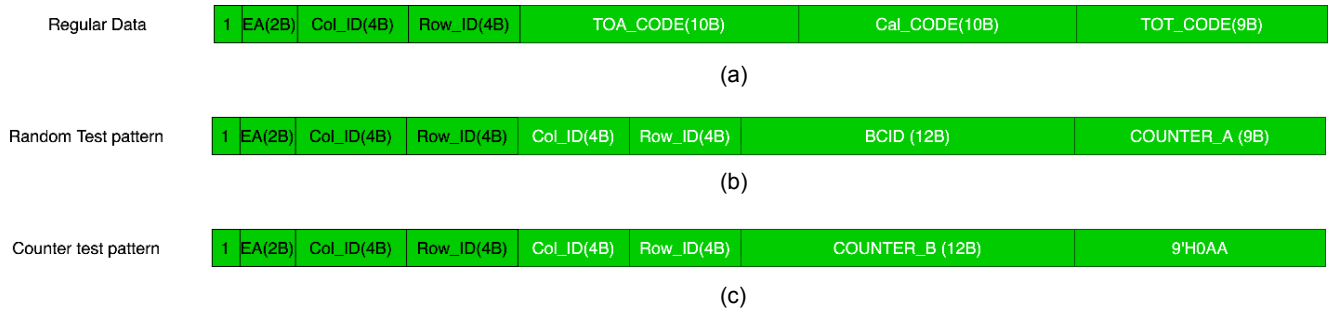


Figure 34. The regular data and test pattern definition: (a)regular data, (b)random test pattern, (c)counter test pattern.

The readout does not distinguish between normal TDC data and test patterns in the entire readout process, except for putting different event types in the header. The test pattern is designed to verify the integrity of the received data and provide relevant data fields. The random test pattern includes an 8-bit pixel ID field in its 29-bit data field, and it is expected to be identical to the pixel ID field in the data word when it is received. This feature checks if the readout is properly assigning the data to the correct pixels. The test pattern includes a 12-bit BCID, and it is expected to match the one in the frame header. This check ensures that the readout correctly collects pixels with the same bunch crossing ID. In test mode, the random L1 trigger is used along with the random test pattern. Since the random L1 trigger is based on a pseudo-random sequence, it is predictable whether the test pattern will be accepted by the random L1A.

There is another pseudo-random generator on each pixel that determines whether there is a hit or not for each bunch crossing. The probability of producing a hit is proportional to the occupancy that user specified. Since the seed of this pseudo-random generator is different for each pixel, the hits produced on pixels are overall random.

When the hit flag of this test pattern is set to true and it is expected to be selected by L1A, the 9-bit counter_A is incremented by 1. This feature enables detection of any missing data by verifying the continuity of the counter field for each pixel. If any data or test pattern is missed, it breaks the continuity of the counter, and the error will be detected.

When the counter test pattern is selected, the L1A trigger is generated every 39 clock periods. The counter test pattern includes a 12-bit counter in the data field, which increases by 1 for each clock period. With the cycle of L1A, it is expected the counter increases by 39 for each pixel in the received data. This feature allows for convenient data analysis using an oscilloscope.

3.4.2 Monitoring and triggering capabilities

The ETROC2 chip provides monitoring and triggering capabilities by directly outputting hit flags to the control room with short and fixed latency for every LHC clock. Users can adjust the timing window to select the hit flag. Due to the wide timing window of the TDC, the ETROC chip can observe hits with TOA up to 12 ns, providing the opportunity to observe off-timing particles.

The granularity of the trigger is programmable based on the 16 $\times$ 16 pixels. In the finest granularity, each trigger hit combines 4 $\times$ 4 pixels and output 16 trigger bits. In the coarsest granularity, the whole 256 pixels produce one bit hit. The different trigger granularities, ranging from the finest granularity of 4x4 pixels to the coarsest granularity of one hit for the entire 256 pixels, are illustrated in Figure 35 **(the plot needs to be updated)**.

Because there is no synchronous information in the raw trigger bit stream, there is no way to identify the bunch crossing identification (BCID) of the trigger bits if they are simply output to users. However, ETROC2 chip takes advantage of the few bunch gap slots in the LHC accelerator to send a special trigger signal that carries the BCID information, allowing the correct identification of the trigger bits. ETROC flashes the trigger bits when the BCID matches the user-specified bunch gap. By searching for the flashing bits on the received trigger bit stream,

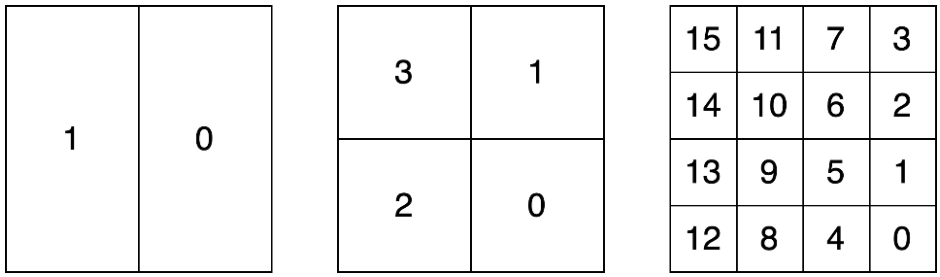


Figure 35. The granularity of the trigger bits.

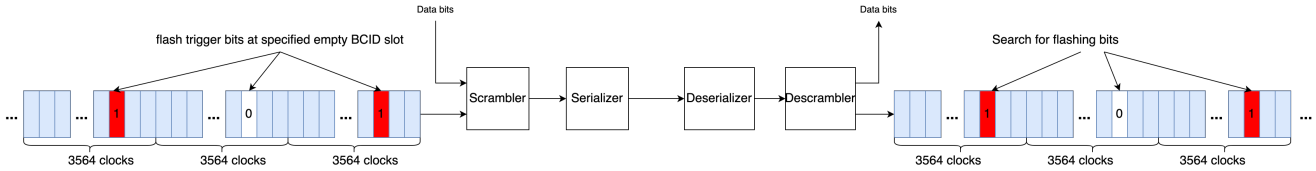


Figure 36. Identify the BCID of the trigger bits with a flashing bits.

users can easily identify the BCID of the corresponding trigger bits. Since there are no expected hits during the bunch gap, this process does not have any impact on the normal trigger bit stream (See figure 36)

3.4.3 Configuration of Readout data/trigger stream

The data frame and trigger bits have different signal paths before the serializers as shown in figure 37. The data frame is buffered in a circular buffer and an L1 buffer, and the L1A signal is used to select the corresponding data for output. Similarly, the trigger bits pass through the circular buffer and L1 buffer to maintain the continuous trigger data stream. The serializers' data rate can be programmed independently to 320 Mbps, 640 Mbps, or 1280 Mbps. The trigger and data can be merged into one serial port or separated into two ports, offering flexibility in the output configuration (see figure 38). The configurations for readout with different parameters are discussed in detailed below.

Disable/enable Trigger and Data The trigger or data of each pixel can be disable independently. If both trigger and data of a pixel, the clock of this pixel is disabled to reduce the clock noise. This feature can be used to study the noise relative to 40 MHz clock signal.

Trigger/data Selection Window The data and trigger can be selected with specific Time of Arrival (TOA), Time over Threshold (TOT), and Calibration Code (CAL) windows, independently for each pixel. The selection of trigger and data does not impact each other.

Data Output Rate and trigger bit size The raw data rate of the two output ports can be configured with 320 Mbps, 640 Mbps or 1280 Mbps. When the data rate is 320 Mbps, the serializer input word is 8 bits and the word rate is 40 M word/second. Because the trigger bits

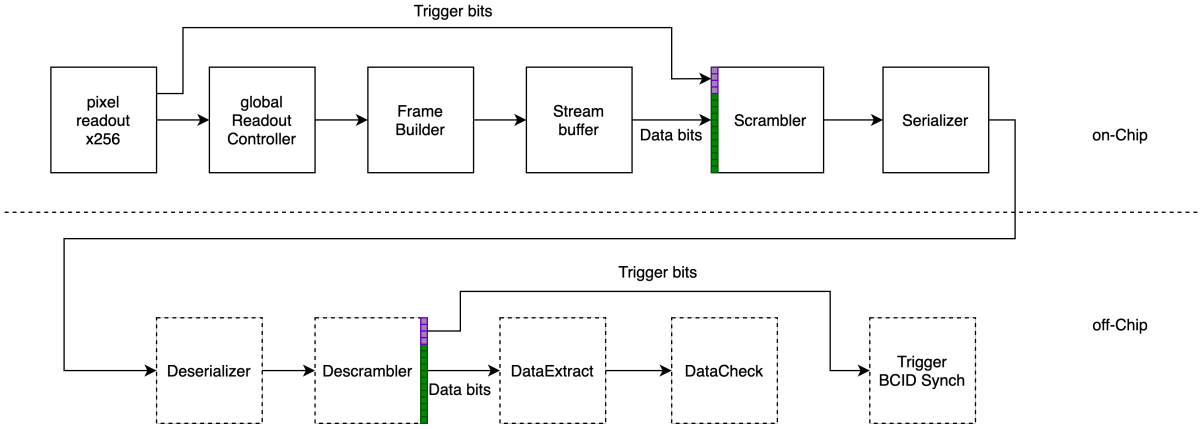


Figure 37. The trigger and data path in ETROC2.

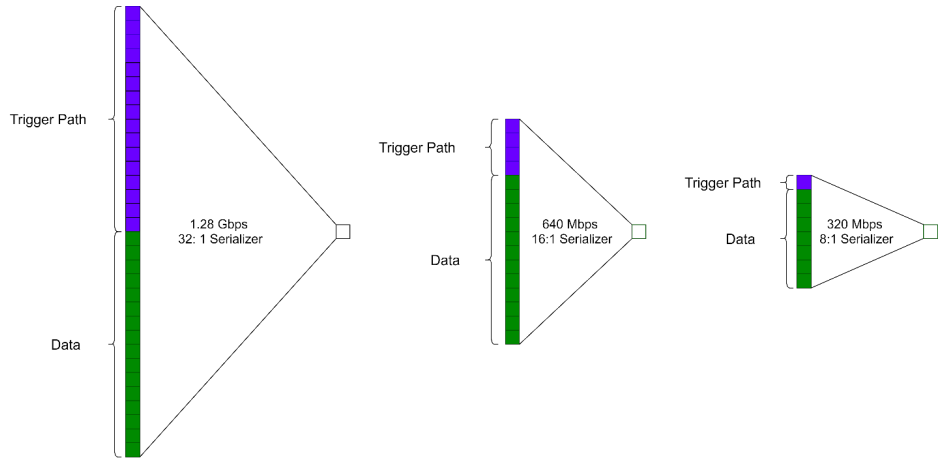


Figure 38. Merge of trigger/data in serializer at different data rate.

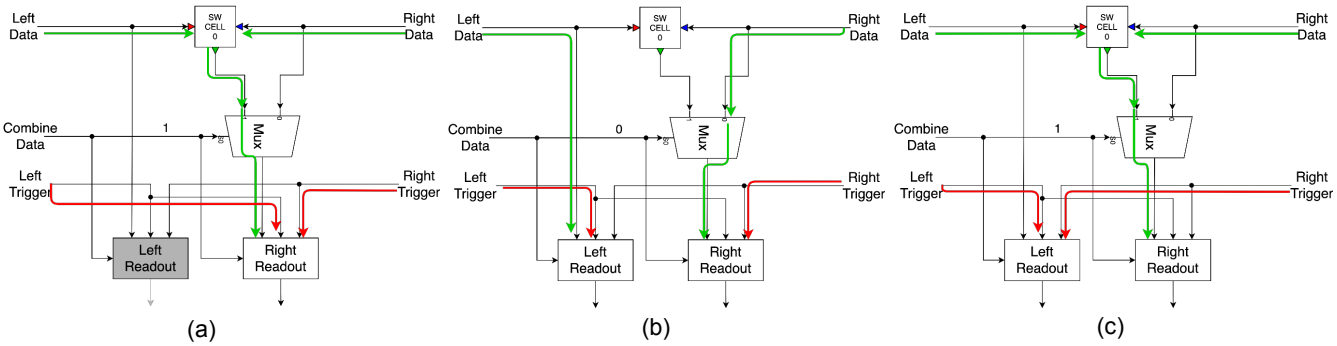


Figure 39. Schematic of trigger/data ports assignment, (a)Trigger/data share single port readout (single port = 1'b1, MergetriggerData=1'bx), (b)Data/trigger is split to two ports and trigger/-data share each port (single port = 1'b0, MergetriggerData=1'b1), (c)Separate trigger and data to two ports (single port = 1'b0, MergetriggerData=1'b0).

are output for every LHC clock period, this means we can select the trigger bit size up to 8. For data rate 640 Mbps and 1280 Mbps, the up-limit trigger bit size of each output port is up to 16 bits (Figure 37).

Data/trigger Configuration ETROC has a right and a left serial port, and it is flexible to configure the trigger and data for two ports. If ETROC operates in single port mode, both data and trigger are output via the right serial port, and the data frame and trigger bits can be correctly separated and reconstructed after the deserializer. If ETROC operates in two-port mode, the data frame and trigger can be merged into a single serial port. The right half pixels are read out from the right serial port and the left half pixels from the left serial port. Alternatively, the data frame and trigger can be separated in two-port mode, with the trigger bits output on the left serial port and the data frame on the right serial port (see figure 39)

Disable Scrambler We can disable scrambler for debug purpose. In the normal data taking, the data is scrambled before the serializer (figure 37). The scrambled data is DC balanced and AC coupling capacitors can be used. For the debug purpose, user may turn off the scrambler to check the raw serial data.

BCID Offset User can specify the BCID value when BCID reset is received. The default setting is 0.

On-chip L1 Config Users can select either the external L1 signal or on-chip L1 generator for use. During normal data taking, the external L1 signal is typically used. However, when the chip is in test mode, it is recommended to choose the on-chip L1 generator to thoroughly test the data integrity with a test pattern.

Command Name	Command Word(Hex)	Description
Idle	0xF0	The ETROC2 expects to receive idle command when no other commands are issued. Idle command is used for alignment as well.
Link Reset	0x33	The ETROC2 sends user defined pattern or PRBS for 25 ns (time period of 40 MHz clock)when received a link reset command.
BCR	0x5A	ETROC2 resets the BX(bunch crossing) counter to zero when receiving BCR command. BX counter rounds back at a user configurable integer (by default 3563).
STP	0x55	synchronize the trigger path. Sending well-defined pattern on trigger
L1A-CR	0x66	ETROC2 resets the L1A counter to zero when receiving L1A-CR
Charge Injection	0x69	injecting charge to the pixels
L1A	0x96	readout trigger
L1A & BCR	0x99	L1A and BCR
WS Start	0xA5	Waveform sampler starts to sample
WS Stop	0xAA	waveform sampler stop sampling

Table 9. Fast commands definition.

Link Reset Test Pattern and Fixed Test Pattern User can specify the test pattern to be either PRBS or fixed test pattern when setting the link to reset status. Users can use this feature for link status test or debug. The link reset can be configured with either slow control or fast control. It is a practical way to test the fast command module by sending a link reset signal on the fast command and verifying the serial port output.

3.5 Fast Control

The ETROC2 employs fast control to receive real-time requests sent by the users.

Ten 8-bit fast commands are pre-defined and built into the ETROC2. The ETROC2 expects to receive the fast command with a serial link at data rate of 320 Mbps continuously, which means ETROC2 receives a fast command every 25 ns. The pre-defined fast commands are summarized in table 9.

The pre-defined fast commands are expected to be sent to the ETROCs in a scheme showing in figure 40. As shown, one lpGBT distributes the fast commands to several ETROC with a two-drop scheme. The quality requirements of the fast command is not stringent in terms of jitter and delay. As also illustrated in the diagram, the clocks will be distributed to the ETROCs with dedicated links as the clocks are required to be with low jitter to achieve good timing performance.

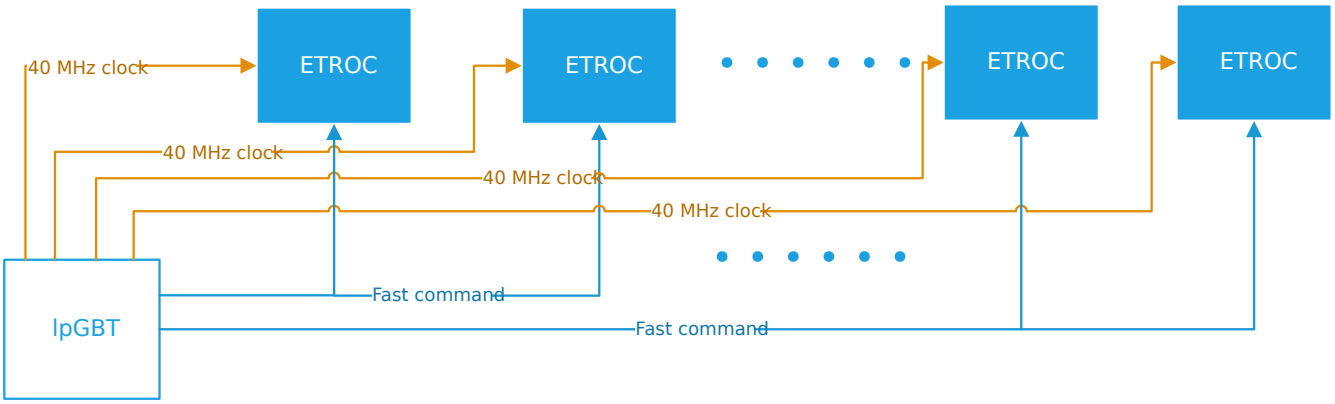


Figure 40. Fast command and clock distribution scheme on the module level.

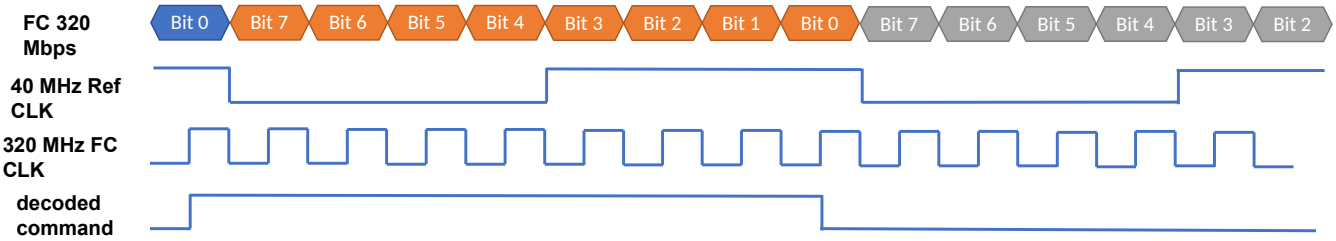


Figure 41. Timing diagram of word decoding.

A block named fast command receiver is designed to receive and decode the fast commands. In the fast command receiver, the incoming fast command bit stream are captured under a 320 MHz clock (FC clock) at rising edges. The captured bits are then interpreted by finding the word boundary and mapping to the pre-defined commands. Figure 41 illustrates the timing diagram of the process mentioned above. The transmission of the fast command is most significant bit(MSB) first, and the ETROC2 expects fast command MSBs at falling edges of the 40 MHz reference clock. The decoded commands are generated around the falling edges of the 40 MHz reference clock, and then used by other blocks at rising edges. This arrangement allows the maximal timing margin of a couple nano seconds.

The phase of the FC clock need to be aligned with the fast command bit stream so that the sampling is reliable. This is called bit alignment. Two bit alignment schemes, manual alignment and self-alignment, are considered. An I2C register, fcSelfAlignEn, is used to choose the alignment schemes between manual alignment and self-alignment.

3.5.1 Manual Alignment

Figure 42 shows the concept of the manual alignment scheme. The idea of the manual alignment is manually adding a delay to the fast command bit stream or the FC clock to avoid metastability when capturing the fast command bit stream with the FC clock.

Figure 43 is a simplified block diagram of the fast command receiving and decoding in manual alignment mode. The clock and fast command bit stream might be delayed in the phase adjuster before being used in the succeeding blocks for word aligning and decoding.

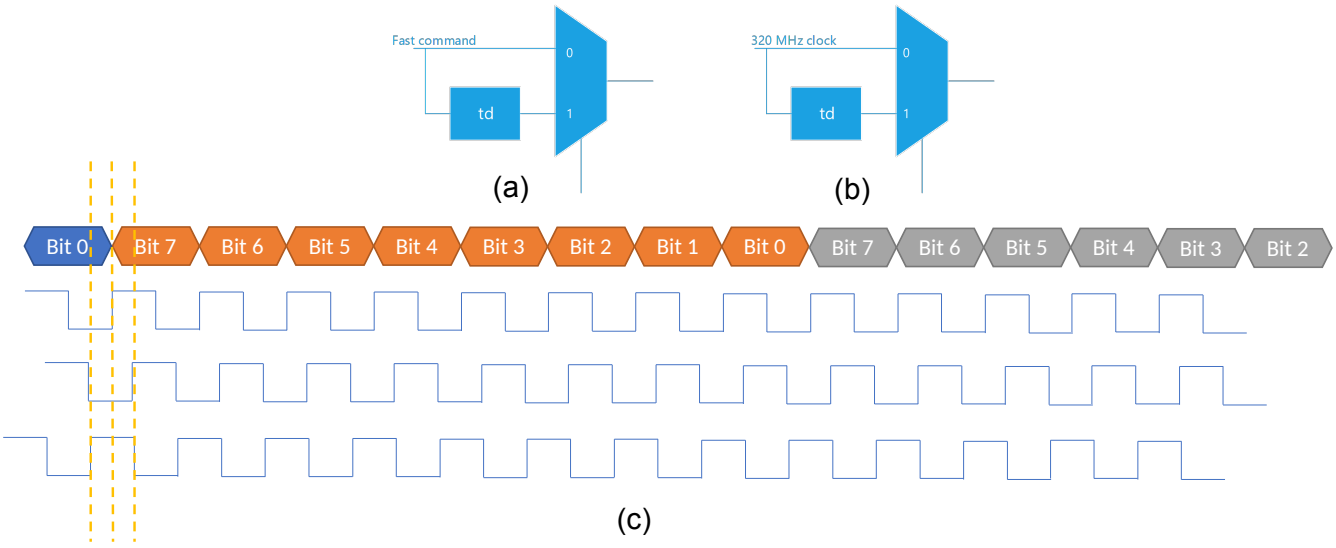


Figure 42. Concepts of the manual alignment, (a) delaying fast command bit stream, (b) delaying the FC clock, (c) timing diagram of the manual alignment.

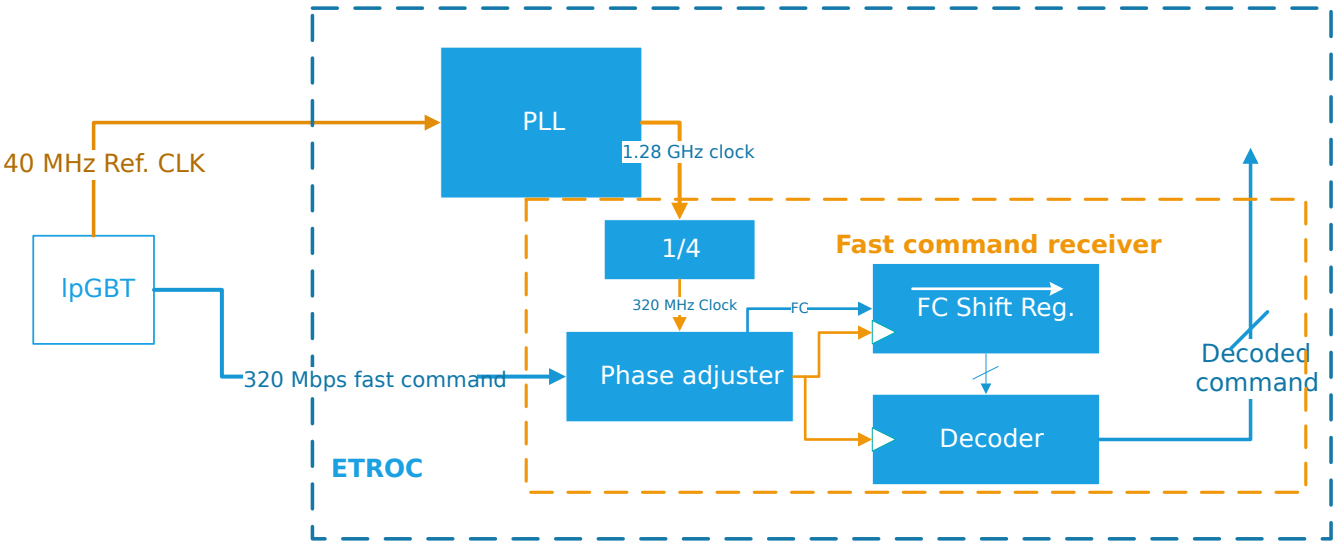


Figure 43. Simplified block diagram of the fast command receiving and decoding in manual alignment mode.

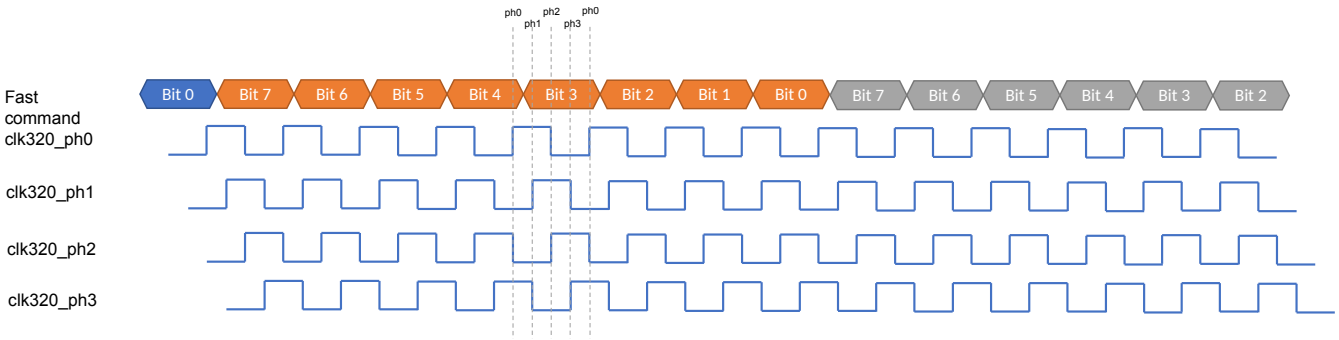


Figure 44. Conceptual timing diagram of the self-alignment.

Two I2C registers, fcClkDelayEn and fcDataDelayEn, are used to add the delay to the FC clock and the fast command bit stream, respectively.

3.5.2 Self-alignment

The self-alignment mode is optional, with the goal to make things easier for users. It is fully implemented at RTL level, and extensively simulated at RTL level. However due to the schedule of ETROC2, this mode is not fully verified for ETROC2.

Figure 44 illustrates the concept of the self-alignment scheme. In the self-alignment scheme, the fast command bit stream is captured by four 320 MHz clocks with equal phase space of 1/4 period of the clock. The captured data are then used to determine an optimal clock phase.

In figure 44, the optimal clock phase might be ph2 or ph3 because they are not close the edges of the bit stream. The self-alignment works in most of case where the jitter of the fast command bit stream is not too large, ignoring the jitter of the FC clock.

Figure 45 illustrates scenarios in which the self-alignment scheme works. The status of the phase decision and the optimal clock phase are shown in the figure. The status of the phase decision are listed below.

- ed0 = 1: at rising edge of p0 clock, preceding two samples (at p2 and p3) are different
- ed1 = 1: at rising edge of p1 clock, preceding two samples (at p3 and p0) are different
- ed2 = 1: at rising edge of p2 clock, preceding two samples (at p0 and p1) are different
- ed3 = 1: at rising edge of p3 clock, preceding two samples (at p1 and p1) are different

In scenarios through A to H, three clock phases would fall into the same bit, and only one clock phase fall into the neighbor bit or the bit edges, because of a small jitter of less then 780 ps.

In scenarios J, K, M and N, two clock phases fall into a bit, and the other two clock phases fall into edges, due to a larger jitter. The self-alignment still work correctly.

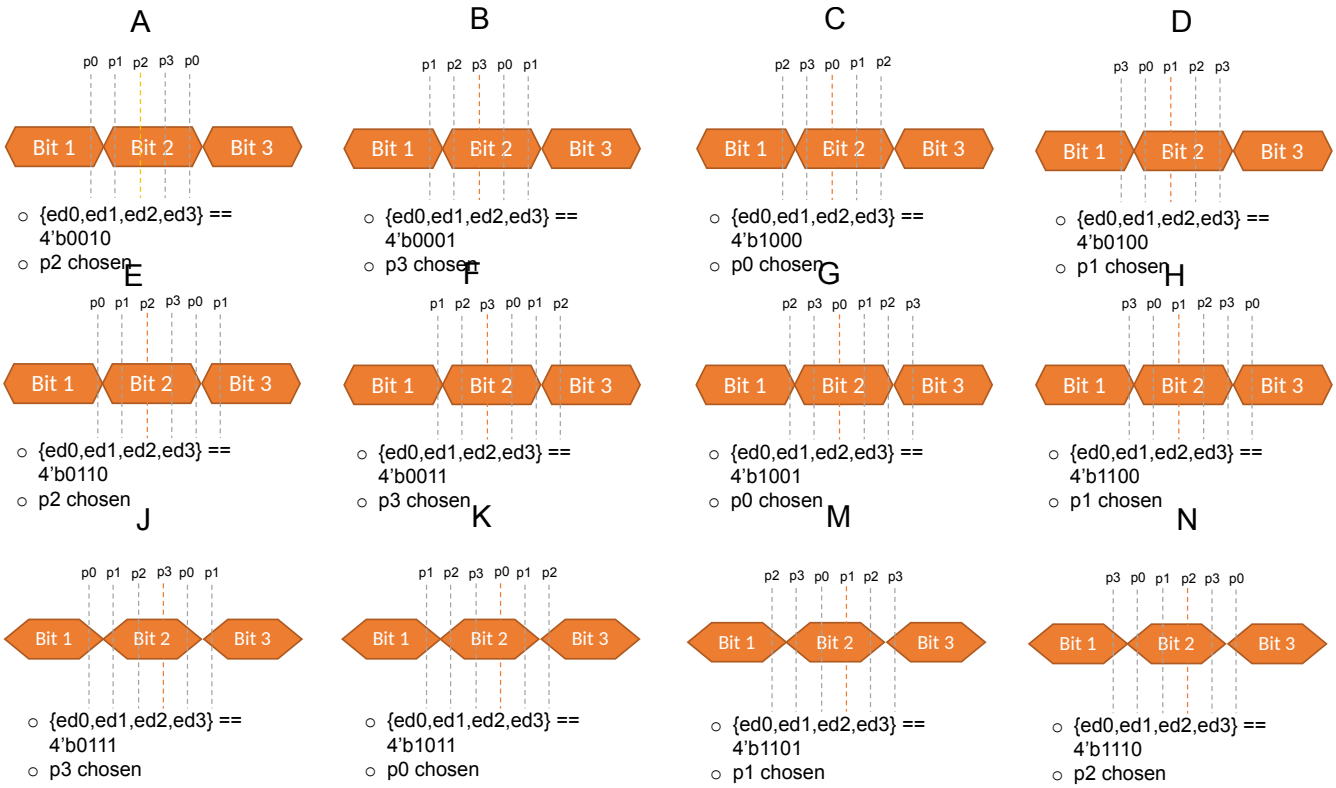


Figure 45. Scenarios of self-alignment working.

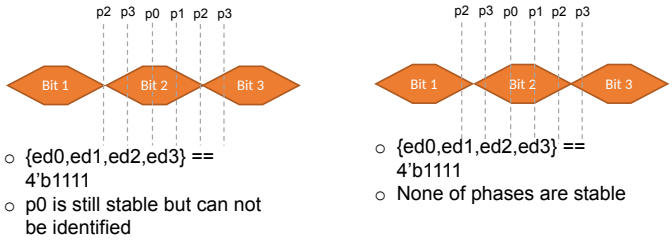


Figure 46. Scenarios of self-alignment not working.

In the cases with even larger jitter, the self-alignment does not function anymore, as depicted in figure 46.

The truth table in figure 47 summarizes all the possible status the might be reported. The phase decision status and error bit can be accessed with the I2C status registers fcAlignStatus[3:0] and fcBitAlignError in table 17.

3.5.3 Word Alignment and Command Decoding

The fast command alignment involves bit alignment and word alignment. Once the bit alignment is done with the manual alignment or self-alignment, the fast command word will be identified and decoded.

A fast command word includes 8 bits as shown in 9. To allow the ETROC2 to identify the word boundary from the fast command bit stream, the users should send IDLE command

	Line	ed0	ed1	ed2	ed3	p0	p1	p2	p3	error
Optimal clock found, and no error reported	1	1	0	0	0	1	0	0	0	0
	2	1	0	0	1	1	0	0	0	0
	3	1	0	1	1	1	0	0	0	0
	4	0	1	0	0	0	1	0	0	0
	5	1	1	0	0	0	1	0	0	0
	6	1	1	0	1	0	1	0	0	0
	7	0	0	1	0	0	0	1	0	0
	8	0	1	1	0	0	0	1	0	0
	9	1	1	1	0	0	0	1	0	0
	10	0	0	0	1	0	0	0	1	0
	11	0	0	1	1	0	0	0	1	0
	12	0	1	1	1	0	0	0	1	0
No good clock found, and error reported	13	0	1	0	1	0	0	0	0	1
	14	1	0	1	0	0	0	0	0	1
	15	1	1	1	1	0	0	0	0	1
	16	0	0	0	0	0	0	0	0	1

Figure 47. truth table of the self-alignment status.

continuously. ETROC2 will find the word boundary by identifying IDLE command word (0xF0) in the bit stream, and then confirm the founded boundary by looking for 16 consecutive IDLE commands.

After bit alignment and word alignment complete, ETROC2 is able to receive and decode the fast command correctly. Figure 48 illusrates an exmaple of fast command receiving and decoding.

Note that the manual alignment is the default and users should use this mode for initial testing for ETROC2. The self-alignment will be studied by experts only and will be made avaiable for ETROC3.

3.6 Slow Control

3.6.1 I2C Control

An I2C slave including peripheral and in-pixel registers is used in ETROC2 for slow control. The I2C slave features a distributed implementation, including three basic blocks, named periphery, column adapter and pixel adapter, respectively. Figure 50 shows the structure of the I2C slave.

Periphery interfaces with an off-chip I2C master, and it coordinates the communication inside the I2C slave. The I2C slave supports a couple of protocols to talk with the master, including single-register write/read, multi-register write/read and broadcast. Figure 51 illustrates the details of the protocols supported by the I2C slave. It should be mentioned that the broadcast is useful when writing a configuration to all the pixels. The address of the I2C slave is 7'b11xxxxx, where the lower 5-bit address is accessible through 5 pads to configure the ETROC2 I2C slave address off-chip.

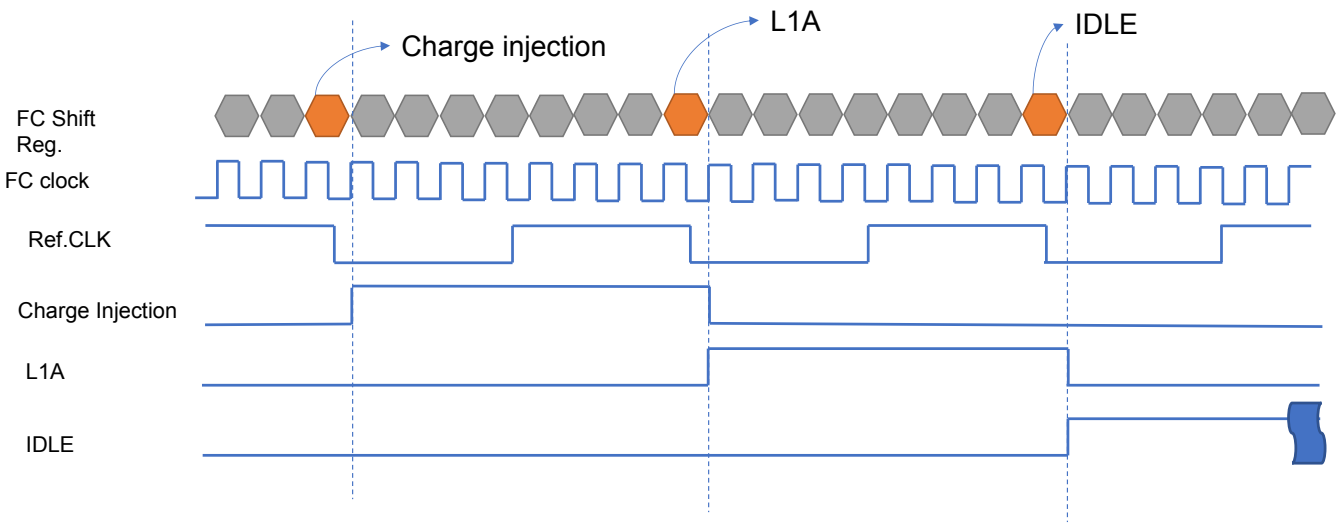


Figure 48. An example of fast command decoding.

Address bits	Description
Bits[15]	1: pixel matrix, 0: periphery
Bits[14]	1: status, 0: configuration
Bits[13]	0: direct message to a specified pixel, 1: broadcast to all pixels
Bits[12:9]	Column Address
Bits[8:5]	Row address
Bits[4:0]	In-pixel register address

Table 10. Address for in-pixel I2C access

The periphery has 32 bytes register for configuration and 16 bytes address for status. A pixel I2C adapter has 32 bytes configuration register and 8 bytes status addressing space. The configuration registers in the periphery have name convention of PeriCfg<N>, where N is the register number ranging from 0 to 31. The status addresses in the periphery use name convention of PeriStat<N>, where N is the address ranging from 0 to 15.

The in-pixel configuration registers addresses have name convention of PixR<R_n>C<C_n>Cfg<N>, where N is the register address ranging from 0 to 31, R_n is the pixel row number and C_n represents the pixel column number. The in-pixel status addressing space have name convention of PixR<R_n>C<C_n>Stat<N>, where N is the address ranging from 0 to 7, R_n is the pixel row number and C_n represents the pixel column number.

Note that configuration registers are writable, but the status is read only.

A pixel can be denoted by a number ranging from 0 to 255, or by the row number and the column number, as illustrated in Figure 49. The row zero is the bottom row and the column zero is the rightmost column.

Table 10 summarizes the 16-bit address used to access pixel configuration registers or status. Note that when broadcasting, the row address and column address are ignored.

Table 11 contains the 16-bit address to access periphery for various purposes. Besides the 32-byte configuration registers and 16-byte status, there are other two features are implemented in periphery, as described in table 11.

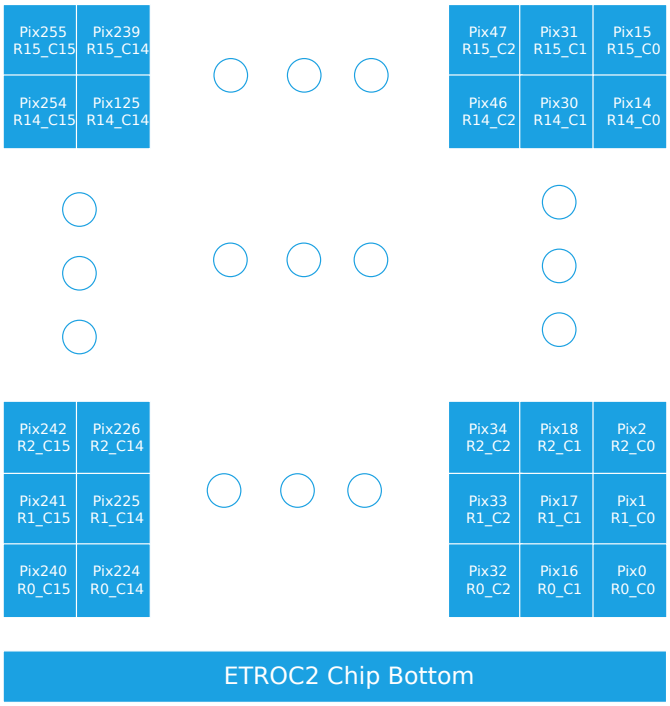


Figure 49. ETROC2 pixel naming convention.

Periphery functions	Address
Configuration	16'h0000 - 16'h001F
Status	16'h0100 - 16'h010F
SEU Counter	16'h0120 - 16'h0123 32 bit counter: 16'h0123∼SEU Counter[31:24] 16'h0122∼SEU Counter[23:16] 16'h0121∼SEU Counter[15:8] 16'h0120∼SEU Counter[7:0]
Magic number	16'h0020 The magic number is 8'b10010110. If reg20(8'h20) equals the magic number and reg21[2:0] equals 3'b000, the clock would be disabled and the I2C salve would not be accessible with normal operation. A hard reset is needed to bring it back. Here reg20 and reg21 are internal registers to control the I2C slave. They are writable and readable with the I2C write/read operation.

Table 11. Address for I2C periphery access.

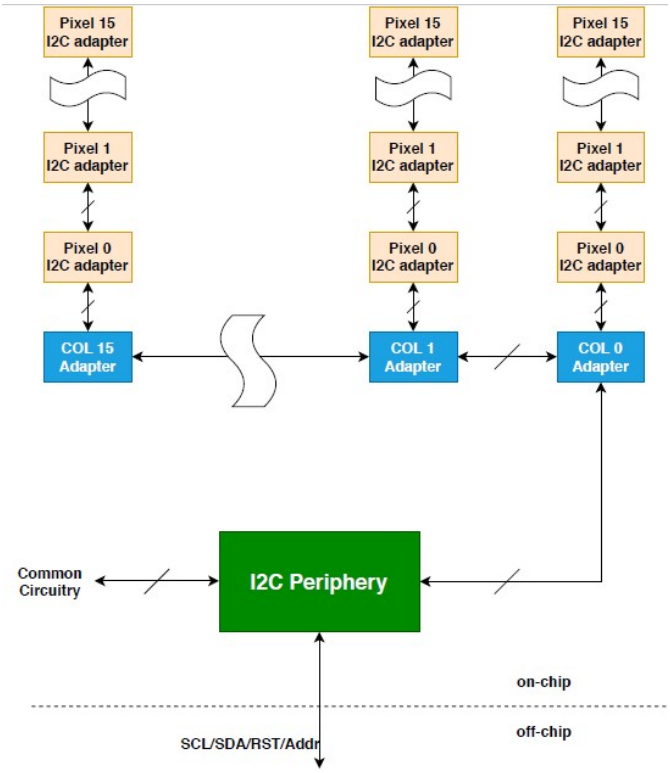


Figure 50. ETROC2 I2C slave structure: distributed implementation with three building blocks, periphery, column adapter and pixel adapter.

A 32-bit counter is implemented to record the SEU events, which is useful during SEU testing. The magic number can be used to protect I2C registers to prevent them from being written.

Table 12 lists all the configuration registers used in ETROC2 peripheral. More detailed explanation could be found in the later sections. The peripheral configuration register map is provided in table 13.

Table 14 lists all the in-pixel configuration registers, and table 15 is the in-pixel configuration register map.

The I2C status of pixel and peripheral are summarized in table 16 and table 17, respectively.

The status address map of pixel and peripheral are summarized in table 19 and table 18, respectively. Notice that the highest byte of pixel configuration, pixelConfig[255:248], is accessible at the highest byte of pixel status, PixR<R_n>C<C_n>STA7.

Table 12. Peripheral configuration registers.

Begin of Table 12				
Num.	**Register**	**Description**	**Default**	**Group Name**
1	readoutClockDelayPixel[4:0]	Phase delay of pixel readout clock, 780 ps a step	5'b00000	Readout
2	readoutClockWidthPixel[4:0]	positive pulse width of pixel clock, 780 ps a step	5'b10000	Readout
3	readoutClockDelayGlobal[4:0]	Phase delay of global readout clock, 780 ps a step	5'b00000	Readout

Continuation of Table 12

Num.	Register	Description	Default	Group Name
4	readoutClockWidthGlobal[4:0]	positive pulse width of global readout clock, 780 ps a step	5'b10000	Readout
5	serRateRight[1:0]	Data rate selection of the right data port 2'b00: 320 Mbps; 2'b01: 640 Mbps; 2'b10: 1280 Mbps.	2'b01	Readout
6	serRateLeft[1:0]	Data rate selection of the left data port 2'b00: 320 Mbps; 2'b01: 640 Mbps; 2'b10: 1280 Mbps.	2'b01	Readout
7	linkResetTestPattern	Link reset test patten selection 0: PRBS,1: fixed pattern	1'b1	Readout
8	linkResetFixedPattern[31:0]	User-specified pattern to be sent during link reset, LSB first	32'HACC78CC5	Readout
9	emptySlotBCID[11:0]	empty BCID slot for synchronization	12'H001	Readout
10	triggerGranularity[2:0]	the trigger data size varies from 0,1,2,4,8,16 0/6/7: trigger data size is 0. 1: trigger data size is 1. 2: trigger data size is 2. 3: trigger data size is 4. 4: trigger data size is 8. 5: trigger data size is 16.	3'H0	Readout
11	disScrambler	Disable scrambler 0: enable scrambler 1: disable scrambler	1'b0	Readout
12	mergeTriggerData	merge trigger and data in a port 0: trigger and data in separate port, only valid when single port is false. 1: trigger and data are merged in serial port	1'b0	Readout
13	singlePort	enable single port or both ports 0: use both left and right serial ports 1: use right serial port only	1'b1	Readout
14	onChipL1AConf[1:0]	On-chip L1A mode 2'b0x: on-chip L1A disable; 2'b10: periodic L1A; 2'b11: random L1A	2'b00	Readout
15	BCIDoffset[11:0]	BCID when BCID is reset	12'H0D0	Readout
16	fcSelfAlignEn	Fast command decoder self-alignment mode enable. 1: self-alignment mode enabled; 0: manual alignment mode enabled	1'b0	Fast Command
17	fcClkDelayEn	Enable clock delay in fast command manual alignment mode	1'b0	Fast Command
18	fcDataDelayEn	Enable data delay in fast command manual alignment mode, active high.	1'b0	Fast Command
19	chargeInjectionDelay[4:0]	The charge injection delay to the 40MHz clock rising edge. Start from the rising edge of 40 MHz clock, each step 781 ps. The pulse width is fixed of 50 ns.	5'H18	Readout
20	RefStrSel[7:0]	TDC reference strobe selection.	8'b00000011	Clock Generator
21	PLL_BIASGEN_CONFIG[3:0]	Charge pump bias current selection, [0 : 8 : 120] uA. **Debugging use only**.	4'b1000	Clock Generator
22	PLL_CONFIG_I_PLL[3:0]	Bias current selection of the I-filter unit cell in PLL mode [0 : 1.1 : 8] uA. **Debugging use only**.	4'b1001	Clock Generator
23	PLL_CONFIG_P_PLL[3:0]	Bias current selection of the P-filter unit cell in PLL mode [0 : 5.46 : 82] uA. **Debugging use only**.	4'b1001	Clock Generator

Continuation of Table 12

Num.	Register	Description	Default	Group Name
24	PLL_R_CONFIG[3:0]	Resistor selection of the P-path in PLL mode [R = 1/2 * 79.8k / CONFIG] Ohm. **Debugging use only**.	4'b0010	Clock Generator
25	PLL_vcoDAC[3:0]	Bias current selection of the VCO core [0: 0.470 : 7.1] mA. **Debugging use only**.	4'b1000	Clock Generator
26	PLL_vcoRailMode	Output rail-to-rail mode selection of the VCO, active low. 1'b0 -> rail-to-rail output. 1'b1 -> CML output. **Debugging use only**.	1'b1	Clock Generator
27	PLL_ENABLEPLL	Enable PLL mode, active high. **Debugging use only**.	1'b0	Clock Generator
28	PLL_FBDiv_skip	Adjusting the phase of the output clk1G28 of freqPrescaler in the feedback divider(N=64) by one skip from low to high. **Debugging use only**.	1'b0	Clock Generator
29	PLL_FBDiv_clkTreeDisable	Disable the feedback divider,active high. 1'b0 : all output clocks with different frequencies (40MHz-2.56GHz) are enabled. 1'b1 : The input clk2G56 from the prescaler and all output clocks are disabled. **Debugging use only**.	1'b0	Clock Generator
30	PLLclkgen_disSER	Disable output clocks for serializer, active high. When PLLclkgen_disSER is high, the following clocks are disabled: clk2g56S, clk2g56SN, clk5g12S, clk5g12SN. **Debugging use only**.	1'b1	Clock Generator
31	PLLclkgen_disVCO	Disable VCO output buffer(associated with clk5g12lshp, clk5g12lshn), active high. clk5g12lsh is the output clock of the first input buffer in prescaler, and the source clock for all output clocks. Once disabled, all output clocks are disabled. **Debugging use only**.	1'b0	Clock Generator
32	PLLclkgen_disEOM	Disable output clocks for EOM, active high. When PLLclkgen_disEOM is high, the following clocks are disabled: clk5g12EOMp, clk5g12EOMn. **Debugging use only**.	1'b1	Clock Generator
33	PLLclkgen_disCLK	Disable the internal clock buffers and 1/2 clock divider in prescaler, active high. When PLLclkgen_disCLK is high, all output clocks are disabled. **Debugging use only**.	1'b0	Clock Generator
34	PLLclkgen_disDES	Disable output clocks for deserializer, active high. When PLLclkgen_disDES is high, the following clocks are disabled: clk2g56Qp, clk2g56Qn, clk2g56Ip, clk2g56In. clk2g56Q is the 2.56 GHz clock for test in ETROC_PLL. clk2g56Q is used as WS clock in ETROC2. **Debugging use only**.	1'b0	Clock Generator
35	CLKSel	Selecting PLL clock or off-chip clock for TDC and readout. 0: using off-chip clocks for TDC and readout; 1: using PLL clocks for TDC and readout. **Debugging use only**.	1'b1	Clock Generator

Continuation of Table 12

Num.	Register	Description	Default	Group Name
36	PS_CPCurrent[3:0]	Charge pump current control bits, range from 0 to 15uA for charge and discharge. **Debugging use only**.	4'b0001	Clock Generator
37	PS_CapRst	Reset the control voltage of DLL to power supply, active high. **Debugging use only**.	1'b0	Clock Generator
38	PS_Enable	Enabling DLL, active high. **Debugging use only**.	1'b1	Clock Generator
39	PS_ForceDown	Force to pull down the output of the phase detector, active high. **Debugging use only**.	1'b0	Clock Generator
40	PS_PhaseAdj[7:0]	Phase selecting control bits, PS_PhaseAdj[7:3] for coarse, PS_PhaseAdj[2:0] for fine.	8'b00000000	Clock Generator
41	CLK40_EnRx	Enable the Rx for the 40 MHz reference clock, active high. **Debugging use only**.	1'b1	Clock Generator
42	CLK40_EnTer	Enable internal termination of the Rx for the 40 MHz reference clock, active high. **Debugging use only**.	1'b1	Clock Generator
43	CLK40_Equ[1:0]	Equalization strength of the Rx for the 40 MHz reference clock. 2'b00: equalization is turned off; 2'b11: maximal equalization. **Debugging use only**.	2'b00	Clock Generator
44	CLK40_InvData	Inverting data of the Rx for the 40 MHz reference clock, active high. **Debugging use only**.	1'b0	Clock Generator
45	CLK40_SetCM	Set common voltage of the Rx for the 40 MHz reference clock to 1/2 vdd, active high. **Debugging use only**.	1'b1	Clock Generator
46	CLK1280_EnRx	Enable the Rx for the 1.28GHz clock, active high. **Debugging use only**.	1'b1	Clock Generator
47	CLK1280_EnTer	Enable the internal termination of the Rx for the 1.28GHz clock, active high. **Debugging use only**.	1'b1	Clock Generator
48	CLK1280_Equ[1:0]	Equalization strength of the Rx for the 1.28GHz clock. 2'b00: equalization off; 2'b11: maximal equalization. **Debugging use only**.	2'b00	Clock Generator
49	CLK1280_InvData	Inverting data of the Rx for the 1.28GHz clock, active high. **Debugging use only**.	1'b0	Clock Generator
50	CLK1280_SetCM	Set common voltage of the Rx for the 1.28GHz clock to 1/2 vdd, active high. **Debugging use only**.	1'b1	Clock Generator
51	FC_EnRx	Enable the Rx for the fast command, active high. **Debugging use only**.	1'b1	Fast Command
52	FC_EnTer	Enable internal termination of the Rx for the fast command, active high. **Debugging use only**.	1'b1	Fast Command
53	FC_Equ[1:0]	Equalization strength of the Rx for the fast command. 2'b00: equalization off; 2'b11: maximal equalization. **Debugging use only**.	2'b00	Fast Command
54	FC_InvData	Inverting data of the Rx for the fast command, active high. **Debugging use only**.	1'b0	Fast Command
55	FC_SetCM	Set common voltage of the Rx for the fast command to 1/2 vdd, active high. **Debugging use only**.	1'b1	Fast Command
56	disPowerSequence	Disabling the power up sequence, active high.	1'b0	Global

Continuation of Table 12

Num.	Register	Description	Default	Group Name
57	softBoot	Reset power sequencer controller, active high.	1'b0	Global
58	eFuse_TCKHP[3:0]	The register controlling the SCLK pulse width, ranging ranges from 3 us to 10 us with step of 0.5 us. The default value is 4 corresponding to 5 us pulse width. **Debugging use only**	4'b0100	Global
59	eFuse_EnClk	eFuse clock enable. 1: enabling the clock of the eFuse controller; 0: disabling the clock of the eFuse controller.	1'b0	Global
60	eFuse_Mode[1:0]	operation mode of eFuse. 2'b01: programming mode; 2'b10: reading mode.	2'b10	Global
61	eFuse_Rstn	Reset signal of the eFuse controller, active low.	1'b1	Global
62	eFuse_Start	Start signal of the eFuse programming. A positive pulse will start the programming.	1'b0	Global
63	eFuse_Prog[31:0]	Data to be written into eFuse	32'H00000000	Global
64	eFuse_Bypass	Bypass eFuse. 1'b0: eFuse output Q[31:0] is output; 1'b1: eFuse raw data from I2C(eFuse_Prog[31:0]) is output.	1'b1	Global
65	lfLockThrCounter[3:0]	If the number of instantLock is true for $2^{lfLockThrCounter}$ in a row, the PLL is locked in the initial status.	4'HB	Clock Generator
66	lfReLockThrCounter[3:0]	If the number of instantLock is true for $2^{lfReLockThrCounter}$ in a row, the PLL is relocked before the unlock status is confirmed.	4'HB	Clock Generator
67	lfUnLockThrCounter[3:0]	If the number of instantLock is false for $2^{lfUnLockThrCounter}$ in a row, the PLL is unlocked.	4'HB	Clock Generator
68	asyAlignFastcommand	The fast command bit clock alignment command issued by I2C. used in self-alignment only. Initializing the clock phase alignment process at its rising edge(synchronized by the 40 MHz PLL clock)	1'b0	Fast Command
69	asyLinkReset	Link reset signal from I2C, active high. If it is high, ETROC2 sends test pattern via link.	1'b0	Readout
70	asyPLLReset	Reset PLL AFC from I2C, active low	1'b1	Clock Generator
71	asyResetChargeInj	Reset charge injection module, active low.	1'b1	Readout
72	asyResetFastcommand	Reset fastcommand from I2C, active low.	1'b1	Fast Command
73	asyResetGlobalReadout	Reset globalReadout module, active low.	1'b1	Readout
74	asyResetLockDetect	Reset lock detect, active low (original lockDetect reset is active high, polarity changed)	1'b1	Clock Generator
75	asyStartCalibration	Start PLL calibration process, active high.	1'b1	Clock Generator
76	VRefGen_PD	Power down voltage reference generator, active high. 1: the voltage reference generator is down. 0: the voltage reference generator is up.	1'b0	Global
77	TS_PD	Power down the temperature sensor, active high. 1: the temperature sensor is down; 0: the temperature sensor is up.	1'b0	Global

Continuation of Table 12

Num.	Register	Description	Default	Group Name
78	TDCClockTest	The TDC clock testing enable. 1: sending TDC clock at the left serial port; 0: sending left serializer data at the left port.	1'b0	Readout
79	TDCStrobeTest	The TDC reference strobe testing enable. 1: sending TDC reference strobe at the right serial port; 0: sending right serializer data at the right port.	1'b0	Readout
80	LTx_AmplSel[2:0]	Left Tx amplitude selction. 3b'000: min amplitude(50mV) 3b'111 = max amplitude(320mV) Step size is about 40mV	3'b100	Readout
81	RTx_AmplSel[2:0]	Right Tx amplitude selction. 3b'000: min amplitude(50mV) 3b'111 = max amplitude(320mV) Step size is about 40mV	3'b100	Readout
82	disLTx	Left Tx disable, active high.	1'b0	Readout
83	disRTx	Right Tx disable, active high.	1'b0	Readout
84	GRO_TOARST_N	GRO TOA reset, active low.	1'b1	Global
85	GRO_Start	GRO Start, active high	1'b0	Global
86	GRO_TOA_Latch	GRO TOA latch clock	1'b1	Global
87	GRO_TOA_CK	GRO TOA clock	1'b1	Global
88	GRO_TOT_CK	GRO TOT clock	1'b1	Global
89	GRO_TOTRST_N	GRO TOT reset, active low.	1'b1	Global

End of Table 12

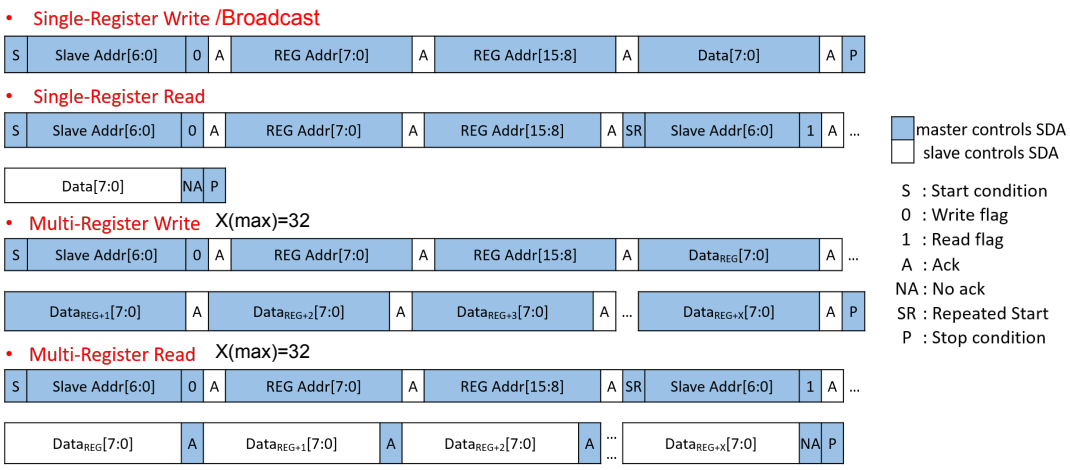


Figure 51. ETROC2 I2C communication protocol: Single-register write/read, multi-register write/read and broadcast.

Peripheral Registers	BITS								Default(Hex)
	7	6	5	4	3	2	1	0	
PeriCfg0	PLL_FBDiv_skip	PLL_FBDiv_clkTreeDisable	CLKSel	PLLclkgen_disVCO	PLLclkGen_disSER	PLLclkgen_disEOM	PLLclkgen_disDES	PLLclkgen_disCLK	0x2C
PeriCfg1	PLL_CONFIG_I_PLL[3:0]				PLL_BiasGen_CONFIG[3:0]				0x98
PeriCfg2	PLL_R_CONFIG[3:0]				PLL_CONFIG_P_PLL[3:0]				0x29
PeriCfg3	VRefGen_PD	−−	PLL_ENABLEPLL	PLL_vcoRailMode	PLL_vcoDAC[3:0]				0x18
PeriCfg4	TS_PD	PS_ForceDown	PS_Enable	PS_CapRst	PS_CPCurrent[3:0]				0x21
PeriCfg5	PS_PhaseAdj[7:0]								0x00
PeriCfg6	RefStrSel[7:0]								0x03
PeriCfg7	GRO_TOARST_N	GRO_Start	CLK40_SetCM	CLK40_InvData	CLK40_Equ[1:0]		CLK40_EnTer	CLK40_EnRx	0xA3
PeriCfg8	GRO_TOA_Latch	GRO_TOA_CK	CLK1280_SetCM	CLK1280_InvData	CLK1280_Equ[1:0]		CLK1280_EnTer	CLK1280_EnRx	0xE3
PeriCfg9	GRO_TOT_CK_Latch	GRO_TOTRST_N	FC_SetCM	FC_InvData	FC_Equ[1:0]		FC_EnTer	FC_EnRx	0xE3
PeriCfg10	BCIDoffset[7:0]								0xD0
PeriCfg11	emptySlotBCID[3:0]				BCIDoffset[11:8]				0x10
PeriCfg12	emptySlotBCID[11:4]								0x00
PeriCfg13	asyPLLReset	asyLinkReset	asyAlignFastcommand	readoutClockDelayPixel[4:0]					0x80
PeriCfg14	asyResetGlobalReadout	asyResetFastcommand	asyResetChargeInj	readoutClockWidthPixel[4:0]					0xF0
PeriCfg15	−−	asyStartCalibration	asyResetLockDetect	readoutClockDelayGlobal[4:0]					0x60
PeriCfg16	LTx_AmplSel[2:0]			readoutClockWidthGlobal[4:0]					0x90
PeriCfg17	RTx_AmplSel[2:0]			chargeInjectionDelay[4:0]					0x98
PeriCfg18	disLTx	onChipL1AConf[1:0]		fcDataDelayEn	fcClkDelayEn	fcSelfAlignEn	softBoot	disPowerSequence	0x00
PeriCfg19	disRTx	singlePort	serRateRight[1:0]		serRateLeft[1:0]		linkResetTestPattern	disScrambler	0x56
PeriCfg20	eFuse_TCKHP[3:0]				triggerGranularity[2:0]			mergeTriggerData	0x40
PeriCfg21	−−	−−	eFuse_Bypass	eFuse_Start	eFuse_Rstn	eFuse_Mode[1:0]		eFuse_EnClk	0x2C
PeriCfg22	eFuse_Prog[7:0]								0x00
PeriCfg23	eFuse_Prog[15:8]								0x00
PeriCfg24	eFuse_Prog[23:16]								0x00
PeriCfg25	eFuse_Prog[31:24]								0x00
PeriCfg26	linkResetFixedPattern[7:0]								0xC5
PeriCfg27	linkResetFixedPattern[15:8]								0x8C
PeriCfg28	linkResetFixedPattern[23:16]								0xC7
PeriCfg29	linkResetFixedPattern[31:24]								0xAC
PeriCfg30	lfReLockThrCounter[3:0]				lfLockThrCounter[3:0]				0xBB
PeriCfg31	−−		TDCStrobeTest	TDCClockTest	lfUnLockThrCounter[3:0]				0x0B

Table 13. Periphery configuration register map.

Table 14. In-pixel configuration registers.

Begin of Table 14				
Num.	**Register**	**Description**	**Default**	**Group Name**
1	CLSel[1:0]	Select of load capacitance of the preamp first stage. 2'b00: 0 fC; 2'b01–> 80 fC; 2'b10–> 80 fC; 2'b01–> 160 fC. **Debugging use only**.	2'b00	Amplifier
2	IBSel[2:0]	Bias current selection of the input transistor in the preamp. 3'b000: I1; 3'b001, 3'b010, 3'b100: I2; 3'b011, 3'b110, 3'b101: I3; 3'b111: I4. I1 > I2 > I3 > I4	3'b111	Amplifier
3	RfSel[1:0]	Feedback resistance selection. 2'b00: 20 kOHm 2'b01: 10 kOHm 2'b10: 5.7 kOHm 2'b11: 4.4 kOHm	2'b10	Amplifier
4	HysSel[3:0]	Hysteresis voltage selection. 4'b0000: Vhys1 4'b0001: Vhys2 4'b0011: Vhys3 4'b0111: Vhys4 4'b1111: Vhys5 Vhys1 > Vhys2 > Vhys3 > Vhys4 = Vhys5 = 0	4'b1111	Amplifier
5	PD_DACDiscri	Power down the DAC and the discriminator in pixels. When PD_DACDiscri is 1, the DAC and the discriminator are powered down.	1'b0	Amplifier
6	QSel[4:0]	Select injected charge, from 1 fC(5'b00000) to 32 fC(5'b11111).	5'b00110	Amplifier
7	QInjEn	enabling the charge injection of the specified pixel, active high.	1'b0	Amplifier
8	autoReset_TDC	TDC automatically reset controller for every clock period.	1'b0	TDC
9	enable_TDC	TDC enable. 1'b1: enable TDC conversion 1'b0: disable TDC conversion	1'b1	TDC
10	level_TDC[2:0]	The bit width of bubble tolerant in TDC encode. It is up to 3'b011	3'b001	TDC
11	resetn_TDC	Reset TDC encoder, active low	1'b1	TDC
12	testMode_TDC	test mode enable of TDC, active high. In test mode, TDC generates a fixed test pulse as input signal for test for every 25 ns.	1'b0	TDC
13	Bypass_THCal	Bypass control of the in-pixel threshold calibration block. 1: bypassing the in-pixel threshold calibration block. DAC is applied to TH. Users can control the threshold voltage through DAC. 0: calibrated threshold is applied to TH. TH = BL + TH_offset	1'b1	Calibration
14	DAC[9:0]	When THCal_Bypass==1'b1, TH = DAC	10'H000	Calibration
15	TH_offset[5:0]	Threshold offset for the calibrated baseline. TH = BL + TH_offset	6'b001010	Calibration
16	RSTn_THCal	Reset of threshold calibration block, active low.	1'b1	Calibration

Continuation of Table 14

Num.	Register	Description	Default	Default
17	ScanStart_THCal	A rising edge of ScanStart_THCal initializes the threshold calibration	1'b0	Calibration
18	BufEn_THCal	threshold clalibration buffer enable. 1: enabling the buffer between discriminator output and the TH_Ctrl 0: disabling the buffer between discriminator output and the TH_Ctrl	1'b0	Calibration
19	CLKEn_THCal	This register is only used when the threshold calibration block is bypassed. 1: enabling the clock for measuring average discriminator output 0: disabling the clock. Measurement of the average discriminator output is not available.	1'b0	Calibration
20	workMode[1:0]	Readout work mode selection. 00: normal, 01: self test, periodic trigger fixed TDC data 10: self test, random TDC data, 11: reserved	2'b00	TDC
21	L1Adelay[8:0]	L1A latency	9'D501	Readout
22	disDataReadout	Disable signal of the TDC data readout. 1: disabling the TDC data readout of the current pixel 0: enabling the TDC data readout of the current pixel	1'b0	Readout
23	disTrigPath	Disable signal of the trigger readout. 1: disabling the trigger readout of the current pixel 0: enabling the trigger readout of the current pixel	1'b0	Readout
24	upperTOATrig	TOA upper threshold for the trigger readout	10'H200	Readout
25	lowerTOATrig	TOA lower threshold for the trigger readout	10'H020	Readout
26	upperTOTTrig	TOT upper threshold for the trigger readout	9'H040	Readout
27	lowerTOTTrig	TOT lower threshold for the trigger readout	9'H010	Readout
28	upperCalTrig	Cal upper threshold for the trigger readout	10'H200	Readout
29	lowerCalTrig	Cal lower threshold for the trigger readout	10'H010	Readout
30	upperTOA	TOA upper threshold for the TDC data readout	10'H200	Readout
31	lowerTOA	TOA lower threshold for the TDC data readout	10'H020	Readout
32	upperTOT	TOT upper threshold for the TDC data readout	9'H100	Readout
33	lowerTOT	TOT lower threshold for the TDC data readout	9'H010	Readout
34	upperCal	Cal upper threshold for the TDC data readout	10'H200	Readout
35	lowerCal	Cal lower threshold for the TDC data readout	10'H010	Readout
36	addrOffset	Enabling of the circular buffer (CB) write address offset by the pixel ID, active high. 1: enabling the CB write address offset 0: disabling the CB write address offset	1'b1	Readout

37	selfTestOccupancy[6:0]	Self-test data occupancy is selfTestOccupancy[6:0]/128. For example: 1: 1/1.28% 2: 2/1.28% 5: 5/1.28% 10: 10/1.28%	7'D1	Readout
End of Table 14				

In-pixel I2C Registers(Ext Addr./Int Addr.)	BITS								Default(Hex)
	7	6	5	4	3	2	1	0	
PixR<R_n>C<C_n>Cfg0/pixelConfig[7:0]	−−	RfSel[1:0]		IBSel[2:0]			CLSel[1:0]		0x5C
PixR<R_n>C<C_n>Cfg1/pixelConfig[15:8]	−−	−−	QInjEn	QSel[4:0]					0x06
PixR<R_n>C<C_n>Cfg2/pixelConfig[23:16]	−−	−−	−−	PD_DACDiscri	HysSel[3:0]				0x0F
PixR<R_n>C<C_n>Cfg3/pixelConfig[31:24]	−−	−−	−−	ScanStart_THCal	CLKEn_THCal	Bypass_THCal	BufEn_THCal	RSTn_THCal	0x05
PixR<R_n>C<C_n>Cfg4/pixelConfig[39:32]	DAC[7:0]								0x00
PixR<R_n>C<C_n>Cfg5/pixelConfig[47:40]	TH_offset[5:0]						DAC[9:8]		0x28
PixR<R_n>C<C_n>Cfg6/pixelConfig[55:48]	enable_TDC	resetn_TDC	autoReset_TDC	testMode_TDC	level_TDC[2:0]			−−	0xC2
PixR<R_n>C<C_n>Cfg7/pixelConfig[63:56]	−−	−−	−−	workMode[1:0]		disTrigPath	disDataReadout	addrOffset	0x01
PixR<R_n>C<C_n>Cfg8/pixelConfig[71:64]	L1Adelay[0]	selfTestOccupancy[6:0]							0x81
PixR<R_n>C<C_n>Cfg9/pixelConfig[79:72]	L1Adelay[8:1]								0xFA
PixR<R_n>C<C_n>Cfg10/pixelConfig[87:80]	lowerCal[7:0]								0x10
PixR<R_n>C<C_n>Cfg11/pixelConfig[95:88]	upperCal[5:0]						lowerCal[9:8]		0x00
PixR<R_n>C<C_n>Cfg12/pixelConfig[103:96]	lowerTOA[3:0]				upperCal[9:6]				0x08
PixR<R_n>C<C_n>Cfg13/pixelConfig[111:104]	upperTOA[1:0]		lowerTOA[9:4]						0x02
PixR<R_n>C<C_n>Cfg14/pixelConfig[119:112]	upperTOA[9:2]								0x80
PixR<R_n>C<C_n>Cfg15/pixelConfig[127:120]	lowerTOT[7:0]								0x10
PixR<R_n>C<C_n>Cfg16/pixelConfig[135:128]	upperTOT[6:0]							lowerTOT[8]	0x00
PixR<R_n>C<C_n>Cfg17/pixelConfig[143:136]	lowerCalTrig[5:0]						upperTOT[8:7]		0x42
PixR<R_n>C<C_n>Cfg18/pixelConfig[151:144]	upperCalTrig[3:0]				lowerCalTrig[9:6]				0x00
PixR<R_n>C<C_n>Cfg19/pixelConfig[159:152]	lowerTOATrig[1:0]		upperCalTrig[9:4]						0x20
PixR<R_n>C<C_n>Cfg20/pixelConfig[167:160]	lowerTOATrig[9:2]								0x08
PixR<R_n>C<C_n>Cfg21/pixelConfig[175:168]	upperTOATrig[7:0]								0x00
PixR<R_n>C<C_n>Cfg22/pixelConfig[183:176]	lowerTOTTrig[5:0]						upperTOATrig[9:8]		0x42
PixR<R_n>C<C_n>Cfg23/pixelConfig[191:184]	upperTOTTrig[4:0]					lowerTOTTrig[8:6]			0x00
PixR<R_n>C<C_n>Cfg24/pixelConfig[199:192]	−−	−−	−−	−−	upperTOTTrig[8:5]				0x02
PixR<R_n>C<C_n>Cfg25/pixelConfig[207:200]	−−	−−	−−	−−	−−	−−	−−	−−	0x00
PixR<R_n>C<C_n>Cfg26/pixelConfig[215:208]	−−	−−	−−	−−	−−	−−	−−	−−	0x00
PixR<R_n>C<C_n>Cfg27/pixelConfig[223:216]	−−	−−	−−	−−	−−	−−	−−	−−	0x00
PixR<R_n>C<C_n>Cfg28/pixelConfig[231:224]	−−	−−	−−	−−	−−	−−	−−	−−	0x00
PixR<R_n>C<C_n>Cfg29/pixelConfig[239:232]	−−	−−	−−	−−	−−	−−	−−	−−	0x00
PixR<R_n>C<C_n>Cfg30/pixelConfig[247:240]	−−	−−	−−	−−	−−	−−	−−	−−	0x00
PixR<R_n>C<C_n>Cfg31/pixelConfig[255:248]	−−	−−	−−	−−	−−	−−	−−	−−	0x00

Table 15. In-pixel configuration register map.

Num.	Reg. name	Reg. Address	Description	Expected value
1	ACC[15:0]	PixR<R_n>C<C_n>STA6[7:0], PixR<R_n>C<C_n>STA5[7:0]	Accumulator of the threshold calibration	−−
2	ScanDone	PixR<R_n>C<C_n>STA1[0]	Scan done signal of the threshold calibration	−−
3	BL[9:0]	PixR<R_n>C<C_n>STA3[1:0], PixR<R_n>C<C_n>STA2[7:0]	Baseline obtained from threshold calibration	−−
4	NW[3:0]	PixR<R_n>C<C_n>STA1[4:1]	Noise width from threshold calibration	less than 10
5	TH[9:0]	PixR<R_n>C<C_n>STA4[7:0], PixR<R_n>C<C_n>STA3[7:6]	10-bit threshold applied to the DAC input	−−
6	THState[2:0]	PixR<R_n>C<C_n>STA1[7:5]	Threshold calibration state machine output	−−
7	PixelID	PixR<R_n>C<C_n>STA0[7:0]	Col[3:0],Row[3:0]	−−

Table 16. In-pixel I2C status address

Num.	Reg. name	Reg. Address	Description	Expected value
1	fcBitAlignError	PeriSTA2[0]	Bit alignment error	1'b0, see 'error' in figure 47
2	PS_Late	PeriSTA0[7]	phase shifter late	keep changing between 0 and 1 once the phase shifter is locked.
3	AFCcalCap[5:0]	PeriSTA0[6:1]	AFC capacitance	expecting die-to-die vairation
4	AFCBusy	PeriSTA0[0]	AFC busy, 1: AFC is ongoing, 0: AFC is done	0
5	fcAlignFinalState[3:0]	PeriSTA1[7:4]	fast command alignment FSM state	4'd9 is expected at 'Done' state
6	controllerState[3:0]	PeriSTA1[3:0]	global control FSM state	4'd11 is expected at 'Done' state.
7	fcAlignStatus[3:0]	PeriSTA2[7:4]	fast command self-alignment error indicator, ed[3:0] in figure 47	see 'ed*' in figure 47
8	invalidFCCount[11:0]	PeriSTA4[3:0], PeriSTA3[7:0]	count of invalid fast command received	0
9	pllUnlockCount[11:0]	PeriSTA5[7:0], PeriSTA4[7:4]	count of PLL unlock detected	few
10	EFuseQ[31:0]	PeriSTA9[7:0], PeriSTA8[7:0], PeriSTA7[7:0], PeriSTA6[7:0]	32-bit eFuse output	what user write into eFuse. 32'b0 before any eFuse burning

Table 17. Periphery I2C status address

Peripheral Status Address	BITS								Expected Valude(Hex)
	7	6	5	4	3	2	1	0	
PeriStat0	PS_Late	AFCcalCap[5:0]						AFCBusy	– –
PeriStat1	fcAlignFinalState[3:0]				controlerState[3:0]				– –
PeriStat2	fcAlignStatus[3:0]				3'h0			fcBitAlignError	– –
PeriStat3	invalidFCCount[7:0]								– –
PeriStat4	pllUnlockCount[3:0]				invalidFCCount[11:8]				– –
PeriStat5	pllUnlockCount[11:4]								– –
PeriStat6	EFuseQ[7:0]								– –
PeriStat7	EFuseQ[15:8]								– –
PeriStat8	EFuseQ[23:16]								– –
PeriStat9	EFuseQ[31:24]								– –
PeriStat10	8'h0								0x0
PeriStat11	8'h0								0x0
PeriStat12	8'h0								0x0
PeriStat13	8'h0								0x0
PeriStat14	8'h0								0x0
PeriStat15	8'h0								0x0

Table 18. Periphery Status address map.

In-pixel Status Address	BITS								Expected Valude(Hex)
	7	6	5	4	3	2	1	0	
PixR<R_n>C<C_n>STA0	PixelID								−−
PixR<R_n>C<C_n>STA1	THState[2:0]			NW[3:0]				ScanDone	−−
PixR<R_n>C<C_n>STA2	BL[7:0]								−−
PixR<R_n>C<C_n>STA3	TH[1:0]		0	0	0	0	BL[9:8]		−−
PixR<R_n>C<C_n>STA4	TH[9:2]								−−
PixR<R_n>C<C_n>STA5	ACC[7:0]								−−
PixR<R_n>C<C_n>STA6	ACC[15:8]								−−
PixR<R_n>C<C_n>STA7	PixR<R_n>C<C_n>Cfg31/pixelConfig[255:248]								−−

Table 19. In-pixel Status address map.

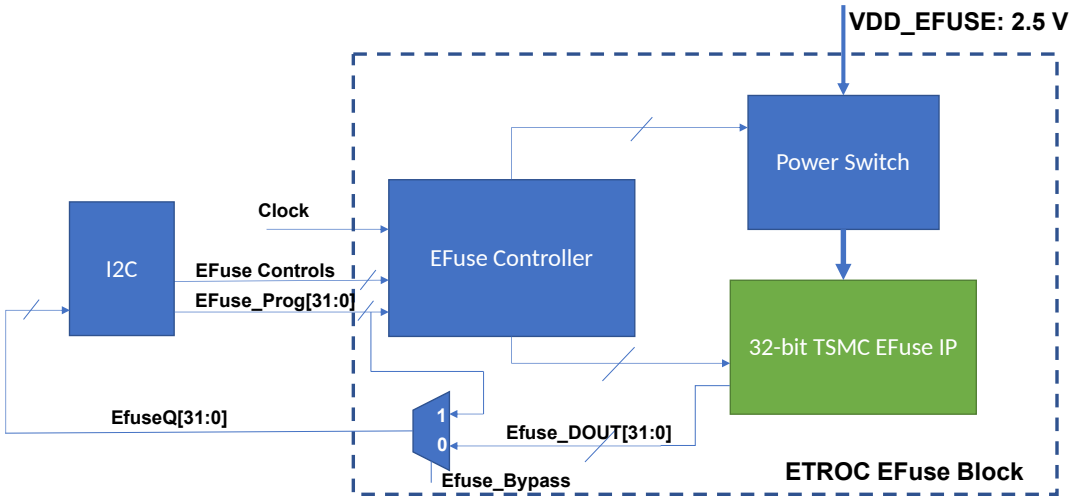


Figure 52. Block diagram of the eFuse block.

3.6.2 eFuse

The ETROC2 includes an eFuse block that stores a 17-bit chip ID. However, based on lpGBT testing results, the eFuse is known to be unreliable during initialization and TID test [6]. The initialization burning error rate is about 5.4e-4 and the failure rate in the TID test is less than 10 bits out of 2048 bits. The actual eFuse block is 32 bits, a Reed-Solomon error correction algorithm is being proposed to correct the potential error bits. The RS(32,17) code is expected to correct up to 7 bits of error. The encoded message is 32 bits and is burned into the eFuse. When the system is powered up, the message is read from ETROC eFuse with I2C, and error correction is done on the back-end computer. The corrected chip ID is then written back to ETROC I2C registers, which are TMR protected. The entire process still needs to be verified in testing. For initial testing, the chip ID can only be specified by I2C.

Figure 52 shows a block diagram of the 32-bit eFuse in ETROC2. An eFuse IP from the foundry is encapsulated into the module named ETROC eFuse block. The other building blocks of the ETROC eFuse block include a power switch, an eFuse controller, and a 32-bit multiplexer.

A 2.5 V power supply is required to program the ETROC eFuse. The eFuse is a one-time programmable device that can be programmed only once. Programming is done bit by bit. When an eFuse bit is not programmed, its representing value is 0. Once it is programmed, the value becomes 1. This feature can be utilized when users want to re-program the eFuse for modifying some bits that were not programmed before. For example, if the eFuse has been programmed to 0xFFFFA330, it can be programmed again to 0xFFFFA335 by programming the additional bits.

Figure 53 illustrates the operation timing diagram of ETROC eFuse block. The signals in the diagram are explained below.

- eFuse_Rstn: reset of eFuse controller, active low.

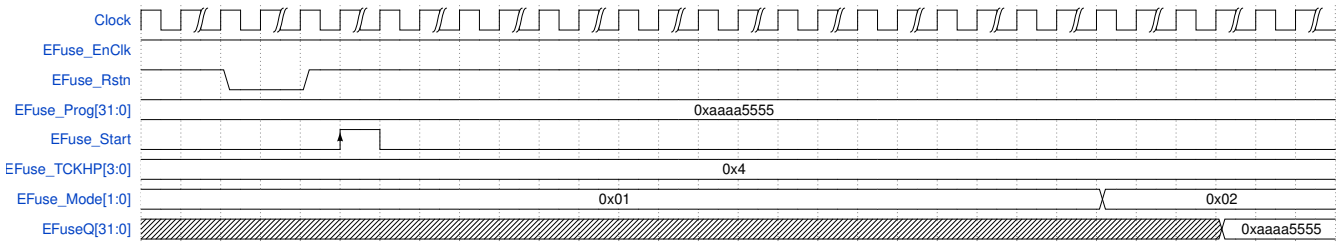


Figure 53. Timing diagram of ETROC eFuse block operation.

- eFuse_EnClk: enable of the clock, active high.
- eFuse_Start: a rising edge initialize the programming
- eFuse_Mode: mode select
  - 2'b01: programming mode
  - 2'b10: reading mode
- eFuse_Prog[31:0]: 32-bit data to be programmed
- EFuseQ: output of ETROC eFuse block
- eFuse_Bypass: bypassing the eFuse block, active high.
- eFuse_TCKHP[3:0]: programming pulse width, the users should use the default value, **debugging use only**.

Figure 54 shows the operation flow of the eFuse.

3.7 Power up Sequence

This section provides information of the power supply of ETROC2 excluding the waveform sampler. The waveform sampler is described in a later section since it has separate power supply and external control. The power up sequence is also described in this section.

3.7.1 Power Domain

The ETROC2 employs multiple power supplies on chip to reduce power supply noise coupling among building blocks in the chip. These power supplies are organized into two groups, analog and digital, from the users point of view. Table 20 summarizes the on-chip power domain and off-chip power group. Note that the discriminator power is shared by the discriminators and the TDCs in the left-half pixel matrix. In the right-half pixel matrix, the TDCs are supplied by the digital power. It's recommended to separate analog and digital power supplies on the test board to get the best performance. Sensor (negative) bias voltage's reference ground can be tied to PCB analog ground plane or use a dedicated ground plane.

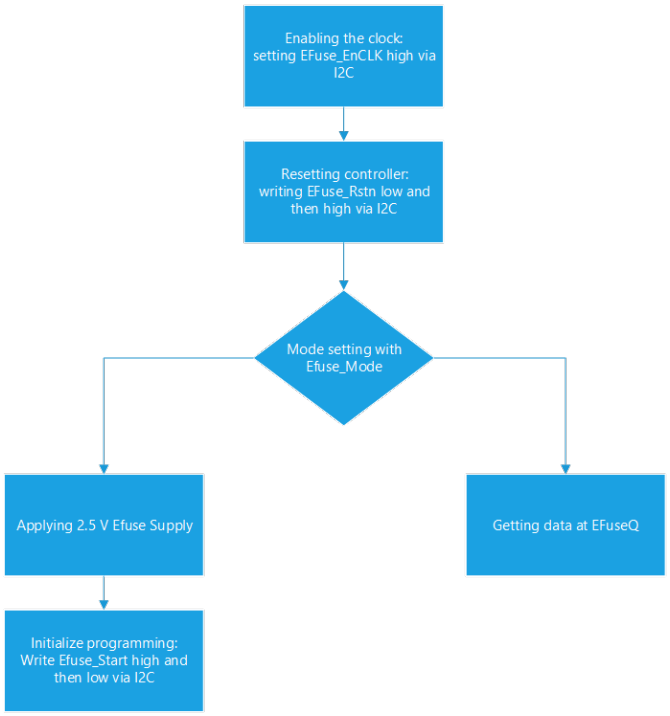


Figure 54. eFuse operation flow.

3.7.2 Power Consumption

Table 21 gives a rough expectation of power consumption at nominal condition. The actual power consumption may vary up to ±20% due to process variation.

3.7.3 Power up Sequence

When involving LGAD sensor,ETROC2 should be powered up before applying HV to the LGAD sensor. After powering up the ETROC2, the preamplifiers establish the operation point at their input node, which prevent ETROC2 from being damaged due to potential exposure to high voltage.

A module named globalController is implemented to control the power up sequence of the ETROC2. Figure 55 shows a diagram including the simplified power up flow and the relevant modules. The power up sequence is briefly described below.

1. When power is up, a power-on-reset (POR) block generates a reset pulse to reset the glocalController to Init state.
2. The PLL is reset in resetPLL state.
3. globalController generates pulses to calibrate the VCO in the PLL in calibratePLL state.
4. Waiting for locking of the PLL in lockDetect state .

Power Domain	Group	Description
PA Power	Analog	Preamplifier power
QInj Power	Analog	Charge injector power
Discriminator power	Analog	This power domain supplies: 1. the discriminators and TDCs in the left-half pixel matrix, 2. The discriminators in the right-half pixel matrix. The TDCs on the right-half pixel matrix use digital supply.
Serial link power	Digital	Tx and Rx power
Clock power	Digital	Clock generator power
Digital power	Digital	Digital power

Table 20. Power Domain Summary

Circuit component	Power consumption per ASIC(mW)
Preamplifier	190 (lower power setting), 390 (high power setting)
Discriminator	215
TDC	26
Pixel readout	41
Rest of blocks in pixels	51
Global readout	100
Clock generator	70
H-tree	27
rest of global blocks	50
Total	770 (PA lower power setting), 970 (PA high power setting)

Table 21. Power consumption estimate.

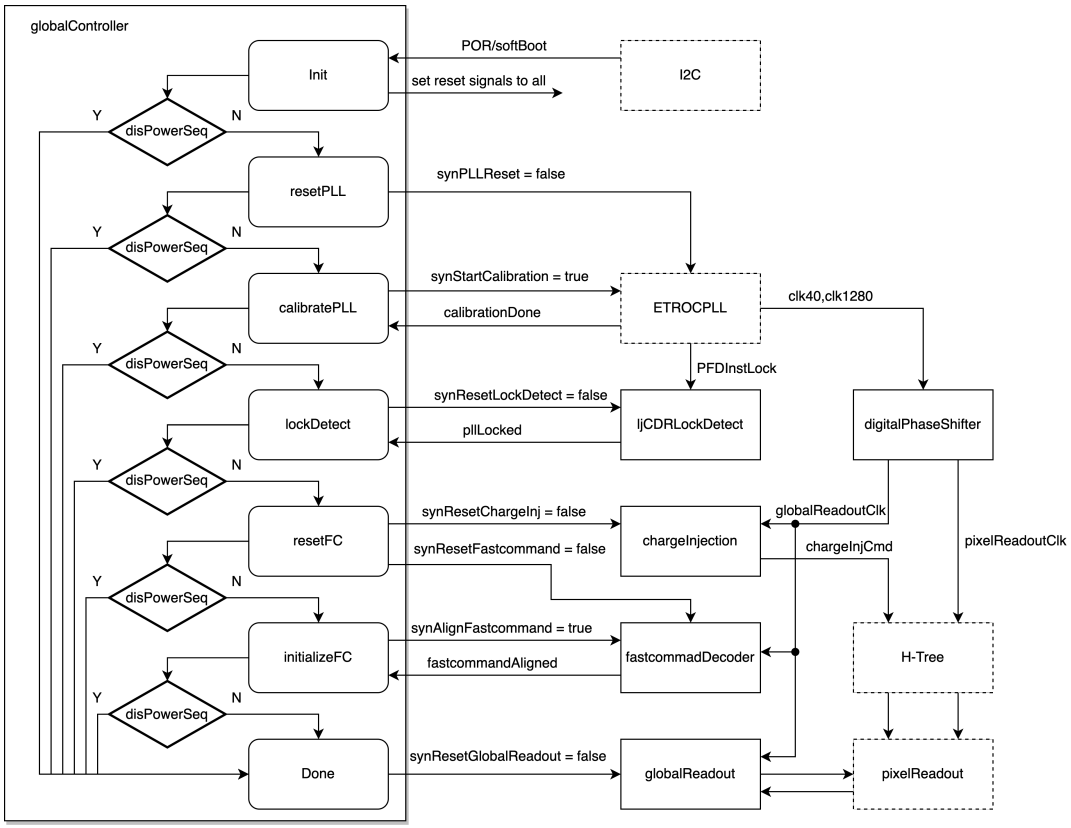


Figure 55. Diagram of ETROC2 power up.

5. The charge injection pulse generator and the fast command decoder are reset in resetFC state.

6. Fast command self-alignment is performed in initializeFC state. This state is reserved for ETROC3. In ETROC2, manual alignment is set by default, and initializeFC state is skipped.

7. globalReadout module is ready in Done state.

The automatic power up sequence can be disabled by setting high to the I2C register bit disPowerSequence. The I2C register bit softBoot is used to set the power up sequence to Init state.

The power up sequence showing in figure 55 has been verified with Universal Verification Methodology (UVM).

3.8 Temperature Sensor

The ETROC2 includes an analog temperature sensor which allows the users to monitor ambient temperature change. The temperature sensor provides an analog voltage output which represents the temperature. The analog voltage can be digitized by the Analog-to-Digital Converter (ADC) in lpGBT. Figure 56 shows a simplified schematic of the temperature sensor.

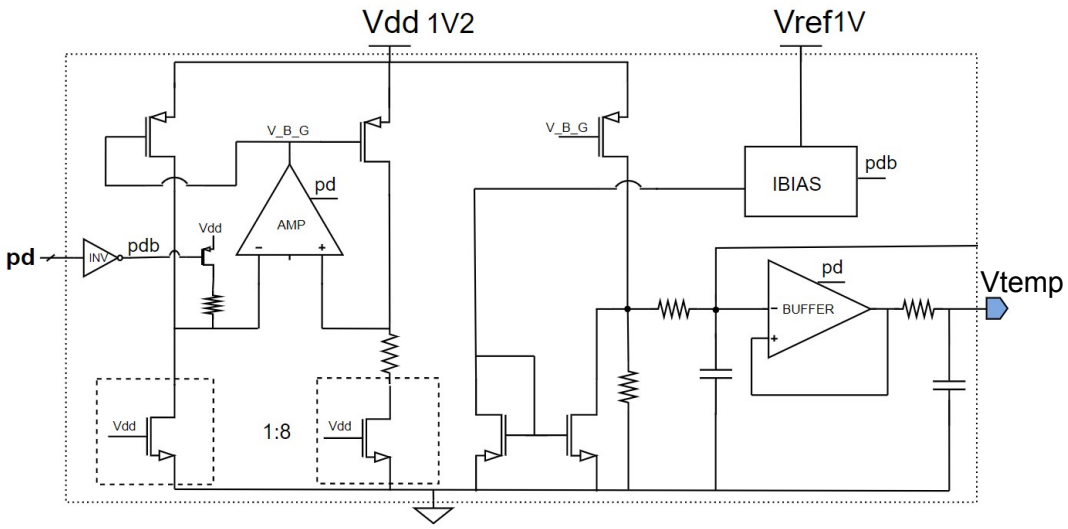


Figure 56. Simplified schematic of the temperature sensor

Ideally, the output voltage can be expressed as:

$$V_{temp} = C_1 \cdot kT/q + C_2, \tag{13}$$

where C_1 is a constant, C_2 is a constant voltage, k is the Boltzmann constant, q is the charge of an electron and T is the temperature in Kelvin. The simulated C_1 and C_2, are 26 and -0.0073 V under typical conditions, respectively. However, in a real chip they may vary from die to die. Therefore, a two-point calibration is required to get the temperature from the temperature sensor for each ETROC2 die.

Simulation suggested that the variation of C_1 from die to die might be small enough. In this case a single-point calibration is sufficient. More studies on the temperature sensor are expected when testing ETROC2 dies.

The temperature sensor has been prototyped in a test chip. Figure 57 shows the results of the temperature sensor.

3.9 Voltage Reference

The ETROC2 generates a 1 V voltage reference in the peripheral and delivers it to all the pixels. Figure 58 shows a block diagram of the voltage reference generation and delivery. A bandgap reference block generates a 200 mV bandgap voltage. The band gap voltage is then filtered and amplified to 1 V before being delivered to the pixels In each pixel, a unity-gain buffer takes the 1 V voltage reference and drives the charge injector and the DAC.

The 1 V voltage reference from the peripheral can be monitored off-chip. The user can provide an off-chip voltage reference to VREF pad directly if the on-chip voltage reference generator has issue.

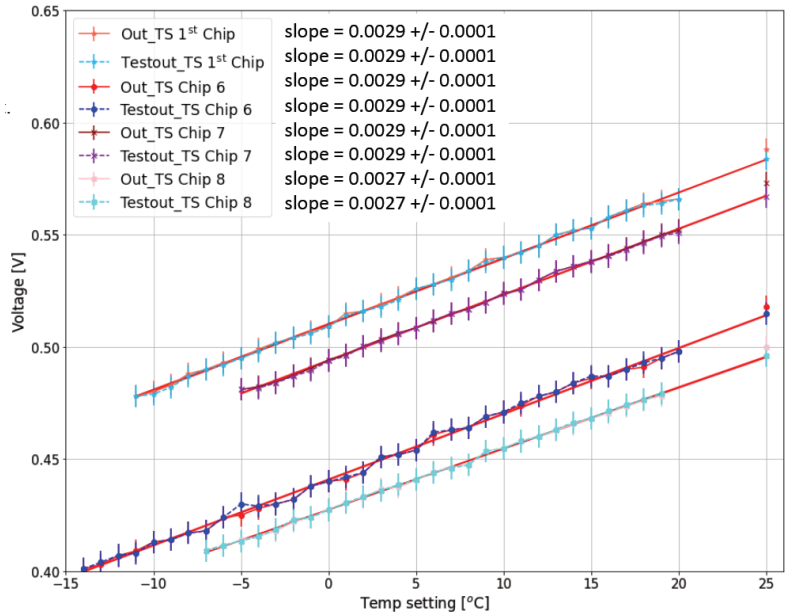


Figure 57. Measurement results of the temperature sensor.

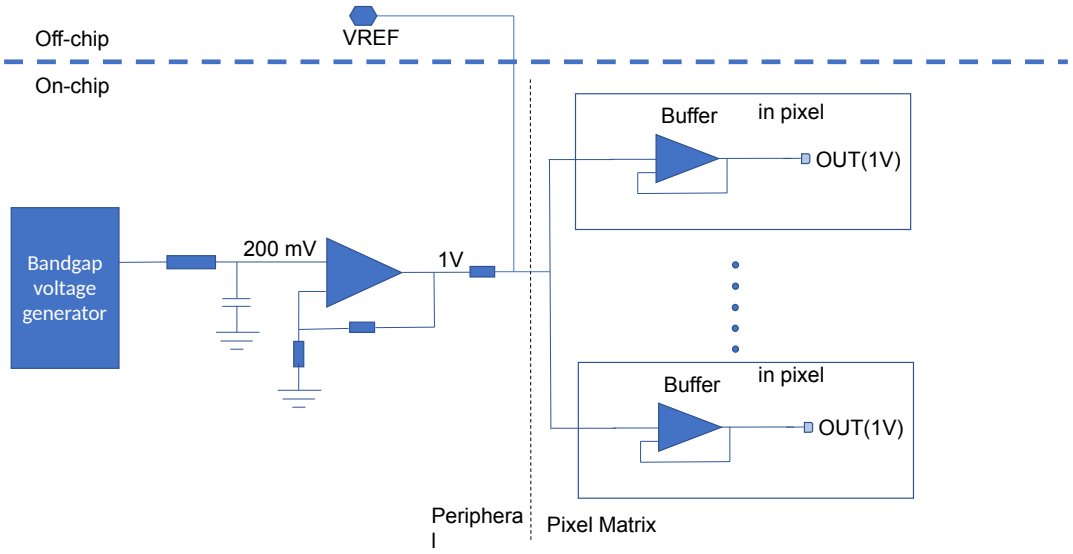


Figure 58. Block diagram of the voltage reference generation and delivery.

4 Pinout

The main ETROC2 (the ETROC2 excluding the waveform sampler) and the waveform sampler have separate pinout naming and numbering conventions. Figure 59(a) illustrates the pinout numbering of both the main ETROC2 and the waveform sampler.

This section gives pinout information of the main ETROC2 and the waveform sampler in the following two subsections, respectively. All the coordinates given are based on the assumption that the bottom left corner of the die is origin(0, 0).

4.1 Main ETROC2 Pinout

The main ETROC2 has 124 rectangular pads on the bottom of the die with a uniform pad pitch of $143\ \mu m$. The rectangular pads' opening is $70 \times 200\ \mu m^2$, and the edge-to-edge space is $73\ \mu m$. These pads are design to be long rectangular for two purposes.

1. The lower part of the pads are used for wire-bonding.
2. The upper part of the pads can be used during wafer probe testing.

Each of these rectangular pad has an associated octagon pad, as shown in figure 59(b). Connected to the associated rectangular pads in the chip, the octagon pads provide possibility of bump-bonding for ETROC2 in case the performance degrade due to the wire-bonding inductance.

The GRO has a dedicated octagon pad at bottom right corner of the die. There is no associated wire-bonding pad for the GRO.

Table 22 summarizes the pads of the main ETROC2.

Table 22. Main ETROC2 Pads

Begin of Table 22

Num.	Internal Name	External Name	Category	Description	rectangular pad center coordinates (μm)	octagon pad center coordinates (μm)
1	VSS_PA	VSS_A	Power	Preamplifier ground, 0V	(3100, 155)	(3100, 505)
2	VSS_PA	VSS_A	Power	Preamplifier ground, 0V	(3243, 155)	(3243, 723)
3	VSS_PA	VSS_A	Power	Preamplifier ground, 0V	(3386, 155)	(3386, 505)
4	VSS_PA	VSS_A	Power	Preamplifier ground, 0V	(3529, 155)	(3529, 723)
5	VSS_PA	VSS_A	Power	Preamplifier ground, 0V	(3672, 155)	(3672, 505)
6	VSS_IO1	VSS_A	Power	analog ESD ground, 0 V	(3815, 155)	(3672, 723)
7	VREF	VREF	Analog Input/output	voltage reference, 1V. Monitoring the on-chip reference or receiving from external reference, or providing 1 V off-chip reference	(3958, 155)	(3958, 505)
8	VDD_PA	VDD_A	Power	Preamplifier supply, nominally 1.2 V	(4101, 155)	(4101, 723)
9	VDD_PA	VDD_A	Power	Preamplifier supply, nominally 1.2 V	(4244, 155)	(4244, 505)
10	VDD_PA	VDD_A	Power	Preamplifier supply, nominally 1.2 V	(4387, 155)	(4387, 723)
11	VDD_PA	VDD_A	Power	Preamplifier supply, nominally 1.2 V	(4530, 155)	(4530, 505)
12	VDD_PA	VDD_A	Power	Preamplifier supply, nominally 1.2 V	(4673, 155)	(4673, 723)

Continuation of Table 22

Num.	Internal Name	External Name	Category	Description	rectangular pad center coordinates (μm)	octagon pad center coordinates (μm)
13	VDD_QInj	VDD_A	Power	Charge injector supply, nominally 1.2 V	(4816, 155)	(4816, 505)
14	VSS_IO1	VSS_A	Power	analog ESD ground, 0 V	(4959, 155)	(4959, 723)
15	VSS_QInj	VSS_A	Power	Charge injector ground, 0 V	(5102, 155)	(5102, 505)
16	VSS_Dis	VSS_A	Power	Discriminator ground, 0 V	(5245, 155)	(5245, 723)
17	VSS_Dis	VSS_A	Power	Discriminator ground, 0 V	(5388, 155)	(5388, 505)
18	VSS_Dis	VSS_A	Power	Discriminator ground, 0 V	(5531, 155)	(5531, 723)
19	VSS_Dis	VSS_A	Power	Discriminator ground, 0 V	(5674, 155)	(5674, 505)
20	VSS_Dis	VSS_A	Power	Discriminator ground, 0 V	(5817, 155)	(5817, 723)
21	VSS_IO1	VSS_A	Power	Analog ESD ground, 0 V	(5960, 155)	(5960, 505)
22	VDD_Dis	VDD_A	Power	Discriminator supply, nominally 1.2 V	(6103, 155)	(6103, 723)
23	VDD_Dis	VDD_A	Power	Discriminator supply, nominally 1.2 V	(6206, 155)	(6206, 505)
24	VDD_Dis	VDD_A	Power	Discriminator supply, nominally 1.2 V	(6389, 155)	(6389, 723)
25	VDD_Dis	VDD_A	Power	Discriminator supply, nominally 1.2 V	(6532, 155)	(6532, 505)
26	VDD_Dis	VDD_A	Power	Discriminator supply, nominally 1.2 V	(6675, 155)	(6675, 723)
27	VSS_S	VSS_A	Power	Ground for substrate, 0 V	(6818, 155)	(6818, 723)
28	VSS_S	VSS_A	Power	Ground for substrate, 0 V	(6916, 155)	(6916, 505)
29	CLK40p	CLK40p	Analog input	40 MHz reference clock, positive	(7104, 155)	(7104, 723)
30	VSS_SL	VSS_D	Power	Serial link ground, 0 V	(7247, 155)	(7247, 505)
31	CLK40n	CLK40n	Analog input	40 MHz reference clock, negative	(7390, 155)	(7390, 723)
32	VSS_SL	VSS_D	Power	Serial link ground, 0 V	(7533, 155)	(7533, 505)
33	CLK1280p	CLK1280p	Analog input	optional 1.28 GHz, positive. use when the PLL fail or test with exteral 1.28 GHz clock	(7676, 155)	(7676, 723)
34	VSS_SL	VSS_D	Power	Serial link ground, 0 V	(7819, 155)	(7819, 505)
35	CLK1280n	CLK1280n	Analog input	optional 1.28 GHz, negative. use when the PLL fail or test with exteral 1.28 GHz clock	(7962, 155)	(7962, 723)
36	VSS_IO2	VSS_D	Power	Digital IO/ESD ground, 0 V	(8105, 155)	(8105, 505)
37	FCp	FC40p	Analog input	Fast command input, positive	(8248, 155)	(8248, 723)
38	VDD_SL	VDD_D	Power	Serial link supply, nominally 1.2 V	(8391, 155)	(8391, 505)
39	FCn	FCn	Analog input	Fast command input, negative	(8534, 155)	(8534, 723)
40	VDD_SL	VDD_D	Power	Serial link supply, nominally 1.2 V	(8677, 155)	(8677, 505)
41	DOLp	DOLp	Analog output	Left side data output port, positive	(8820, 155)	(8820, 723)
42	VDD_SL	VDD_D	Power	Serial link supply, nominally 1.2 V	(8963, 155)	(8963, 505)
43	DOLn	DOLn	Analog output	Left side data output port, negative	(9106, 155)	(9106, 723)
44	VSS_IO2	VSS_D	Power	Digital IO/ESD ground, 0 V	(9249, 155)	(9249, 505)
45	DORp	DORp	Analog output	Right side data output port, positive	(9392, 155)	(9392, 723)
46	VSS_IO2	VSS_D	Power	Digital IO/ESD ground, 0 V	(9535, 155)	(9535, 505)
47	DORn	DORn	Analog output	Right side data output port, negative	(9678, 155)	(9678, 723)
48	VSS_CLK	VSS_D	Power	Clock generator ground, 0 V	(9821, 155)	(9821, 505)
49	VSS_CLK	VSS_D	Power	Clock generator ground, 0 V	(9964, 155)	(9964, 723)
50	VSS_CLK	VSS_D	Power	Clock generator ground, 0 V	(10107, 155)	(10107, 505)
51	VP	VP	Analog Input/output	VCO supply stablization pin. Connecting to an external capacitor for improving PLL clock jitter	(10250, 155)	(10250, 723)
52	VDD_CLK	VDD_D	Power	Clock generator supply, nominally 1.2 V	(10393, 155)	(10393, 505)
53	VDD_CLK	VDD_D	Power	Clock generator supply, nominally 1.2 V	(10536, 155)	(10536, 723)
54	VDD_CLK	VDD_D	Power	Clock generator supply, nominally 1.2 V	(10679, 155)	(10679, 505)
55	VSS_IO2	VSS_D	Power	Digital IO/ESD ground, 0 V	(10679, 155)	(10679, 723)
56	VSS_D	VSS_D	Power	Digital core ground, 0 V	(10965, 155)	(10965, 505)
57	VSS_D	VSS_D	Power	Digital core ground, 0 V	(11108, 155)	(11108, 723)
58	VSS_D	VSS_D	Power	Digital core ground, 0 V	(11251, 155)	(11251, 505)
59	VSS_IO2	VSS_D	Power	Digital IO/ESD ground, 0 V	(11394, 155)	(11394, 723)
60	VDD_IO2	VDD_D	Power	Digital IO/ESD supply, nominally 1.2 V	(11537, 155)	(11537, 505)
61	VDD_D	VDD_D	Power	Digital core supply, nominally 1.2 V	(11680, 155)	(11680, 723)
62	VDD_D	VDD_D	Power	Digital core supply, nominally 1.2 V	(11823, 155)	(11823, 505)
63	VDD_D	VDD_D	Power	Digital core supply, nominally 1.2 V	(11966, 155)	(11966, 723)
64	VDD_D	VDD_D	Power	Digital core supply, nominally 1.2 V	(12109, 155)	(12109, 505)

Continuation of Table 22

Num.	Internal Name	External Name	Category	Description	rectangular pad center coordinates (μm)	octagon pad center coordinates (μm)
65	VDD_D	VDD_D	Power	Digital core supply, nominally 1.2 V	(12252, 155)	(12252, 723)
66	VDD_D	VDD_D	Power	Digital core supply, nominally 1.2 V	(12395, 155)	(12395, 505)
67	VDD_IO2	VDD_D	Power	Digital IO/ESD supply, nominally 1.2 V	(12538, 155)	(12538, 723)
68	VSS_IO2	VSS_D	Power	Digital IO/ESD ground, 0 V	(12681, 155)	(12681, 505)
69	VSS_D	VSS_D	Power	Digital core ground, 0 V	(12824, 155)	(12824, 723)
70	VSS_D	VSS_D	Power	Digital core ground, 0 V	(12967, 155)	(12967, 505)
71	VSS_D	VSS_D	Power	Digital core ground, 0 V	(13110, 155)	(13110, 723)
72	VDD_D	VDD_D	Power	Digital core supply, nominally 1.2 V	(13253, 155)	(13253, 505)
73	VDD_D	VDD_D	Power	Digital core supply, nominally 1.2 V	(13396, 155)	(13396, 723)
74	VDD_D	VDD_D	Power	Digital core supply, nominally 1.2 V	(13539, 155)	(13539, 505)
75	VDD_D	VDD_D	Power	Digital core supply, nominally 1.2 V	(13682, 155)	(13682, 723)
76	VDD_D	VDD_D	Power	Digital core supply, nominally 1.2 V	(13825, 155)	(13825, 505)
77	VDD_D	VDD_D	Power	Digital core supply, nominally 1.2 V	(13969, 155)	(13968, 723)
78	VDD_IO2	VDD_D	Power	Digital IO/ESD supply, nominally 1.2 V	(14111, 155)	(14111, 505)
79	VSS_IO2	VSS_D	Power	Digital IO/ESD ground, 0 V	(14397, 155)	(14397, 723)
80	VSS_D	VSS_D	Power	Digital core ground, 0 V	(14397, 155)	(14397, 505)
81	VSS_D	VSS_D	Power	Digital core ground, 0 V	(14540, 155)	(14540, 723)
82	VSS_D	VSS_D	Power	Digital core ground, 0 V	(14683, 155)	(14683, 505)
83	I2CAddr0	I2CAddr0	Digital Input	I2C address	(14826, 155)	(14826, 723)
84	I2CAddr1	I2CAddr1	Digital Input	I2C address	(14969, 155)	(14969, 505)
85	I2CAddr2	I2CAddr2	Digital Input	I2C address	(15112, 155)	(15112, 723)
86	RSTn	RSTn	Digital Input	I2C reset, active low	(15255, 155)	(15255, 505)
87	VSS_IO2	VSS_D	Power	Digital IO/ESD ground, 0 V	(15398, 155)	(15398, 723)
88	VDD_IO2	VDD_D	Power	Digital IO/ESD supply, nominally 1.2 V	(15541, 155)	(15541, 505)
89	SCL	SCL	Digital input	I2C clock, nominally 400 kHz	(15684, 155)	(15684, 723)
90	SDA	SDA	Digital input/output	I2C data	(15827, 155)	(15827, 505)
91	I2CAddr3	I2CAddr3	Digital Input	I2C address	(15979, 155)	(15970, 723)
92	I2CAddr4	I2CAddr4	Digital Input	I2C address	(16113, 155)	(16113, 505)
93	VSS_D	VSS_D	Power	Digital core ground, 0 V	(16256, 155)	(16256, 723)
94	VSS_D	VSS_D	Power	Digital core ground, 0 V	(16399, 155)	(16399, 505)
95	VSS_D	VSS_D	Power	Digital core ground, 0 V	(16542, 155)	(16542, 723)
96	VDD_EFUSE	VDD_EFUSE	Power	EFuse supply, nominally 2.5 v	(16685, 155)	(16685, 505)
97	VSS_S	VSS_A	Power	Ground for substrate, 0 V	(16828, 155)	(16828, 723)
98	VSS_S	VSS_A	Power	Ground for Aluminum shielding layer on the top of the chip, 0 V	(16971, 155)	(16971, 505)
99	VSS_Dis	VSS_A	Power	Discriminator ground, 0 V	(17114, 155)	(17114, 723)
100	VSS_Dis	VSS_A	Power	Discriminator ground, 0 V	(17257, 155)	(17257, 505)
101	VSS_Dis	VSS_A	Power	Discriminator ground, 0 V	(17400, 155)	(17400, 723)
102	VSS_Dis	VSS_A	Power	Discriminator ground, 0 V	(17543, 155)	(17543, 505)
103	VSS_Dis	VSS_A	Power	Discriminator ground, 0 V	(17686, 155)	(17686, 723)
104	VSS_IO3	VSS_A	Power	Analog ESD ground, 0 V	(17829, 155)	(17829, 505)
105	VDD_Dis	VDD_A	Power	Discriminator supply, nominally 1.2 V	(17972, 155)	(17972, 723)
106	VDD_Dis	VDD_A	Power	Discriminator supply, nominally 1.2 V	(18115, 155)	(18115, 505)
107	VDD_Dis	VDD_A	Power	Discriminator supply, nominally 1.2 V	(18258, 155)	(18258, 723)
108	VDD_Dis	VDD_A	Power	Discriminator supply, nominally 1.2 V	(18401, 155)	(18401, 505)
109	VDD_Dis	VDD_A	Power	Discriminator supply, nominally 1.2 V	(18544, 155)	(18544, 723)
110	VSS_QInj	VSS_A	Power	Charge injector ground, 0 V	(18687, 155)	(18687, 505)
111	VSS_IO3	VSS_A	Power	Analog ESD ground, 0 V	(18830, 155)	(18830, 723)
112	VDD_QInj	VDD_A	Power	Charge injector supply, nominally 1.2 V	(18973, 155)	(18973, 505)
113	VDD_PA	VDD_A	Power	Preamplifier supply, nominally 1.2 V	(19116, 155)	(19116, 723)
114	VDD_PA	VDD_A	Power	Preamplifier supply, nominally 1.2 V	(19259, 155)	(19259, 505)
115	VDD_PA	VDD_A	Power	Preamplifier supply, nominally 1.2 V	(19402, 155)	(19402, 723)
116	VDD_PA	VDD_A	Power	Preamplifier supply, nominally 1.2 V	(19545, 155)	(19545, 505)
117	VDD_PA	VDD_A	Power	Preamplifier supply, nominally 1.2 V	(19688, 155)	(19688, 723)
118	VSS_IO3	VSS_A	Power	Analog ESD ground, 0 V	(19831, 155)	(19831, 505)
119	VTemp	VTemp	Analog Output	Temperature sensor output	(19974, 155)	(19974, 723)

No.	Name	Type	Description	Typical Value	center coordinates (μm)
1	DVSS	Digital ground	Digital ground	0 V	(300, 100)
2	DVDD	Digital power	Digital power	1.2 V V	(430, 100)
3	I2C_addr[4]	I2C address pads	I2C address	N/A	(560, 100)
4	I2C_addr[3]	I2C address pads	I2C address	N/A	(690, 100)
5	I2C_addr[2]	I2C address pads	I2C address	N/A	(820,100)
6	I2C_addr[1]	I2C address pads	I2C address	N/A	(950,100)
7	I2C_addr[0]	I2C address pads	I2C address	N/A	(1080,100)
8	WR_EN_EXT	Digital input	External write enable signal	1.2 V	(1210, 100)
9	DVSS	Digital ground	Digital ground	0 V	(1340, 100)
10	DVDD	Digital power	Digital power	1.2 V	(1470, 100)
11	AVSS	Analog ground	Analog ground	0 V	(1600, 100)
12	AVDD	Analog power	Analog power	1.2 V	(1730, 100)
13	AVSS	Analog ground	Analog ground	0 V	(1860, 100)
14	AVDD	Analog power	Analog power	1.2 V	(1990, 100)
15	SCL	I2C pads	I2C pads	N/A	(2120, 100)
16	SDA	I2C pads	I2C pads	N/A	(2250, 100)
17	RST	I2C pads	I2C pads	N/A	(2380, 100)
18	AVSS	Analog ground	Analog ground	0 V	(2510, 100)
19	AVDD	Analog power	Analog power	1.2 V	(2640, 100)
20	AVSS	Analog ground	Analog ground	0 V	(2770, 100)
21	AVDD	Analog power	Analog power	1.2 V	(2900, 100)

Table 23. Waveform sampler pads.

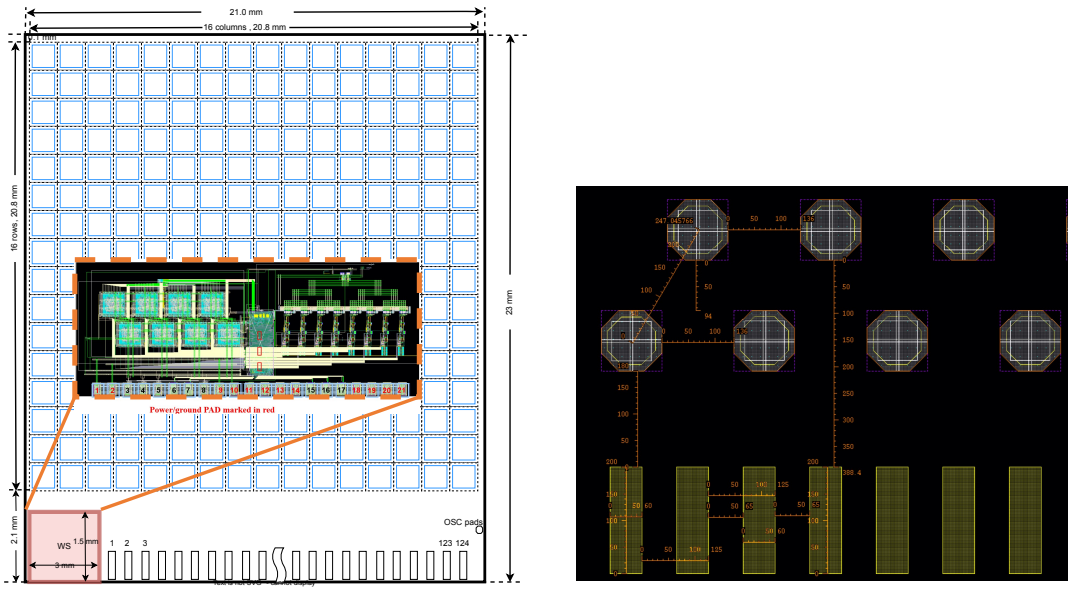
Continuation of Table 22

Num.	Internal Name	External Name	Category	Description	rectangular pad center coordinates (μm)	octagon pad center coordinates (μm)
120	VSS_PA	VSS_A	Power	Preamplifier ground, 0V	(20117, 155)	(20117, 505)
121	VSS_PA	VSS_A	Power	Preamplifier ground, 0V	(20260, 155)	(20260, 723)
122	VSS_PA	VSS_A	Power	Preamplifier ground, 0V	(20403, 155)	(20403, 505)
123	VSS_PA	VSS_A	Power	Preamplifier ground, 0V	(20546, 155)	(20546, 723)
124	VSS_PA	VSS_A	Power	Preamplifier ground, 0V	(20689, 155)	(20689, 505)

End of Table 22

4.2 Waveform Sampler Pinout

The waveform sampler has 21 pads on the bottom, as shown in figure 59(a). The openings of all the pads are $70 \times 116\ \mu m^2$, and the pitch is $130\ \mu m$. Table 23 summarizes the pads of the waveform sampler.



(a) (b)

Figure 59. (a)ETROC2 pinout numbering, (b) An illustration of the main ETROC2 wire-bonding pads and bump-bonding pads.

5 Operation Guide

5.1 A Quick Reference Operation Flow

A reference operation flow is given below.

1. provide reference clock and fast command (IDLEs).
2. power up ETROC2.
3. configure ETROC2 with I2C
   - measurement window
   - dual port or single port
   - trigger granularity
   - data link
     (a) select data rate
     (b) enable or disable the scrambler
     (c) check link with link reset command.
4. fast command alignment
5. threshold calibration
6. ready to take LGAD signal or charge injection

5.2 An Operation Example with Charge Injection

An operation example of simulation with charge injection is given in listing 2.

Listing 2: A Verilog example for simulation with charge injection.

```
#148    // write 0x80 to phase shifter
periConfig = 8'h80;
periConfigAddrIn = 5'b00101 ;
i2c_write_single(SlaveAddr, periConfigAddrIn | 16'h0, periConfig, 1'b1);

#148    // write 0x1 to disScrambler, to disable the scrambler
periConfig = 8'h57;
periConfigAddrIn = 5'b10011 ;
i2c_write_single(SlaveAddr, periConfigAddrIn | 16'h0, periConfig, 1'b1);

#120_000        // waiting for power up sequence done

// writing DAC of 266 to pixels with broadcast, to set the VTH to around 702.4 mV
#1_000
pixel_x = 4'hf;
pixel_y = 4'hf;
pixelConfig = 8'h08;
pixelConfigAddrIn = 5'h04;
pixel_bc(pixel_x, pixel_y, pixelConfig, pixelConfigAddrIn);
pixelConfig = 8'h29;
pixelConfigAddrIn = 5'h05;
pixel_bc(pixel_x, pixel_y, pixelConfig, pixelConfigAddrIn);

// writing 0 to QInjEn with broadcast
#15_000
pixel_x = 4'hf;
pixel_y = 4'hf;
pixelConfig = 8'h06;
pixelConfigAddrIn = 5'h01;
pixel_bc(pixel_x, pixel_y, pixelConfig, pixelConfigAddrIn);

// writing Pixel255(Row15, Col15): 1 to QInjEn
#1_000
pixel_x = 4'hf;
pixel_y = 4'hf;
pixelConfig = 8'h26;
pixelConfigAddrIn = 5'h01;
pixel_write(pixel_x, pixel_y, pixelConfig, pixelConfigAddrIn);

// writing Pixel18(Row2, Col1): 1 to QInjEn, 24 to QSel
#1_000
pixel_x = 4'h2;
pixel_y = 4'h1;
pixelConfig = 8'h38;
pixelConfigAddrIn = 5'h01;
pixel_write(pixel_x, pixel_y, pixelConfig, pixelConfigAddrIn);
```

```
// writing Pixel125(Row14, Col14): 1 to QInjEn, 31 to QSel
#1_000
pixel_x = 4'he;
pixel_y = 4'he;
pixelConfig = 8'h3f;
pixelConfigAddrIn = 5'h01;
pixel_write(pixel_x, pixel_y, pixelConfig, pixelConfigAddrIn);
```

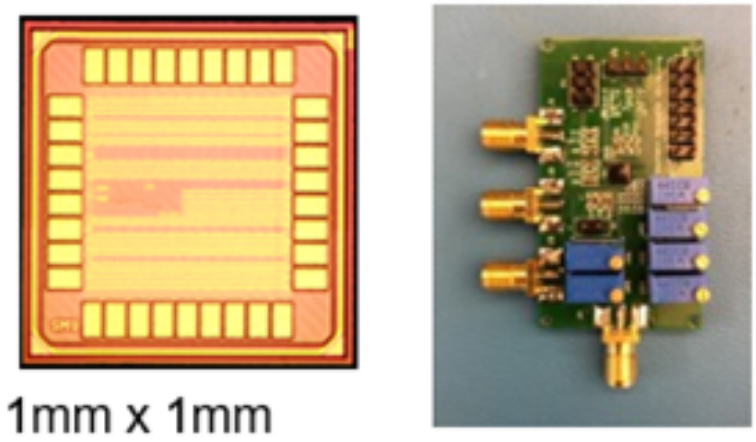


Figure 60. Taped-out SAR ADC die and its test board in 2020.

6 Waveform Sampler

6.1 Introduction

The waveform sampler (Waveform Sampler (WS)) is developed as part of the Endcap Timing Readout Chip (ETROC) for the CMS MTD Endcap Timing Layer (ETL) for the HL-LHC. One of the ETROC 16×16 pixels (pixel 224) is equipped with the WS to sample the pixel's preamplifier output waveform at 2.56GS/s and convert it into digital domain. This is similar to using an oscilloscope to record the waveform except this is on chip. The reconstructed waveforms will be used for periodically LGAD signal monitoring purposes during the CMS ETL detector operation.

The development of the WS chip consists of three versions [9] [10]. The first version (taped-out in 2020) is a 12-bit 320 MS/s single-channel ADC. Pipeline-SAR ADC architecture is implemented as the pipeline structure improves the overall conversion speed and alleviates the noise and linearity requirement for the LSBs. In addition, the SAR architecture is mostly digital, which benefits from technology scaling and has high power-efficiency. Implemented with 65nm CMOS technology process, the single-channel ADC can operate up to 350MS/s and achieve an Effective-Number-of-Bits (ENOB) of 9.6 Bits at Nyquist input frequency (170 MHz). The measured power consumption at 350 MS/s is 2.86 mW. The chip photo and the test board is shown in Figure 60.

The second version (taped-out in 2021) consists of an 8-channel Time-Interleaved ADC and an on-chip memory. By interleaving 8 single-channel sub-ADCs, each operating at 320 MS/s, the 8-channel ADC achieves a total sampling rate of 2.56 GS/s. The on-chip memory can store the digital outputs of 8192 samples from the 2.56 GS/s ADC. At 2.56 GS/s, the measured ADC achieves an ENOB of 9.0 Bits at the Nyquist input (1.2 GHz). The chip photo and the test board is shown in Figure 61

The third version (taped-out in 2022) is the rad-hard version that is integrated inside the ETROC2 chip. An amplifier with high linearity has been implemented in the WS chip to provide

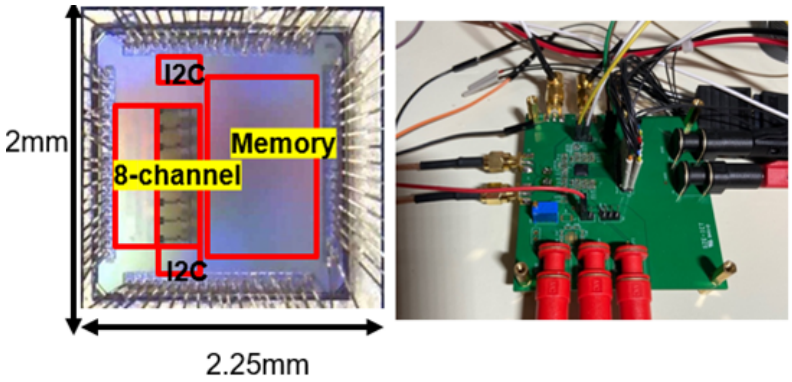


Figure 61. Taped-out SAR ADC die and its test board in 2021.

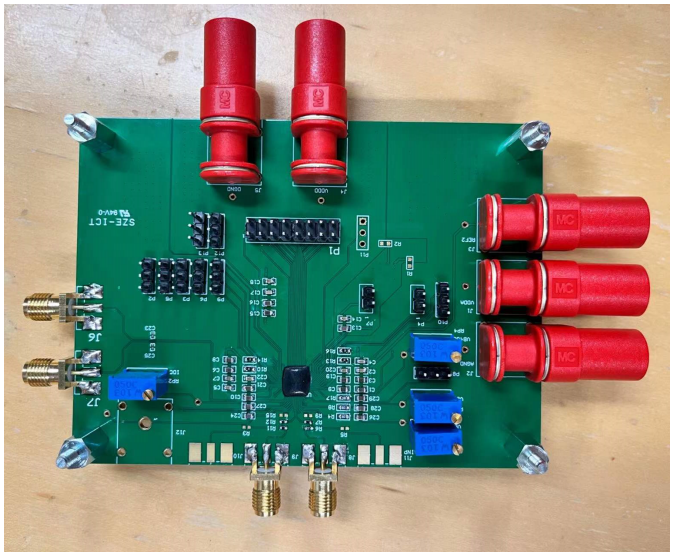


Figure 62. Taped-out SAR ADC test board in 2022.

some extra gain, so that the ADC can still monitor the LGAD signal from the preamplifier when the LGAD gain is reduced towards the end of HL-LHC operation. The amplifier can be bypassed to monitor the preamplifier signal directly. We adapted the latch-based memory provided by CERN ASIC Support Team for the on-chip memory to lower the power consumption. The depth of the on-chip memory has been modified to 1024, corresponding to a 400 ns time window (16 bunch crossings). The test board is shown in Figure 62.

6.2 WS Operation Modes

This chapter is to provide WS operation details for users. There are two operation modes:

1. VGA Mode (S1, S4 open, S2, S3 closed): A high-linearity amplifier has been implemented in the WS front end to provide some extra gain, so that the ADC can still monitor the LGAD signal from the preamplifier when the LGAD gain is reduced towards the end of HL-LHC

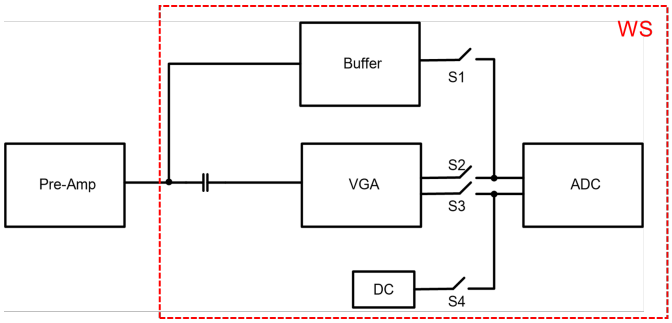


Figure 63. The switches for operations mode control.

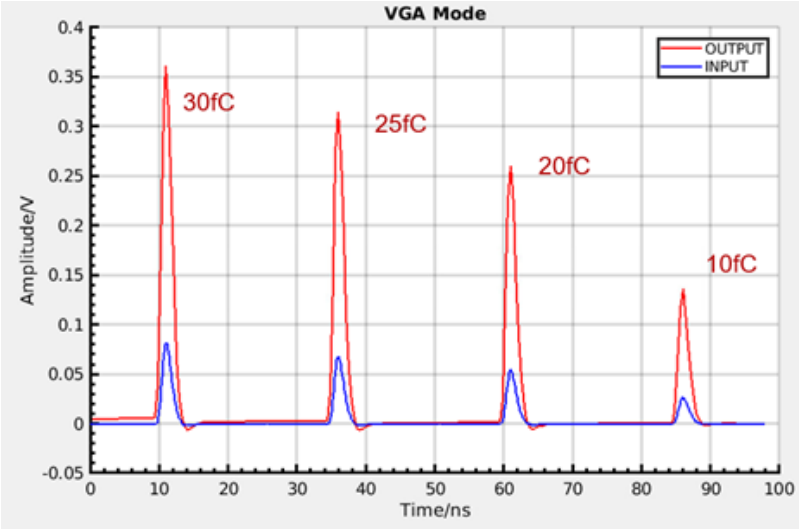


Figure 64. The input and output waveform in the VGA mode in the simulation.

operation.

2. Bypass Mode (S1, S4 closed; S2, S3 open): The amplifier is bypassed so the preamplifier signal can be monitored directly.

We uses 4 switches to control the operation mode as shown in Figure 63.

The input and output waveform in the VGA mode in the simulation is shown in Figure 64.

The input and output waveform in the bypass mode in the simulation is shown in Figure 65.

6.3 WS Power-on/Power-down Modes

The WS also has three power-on/Power-down modes:

- Mode 1: Upon receiving the ws_start from the fast command, the WS will be powered on for the next 400ns and then automatically powered down.
- Mode 2: Upon receiving the ws_start from the fast command, the WS will be powered on till it receives the ws_stop from the fast command. WS will keep overwriting the old data.

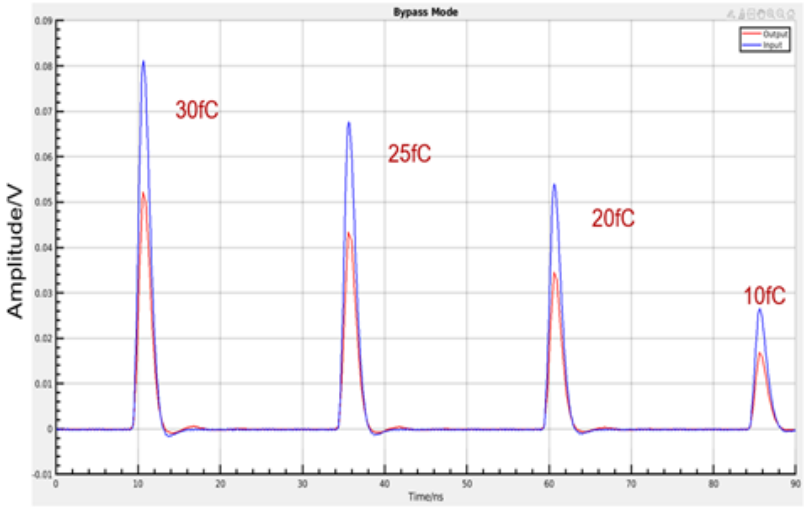


Figure 65. The input and output waveform in the bypass mode in the simulation.

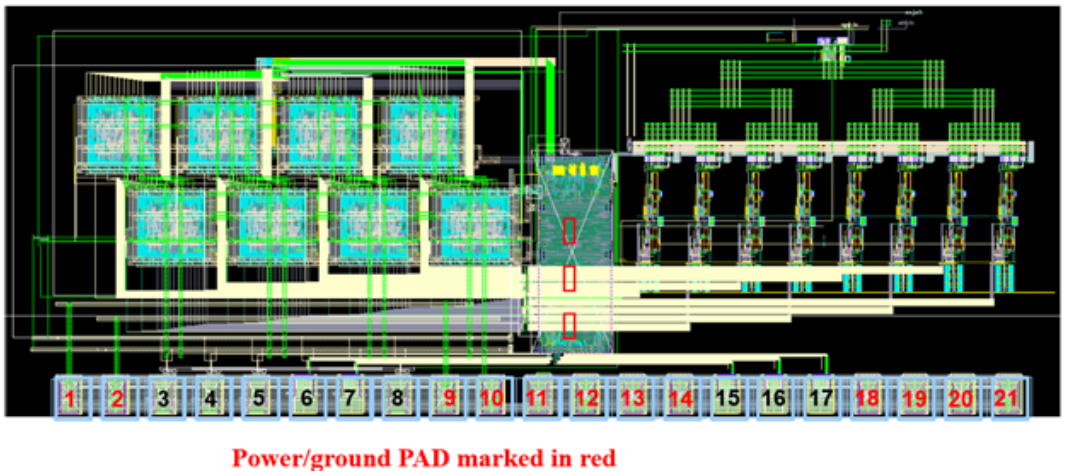


Figure 66. The IO pads for waveform sampler.

Hence only the latest 400ns of data will be stored in WS.

- Mode 3: The state of WS would be configured by the external PAD (wr_en).

6.4 I/O Pad Descriptions

Figure 66 and table 23 show the I/O pad assignment.

6.5 I2C Information

The waveform sampler has a separate I2C slave. The address of the I2C slave is 7'b10xxxxx, where the lower 5-bit address is accessible through5 pads to configure the WS I2C slave address off-chip. The slave provides 32 bytes configuration (write/read) and 16 bytes status (read). The communication follows traditional I2C protocol, as illustrated in figure 67. Note that multi-

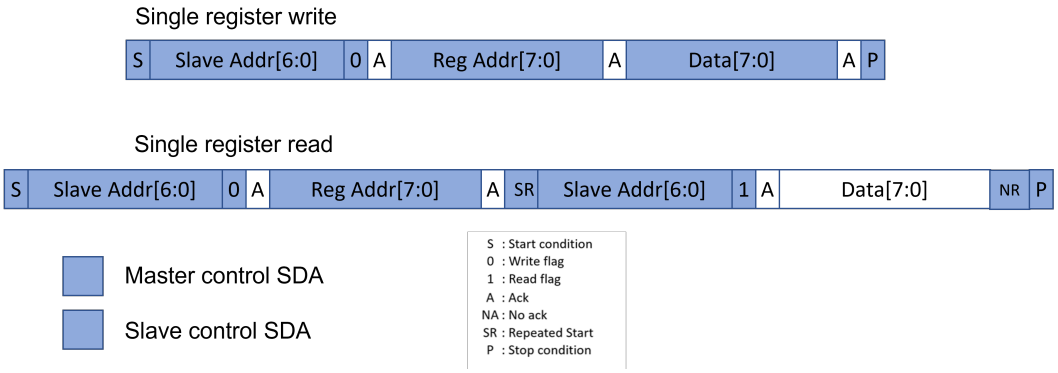


Figure 67. Single register write/read I2C communication protocol of WS.

register operation is also supported. The maximum register for single register write is 32, and the maximum register for single register read is 16.

The configuration registers are summarized in table 24. The hex following **regOut** is the register address. Table 25 summarizes the I2C status and address map of WS. The hex following **regIn** is the register address.

Table 24. WS I2C configuration register definition and address map

Begin of Table 24				
No.	**I2C register (regOut)**	**WS signal**	**Description**	**Typcial Value**
1	regOut1F<7>	sel1	VGA/Bypass mode selction	'1' ⇒ VGA mode
2	regOut1F<6>	sel2	WS power-on/power-down mode selction	'1' ⇒ power on for 400 ns
3	regOut1F<5>	sel3	on-chip/off-chip write enable selection	'0' ⇒ on-chip WR_EN
4	regOut1F<4>	–	–	–
5	regOut1F<3>	clk_gen_rstn	ADC clock generation reset	'1'
6	regOut1F<2>	rd_en_I2C	read enable	'1' ⇒ read mode
7	regOut1F<1>	en_clk	Enable signal for lock delivery from PLL	'1'
8	regOut1F<0>	mem_rstn	Memory reset	'1'
9	regOut1D<7>	rd_addr<9>	Memory read address	N/A
10	regOut1D<6>	rd_addr<8>	Memory read address	N/A
11	regOut1D<5>	rd_addr<7>	Memory read address	N/A
12	regOut1D<4>	rd_addr<6>	Memory read address	N/A
13	regOut1D<3>	rd_addr<5>	Memory read address	N/A
14	regOut1D<2>	rd_addr<4>	Memory read address	N/A
15	regOut1D<1>	rd_addr<3>	Memory read address	N/A
16	regOut1D<0>	rd_addr<2>	Memory read address	N/A
17	regOut1C<7>	rd_addr<1>	Memory read address	N/A
18	regOut1C<6>	rd_addr<0>	Memory read address	N/A
19	regOut0F<7>	DDT<15>	Time skew calibration	'0'
20	regOut0F<6>	DDT<14>	Time skew calibration	'0'
21	regOut0F<5>	DDT<13>	Time skew calibration	'0'
22	regOut0F<4>	DDT<12>	Time skew calibration	'0'

Continuation of Table 24				
No.	**I2C register (regOut)**	**WS signal**	**Description**	**Typcial Value**
23	regOut0F<3>	DDT<11>	Time skew calibration	'0'
24	regOut0F<2>	DDT<10>	Time skew calibration	'0'
25	regOut0F<1>	DDT<9>	Time skew calibration	'0'
26	regOut0F<0>	DDT<8>	Time skew calibration	'0'
27	regOut0E<7>	DDT<7>	Time skew calibration	'0'
28	regOut0E<6>	DDT<6>	Time skew calibration	'0'
29	regOut0E<5>	DDT<5>	Time skew calibration	'0'
30	regOut0E<4>	DDT<4>	Time skew calibration	'0'
31	regOut0E<3>	DDT<3>	Time skew calibration	'0'
32	regOut0E<2>	DDT<2>	Time skew calibration	'0'
33	regOut0E<1>	DDT<1>	Time skew calibration	'0'
34	regOut0E<0>	DDT<0>	Time skew calibration	'0'
35	regOut0D<7>	comp_cali<2>	Comparator calibration	'0'
36	regOut0D<6>	comp_cali<1>	Comparator calibration	'0'
37	regOut0D<5>	comp_cali<0>	Comparator calibration	'0'
38	regOut0D<4>	CTRL<1>	Sampling MEM Effect Reduction	'1'
39	regOut0D<3>	CTRL<0>	Sampling MEM Effect Reduction	'1'
End of Table 24				

Table 25. WS I2C status definition and address map

Begin of Table 25				
No.	**I2C status (regIn)**	**WS signal**	**Description**	**Typcial Value**
1	regIn21<7>	dout<13>	WS digital output	N/A
2	regIn21<6>	dout<12>	WS digital output	N/A
3	regIn21<5>	dout<11>	WS digital output	N/A
4	regIn21<4>	dout<10>	WS digital output	N/A
5	regIn21<3>	dout<9>	WS digital output	N/A
6	regIn21<2>	dout<8>	WS digital output	N/A
7	regIn21<1>	dout<7>	WS digital output	N/A
8	regIn21<0>	dout<6>	WS digital output	N/A
9	regIn20<7>	dout<5>	WS digital output	N/A
10	regIn20<6>	dout<4>	WS digital output	N/A
11	regIn20<5>	dout<3>	WS digital output	N/A
12	regIn20<4>	dout<2>	WS digital output	N/A
13	regIn20<3>	dout<1>	WS digital output	N/A
14	regIn20<2>	dout<0>	WS digital output	N/A
End of Table 25				

The signals are described as follows.

- **sel1**
  - Value = '0': Pre-Amp output directly connected to the ADC

  - Value = ‘1’: Pre-Amp output connected to the ADC with an amplifier in between to provide extra gain.
  - User note: Set ‘sel1’ to ‘0’ as default; change it to ‘1’ when the amplitude of observed signal is too small.

- **sel2**
  - Value = ‘0’: Upon receiving the ‘ws_start’ from the fast command, the WS will be powered on for the next 400 ns and then automatically powered down.
  - Value = ‘1’: Upon receiving the ‘ws_start’ from the fast command, the WS will be powered on till it receives the ‘ws_stop’ from the fast command. WS will keep overwriting the old data. Hence only the latest 400 ns of data will be stored in WS.
  - User note: Set ‘sel2’ to ‘0’ as default, the WS will store 400 ns of data following the ‘ws_start’; change ‘sel2’ to ‘1’ if user would like to observe 400 ns of data right before ‘ws_stop’.

- **sel3**
  - Value=‘0’: Write enable would be given on chip. Once WS is powered on, data is kept being written into the memory.
  - Value=‘1’: Write enable would be given off chip through PAD. Sel2 is not in use now.

- **clk_gen_rstn**
  - User note: Set ‘clk_gen_rstn’ to ‘1’ as default.

- **rd_en_I2C**
  - User note: set to ‘1’ for memory read and WS then is not required to be powered on.

- **en_clk**
  - User note: clock from PLL needs to be enabled for WS to work properly. Set to ‘1’ when WS is in use.

- **mem_rstn**
  - User note: Set to ‘1’ as default. Set to ‘0’ to reset the memory.

- **rd_addr<9:0>**
  - User note: The WS works at 2.56Ghz and has a memory with 1024 depth (Overall, it can store 400ns of data). For readout operation, use the slow command to send out the ‘rd_en’ to WS, and then change rd_addr<9:0> from 10’b0 to 10’b1111111111 to read out data stored in the memory.

- **DDT<15:0>**

	Analog	Digital	Total
WS Enabled	72.6 mW	8.1 mW	80.7 mW
WS Disabled while Memory Read Enabled	3.6 mW	1.7 mW	5.3 mW
Idle	3.6 mW	0.4 mW	4 mW

Table 26. Power consumption of WS.

  - Set ‘DDT<15:0>’ to ‘0’ as default.

- **comp_cali<2:0>**
  - User note: Set ‘comp_cali<2:0>’ to 3’b0 as default (no calibration). If calibration is needed, set ‘comp_cali<1>’ to 1b’1 for 25 ns and set it back to 1b’0. Then set both ‘comp_cali<2> and <0>’ to 1’b1. The WS will perform comparator calibration automatically and the calibration will be finished in 100 us. Set both ‘comp_cali<2> and <0>’ back to 1’b0 when calibration is done.

6.6 Measurement Procedure

The procedure to use the WS is:

1. Set VGA/Bypass mode, power-on/power-down mode and on-chip/off-chip write enable through ‘sel1’- ‘sel3’.
2. Enable the clock from PLL (en_clk=1) and power on WS.
3. Power down WS and read data from memory (rd_en_I2C=1).
4. Disable the clock from PLL. (en_clk=0)
5. Perform data reconstruction.(Please refer to the Matlab code)

The matlab code for reconstruction is provided in list 3 in the appendix as an example.

6.7 Power consumption

The simulated power consumption of the WS is summarized in table 26.

References

[1] C. CMS, "A MIP Timing Detector for the CMS Phase-2 Upgrade," CERN, Geneva, Tech. Rep., 2019. [Online]. Available: https://cds.cern.ch/record/2667167

[2] Q. Sun, S. M. Dogra, C. Edwards, D. Gong, L. Gray, X. Huang, S. Joshi, J. Lee, C. Liu, T. Liu, T. Liu, S. Los, C.-S. Moon, G. Oh, J. Olsen, L. Ristori, H. Sun, X. Wang, J. Wu, J. Ye, Z. Ye, L. Zhang, and W. Zhang, "The analog front-end for the lgad based precision timing application in cms etl," *arXiv*, 2020. [Online]. Available: https://arxiv.org/abs/2012.14526

[3] W. Zhang, H. Sun, C. Edwards, D. Gong, X. Huang, C. Liu, T. Liu, T. Liu, J. Olsen, Q. Sun, X. Sun, J. Wu, J. Ye, and L. Zhang, "A low-power time-to-digital converter for the cms endcap timing layer (etl) upgrade," *IEEE Transactions on Nuclear Science*, vol. 68, no. 8, pp. 1984–1992, 2021.

[4] H. Sun, D. Gong, C. Edwards, G. Huang, X. Huang, C. Liu, T. Liu, T. Liu, J. Olsen, Q. Sun, J. Wu, J. Ye, L. Zhang, and W. Zhang, "In-pixel automatic threshold calibration for the cms endcap timing layer readout chip," *Journal of Instrumentation*, vol. 16, no. 09, p. T09006, sep 2021. [Online]. Available: https://dx.doi.org/10.1088/1748-0221/16/09/T09006

[5] H. Sun, D. Gong, W. Zhang, C. Edwards, G. Huang, X. Huang, C. Liu, T. Liu, T. Liu, J. Olsen, Q. Sun, J. Wu, J. Ye, and L. Zhang, "Characterization of the cms endcap timing layer readout chip prototype with charge injection," *Journal of Instrumentation*, vol. 16, no. 06, p. P06038, jun 2021. [Online]. Available: https://dx.doi.org/10.1088/1748-0221/16/06/P06038

[6] "lpgbtv1 manual," https://lpgbt.web.cern.ch/lpgbt/v1/index.html.

[7] H. Sun, Q. Sun, S. Biereigel, R. Francisco, D. Gong, G. Huang, X. Huang, S. Kulis, P. Leroux, C. Liu, T. Liu, T. Liu, P. Moreira, J. Prinzie, J. Wu, J. Ye, L. Zhang, and W. Zhang, "A radiation tolerant clock generator for the cms endcap timing layer readout chip," *Journal of Instrumentation*, vol. 17, no. 03, p. C03038, mar 2022. [Online]. Available: https://dx.doi.org/10.1088/1748-0221/17/03/C03038

[8] L. Zhang, C. Edwards, D. Gong, X. Huang, J. Lee, C. Liu, T. Liu, T. Liu, J. Olsen, Q. Sun, J. Wu, J. Ye, and W. Zhang, "An fpga-based readout chip emulator for the cms etl detector upgrade," *Journal of Instrumentation*, vol. 18, 2023.

[9] L. Fang, X. Wen, T. Fu, G. Wang, S. Miryala, T. T. Liu, and P. Gui, "A 2.56-gs/s 12-bit 8x-interleaved adc with 156.6-db foms in 65-nm cmos," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 30, no. 2, pp. 123–133, 2022.

[10] X. Wen, T. Fu, L. Fang, G. Wang, S. Miryala, T. T. Liu, and P. Gui, "Waveform sampler: A 2.56gs/s 12-bit time-interleaved pipelined-sar analog-to-digital converter (adc) with on-chip memory," in *TWEPP 2022 Topical Workshop on Electronics for Particle Physics*, no. 2, Sep. 2022, pp. 161–162.

.1 Appendix A. An example of data reconstruction of the WS

Listing 3: A Matlab example for waveform sampler data reconstruction.

```
clear
 data = csvread('data.csv',1,1);%1024x14
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

pointer_Column(:,1)=data(:,1);
pointer=find(pointer_Column==1);
data(:,1:13)=data(:,2:14);

zeros=Dout_S1(1024,1);
zeros=Dout_S2(1024,1);
zeros=Dout(1024,1);
zeros=Reformed_Dout(1024,1);

coeff=0.04/5*8.5;

    for i=1:1:1024

       Dout_S1(i)= data(i,1)*32+data(i,2)*16+data(i,3)*8+data(i,4)*4+data(i,5)*2+
           ↪ data(i,6);
       Dout_S2(i)= data(i,7)*24+data(i,8)*16+data(i,9)*10+data(i,10)*6+data(i,11)
           ↪ *4+data(i,12)*2+data(i,13);

       Dout(i)=Dout_S1(i)*coeff+Dout_S2(i);

    end

Reformed_Dout(1:1024-pointer)=Dout(pointer+1:1024);
Reformed_Dout(1024-pointer+1:1024)=Dout(1:pointer);
plot(Reformed_Dout)
```

.2 Appendix B. Known Issues

1. TMR is fully applied at RTL level, but not fully implemented and verified.
2. The optional fast command self-alignment is implemented and verified at RTL level, but not fully implemented on the back-end due to tight schedule.
3. The 'late' signals from the analog phase shifter keep changing when the loop is locked. The signals are connected to I2C through long traces, which it consumes power more than necessary. As the phase shifter is silicon proven, there is no need to monitor the signals anymore.

.3 Appendix C. FAQ

1. Q: VREF: does this need to be connected to the RB? If so, as an input or output? I understand that it can function as both, possibly for diagnostic purposes, but we need to know it needs to be connected on the RB (if at all).
   A: VREF: this is a reference voltage used by DAC (directly related to the discriminator threshold) and charge injection. It is generated inside the chip, with the option that can be provided/applied externally (this option is for testing purpose only). By default, it is used for monitoring purpose. On the ETROC2 chip test board, VREF is output for diagnostic purposes or input in case the internal VREF generator doesn't work (for testing purpose only). In the system (module and readout board design), it is recommended to monitor VREF with an ADC as it is directly related to discriminator threshold.

2. Q: VP: Do you have any recommendation for the capacitance (or possible range of capacitances)?
   A: This is meant for stabilizing the PLL VCO (voltage controlled oscillator) power supply. A 0.1 uF capacitor is recommended for VP. This is optional. The design team recommend to have this option for the first round of prototype design and will test to see if we really need it or not (floating vs 0.1uF).

3. Q: CLK1280P/N: can you confirm whether this can be left floating?
   A: This is 1.28 GHz optional clock input for testing and performance study purpose only. Not required for module design, and CLK1280P/N can be left floating if not being used.

4. Q: VDD_EFUSE: does this need to be connected on module or can it be left floating?
   A: This is the 2.5V supply for EFUSE programing. Should leave it floating in the final system as we plan to program EFUSE during wafer probing testing stage. For the first module/readout board design, if feasible, consider using it only if one wants to program the EFUSE to keep track of the chips (note: the initial diced ETROC2 chips will not have the EFUSE programed).

5. Q: VTemp: Can you please specify some details about this signal? e.g. What is the voltage range?
   A: This is the output of the build-in Temp Sensor on ETROC2. Figure 68 shows VTemp vs temperature in simulations and measurements from ETROC I2C test chips. The measured curves from test chips are offset down from 50mV to 150 mV, but the sensitivity (delta VTemp/ delta temperature) is close to what predicted by the simulations. The chip-to-chip variation of the sensitivity is small enough for the purpose of monitoring temperature changes. The expected voltage range is approximately from 300 mV to 700 mV(over temperature from -40 to 50C, including chip process variation). The VTemp should be monitored by an ADC in SCA or lpGBT (10-bits ADC good enough).

6. Q: As you know, AVDD and DVDD will be from a common supply. Can we simply treat them as one net or is any kind of filtering required for any of the analog supplies?
   A: If feasible, it is recommended to isolate AVDD and DVDD with ferrite bead.

7. Q: Can waveform sampler AVDD / DVDD be treated as a single net along with the ETROC voltage or must the AVDD be filtered in some way? (same question as above for the ETROC)
   A: Based on most recent testing on Rad-hard waveform sampler chips, the AVDD and DVDD are treated as a single net on the testing board (using ideal power supply at lab).

8. Q: Do you have any simulations/recommendations for power supply decoupling?
   A: We don't have detailed simulation. We would recommend some 0.1uF ceramic capacitors placed very close to the ETROC. Besides that, we expect some capacitors(10 uF) for DC/DC power supply filtering on the readout board close to the connector to the module.

9. Q: Do the ADDR inputs have built-in pull-ups or pull-downs?
   A: Yes, there are 40 kOHM internal pull-down resistance on these input.

10. Q: Does the RESET input have built-in pull-up or pull-down?
    A: Yes, there is 40 KHM pull-up resistor for Reset input.

11. Q: Since this presentation(https://indico.cern.ch/event/1139028/contributions/4779384/attachments/2
    %20Read-Only.pdf), do you have any update on a preferred ground scheme? (AGND/DGND vs. ASIC GND / HV GND)?
    A: yes, see figure 69 and figure 70.

12. Q: Can the waveform sampler be on the same I2C bus as the ETROC? What is the base I2C address of the waveform sampler and the ETROC?
    A: There are 5-bits I2C address for WS and ETROC, and each can be set separately by user on test board or module. There are 2-bit internal I2C address. For main ETROC2, the 2-bits will be set as 11 internally (two MSB). For WS, the 2-bits will be set as 01 internally (two MSB).

13. Q: Do the waveform sampler I2C pins have integrated pull-ups / pull-downs?
    A: For ETROC2 waveform sampler, the I2C pins do not have integrated pull-ups/downs. For ETROC3, the plan is to have the I2C for WS integrated with the main ETROC.

14. Q: What is the WR_EN pin of the waveform sampler and what (if anything) should it be connected to?
    A: wr_en: External write enable signal for the memory (Can be used to test WS without involving fast commands). This is for testing purpose only. No need to connect to it for module design.

15. Q: Does the RSTN of the waveform sampler have integrated pull-up / pull-down? Can it be bonded to the same reset as the ETROC or should they be connected to separate lpgbt / sca GPIO?
    A: The I2C RSTN pad has integrated pull-up (the earlier slide labeled it as RST, should be RSTN). For ETROC2, it is recommended to keep the reset separate for the main ETROC2 and the waveform sampler.

16. Q: does the 40MHz clock input have internal termination or does it need an external 100 ohm resistor?
    A: Each of these serial inputs(40 MHz clock/Fast command/1.28 GHz clock) has optional internal termination which can be enabled or disabled with I2C. But the internal termination resistors have tolerance of about +-20%. Users can choose to use the internal termination or put an external termination on the board with better tolerance. Suggestion: if feasible, for the first prototype, it is better to include an external termination for testing purpose (to compare with using internal termination).

17. Q: And same question for the Fast command port.. I know it is multi-drop on our module but I'm not sure if that needs to be done with external resistors or by I2C configuration.
    A: For fast command, the multi-drop is two stage, only the second one needs termination. One could simply use I2C to enable the second one (and disable the first one). But if you have room, you can also put an external termination on the second one for testing purpose. If turns out the external termination and internal termination all work the same, then later you can remove the external termination for the next version.

18. Q: Does the downlink on the ETROC support polarity inversion, or must the P and N sides of the diff pair be assigned correctly?
    A: Yes, ETROC supports it.

19. Q: Do you have any approximate breakdown of the power consumed by the analog and digital domains separately? I have seen total power estimates but wondering how that breaks down, e.g. 0.25W analog 0.75W digital or whatever the real number is.
    A: we know most the measured power consumption for almost all (if not all) of the analog blocks from test chips. For the digital, we had some estimate, but chance is good that the actual could be lower than the estimate due to extensive optimization. But we wouldn't know for sure until more detailed power analysis is done after full chip integration is finalized. . . . I presented the details of the breakdowns (an update of power consumption of ETROC) a while back, need to find that talk at ETL Front-end meeting (sometime last year). Also need to include the waveform sampler (only enabled for a short period once in a while). See figure 71.

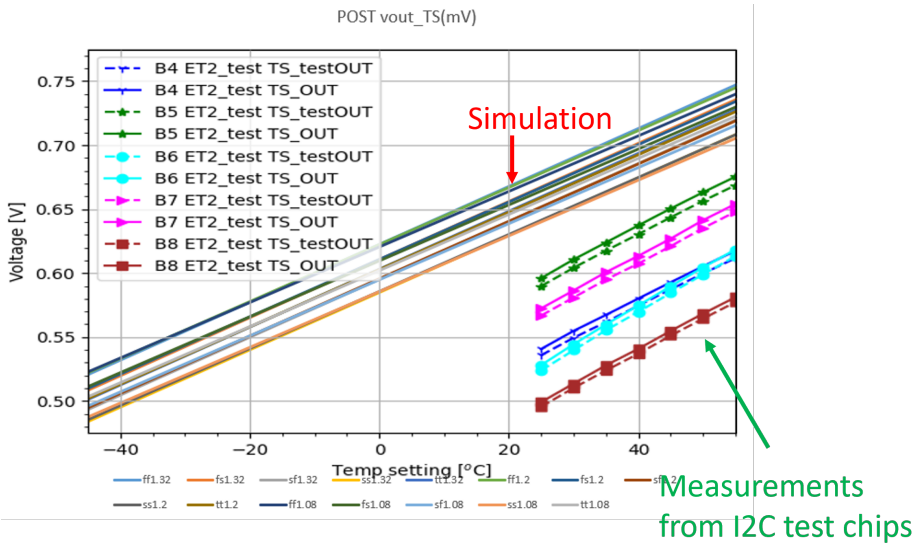


Figure 68. Simulation and measurement results of temperature sensor.

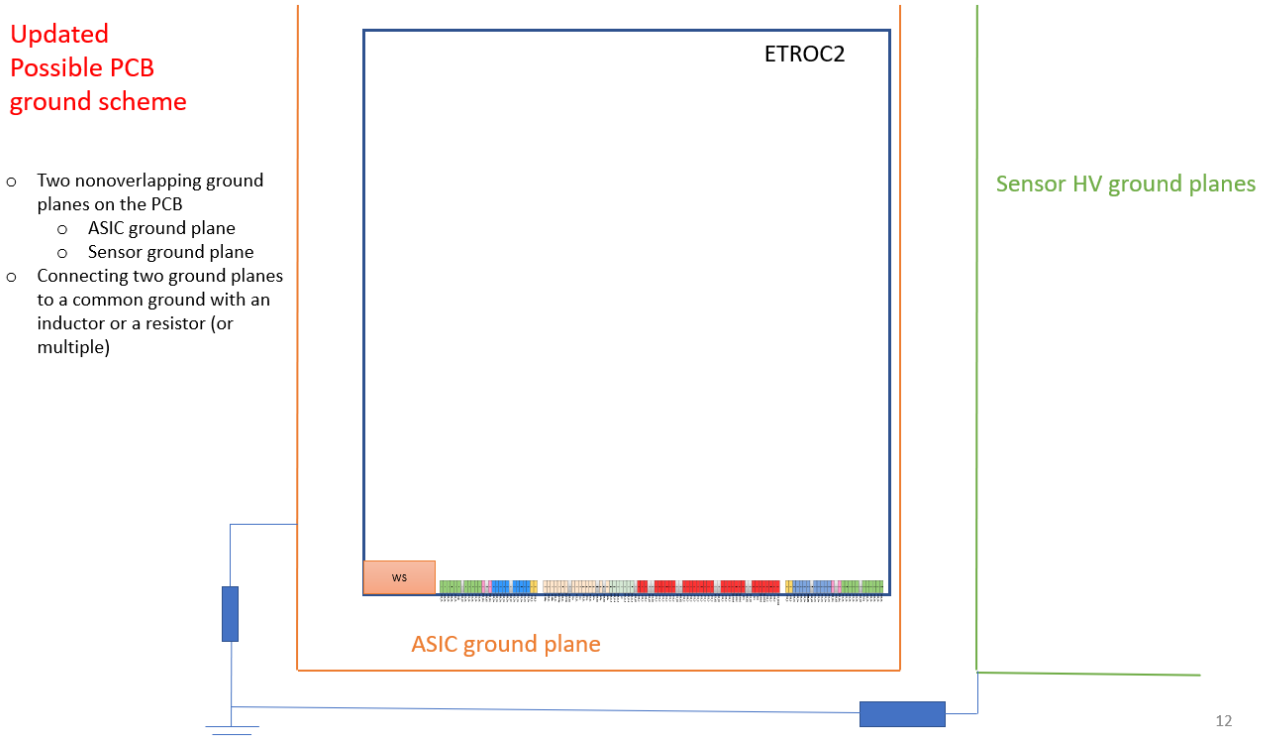


Figure 69. PCB grounding scheme 1.

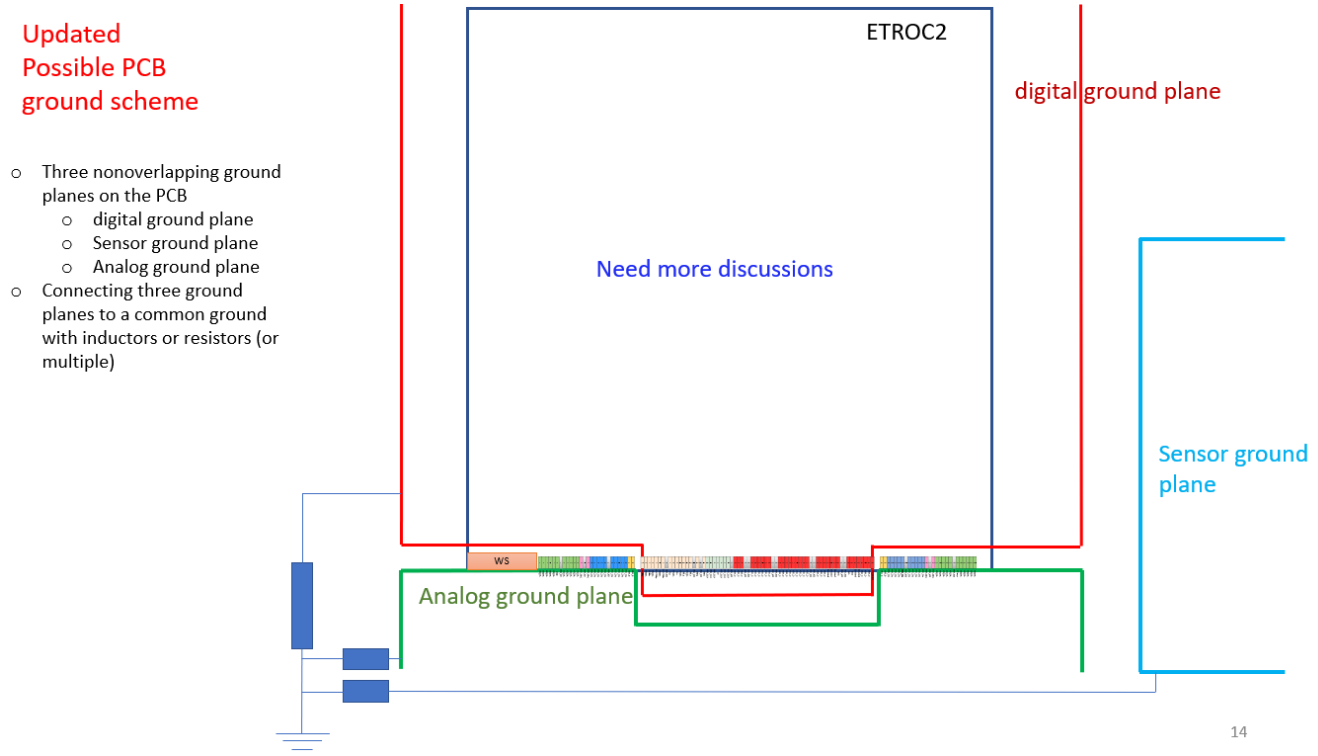


Figure 70. PCB grounding scheme 2.

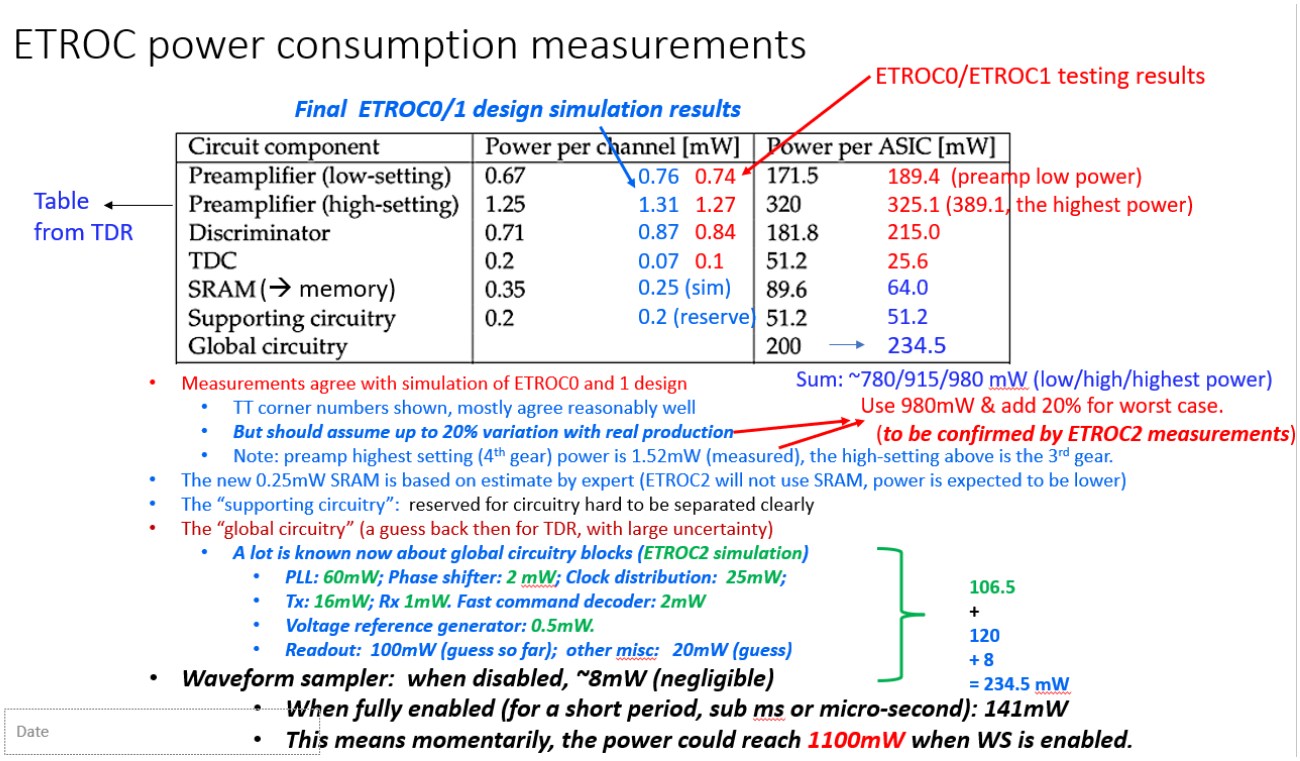


Figure 71. Power conumption breadown.