The Elefant Chip

Electronics for Amplitude and Timing Measurment of Drift Chamber Readouts

Alan Chau, Derek DeBusschere, Scott Dow, Jeremy Flasck, Carl Grace,

Richard Jared, Michael Levi, Fred Kirsten, Oren Milgrome,

Dinis Santos and Edwin Su

Ernest Orlando Lawrence Berkeley National Laboratory

DRAFT - July 23, 1996

1.0 Introduction

This report describes the design of an integrated-circuit chip - the "Elefant chip" for processing the signals developed by Amplfier IC chips that are directly connected to wires of drift chambers. "Elefant" is an acronym for ELEctronics For Amplitude 'N' Timing. The Elefant chip is designed to be replicated in large quantities at a low cost per channel, and is therefore suitable for general application in instrumenting large drift chambers in large-scale physics detectors.

This report describes the functional design, implementation and operation of the Elefant chip. The Amplifier IC chips that directly connect to the chamber wires are described in another report[1]. Other parts of the system are described only in sufficient detail to provide the operating context of the Elefant chip.

The chip is implemented as an ASIC (Application Specific Integrated Circuit). Each chip processes the signals from two Amplifier ICs that service eight wire-chamber channels. In response to triggers, the chip saves packets of event data. On command of a readout controller, the saved packets are output to downstream parts of the system.

The Elefant digitizer IC is designed to measure both the charge and the single electron time of arrival for eight channels of the BaBar drift chamber. By combining an FADC and an TDC (Time-to-Digital Converter) for each channel, each converter can be optimized, and simplified, to achieve the physics goals required of the drift chamber while reducing the per channel costs of the electronics. The Elefant IC also implements portions of the BaBar Fast Control and Trigger protocols to simplify the connection of the chip to the rest of the BaBar detector.

1.1 Nomenclature

The timing reference for the chips is a System Clock whose frequency is 59.5 MHz. A Sample Clock of 14.875 MHz is derived from the System Clock within each chip. For convenience, the System

[1] "Specification of the Amplifier IC for the BABAR Drift Chamber," D. Coupal, et al, University of California at Santa Cruz and Stanford Linear Accelerator Center. June, 1996.

and Sample clock frequencies are indicated as 60 and 15 MHz, respectively, in this document. Also, the terms "Elefant chip" and "chip" are used interchangably.

Names in bold-face type, such as **dout**, refer to signal names as shown on the included block diagrams. Names in italics, such as *control_logic*, refer to blocks in the block diagrams. The circuit blocks names correspond to the names in the Elefant IC schematics. The Elefant IC schematics should be considered supplementary documentation.

A signal name like **out(7:0)** carries the meaning that the signal is 8 bits wide, and the bits are numbered 7 through 0. Bit 0 is the least significant bit; bit 7 the highest.

1.2 Interface to Amplifier ICs

Sets of several Elefant chips will be mounted on printed-circuit (PC) boards along with a local board controller in the form of a Field-Programmable Gate Array (FPGA). The Amplifier IC will be connected to the Elefant chips by short low signal level differential wire lines. The FPGA controls the initialization of the Elefant chips, organizes the readout of data, and provides an interface to the serialized control signals of the output fiber optic link, the C-link. The topology for the placement of the Analog chip, the Elefant chip and the readout controller FPGA has not been finalized.

As shown in Fig. 1, the preamplifier within the Amplifier IC is directly connected to a wire of the drift chamber and amplifies the wire signals. The peaking time of the shaping amplifier will be in the range 50 to 100 nsec. The discriminator is used to measure the times-of-arrival of the leading edges of the output pulses from the preamplifier.

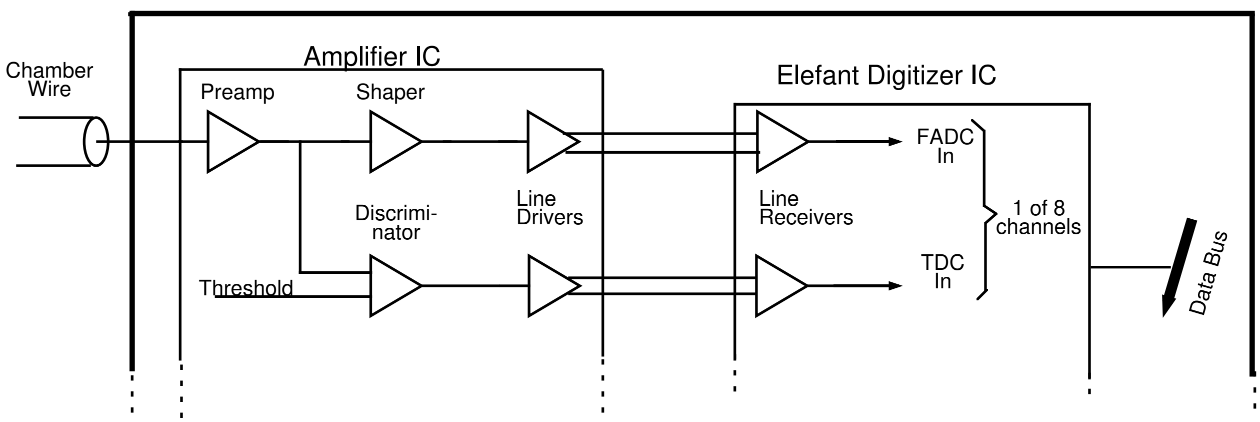

Fig. 1. Simplified block diagram of the front-end electronics for a single wire of a drift chamber.

The time relationship of the outputs of the preamplifier, shaper and discriminator is shown in Fig. 2. The timing jitter of the discriminator in generating the leading edge of its output pulse should be accurate to 200 ps. The output pulse of the discriminator is shown as time-over-threshold, although only the leading edge is significant in indicating time-of-arrival. There is a 40 ns hysterisis on the falling edge.

The outputs of both the shaping amplifier and the discriminator are individually transmitted via balanced transmission lines to line receivers on the Elefant IC. The discriminator signals are destined to the TDCs in the chip; the shaping amplifier signals go to the FADCs.

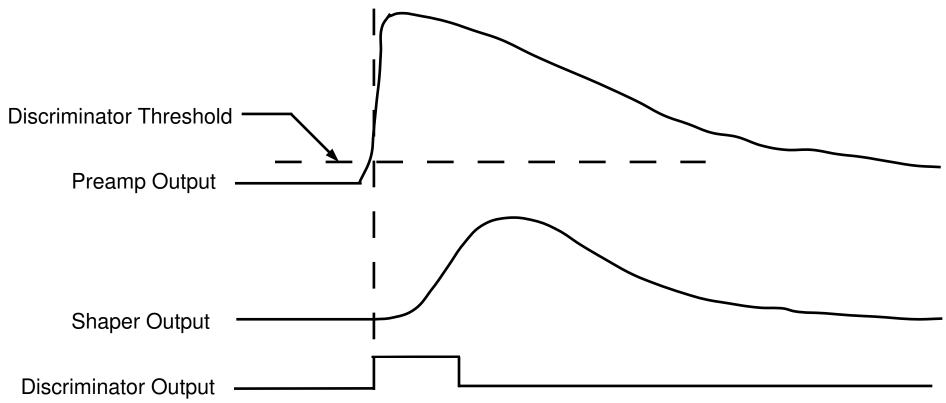

Fig. 2. Time relationship beween the preamp output and the output of the discriminator. The output of the shaper is fed to the FADC. The output of the discriminator goes to the TDC.

1.3 Overview of the Elefant Chip

Figure 3 shows a simplified block diagram of a Elefant chip, suitable for a description of its basic functions. This figure is mainly a pictoral view of the Elefant IC and is not strictly accurate. Each channel has its own time-to-digital converter (TDC), flash analog-digital converter (FADC) and multiplexer (MUX). The MUX merges the streams of data from the TDC and FADC. A single channel's worth of TDC, FADC and MUX functions are shown; all eight channels are then physically combined into one datastream in the DRAM (Dynamic Random-Access Memory) and following functions.

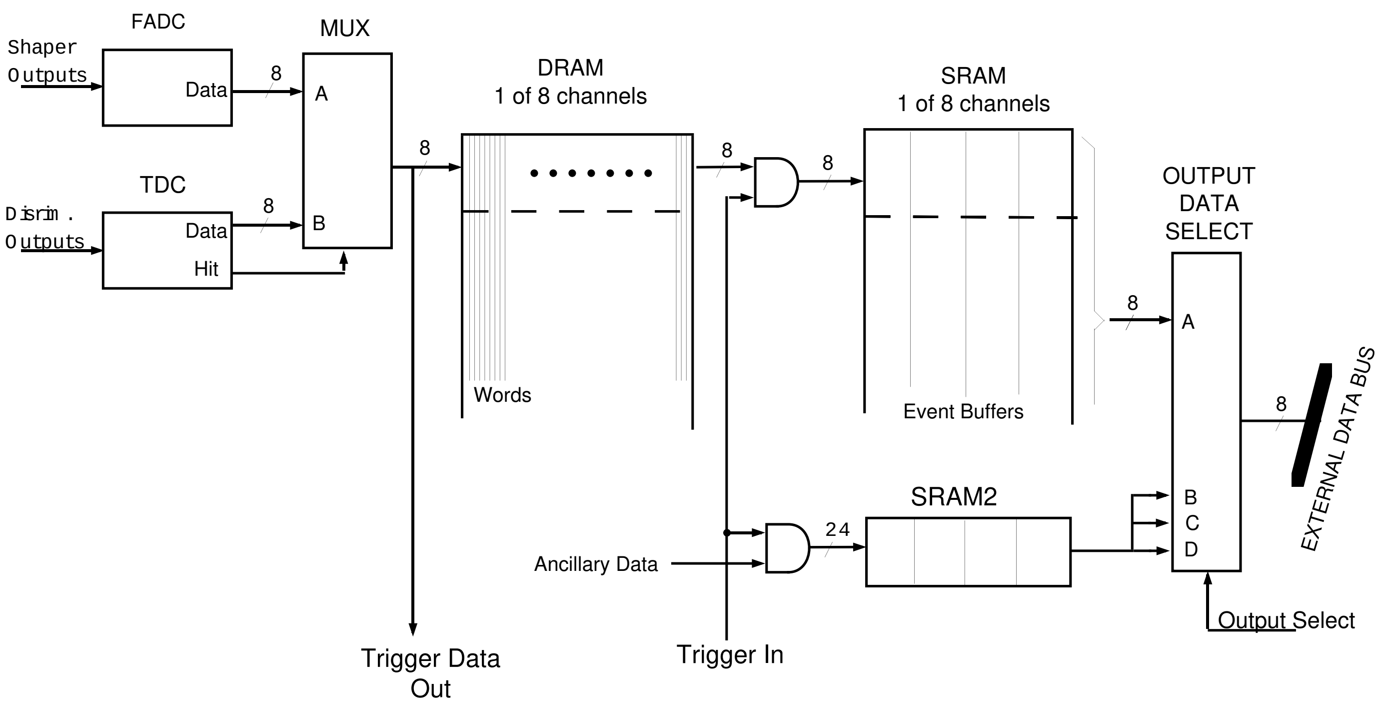

Fig. 3. A very simplified block diagram of the Elefant digitizer IC.

The basic tasks of the chip are:

1) To continuously digitize, at a 15 MHz rate and 6- to 8-bit dynamic range, the waveforms of the analog signals from eight channels of the Amplfier ICs;

2) To record the time-of-arrival, using 1.0 ns bins, of each significant pulse in the digital signals from eight channels;

3) To continuously output byte-wide streams of trigger data at Sample Clock rate, conveying the history of "hits" in each channel;

4) To save all information in 1) and 2) for a length of time equal to the Level 1 trigger latency time, which is assumed to be 11.5 ± 0.5 μsec;

5) For each Level 1 trigger, to create packets of 32 bytes for each channel, each packet containing the history of 1) and 2). These histories cover a period of 2.15 μsec - a period that encompasses the time of the event associated with the trigger. Simulataneously, ancillary data is stored in a second memory;

6) On command from the FPGA, output the data packets and ancillary data. The chip supports either complete or sparse readout of event data.

Each of the above tasks is accomlished within different sub-circuits of the Elefant IC. The analog signals are digitized with an 6-bit magnitude, 8-bit dynamic range, FADC that can be operated in a piecewise linear mode to achieve the complete 8-bit dynamic range. The digital signals are binned with a 1.04 ns resolution using a TDC based on a 64-tap delay-locked loop. A trigger module derives trigger primitives from either the FADC or the TDC and outputs an eight bit word which indicates hits in each of the eight channels. The output of each channel is physically combined into one 192 word dual-port DRAM latency buffer. Upon receipt of a L1_accept a data packet is transferred into one of four SRAM output buffers. In addition to the data packet, ancillary data related to the event is stored in the corresponding buffer in another memory; SRAM2.

Within the chip, all of these operations may be proceeding simultaneously without interfering with each other.

2.0 Design and Operational Details

In this section, attention is primarily on the design and operational details of the various functions of the Elefant chip. The reader should refer to the schematic diagram shown in Fig. 4.

2.1 Flash ADCs

Each FADC measures the amplitude of its analog input signal at the Sample Clock rate and, for each measurement, generates a 6-bit binary number representing the instantaneous amplitude of the input signal. The chip contains eight FADCs, one for each of the eight signal channels. The FADCs are contained in the *fadcX8* block in Fig. 4. The *fadcX8* block also contains eight *delta_gen* blocks, one for each FADC, which perform the δ tests. The δ tests are used to measure the rate of change of the rising edge of the analog inputs.

The signals from the shaping amplifiers are transmitted to the chip over balanced two-wire transmission lines. They are received on the chip by pins **ap[7:0]** and **am[7:0].** The *analog input_conv* block in Fig. 4 contains line receivers that amplify and convert the signals to single-ended analog signals which are then applied to the FADC inputs, **fadc_in[7:0].**

Insert Fig. 4. - Simplified schematic.

Fig. 4. Simplified Elefant digitizer IC schematic.

The timing of the FADC data conversion is controlled by **phi1** and **phi2**, two phases of the Sample Clock that are derived from the TDC vernier (Section 2.2.3).

Within the *fadcx8* block, two bits, the seventh and eighth bits, are appended to form byte output words. The seventh bit is always 0; the eight bit reflects the result of the δ calculation.

2.1.1 The δ Calculation

The eighth bit is appended by the *delta_gen* block attached to each *FADC* block. This bit carries the result of the test to determine if the current rate of change of the analog input is greater than a threshold value. The output of this test is the one-bit variable, δ, where:

$$\delta_i = 0 \text{ if } (\mathbf{dout[5{:}0]}_i - \mathbf{dout[5{:}0]}_{i-2}) > \mathbf{thres[1{:}0];} \text{ otherwise, } \delta_i = 1.$$

Here, $\mathbf{dout[5{:}0]}_i$ is the value of the six-bit number output by the FADC at the *i*th clock period. The externally supplied number, **thresh[1:0],** can be preset by external controller to 0, 1, 2 or 3. The threshold is the same for all eight channels. The δ bit is normally one to insure all data words in the datastream contain at least one non-zero bit. This assists the FPGA controller in keeping streams of zero containing datawords to less than four.

Complete bytes are output from the *fadcX8* block as **fadc_outX[7:0]**. The format is:

Bit 0 ... 5	6	7
6-bit amplitude data	0	δ

2.1.2 Design Details of the FADC

A simplified block diagram of one FADC is shown in Fig. 5. The design is based on a gated-comparator technique using a published comparator design[2]. Since this is a 6-bit measurement, there are 63 comparators, each of which has two signal inputs, **vref** and **FADC In**, and two clock inputs, **phi1** and **phi2**. One signal input is connected to the input signal, **FADC In**; the other to a fixed voltage that is the threshold value for that comparator. The threshold values are derived from the divider network, which is essentially a chain of 64 resistors of equal value.

On the falling edge of **phi2** the gated comparator takes the initial sample and starts the regeneration process. When **phi1** is high a second regeneration phase starts that eventually stores the comparison result in the output latch of the comparator. If the signal input voltage was more positive than the threshold value, the comparator will output a high level; if the input was less positive, the comparator will output a low level. Finally, on the next falling edge of **phi2**, the encoded output is stored in the FADC output latches. Note that the falling edge of **phi2** is aligned with the rising edge of sample clock. Therefore, the FADC samples the input analog signal on the rising edge of Sample Clock and outputs that sample on the next rising edge of Sample Clock.

As shown in the figure, one end of the divider network is connected to a voltage near ground, **bot_ref**; the other end to a reference voltage, **top_ref**; and the midpoint of the network to a voltage **mid_ref**. As an (atypical) example, if **bot_ref** = 0 volts, and if the **mid_ref** voltage is left floating**,** the transfer characteristic of the converter will be linear. The threshold value for comparator 1 will then be

---

[2] G.M. Yin, F. Op't Eynde, W. Sansen, "A High-Speed CMOS Comparator with 8-b Resolution," IEEE J. Solid State Circuits, v27, n2, Feb. 1992.

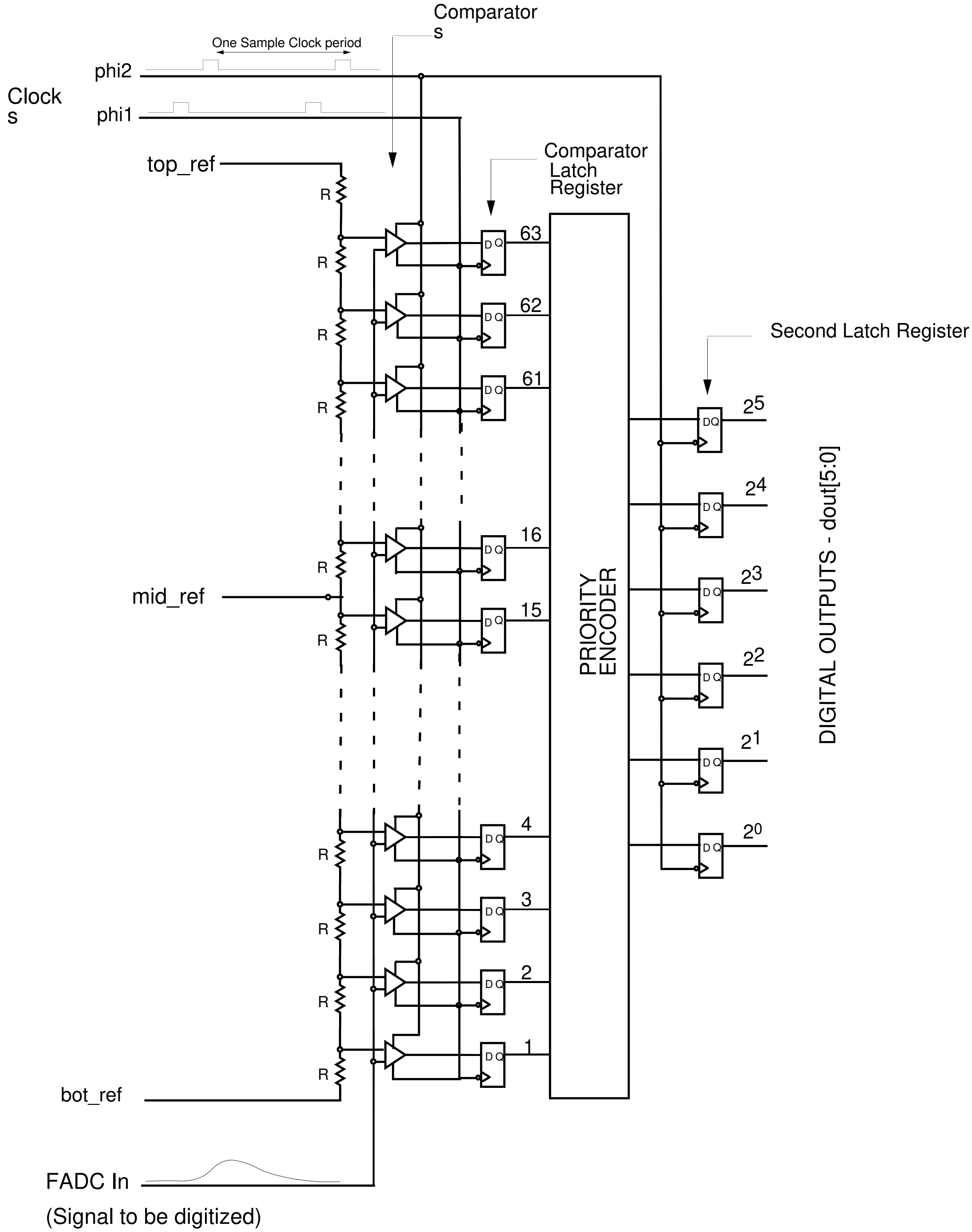

Fig. 5. Simplified block diagram of one FADC.

top_ref/64; that for comparator 2 will be (2 x **top_ref**)/64, etc. Thus, if the signal voltage is greater than (n x **top_ref**)/64, but less than (n+1 x **top_ref**)/64, then all comparators numbered 1 through n will generate a high output and all comparators numbered n+1 through 64 will generate a low output. The binary number output by the FADC will have the value n. Typical operating voltages for **bot_ref** and **top_ref** are 1.0 V and 4.0 V, respectively.

2.1.3 Dynamic Range

The span of the 6-bit FADC is from **bot_ref** to **top_ref;** an input voltage equal to **bot_ref** results in a zero value output; an input equal to **top_ref** results in a full-scale output of 63 . Non-zero values of **bot_ref** (typically, 1 Volt) are used to compensate for a pedestal in the analog input. The range of usable values of the three reference voltages is given in Table V.

The equivalent of a 7- or 8-bit dynamic range can be achieved by applying a dc voltage to **top_ref**, and another dc voltage to **mid_ref**, where, usually, **mid_ref** < 1/2 (**top_ref + bot_ref**) + **bot_ref**. This results in double-sloped transfer characteristic, where the break point is at the output value of 31. The span of the output values 0 to 31 is equal to (**mid_ref - bot_ref**); the span of the output values 32 to 63 is (**top_ref - mid_ref**). When the span of voltage for output values 0 to 31 is 1/4 the span of voltage for output values 32 to 63 the FADC will be operating with 8-bit accuracy.

2.1.4 Implementation Details

The top level schematic of one *FADC* is shown in Fig. 6. The reference voltage range is established by **top_vref_IN**, **mid_vref_IN**, and **bot_vref_IN**. These DC levels are input to the *fadc_res* block which contains the resistive ladder.

Sixty-three comparators are organized in two groups on either side of *fadc_res*; 32 reside in *fadc_cell0*, and 31 reside in *fadc_cell1*. Likewise, the output encoding stage is divided into two parts, *fadc_encoder1* and *fadc_encoder2*. The latter generates the LSB of the output, while the former generates the upper five output bits. The analog input voltage **v1_IN** is input to the comparators in *fadc_cell0* and *fadc_cell1*.

The *fadc_encoder1* output is buffered by *fadc_buf* before being latched by *fadc_latch2*. The *fadc_encoder2* output is buffered by *fadc_buf2* before being latched by *fadc_latch3*. The complete *FADC* output is **dout[5:0].**

Two FADC clocks, **phi1_IN** and **phi2_IN**, are buffered by *fadc_clkbuf* before reaching the comparators and output latches. This reduces the load on the global FADC clock routes. Timing diagrams for **fadc_phi1** and **fadc_phi2** are shown in Fig. 15. Externally to the FADC block **phi1_IN** and **phi2_IN** are named **fadc_phi1** and **fadc_phi2**, respectively.

2.2 TDCs

The following discussion is derived, in part, from three informal reports.[3,4,5] Reference can also be made to a published report.[6]

Each chip includes eight TDCs, one for each signal channel, shown as the *tdcX8* block in Figure 4. Each TDC receives the output of a discriminator in the analog section. The purpose of the TDC is to

---

3 "Test Results from the TDC Chip," Edwin K. Su, Lawrence Berkeley Laboratory. Informal, undated report.

4 "A 50 MHz CMOS delay-locked loop and timing generator," D. Santos, S. Dow, M. Levi, Lawrence Berkeley Laboratory. Informal, undated report.

5. "A 312.5 ps resolution time-to-digital converter chip," D. Santos, S. Dow, M. Levi, Lawrence Berkeley Laboratory. Informal, undated report.

6. Arai, Y., Matsumura, T., Endo, L.: "A CMOS Four-Channel x 1K time Memory LSI with 1-ns/b Resolution," *IEEE J. Solid-State Circuits,* 27, No. 3, pp359-364, March 1992.

FADC schematic

Fig. 6. Top level FADC schematic.

measure, using a vernier technique, the time of arrival of the leading edge of the discriminator pulse (the low-to-high transition) to a precision of approximately 1 nsec. The arrival of a leading edge denotes a wire chamber "hit."

The signals from the discriminators are transmitted to the chip over balanced two-wire transmission lines. They are received on the chip by pins **dp[7:0]** and **dm[7:0]**. The *digital input_conv* block on Figure 4 contains line receivers that convert the signals to single-ended digital signals, which are then applied to the TDC inputs via the 8-bit bus, **DataStrobe.** Six bit digital data, together with two tag bits are output from each TDC on **tdc_outX[7:0].**

Each TDC holds the 6-bit binary number at its output. This number is held until the next **DataStrobe**. Normally, these numbers have no significance and are not allowed into the data stream. However, whenever the leading edge of a discriminator pulse is received, the value of significance is changed and multiplexed into the data stream two cycles after the leading edge was received. Its value denotes the time-of-arrival of the leading edge measured with respect to the start (low-to-high transition) of the current Sample Clock cycle. The least count of the output number is 1/64 of a Sample Clock cycle (1.05 nsec). The time of arrival of the discriminator pulse is thereby defined by two quantities. First, it is defined in units of 67 nsec by the particular Sample Clock cycle during which the output occurs; second, it is defined in units of 1.05 nsec by the value of the 6-bit number. The time-of-arrival is thus:

$$(N - 2) \times 67 \text{ nsec} + V \times 1.047 \text{ nsec};$$

where: N is the number of the Sample Clock cycle when the 6-bit hit value is output; and V is the hit value contained in the six-bit binary number output by the TDC. Note that the range of V is 0 to 63.

Each TDC also outputs a one-bit "Hit" signal. This signal is normally "0". It is set to "1" during the Sample Clock cycle that the time measurement is output. The "Hit" bit is used to steer the MUX for that channel (see section 2.3). The Hit signals for all channels are output from the *tdcX8* block as **tdc_hit[7:0]**.

Two bits are added to the output data from each TDC to form eight byte words. The seventh bit is always "1"; the eighth bit is always "0." This pattern indicates to the downstream processor that the data word originated in the TDC. The complete bytes are output as **tdc_outX[7:0]**, with the format:

Bit 0 … 5	6	7
6-bit time-of-arrival data	1	0

2.2.1 Details of Operation.

The block diagram of the TDC is shown in Fig. 7a. The TDC contains a single 64-step delay line. This delay line generates 64 outputs, **TDC_taps[63:0]**, each of which is a replica of the Sample Clock, but at each step, the output is displaced from the previous step's output by 1/64 of a clock cycle.

The **TDC_taps[63:0]** signals are applied to the D-inputs of eight sets of 64-bit flip-flop register blocks; one register for each of the eight TDC channels. All 64 flip-flops of the register are simultaneously clocked by the low-to-high transition of the **DataStrobe[X]** input. The Encoder receives the 64 ouputs of the register, and generates a binary Gray code whose value can be equated to the number of the step in the 64-step delay line at which the **DataStrobe[X]** transition arrived.

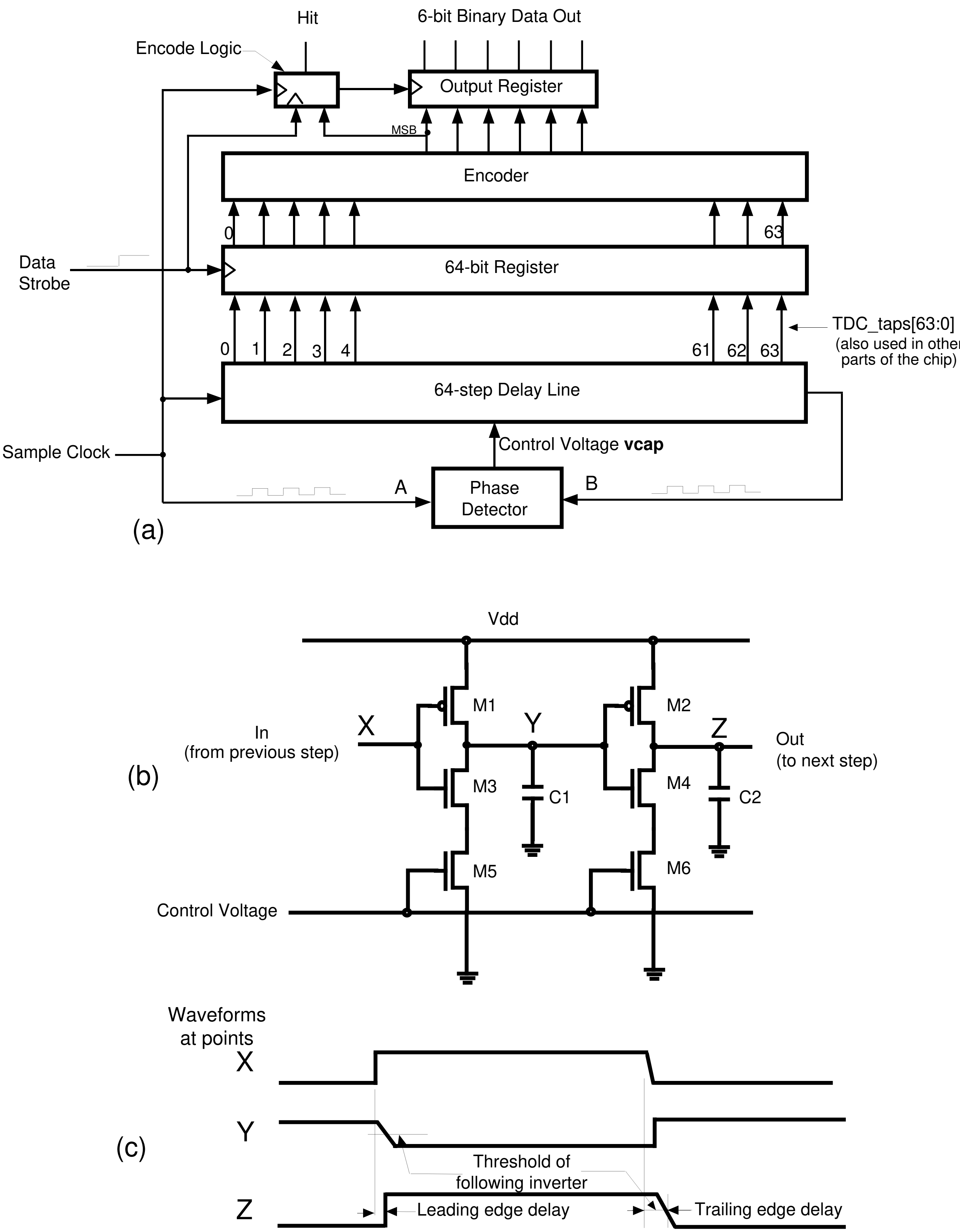

Fig. 7. a) Block diagram of the Time-to-Digital Converter (TDC). b) One delay element of the 64-step delay line. c) Waveforms in the delay element.

The Output Register saves the value of the Encoder Output in synchronism with the Sample Clock. A seventh clocked output records the fact that a "Hit" (a low-to-high transition of the input signal) occurred during the clock cycle corresponding to the current output of the Output Register.

2.2.2 Details of the Vernier

Each Sample Clock cycle is divided into 64 vernier periods in the block labeled "64-step Delay Line" in Fig. 7a. This is accomplished by means of a chain of 64 successive delay elements, each composed of two current-starved CMOS inverters.

A schematic of a single delay element is shown in Fig. 7b, with relevant waveforms shown in Fig.7c. Note that transistors M1 and M3 constitute a digital inverter. When input X changes from low to high capacitor C1 is discharged through transistors M3 and M5. The gate voltage of transistor M5, **vcap**, controls the amount of current available to discharge C1. The current through M5 therefore controls the falling edge delay of Y and, consequently, the rising edge delay of Z.

Transistors M2, M4, M6 in conjuction with capacitor C2, have a similar behavior except that they influence the trailing edge delay shown in Figure 7c. This influence is again under control of **vcap**. Thus the two inverters together can effect a delay on the total clock waveform, while preserving its duty cycle. Conceivably, two separate control voltages could be used; one being applied to the M5s and the other to the M6s of all the delay elements so as to be able to control the leading-edge and falling-edge delays independently. However, tests have shown that this added complexity is not necessary[7].

The Sample Clock signal is applied to the first element of the delay line. The delay of each element in the chain is adjusted by varying **vcap**, which is common to all 64 stages. The inverters are physically very nearly identical (of identical construction on a small area of silicon), so that the delay in each inverter is very close to the delay in every other inverter. Thus, if the total delay of the delay chain can be adjusted to be exactly one Sample Clock cycle, the delay in each element is expected to be 1/64 of a Sample Clock period, or 1.05 nsec. Deviations in the delay per each element caused by processing variations will be much smaller than the width of one bin. Measurements on a test chip showed a jitter of < 50 psec, and a differential non-linearity of 130 psec (see reference in footnote 3).

2.2.3 The Phase Detector

The phase detector for the delay-locked loop, shown in Fig. 8a, performs an automatic adjustment of **vcap**. It examines the time difference (phase) between the edges of the Sample Clock signal and of the replica that has traversed the delay chain. If the time difference is not zero, then the phase detector adjusts **vcap** so as to make it zero. The outputs from successive taps are then replicas of the Sample Clock signal, delayed successively by 1/64 of a clock period.

Fig. 8c diagrams the P and N pulses needed to drive the charge pump, which are generated by a phase detector based on the Muller C-element[8], shown in Fig. 8b. Figure 8c shows relevant idealized waveforms for the complete phase detector. Note that the output of the C-element follows whichever of the inputs, A or B, lags the other. The only condition for this behavior is that the inputs overlap.

---

[7] D.M. Santos, et al, "A CMOS Delay Locked Loop and Sub-Nanosecond Time-to-Digital Converter Chip," IEEE Transaction on Nuclear Science, Vol 43, No, 2, pp 1717-19, June 1996

[8] See, e.g., Mead and Conway, *Introduction to VLSI systems*, Addison Wesley, 1980, pg. 250, 254.

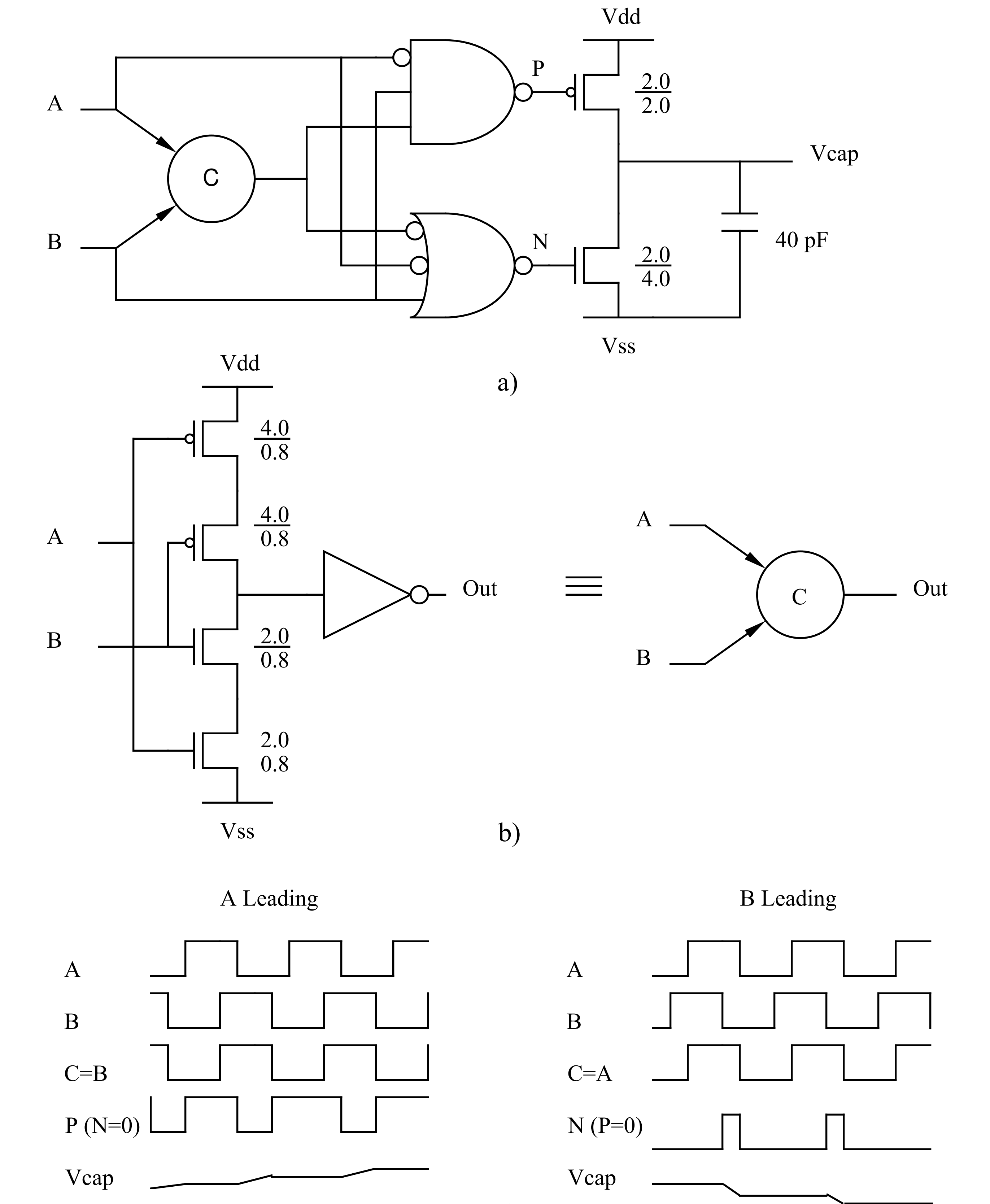

Fig. 8. a) TDC delay-locked loop phase detector. b) Muller C-element. c) Idealized waveforms showing proportional control.

As implemented, the circuit operates on the falling edge of waveforms A and B. Feedback is negative; a positive change in **vcap** decreases the delay. Under lock only very small amounts of charge must be injected into the **vcap** node to compensate for charge lost through leackage on capacitor C_h. This makes the P and N signals extremely small spikes of current.

The analog control signal, **vcap**, is buffered by the *opamp_buffer* and output from the Elefant chip as **vctrl**, so that its value can be monitored external to the chip. In addition, a digital signal, **lock**, indicates when the 64-step delay loop is in "lock," i.e., its total delay is exactly equal to one Sample Clock cycle.

2.2.4 Use of Vernier Signals by Control Logic

Since the 64 outputs of the vernier represent 64 phases of Sample Clock, they are also used for timing functions in the *control_logic* block, where the 15 actual signal used are input as **timing_taps[14:0]**.

2.2.5 Details of the Encoding

Figure 9 illustrates a simplified example of the the TDC operation. In this example, the delay loop is assumed to have only eight taps, and the clock cycle is divided into eight increments, which are labelled 0 through 7.

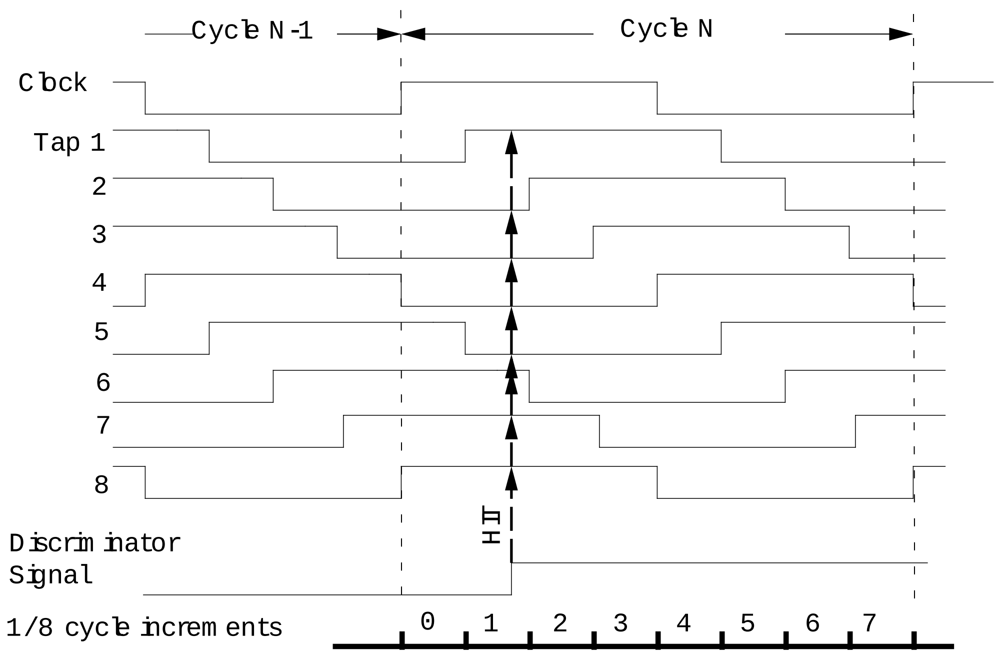

Fig. 10. Waveforms for an example of a TDC having an eight-step vernier. The states ("1" or "0") of the eight taps are sampled and encoded at the time of a Hit.

A Hit is assumed to arrive during increment 1. Note that the set of states of the eight tap signals is unique to that increment. The states of Taps 2, 3, 4 and 5 are low; the rest are high. The encoder results for all increments of delay are listed in Table I.

Note that, for all values of X, four flip-flops in the D-register (an eight flip-flop register in this example) are set, and four are reset. The encoder operates by scanning for the first 1 ->0 transition. Each value latched in the D-register has a unique position for this 1 -> 0 transition therby allowing a unique encoded value.

Table I

Values latched in D-register and Encoder Outputs
X is the hit delay, in units of 1/8 clock cycle.

X	Latched in D-register								Encoder Output		
	1	2	3	4	5	6	7	8	2	1	0
0	0	0	0	0	1	1	1	1	0	0	0
1	1	0	0	0	0	1	1	1	0	0	1
2	1	1	0	0	0	0	1	1	0	1	0
3	1	1	1	0	0	0	0	1	0	1	1
4	1	1	1	1	0	0	0	0	1	0	0
5	0	1	1	1	1	0	0	0	1	0	1
6	0	0	1	1	1	1	0	0	1	1	0
7	0	0	0	1	1	1	1	0	1	1	1

2.2.6 Metastability in the TDC

Because the TDC is a clocked circuit featuring asynchronous **DataStrobe** inputs, its design must address the problem of metastability. This section describes our attempts to minimize the probability that a metastable state will cause an error in the output word.

The possibility of a meta-stable state can occur in two different section of the TDC. First, meta-stability can occur when the **DataStrobe**, which stores the state of the Sample Clock, tries to store a value that is in transition on the delay chain in one of the 64-bit register blocks. Meta-stability can also occur during the determination of which Sample Clock cycle the **DataStrobe** occured. Both cases will be discussed below.

The *Ctrllogic* block (part of the *TDC* block) is responsible for generating a correctly timed, synchronous "hitbit" for each asynchronous **DataStrobe** input. The schematic for the *Ctrllogic* block is shown in Fig. 10. The key signals in the *Ctrllogic* block are:

Phi_0 15MHz Sample Clock

DataStrobe_0 Asynchronous input for a given TDC channel

Hit_Out_5 Hitbit output which must become asserted one full clock cycle after the one in which the asynchronous **DataStrobe_0** input arrived. **Hit_Out_5** must stay high for exactly one clock cycle.

The strategy of the *Ctrllogic* block design is to latch the input on both edges of the clock, and then use the most significant bit (MSB) of the encoded output to steer only one latched input to the hitbit output. Thus, although one of the latched input signals may be metastable, the other will be suitable in

Fig. 10 TDC Ctrllogic block

Fig. 10. The TDC control logic.

forming the hitbit. This procedure assumes that the delay in the generation of the MSB does not exceed one-half of a clock cycle, as explained below.

A timing diagram that details the meta-stability problems associated with generating a hit bit is shown in Fig. 11. This timing diagram requires some explanation of the flip-flop elements in *Ctrllogic*. *DS_Latch* has the property that as long as its **Reset_bar** input is low, the output **DS_Latch_bar3** is high. When **Reset_bar** rises, the output will go low on the next rising edge of **DataStrobe**, and will stay low until the next falling edge of **Reset_bar** (independently of subsequent **DataStrobe** activity). The *Dff_Pos_res* circuit acts as a positive-edge triggered D flip-flop with one important exception: the output will fall to zero whenever the input is high and the clock is low. The *Dff_PosEdge* circuit is a standard positive-edge triggered dynamic D flip-flop.

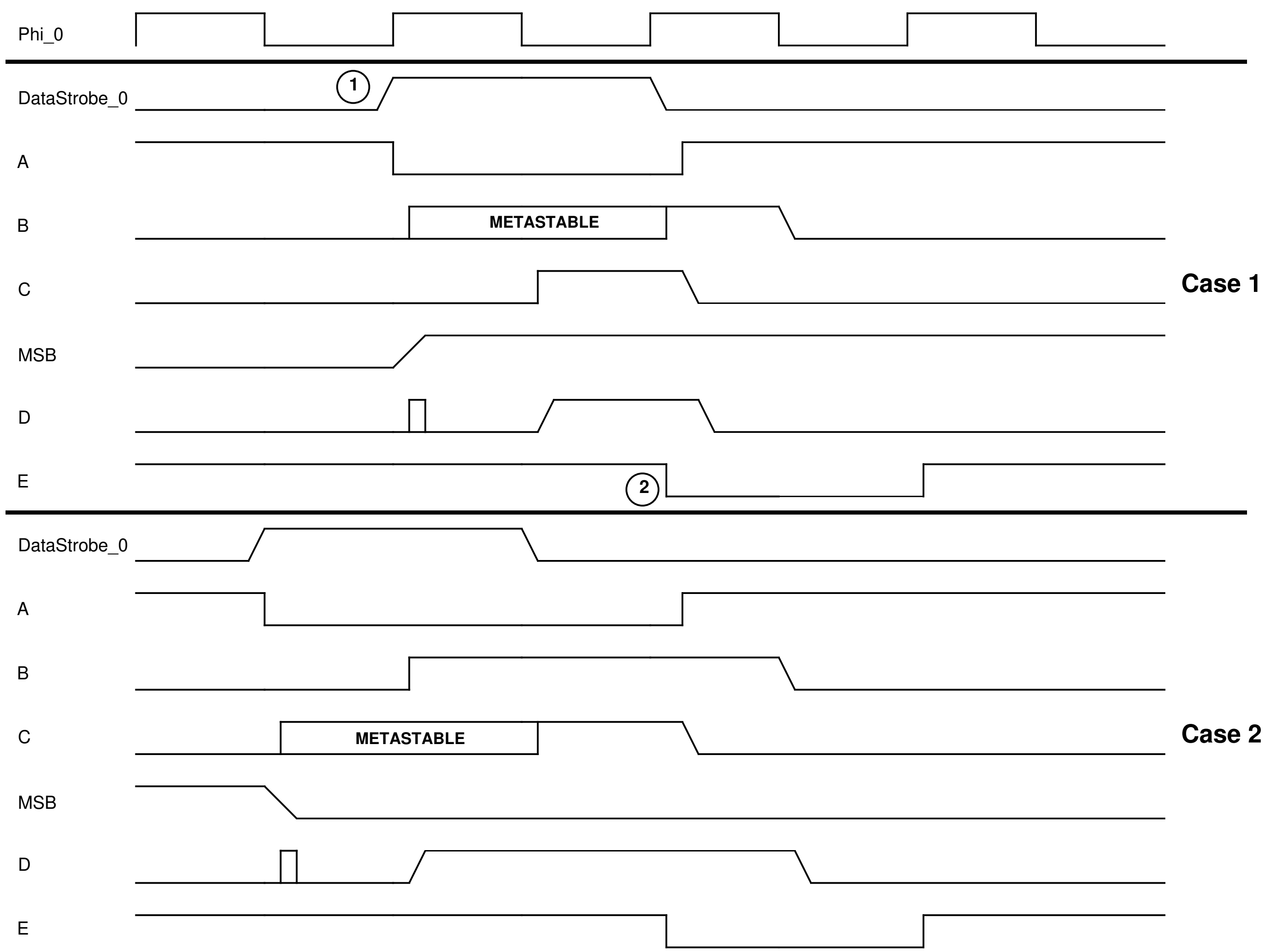

Fig. 11. Worst case scenarios for meta-stablility in the Hit bit. Case 1) The DataStrobe occurs "just" before a rising clock edge. Case 2) The DataStrobe occurs "just" before a falling clock edge. In both cases the encoder operates correctly but the timing of the DataStrobe is such that the latched output goes into a meta-stable state.

In Fig. 11, Case 1 depicts one of the worst-case scenarios for metastability in the hitbit. The input **DataStrobe** arrives just before a rising clock edge (1), causing the input(A) of flip-flops *Dff_Pos_Res_S_i0* and *Dff_Pos_Res_S_i1* to change at the same time as the clock. The output of *Dff_Pos_Res_S_i0* (B) is thus very likely to become meta-stable, whereas the output of *Dff_Pos_Res_S_i1* (C) will transition normally at the next falling clock edge.

The *Encode2c* MSB is valid (logical high) soon after the **DataStrobe** arrives, and steers signal C to the input (D) of flip-flop *Dff_PosEdge_S_i0*. At the next rising clock edge (2), signal E (The inverse of the Hit bit) falls, causing the *DS_Latch* output (A) to be re-asserted. The next falling clock edge causes signal B to fall, so that by the following rising edge signal D is low and causes signal (E) to rise. Thus, despite the arrival of the **DataStrobe** input at the transition between clock periods, the hitbit comes out at the correct time and persists for only one clock cycle.

Case 2 depicts the other worst-case scenario where the **DataStrobe** arrives close to a falling clock edge. This time, signal C may become meta-stable while signal B will be steered by the *encode2c* MSB to the input of *Dff_PosEdge_S_i0*. The timing is very similiar to case1 and the hitbit will be one cycle wide and correctly timed with respect to the **DataStrobe** input.

There is also the possibility of meta-stable states occuring in the outputs of the 64-bit delay-element register. As the delay chain clocks in the delayed version of the input clock, only two of the latches in the 64-bit register can go meta-stable. Of these two latches, only the 1 -> 0 transition is of importance because the encoder basis its decision on this transition.

Fig. 12 depicts a simplified schematic showing the pertinent bits of the register and encoder. The meta-stable state causes the 1 -> 0 transition point in the 64-bit register to be undefined. The meta-stable state is depicted as X in Fig. 12. The meta-stable state causes an error by turning on two legs of the encoder. However, since the 6-bit output number is Gray encoded, the meta-stable state can only cause an error of one 1.05 ns bin. Calculations indicate that a meta-stable state will only occur 0.5×10^{-3} % of the time; much smaller than the requirement imposed on the drift chamber electronics that each layer has less than 1 % loss of efficiency in reporting a hit.

2.3 Multiplexers; Data stream formatting

Each of the eight channels of the chip includes a 3-input MUX (multiplexer) whose task is to selectively accept digital data from three sources and thereby format the data stream that enters the DRAM. On Fig. 4, the eight MUXs are shown as *chan_mux* blocks. The three data sources and their designations in the *chan_mux* blocks are:

1) the FADC for that channel, shown as **bus_in0[7:0]**;

2) the TDC for that channel shown as **bus_in1[7:0]**; and

3) the upper 8-bits, **counter_out[9:2]**, of the 10-bit synchronization counter in the *contol_logic* block that counts System Clock cycles, shown as **bus_in2[7:0]**. Since the System Clock is a 60 MHz clock, the upper 8-bits count Sample Clock cycles. The counter is common to all eight channels.

Each channel's MUX independently steers one of the three inputs to its output according to steering commands impressed on two-bit input, **sel[1:0]**, as shown in Table II. The signal **sel[1]** is controlled by the external input **test_mode,** whereas **sel[0]** is individually derived for each channel from that channel's TDC **Hit** signal**.**

During normal data acquisition, **test_mode,** and hence **sel[1]**, is held in the low state. For each channel, **Hit** will be at high level only for a single clock cycle when the TDC has detected a wire chamber

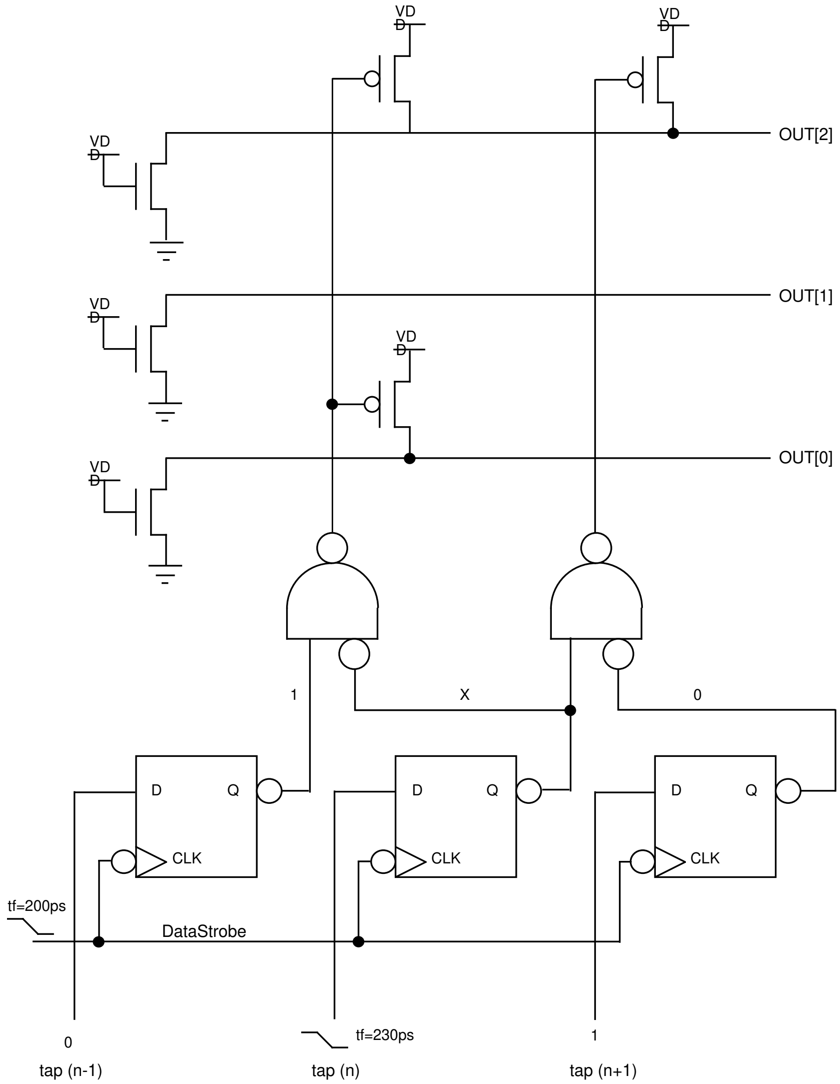

Fig. 12. Simplifed portion one 64-bit TDC register and encoder indicating meta-stable operating conditions.

Table II

For all channels **sel[1]** (**=test_mode**)	For each channel **sel[0]** (= TDC **Hit** for that channel)	Each channel's MUX output is derived from
0	0	Channel's FADC
0	1	Channel's TDC
1	Don't care	Sample Clock counter

hit. The output of the MUX, therefore, consists of a sequence, at Sample Clock rate, of FADC numbers, with a single TDC number substituted into the sequence in response to the TDC hit.

When the TDC number is substituted, the FADC number that would have gone into that time slot is disregarded. This does not represent a loss of data, since the *delta_gen* block is configured to insure that the FADC data is delayed by one Sample Clock cycle with respect to the TDC data.

When the external control signal, **test_mode** (and hence **sel[1]**) is set to "1," each MUX selects the **bus_in2[7:0]** input, and its output consists of a sequence of binary numbers in ascending order generated by the Sample Clock counter. When **sel[1]** is "1," the action of **sel[0]** is disabled. This mode is used for chip diagnostics. When **test_mode** is low the counter output to the MUX is disabled thereby decreasing the amount of noise broadcast during normal data acquisition.

The propagation time through the MUX is very small; it adds no additional clock periods to the data propagation time.

2.4 Trigger Data Output

Bytes of trigger data are output at Sample Clock rate on **trig[7:0]**. Each channel is represented in the byte by one bit. Under control of the externally generated **mode_trig** signal, the bits are either: a) the presence or absence of TDC **Hit**s for the present clock cycle or b) the result of the FADC's δ calculation.

The a) or b) selection is made in the *trigger_mode_ctrl* block. The TDC hits are input to the block via **tdc_topbit[7:0]**. The FADC δs are input via **tdc_topbit[7:0]**.

2.5 DRAM memory

A heuristic block diagam of the DRAM is given in Fig. 13. However, for completeness, the labelling on this block diagram corresponds to the actual blocks in the DRAM top level schematic, *DRAM2*. The 8-bit outputs from the eight MUXs are assembled into a 64-bit word and applied to the input of the *DRAM* block on **din[63:0]**. These inputs are then buffered by the *DRAM_Indr_array* to drive the *DRAM_memcore* write bit lines. The channels are in order, with channel 0 occupying the least significant eight bits; channel 7 the most significant eight bits. One 64-bit word is stored in the DRAM each Sample Clock cycle.

The DRAM is functionally organized as a FIFO of 192 words. It therefore has a fixed fall-through time that accomodates a Level 1 trigger latency time of 12.86 μsec.

The DRAM is a dual-port memory, having the ability to simultaneously perform read and write cycles, as long as the read and write addresses refer to different memory locations. In this application, it is continuously storing input data. To reduce power consumption, however, data is actually output from the DRAM only when being actively transferred to the SRAM (following a trigger). Some results from a DRAM test chip are given in a reference.[9]

2.5.1 DRAM Addressing

Within the *control_logic* block, the *dclog_DRAM_rw_ctrl* contains one 8-bit counter that generate the DRAM write and read addresses. The read and write addresses are then registered on the next Sample Clock edge. This insures, except immediately after a reset, that the read address is always one count smaller the write address. At **reset** both the counter and the registers are reset to 0. After two Sample Clock cycles the zeroes are flushed from the regeisters and the write counter maintains a value one less than

---

[9] "Test Results from the DRAM Chip," Edwin K. Su, Lawrence Berkeley Laboratory. Informal, undated report

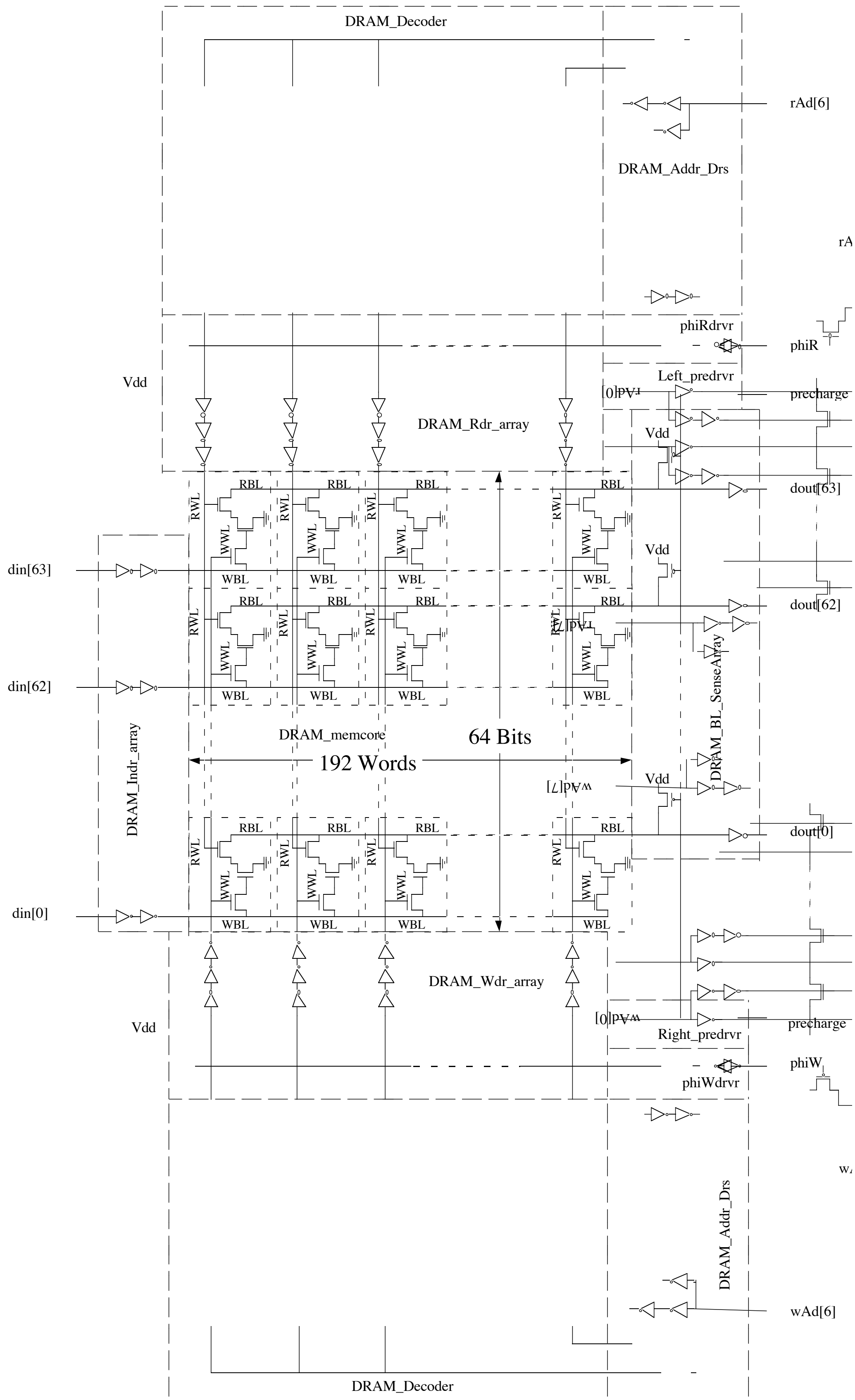

Fig. 13. DRAM heurisic schematic.

the read register. In the steady state, then, a memory location will be read 192 cycles after being written; that same location will then be over-written the following cycle.

As shown on Fig, 4, the write and read addresses are input to the *DRAM* block on **wAd[7:0] and rAd[7:0]**, respectively. Inside the DRAM, each address set is buffered by a *DRAM_Addr_Drs* block before entering a NAND-style *DRAM_Decoder*. The write address decoder output is buffered by *DRAM_Wdr_array* before entering the *DRAM_memcore*. The *DRAM_Wdr_array* also precharges the write *DRAM_Decoder* output using the **phiW** signal, which is buffered by *phiWdrvr*.

The read address decoder output is buffered by *DRAM_Rdr_array* before entering the *DRAM_memcore*. The *DRAM_Rdr_array* also precharges the read *DRAM_Decoder* output using the **phiR** signal, which is buffered by *phiRdrvr*.

2.5.1 DRAM Read Cycles

During a read cycle, output data, **dout[63:0]**, passes through the *DRAM_BL_SenseArray*, where it is only lightly buffered. The DRAM_BL_SenseArray also precharges the read bitlines using the precharge signal, which is buffered by *Left_predrvr* and *Right_predrvr*.

The **in[63:0]**, **wAd[7:0]**, and **rAd[7:0]** signals change on the rising edge of clk_15 (the internal Sample clock). Timing diagrams for **phiR**, **phiW**,and **precharge** are shown in Fig. 14. All timing signals are generated in the *dclog_timing_gen* block contained in the *control_logic* block.

2.6 Triggering; Transfer of Data Into SRAM and SRAM2

The SRAM and SRAM2 memories are within the *SRAM* and *SRAM2_module* blocks shown in Fig. 4. Additional details of the *SRAM* block are in Fig. 16.

The SRAM is a two-port memory of 128 64-bit words. Its write port is 64 bits wide; the read port is 8 bits wide. For both writing and reading, it is logically organized into four partitions, or buffers, of 32 words each. For reading, it is also logically partitioned into eight channels. In a write cycle, a 64-bit word containing one byte from each of the eight channels is written into memory. In a read cycle, a single byte of data is read from memory.

2.6.1 Triggering

At its output end, the DRAM is capable of outputting a 64-bit word at the Sample Clock rate. These words are the ensemble of the bytes from each of the eight channels that were input approximately 12 μsec earlier. When an **L1_accept** is received, an event packet of 32 of these words is read from the DRAM and written into the SRAM (Static Random Access Memory) via a 64-bit bus. This transfer occurs at the Sample Clock rate. The data transfer is controlled by the *dclog_SRAM_rw_ctrl* block which is contained in the *control_logic* block.

The packet of 32 words represents a time slice of 2.14 μsec which includes both pre-history and post-history for the trigger event.

2.7 SRAM Writing Operation

The filling of an SRAM buffer with data emanating from the DRAM is initiated by the receipt of a trigger at the chip's **L1_accept** input. As shown in Fig. 4, input data arrives via **din[63:0]** . The top level

Fig. 14. Timing Diagram

Fig. 14. Complete timing waveforms generated from the *dclog_timing_gen* block.

schematic of the SRAM is shown in Fig. 15. The input data on **din[63:0]** is buffered by *inbuf* before passing through *BLpullup* and into the *memcore* bit lines.

The write address, **waddr[6:0]**, is buffered by *addr_drivers* before entering a NAND-style decoder. The decoder output is buffered by *WWLdrivers* before entering the *memcore*. *WWLdrivers* also precharges the write address decoder output using the **wPC_IN** signal, which is buffered by *wIOdr*.

The **din[63:0]** and **waddr[6:0]** signals change on the rising edge of Sample Clock. Timing diagrams for relevant signals are shown in Fig. 14.

The 7-bit write address is composed of two fields:

1) The trigger is accompanied by a 2-bit number, arriving via the chip's **buff_wr[1:0]** inputs; this number becomes the highest two bits, **waddr[6:5]**, of the write address, and thereby selects which of the four buffer spaces in SRAM the data will be written into.

2) The trigger also zeros a 5-bit write address counter in the *dclog_SRAM_rw_ctrl* block. The counter contents is output as **sram_waddr[4:0]** from the *control_logic* block and becomes **waddr[4:0]**.

Following the trigger, the SRAM write is enabled for 32 sample clock cycles by the *dclog_SRAM_rw_ctrl* block; the write address counter is incremented each cycle. When the 5-bit counter turns over to zero count, it disables itself and the write strobes. The buffer is now filled and the process rests until the next trigger.

Buff_wr[1:0] is registered on **L1_accept**. Another **L1_accept** signal should not be issued until the chip has finished its response to the previous Trigger; approximately 2.2 μs. The external trigger controller is also responsible for managing the four buffer spaces - e.g., it should not overwrite a buffer that has not been emptied.

2.8 Writing into SRAM2

The SRAM2 is organized as four 24-bit words. The four 24-bit words are logically equivalent to the four buffers of the SRAM and each word contains three pieces of 8-bit data as shown below. **Buf_wr[1:0]** selects the write buffer for both the SRAM and SRAM2. **Buf_rd[1:0]** selects the read buffer for both the SRAM and SRAM2. This insures the SRAM and SRAM2 buffers are always synchronized.

Field:	Byte 0	Byte 1	Byte 2		
Bit No.:	0 7	8 15	16 20	21 22	23
	HIT MAP	SYSCLK COUNT	TRIG TAG	BUFF#	0

2.8.1 The HIT MAP Field - Byte 0

The HIT MAP field, bits 0 through 7, represents a map of the hits in each of the eight channels (0 through 7). For a given channel, hits are detected by scanning the data during an event transfer of 32 words as it is being written into the current buffer in the SRAM. The scanning is done in the *dclog_hit_decode* block; its output is **hitmap[7:0]**.

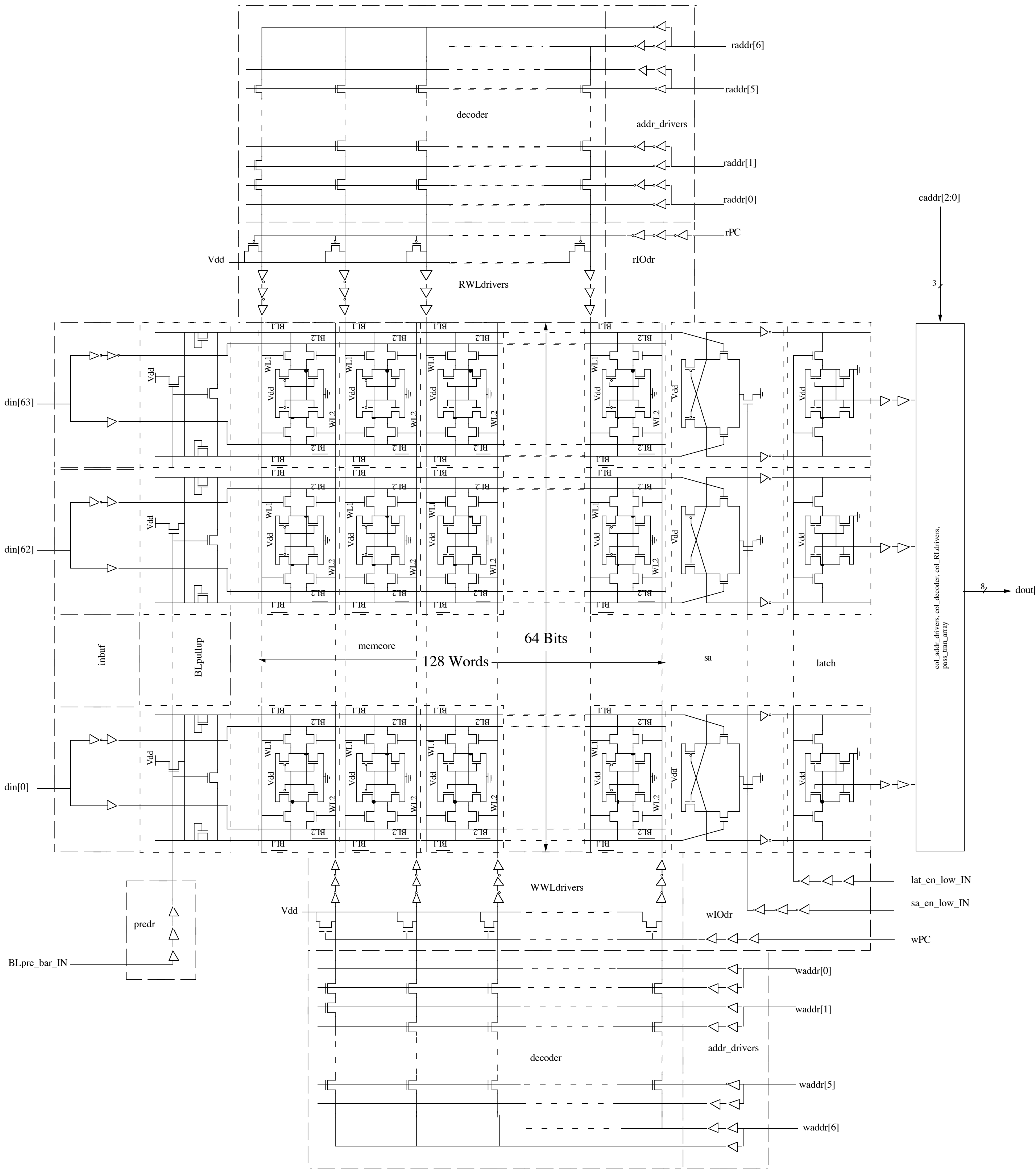

Fig. 15 SRAM heuristic schematic.

There are two definitions of a hit, one or the other is selected by the externally supplied **mode_sparse** signal. If **mode_sparse** = 0, a hit is the presence in that event data packet of one or more

TDC hits in that channel; if **mode_sparse** = 1, a hit is the presence of at least one $\delta = 1$ in that channel. If a HIT MAP field bit is 0, a hit was not detected in the corresponding channel in that event data packet.

The *dclog_hit_decode* block contains a Hit Register that consists of eight clocked Set-Reset flip-flops, one per channel. The Set input of each flip-flop is connected to either bit 6 or bit 7 of the data byte signals from its particular channel, as dictated by the state of **mode_sparse**. All flip-flops are reset at the receipt of an **L1_accept**. If, as the SRAM filled, a "1" is present on its Set input for any of the 32 bytes in the channel's data packet, then the corresponding flip-flop is set, and remains set until reset at the start of the next **L1_accept**.

After filling the buffer, the state of the eight flip-flops in the Hit Register is transferred into the HIT field of the current write buffer of the SRAM2 on **hitmap[7:0]**.

During readout, the HIT field is available to the external readout controller, and can be used to effect sparse readout - i.e., the read out of SRAM data only from those channels, if any, that were hit in that particular event (See Section 3.5.2.).

2.8.2 The COUNT Field - Byte 1

The COUNT field contains the contents of the System Clock counter (in the *control_logic* block) latched on the most recent **L1_accept**. Ideally, the COUNT field for this trigger from all Elefant chips will have identical contents. This value is input to SRAM2 via **internal_count[7:0]**.

2.8.3 The TRIG TAG and BUFF# Fields - Byte 2

Byte number 2 contains a 5-bit field in which the event number for the trigger that filled the current buffer is recorded. This event sequence number is input from an external source on **tag[4:0]** at the instant the **L1_accept** is received by the chip. The number is input to SRAM2 on **trigger_tag[4:0]**.

Byte number 2 also contains a 2-bit field in which the current buffer number is recorded. This number is input from an external source on **buf_wr[1:0]** at receipt of **L1_accept**. It is input to SRAM2 as **waddr[1:0]**. This presence of this number in the data aids downstream processors in following the history of the data.

2.9 Readout of SRAM and SRAM2

The output ports of the SRAM and SRAM2 are byte wide. From an output standpoint, the four buffer spaces of the SRAM are also divided into eight (byte-wide) channel spaces.

The readout of data from SRAM and SRAM2 is completely under control of the downstream readout controller. The data is output from the chip via tri-state drivers on eight *Tristate Pads*, that are connected to an 8-bit data bus on the PC card. The relevant signals are discussed below and are listed in the Data Output Signals section of Table V.

In the readout protocol, the controller specifies the read buffer number**, buf_rd[1:0],** and the channel number, **ch_sel[2:0],** containing the byte data to be transferred to the downstream electronics. Having specified the channel-buffer space, the controller can then reset the chip's byte address register to zero and begin reading data. Each read cycle automatically increments the address register.

2.9.1 SRAM Read Timing

Following is a description of the signals by which the external readout controller accesses the event data from the Elefant chip. Note that all external signals are registered on th rising edge of Sample Clock.

Chip Enable - The *Tristate Pads* (see Fig. 4) output byte-wide signals on **databus[7:0]**. The tri-state drivers are in the high-impedance state except when the externally applied chip enable, **cs**, is asserted by the readout controller. Likewise, all readout control signals are ignored when **cs** is de-asserted.

Choice of SRAM or SRAM2 - The data sources within the chip are SRAM and SRAM2. One or the other is chosen by the readout controller via the **SRAMen** signal. This signal also directs each *Tristate Pad* to accept its input signal from the **A** or **B** input. If **SRAMen** is de-asserted, the pads accept data on their **B** inputs from the *output_MUX* block **out[7:0]** bus which is the output from the SRAM2.

Selection of Read Buffer - In addition, regardless of the state of **SRAMen**, the two-bit input, **buff_rd[1:0]** specifies which of the four buffer spaces are to be the source of the data.

Selection of SRAM2 Data - As shown in Table III, when **SRAMen** is in the low state (SRAM2 selected), the lower two-bits of the channel select field, **ch_sel[1:0]**, specify which of the three one-byte fields of SRAM2 (see Section 2.8) is to be output.

Table III

Value of **ch_sel[1:0]**	SRAM2 Byte Selected	Byte Contents
0	Byte 0	Hit Map
1	Byte 1	Clock Counter
2	Byte 2	Trigger Tag
3	Not specified	

Selection of the Data Output Byte in SRAM- Although SRAM is written with 64-bit words, it is read out in bytes. The read addressing scheme includes three fields which have effect only when **SRAMen** is asserted (SRAM selected). Table IV summarizes the SRAM readout signals. The three fields, all under the control of the external readout controller, are:

1) the 2-bit field, **buff_rd[1:0]**, selects one of the four read buffers, each containing 32 64-bit words.

2) a 5-bit counter in the *control_logic* block generates **sram_raddr[4:0]**, which specifies one of the 32 words in the buffer. The counter is initially reset to zero by **data_start**. As data is read out, the counter is incremented under the control of **data_inc_word**.

3) the 3-bit field, **ch_sel[2:0]**, selects one of eight bytes in the 64-bit word, thereby specifying which signal channel originated the data byte.

Table IV

Signals for controlling readout of chip.

SIGNAL NAME	Function when reading out SRAM2	Function when reading out SRAM
Chip Enable, **cs**	Enables the chip for readout	
SRAMen	Selects SRAM2 when LOW	Selects SRAM when HIGH
buff_rd[1:0]	Specifies the SRAM2 buffer to be read	Specifies the SRAM buffer to be read
ch_sel[2:0]	Lower two bits select SRAM2 field (See Table III)	Specifies one of eight signal channels
data_start	Not used	Zeros the read address counter
data_inc_word	Not used	Enables the incrementing of the read address counter

2.9.2 SRAM Read Timing

The read address, **raddr[6:0]**, is buffered by *addr_drivers* (see Fig.15) before entering a NAND-style decoder. The read address decoder output is buffered by *RWLdrivers* before entering the *memcore*. *RWLdrivers* also precharges the read address decoder output using the **rPC_IN** signal, which is buffered by *rIOdr*.

During a read operation, output data column decoding is accomplished with the column address **caddr[2:0]**, which is buffered by *col_addr_drivers* before entering the NAND_style *col_decoder*. The decoder output is buffered by *col_RLdrivers* before entering the *pass_tran_array*. The *col_RLdrivers* block also precharges the *col_decoder* output using the colPC_IN *signal*, which is buffered by **rIOdr**.

The *BLpullup* block precharges the read bit lines using the **BLpre_bar_IN** signal, which is buffered by *predr*.

Output data, **dout[7:0]**, passes through the *sa* (sense amplifiers) and *latch* blocks before output column decoding takes place inside *pass_tran_array*. Sense amplification is controlled by *sa_en_low_IN*, while output data latching is controlled by *lat_en_low_IN*. Both signals are buffered by *wIOdr*.

The **raddr[6:0],** and **caddr[2:0]** signals are expected to change on a rising edge of Sample Clock. Timing diagrams for relevant signals are shown in Fig. 14.

2.10 Control Logic

As shown in Fig. 4, the *control_logic* block is central to the control of many chip operations. The schematic, showing the organization of the block, is shown in Fig. 16.

The *control_logic* contains four subcircuits:

dclog_fast_control: contains the 10-bit binary counter from which the 15MHZ Sample clock is derived. Synchronization of **reset**, **sync**, and **trigger** inputs to the System Clock also occurs here.

dclog_DRAM_rw_ctrl: provides the DRAM read and write address pointers.

Fig. 16 Control Logic Block

Fig. 16. The *control_logic* top level schematic.

dclog_SRAM_rw_ctrl: provides the SRAM write address during data transfer from DRAM; provides the SRAM read address during readout; enables dclog_timing_gen to generate the SRAM and SRAM2 read control signals during readout.

dclog_timing_gen: derives all clocks which are generated from the TDC delay-locked loop (DLL) taps.

The control_logic input signals are:

sys_clk	the 60MHz clock.
trigger	initiates data transfer from the DRAM to SRAM event buffer.
sync	resets the 10-bit counter for chip synchronization.
reset	global chip reset.
cs	chip select which enables the SRAM and SRAM2 read control signals.
SRAMen	enables either the SRAM or the SRAM2 read control signals when **cs** is asserted.
test_mode	enables **counter_out[9:2]** for memory buffer testing.
timing_taps[14:0]	TDC DLL taps used to make memory and FADC control signals.
data_start	resets the SRAM read address.
data_inc_word	allows the SRAM read address to be incremented.

The control_logic output signals are:

sample_clk	15MHz clock derived from the **sys_clk.**
trig_pulse	60MHz-referenced trigger pulse.
reset_int	60MHz-referenced internal reset pulse.
dram_raddr[7:0]	DRAM read address.
dram_waddr[7:0]	DRAM write address.
sram_raddr[4:0]	lower 5 bits of SRAM read address.
sram_waddr[4:0]	lower 5 bits of SRAM write address.
counter_out[9:2]	upper 8 bits of 10-bit chip counter used for memory testing.
reg_out[7:0]	lower 8 bits of 10-bit chip counter used to form the SRAM2 COUNT field during a trigger event.

Timing diagrams for the FADC, DRAM, SRAM, and SRAM2 control signal outputs are shown in Figure 14.

3.0 Operational Procedures

3.1 Power Up

Following is a possible start-up and testing sequence:

1) At start-up, the Sample Clock will be operational;

2) Issue a **reset** pulse. On the next Sample Clock cycle, the Test and Clock counters, the DRAM read and write address counters and the SRAM write address counter are reset;

3) Issue a **sync** pulse;

4) **tdc_reset** should be pulsed and then **lock** should be monitored to see that DLL lock has occurred. (The TDC vernier delay loop is used to derive certain timing signals necessary for the chip's proper operation.)

The chip is now fully operational. Data is being continually stored in the DRAM, overwriting data stored 192 clock periods earlier. The test and clock counters are counting Sample Clock pulses.

3.2 Testing

When the **test_mode** signal is set to the "1" state, the eight input MUXs are steered by the **sel[1]** signal to output the test counter data, thereby filling the DRAM with sequences of binary numbers. The same value is written into each of the eight byte fields currently addressed - i.e., all channels receive the same data.

Now, triggers start to arrive, according to the protocol given in Section 3.4. Each **trigger** is accompanied by a trigger tag, **trig[4:0]**, and by **buff_wr[1:0]** to specify a buffer number. The reading of the DRAM is enabled and, for each channel, 32 64-bit data words are read from the DRAM and transferred to the specified buffer of the SRAM. When the buffer is full, the transfer from DRAM to SRAM is disabled until the next trigger arrives.

When one or more buffers are filled, readout of data may commence. The readout protocol is given in Section 3.5.1.

The system controller checks the data read out for consistency. The channel data should consist of sequences of binary numbers; the same sequence for all eight channels. The HIT map read from SRAM2 should be consistent with the values of the numbers in the packets and the current value of **mode_sparse**. Likewise, the **trig[7:0]** output data should also be consistent with the values of the numbers in the packets and the current value of **mode_trig**.

The system controller can issue Sync pulses at its pleasure to re-synchronize the count in the Test counter.

If the consistency checks are passed, the **test_mode** signal is lowered. The chip is now ready to be prepared for normal data-taking operation.

3.3 Initialization for Normal Operation

To prepare the Elefant chip for normal data taking, the following signals must be established by the controller. The order of establishment is not important.

1) Establish **top_ref**, **mid_ref** and **bot_ref**, the ladder reference voltages for the FADCs;

2) Establish **thresh[1:0]**, the threshold for the FADC δ calculation;

3) **mode_trig** should be set high or low, depending on the type of trigger data to be output;

4) **mode_sparse** should be set high or low, depending on the type of sparsification desired.

Now, after these steps are completed, data derived from the wire chamber signals is being output from the FADCs and the TDCs and is being continuously written into the DRAM. DRAM data older than the trigger latency period is being overwritten. Fast trigger data is output. Triggering on **L1_accept** signals and the readout of buffered data proceeds as described below.

3.4 Triggering Protocol

The result of a trigger is to store an event packet of data in SRAM, and a 24-bit word of ancillary information in SRAM2. Following are the steps the downstream controller must perform when issuing a trigger to the Elefant chip.

1) Establish the **buff_wr[1:0]** word at the input to the chip. The value of this word specifies which of four buffers in SRAM and SRAM2 will hold the event packet. This word must be asserted during the rising edge of **L1_accept**;

2) Establish the **tag[5:0]** word at the input to the chip. This tag is typically an event serial number, modulo 32. This number is stored in the SRAM2 buffer chosen in step 1). This number is also latched on the rising edge of **L1_accept**;

3) Pulse the **L1_accept** line. The **L1_accept** is internally sampled with respect to the leading edge of the System Clock signal.

Steps 1), 2) and 3) can be repeated as soon as 33 Sample Clock cycles have elapsed since the last **L1_accept**, provided buffer space is available. Since the chip has a capacity of only four buffers, the controller must insure that a buffer is not over-written with a new packet before the previous packet has been read out. Internally to the IC the trigger operation is synchronized to the Sample Clock. This happens because data flows only at the Sample Clock rate.

3.5 The Readout Protocol

In normal operation, the readout controller will read out from the chip the data that was stored in SRAM and SRAM2 in response to the receipt of a trigger - i.e., one event data packet stored in one of the four memory buffers. The read out of an event data packet can commence as soon as the packet has been completely stored in the SRAM, or it can be delayed at the pleasure of the readout controller. The readout controller also has the option of performing a complete readout, or a sparse readout, sequence.

3.5.1 Complete Readout

A complete readout is one in which all data bytes of the event data packet are read out - i.e., all three bytes from SRAM2 and all 32 bytes of all eight channels from SRAM. The protocol for a complete readout is as follows:

1) Assert chip select, **cs**. Hold it for the duration of the readout.

2) Set **buff_rd[1:0]** to select one of the four buffers to be read. Hold this signal for the duration of the readout.

3) Set **SRAMen** to LOW level, thereby selecting the SRAM2. Hold it for the next three steps, where each step is a minimum of one Sample Clock cycle.

4) Set **ch_sel[2:0]** to 0, read HIT field of the SRAM2.

5) Set **ch_sel[2:0]** to 1, read COUNTER field of the SRAM2.

6) Set **ch_sel[2:0]** to 2, read TRIG TAG byte of the SRAM2.

7) Set **SRAMen** to HIGH level, thereby selecting the SRAM. Hold it until the readout of the SRAM is completed.

8) Set **ch_sel[2:0]** to 0, thereby selecting channel 0's data.

9) Pulse the **data_start** signal, thereby setting the SRAM read address counter to 0 at the next Sample Clock transition. Read the zeroth data byte of the selected channel in the buffer selected in step 2).

10) Access a subsequent byte. To access a subsequent byte, the **data_inc_word** signal must be true on the 0->1 transition of Sample Clock. (Thus, if **data_inc_word** is held asserted, data bytes will be output at Sample Clock rate. Alternatively, **data_inc_word** pulses can be issued at a rate less than Sample Clock (e.g., 1/2 Sample Clock rate) to slow the output byte stream.)

11) Repeat step 10) 30 more times. At the end of this step, all 32 data bytes from the addressed channel will have been read out.

12) Increment the value of **ch_sel[2:0]**, thereby selecting the next channel.

13) Repeat steps 9) through 11). Note: The daring may wish to repeat only steps 10) through 11), since the SRAM read address counter has a maximum count of 31. However, if the full 32 bytes per channel are not read out, step 9) must be included.

14) Repeat steps 12) and 13) six more times. At the end of this step, all event data from the specified packet will have been read out.

15) De-assert **cs**.

These 15 steps will result in a stream of data bytes as shown in Fig. 17.

Most of the time, either zero or one of the data bytes from a channel will be a TDC byte; the rest will be FADC bytes. However, since the preamp peaking time is short compared to the time interval represented by a packet, it is possible that there may be more than one TDC byte in some packets.

3.5.2 Sparse Readout Sequences

In an alternate, sparse, sequence, the readout controller examines the Hit map in the first byte and, based on the content of that byte, selectively reads out only the channels that contain hits. In this sequence, the first seven steps as in the complete readout sequence will be performed. Then:

8) If no channels have a hit, as determined by reading the HIT map, terminate the readout now by going to step 13). Otherwise,

9) Set **ch_sel[2:0]** to a channel with a hit, as determined by reading the HIT map.

10) Pulse the **data_start** signal, thereby setting SRAM read address counter to 0. Read zeroth data word of the selected channel in the buffer selected in step 2).

11) Access a subsequent byte. See step 10) in section 3.5.1.

HIT map

COUNTER
TRIG TAG/BUFF NUMBER
Channel 0 data byte 0 • • • Channel 0 data byte 31
Channel 1 data byte 0 • • • Channel 1 data byte 31
• • • • •
Channel 7 data byte 0 • • • Channel 7 data byte 31

Fig. 17. Sample data stream output.

12) Repeat step 11) 30 more times. At the end of this step, all 32 data bytes from the addressed channel will have been read out.

13) Repeat steps 9) through 12) as many times as there are more channels with hits.

14) De-assert **cs**.

3.7 Operation with Multiple Chips on the PC card

There will be typically be several Elefant chips on a PC board. In normal data-taking operation, each chip is continually accepting and processing signals from the analog section that originated at the chamber wires.

It is anticipated that triggers will be delivered simultaneously to all chips on the board, with identical event serial numbers and write buffer number assignments for all chips.

The readout of the chips is done individually. Although the read address fields are delivered to all chips at the same time, only one chip will have its Chip Select, **cs**, active at any one time. All chips are read out onto the same 8-bit external data bus. For this reason, the chip's bus drivers are be of the tri-state variety. An inactive Chip Select puts the drivers into the high-impedance state.

4.0 Signal List

A complete list of the chip signals (excluding power and ground) is given in Table V. All pin assignments are given in the table.

Table V - Signal List

Signal assignments for all pins of the Elefant chip, ordered by functional families.

GENERAL

Digital Inputs

Description - Unless otherwise noted (by "analog" or "special", all digital inputs are CMOS voltage levels and positive logic.

Electrical Characteristics (CMOS inputs)

Input HIGH voltage	3.0 V (min); V_{DD} (max)
Input LOW voltage	0.0 V (min); 0.8 V (max)

Logical Characteristics (Positive logic)

Logic ZERO (de-assertion)	Input LOW
Logic ONE (assertion)	Input HIGH

Digital Outputs

Description - Unless otherwise noted (by "analog"), all digital outputs are CMOS voltage levels and positive logic. All such outputs have rise and fall times slew-rate limited.

Electrical Characteristics (CMOS outputs)

Output HIGH voltage	V_{DD}
Output LOW voltage	GND
Rise and fall times (slew-rate limiting)	5 nsec (min); 20 nsec (max)

Logical Characteristics (Positive logic)

Logic ZERO (de-assertion)	Output LOW
Logic ONE (assertion)	Output HIGH

FADC Signal Set

top_ref Pin 94 (Analog Input)

Description - Reference voltage applied to top end of resistive ladders in all eight FADCs. Typically a dc voltage, but analog signals can be used. This signal defines the input signal level that corresponds to a full-scale binary output (decimal value = 63) from the FADC. If connected to an analog signal having time correlation with the input signal, a non-linear characteristic can be achieved.

Electrical -

Maximum operating value	4.3 volts
Typical operating value	4.0 volts
Maximum allowable value	5 volts
Minimum allowable value	0 volts
Typical input impedance	400 ohms resistive

mid_ref Pin 95 (Analog Input)

Description - Reference voltage applied to the midpoint of resistive ladders in all eight FADCs. This signal defines the input signal level that corresponds to half-maximum binary output (decimal value = 31) from the FADC.

Usage - The use of this signal is optional. If left floating, the **mid-ref** signal assumes a value midway between **top_ref** and **bottom_ref**, and the FADCs have linear analog-digital conversion characteristics.
If connected to a dc voltage source, the FADCs will have a two-slope conversion characteristic, with the two slopes (one for output range 0 to 31; the other for output range 32 to 63) determined by the values of **top_ref, bottom_ref** and **mid_ref.**

Electrical Characteristics

Typical value (floating)	2.5 V ([top_ref + bottom_ref]/2)
Typical value (connected to dc source)	1.75 volts
Maximum allowable value	5 volts
Minimum allowable value	0 volts
Typical input impedance	200 ohms resistive

ap[7:0] and am[7:0] (Analog Inputs)

Signal Name	Pin Number
ap[0]	6
am[0]	5
ap[1]	8
am[1]	7
ap[2]	9
am[2]	10
ap[3]	11
am[3]	12
ap[4]	22
am[4]	21
ap[5]	24
am[5]	23
ap[6]	25
am[6]	26
ap[7]	27
am[7]	28

Description - Differential analog signal inputs to the eight FADCs. These are the shaped signals from the analog shapers in the Amplifier ICs. Pair **ap(4)** and **am(4)** are the signal pair for FADC4, etc. Each pair carries low-level signals that are received, amplified and converted to single-ended signals by the **analog_input_conv** block before being input to the FADCs. (The differential gain of each **input_conv** channel is 10.)

Electrical Characteristics -

AC coupled signal range on **ap** pins	±0.15 volts
AC coupled signal range on **am** pins	±0.15 volts
Maximum allowable signal range on either pin	0 to +5 volts

bottom_ref Pin 96 (Analog Input)

Description - Reference voltage applied to the bottom end of resistive ladders in all eight FADCs. This signal defines the input signal level that corresponds to a zero binary output from the FADC.

Electrical Characteristics

Typical value	1.0 volts
Maximum allowable value	5 volts
Minimum allowable value	0 volts
Typical source impedance	400 ohms resistive

TDC SIGNAL SET

dp[7:0] and dm[7:0] (Special Digital Inputs)

Signal Name	Pin Number
dp[0]	1
dm[0]	2
dp[1]	3
dm[1]	4
dp[2]	16
dm[2]	15
dp[3]	14
dm[3]	13
dp[4]	17
dm[4]	18
dp[5]	19
dm[5]	20
dp[6]	32
dm[6]	31
dp[7]	30
dm[7]	29

Description - Differential low-level digital signal inputs to the eight TDCs. These are the signals from the discriminators in the Amplifier ICs. Pair **dp(4)** and **dm(4)** are the signal pair for TDC4, etc. Each pair carries low-level signals, typically ± 0.08 Volts, that are received, amplified and converted to chip-level logic signals by the **input_conv** block before being input to the TDCs. (The differential gain of each **input_conv** channel is 8.)

Electrical Characteristics -

Typical binary Zero/One signal levels on **dp** pins	4.92/5.0 volts
Maximum allowable signal range on either pin	0 to +5 volts
Termination impedance, each pin	200 ohms to ground

tdc_reset Pin 37 (Digital Input)

Description - When asserted, forces the DLL (delay locked loop) control voltage for all TDCs to a high value. When de-asserted, the DLL will quickly find its proper operating value for the control voltage.

Electrical Characteristics

Minimum pulse length	67.2 nsec

vctrl Pin 35 (Analog Output)

Description - Allows external measurement of the DLL control voltage. For diagnostic purposes.

Electrical Characteristics

Output voltage range	0 V to 5 V
Typical output voltage when DLL operating	2.5 V
Maximum load	1 Mohms, 50 pF

lock Pin 36 (Digital Output)

Description - Allows verification that the DLL has achieved a stable lock condition---i.e., **vctrl** is in the range of about 1.5 to 3.0 V. For diagnostic purposes.

Electrical Characteristics

Voltage when locked	High signal level

TRIGGER DATA OUTPUT

trig[7:0] Pins 90 down to 83, respectively (Digital Outputs)

Description - Each single-bit signal denotes the current state of a predefined trigger data output condition, in its channel, of relevance to the trigger processor; **trig(3)** carries the state of the condition in signal channel 3, etc. The criteria of the conditions are described in the **mode_trig** signal description.
The states are updated each Sample Clock cycle (every 67.2 nsec).

Electrical Characteristics
Normal CMOS slew-rate limited outputs

mode_trig Pin 93 (Digital Input)

Description - Selects one of two predetermined trigger output data conditions that are output by the set **trig[7:0]**. The selection is applied to all eight signal channels.
In data condition 0, the condition for each channel is the current presence of a TDC hit signal.
In data condition 1, the condition for each channel is that the difference (δ) between the current FADC data value and the value two cycles ago is greater than the value of **thresh[1:0].**
This signal can be changed at any time without affecting any chip functions other than trigger data output.

Electrical Characteristics

Select trigger output data condition 1 (FADC related)	High signal level
Select trigger output data condition 0 (TDC related)	Low signal level

thresh[1:0] Pins 57 and 56 (Digital Inputs)

Description - Sets the threshold for trigger output data condtion 1 (see **mode_trig**). These signals can be changed at any time without affecting any chip functions other than trigger data output.

Electrical Characteristics
Normal CMOS inputs

CHIP CONTROL SIGNALS

sysclk Pin 53 (Digital Input)

Description - Receives a signal representing the 59.5 MHz System Clock. This is the primary timing signal for the acquisition and processing of input signals.

Electrical Characteristics - The chip expects a 50% (±10%) duty cycle signal, timed to the system time reference to within ±0.5 nsec.

clk15_out Pin 54 (Digital Output)

Description - Outputs a replica of the **Sample Clock** signal at one-fourth the **sysclk** frequency---i.e., 15 MHz. This clock is used internally for timing the digitizing operations,the reading and writing of DRAM and the writing of SRAM. This signal is provided as an output for the readout controller.

Electrical Characteristics
Normal CMOS slew-rate limited outputs

sync Pin 48 (Digital Input)

Description - Used to enable reset to zero the 10-bit counter that counts **sysclk** cycles. The simulataneous resetting of these counters in all chips results in a common time base for all chips in the local system. Resetting occurs on the 0->1 transition of the next **sysclk** cycle after **sync** is asserted.

Electrical Characteristics -

Set-up time (before next 0->1 transition of **sysclk**)	1 nsec
Hold time (after 0->1 transition of **sysclk**)	1 nsec

reset Pin 38 (Digital Input)

Description - Used to enable resetting of the Write and Read addresses of the DRAM and the Write address of the SRAM. Also resets the 10-bit counter that counts sysclk cycles. Resetting occurs on the 0->1 transition of the next **sysclk** cycle after **reset** is asserted.

Electrical Characteristics

Set-up time (before next 0->1 transition of **sysclk**)	1 nsec
Hold time (after 0->1 transition of **sysclk**)	1 nsec

test_mode Pin 55 (Digital Input)

Description - Used to select test or normal modes of operation; HIGH selects test mode; LOW selects normal mode. In test mode, each channel's MUX selects the test counter as the source of "event" data.

Electrical Characteristics

HIGH level must be maintained as long as test mode is desired.

TRIGGERING

tag[4:0] Pins 42 up to 46, respectively (Digital Inputs)

Description - Defines a sequence (event) number that is latched for later loading into SRAM2 upon receipt of an **L1_accept** signal.

Electrical Characteristics

Set-up time before leading edge of an **L1_accept**	1 nsec
Hold time	1 nsec

buff_wr[1:0] Pins 58, 59 (Digital Inputs)

Description - Selects which of four SRAM and SRAM2 write buffer spaces (0, 1, 2 or 3) is to be filled with event data after receipt of the next **L1_accept**. This number is also stored in the selected SRAM2 buffer space as bits 5 and 6 in the 1-byte tag field.

Electrical Characteristics

Set-up time before leading edge of **L1_accept**	1 nsec
Hold time	1 nsec

L1_accept Pin 47 (Digital Input)

Description - The 0->1 transition initiates filling the SRAM and SRAM2 buffer spaces selected by **buff_wr[1:0]**.

Electrical Characteristics

Set-up time (before next 0->1 transition of **sysclk**)	1 nsec
Hold time (after 0->1 transition of **sysclk**)	1 nsec
Minimum separation between pulses	2.20 μsec

mode_sparse Pin 41 (Digital Input)

Description - Determines the nature of the hitmap contents of the current write buffer of SRAM2. If **mode_sparse** = 0, the hitmap records the presence of TDC hits for each channel into the event data packet designated by **buff_wr[1:0]**. If **mode_sparse** = 1, the hitmap records the presence of FADC differences (δ's) for each channel in the data packet.

Electrical Characteristics

Set-up time before leading edge of next **trigger** when changing mode	1 nsec
Minimum Hold time	33 Sample Clock cycles (2.2 usec) after receipt of trigger (should normally be held constant unless mode is changed)

DATA OUTPUT SIGNALS

cs Pin 67 (Digital Input)

Description - Chip Select. Enables the tri-state output drivers that place output data signals on the **databus[7:0]** pins. When **cs** is not asserted, the output drivers are in the high-impedance state. **Cs** is normally held in the asserted state during a complete readout sequence.

Electrical Characteristics

Enable output drivers (asserted state)	High level
Set-up time (before next 0->1 transition of **clk15_out**)	1 nsec
Hold time (after 0->1 transition of **clk15_out**)	1 nsec

SRAMen Pin 68 (Digital Input)

Description - If asserted, selects SRAM as the source of data to be output during readout. If de-asserted, selects SRAM2 as the source of output data.

Electrical Characteristics

Set-up time (before next 0->1 transition of **clk15_out**)	1 nsec
Hold time (after 0->1 transition of **clk15_out**)	1 nsec

buff_rd[1:0] Pins 61, 60 (Digital Input)

Description - Selects which of the four data-buffer spaces (0, 1, 2 or 3) in SRAM and SRAM2 are enabled for readout.

Electrical Characteristics

Set-up time (before next 0->1 transition of **clk15_out**)	1 nsec
Hold time (after 0->1 transition of **clk15_out**)	1 nsec

ch_sel[2:0] Pins 62, 63, 64, respectively (Inputs)

Description - When **SRAMen** is de-asserted (SRAM2 selected), the two less significant bits of this signal set select one of three bytes from the currently specified SRAM2 read buffer. Depending on the value of these two bits, the source selected is:

VALUE	SOURCE
0	Hit map from SRAM2 selected by **buff_rd** set
1	Counter field from SRAM2 selected **buff_rd** set
2	Tag from SRAM2 selected by **buff_rd** set
3	Action not specified

When **SRAMen** is asserted, all three bits of this signal set designate a channel number. The data in SRAM from this channel number is then accessible for readout.

Electrical Characteristics

Set-up time (before next 0->1 transition of **clk15_out**)	1 nsec
Hold time (after 0->1 transition of **clk15_out**)	1 nsec

data_start Pin 69 (Digital Input)

Description - Zeros SRAM read address counter. This signal is issued before commencing the readout of a selected channel in a selected buffer of SRAM. After the first channel is read, its use on subsequent channels is optional if all channels are read out completely (all 32 words per channel).

Electrical Characteristics

Set-up time (before next 0->1 transition of **clk15_out**)	1 nsec
Hold time (after 0->1 transition of **clk15_out**)	1 nsec

data_inc_word Pin 70 (Digital Input)

Description - While asserted, this signal enables incrementing the read address counter at the 0->1 transition of Sample Clock. If held asserted, data output proceeds at Sample Clock frequency. This signal can be raised and lowered to accomplish readout at less than Sample Clock frequency.

Electrical Characteristics

Set-up time (before next 0->1 transition of **clk15_out**)	1 nsec
Hold time (after 0->1 transition of **clk15_out**)	1 nsec

databus[7:0] Pins 71 up to 78, respectively (Tri-State Outputs)

Description - Conveys byte data output from the chip via tri-state drivers.

Electrical Characteristics

High-impedance output state	when **cs** in low state
Normal CMOS slew-rate limited outputs	when **cs** in high state
Maximum load capacitance	50 pF

POWER AND GROUND

AGND Pins 33, 98

Description - Digital ground pins.

GND Pins 39, 49, 51, 65, 79, 82, 92

Description - Digital ground pins.

AVDD Pins 34 and 97

Description - Analog power — 4.5V ±0.5V

VDD Pins 40, 50, 52, 66, 80, 81, 91

Description - Digital power — 4.5V ±0.5V

N/C Pins 99 and 100