CDF/DOC/TRIGGER/CDFR/4812
Version 1.0
November 20, 1998

CrateSum Board Design Document

Upgrade for Run II

Nancy J. Lai and Dave Toback
University of Chicago

The main CrateSum document [1] discusses the functionality and general operation of the CrateSum board. This note describes in detail the innards of the board, including hardware components and the control and data signals passed amongst them. Familiarity with the content of the primary CrateSum document is assumed and frequent reference will be made to it. The DIRAC and PreFRED documents [2, 3, 4] may also be useful. All signal names and pin ordering for external board connections, if not included in this document, can be found in these references.

CrateSum has been designed using the Design Architect program within the commercially available Mentor Graphics board design environment. This note adopts several terms used in Mentor Graphics. "Module" designates components of the top-level schematic, "submodule" refers to the subunits of modules, and the term "symbol" is used to refer to the symbolic representations of discrete semiconductor chips within modules and submodules. "External connection buses" are those buses which link the modules in the top-level schematic diagram. "Internal buses" are buses which are used to link symbols or submodules within a module. All bus, symbol, module, and submodule names are written in boldface. A list of selected bus signal names can be found at the end of this document.

There are seven modules[1] in the top-level schematic: **VME_Interface, CONTROL**[2] **P3, P3_Latch, Sum, DAQ**, and **FP**. See Figure 1. **VME_Interface** is responsible for making sure that communications on the J1 and J2 backplane between VME bus masters and the rest of the CrateSum board are translated into signals that can be understood by the recipient of the communication. **CONTROL** decodes VME addressing, issues board control signals, interprets TSI (Trigger Supervisor Interface) messages, and controls board operations during diagnostic tests. **P3** reads in DIRAC energy and trigger information from the J3 backplane [5] via the P3 connector. This data is latched in **P3_Latch** and made available to **Sum** for processing. The summarized data is stored in FIFOs contained inside **DAQ**. Selected events (those accepted by the Level 1 trigger) are stored in register files ("Buffers"). The data summaries are also sent to the front panel (**FP**), where the data is transmitted to PreFRED[3].

[1] Eight, if the power section is counted. It is not involved in any aspect of dataflow on CS.

[2] The **CONTROL** module will be written with capitals to distinguish it from the **Control** bus.

[3] Strictly speaking, CrateSum's summaries are sent to the Aux-cards [5] in the Global Level 1 crate where PreFRED resides.

1 The VME_Interface Module

1.1 Purpose

This module provides an interface between CrateSum and the DAQ during downloading, readout, and diagnostics, and between CrateSum and the TSI during run-time. It reads in signals from the VRC and TRACER[4] via the P1 and P2 VME connectors and processes them. In particular, the **VME_Interface** module performs the following functions:

1. Connects the J1 and J2 VME backplanes to the CS-P1 and CS-P2 connectors, respectively. The J0 backplane, containing only connections to power, is not connected to CrateSum, since CrateSum does not require more power than is provided by the existing power pins on P1.

2. Routes the 5-bit Geographical Address onto the **GA** bus. The Geographical Address bits indicate the slot number of a board in the crate and are individually hardwired on the backplane either to ground or to power. These bits are used by **CONTROL** to determine if it is the intended target of a VME read or write request. They are also used to form part of the board header word.

3. Routes the VME Address bits onto the **VME_Address** bus, except for the last two bits, which are not relevant to board addressing and are ignored.

4. Routes the VME Data bits onto the **VME_Data** bus.

5. Routes the VME Control bits onto the **VME_Control** bus.

6. Routes the TSI Control bits onto the **Trig_Ctrl** bus.

See Figure 2 for the schematic.

1.2 Components of the VME_Interface

The brain of the **VME_Interface** is an Altera EPM7128S programmable logic device (PLD). Since this family of chips has nonvolatile memory, it does not need external EPROMs (erasable programmable read-only memories) for program downloading during board power-up.

The **VME_Interface** module includes a pair of 16-bit Texas Instruments SN74ABTE16245 data transceiver chips to allow data from the DAQ to be written to CrateSum's registers during downloading and to permit CrateSum to send data to the DAQ during data and register readouts and during diagnostics.

The **VME_Interface** does not respond to VME requests intended for another board in the crate. It honors only those requests accompanied by a VME address containing Geographical Address bits matching those of the board.

[4]The VRC (VME Readout Controller) is responsible for managing VME and DAQ interactions. The TRACER [6] passes along signals from the TSI.

Note that this module is almost exactly the same as that in DIRAC and PreFRED [2, 4]. Functionally, the **VME_Interface**s amongst these boards are identical. Only at the schematic level is there a small difference: CrateSum's **VME_Interface** design has incorporated a bug fix from PreFRED[5].

1.3 External Connection Buses

VME_Data(31:0), **VME_Address(26:2)**, **Trig_Ctrl(38:0)**, **VME_Control(7:0)**, **VME_config(9:0)**, **GA(4:0)**, **P0(94:0)**[6], **P1(159:0)**, and **P2(159:0)**.
Tables 1, 2, 3, 4, and 5 list the names and numbers of signals of selected buses.

2 The CONTROL Module

2.1 Purpose

This module is responsible for decoding VME addresses and for issuing signals to control the operation of the board. These control signals are organized, appropriately enough, on the **Control** bus. See Table 6 for a complete listing. This module also interprets TSI messages, manages diagnostics, and contains the board status register bits. To be specific, **CONTROL** performs the following tasks:

1. Latches TSI control signals on **Trig_Ctrl** every 132ns. The latched signals are sent to the Controller (see Figure 3), where they are interpreted.
2. Generates data processing control signals for use by the **P3_Latch**, **Sum**, and **DAQ** modules.
3. Provides VME address decoding and interfaces with all other VME interactions.
4. Controls the lights (LEDs) in the **FP** module. These signals indicate the status of, and current operations on, the board.
5. Controls a full set of diagnostic commands on the modules within CrateSum.
6. Detects full FIFO errors. The Controller takes the logical .OR. of the FIFO full flags located on the **Control** bus and stores the result inside the "board_error" register. If one or more FIFO full flags are high, it asserts the _cdf_error line, which is connected to the board _ERROR signal on the P2 bus.
7. Contains an 8-bit counter labeling the bunch number of data.

[5] This change in the design involves the Texas Instruments SN74BCT760 octal buffer/driver chip. The modifications that have been made can be directly seen by comparing the **VME_Interface** schematics of CrateSum and PreFRED.

[6] Note that there is no P0 connector.

2.2 Components of the CONTROL Module

2.2.1 Controller

The Controller is the brain, heart, and soul[7] of the **CONTROL** module and of the board itself. It regulates dataflow on the board by generating various signals on the **Control** bus to latch data at various stages of the data processing and storage. From the registers B0_offset_ET(5:0) and DIRAC_ET_delay(2:0), the Controller determines the value of $T_{\mathrm{P3\_Latch\_ET}}$, and from B0_offset_Trig(5:0) and DIRAC_Trig_delay(2:0), the value of $T_{\mathrm{P3\_Latch\_Trig}}$. The values $\Delta T_{\mathrm{proc\_latch\_ET}}$, $\Delta T_{\mathrm{proc\_latch\_Trig}}$, $\Delta T_{\mathrm{L1FIFO\_ET\_Sum}}$, $\Delta T_{\mathrm{L1FIFO\_Trig\_Sum}}$, $\Delta T_{\mathrm{L1FIFO\_ET\_Raw}}$, and $\Delta T_{\mathrm{L1FIFO\_Trig\_Raw}}$ have been pre-selected and encoded into the program of the PLD. Please refer to the functional CrateSum document for details. To initiate data processing of an event, the Controller generates strobes (**Control(80)** and **Control(81)**) at times $T_{\mathrm{P3\_Latch\_ET}}$ and $T_{\mathrm{P3\_Latch\_Trig}}$ to enable **P3_Latch** to latch in DIRAC energy and trigger data, respectively, from the P3 connector. The data is then available to **Sum** for summarizing. The processed results are latched inside **Sum** when **CONTROL** issues strobes (**Control(90)** and **Control(91)**) at $\Delta T_{\mathrm{proc\_latch\_ET}}$ and $\Delta T_{\mathrm{proc\_latch\_Trig}}$ after the associated P3 latch signal. **CONTROL** also generates strobes (**Control(5)** and **Control(6)**) to clock summarized data into FIFOs, starting at times $\Delta T_{\mathrm{L1FIFO\_ET\_Sum}}$ and $\Delta T_{\mathrm{L1FIFO\_Trig\_Sum}}$. Also, beginning at times $\Delta T_{\mathrm{L1FIFO\_ET\_Raw}}$ and $\Delta T_{\mathrm{L1FIFO\_Trig\_Raw}}$, the raw $\overset{\mathrm{DIRAC}}{\Sigma}\ \mathrm{E_T}$ and trigger bit words received by CrateSum on the backplane are clocked into FIFOs (**Control(83:82)** and **Control(87:84)**). (All "Δ"s are measured with respect to the associated P3 latch signal.) These strobes are sent every 132ns until the TSI issues a HALT command to discontinue writing to FIFOs and to cease data processing on the board.

Implemented inside the Controller is a "Bunch Counter". The number in this counter is latched into the appropriate L1FIFO every 132ns and labels the specific bunch that the ET and trigger data corresponds to.

The Controller also controls the LEDs on the front panel. The **Lights(7:0)** bus from the Controller is routed to the FP module, described below.

The Controller is an Altera EPF10K30 PLD. Since the FLEX 10K family of chips from Altera has volatile memory, program instructions must be stored in external EPROMs which automatically initiate program downloading at board power-up. The symbols for these EPROMs can be found on the FP schematic, Figure 6.

2.2.2 IDT74FCT574A Octal D Registers

TSI Control signals routed on the **Trig_Ctrl** bus are only guaranteed to have a 10ns setup and hold time with respect to the CDF_CLK rising edge. Please see Reference [6] for details. Thus the TSI Control signals must be latched for use on the board. The IDT74FCT574A chip captures the state of the _B0, _L1A, _L1R, _L2B0, _L2B1, _HALT, and _RECOVER signals every 132ns, on the rising edge of the CDF_CLK signal. These latched signals are then sent to the Controller.

[7]Adopting and extending the anatomical language used in the DIRAC board design document [3].

2.2.3 Timing Chips

There are two sets of 22ns tap delays used by the Controller. The first set is located near the middle of the schematic and slightly to the left. They are used to generate timing signals for VME transactions and data processing. The delay chip on top is for the CDF_CLK signal. The result is a total of 6 tapped signals, consecutively delayed by about 22ns. The delay chip below is for the inverted _vme_data_str signal coming from the **VME_Control** bus. Again, the result is 6 delayed lines. These are used as gates for performing various VME read and write transfers.

The second set of delay chips are located at the upper right-hand corner of the schematic. They delay the signals ET_clk0 and Trig_clk0, which are just the signals telling the ET or trigger P3 latches when to store new data. The consecutively 22ns-tapped delays are sent back to the Controller, which selects several of these lines as strobe signals for latching the summaries in the processor and for writing raw and summed data in the L1 FIFOs. The lines to be used have been selected and encoded in the program of the Controller chip.

2.2.4 Buffer Chips

Below the delay chips in the extreme upper right-hand corner of the schematic are two octal buffer chips. These chips make the L2 buffer write and read addresses and the L2 buffer write enable signals from the Controller chip stronger to enable them to drive a large number of L2 buffer chips. These signals from the Controller are also fanned out by the octal buffer chips. That is, outputs from these chips are not sent to all 60 of the L2 Buffer register file chips. The octal buffer chip on the left duplicates the read and write address lines from the Controller, so that there are two sets of such addresses, each set being sent to roughly 30 chips. The octal buffer chip on the right makes four copies of the L2 buffer write enable signal coming from the Controller, with each output line from the octal buffer going to approximately 15 L2 buffer chips.

The **Control** signals that are input to the octal buffer chips are **Control(56:52)** and those that are output are **Control(14:13, 51:50, 89:88, 126:125, 130:127)**. See Table 6 for further information.

2.2.5 Test Connectors and Testpoints

To visually probe the state of important board signals by using Logic Analyzers, two 140-pin connectors have been added. The first test connector, whose symbol can be found near the top left-hand corner of the schematic, allows signals from the **Trig_Ctrl**, **VME_Control**, **VME_Address**, and **VME_Data** buses, as well as the tap-delayed ET_clk0, Trig_clk0, CDF_CLK, and _vme_data_str lines, to be probed. The second test connector, located at the lower left side of the schematic, has connections to selected signals on the **Control** bus. The signals routed to the pins on these connectors are listed in Tables 7 and 8 at the back of this document.

A number of easily-accessible testpoints have also been added to aid in board debugging. Some of the lines with testpoint loops are connected to pins on the test connectors as well.

2.3 External Connection Buses

VME_Data(31:0), **VME_Address(26:2)**, **Trig_Ctrl(38:0)**, **VME_Control(7:0)**, **Control(130:0)**, **Lights(7:0)**, and **GA(4:0)**.

3 The P3 Module

3.1 Purpose

This module provides the interface between CrateSum and the custom VME J3 backplane. In particular, the **P3** module:

1. Connects the J3 backplane to the CS-P3 connector.
2. Reads in signals from DIRAC via the J3 backplane and CS-P3 and places them on the **ET_RAW** and **TRIG_RAW** buses.

Note that latching of the input energy and trigger information from DIRAC is not done in this module. Rather, it is performed in the **P3_Latch** module.

3.2 External Connection Buses

P3(263:0), **ET_RAW(63:0)**, and **TRIG_RAW(127:0)**.

The signals on the **ET_RAW** and **TRIG_RAW** buses are listed in Table 9.

4 The P3_Latch Module

4.1 Purpose

This module contains a 24-chip bank of 8-bit latches which are enabled by **Control(79)**. **ET_RAW(63:0)** and **TRIG_RAW(127:0)** data from DIRAC are latched in on the assertion of **Control(80)** and **Control(81)**, respectively.

4.2 External Connection Buses

ET_RAW(63:0), **TRIG_RAW(127:0)**, **ET_LATCHED(63:0)**, and **TRIG_LATCHED(127:0)**.

5 The Sum Module

5.1 Purpose

The Sum module consists of the Processor symbol, which computes and latches the $\overset{\text{Wedge}}{\Sigma}\ \mathrm{E_T}$ and trigger bit summaries. To be specific, the Processor does the following:

1. Produces a 10-bit $\sum^{\text{Wedge0}} E_T$ word for the 4 DIRAC boards making up "Wedge 0" and a 10-bit $\sum^{\text{Wedge1}} E_T$ word for "Wedge 1". The Processor also computes a 20-bit trigger summary consisting of four spare bits, three 2-bit numbers and ten 1-bit numbers, as described in the main CrateSum document.

2. Latches both wedge energy sums internally and onto **ET_SUM(19:0)** on **Control(90)** and the trigger bits summaries onto **TRIG_SUM(19:0)** on **Control(91)**.

3. Sends the energy summary for each wedge to the **FP** module in two sequential steps of 66ns each via **ET_SUM_MUX(9:0)**. The multiplexing is controlled by the Controller on **Control(108)**. The "parity" status bit controls which wedge summary is sent out first. The Processor also outputs the trigger summaries in two steps via **TRIG_SUM_MUX(9:0)**. This is controlled by **Control(109)**. The three 2-bit trigger result is sent in the first 66ns stage, followed by the ten 1-bit trigger bits.

The Processor is implemented by an Altera FLEX10K70 PLD. This chip has volatile memory and hence requires that program instructions be stored in external EPROMs which automatically download the program upon board power-up. The same EPROMs are used for both Processor and Controller programs. These chips are located on the **FP** schematic, Figure 6.

5.2 External Connection Buses

ET_LATCHED(63:0), **TRIG_LATCHED(127:0)**, **Control(130:0)**, **VME_Data(31:0)**, **ET_SUM(19:0)**, **TRIG_SUM(19:0)**, **ET_SUM_MUX(9:0)**, and **TRIG_SUM_MUX(9:0)**.

See Table 10 for a list of the signals on the **ET_SUM_MUX** and **TRIG_SUM_MUX** buses.

6 The DAQ Module

6.1 Purpose

The **DAQ** module (Figure 4) consists of five separate submodules: **Bunch Counter**, **ET_Sum_DAQ**, **Trig_Sum_DAQ**, **ET_RAW_DAQ**, and **TRIG_RAW_DAQ**. Each submodule contains banks of FIFOs and L2 buffer chips. The **Bunch_Counter** submodule appears as Figure 5. The other submodules contain larger numbers of FIFOs and buffer chips, but are otherwise very similar.

This module stores the raw DIRAC energy and trigger data as well as the summaries computed by the Processor. It may read out all or parts of this data into the datastream following a L2 accept. In detail, the **DAQ** module:

1. Latches **ET_SUM(19:0)** into the ET_Sum L1FIFOs on the assertion of **Control(5)**. It latches **TRIG_SUM(19:0)** into the Trig_Sum L1FIFOs on **Control(6)**. The bunch

counter value on **Control(78:71)** is latched into the Bunch_Counter L1FIFOs on **Control(9)**. Writing of the separate ET "wedges" and trigger bit "quarters" of raw DIRAC data into FIFOs is controlled by separate signals from the Controller. When the appropriate FIFO write signal in **Control(87:82)** is received, the **DAQ** module will load the indicated "wedge" or "quarter" of raw DIRAC data into the ET_Raw or Trig_Raw FIFOs.

2. Writes the L1FIFO data into the specified L2 buffer by using the write addresses specified on **Control(89:88)** and **Control(126:125)**.[8] Each of the L2 buffers is loaded on the assertion of one of the **Control(130:127)** signals.[9]

3. Replaces the data currently on the output registers of the FIFOs with data from the succeeding event. This action is taken on the **Control(11)** strobe.

4. Consequent to a L2A, reads out onto **VME_Data(31:0)** all or parts of the ET_Sum, Trig_Sum, ET_Raw, and Trig_Raw data, depending on the datastream readout mode. The particular buffer to be read out is indicated by the buffer read address lines **Control(14:13)** and **Control(51:50)**. The read enable signals are **Control(65:57)** and permit readout of the bunch counter value, ET_Sum, Trig_Sum, one ET "wedge", or one trigger bit "quarter" at one time.

6.2 External Connection Buses

ET_SUM(19:0), **TRIG_SUM(19:0)**, **ET_RAW(63:0)**, **TRIG_RAW(127:0)**, **Control(130:0)**, and **VME_Data(31:0)**.

7 The FP Module

7.1 Purpose

This module reads the 10-bit multiplexed half-crate energy and trigger summary signals from **ET_SUM_MUX(9:0)** and **TRIG_SUM_MUX(9:0)**, respectively, and sends them differentially to the Aux-Cards in the Global Level 1 Crate via front panel connectors. It also contains the front panel LED display and the EPROMs and connectors used for configuring the PLDs. In detail, this module does the following:

1. In two sequential 66ns steps, takes half of the energy summary result from **ET_SUM_MUX(9:0)**, uses differential drivers to convert it to differential signals, and outputs it via CS-FR1. See Reference [1] for further details.

[8] Recall that these are outputs of the line driver chip discussed in the **CONTROL** module section above and have a pair of buffer write address lines from the Processor as inputs. Hence these two sets of addresses are identical.

[9] Again, each signal in this set of signals are the same and come from the line driver chips in the **CONTROL** module.

2. In two sequential 66ns steps, takes half of the trigger summary result from **TRIG_SUM_MUX(9:0)**, uses differential drivers to convert it to differential signals, and outputs it via CS-FR2. The five double-bit trigger bits are sent in the first 66ns and the ten single-bit trigger bits in the subsequent 66ns. See Reference [1] for further details.

3. Takes in signals from the **Lights** bus to turn the front panel LEDs on or off. Table 11 lists the LEDs on the front panel.

4. Contains the symbols for the twin EPC1 chips containing the programming data for the two FLEX10K devices, the Controller and the Processor. The bus lines involved in configuration are **Control(124:119)**.

7.2 External Connection Buses

Lights(7:0), **ET_SUM_MUX(9:0)**, **TRIG_SUM_MUX(9:0)**, **CS_OUT(19:0)**[10], **VME_config(9:0)**, **FLEX_config(9:0)**, and **Control(130:0)**.

[10] This bus is not used at the current time.

8 External Buses (between top-level modules)

GA(4:0)

Signal Name	Description
GA(4:0)	Geographical Addresses

Table 1: These lines are used only by the **CONTROL** module, although not by the Controller.

VME_Address(26:2)

Signal Name	Description
VME_Address(26:2)	Internal Addresses

Table 2: These bus lines are sent by the **VME_Interface** module to the **CONTROL** module.

VME_Data(31:0)

Signal Name	Description
VME_Data(31:0)	Data

Table 3: This bus is used to carry up to 32 bits of data during VME write and read transactions.

VME_Control(7:0)

Signal Name	Description
VME_Control(0)	MODSEL*
VME_Control(1)	VME_DATA_STR*
VME_Control(2)	Not used
VME_Control(3)	Not used
VME_Control(4)	ACK*
VME_Control(5)	VME_ERROR*
VME_Control(6)	VME_WRITE
VME_Control(7)	SYSCLK

Table 4: These lines pass VME control signals from the **VME_Interface** to the **CONTROL**.

Trig_Ctrl(38:0)

Signal Name	Description
Trig_Ctrl(0)	CLK
Trig_Ctrl(1)	CLK*
Trig_Ctrl(3)	B0*
Trig_Ctrl(4)	L1A*
Trig_Ctrl(5)	L1R*
Trig_Ctrl(6)	L2B0*
Trig_Ctrl(7)	L2B1*
Trig_Ctrl(8)	HALT*
Trig_Ctrl(9)	RECOVER*
Trig_Ctrl(14)	CDF_ERROR*

Table 5: The **Trig_Ctrl** bus passes TSI control signals from the **VME_Interface** module to the **CONTROL** module. Only the signals used by CrateSum are listed above.

Control(130:0)

Control()	Name	Description
0	run	1 = run mode; 0 = load mode
1	src_ctrl_config	0 = EPROMs; 1 = FLEX Download Cable
2	_recover_	Recover and halt; output from controller
3	run_or_vme / _src_data	0 = VME/fake data; 1 = backplane DIRAC data
5	L1FIFO_Wclk_ET_Sum	ET_Sum FIFO write clock
6	L1FIFO_Wclk_Trig_Sum	Trig_Sum FIFO write clock
9	L1FIFO_Wclk_Bunch_Ctr	Bunch Counter FIFO write clock
11	L1FIFO_Rclk	FIFO read clock
14:13	L2BD_Set1_Driver(1:0)	Driver output: read address B (L2BD1) and A (L2BD0), Set 1
15	proc_version_ren	Read enable for processor version number
17:16	_EF_ET_Sum(1:0)	ET_Sum FIFOs(1:0) Empty Flag
19:18	_FF_ET_Sum(1:0)	ET_Sum FIFOs(1:0) Full Flag
21:20	_EF_Trig_Sum(1:0)	Trig_Sum FIFOs(1:0) Empty Flag
23:22	_FF_Trig_Sum(1:0)	Trig_Sum FIFOs(1:0) Full Flag
27:24	_EF_ET_Raw(3:0)	ET_Raw FIFOs(3:0) Empty Flag
31:28	_FF_ET_Raw(3:0)	ET_Raw FIFOs(3:0) Full Flag
39:32	_EF_Trig_Raw(7:0)	Trig_Raw FIFOs(7:0) Empty Flag
47:40	_FF_Trig_Raw(7:0)	Trig_Raw FIFOs(7:0) Full Flag
48	_EF_Bunch_Ctr	Bunch Counter Empty Flag
49	_FF_Bunch_Ctr	Bunch Counter Full Flag
51:50	L2BD_Set2_Driver(1:0)	Driver output: read address B (L2BD1) and A (L2BD0), Set 2
53:52	L2BD(1:0)	Altera output: read address B (L2BD1) and A (L2BD0)
55:54	L2B(1:0)	Altera output: write address B (L2B1) and A (L2B0)
56	_L2B_W	Altera output: buffer write enable
57	_L2BD_R0	Board ID and bunch counter value read enable
58	_L2BD_R1	ET_Sum read enable
59	_L2BD_R2	Trig_Sum read enable
60	_L2BD_R3	ET_Raw wedge 0 read enable (ET_Raw buffers 0-7)
61	_L2BD_R4	ET_Raw wedge 1 read enable (ET_Raw buffers 8-15)
62	_L2BD_R5	Trig_Raw(1:0) read enable (Trig_Raw buffers 0-7)
63	_L2BD_R6	Trig_Raw(3:2) read enable (Trig_Raw buffers 8-15)
64	_L2BD_R7	Trig_Raw(5:4) read enable (Trig_Raw buffers 16-23)
65	_L2BD_R8	Trig_Raw(7:6) read enable (Trig_Raw buffers 24-31)
70:66	serial_number(4:0)	Serial number indicated by the DIP switches
78:71	bunch_cnt(7:0)	Bunch number from bunch counter
79	_P3_Latch_OE	Output enable to latch raw DIRAC data in
80	P3_Latch_clk_ET	Clock to latch raw DIRAC ET data in
81	P3_Latch_clk_Trig	Clock to latch raw DIRAC Trig data in
82	L1FIFO_Wclk_ET_Raw_wedge0	Write ET_Raw wedge 0 clock
83	L1FIFO_Wclk_ET_Raw_wedge1	Write ET_Raw wedge 1 clock
84	L1FIFO_Wclk_Trig_Raw_qtr1	Write Trig_Raw quarter 1 clock
85	L1FIFO_Wclk_Trig_Raw_qtr2	Write Trig_Raw quarter 2 clock
86	L1FIFO_Wclk_Trig_Raw_qtr3	Write Trig_Raw quarter 3 clock
87	L1FIFO_Wclk_Trig_Raw_qtr4	Write Trig_Raw quarter 4 clock
89:88	L2B_Set1_Driver(1:0)	Driver output: write address B (L2B1) and A (L2B0), Set 1
90	proc_latch_clk_ET / et_run_clk	Latch summed ET data into processor
91	proc_latch_clk_Trig / trig_run_clk	Latch summed trigger data into processor
99:92	mask_DIRAC(7:0)	DIRAC board(7:0) input masking bits
100	one_or_two_bits0_0	Trig(7:0)(1:0) summing: 0 = 8*2-to-2; 1 = two 8-to-1
101	one_or_two_bits0_1	Trig(7:0)(3:2) summing
102	one_or_two_bits1_0	Trig(7:0)(5:4) summing
103	one_or_two_bits1_1	Trig(7:0)(7:6) summing
104	one_or_two_bits2_0	Trig(7:0)(9:8) summing
105	one_or_two_bits2_1	Trig(7:0)(11:10) summing
106	one_or_two_bits3_0	Trig(7:0)(13:12) summing
107	one_or_two_bits3_1	Trig(7:0)(15:14) summing
108	FPmux_ET	132ns signal used by controller for multiplexing ET data
109	FPmux_Trig	132ns signal used by controller for multiplexing Trig data
110	vme_clk_ET	Clock to latch VME ET data into the processor
111	vme_clk_Trig	Clock to latch VME trigger data into the processor
112	store_vme_wedge_sel	VME ET latch wedge selector
114:113	store_vme_qtr_sel(1:0)	VME Trig latch qtr selector
115	proc_out_write / write_to_vme	Enables processor write to VME data bus
118:116	proc_out_sel(2:0)	Select lines for VME data written by processor
124:119	config(5:0)	Lines used to configure the FLEX10K devices
126:125	L2B_Set2_Driver(1:0)	Driver output: write address B (L2B1) and A (L2B0), Set 2
130:127	_L2B_W_Driver(3:0)	Driver output: buffer write enables (3:0)

Table 6: Bus signals used to distribute commands throughout the CrateSum board. Any numbers not appearing in the list above are not used and are spare bits.

Test Connector 1 Pins

Connector Pin No.	test_conn1 Bus Line No.	Description
31:23, 20:5	24:0	VME_Address(26:2)
32	25	_B0
33	26	_L1A
34	27	_L1R
36:35	29:28	_L2B(1:0)
37	30	sysclk (VME_Control(7))
38	31	_HALT
41	32	_RECOVER
42	33	vme_write (VME_Control(6))
43	34	_modsel (VME_Control(0))
44	35	_ERROR
45	36	_ACK (VME_Control(4))
46	37	_vme_error (VME_Control(5))
52:47	43:38	cdf_132ns(5:0)
60:59, 56:53	49:44	vme_tap(5:0)
104:91, 88:73, 70:69	81:50	VME_Data(31:0)
106:105, 114: 109	89:82	ET_clk(8:1)
122:115	97:90	Trig_clk(8:1)

Table 7: Test Connector 1 signal names and numbers.

Test Connector 2 Pins

Connector Pin No.	**Control** Bus Signal No.	Signal Name
5	0	run
6	1	src_ctrl_config
7	2	_recover_
8	3	run_or_vme / _src_data
10	5	L1FIFO_Wclk_ET_Sum
11	6	L1FIFO_Wclk_Trig_Sum
14	9	L1FIFO_Wclk_Bunch_Ctr
16	11	L1FIFO_Rclk
17	15	proc_version_ren
19:18	17:16	_EF_ET_Sum(1:0)
23, 20	19:18	_FF_ET_Sum(1:0)
25:24	21:20	_EF_Trig_Sum(1:0)
27:26	23:22	_FF_Trig_Sum(1:0)
31:28	27:24	_EF_ET_Raw(3:0)
35:32	31:28	_FF_ET_Raw(3:0)
45:41, 38:36	39:32	_EF_Trig_Raw(7:0)
53:46	47:40	_FF_Trig_Raw(7:0
54	48	_EF_Bunch_Ctr
55	49	_FF_Bunch_Ctr
59, 56	51:50	L2BD_Set2_Driver(1:0)
69, 60	53:52	L2BD(1:0)
73, 70	55:54	L2B(1:0)
74	56	_L2B_W
83:75	65:57	_L2BD_R(7:0)
93:91, 88:84	78:71	bunch_cnt(7:0)
94	79	_P3_Latch_OE
95	80	P3_Latch_clk_ET
96	81	P3_Latch_clk_Trig
97	82	L1FIFO_Wclk_ET_Raw_wedge0
98	83	L1FIFO_Wclk_ET_Raw_wedge1
99	84	L1FIFO_Wclk_Trig_Raw_qtr1
100	85	L1FIFO_Wclk_Trig_Raw_qtr2
101	86	L1FIFO_Wclk_Trig_Raw_qtr3
102	87	L1FIFO_Wclk_Trig_Raw_qtr4
103	90	proc_latch_clk_ET / et_run_clk
104	91	proc_latch_clk_Trig / trig_run_clk
114:109, 106:105	99:92	mask_DIRAC(7:0)
122:115	107:100	one_or_two_bits(3:0)_(1:0)
123	108	FPmux_ET
124	109	FPmux_Trig
127	110	vme_clk_ET
128	111	vme_clk_Trig
129	112	store_vme_wedge_sel
131:130	114:113	store_vme_qtr_sel(1:0)
132	115	proc_out_write / write_to_vme
135:133	118:116	proc_out_sel(2:0)

Table 8: The map between Test Connector 2 pins and **Control** bus pin signals.

ET_RAW(63:0) AND TRIG_RAW(127:0)

Signal Name	Description
ET_RAW(0)	Least Significant Bit of E_T: DIRAC 0
ET_RAW(1)	Next-to-Least Significant Bit of E_T: DIRAC 0
⋮	⋮
ET_RAW(6)	Next-to-Most Significant Bit of E_T: DIRAC 0
ET_RAW(7)	Most Significant Bit of E_T: DIRAC 0
ET_RAW(8)	Least Significant Bit of E_T: DIRAC 1
ET_RAW(9)	Next-to-Least Significant Bit of E_T: DIRAC 1
⋮	⋮
ET_RAW(54)	Next-to-Most Significant Bit of E_T: DIRAC 6
ET_RAW(55)	Most Significant Bit of E_T: DIRAC 6
ET_RAW(56)	Least Significant Bit of E_T: DIRAC 7
ET_RAW(57)	Next-to-Least Significant Bit of E_T: DIRAC 7
⋮	⋮
ET_RAW(62)	Next-to-Most Significant Bit of E_T: DIRAC 7
ET_RAW(63)	Most Significant Bit of E_T: DIRAC 7
TRIG_RAW(0)	Trigger bit 0 - DIRAC 0
TRIG_RAW(1)	Trigger bit 1 - DIRAC 0
⋮	⋮
TRIG_RAW(14)	Trigger bit 14 - DIRAC 0
TRIG_RAW(15)	Trigger bit 15 - DIRAC 0
⋮	⋮
⋮	⋮
TRIG_RAW(112)	Trigger bit 0 - DIRAC 7
TRIG_RAW(113)	Trigger bit 1 - DIRAC 7
⋮	⋮
TRIG_RAW(126)	Trigger bit 14 - DIRAC 7
TRIG_RAW(127)	Trigger bit 15 - DIRAC 7

Table 9: The **ET_RAW** and **TRIG_RAW** buses are used to send DIRAC energy and trigger information from the **P3** module to the **P3_Latch** section. The order of bits is preserved at the outputs of **P3_Latch**, the **ET_LATCHED(63:0)** and **TRIG_LATCHED(127:0)** buses.

ET_SUM_MUX(9:0) AND TRIG_SUM_MUX(9:0)

Signal Name	Description
ET_SUM_MUX(9)	Most Significant Bit of $\overset{\text{Wedge}}{\Sigma}\mathrm{E_T}$
ET_SUM_MUX(8)	Next-to-Most Significant Bit of $\overset{\text{Wedge}}{\Sigma}\mathrm{E_T}$
⋮	⋮
ET_SUM_MUX(1)	Next-to-Least Significant Bit of $\overset{\text{Wedge}}{\Sigma}\mathrm{E_T}$
ET_SUM_MUX(0)	Least Significant Bit of $\overset{\text{Wedge}}{\Sigma}\mathrm{E_T}$
TRIG_SUM_MUX(9)	TRIG(19) (Step 1) / TRIG(9) (Step 2)
TRIG_SUM_MUX(8)	TRIG(18) (Step 1) / TRIG(8) (Step 2)
TRIG_SUM_MUX(7)	TRIG(17) (Step 1) / TRIG(7) (Step 2)
TRIG_SUM_MUX(6)	TRIG(16) (Step 1) / TRIG(6) (Step 2)
TRIG_SUM_MUX(5)	TRIG(15) (Step 1) / TRIG(5) (Step 2)
TRIG_SUM_MUX(4)	TRIG(14) (Step 1) / TRIG(4) (Step 2)
TRIG_SUM_MUX(3)	TRIG(13) (Step 1) / TRIG(3) (Step 2)
TRIG_SUM_MUX(2)	TRIG(12) (Step 1) / TRIG(2) (Step 2)
TRIG_SUM_MUX(1)	TRIG(11) (Step 1) / TRIG(1) (Step 2)
TRIG_SUM_MUX(0)	TRIG(10) (Step 1) / TRIG(0) (Step 2)

Table 10: These lines are used to transfer the energy and trigger bit summaries from the **Sum** module to the **FP** module. Note that $\overset{\text{Wedge}}{\Sigma}\mathrm{E_T}$ is for either $\overset{\text{Wedge0}}{\Sigma}\mathrm{E_T}$ or $\overset{\text{Wedge1}}{\Sigma}\mathrm{E_T}$, with the order of presentation of wedge data being determined by the state of the downloadable "parity" register in the Controller chip. The steps refer to the two 66ns sequential steps in which the energy and trigger summaries are sent out on the Front Panel. See the CrateSum main document [1] for the definitions of TRIG(x).

<u>LEDs on the Front Panel</u>

LED Name	Color
Power	Green
Run	Green
Clock	Yellow
VME Active	Green
L1 Decision	Yellow
Load	Yellow
Halt*	Red
Error	Red
EPROM Disabled	Red
Parity	Yellow

Table 11: The LEDs are listed in order of location on the front panel, from top to bottom. The parity LED indicates whether the board outputs the even-wedge or odd-wedge energy summary result first.

References

[1] N. J. Lai and D. Toback, *CrateSum Functional Document*, CDF Note 4811, Nov. 1998.

[2] J. Wahl, *DIRAC - The FINAL Specification*, CDF Note 4225, June 1997.

[3] J. Wahl, *The Anatomy of a DIRAC*, CDF Note 4231, June 1997.

[4] A. Amadon, M. Bogdan, and P. Wilson, *Specifications and Design of the Level 1 Calorimeter Trigger PreFRED Modules for the Run II Upgrade*, CDF Note 4340, Dec. 1997.

[5] P. Wilson, *J3 Backplane and Transition Module (L1AUX) for the Level 1 Calorimeter Trigger Upgrade*, Trigger Note #92, Dec. 1997.

[6] T. Shaw, *Specification for TRigger And Clock + Event Readout Module (TRACER)*, CDF Note (?), Aug. 1997.

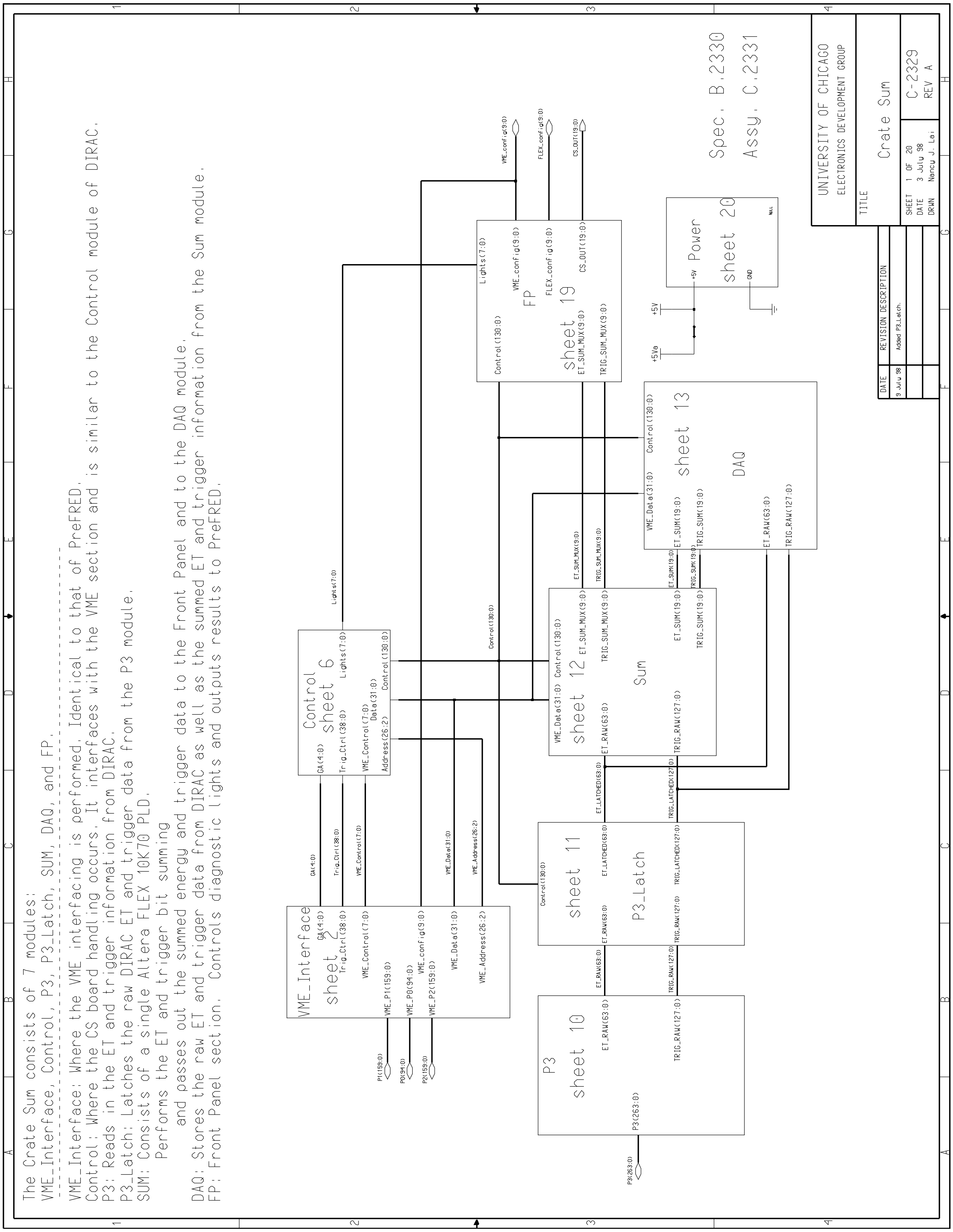

Figure 1: CrateSum Top-Level Diagram

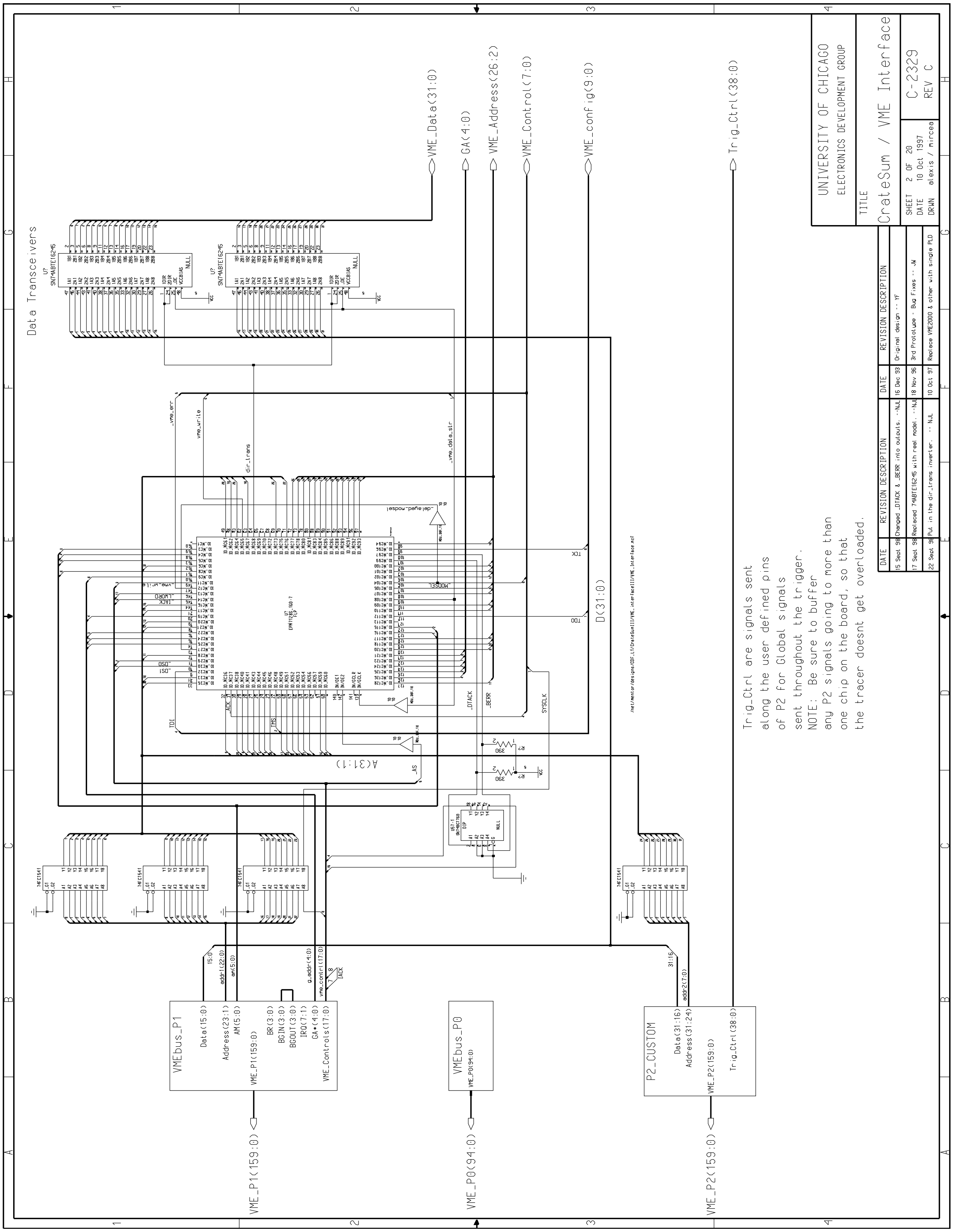

Figure 2: The **VME_Interface** Module

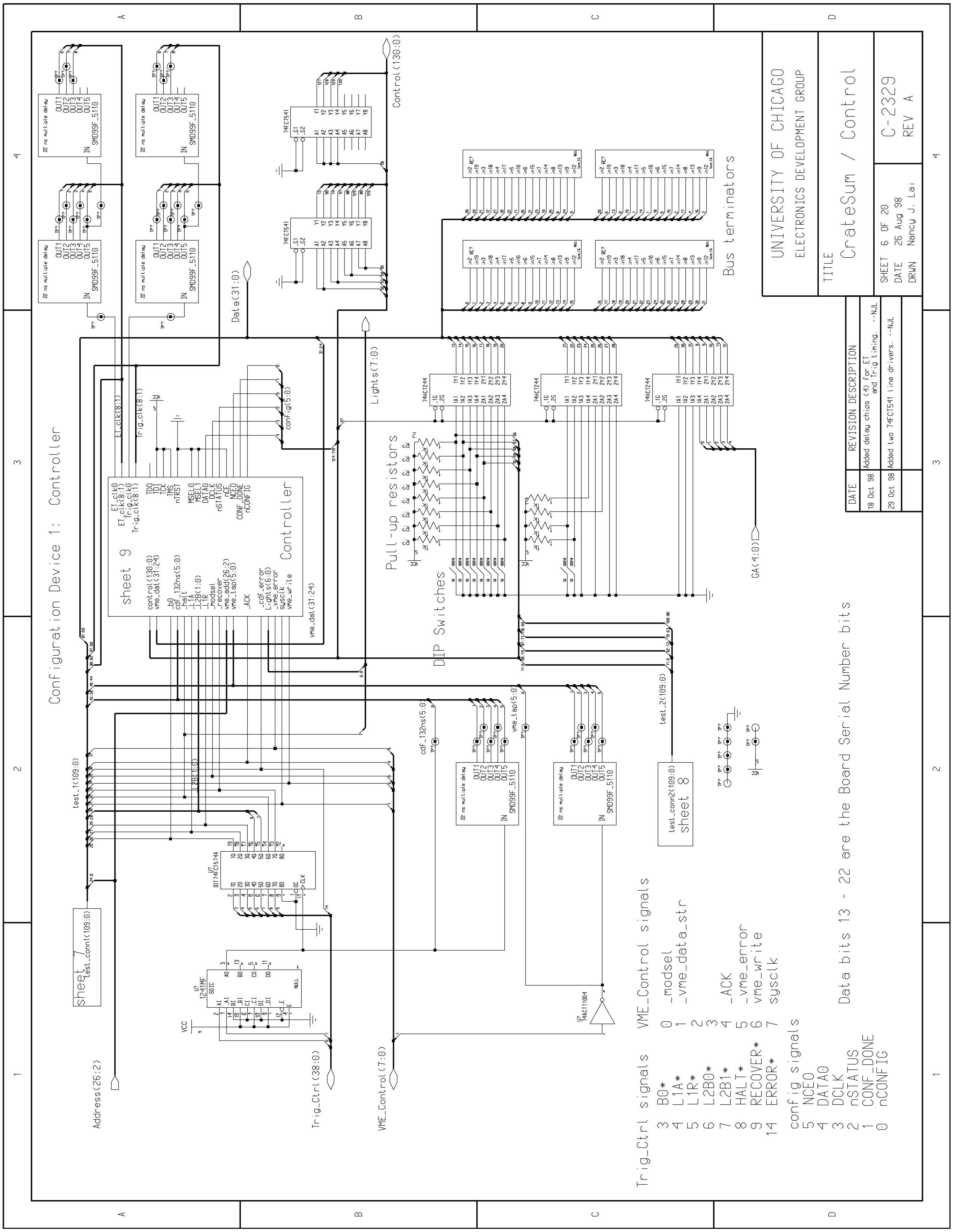

Figure 3: The **CONTROL** Module

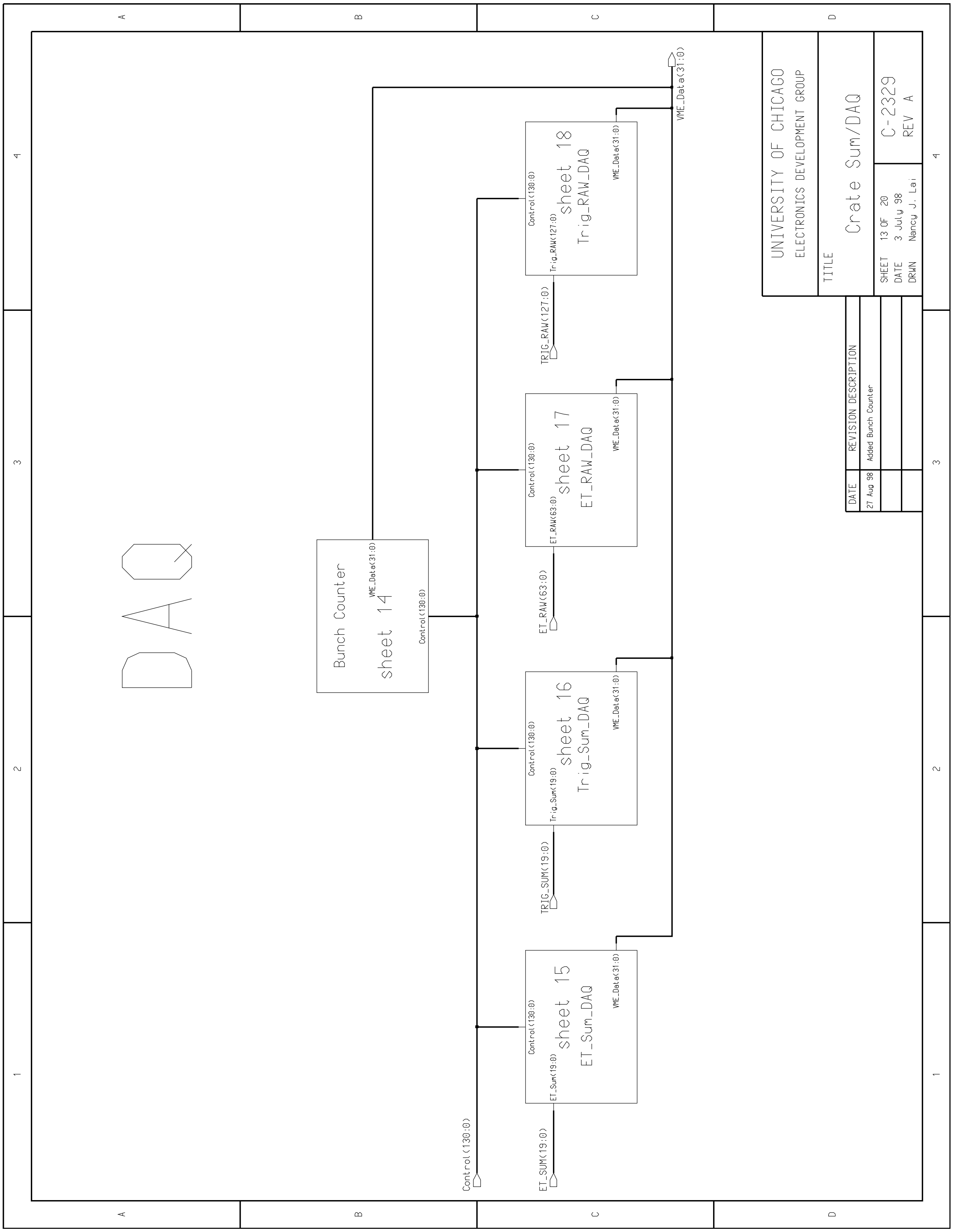

Figure 4: The **DAQ** Module

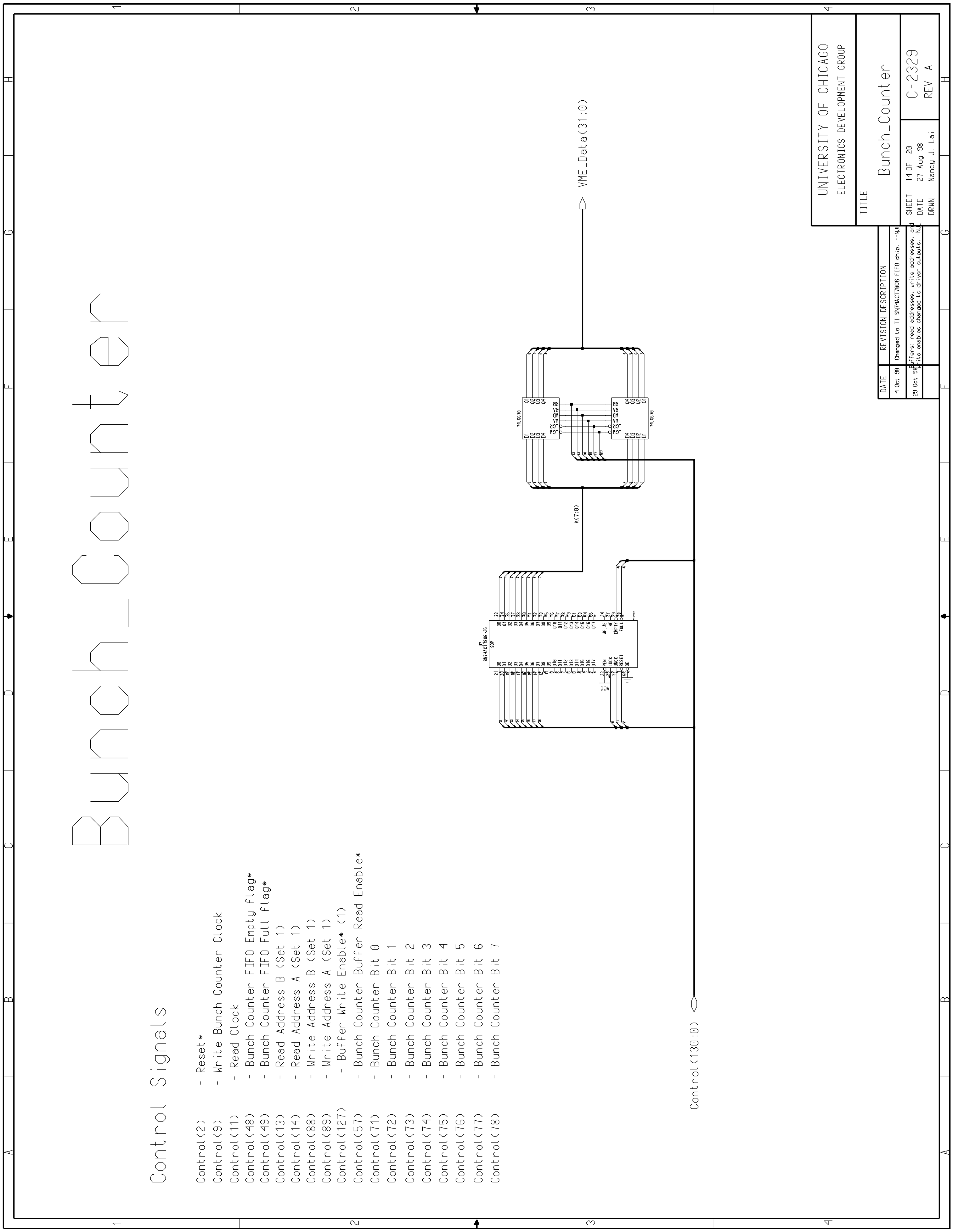

Figure 5: The **Bunch Counter** Submodule

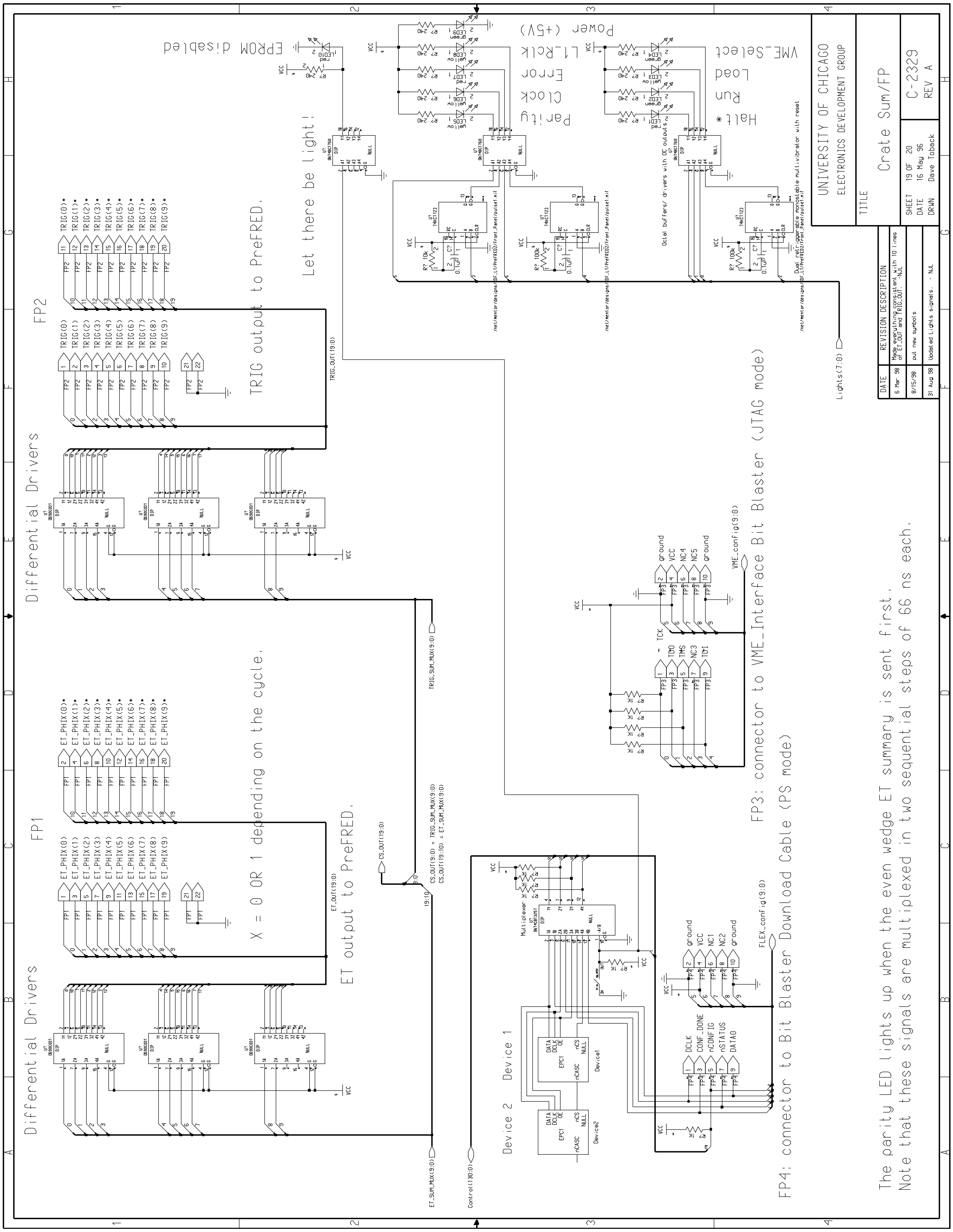

Figure 6: The **FP** Module