OPERATOR'S MANUAL

MODEL 2367

FASTCAMAC UNIVERSAL LOGIC MODULE

CE

Corporate Headquarters
700 Chestnut Ridge Road
Chestnut Ridge, NY 10977-6499
Tel: (914) 578-6013 Fax: (914) 578-5984
E-mail: lrs_sales@lecroy.com
lrs_support@lecroy.com

CE CONFORMITY

CONDITIONS FOR CE CONFORMITY

Since this product is a subassembly, it is the responsibility of the end user, acting as the system integrator, to ensure that the overall system is CE compliant. This product was demonstrated to meet CE conformity using a CE compliant crate housed in an EMI/RFI shielded enclosure. It is strongly recommended that the system integrator establish these same conditions.

CAUTION

IMPORTANT NOTICE

The 2367 ULM is shipped with both input and output level converters installed. No damage will occur if the unit is powered with the standard EPROM program, which does not use the front panel I/O. However, before any other user program is loaded, the unit must be configured with the desired I/O pattern. For each pair of sockets, for the 10124 and 10125, only one should be occupied. In total, 17 ICs should be removed.

TABLE OF CONTENTS

GENERAL INFORMATION

PURPOSE

This manual is intended to provide instruction regarding the setup and operation of the covered instruments. In addition, it describes the theory of operation and presents other information regarding its functioning and application.

UNPACKING AND INSPECTION

It is recommended that the shipment be thoroughly inspected immediately upon delivery. All material in the container should be checked against the enclosed Packing List and shortages reported promptly. If the shipment is damaged in any way, please notify the Customer Service Department or the local field service office. If the damage is due to mishandling during shipment, you may be requested to assist in contacting the carrier in filing a damage claim.

WARRANTY

LeCroy warrants its instrument products to operate within specifications under normal use and service for a period of one year from the date of shipment. Component products, replacement parts, and repairs are warranted for 90 days. This warranty extends only to the original purchaser. Software is thoroughly tested, but is supplied "as is" with no warranty of any kind covering detailed performance. Accessory products not manufactured by LeCroy are covered by the original equipment manufacturers' warranty only.

In exercising this warranty, LeCroy will repair or, at its option, replace any product returned to the Customer Service Department or an authorized service facility within the warranty period, provided that the warrantor's examination discloses that the product is defective due to workmanship or materials and has not been caused by misuse, neglect, accident or abnormal conditions or operations.

The purchaser is responsible for the transportation and insurance charges arising from the return of products to the servicing facility. LeCroy will return all in-warranty products with transportation prepaid.

This warranty is in lieu of all other warranties, express or implied, including but not limited to any implied warranty of merchantability, fitness, or adequacy for any particular purpose or use. LeCroy shall not be liable for any special, incidental, or consequential damages, whether in contract, or otherwise.

PRODUCT ASSISTANCE

Answers to questions concerning installation, calibration, and use of LeCroy equipment are available from the Customer Service Department, 700 Chestnut Ridge Road, Chestnut Ridge, New York, 10977-6499, (914) 578-6030.

MAINTENANCE AGREEMENTS

LeCroy offers a selection of customer support services. For example, Maintenance Agreements provide extended warranty that allows the customer to budget maintenance costs after the initial warranty has expired. Other services such as installation, training, on-site repair, and addition of engineering improvements are available through specific Supplemental Support Agreements. Please contact the Customer Service Department for more information.

DOCUMENTATION DISCREPANCIES

LeCroy is committed to providing state-of-the-art instrumentation and is continually refining and improving the performance of its products. While physical modifications can be implemented quite rapidly, the corrected documentation frequently requires more time to produce. Consequently, this manual may not agree in every detail with the accompanying product and the schematics in the Service Documentation. There may be small discrepancies in the values of components for the purposes of pulse shape, timing, offset, etc., and, occasionally, minor logic changes. Where any such inconsistencies exist, please be assured that the unit is correct and incorporates the most up-to-date circuitry.

SOFTWARE LICENSING AGREEMENT

Software products are licensed for a single machine. Under this license you may:

- Copy the software for backup or modification purposes in support of your use of the software on a single machine.
- Modify the software and/or merge it into another program for your use on a single machine.
- Transfer the software and the license to another party if the other party accepts the terms of this agreement and you relinquish all copies, whether in printed or machine readable form, including all modified or merged versions.

SERVICE PROCEDURE

Products requiring maintenance should be returned to the Customer Service Department or authorized service facility. If under warranty, LeCroy will repair or replace the product at no charge. The purchaser is only responsible for the transportation charges arising from return of the goods to the service facility. For all LeCroy products in need of repair after the warranty period, the customer must provide a Purchase Order Number before any inoperative equipment can be repaired or replaced. The customer will be billed for the parts and labor for the repair as well as for shipping. All products returned for repair should be identified by the model and serial numbers and include a description of the defect or failure, name and phone number of the user. In the case of products returned, a Return Authorization Number is required and may be obtained by contacting the Customer Service Department at (914) 578-6030.

PRODUCT DESCRIPTION

GENERAL

The CAMAC Model 2367 is a general purpose programmable logic module using state-of-the-art field programmable gate array technology. In addition to performing simple logic functions on the inputs, it is capable of elaborate arithmetic calculations such as histogramming, by virtue of a 3 megabyte memory and a TMS320F206S DSP processor. The module also contains 16 kBytes of non-volatile memory internal to the DSP chip (principally for DSP program storage) and an on-board 64 kBytes of non-volatile EEPROM memory for FPGA program retention. Virtually no digital application is out of its range.

The desired logic operations are programmed in a Xilinx 4013E gate array chip. Any logic that can be implemented as a synchronous (clocked) state machine may be programmed, subject only to the size limitation of the Xilinx gate array of approximately 13,000 equivalent gates. There are 3 clocks available on-board (40, 20, and 10 MHz), and any of 3 special front panel inputs may be used as a clock. Input and output signals use standard 10124 and 10125 TTL-ECL level translators. Input signals as short as 5 nanoseconds can be latched and synchronized with the internal state machine logic.

CONFIGURATION COMMANDS SUMMARY

A few basic CAMAC function codes are implemented in hardware, and are available on power up. These are used only to program the Xilinx chip, and all but F30 disappear after a CAMAC clear operation (F9, C, or Z).

F30 A0-A15: Enter general program mode (EEPROM configuration mode, bank 0 selected)

F29 A0-A15: Enter EEPROM programming mode (must enter general program mode first)

F28 A0-A15: Select CAMAC configuration mode.

F25 A0-A15: Program (configure) Xilinx chip (in the selected mode)

F21 A0-A15: Select Bank 1. (Either for EEPROM programming or Xilinx configuration)

F16 A0-A15: Write 8 bits (W1-W8) to the Xilinx.

F17 A0-A15: Write 8 bits (W1-W8) to the EEPROM, at the address specified by (W9-W23), in the specified bank, provided EEPROM programming mode has been entered.

F14 A0-A15: Test Xilinx INIT line.

F13 A0-A15: Test Xilinx program DONE.

F12* **A0-A15:** Test Xilinx RDY line (usually unnecessary).

F9 **A0-A15:** Disable all configuration function codes, except for F30 (CAMAC C, Z have the same effect).

*F12 can be used to test the Xilinx Ready line before each write operation. This is not required unless the CAMAC host is capable of CAMAC operations at a rate greater than 500 kHz.

SPECIFICATIONS

Please see the technical data sheet at the front of this manual for the list of specifications.

SOFTWARE SUPPORT

Please see the LRS customer support page on the LeCroy website for Xilinx .bit files and DSP programs referenced in this manual. The web page is http://www.lecroy.com/lrs/support.htm.

XILINX FPGA CONFIGURATION OPTIONS

It should be noted that the EEPROM memory, with a capacity of 64K bytes, has enough room for two distinct 4013E configurations, given that each configuration program occupies 30989 bytes. These two configurations are referred to as "Bank 0" and "Bank 1". Bank 1 is also referred to as the "Alternate Configuration".

Xilinx FPGA Configuration Options include:

1. Bank 0 (power-up default) configuration
2. Bank 1 (Alternate) configuration
3. User-defined configuration loaded via CAMAC

Power-up Configuration

At power-up, the gate array automatically configures itself out of “bank 0” of the EEPROM memory. At the factory, this EEPROM bank 0 is loaded with a firmware called T2367M (file T2367M.BIT on the LeCroy website). This firmware makes the 2367 act as a CAMAC dataway tester and simple 3 MByte memory CAMAC module, described in Appendix B.

Reloading Power-up (Bank 0) Program

If at any time the power-up configuration needs to be reloaded, it can be done by entering the programming mode (F30), and sending a program pulse (F25). The program mode should then be exited with F9. The progress of the configuration process (before issuing the exit) can be monitored by probing the DONE bit (Q response to a F13 cycle).

Reload Bank 0 Sequence:

F30	Enter program mode
F25	Commence bank 0 program reload (takes several ms)
F13*	Probe DONE bit (Q response)
F9	Exit program mode

** indicates repeated command*

LOADING ALTERNATE (BANK 1) PROGRAM

The EEPROM has enough capacity to hold an alternate configuration in bank 1. To activate the bank 1 configuration, one must enter the programming mode (F30), select the alternate bank (F21), and send a program pulse (F25). One should then exit the programming mode with an F9. The completion of the configuration out of bank 1 may be tested, if wanted, by probing the DONE bit with a F13 command before issuing the exit. At the factory, bank 1 was loaded with a configuration file that makes the 2367 act as a LeCroy Model 4302 FERA Memory. See Appendix B for the details.

Alternate (Bank 1) Program Load Sequence:

F30	Enter program mode
F21	Select Alternate program
F25	Commence Alternate program load (takes several ms)
F13*	Probe DONE bit (CAMAC Q =True)
F9	Exit program mode

**indicates repeated command*

Configuration Via CAMAC

The configuration information can also be transmitted directly into the Xilinx FPGA via CAMAC. For this, one must enter the programming mode (F30), and select the CAMAC mode (F28). After sending an F25, wait until the INIT signal goes high (polling for Q response with F14). The configuration data is then entered, byte by byte, in ascending address order by a succession of F16 commands. When loading data from a Xilinx .bit file (such as the file lrs4302.bit), begin this upload with the first “FF” data byte in the file. (See sample code in Appendix A.)

When the configuration is done, the DONE bit (tested via a Q response on F13) becomes high and program mode should be exited (F9). After each F16 command, the RDY bit should, in principle, be tested with an F12 to know if the unit is ready for a next byte. However, the CAMAC interface is generally slow enough to ensure that RDY is high by the next cycle without having to test for it.

Loading Xilinx Configuration via CAMAC Sequence
(see sample code in Appendix A)

Initializations:

F30	Enter program mode
F28	Select CAMAC programming mode
F25	Start Xilinx Configuration (takes a few ms)
F14*	Repeat until CAMAC Q=True

Program Upload:

F16* (D8)	Upload one byte of data for programming. (using CAMAC W1-W8)
F13	Probe DONE bit (CAMAC Q =True)
F9	Exit program mode

* *indicates repeated commands*

Changing EEPROM Programs

Both the power-up and the alternate configurations can be altered from their initial settings. To do this, one must first enter the general program mode (F30); then the EEPROM programming mode is entered (F29). To avoid potential conflict on the data bus, it is recommended to clear the Xilinx configuration with F28 followed by F25. After sending an F25, wait until the INIT signal goes high (polling for Q response with F14). The alternate EEPROM memory may then be selected via F21, if desired. (See sample code in Appendix A.)

The EEPROM can then be written (in any order) by writing the concatenated data and address information with F17 cycles (W1-W8:data; W9-W23: address). When loading data from a Xilinx .bit file (such as the file lrs4302.bit), locate the first “FF” data byte in the file. This byte should be uploaded with address “0”; continue loading bytes from the file while incrementing the address by one. A 5 ms delay between F17 commands must be maintained; failure to do so may result in damage to the EEPROM.

On the next power-up, the Xilinx FPGA will be configured with new program, unless the Alternate program was chosen. If you do not want to recycle the power, execute the steps shown above for reloading the power-up configuration. After the EEPROM has been altered, if you wish to return to the factory-set power-up or alternate configuration, the EEPROM contents have to be overwritten again. Download the file from the LeCroy website to obtain the original EEPROM configuration (http://www.lecroy.com/lrs/support.htm).

Changing EEPROM Programs
(see sample code in Appendix A)

Initializations:

F30	Enter program mode
F29	Select EEPROM programming mode
F28	Select CAMAC mode to prepare for Xilinx
F25	De-configure Xilinx
F14*	Repeat until CAMAC Q=True
F21	(Optional) Selects alternate program

EEPROM Programming:

F17* D24	Upload 1 Byte data at 15-Bit Address
.	Data Byte: CAMAC W1-W8
.	Address Bytes: CAMAC W9-W23

**indicates repeated command.*

(It is imperative to respect a 5 ms delay between successive F17 operations.)

PROGRAMMABLE XILINX FPGA PINS

The 192 Xilinx programmable pins divide in the following categories:

1. CAMAC bus signals
2. FASTCAMAC buffering signals
3. RAM control signals
4. Digital Signal Processor (DSP) related signals
5. Timing and Clocks
6. LEDs
7. Front Panel
8. Unconnected pins

CAMAC Bus Signals

They include F, A, N, S1, S2, Z, C, I, X, Q, L.

F Input [4:0] Pins 234, 235, 236, 237, and 4 for bits 4, 3, 2, 1, 0 respectively. Active high. Receives the function code from the CAMAC bus.

A Input [3:0] Pins 5, 8, 9, 10 for bits 3, 2, 1, 0 respectively. Active high. Receives the sub-address code from the CAMAC bus.

N Input. Pin 11. Active high. Receives the “N” signal indicating that the slot is selected.

S1 Input. Pin 23. Active high. Receives the “S1” strobe signal after an adjustable delay.

S2 Input. Pin 24. Active high. Receives the “S2” strobe signal.

Z Input. Pin 6. Active high. Receives the initialize “Z” signal.

C Input. Pin 7. Active high. Receives the clear “C” signal.

I Input. Pin 17. Active high (indicates inhibited). Receives the inhibit signal.

X Output. Pin 25. Active low. Transmits a “X” response.

Q Output. Pin 26. Active low. Transmits a “Q” response .

L Output. Pin 27. Active low. Issues a “Look at me” request.

Read/ Write Lines Both Input and Output (pin assignments in Table 1 below). Active high. DATABUS[48:1] is split in two 24-bit words. The upper 24 bits (DATABUS [48:25]) are connected to the buffer that brings the CAMAC W lines to the module. The numbering of the W lines arriving on DATABUS is a simple offset by 24. The lower 24 bits are connected to the buffer that drives the CAMAC R lines, with a straightforward numbering.

Table 1: PINOUTS OF CAMAC READ/WRITE LINES

DATABUS [ ] index	Buffered to line	Pin number	DATABUS [ ] index	Buffered to line	Pin number
1	R1	123	25	W1	79
2	R2	129	26	W2	81
3	R3	141	27	W3	82
4	R4	148	28	W4	84
5	R5	152	29	W5	85
6	R6	159	30	W6	86
7	R7	173	31	W7	87
8	R8	177	32	W8	88
9	R9	183	33	W9	92
10	R10	184	34	W10	93
11	R11	187	35	W11	94
12	R12	188	36	W12	95
13	R13	202	37	W13	96
14	R14	203	38	W14	97
15	R15	209	39	W15	99
16	R16	210	40	W16	100
17	R17	213	41	W17	102
18	R18	214	42	W18	103
19	R19	220	43	W19	104
20	R20	221	44	W20	105
21	R21	232	45	W21	107
22	R22	233	46	W22	108
23	R23	238	47	W23	109
24	R24	3	48	W24	110

FASTCAMAC Buffering

Some signals needed for FASTCAMAC operation must be defined even for normal operation. Below we identify these signals and show how they should be defined for normal CAMAC operation. The assignments shown below are needed for normal CAMAC operation.

IN Output. Pin 12. Active high. Indicates the current operation is a CAMAC write of data, and the buffers should be enabled in the “inwards” direction.

For normal CAMAC operation, one should define:
IN := N & F[4] & !F[3].

EN_W Output. Pin 13. Active high. Indicates that the buffer connected to the CAMAC W lines is to drive signals (active low).

For normal CAMAC operation:
EN_W := !(N & F[4] & !F[3])

EN_R Output on Pin 15. Active low. Indicates that the buffer connected to the CAMAC R lines is to drive signals.

For normal CAMAC operation:
EN_R := !(N & !F[4])

SWAP Output. Pin 142. Active high. Indirectly causes three octal buffer chips to copy the data present on Data[47:24] onto Data[23:0]. This typically might be needed during a CAMAC write cycle. Anyway, it is harmless in such a case.

Thus, for normal operation:
SWAP := N & F[4] & !F[3]

SIXTEEN Output, Pin 18, should be kept low.

NORMAL Output, Pin 16, should be kept high

BOTH Output, Pin 20, should be kept low.

FCLOCK Input, Pin 21. Can be ignored for normal CAMAC operation. It indicates a new data is to be read under FASTCAMAC block transfer. Beware that under certain compilers, an ignored input might get used as an outside node for an intermediate calculation. This danger is avoided if the input to be ignored is part of a “dummy” logic function.

The above assignments get modified for level I and level II FASTCAMAC operation.

Under FastCAMAC level I, both EN_R and EN_W can be active simultaneously, making effectively the CAMAC data bus 48-bits wide, and there is otherwise no change with respect to normal CAMAC.

Under FASTCAMAC level II, NORMAL is set low, and BOTH (active high) controls whether read data is clocked out of the module at only the leading edges of S1 or both edges. SIXTEEN then indicates whether the data are 16-bit wide. This is not used in the present implementation of FASTCAMAC.

RAM Control Signals

The 3 MByte RAM consists of six chips. Each chip is connected to eight different bits of the 48-bit wide databus, and shares a common address bus. Two of the six chips are accessible either to the user Xilinx program, or to the DSP program, under user Xilinx program control. Here, for simplicity, we will consider that the two RAM chips are under user program control.

memAdr[18:0] Outputs, active high. Constitute the address at which the RAM will be addressed. The pins are assigned as per the table below. Obviously, since this is a RAM, the address bits are interchangeable. It is thus possible to assign them differently, except of course when the DSP is involved and continuous information has to be exchanged.

Table 2: memAdr[18:0] pinouts

Bit	Pin Number
0	34
1	35
2	36
3	38
4	39
5	41
6	42
7	43
8	44
9	46
10	47
11	48
12	49
13	50
14	51
15	52
16	53
17	54
18	55

memWE[5:0] Outputs, active low. Each line indicates that writing of data is enabled for the respective RAM chip. See Table 3 for pinouts, chip numbers, and databus mapping.

Actual writing to the memory does not occur unless the memory is enabled.

memSTRB[5:0] Outputs, active low. Each line strobes a particular RAM chip to be read or written. See Table 3 for pinouts, chip numbers, and databus mapping.

memRE Output, active low. Pin 56. Indicates that a read operation is to be done on the RAM chips that are strobed by memSTRB.

Table 3: Pinouts for memWE[5:0], memSTRB[5:0]

Index	Chip	WE pin	STRB pin	DATABUS bits
0	U4	70	71	1-8
1	U3	67	69	9-16
2	U6	65	66	17-24
3	U1	72	73	25-32
4	U2	78	77	33-40
5	U5	76	74	41-48

DSP Pins

These pins permit control of the DSP and detection of certain DSP conditions. (Please refer to the DSP data sheet for definitions and further instructions.)

DSPIO[0, 1, 2, 3] Input / Output. Pins 28, 31, 32, 33. These pins are also configurable as either input or output on the DSP side, and represent a general purpose real-time interface capability between the FPGA and the DSP.

DSP_TRST Output. Pin 216. Active high. Connected to the RST pin of the JTAG interface of the DSP. To avoid inadvertent JTAG programming of the DSP, it is imperative to program this pin to be low. It must be brought high for any JTAG programming to take place.

DSP_TCLK_OFF Output. Pin 217 (dual function). It is connected to the CLK pin of the JTAG interface of the DSP. It is also connected to the active-low OFF signal of the DSP. This means that this pin is brought high in order to activate the DSP (and give it control of two of the six memory chips) and must be brought low to regain access to these memory chips after the DSP has ran.

DSP_TMS Output. Pin 218. Active high. Connected to the Mode Select pin of the JTAG interface of the DSP. If the JTAG port is not used, leave it low.

DSP_TDI	Output. Pin 223. Active high. Connected to the Data In pin of the JTAG interface of the DSP. It can safely be left low.
DSP_TDO	Input. Pin 224. Active high. Connected to the Data Out pin of the JTAG interface of the DSP. The state of this pin can be ignored unless doing JTAG programming.
DSP_HOLD	Output. Pin 225. Active low. Connected to the “HOLD” Interrupt pin of the DSP. Should normally be left high. The HOLD interrupt can be used to temporarily hold the DSP off the bus.
DSP_NMI	Output. Pin 226. Active low. Connected to the non-maskable interrupt pin of the DSP. Should normally be left high. The NMI steers the DSP program execution almost unconditionally.
DSP_BOOT	Output. Pin 228. Connected to a pin of the DSP which selects whether execution at reset will proceed using the internal program memory or the external program memory. External program memory must be used until the internal non-volatile program memory is programmed. Low (0) means internal, high (1) means external.
DSP_RS	Output. Pin 229. Active high, is connected (via an inverter) to the active-low reset pin of the DSP; should normally be left low, at least while DSP_TCLK_OFF is high. The reset condition steers the DSP program execution unconditionally. It also clears most of the DSP registers.
DSP_BIO_ON	Output. Pin 230. (dual purpose). It is connected to the active-low BIO pin of the DSP, and permits real-time steering of the DSP program execution, if the program execution proceeds within internal ROM and RAM. Its other function is an active high signal that lets the DSP control the upper address bits of the memory. While it is high, the DSP I/O space is mapped at addresses 0x50000 through 0x5FFFF, the DSP data space is mapped at addresses 0x60000 to 0x6FFFF, and the DSP program space is mapped at addresses 0x70000 to 0x7FFFF. While it is low, the DSP has no control on the memory signals.
DSP_HOLDA	Input. Pin 231, active low. Connected to the active-low Hold Acknowledge pin of the DSP. It can be ignored if the HOLD mechanism is not used.

Timing and Clocks

Four pins control the timing of S1 detection and 3 pins receive clocks.

DELAY[7, 6, 5, 4] Outputs. Pins 174, 239, 153 and 181. Control the delay after which S1 will be “seen and validated”. S1 pulses shorter than this delay value will be ignored. The actual delay is produced by a specialized chip externally. The control of DELAY[4], which is output via the TDO pin, requires the use of the special symbol “TDO”. Unless absolutely needed for a very good precision in the value of the generated delay, it is safe to abstain from programming this pin, since TDO will generally remain a constant value. Any value is acceptable for normal CAMAC operation.

CLK40 Input. Pin 63. Receives a 40 MHz clock, which may be used as the global clock for the design.

CLK20 Input. Pin 124. Receives a 20 MHz clock, which may be used as the global clock for the design. If used to define a phase, it is useful to know that CLK20 changes near the rising edge of CLK40 (a 74F74 circuit generates it).

CLK10 Input. Pin 2. Receives a 10 MHz clock, which may be used as the global clock for the design. If used to define a phase, it is useful to know that CLK10 changes near the rising edge of CLK20 (a 74F74 circuit generates it).

LEDs

During configuration, two LEDs have well defined status. After configuration completes, their statuses are programmable. A “1” means lit.

HDC Output. Pin 64, active high. Controls the upper, green LED (CR3) on the front panel.

LDC Output. Pin 68 active high. Controls the lower, yellow LED (CR5) on the front panel.

Front Panel

The front panel has 59 differential ECL signal input/output

The Input /Output Xilinx lines P1[7:0], P2[15:0], P3[15:0], P4[15:0] simply control/receive to their respective dECL output or input on the front panel. The pin assignment is shown in Table 4.

Table 4: Front Panel ← → Xilinx pinouts

Bit #	Connector ch.	P1 (Connector A)	P2 (Connector B)	P3 (Connector C)	P4 (Connector D)
0	1	193	192	136	133
1	2	194	191	137	134
2	3	197	190	138	130
3	4	198	189	144	131
4	5	199	186	139	132
5	6	200	185	145	126
6	7	205	176	147	128
7	8	206	175	146	117
8	9		172	156	127
9	10		171	149	114
10	11		170	157	116
11	12		169	154	112
12	13		168	155	115
13	14		167	160	125
14	15		165	162	113
15	16		164	163	111
16	17		178*	118*	57*

** These pins differ from the other front panel I/O pins; see the next section.*

P2_16_CLK, P3_16_CLK, P4_16_CLK Input / Output. Pins 178, 118, 57 respectively. These pins map to the 17th pair on connectors B, C, and D. These lines simply receive or control their respective dECL input or output on the front panel. They distinguish from the other front-panel I/O pins in that they are connected to low-skew global “secondary clock” lines inside the FPGA.

Unconnected Pins

Pins 207, 208, and 215 do not have any board connection. They may be used for dummy logic functions, or to hold / transmit intermediate logic result. Also note that the INIT* pin, which is used during configuration, is programmable after configuration, but it is not connected then to any functionality

FRONT PANEL SIGNALS

Polarity

The input/output polarity is such that a positive-going ECL edge on an odd numbered pin on the input connector produces a positive-going TTL edge at the Xilinx input. (Note that this is opposite to the mapping in the LeCroy model 2366, in which a positive edge at the front panel was converted to a negative edge at the Xilinx input pin.)

Leading Edge Input Latching

For best performance, the FPGA should be configured such that the input signals should be used as the clock input of a D Flip Flop with a logic 1 as the D input. Thus, the input level change will be latched.

Configurability as Inputs or Outputs

All front panel I/O is configurable as input or output in groups of 4, as indicated below. The 3 clocks input/outputs are separately configurable). This is accomplished by installing the socketed ECL-TTL or TTL-ECL level converters and the appropriate termination resistor networks. All inputs and outputs are differential ECL. The corresponding Xilinx pins must be programmed as either inputs or outputs.

For front panel I/O lines programmed as inputs: Install the 10125 ECL-to-TTL converters and the 56 ohm termination resistor SIPs The inputs are properly terminated for twisted pair cable.

For front panel I/O lines programmed as outputs: Install the 10124 TTL-to-ECL converters and the 390 ohm pull-down resistor SIPs.

Note: The module will work properly with short cables (but with reduced noise immunity) even if both the input termination SIPs and the output pull-down SIPs are installed.

Naming and Grouping of Front Panel I/O Lines

The logical polarity of any signal is programmable within the Xilinx chip. There are a total of 59 I/O signals. The following list is a description of the Front panel I/O lines, and their grouping. The connectors on the front panel are labeled a though D; on the schematic, they are labeled P1 through P4.

1 16 pin (8 pairs) header:

I/O A 1-4
I/O A 5-8

3 34 pin (17 pairs) headers:

I/O B 1-4
I/O B 5-8
I/O B 9-12
I/O B 13-16
I/O B 17 (can be connected to SGCK4)

I/O C 1-4
I/O C 5-8
I/O C 9-12
I/O C 13-16
I/O C 17 (can be connected to SGCK3)

I/O D 1-4
I/O D 5-8
I/O D 9-12
I/O D 13-16
I/O D 17 (can be connected to SGCK2)

SGCK2, 3 and 4 are Xilinx internal clock distribution networks.

Configuring I/O for TTL Operation

It is possible to convert the front panel I/O to bidirectional TTL. This user modification cannot be wholeheartedly recommended, however, since little protection is provided for the Xilinx chip. This method will only work at low speeds (less than 1 MHz) and for very short cable lengths. Deglitching will be needed at all receivers. The Xilinx pins cannot drive 100 Ω terminated lines. For best results we recommend using the terminated ECL for the cable runs, and converting to TTL at the destination circuit board. This is necessary if high speed performance is required.

To provide TTL connections, construct this 16 pin dip header with 4 100 Ω resistors, for each group of 4 to be used as TTL. Install this header in the 10124 location after removing both resistor packs, and both the 10124 and 10125 chips.

This user modification connects the Xilinx pin directly to the odd numbered pin on the front panel connector. The corresponding even numbered pin is grounded. There is no buffer, ONLY a series resistor to prevent damage to the Xilinx chip.

The Xilinx chip can be programmed for either INPUT, OUTPUT or as a TRISTATE pin.

Please Use With Care! See Table 5 for modification instructions.

Table 5: Conversion to TTL Inputs

Pin	Modification
1	connect to 16
2	connect to 16
3	100 Ω resistor to 7
4	100 Ω resistor to 5
5	100 Ω resistor to 4
6	
7	100 Ω resistor to 3
8	
9	
10	100 Ω resistor to 12
11	100 Ω resistor to 13
12	100 Ω resistor to 10
13	100 Ω resistor to 11
14	connect to 16
15	connect to 16
16	connect to 1,2,14,15 (ground)

Note: Only 4 100 Ω resistors are required.

APPENDIX A

APPLICATION NOTES

A-1. How to Program the LeCroy Model 2367 Universal Logic Module Xilinx 4013E

There are basically four steps to programming the Xilinx 4013E FPGA, and integral part of the LeCroy Model 2367 Universal Logic Module. These steps are:

1. The creation of a schematic/design.
2. Its conversion to a XILINX formatted netlist.
3. The generation of a binary “bit” file.
4. The actual configuration of the FPGA with the desired configuration.

• Example of the first step:

You may make a copy of and edit the examples provided on the LeCroy website to prepare a new VERILOG source. As an alternative, a tool such as the XILINX foundation software would allow you to enter a schematic in graphical “What You See Is What You Get” format.

• Example of the second step:

Enter the schematic in VERILOG, and compile it into the xnf format, using a design compiler.

dc_shell_exec ...xxxx.script

where xxxx.script contains the pin assignments and refers to the xxxx.v VERILOG source program.

The name of the command that translates a distinct net list format to the XILINX net list format may take the form “...2xnf”.

In all cases, a “sxnf” (the initials stand for XILINX net format) file should be generated.

• Example of the third step:

The following command file (with the *.MAK extension) is used on the PC to convert the *.XNF file into a binary file with the extension *.BIT. This example uses the XILINX software version 5. Note that the lines beginning with a “#” are comment lines inserted to explain the contents of the *.MAK file.

```
############ Start of MAK file ######################
# xxxx.mak PC MAKE file for XILINX builds

# Environment inherited macros, usually directories.
DESIGN_DIR=c:\xact\designs\xxxx\

# Miscellaneous files...*.XNF is input, others are outputs
SCHEM_FILE = $(DESIGN_DIR)xxxx.xnf
MERGED_FILE = $(DESIGN_DIR)xxxx.xff
PREP_FILE = $(DESIGN_DIR)xxxx.xtf
PPR_PARAFILE = $(DESIGN_DIR)xxxx.ppr
```

```
# Compile and command macros
cmd_merge = xnfmerge $(SCHEM_FILE) $(MERGED_FILE)
cmd_prep = xnfprep $(MERGED_FILE)
cmd_ppr = ppr $(PREP_FILE) paramfile = $(PPR_PARAMFILE)
cmd_makebits = makebits xxxx.lca

# Specify ultimate target of MAK.
all: $(DESIGN_DIR)xxxx.bit

# Build the merged design.
$(MERGED_FILE):              $(SCHEM_FILE)
                                    $(cmd_merge)

# Build final whatever.
$(DESIGN_DIR)test3377.bit:    $(MERGED_FILE)
                                    $(cmd_prep)
                                    $(cmd_ppr)
                                    $(cmd_makebits)

############ End of MAK file ######################
```

• Example of the fourth step:

Below is a C program to download the bit file (you may consult the LRS website for similar BASIC program used in the 2366). It is assumed here that there exists a FAN function performing the CAMAC cycle specified by the F, A, and N arguments that follow, a FANW function that writes the specified word, and a Q function returning the Q status for the crate specified (somehow) by c.

```
// minimal program to configure a 2367 via CAMAC using a bitfile.
int main(int argc, char *argv[])
{
    FILE *fp=fopen("design.bit","rb");
    CAMAC_CONTROLLER c;
    int slot;
    unsigned char ch;
    printf("enter slot number :");
    scanf("%d",&slot);
    FAN(c,30,0,slot);                   //enter general programming mode
    FAN(c,28,0,slot);                   //select CAMAC configuration mode
    FAN(c,25,0,slot);                   // re-configure
    do FAN(c,14,0,slot); while (!Q(c)); // wait for INIT to come high
    do ch=getc(fp); while (ch != 0xff); // advance to first FF in file
    do {
    FANW(c,16,0,slot,ch);               // write configuration byte
    FAN(c,13,0,slot);                   // done => Q
    ch=getc(fp);                        // get next one
    if (feof(fp)) return 1;             // error if end of file reached before done
    } while (!Q(c);)                    // until Q, i.e., until done
                                        //
    FAN(c,9,0,slot);
    fclose(fp);
    return 0;
}
```

A program line:

```
do FAN(c,12,0,slot); while (!Q(c));
```

just before writing the configuration bytes would cause the program to wait for the RDY signal before the next write of a configuration byte, but this is not normally needed.

To write the configuration file to the non-volatile EEPROM, essentially the same program could be used, but writing with F17 instead of F16:

```
// minimal program to overwrite non-volatile configuration of a 2367 via CAMAC
using a bitfile.
int main(int argc, char *argv[])
{
    FILE *fp=fopen("design.bit","rb");
    CAMAC_CONTROLLER c;
    int slot;
    unsigned char ch;
    printf("enter slot number :");
    scanf("%d",&slot);
    FAN(c,30,0,slot);                    //enter general programming mode
    FAN(c,29,0,slot);                    // select EEPROM programming mode
    FAN(c,28,0,slot);                    // select CAMAC configuration mode
    FAN(c,25,0,slot);                    // to de-configure XILINX
    do FAN(c,14,0,slot); while (!Q(c));  // when INIT comes high, XILINX cleared

    do ch=getc(fp); while (ch != 0xff);  // advance to first FF in file
    addr=0;                              // start at address 0
    do {
      FANW(c,17,0,slot,(addr<<8)+ch);    // write configuration byte
      Sleep(5);                          // wait 5 ms (to allow EEPROM processing)
      addr++;                            // go to next address
      ch=getc(fp);                       // get next configuration byte
    } while (!feof(fp));                 // until end of file
                                         //

    FAN(c,9,0,slot);
    fclose(fp);
    return 0;
}
```

After this program is written to non-volatile memory, one would cycle the power, and the newly written configuration would be active after power-up. Alternatively, one would issue F30, and F25 to reload the program without cycling the power. If a F21 is inserted after F29, EEPROM bank 1 will be overwritten instead of the power-up default bank 0. To activate the bank 1 configuration, one must issue: F30, F21, and then F25.

APPENDIX B

DEFAULT EEPROM LOADS FOR XILINX CONFIGURATION AND DSP PROGRAMMING EXAMPLE

INTRODUCTION

The following section includes some description of the program loaded in the 2367 EEPROM at the factory. By discussing the VERILOG source, it also discusses possible approaches to tap into the 2367's capabilities, particularly for what regards FASTCAMAC, memory, and DSP usage. For more examples, one can refer to the 2366 users' manual and application notes.

Further examples of configuration of the 2367 are available for downloading from the LeCroy ftp and web sites (http://www.lecroy.com/lrs/support.htm).

B-1. T2367M: CAMAC Dataway & Memory Tester

DESCRIPTION

This configuration file makes the 2367 behave as a CAMAC Dataway and Memory Tester. The memory structure is 1M addresses x 24 bit data words.

The DSP execution on a program loaded in RAM can be started and stopped. **This configuration is available on power-up**, as long as the EEPROM is not overwritten with a distinct program. The DSP can be used, within this power-up configuration, to sum a large array of 16 bit values in memory quickly. An example program to do that is shown below. Refer to the TMS320F206S documentation to learn more about the available DSP op-codes. See the Verilog listing below that shows how these functions are realized.

F Code (Function)	A Code (Subaddress)	
	0	15
0	read 24 bit register	read 24 bits RAM memory at address and increment address
1	read status of A, F, N, C, Z, I at last S2 (*)	read 20 bit RAM memory address
9	stop DSP	
16	write 24 bit register	write 24 bits RAM memory at address and increment address
17		write 20 bit RAM memory address
25	start DSP	

(*) See table below for decoding of output.

Decoding of F(1) A(0) Output

R13	R12	R11	R10	R9	R8	R7	R6	R5	R4	R3	R2	R1
N	C	Z	I	F16	F8	F4	F2	F1	A8	A4	A2	A1

DSP Programming Example

```
// example of a DSP program loading and execution
// The DSP will sum up the values present at locations 0xE1800 through 0xEFFFE of the RAM
and put
// the result in 0xEFFFF.

int main(int argc, char *argv[])
{
   CAMAC_CONTROLLER c;
   int slot;
   unsigned char ch;
   printf("enter slot number :");
   scanf("%d",&slot);
   FANW(c,17,15,slot,0xF0000);                    //go to start of program area

   // The following lines enter DSP program op-codes in the DSP program
memory area

   FANW(c,16,15,slot,0xbf8000L);                  //                        ACC=
   FANW(c,16,15,slot,0x000001L);                  //                           0
   FANW(c,16,15,slot,0xbf0902L);                  //                          A1=
   FANW(c,16,15,slot,0xe7fe03L);                  //                       0xe7fe
   FANW(c,16,15,slot,0xbf0b04L);                  //                          A3=
   FANW(c,16,15,slot,0x180005L);                  //                       0x1800
   FANW(c,16,15,slot,0x8b8b06L);                  //          next aux. Reg. is A3
   FANW(c,16,15,slot,0x20a907L);                  // ACC = ACC + @(A3++), next aux. Reg. A1
   FANW(c,16,15,slot,0x7b9b08L);                  // BANZ (A1)- , next aux. Reg. is A3
   FANW(c,16,15,slot,0x000709L);                  //
   FANW(c,16,15,slot,0xbf090aL);                  // DSP program  A1=
   FANW(c,16,15,slot,0xffff0bL);                  //                  0xffff
   FANW(c,16,15,slot,0x8b890cL);                  //          next aux. Reg. is A1
   FANW(c,16,15,slot,0x90a90dL);                  // store ACCLÆ @(A1)++, next aux. Reg. A1
   FANW(c,16,15,slot,0x79890eL);                  //                  loop
   FANW(c,16,15,slot,0x000e0fL);                  //                  forever

   FAN(c,25,0,slot);                              //start DSP
   Sleep(1000);                                   //let DSP run
   FAN(c,9,0,slot);                               //stop DSP
   FANW(c,17,15,slot,0xEFFFF);                    //go to the data area
   printf(" Sum =%d\n",FAN(c,0,15,slot)>>8);      //retrieve and print sum
   return 0;
}
```

B-2. Default Alternate (Bank 1) Program: FERA Memory

DESCRIPTION

This configuration file makes the 2367 behave as a LeCroy Model 4302 FERA memory, with 512k x 16 bit memory. This is a 32-fold larger memory than the 4302 itself.

The exact similarity between a configured 2367 and a 4302 has not been fully tested at this point.

INSTALLATION

P2 through P4 MUST be configured for input, and P1 MUST be configured for output, by removing the appropriate 10125 and 10124 chips BEFORE this configuration is used.

This Xilinx configuration is present on bank 1 of the EEPROM, and is activated by F30, F21, F25, F9. Do not activate this configuration before removing the appropriate chips.

OPERATING INSTRUCTIONS

When configured as a “4302”, the 2367 will store input data that is coincident with a strobe signal (WSI). An Acknowledge signal (ACK) is returned, if the data is accepted with a delay of 35 nsec.

Overflow Operation

The Overflow condition is fixed at 7/8 of the memory instead of being adjustable via jumpers. When the memory fills to this point, the OVF output becomes TRUE.

FULL Operation

The FULL output becomes TRUE when the memory is completely full.

ECL / CAMAC Modes

The module can be enabled in either ECL or CAMAC mode. When in ECL mode, the module can store data. In CAMAC mode, the ECL inputs are disabled, and the memory may be readout. The TOP LED shows the current mode (ON=ECL, OFF =CAMAC).

Front Panel I/O Assignments

When this configuration of the FPGA is active and the inputs and outputs are properly configured, the module has the following dECL inputs and outputs:

4302 Signal Name	Signal Function	Emulated by 2367 Signal
WSI	Input	B (17)
VETO	Input	C (17)
N(1-16)	Input	D (1-16)
OVF	Output	A (1)
ACK	Output	A (2)
FULL	Output	A (3)

LED Assignments

TOP LED:
- ON: Module is enabled for ECL inputs
- OFF: Module is enabled for CAMAC readout

BOTTOM LED:
- OFF: Memory pointer = 0 (empty)
- DIM: Memory pointer > 0, < 7/8 full
- ON: Memory pointer ≥ 7/8 Full (Overflow condition)
- FLASH: Memory pointer = 512k (Full condition)

CAMAC Commands

F Code (Function)	A Code (Subaddress)	
	0	1
0	Read memory at pointer and increase pointer	
1	Read pointer	Read modes (data=1, camac; data=3, ECL)
2	Read memory at pointer and decrease pointer. If first F2, read pointer.	
4	Read program ID (returns decimal 4302)	Reads program version (returns decimal 11 as of this printing)
8	Test LAM (Q if LAM)	
9	Clear	
10	Test and clear LAM	
16	Write memory at pointer and increase pointer	
17	Write address pointer	Write input mode (data=1, camac; data=3, ECL)
24	disable LAM	
26	enable LAM	