TEST AND VERIFICATION PLAN

EVLA Correlator Chip Prototype Test Plan

TVP Document: **A25082N0005**

Revision: DRAFT

Brent Carlson, April 18, 2006

National Research Council Canada
Herzberg Institute of Astrophysics
Dominion Radio Astrophysical Observatory

P.O. Box 248, 717 White Lake Rd
Penticton, B.C., Canada
V2A 6J9

Table of Contents

1 REVISION HISTORY 4

2 INTRODUCTION 5

3 VERIFICATION MATRIX 6

4 VERIFICATION TEST CASES 17

5 REFERENCES 41

6 APPENDIX I: TEST VECTOR STRUCTURE 42

7 APPENDIX II: TRANSMIT FPGA REGISTER SET DESCRIPTION 47

7.1 MASTER CONTROL/STATUS REGISTER (MCSR) R/W ADDR=0X00 RESET=0X7F 49
7.2 PERIOD REGISTER (PEREG) R/W ADDR=0X01 RESET=0X10 49
7.3 DUMPTRIG ACCUMULATE CONTROL REGISTER (DTACCR) (R/W) ADDR=0X02 RESET=0X00 50
7.4 CRC ERROR GENERATION REGISTER (CRCERRGEN) R/W ADDR=0X03 RESET=0X0000 50
7.5 SHIFT ENABLE CLOCK MODULO REGISTER (SECMR) R/W ADDR=0X07 RESET=0X00 51
7.6 DATA STREAM IDS REGISTERS 0-7 (DSIR-0…DSIR-7) R/W ADDR=0X10…0X17 52
7.7 PHASEMOD COEFFICIENT REGISTERS 0-7 (PMCR-0…PMCR-7) R/W ADDR=0X20…0X3F 53
7.8 PHASERR COEFFICIENT REGISTER (PECR) R/W ADDR=0X40 RESET=0X0000 54

8 APPENDIX III: TRANSMIT FPGA GUI PANELS 55

8.1 TIMING AND DUMP CONTROL 55
8.2 DATA STREAM CONTROL 57

9 APPENDIX IV: CORRELATOR BACKEND (CBE) REQUIREMENTS 59

List of Abbreviations and Acronyms

ASIC – Application Specific Integrated Circuit. This is a chip built to perform a specific set of functions. This is also referred to as a "custom chip" or "custom ASIC".

EVLA – Expanded Very Large Array. This is the project that the EVLA correlator, including the EVLA correlator chip is being built for.

FPGA – Field Programmable Gate Array. A programmable chip that contains an array of logic functions surrounded by programmable interconnects.

FSM – Finite State Machine. Logic that implements some controller function that contains a finite number of states, responds to input stimuli, and produces output signals.

PAR – Refers to Place and Route of the correlator chip.

PPAR – Refers to Post Place and Route of the correlator chip.

RTL – Register Transfer Language. This is synthesizable Verilog (VHDL is another industry standard that is not used for the correlator chip) code used to build the chip's logic functions. RTL is a subset of the entire HDL (Hardware Description Language)

Structured ASIC – This is an ASIC that contains a sea of gates or cells and a fixed number of specialized functions such as PLLs and memory. The Structured ASIC is frozen to perform fixed functions once metal interconnect layers are added in the design and manufacturing process.

VLA – Very Large Array. Array of 27, 25 m antennas in New Mexico built in the 1970s and operating since ~1980.

VLSI – Very Large Scale Integration.

1 Revision History

Revision	Date	Changes/Notes	Author
DRAFT	April 18, 2006	Initial DRAFT release for review.	B. Carlson

2 Introduction

The EVLA Correlator Chip is a standard-cell ASIC (Application Specific Integrated Circuit), fabricated in 0.13 μm CMOS technology and containing approximately 4 million gates. Extensive testing of the functionality [1] [2] of the design and of the gate-level netlist [3] has been performed. The next step in the process is to verify that the physical prototypes function as designed with the desired performance and power dissipation.

Testing of the prototype Correlator Chips is a crucially important step to ensure the success of the EVLA Correlator project. While the impact of a mistake in testing up to this point means some additional cost in re-spinning the prototypes, a mistake in testing of the prototypes for approval to proceed with production can have expensive and disastrous consequences. Thus, this test plan is a superset of any functional or gate-level testing that has been performed to date. Its aim is to exhaustively test the Correlator Chip prototypes in all conceivable operating modes, making no assumptions or conclusions about what should work given that something similar has been tested. It is, of course, impossible to achieve 100% test coverage, although this test plan aims to do so. Unfortunately, there is no external way of quantitatively measuring test coverage and it is up to critical review of this test plan, and of subsequent test results to determine if sufficient test coverage has been achieved.

This plan contains two components. The first component is the Verification Matrix and exhaustively indicates all items and functionality in the chip that are to be tested, indicating for each one which of the Verification Test Cases addresses or tests that particular item. The second component is the table of Verification Test Cases, which describes a number of tests that are to be performed. These Test Cases are referenced in the Verification Matrix.

In some cases test coverage is most easily described in the Verification Matrix, and in other cases test coverage is most easily described as a Verification Test Case. Testing is complete when all items in both the Verification Matrix and the Verification Test Cases have been addressed.

The appendices of this document contain valuable background information on the structure of test vectors and software used for testing the chip. Interpretation of the results of each test, and comparison with the reference behavioural software correlator requires a thorough understanding of test vector structure and it is here for this purpose.

3 Verification Matrix

The verification matrix, including coverage up to test case 16, is shown in the following table. For the complete matrix, including coverage for all test cases, refer to C25082N0006. Test cases are defined in the table of section 4.

Test Case Number	**0**	**1**	**2**	**3**	**3a**	**4**	**5**	**5a**	**5b**	**6**	**6a**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
Test Coverage Description																					
NOTE: All tests run with 256 MHz clock, unless otherwise noted.																					
I/O connectivity, timing, voltage, test vector mode																					
X input test vector detection/sync with no errors generated or detected. TVEN=1		x																			
Y input test vectors detection/sync with no errors generated or detected. TVEN=1		x																			
X input test vectors with errors generated and detected on each X input line. TVEN=1			x																		
Y input test vectors with errors generated and detected on each Y input line. TVEN=1			x																		
X daisy-chain test vectors. Complete 8x8 array of chips active. No errors. TVEN=1				x																	
Y daisy-chain test vectors. Complete 8x8 array of chips active. No errors. TVEN=1				x																	
X daisy-chain test vectors. Complete 8x8 array of chips active. Errors. TVEN=1						x															
Y daisy-chain test vectors. Complete 8x8 array of chips active. Errors. TVEN=1						x															
X PLL output jitter, 1 chip, <150 ps pk-pk cycle-cycle, record stats.	x																				
Y PLL output jitter, 1 chip, <150 ps pk-pk cycle-cycle, record stats.	x																				
X PLL output jitter 8X daisy chain <150 ps pk-pk cycle-cycle, record stats.																					
Y PLL output jitter 8X daisy chain <150 ps pk-pk cycle-cycle, record stats.																					
Test Case Number	**0**	**1**	**2**	**3**	**3a**	**4**	**5**	**5a**	**5b**	**6**	**6a**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
The following measurements can be done in any one of several test cases. The test cases indicated are those which may most likely be used to make the measurements.																					
XCLOCK_out to X data out, test vectors, tsu, record (>0.7 nsec)				x																	
XCLOCK_out to X data out, test vectors, thold, record (>0.5 nsec)				x																	

YCLOCK_out to Y data out, test vectors, tsu, record (>0.7 nsec)				x																
YCLOCK_out to Y data out, test vectors, thold, record (>0.5 nsec)				x																
X outputs: record voltage levels (must be within 2.5 V I/O standard)				x																
Y outputs: record voltage levels (must be within 2.5 V I/O standard)				x																
XCLOCK_O duty cycle: 47% < duty_cycle < 53%				x																
YCLOCK_O duty cycle: 47% < duty_cycle < 53%				x																
LCI clock to LCI data tco, record (max=1.5 nsec; min=-0.2 nsec)				x																
LCI inputs from LTA FPGA, tsu, record				x																
LCI inputs from LTA FPGA, thold, record				x																
LCI_DATA_CLKOUT, jitter < 150 ps, all chips daisy-chained, record.				x																
LCI interface signals: record voltage levels (must be within 2.5 V I/O standard)				x																
MCB clock to MCB_DATA, tco, record				x																
MCB clock tsu, record				x																
MCB clock thold, record				x																
MCB DATA out: record voltage levels (must be within 2.5 V I/O standard)				x																
LCI corr chip-to-LTA data interface, normal data transfer cycle							x													
LCI corr chip-to-LTA data interface, frame abort cycle.																				
schid_frame sync with X-path > Y-path by ~10 m, test corr ok																				
schid_frame sync with Y-path > X-path by ~10 m, test corr ok																				
JTAG test ok. Conforms to IEEE-1149.1																				
RESET_pad_ ok. Does not affect daisy-chain outputs.				x																
PLL_RESET_pad_ ok. Record lock time, if possible.				x																
Power dissipation, all I/Os active, 200 usec integration time, 256 Ms/s, all CCCs active, record																				
X input-to-final X output daisy chain delay, all 8 chips in column active; record; max=99.7 ns, including PCB wiring delays.				x																
Y input-to-final Y output daisy chain delay, all 8 chips in column active; record; max=99.7 ns, including PCB wiring delays.				x																
X/Y test vectors, all chips daisy-chained, no errors, 300 MHz. TVEN=1					x															

Normal operation, all CCCs independently correlate, all inputs active, all poln products, 300 MHz. Record core voltage and power required for this test.											x										
Normal operation, all CCCs concatenated, inputs 0,1, 300 MHz.																					
Normal operation, all CCCs concatenated, inputs X 6,7; Y 0,1, 300 MHz.																					
Test Case Number	**0**	**1**	**2**	**3**	**3a**	**4**	**5**	**5a**	**5b**	**6**	**6a**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
Chaining configurations tests (a==all)																					
CCC-0 X: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			P		S			P	S			P		P
CCC-0 Y: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			P		S			P	S			P		A
CCC-1 X: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			P		S			P	S			P		A
CCC-1 Y: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			S		P			S	P			S		P
CCC-2 X: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			S		P			S	P			S		S
CCC-2 Y: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			P		S			P	S			P		A
CCC-3 X: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			S		P			S	P			S		A
CCC-3 Y: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			S		P			S	P			S		S
CCC-4 X: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			P		S			P	S					P
CCC-4 Y: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			P		S			P	S					A
CCC-5 X: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			P		S			P	S					A
CCC-5 Y: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			S		P			S	P					P
CCC-6 X: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			S		P			S	P					S
CCC-6 Y: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			P		S			P	S					A
CCC-7 X: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			S		P			S	P					A
CCC-7 Y: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			S		P			S	P					S
CCC-8 X: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			P		S			P	S					P
CCC-8 Y: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			P		S			P	S					A
CCC-9 X: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			P		S			P	S					A
CCC-9 Y: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			S		P			S	P					P
CCC-10 X: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			S		P			S	P					S
CCC-10 Y: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			P		S			P	S					A
CCC-11 X: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			S		P			S	P					A
CCC-11 Y: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			S		P			S	P					S

CCC-12 X: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			P		S			P	S					P
CCC-12 Y: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			P		S			P	S					A
CCC-13 X: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			P		S			P	S					A
CCC-14 Y: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			S		P			S	P					P
CCC-14 X: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			S		P			S	P					S
CCC-14 Y: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			P		S			P	S					A
CCC-15 X: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			S		P			S	P					A
CCC-15 Y: A-adjacent; P-primary; S-seconday; U-undef. 256 Ms/s							U			S		P			S	P					S
Note: Test #5 sequentially turns on each CCC, one at a time, to test for the correctness of the output frame and the accumulator values. As a consequence, each CCC will also be connected to the undefined (U) input, to deactivate it, thus resulting in testing each CCC for that "non-connection".																					
Test Case Number	**0**	**1**	**2**	**3**	**3a**	**4**	**5**	**5a**	**5b**	**6**	**6a**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
CCQ-0 X Switch 0 (P--Primary; S-Secondary) 256 Ms/s										P		P			P	P			S		P
CCQ-0 X Switch 1 (P--Primary; S-Secondary) 256 Ms/s										P		P			P	P			S		P
CCQ-0 Y Switch 0 (P--Primary; S-Secondary) 256 Ms/s; YS0 is no connect										P		P			P	P			P		P
CCQ-0 Y Switch 1 (P--Primary; S-Secondary) 256 Ms/s; YS1 is no connect										P		P			P	P			P		P
CCQ-1 X Switch 0 (P--Primary; S-Secondary) 256 Ms/s										P		P			S	S					P
CCQ-1 X Switch 1 (P--Primary; S-Secondary) 256 Ms/s										P		P			S	S					P
CCQ-1 Y Switch 0 (P--Primary; S-Secondary) 256 Ms/s										P		P			S	S					P
CCQ-1 Y Switch 1 (P--Primary; S-Secondary) 256 Ms/s										P		P			S	S					P
CCQ-2 X Switch 0 (P--Primary; S-Secondary) 256 Ms/s										P		P			S	S					P
CCQ-2 X Switch 1 (P--Primary; S-Secondary) 256 Ms/s										P		P			S	S					P
CCQ-2 Y Switch 0 (P--Primary; S-Secondary) 256 Ms/s										P		P			S	S					P
CCQ-2 Y Switch 1 (P--Primary; S-Secondary) 256 Ms/s										P		P			S	S					P
CCQ-3 X Switch 0 (P--Primary; S-Secondary) 256 Ms/s										P		P			S	S					P
CCQ-3 X Switch 1 (P--Primary; S-Secondary) 256 Ms/s										P		P			S	S					P
CCQ-3 Y Switch 0 (P--Primary; S-Secondary) 256 Ms/s										P		P			S	S					P
CCQ-3 Y Switch 1 (P--Primary; S-Secondary) 256 Ms/s										P		P			S	S					P

Test Case Number	0	1	2	3	3a	4	5	5a	5b	6	6a	7	8	9	10	11	12	13	14	15	16
CCC-0 X: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					P	S			P	S	P	PS	
CCC-0 Y: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					P	S			P	S	P	PS	
CCC-1 X: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					P	S			P	S	P	PS	
CCC-1 Y: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					S	P			S	P	S	PS	
CCC-2 X: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					S	P			S	P	S	PS	
CCC-2 Y: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					P	S			P	S	P	PS	
CCC-3 X: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					S	P			S	P	S	PS	
CCC-3 Y: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					S	P			S	P	S	PS	
CCC-4 X: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					P	S			P	S		PS	
CCC-4 Y: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					P	S			P	S		PS	
CCC-5 X: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					P	S			P	S		PS	
CCC-5 Y: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					S	P			S	P		PS	
CCC-6 X: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					S	P			S	P		PS	
CCC-6 Y: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					P	S			P	S		PS	
CCC-7 X: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					S	P			S	P		PS	
CCC-7 Y: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					S	P			S	P		PS	
CCC-8 X: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					P	S			P	S		PS	
CCC-8 Y: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					P	S			P	S		PS	
CCC-9 X: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					P	S			P	S		PS	
CCC-9 Y: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					S	P			S	P		PS	
CCC-10 X: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					S	P			S	P		PS	
CCC-10 Y: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					P	S			P	S		PS	
CCC-11 X: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					S	P			S	P		PS	
CCC-11 Y: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					S	P			S	P		PS	
CCC-12 X: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					P	S			P	S		PS	
CCC-12 Y: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					P	S			P	S		PS	
CCC-13 X: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					P	S			P	S		PS	
CCC-14 Y: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					S	P			S	P		PS	
CCC-14 X: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					S	P			S	P		PS	

CCC-14 Y: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					P	S			P	S		PS	
CCC-15 X: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					S	P			S	P		PS	
CCC-15 Y: A-adjacent; P-primary; S-seconday. <256 Ms/s								U					S	P			S	P		PS	
Test Case Number	**0**	**1**	**2**	**3**	**3a**	**4**	**5**	**5a**	**5b**	**6**	**6a**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
CCQ-0 X Switch 0 (P--Primary; S-Secondary) <256 Ms/s													P	P			P	P	S	a	
CCQ-0 X Switch 1 (P--Primary; S-Secondary) <256 Ms/s													P	P			P	P	S	a	
CCQ-0 Y Switch 0 (P--Primary; S-Secondary) <256 Ms/s; YS0 is no connect													P	P			P	P	P	P	
CCQ-0 Y Switch 1 (P--Primary; S-Secondary) <256 Ms/s; YS1 is no connect													P	P			P	P	P	P	
CCQ-1 X Switch 0 (P--Primary; S-Secondary) <256 Ms/s													P	P			S	S		a	
CCQ-1 X Switch 1 (P--Primary; S-Secondary) <256 Ms/s													P	P			S	S		a	
CCQ-1 Y Switch 0 (P--Primary; S-Secondary) <256 Ms/s													P	P			S	S		a	
CCQ-1 Y Switch 1 (P--Primary; S-Secondary) <256 Ms/s													P	P			S	S		a	
CCQ-2 X Switch 0 (P--Primary; S-Secondary) <256 Ms/s													P	P			S	S		a	
CCQ-2 X Switch 1 (P--Primary; S-Secondary) <256 Ms/s													P	P			S	S		a	
CCQ-2 Y Switch 0 (P--Primary; S-Secondary) <256 Ms/s													P	P			S	S		a	
CCQ-2 Y Switch 1 (P--Primary; S-Secondary) <256 Ms/s													P	P			S	S		a	
CCQ-3 X Switch 0 (P--Primary; S-Secondary) <256 Ms/s													P	P			S	S		a	
CCQ-3 X Switch 1 (P--Primary; S-Secondary) <256 Ms/s													P	P			S	S		a	
CCQ-3 Y Switch 0 (P--Primary; S-Secondary) <256 Ms/s													P	P			S	S		a	
CCQ-3 Y Switch 1 (P--Primary; S-Secondary) <256 Ms/s													P	P			S	S		a	
Note: Test# 15 is specifically designed to ensure that SE_CLK routing is correct in the chip, with no assumptions made. I.e. it assigns different sample rates to each input and ensures that SE_CLK routing is correct for each configuration in Tests 6-14. Test# 15 thus tests all (a) routing paths at sample rates <256 Ms/s.																					
Test Case Number	**0**	**1**	**2**	**3**	**3a**	**4**	**5**	**5a**	**5b**	**6**	**6a**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
Multiply, Phase-rotate, Accumulate tests.																					
CCC-0: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, 256 Ms/s.							x														
CCC-1: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, 256 Ms/s.							x														

CCC-2: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, 256 Ms/s.						x														
CCC-3: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, 256 Ms/s.						x														
CCC-4: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, 256 Ms/s.						x														
CCC-5: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, 256 Ms/s.						x														
CCC-6: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, 256 Ms/s.						x														
CCC-7: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, 256 Ms/s.						x														
CCC-8: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, 256 Ms/s.						x														
CCC-9: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, 256 Ms/s.						x														
CCC-10: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, 256 Ms/s.						x														
CCC-11: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, 256 Ms/s.						x														
CCC-12: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, 256 Ms/s.						x														
CCC-13: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, 256 Ms/s.						x														
CCC-14: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, 256 Ms/s.						x														
CCC-15: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, 256 Ms/s.						x														
CCC-0: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, <256 Ms/s.							x													
CCC-1: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, <256 Ms/s.							x													
CCC-2: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, <256 Ms/s.							x													
CCC-3: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, <256 Ms/s.							x													
CCC-4: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, <256 Ms/s.							x													
CCC-5: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, <256 Ms/s.							x													
CCC-6: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, <256 Ms/s.							x													
CCC-7: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, <256 Ms/s.							x													
CCC-8: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, <256 Ms/s.							x													
CCC-9: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, <256 Ms/s.							x													
CCC-10: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, <256 Ms/s.							x													
CCC-11: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, <256 Ms/s.							x													
CCC-12: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, <256 Ms/s.							x													
CCC-13: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, <256 Ms/s.							x													
CCC-14: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, <256 Ms/s.							x													
CCC-15: all Re, Im accums ok, 500 usec, no OV, c/w full-cycle phase, <256 Ms/s.							x													

Test Case Number	0	1	2	3	3a	4	5	5a	5b	6	6a	7	8	9	10	11	12	13	14	15	16
CCC-0: all Re, Im accums ok, >600 usec, OV									x												
CCC-1: all Re, Im accums ok, >600 usec, OV									x												
CCC-2: all Re, Im accums ok, >600 usec, OV									x												
CCC-3: all Re, Im accums ok, >600 usec, OV									x												
CCC-4: all Re, Im accums ok, >600 usec, OV									x												
CCC-5: all Re, Im accums ok, >600 usec, OV									x												
CCC-6: all Re, Im accums ok, >600 usec, OV									x												
CCC-7: all Re, Im accums ok, >600 usec, OV									x												
CCC-8: all Re, Im accums ok, >600 usec, OV									x												
CCC-9: all Re, Im accums ok, >600 usec, OV									x												
CCC-10: all Re, Im accums ok, >600 usec, OV									x												
CCC-11: all Re, Im accums ok, >600 usec, OV									x												
CCC-12: all Re, Im accums ok, >600 usec, OV									x												
CCC-13: all Re, Im accums ok, >600 usec, OV									x												
CCC-14: all Re, Im accums ok, >600 usec, OV									x												
CCC-15: all Re, Im accums ok, >600 usec, OV									x												
Test Case Number	**0**	**1**	**2**	**3**	**3a**	**4**	**5**	**5a**	**5b**	**6**	**6a**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
Output frame contents (checked for each CCC):																					
CMD--all bit combinations applicable: 000, 001, 010, 011, 100, 101.																					
CCC#--same as the CCC under test.							x	x	x												
NUM_CLAGS=128							x	x	x												
STATUS bits:																					
OVR																					
ACC_OV									x												
XSyncerr																					
YSyncerr																					
X_CCC_input																					x
Y_CCC_input																					x
SID-X: all bits exercised independently.																					
SBID-X: all bit combinations.																					

BBID-X: all bit combinations.																				
SID-Y: all bits exercised independently.																				
SBID-Y: all bit combinations.																				
BBID-Y: all bit combinations.																				
RECIRC_BLK-X: all bits exercised independently.																				
RECIRC_BLK-Y: all bits exercised independently.																				
Phase Bin: bins 0-999 and banks 0,1																				
TIMESTAMP-0: all bits exercised independently.																				
TIMESTAMP-1: all bits exercised independently.																				
DVCOUNT-Center: exercised for range of integration times from 20 usec to 600 usec																				
DVCOUNT-Edge: exercised for range of integration times from 20 usec to 600 usec																				
DATA_BIAS: exercised for range of integration times from 20 usec to 600 usec																				
Lag accumulators: bit exact with behavioural sim; minimum complete wrap of phase, range of integration times from 20 to 600 usec.																				
START SYNC Word correct.							x													
END SYNC Word correct.							x													
CHECKSUM correct.							x													
Test Case Number	**0**	**1**	**2**	**3**	**3a**	**4**	**5**	**5a**	**5b**	**6**	**6a**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
Dump control parameters.																					
NOTE: all bits exercised, to be considered tested.																					
X-0 input dump_en: F=function; C=CMD; P=Phase bin; R=Recirc block; H=holdof							F														
X-1 input dump_en: F=function; C=CMD; P=Phase bin; R=Recirc block; H=holdof							F														
X-2 input dump_en: F=function; C=CMD; P=Phase bin; R=Recirc block; H=holdof							F														
X-3 input dump_en: F=function; C=CMD; P=Phase bin; R=Recirc block; H=holdof							F														
X-4 input dump_en: F=function; C=CMD; P=Phase bin; R=Recirc block; H=holdof							F														
X-5 input dump_en: F=function; C=CMD; P=Phase bin; R=Recirc block; H=holdof							F														
X-6 input dump_en: F=function; C=CMD; P=Phase bin; R=Recirc block; H=holdof							F														
X-7 input dump_en: F=function; C=CMD; P=Phase bin; R=Recirc block; H=holdof							F														
NOTE: all bits exercised, to be considered tested.																					

Y-0 input dump_en: F=function; C=CMD; P=Phase bin; R=Recirc block; H=holdof							F														
Y-1 input dump_en: F=function; C=CMD; P=Phase bin; R=Recirc block; H=holdof							F														
Y-2 input dump_en: F=function; C=CMD; P=Phase bin; R=Recirc block; H=holdof							F														
Y-3 input dump_en: F=function; C=CMD; P=Phase bin; R=Recirc block; H=holdof							F														
Y-4 input dump_en: F=function; C=CMD; P=Phase bin; R=Recirc block; H=holdof							F														
Y-5 input dump_en: F=function; C=CMD; P=Phase bin; R=Recirc block; H=holdof							F														
Y-6 input dump_en: F=function; C=CMD; P=Phase bin; R=Recirc block; H=holdof							F														
Y-7 input dump_en: F=function; C=CMD; P=Phase bin; R=Recirc block; H=holdof							F														
X-Timestamp: all bits exercised.																					
Y-Timestamp: all bits exercised.																					
Test Case Number	**0**	**1**	**2**	**3**	**3a**	**4**	**5**	**5a**	**5b**	**6**	**6a**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
High performance dump and data integrity.																					
Fast, burst-dump, all CCCs, no overruns (dump-save ~160 usec)																					
Fast, burst-dump, all CCCs, overruns (dump-save ~20 usec)																					
Fast, burst-dump, all CCCs, overruns (dump-integrate-save ~20 usec)																					
Autocorrelator mode tests.																					
Note: for autocorrelator mode tests, the Y_CLOCK input held low.																					
Autocorrelator mode: X input 0 active, 256 Ms/s																					
Autocorrelator mode: X input 1 active, 256 Ms/s																					
Autocorrelator mode: X input 2 active, 256 Ms/s																					
Autocorrelator mode: X input 3 active, 256 Ms/s																					
Autocorrelator mode: X input 4 active, 256 Ms/s																					
Autocorrelator mode: X input 5 active, 256 Ms/s																					
Autocorrelator mode: X input 6 active, 256 Ms/s																					
Autocorrelator mode: X input 7 active, 256 Ms/s																					
Autocorrelator mode: X input 0 active, <256 Ms/s																					
Autocorrelator mode: X input 1 active, <256 Ms/s																					

Autocorrelator mode: X input 2 active, <256 Ms/s																					
Autocorrelator mode: X input 3 active, <256 Ms/s																					
Autocorrelator mode: X input 4 active, <256 Ms/s																					
Autocorrelator mode: X input 5 active, <256 Ms/s																					
Autocorrelator mode: X input 6 active, <256 Ms/s																					
Autocorrelator mode: X input 7 active, <256 Ms/s																					
Test Case Number	**0**	**1**	**2**	**3**	**3a**	**4**	**5**	**5a**	**5b**	**6**	**6a**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
MCB register bits.																					
NOTE: all MCB control and status register tests must ensure the bit under test operates properly in both its set (1) and reset (0) state																					
MCSR-CEN							x														
MCSR-TVEN		x	x	x		x															
MCSR-PhEN																					
MCSR-OVR																					
MCSR-AOV									x												
MCSR-TvER						x															
MCSR-SyncER																					
MCSR-X/Y_DS																					
NOTE: CCCSCR_0_1 … CCCSCR_14_15 functions tested with CCC input combinations. This tests the ability to set/reset each bit independently.																					
NOTE: CCQR_1/2, CCQR_3/4 functions tested with CCC input combinations. This tests the ability to set/reset each bit independently.																					
XYSSR: XAd, YAd: 0,1,2, …, C						x															
XSTATUS, all XAds, all bits						x															
YSTATUS, all YAds, all bits						x															
DESSR_0/7																					
DESSR_8/15																					
Miscl tests.																					
Correlation, all CCCs in any chaining configuration, with no blanking time around the dump pulse. Statistical comparison with behavioural simulation.																					

4 Verification Test Cases

Test #	Purpose of Test	Description/Test Setup	Required Outcome	Notes
		Note: all tests are with constant Corr Chip case temperature of ~100 C unless otherwise noted.		
0	PLL Jitter measurement	Set the chip for testvector mode (TVEN=1). Measure the jitter into the X and Y clock inputs, measure the jitter on the X and Y clock outputs.	The short-term cycle-to-cycle jitter on the output must be less than on the input. Long-term jitter may increase, but not by a significant amount (TBD…possibly 0.05 UI)	Absolute jitter is not of concern. The only concern is that jitter does not build up to the point where downstream chips cannot lock to the clock, or attached LTA FPGAs cannot lock to the clock (this may set the most stringent requirement that jitter is <150 ps pk-pk.)
1	Test vector reception	Set the chip for testvector mode (TVEN=1). Set the Recirc FPGAs to generate testvectors with no errors. Corr Chip case temperature is 100 C with 2 cycles to -35 C with cold spray.	The TvER bit of the MCSR must not be reporting errors. The XSTATUS and YSTATUS registers must not be reporting errors.	

2	Test vector reception with errors	Set the chip for testvector mode (TVEN=1). Set the Recirc FPGAs to generate testvectors. For each of X and Y, one line at a time, set the BFCR register in Recirc FPGAs to force lines low and high.	The TvER bit of the MCSR must be reporting errors. The individual status bits of the XSTATUS and YSTATUS registers must be reporting errors on the particular line that is being forced into an error condition.	The following conditions will not result in errors: DVALID low, SE_CLK high. If the DUMP_SYNC line is held high or low, unpredictable operation will occur…what happens should be noted. If the SCHID_FRAME input is held high or low, unpredictable operation will occur...what happens should be noted.

3	Test vector reception with all chips daisy-chained (all chips on the board installed); measurement of Corr Chip X/Y output timing.	Set all Corr Chips for testvector mode (TVEN=1). Set the Recirc FPGAs to generate testvectors.	The TvER bit of the MCSR in all Corr Chips on the board must not be reporting errors. The XSTATUS and YSTATUS registers must not be reporting errors. Measure and record the tsu and thold time of as many X/Y outputs as possible relative to X/Y clocks for at least 5 chips in arbitrary locations on the board. Measure X and Y output clock jitter for the all of the last chips of rows and columns.	This test will only be done once all single-chip tests are complete.

3a	Repeat Test #3, but at 300 MHz.	Same as Test #3, but at 300 MHz.	The chip should operate normally with no errors.	This is a design margin test and may require setting the Accel voltage higher. Failure of this test may not mean failure of the correlator chip. 300 MHz is likely the limit because of limitations in the speed of the HM Gps link (I.e. 300 MHz means the link must operate at 1.2 Gs/s)

4	Test vector reception with all chips daisy-chained (all chips on the board installed), with errors	Set all chips for testvector mode (TVEN=1). Set the Recirc FPGAs to generate testvectors. For each of X and Y Recirc FPGAs, one line at a time, set the BFCR register in Recirc FPGAs to force lines low and high. Proceed in a row-by-row, and column by column fashion.	The TvER bit of the MCSR in all chips in the same row or column must be reporting errors. The individual status bits of the XSTATUS and YSTATUS registers in all chips in the same row or column must be reporting errors on the particular line that is being forced into an error condition.	The following conditions will not result in errors: DVALID low, SE_CLK high. If the DUMP_SYNC line is held high or low, unpredictable operation will occur…what happens should be noted. If the SCHID_FRAME input is held high or low, unpredictable operation will occur...what happens should be noted.

5	First functional test, one CCC active one at a time in one Corr Chip, primarily for analysis of the correctness of the output frame, and of each CCC's functionality. Test dumping with both X and Y	Set the Corr Chip so that, starting with CCC0, each CCC is active one at a time. Correlate testvectors 6,7 (requires x-bar setting in Recirc FPGA to route to stream 0 going to the Corr Chip), 4,5, and 2,3 set for 256 Ms/s, 500 usec integration time. Set dump CMD to "Speed Dump" at full output rate of 2000 frames/sec. Capture data frames in the Backend, and integrate for 100 seconds for each Backend output.	Observe raw frames with no Backend integration for correct content. Integrate frames for 100 seconds in Backend (200,000 frames) and compare ASCII Re/Im with behavioural simulation results file: **CC6x7_test00_128lagsD0.dat CC4x5_test00_128lagsD0.dat CC2x3_test00_128lagsD0.dat.** Comparison must yield errors that are due only to double-precision floating point numerical error (~10^-15) Comparison of many 100-second dumps over 24 hours should be made to ensure there are no intermittent errors.	The entire data path all the way to the Backend must work for this test to be successful. This test bypasses the LTA DDR SDRAM so the CMIB cannot read frames from the LTA. If necessary (i.e. can't get data to Backend) use an ad-hoc LTA design to capture raw frames for readout by the CMIB. Using test vectors 6,7 ensures complete phase wraps in the chip.

5a	Same as Test #5, except for 128 Ms/s	Same as Test #5, except for 128 Ms/s	Same as Test #5, except for 128 Ms/s	

5b	Same as Test #5, except set integration time for 600 usec to induce overflow.	Same as Test #5, except set integration time for 600 usec to induce overflow. Test need only be performed on set of test vectors, and only for short period of time.	Ensure overflow bit in frame header and MCSR register are set. Accumulator values should indicate that overflow occurred (I.e. the MSBits of all accumulators should be reset).	

6	Full functional test, all CCCs active in one Corr Chip, all input streams active @ 256 Ms/s, all poln products, LTA integration.	Set the Corr Chip so that each CCC is correlating independently, all polarization products in "normal" all products mode: CCC0-0x0; CCC1-0x1; CCC2-1x0; CCC3-1x1 etc. 256 Ms/s, all streams. Set for LTA integration time for max or 1.0 secs, primarily to limit frame rate to Backend. Backend integration time of ~100 secs. Corr Chip case temperature is 100 C with 2 cycles to -35 C with cold spray during test.	Compare integrations from Backend ASCII Re/Im with behavioural simulation result files: **CC0x0_test00_128lagsD0.dat CC0x1_test00_128lagsD0.dat CC1x0_test00_128lagsD0.dat CC1x1_test00_128lagsD0.dat … CC7x7_test00_128lagsD0.dat** etc. Comparison of many 100-second dumps over 2 hours should be made to ensure there are no intermittent errors.	This test uses the LTA DDR SDRAM for integration and completely exercises the data path all the way to the Backend. If errors are detected, ensure that it is not problems with the LTA or LTA RAM.

6a	Same as Test #6, except set for 300 MHz	Same setup as Test #6, 300 MHz.	Same comparisons as Test #6	The LTA FPGA may not meet timing at this speed and may be the cause for failure. Re-compilation of the LTA FPGA for 150 MHz may be required.

7	Same as Test #6, except each CCC correlates the other input of the pair.	Set the Corr Chip so that each CCC is correlating independently, all polarization products in "normal" all products mode: CCC0-1x1; CCC1-1x0; CCC2-0x1; CCC3-0x0 etc. 256 Ms/s, all streams. Set for LTA integration time max or 1.0 secs, primarily to limit frame rate to Backend. Backend integration time of ~100 secs. Corr Chip case temperature is 100 C with 2 cycles to -35 C with cold spray during test.	Compare integrations from Backend ASCII Re/Im with behavioural simulation result files: **CC1x1_test00_128lagsD0.dat CC1x0_test00_128lagsD0.dat CC0x1_test00_128lagsD0.dat CC0x0_test00_128lagsD0.dat … CC6x6_test00_128lagsD0.dat** etc. Comparison of many 100-second dumps over 2 hours should be made to ensure there are no intermittent errors.	

8	Same as Test #6, except all streams at 128 Ms/s, 64 Ms/s, 32 Ms/s, …, 62.5 ks/s.	Same as Test #6, except all streams at lower sample rates. This requires stretching out TIMECODE T-bit generation by an appropriate factor. DUMPTRIG period must be appropriately set as well.	Same as Test #6, except all streams at 128 Ms/s, 64 Ms/s, …, 62.5 ks/s. The results should be the same, since the same test vectors get correlated. 2 hours not required…short test for each sample rate only.	

9	Same as Test #7, except all streams at 128 Ms/s, 64 Ms/s, 32 Ms/s, …, 62.5 ks/s.	Same as Test #7, except all streams at lower sample rates. This requires stretching out TIMECODE T-bit generation by an appropriate factor. DUMPTRIG period must be appropriately set as well.	Same as Test #7, except all streams at 128 Ms/s, 64 Ms/s, …, 62.5 ks/s. The results should be the same, since the same test vectors get correlated. 2 hours not required…short test for each sample rate only.	

10	Stream 0, 1 distribution and correlation test @ 256 Ms/s.	Set the Corr Chip so that each CCQ is identically correlating the following products: 0x0, 0x1, 1x0, 1x1. Set for LTA integration time (max) of 1.0 secs. Backend integration time of ~100 secs. Corr Chip case temperature is 100 C with 1 cycle to -35 C with cold spray during test.	Compare integrations from Backend ASCII Re/Im with behavioural simulation results files: **CC0x0_test00_128lagsD0.dat CC0x1_test00_128lagsD0.dat** etc… Comparison of many 100-second dumps over 2 hours should be made to ensure there are no intermittent errors.	

11	Same as Test #10, except correlate opposite streams within each CCC.	Same as Test #10, except that each CCQ is identically correlating the following products: 1x1, 1x0, 0x1, 0x0.	Same as Test #10, with appropriate comparisons.	

12	Same as Test #10, except all streams at 128 Ms/s, 64 Ms/s, 32 Ms/s, …, 62.5 ks/s.	Same as Test #10, except lower sample rates as noted.	Same ast Test #10, except lower sample rates as noted. 2 hours not required…short test for each sample rate only.	

13	Same as Test #11, except all streams at 128 Ms/s, 64 Ms/s, 32 Ms/s, …, 62.5 ks/s.	Same as Test #11, except lower sample rates as noted.	Same ast Test #11, except lower sample rates as noted. 2 hours not required…short test for each sample rate only.	

14	Test X stream 6 and 7 routing to CCQ-0 at 256 Ms/s	Set CCQ-0 to correlate the products 6x0, 6x1, 7x0, 7x1.	Compare integrations from Backend ASCII Re/Im with behavioural simulation results files: **CC6x0_test00_128lagsD0.dat CC6x1_test00_128lagsD0.dat** etc… Comparison of many 100-second dumps over 2 hours should be made to ensure there are no intermittent errors.	

15	Repeat tests 6-14, but with different sample rates for each input stream number. This test verifies that all SE_CLKs in the chip get routed to the correct places.	Repeat tests 6-14. Short verification tests only. If this is done with test vector stimulus (rather than a Station Board) then the Recirc FPGA must be set to use SE_CLKs that are incompatible with test vectors since testvectors are generated with sample rates that are the same for all streams. If a particular stream and CCC under consideration is set to produce the correct results, other streams can be set to produce incorrect results to verify correct SE_CLK routing in the chip.	Verify for each input stream and each test, that the SE_CLKs are routed to the correct CCC.	

NOTE: Tests 6-15 verify that all X/Y streams are routed to their correct locations in the chip, at all sample rates, including data valid, se_clk, dump_en routing. For better coverage, the tests could be repeated with the X-bar switch in the Recirc FPGA set to "cross-route" streams (e.g. test vector stream 0 routes to output 7, test vector stream 1 routes to output 6 etc.). This will maximize the number of different combinations of vectors going into each CCC, and minimize the pathological case where (for example) a CMAM multiplier did not exercise all combinations of inputs because the test vector sequence may not have been long enough.

16	Concatenate adjacent CCCs, 256 Ms/s, form 2 poln products per CCQ.	Concatenate adjacent CCCs within CCQs (e.g. CCC0-CCC1; CCC2-CCC3 etc.). Connect to all input streams forming 2 polarization pairs: CCC0-CCC1--0x0; CCC2-CCC3--1x1 etc.	Verify for each pair of concatenated CCCs that frames are produced with identical timestamps, and that results are within double precision numerical precision of the following files: **CC0x0_test16_256lagsD0.dat** **CC1x1_test16_256lagsD0.dat** etc… Backend integration time only long enough to verify correctness.	This tests dump_en and frame building logic for this concatenation setup.

17	Same as Test #16, except from cross poln products.	Same as Test #16, except adjacent concatenated CCCs form the cross polarization products: 0x1, 1x0 etc.	Verify for each pair of concatenated CCCs that frames are produced with identical timestamps, header information/IDs are correct, and that results are within double precision numerical precision of the following files: **CC0x1_test00_256lagsD0.dat** **CC1x0_test00_256lagsD0.dat** etc.	This tests dump_en and frame building logic for this concatenation setup.

18	Same as Test #17, except form opposite cross poln products.	Same as Test #17, except adjacent concatenated CCCs form the opposite cross polarization products: 1x0, 0x1 etc.	Verify for each pair of concatenated CCCs that frames are produced with identical timestamps, header information/IDs are correct, and that results are within double precision numerical precision of the following files: **CC1x0_test00_256lagsD0.dat** **CC0x1_test00_256lagsD0.dat** etc.	This tests dump_en and frame building logic for this concatenation setup.

19	Same as Test #16, except form opposite 2 poln products in each CCQ	Concatenate adjacent CCCs within CCQs. Connect to all input streams forming 2 polarization pairs: CCC0-CCC1--1x1; CCC2-CCC3--0x0 etc.	Same as Test #16, except opposite 2 poln products.	This tests dump_en and frame building logic for this concatenation setup.

20	Repeat tests 16-19, but use different sample rates on each stream to ensure that SE_CLK routing in the chip is correct.	Repeat tests 16-19, but use different sample rates to ensure SE_CLK routing in the chip is correct. With test vectors this can be accomplished by setting the correct sample rate in the Recirc FPGA for the pair of CCCs currently under consideration. Only 1 sample rate less than 256 Ms/s need be used to ensure full test coverage.	Same verification procedure as Tests 16-19. Note that data valid counts should be the same factor more than Tests 16-19 as the sample rate reduction factor.	This tests SE_CLK routing logic for this concatenation setup.

21	Concatenate adjacent CCCs within each CCQ, connect all of them to inputs 0 and 1, form poln and cross-poln products. 256 Ms/s	Concatenate adjacent CCCs within CCQs. Connect all to input streams 0 and 1, sequentially forming 0x0, 1x1, 0x1, and 1x0 products.	Verify that frames from redundant correlations in different CCQs are identical. Verify against CC0x0_test16_256lagsD0.dat etc.	This tests dump_en and frame building logic for this concatenation setup.

22	Concatenate all CCCs within each CCQ. Correlate with even-stream inputs. 256 Ms/s	Concatenate all CCCs within each CCQ. Connect to even input streams (0, 2, 4, 6) forming 1 polarization product: CCC0-CCC1-CCC2-CCC3--0x0 etc.	Verify for each set of concatenated CCCs that frames are produced with identical timestamps, header information/IDs are correct, and that results are within double precision numerical precision of the following files: **CC0x0_test00_512lagsD0.dat** **CC2x2_test00_512lagsD0.dat** etc.	This tests dump_en and frame building logic for this concatenation setup. It also tests CCC concatenation logic.

23	Same as Test #22, except correlate with odd-stream inputs.	Concatenate all CCCs within each CCQ. Connect to odd input streams (1, 3, 5, 7) forming 1 polarization product: CCC0-CCC1-CCC2-CCC3--1x1 etc.	Same verification as Test #22, except compare with odd stream correlations: **CC1x1_test00_512lagsD0.dat** **CC3x3_test00_512lagsD0.dat** etc.	This tests dump_en and frame building logic for this concatenation setup. It also tests CCC concatenation logic.

24	Same as Test #22, except correlate with cross-stream inputs.	Concatenate all CCCs within each CCQ. Form even x odd cross-products: CCC0-CCC1-CCC2-CCC3--0x1; CCC4-CCC5-CCC6-CCC7--2x3 etc.	Same verification as Test #22, except compare with cross-stream correlations: **CC0x1_test00_512lagsD0.dat** **CC2x3_test00_512lagsD0.dat** etc.	This tests dump_en and frame building logic for this concatenation setup. It also tests CCC concatenation logic.

25	Same as Test #24, except correlate with opposite cross-stream inputs.	Concatenate all CCCs within each CCQ. Form odd x even cross-products: CCC0-CCC1-CCC2-CCC3--1x0; CCC4-CCC5-CCC6-CCC7--3x2 etc.	Same verification as Test #24, except compare with cross-stream correlations: **CC1x0_test00_512lagsD0.dat** **CC3x2_test00_512lagsD0.dat** etc.	This tests dump_en and frame building logic for this concatenation setup. It also tests CCC concatenation logic.

26	Repeat tests 22-25, except at lower sample rate.	Set sample rate lower than 256 Ms/s to ensure correct SE_CLK routing in the chip.	Same verification as Test 22-25. Note that data valid counts should be the same factor more than Tests 22-25 as the sample rate reduction factor.	This tests SE_CLK routing for this concatenation in the chip.

27	Concatenate all CCCs within a CCQ, connect all to inputs 0 and 1. 256 Ms/s.	Concatenate all CCCs within each CCQ. Connect all to inputs 0 and 1, successively forming 0x0, 1x0, 0x1, and 1x1 products.	Verify for each set of concatenated CCCs and for each product that redundant results are identical, and that results are within double precision numerical precision of the following files: CC0x0_test00_512lagsD0.dat CC1x1_test00_512lagsD0.dat etc.	This tests dump_en and frame building logic for this concatenation setup.

28	Concatenate all CCCs within a CCQ; concatenate adjacent CCQs. 256 Ms/s.	Concatenate all CCCs within each CCQ and then concatenate adjacent CCQs. Within each 1024-lag concatenation, perform the full battery of cross-product tests for all possible inputs: CCQ0-CCQ1: 0x0, 0x1, 0x2, 0x3, 1x0, 1x1, 1x2, 1x3. CCQ2-CCQ3: 4x6, 4x7, 5x6, 5x7.	Verify for each set of concatenated CCCs and for each product that frame headers are correct, and that results are within double precision numerical precision of the following files: **CC0x0_test00_1024lagsD0.dat** **CC0x1_test00_1024lagsD0.dat** etc.	This tests dump_en, frame building, and concatenation logic for this setup.

29	Same as Test #28, except use a lower sample rate.	Same as Test #28, except use a lower sample rate to ensure correct SE_CLK routing. If using test vectors, set the Recirc FPGA so that it is using the correct sample rate for only the particular streams and cross-correlation under consideration so that any SE_CLK routing errors show up as bad results.	Same verification as Test #28, except that the data valid counts should be the same factor more as the sample rate reduction factor.	This tests SE_CLK routing for this setup.

30	Same as Test #28, except connect all to inputs 0 and 1	For each 1024-lag section, test products 0x0, 0x1, 1x0, 1x1.	Same verification as Test #28.	This tests dump_en, frame building, and concatenation logic for this setup.

31	Same as Test #30, except lower sample rate.	Same as Test #30, except lower sample rate.	Same verification as Test #30, except that the data valid counts should be the same factor more as the sample rate reduction factor.	This tests SE_CLK routing for this setup.

32	Concatenate all CCCs in the chip. 256 Ms/s	Concatenate all CCCs with the chip. Test for the following cross products: 0x0, 0x1, 0x6, 0x7, 1x0, 1x1, 1x6, 1x7, 6x0, 6x1, 6x6, 6x7, 7x0, 7x1, 7x6, 7x7. Corr	Verify that frame headers are correct and that results are within double precision numerical precision of the following files: **CC0x0_test00_2048lagsD0.dat** **CC0x1_test00_2048lagsD0.dat** etc.	This tests dump_en, frame building, and concatenation logic for this setup.

32a	Same setup as Test #32, except run at 300 MHz.	Same setup as Test #32.	Same as Test #32	This is a speed margin test that may require compilation of the LTA FPGA at 150 MHz. The Accel voltage max may need to be set for this test. Failure of this test may not mean chip failure.

33	Same as Test #32, except lower sample rate.	Test all cross products same as Test #32, except use lower sample rate. If using test vectors, set the Recirc FPGA so that it is using the correct sample rate for only the particular streams and cross-correlation under consideration so that any SE_CLK routing errors show up as bad results.	Same verification as Test #32, except that the data valid counts should be the same factor more as the sample rate reduction factor.	This tests SE_CLK routing for this setup.

34	Different concatenation settings in each CCQ, 256 Ms/s	CCQ0: 4 poln products, inputs 0,1; CCQ1: 2 poln products, inputs 2,3; CCQ2: 1 product, inputs 4,5; CCQ3: 2 products, inputs 6,7 (CCC12-14:7x6; CCC15;6x7).	Verify that frame headers are correct and that results are within double precision numerical precision of the following files: **CC0x0_test00_128lagsD0.dat** **CC0X1_test00_128lagsD0.dat** **CC1x0_test00_128lagsD0.dat** **CC1x1_test00_128lagsD0.dat** **CC2x3_test00_256lagsD0.dat** **CC3x2_test00_256lagsD0.dat** **CC4x5_test00_512lagsD0.dat** **CC7x6_test00_384lagsD0.dat** **CC6x7_test00_128lagsD0.dat**	This mixes up concatenation settings within each CCQ within the chip. Tests to ensure that configuration bit settings in the CCCSCR_0_1, 2_3, 4_5, 6_7 registers are not, somehow, stuck or connected to each other. Not all mixtures/permutations can be tested.

35	Different concatenation settings in CCQs, 256 Ms/s, rotated by one CCQ relative to Test #34	Same kind of setup as Test #34, except rotate setup to next CCQ. E.g. CCQ0 uses Test #34 CCQ3 setup, CCQ1 uses Test #34 CCQ0 setup etc.	Verify that frame headers are correct and that results are within double precision numerical precision of the following files: **CC2x2_test00_128lagsD0.dat** **CC2X3_test00_128lagsD0.dat** **CC3x2_test00_128lagsD0.dat** **CC3x3_test00_128lagsD0.dat** **CC4x5_test00_256lagsD0.dat** **CC5x4_test00_256lagsD0.dat** **CC6x7_test00_512lagsD0.dat** **CC1x0_test00_384lagsD0.dat** **CC0x1_test00_128lagsD0.dat**	This mixes up concatenation settings within each CCQ within the chip, but rotated by one CCQ relative to Test #34.

36	Different concatenation settings in CCQs, 256 Ms/s, rotated by two CCQs relative to Test #34	Same kind of setup as Test #35, except rotate setup to next CCQ. E.g. CCQ0 uses Test #35 CCQ3 setup, CCQ1 uses Test #35 CCQ0 setup etc.	Verify that frame headers are correct and that results are within double precision numerical precision of the following files: **CC4x4_test00_128lagsD0.dat** **CC4x5_test00_128lagsD0.dat** **CC5x4_test00_128lagsD0.dat** **CC5x5_test00_128lagsD0.dat** **CC6x7_test00_256lagsD0.dat** **CC7x6_test00_256lagsD0.dat** **CC0x1_test00_512lagsD0.dat** **CC3x2_test00_384lagsD0.dat** **CC2x3_test00_128lagsD0.dat**	This mixes up concatenation settings within each CCQ within the chip, but rotated by two CCQs relative to Test #34.

37	Different concatenation settings in CCQs, 256 Ms/s, rotated by three CCQs relative to Test #34	Same kind of setup as Test #36, except rotate setup to next CCQ. E.g. CCQ0 uses Test #36 CCQ3 setup, CCQ1 uses Test #36 CCQ0 setup etc.	Verify that frame headers are correct and that results are within double precision numerical precision of the following files: **CC6x6_test00_128lagsD0.dat** **CC6x7_test00_128lagsD0.dat** **CC7x6_test00_128lagsD0.dat** **CC7x7_test00_128lagsD0.dat** **CC0x1_test00_256lagsD0.dat** **CC1x0_test00_256lagsD0.dat** **CC2x3_test00_512lagsD0.dat** **CC5x4_test00_384lagsD0.dat** **CC4x5_test00_128lagsD0.dat**	This mixes up concatenation settings within each CCQ within the chip, but rotated by three CCQs relative to Test #34.

38	Repeat tests 34-37, but at different sample rates for each input stream	Same setup as tests 34-37	Same comparisons as tests 34-37	When testing with testvectors, one stream pair, and thus one CCQ will be tested at a time.

NOTE: Tests 16-39 check all concatenation modes in the chip and, within each concatenation mode, all possible combinations of input streams are checked for each set of concatenated CCCs. Data valid, dump_en, SE_CLK routing is also check vigorously for each one. Additionally, tests 34-38 test mixed concatenation modes in the chip, although it is impossible to test all possible permutations of mixed concatenation modes.

40	**Autocorrelator mode** tests, all streams, 256 Ms/s	Set the Corr Chip and Recirc FPGAs for autocorrelator mode, successively autocorrelating streams 0, 1, 2, …, 7. Corr Chip case temperature is 100 C with 1 temperature cycle to -35 C with cold spray.	Verify lags and autocorrelation spectrum with "expected results". A direct comparison with behavioural simulation results is not possible due to the inability of the behavioural simulator to put the peak of the autocorrelation function at the edge lag. However, the lag and spectral signature should be unmistakable, for each input stream.	This tests autocorrelator mode at full speed, all streams. Autocorrelator mode is not normally required for the EVLA.

41	**Autocorrelator mode** tests, all streams, lower sample rate	Same as Test #40, at lower sample rate. No temperature cycle is required.	Same verification as Test #40, except that at lower sample rate.	

50	Fast dump and integrate, all CCCs active, no overruns.	Set the dump period to 200 usec. Use any concatenation mode where all CCCs are active. Set the LTA integration time for max or 1.0 seconds. Set the Backend integration time for 100 seconds. Corr Chip case temperature is 100 C with 1 temperature cycle to -35 C with cold spray.	Verify that no dump overruns or overflows are detected and that frame headers and results are correct, in comparison also with the behavioural simulation results.	This tests the fast dump and integrate, all CCCs active function of the chip.

50a	Fast burst dump, all CCCs active, no overruns.	Set the dump period to 160 usec. Use any concatenation mode where all CCCs are active. Set the LTA integration time for max or 1.0 seconds. Set the Backend integration time for 100 seconds. Corr Chip case temperature is 100 C with 1 temperature cycle to -35 C with cold spray.	Verify that no dump overruns or overflows are detected and that frame headers and results are correct, in comparison also with the behavioural simulation results.	This tests the fast burst dump, all CCCs active function of the chip.

51	Fast dump and integrate, all CCCs active, with overruns	Set the dump period to 20 usec. Use any concatenation mode where all CCCs are active. Set the LTA integration time for max or 1.0 seconds. Set the Backend integration time for 100 seconds. Corr Chip case temperature is 100 C with 1 temperature cycle to -35 C with cold spray.	Verify that dump overruns occur…their occurrence is rather random and impossible to predict, however the CCCs with longer integration times should have fewer overrun occurences. The chip should not lock up, no accumulator overflows should occur, and all data products coming out of the Backend should be good and compare favourably with the behavioural simulation results. Ensure the OVR bits in every CCC's output frame header get set.	This tests very fast dump and integrate, all CCCs active, and the Corr Chip's ability to handle pathological dump conditions. This may likely result in the LTA completely filling up, and then emptying as priority transfers from the Corr Chip-to-LTA to LTA-to-GigE data transfer.

52	Fastest dump and integrate, one CCC active at a time, no overruns.	Set the dump period to 15 usec (if possible, 20 usec if not possible). Set all CCCs for individual correlations, sequentially with one CCC active at a time. Set the LTA integration time for max or 1.0 seconds. Set the Backend integration time for 100 seconds.	Verify that no dump overruns occur (OVR bit of the MCSR, OVR bit in frame header), there are no accumulator overflows (AOV bit in MCSR, ACC_OV bit in the frame header), that frame headers are all good, and that data compares favourably with the behavioural simulation results.	This tests the fastest dump and integrate in the chip, one CCC active at a time.

53	Fastest burst dump, one CCC active at a time, no overruns.	Set the dump period to 10 usec (if possible, 20 usec if not possible). Set all CCCs for individual correlations, sequentially with one CCC active at a time. Set the LTA integration time for burst dump. Set the Backend integration time for no integration and for 100 second integration and check results accordingly in both cases.	Verify that no dump overruns occur, there are no accumulator overflows, that frame headers are all good, and that data compares favourably with the behavioural simulation results. No overrun frames should be detected.	This test generates 100,000 frames/sec (10 usec) or 50,000 frames/sec (20 usec) and, since this is near the ~100 kframes/sec limit of GigE, and likely a much lower limit for Backend CPU, a significant number of frames will likely be missing.

53a	Fastest burst dump, all CCCs active, overruns.	Same setup as Test #53, except all CCCs active.	Dump overruns occur, but any frames produced are good frames, comparing favourably with the behavioural simulation results. The chip must not lock up.	

NOTE: Tests 40-43 test the fast dumping and integrating capabilty of the Corr Chip. It is important that the chip not lock up during any of these tests.

60	X/Y_DS (X or Y dump select) test	Set the dump period for stream 0 to <500 usec, and for stream 1 for > 500 usec. Correlate, using one or more of any CCCs 0x1, first with dumping from X and then from dumping from Y. Backend integration time is irrelevant.	When dumping from X, there should be no accumulator overflows. When dumping from Y, there are accumulator overflows, checked by looking at the overflow bit of the output frame and the AOV bit of the MCSR. Check ACC_OV bit set in the frame header for each CCC when dump period is >500 usec.	

61	On-line synchronization error detects on all inputs.	Set Corr Chip for correlating all CCCs using all input streams. Use the Recirc FPGA BFCR function to individually set each X and Y input line forced to 0 or 1.	The SyncER bit in the MCSR, as well as the similar bit in the output frame, and the XSTATUS and YSTATUS registers must indicate detection of the error condition.	Some inputs forced high or low will not ellicite error detection (refer to Test #2)

62	Phase rotation enable/disable test (PhEN bit of the MCSR)	Set Corr Chip for correlating all CCCs using all input streams. With the PhEN bit set (1) observe normal output using whatever LTA and Backend integration time is desired. Reset (0) the PhEN bit.	The Im lags for all outputs must be zero. This indicates that phase rotation has been successfully disabled.	

63	DESSR register checks	This test covered under previous tests when correlating different streams, since (with test vectors) each stream has a different dump period.	Ensure with previous tests when not correlating the same stream that each DESSR register bit is set and reset.	Observe DESSR registers from previous tests to ensure testing.

64	Frame header input X and Y ID checks	Set CCCs for all independent correlations, all poln products, all streams active; set CCQs for concatenation of CCCs within them; set for concatenation of all CCCs; set for all CCCs independently correlating stream 0; set for all CCCs independently correlating stream 1. Set X and Y SIDs, SBIDs, and BBIDs through the following combinations on each data stream independently; other data streams should have IDs set to all 1's or all 0's when not under test. SIDs: 00000000, 11111111, 01010101, 10101010, 11110000, 00001111, 10000000, 01000000, 00100000, 00010000, 00001000, 00000100, 00000010, 00000001; SBIDs: all bit cominations from 0x00 to 0x11; BBIDs: all bit combinations from 0 to 7.	Observe the IDs within the frame headers for every ID and CCC under test, in every configuration to ensure correctness.	

65	Frame header status bits X_CCC_input, Y_CCC_input checks.	Set for all independent correlations; set for all concatenated correlation; set each CCC for X concatenated but Y not concatenated; set each CCC for Y concatenated but X not concatenated.	Observe the X_CCC_input and Y_CCC_input status bits in the frame header of every CCC in every configuration to ensure correctness.	

66	Phase bin bit combinations in the output frame header.	Set various chained and non-chained lag combinations. Set the phase bin on the dump_en line to all bit combinations from 0 to 1024 (B0…B9) and MSBit set/reset, on all stream inputs, one at a time, sequentially. If possible, set B10…B14 to ensure full coverage, although the test vector generator may have to be modified to exercise these bits. Test both X and Y inputs, all input stream dumps.	Observe the phase bin in the output header and ensure that the bit combinations are covered. Normal LTA operation, with sequential phase bins generated by the test vector generator should cover all bit combinations in many of the above tests.	Can observe operation in many previous tests to ensure coverage as well.

67	CCC# in output frame header	Enable 1 CCC at a time to correlate.	Observe the output frame header to ensure that the CCC# is correct.	Can observe operation in many previous tests to ensure coverage as well.

68	CMD in output frame header	Enable 1 CCC at a time to correlate. Set for dump and integrate, single dump, and speed dump. Test both X and Y inputs, all input stream dumps.	Observe correct operation with the LTA when dump and integrate, and single dump. Observe that CMD = 011 when speed dump mode (LTA correct operation will also indicate this…)	Can observe operation in many previous tests to ensure coverage as well.

69	RECIRC_BLK-X in output frame header.	Set each CCC for individual correlation, one at a time, connected to all input streams; set CCQs for CCCs concatenated connected to all input streams; set for concatenation of all CCCs connected to streams 0,1,6,7. Set Recirc X chip for "static recirculation" so that the recirculation block bit combinations, identical to SID bit combinations of Test #64,	Observe the RECIRC_BLK-X bits in the output header to ensure that it is correct according to the Recirc X static recirculation configuration.	

		are tested.		

70	RECIRC_BLK-Y in output frame header.	Same setup as Test #69, only it is for Y.	Same as Test #69, only it is for Y.	

71	TIMESTAMP in output frame header.	Set Corr Chip for correlation as in Test #69. Set TIMESTAMP-0 and TIMESTAMP-1 to go through all possible bit combinations: the test vector generator rapidly increments the 1 second time thereby exercising TIMESTAMP-0; TIMESTAMP-1 can be exercised by increasing the TIMECODE period, with various associated dump periods. Test both X and Y inputs, all streams.	Ensure all TIMESTAMP-0 and TIMESTAMP-1 bits are exercised by observing the output frame contents in speed dump mode. TIMESTAMP-0 should be counting up, and TIMESTAMP-1 depends on the TIMECODE period and dump period.	Can observe operation in many previous tests to ensure coverage as well. However, this test specifically targets TIMESTAMP and careful attention must be paid to ensure TIMESTAMP bit combinations have been exercised.

72	Correlation test with no blanking time around the Dump Trigger.	(With test vectors) set the TIMECODE period and DUMPTRIG period so that data valid is not blanked by test vectors when a Dump Trigger occurs. Set all CCCs for any chaining, and all correlating test vector streams 6 and 7. Corr Chip case temperature 100 C with 1 temperature cycle to -35 C with cold spray.	Compare output frames with testvector streams 6 and 7 behavioural simulation in any chaining configuration set. A bit exact comparison cannot be done, but a statistical comparison should reveal that there are no biases or offsets present.	All previous tests are structured with test vectors so that data valid is off before and after Dump Trigger. This factors out pipeline delay mismatches in the CMAM between the actual correlator chip and the behavioural simulator so that bit-exact comparisons (to within double precision numerical error) can be performed. This test ensures that there is no pathological case that could cause problems when data valid is not blanked around a Dump Trigger by the incoming data stream.

73	Walking 1's test through switch configuration registers.	Walk '1' through all switch configuration registers. All other bits zero.	Register writes and reads must be consistent, chip may or may not produce frames. Frame examination not important.	

74	Ad-hoc CCC configuration, sample rate and integration time tests, all corr chips active on the board.	Spend 3 days setting the chip for various and mixed CCC configurations and sample rates on input streams, with all chips active. Perform two tests with X=10 m cable Y=1 m cable, and vice versa.	Ensure correct frame content and bit-exact comparisons for each setup. Note setup for each test. For the cable test, ensure proper schid_frame sync in the chip.	This is a random configuration/user test, and tests cable-length mismatch synchronization.

75	DVCOUNT and DATA_BIAS test, range of integration times. All CCCs active, correlate any bit streams, 256 Ms/s	Set all CCCs for range individual and concatenated correlations, all correlating testvectors 6,7 @ 256 Ms/s. Set range of integration times from 20 usec to 600 usec, in 20 usec steps.	Check to ensure that DVCOUNTs and DATA_BIAS values are consistent across all CCCs for each integration time. Compare with behavioural sim--bit exact comparison indicates that DATA_BIAS values are correct. Bit exact comparisons with a range of CCC concatenation modes indicates the correctness of both the center and edge DVCOUNTS.	Coverage for this functionality is provided by many other tests, however this test checks a range of integration times.

76	DUMP_EN holdoff test, X, Y.	Set the Recirc FPGA for recirculation mode. Generate dumps for single-dump save. Chain all lags together. Test both X and Y inputs, all streams active, one at a time.	DATA_BIAS and DVCOUNTS must indicate that the holdoff function is working for the sample rate, and recirculation block size. Recirculation block X and Y in the output header should be consistent with the recirculation mode selected.	The Recirc FPGAs will operate in recirculation mode, but the test vector generator will not send periodic RRC "Reset Recirculation Counter" commands.

80	Full board, all chips active correlation test.	Setup all correlator chips the same as Test #6. Feed the same test vectors to all chips. Set Backend integration time to 100 seconds. Corr Chip case temperature 100 C with 2 temperature cycles to -35 C with cold spray. Measure output X and Y clock jitter from last chips in row or column. Record values.	Direct frame header comparisons for each CCC for every Corr Chip must indicate exact matches. Backend integrated data for each CCC for every Corr Chip must indicate no differences. Jitter should be <150 ps pk-pk, and top row/column LTAs should operate properly.	This test ensures that the chip functions in every location in its intended environment.

81	Power dissipation test; only 1 chip needs to be active (measuring with more chips active may yield better accuracy in the measurement).	Setup all Corr Chips the same as Test #6, except all CCCs are correlating test vectors 6,7, which have the max phase rates. Set for dump period of ~250 usec, all CCCs. Adjacent Corr Chips must be installed on the board, but must not be active (no power, PLLs reset). Set test vectors for no data valid blanking (probably needs a modification of the test vector generator FPGA to do this). Corr Chip case temperature 50 C.	Measure and record the chip power by measuring the difference in the total board -48 VDC current with the chip active and the chip not active (power to chip off, PLL reset). Must account for the efficiency of the -48 VDC power supply, and also measure the -48 VDC voltage into the power supply. This test rolls in the power dissipation of the Accel chip as well. It is preferred to measure the current into the Corr Chip core and I/O directly, but this may be difficult to do.	This test should indicate Corr Chip and Accel chip power dissipation is less than ~5 W. Performing this measurement with more chips active may yield more accurate results.

5 References

[1] Smegal, R., TEST AND VERIFICATION PLAN: EVLA Correlator Chip Functional Test Cases, TVP document: A25082N0002, Revision 1.0, January 30, 2006.

[2] Smegal, R., TEST AND VERIFICATION REPORT: EVLA Correlator Chip Functional Test Results, TVR document: A25082N0003, Revision 1.0, January 30, 2006.

[3] Carlson, B., TEST AND VERIFICATION REPORT: EVLA Correlator Chip Gate-level Verification Report, TVR document A25082N0004, Revision 1.0, December 22, 2005.

[4] Carlson, B. REQUIREMENTS AND FUNCTIONAL SPECIFICATION: EVLA Correlator Chip, A25082N0000, Revision 2.4, February 15, 2005.

[5] Carlson, B., TEST AND VERIFICATION PLAN: EVLA Correlator Chip, TVP document A25082N0001, Revision DRAFT, August 26, 2004.

[6] Carlson, B., Application Note: Timecode Gbit Test FPGAs for EVLA Correlator Testing, Version DRAFT, May 5, 2005.

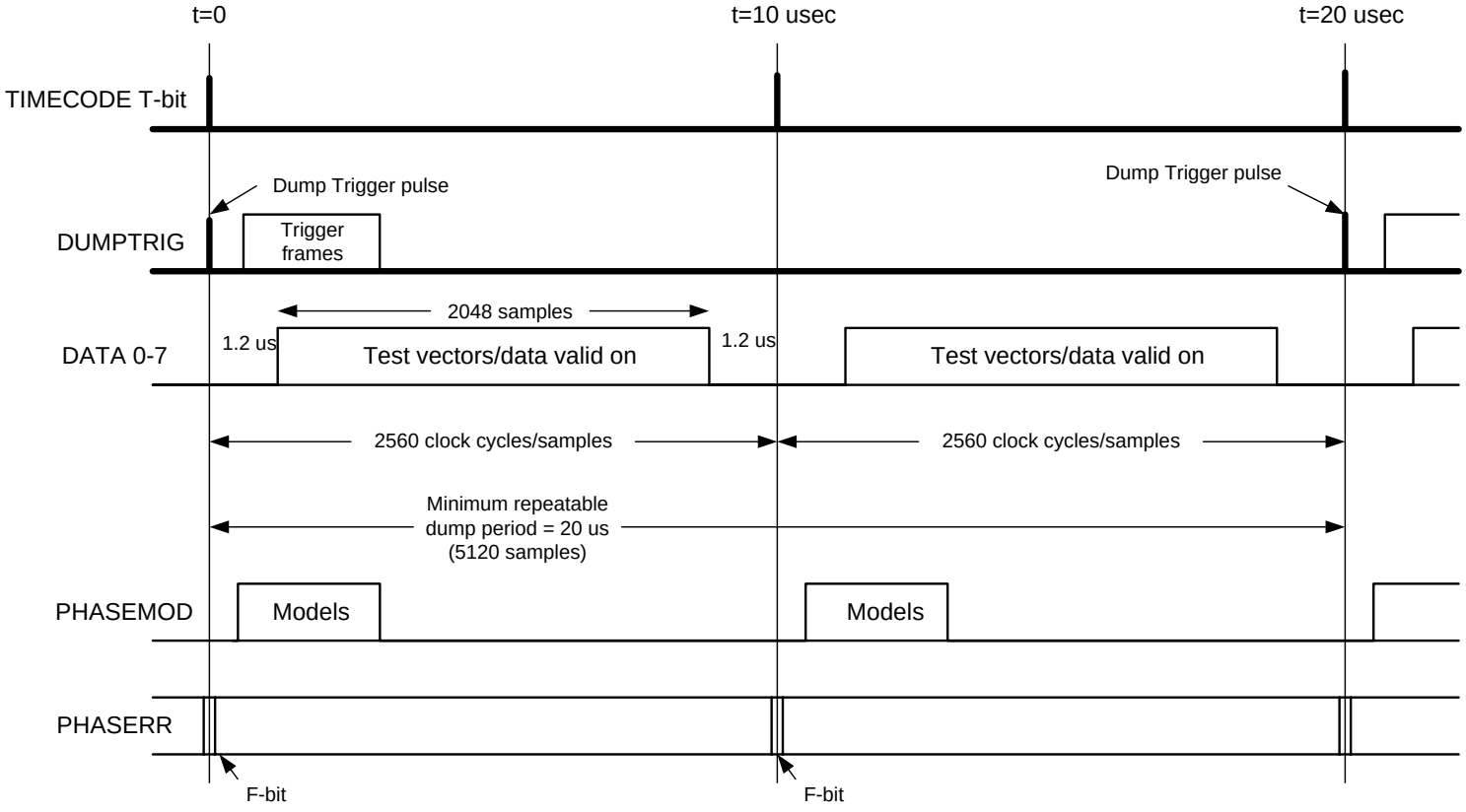
[7] Carlson, B., Additional Tests of Correlator Chip Functionality as Required by the Correlator Chip Critical Design Review, NRC-EVLA Memo# 023, March 14, 2005.

[8] Carlson, B., TEST AND VERIFICATION PLAN: EVLA Baseline Board Prototypes, TVP document A25081N0001, Revision DRAFT, March 17, 2005.

6 Appendix I: Test Vector Structure

The test vectors for correlator chip prototype verification will be generated by an FPGA on the Timecode Board prototype. This FPGA has been programmed to generate a canned sequence of test vectors that stimulate the Recirculation Controller FPGAs on the Baseline Board that, in turn, stimulate a row or column of correlator chips, with the correct protocol. Thus, testing of the correlator chip relies on proper functioning of the Recirculation Controller FPGA. The test vector sequence has been built to facilitate testing the complete range of correlator chip functions with minimal control requirements, and to allow bit-exact[1] comparison of the results from the physical prototype chip with the correlator 'C' behavioural/reference simulator[2]. A complete discussion of the functions and register set of this FPGA is defined in section 7. This section will focus on describing the structure, capability, and limitations of this test vector sequence.

Figure 6-1 is a time-sequence diagram of the test vectors that the Timecode Board FPGA generates, in a typical configuration.



[1] Actually it is not quite bit-exact; it is only to double-precision numerical error precision, or 15 significant figures, since comparisons are made after dividing by the data valid count.

[2] The test vector files for the 'C' behavioural simulator exist now and are available for use.

Figure 6-1 Timecode Board transmit FPGA default configuration test vector sequence.

In the above configuration, the FPGA generates a Timecode T-bit once every 10 usec. In this case, the T-bit is asserted (1) every 10 ticks, although this has no bearing on the vectors that eventually find their way to the correlator chip. The DATA streams contain the test vector sequence, with each stream containing a unique set of vectors with unique phase rotation and spectral signature. The sequence repeats every time there is a T-bit:

- T-bit is asserted.
- 1.2 usec (~300 samples) later the test vectors contain valid data. This blanking time has been chosen so that the actual correlator chip blanking time of 1 usec is factored out of the results.
- 2048 samples later data valid is pulled low, and remains low until 1.2 usec after the next T-bit.
- The final trailing 1.2 usec blanking time has been chosen so that any pipeline delays in the correlator chip prototype are factored out and do not have to be exactly modeled in the 'C' behavioural/reference simulator.

The default DUMPTRIG period is 20 usec, and this is chosen so that no matter what the lag-chain length, an integration of an arbitrary number of dumps will, once coefficients are normalized by dividing by the data valid count, yield the same results. This is because in 5120 clock cycles, the samples will traverse down a 2048 lag chip (1024 shift register elements) exactly 5 times (i.e. 5120/1024=5). Thus, on every dump, the correlator chip—even with shorter lag lengths—is correlating exactly the same data. The result of this arrangement is that by setting different DUMPTRIG and integration configurations in the test vector generator FPGA, integration can be performed for an arbitrarily long period of time and it will produce the same deterministic result. Thus, it will be easy to perform a bit-exact comparison with the 'C' behavioural simulator even on long integrations in the physical prototype that could pick up minute glitches or defects.

Additionally, DUMPTRIG signaling for each DATA stream is different so that it is possible to test/verify routing of dump control signals in the correlator chip in a deterministic way. DUMPTRIG timing for each of the DATA streams is such that:

- DATA-0 DUMPTRIG has the shortest period, set by the configuration registers in the MCSR, and is thus the reference.
- DATA-1…DATA-7 DUMPTRIG period is longer by a factor of *data stream number + 1*. For example, DATA-1 has a DUMPTRIG period that is 2X longer than DATA-0, and DATA-7 has a DUMPTRIG period that is 8X longer than DATA-0. What this means for the correlator chip is that DUMP_SYNC is set at the rate defined by DATA-0 DUMPTRIG timing, and that the corresponding DUMP_EN lines for DATA-1…DATA-7 are asserted at increasingly lower rates.

An important additional note is that the transmit FPGA's internal "dump frame counter" is 16 bits, and that once it rolls over to zero, the phase bin gets reset, and "dump clears" (dump command "000") are generated for all streams. For example, if the shortest DUMPTRIG period is 20 usec, this counter will roll over in 20 usec x 2^{16} = 1.31072 seconds.

PHASEMOD is generated for the next tick period, every TIMECODE T-bit. Since the vectors repeat themselves, every generation of PHASEMOD is identical. Note, however, that a separate phase model is generated for each stream.

PHASERR contains whatever the transmit FPGA is told to generate, and this is a repeating pattern. Note that at the TIMECODE T-bit, PHASERR contains a specific "F-bit" pattern as defined by the HM Gbps protocol.

If the FPGA is configured so that the TIMECODE T-bit period is stretched out, then once the 2048 test vectors have passed, data valid is held low until the next TIMECODE T-bit occurs. Thus, with increasing TIMECODE T-bit period, the duty cycle of valid data becomes smaller and smaller. This case is shown in Figure 6-2, where the TIMECODE T-bit period is 20 usec instead of 10 usec. In practice, there is no reason to change the TIMECODE T-bit period from its default of 10 usec.

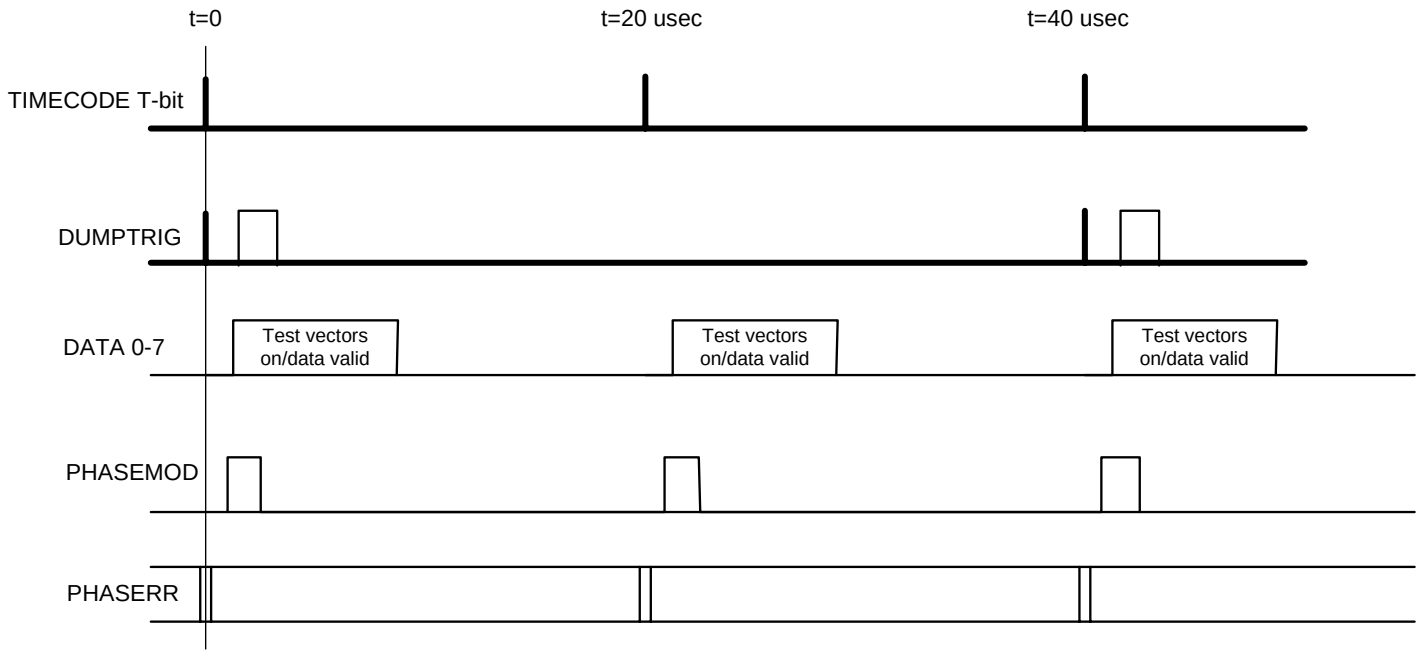


Figure 6-2 Test vector timing diagram for a TIMECODE T-bit period of 20 usec. The test vector on/data valid duty cycle has decreased.

Other points to note with the test vector sequence, and how it impacts correlator chip prototype testing:

- Test vectors can be generated by the transmit FPGA at lower sample rates, but if this is done, then ***every*** DATA stream operates at the lower sample rate. When the sample is a "repeat" in this case, it is set to data invalid to ensure that the possibility of timing or logic slips along its path to the correlator chip is detected. In this case, it is necessary to re-calculate PHASEMOD models for the lower sample rate. Note that the Recirculation Controller FPGA must be programmed to generate the correct sample rate on the correct stream—not doing so could result in, for example, every other sample going to the correlator chip being invalid and this will result in a different spectral signature[3]. If handled correctly this capability can be used to verify that the correlator chip is routing se_clks and data valids as expected.

- Since the same data from the Timecode Board transmit FPGA is routed to the X and Y Recirculation Controller FPGAs, it is necessary to be aware of the spectral signature that are produced when like or un-like streams are correlated. The results can be summarized as follows:

    - DATA0 x DATA1 – Continuum correlation with 1 spectral line.
    - DATA2 x DATA3 – Continuum correlation with 2 spectral lines.
    - DATA4 x DATA5 – Continuum correlation with 3 spectral lines.
    - DATA6 x DATA7 – Continuum correlation with 4 spectral lines.
    - DATAn x DATAn – Autocorrelation. If phase rotation in the correlator chip is enabled, this will result in long swatches where either in-phase or quadrature lags are zero. If phase rotation is not enabled, this is a pure auto-correlation and the quadrature lags will be zero.
    - DATAn x DATAm (where n and m are not one of the above pairs—e.g. DATA1 x DATA2, DATA1 x DATA7 etc.) – No continuum correlation, but with the least number of spectral lines correlating. E.g. for DATA1 x DATA2, 1 spectral line will correlate.

- The test vector generator FPGA does not generate 7-bit samples. 7-bit correlation is not necessary for prototype verification, since it has already been demonstrated to work as designed in the RTL model of the chip [7].

- Since the test vector sequences are so short, DATA6 and DATA7 are the only outputs that contain complete wraps of phase[4] within one 2048-point test vector sequence. Thus, when tests are run to ensure that lags are performing properly,

---

[3] If the sample rate set in the Recirculation Controller FPGA is higher than the sample rate coming from the FPGA test vector generator, then a zero-insertion interpolation is occurring, and this has a very distinct spectral signature (repeats of the spectral signature in the output). If it is the other way around, then it is equivalent to sampling at lower than the Nyquist rate, and this will result in the spectrum piling on top of itself by the difference factor.

[4] i.e. each DATA stream output pair (0,1; 2,3 etc) has a different phase rate.

including hitting all phase states in the phase rotator, DATA6 and DATA7 should be used for this purpose.

- The transmit FPGA can be set to contain different IDs for each DATA stream output. Thus, it is possible to fully exercise the ID capture capability of the correlator chip.
- The transmit FPGA automatically generates sequential phase bin numbers so that phase bin capture and inclusion in the correlator chip output data frame can be exercised. Phase bins, where the bin number is used for frames saved in LTA memory, go as high as 1023, and so exercise the LTA's ability to discard phase bins that are out of range of the 0-999 limit.
- Any X and Y recirculation block numbers must be set in the Recirculation Controller, by putting it into "static" or "flow-through" recirculation mode.
- It is possible to set DUMPTRIG and TIMECODE so that they do not repeat in a harmonically related fashion. In this case, correlation will still be possible, however, bit-exact comparison with the 'C' behavioural simulator may not be possible.
- For **correlator chip power dissipation tests**, it is necessary to compile the transmit FPGA design with continuous test vectors so that the chip is exercised all of the time. In this case it is not possible to perform a bit-exact comparison between the correlator chip data frames and the 'C' behavioural simulator. This will, in practice, be a separate bit-stream configuration file for the FPGA.

7 Appendix II: Transmit FPGA register set description

This section contains a description of the transmit FPGA register set [6]. Some default/reset register settings have been changed from the application note and thus a complete description of this register set is included here.

A block diagram of the Timecode Board and Transmitter FPGA is shown in Figure 7-1.

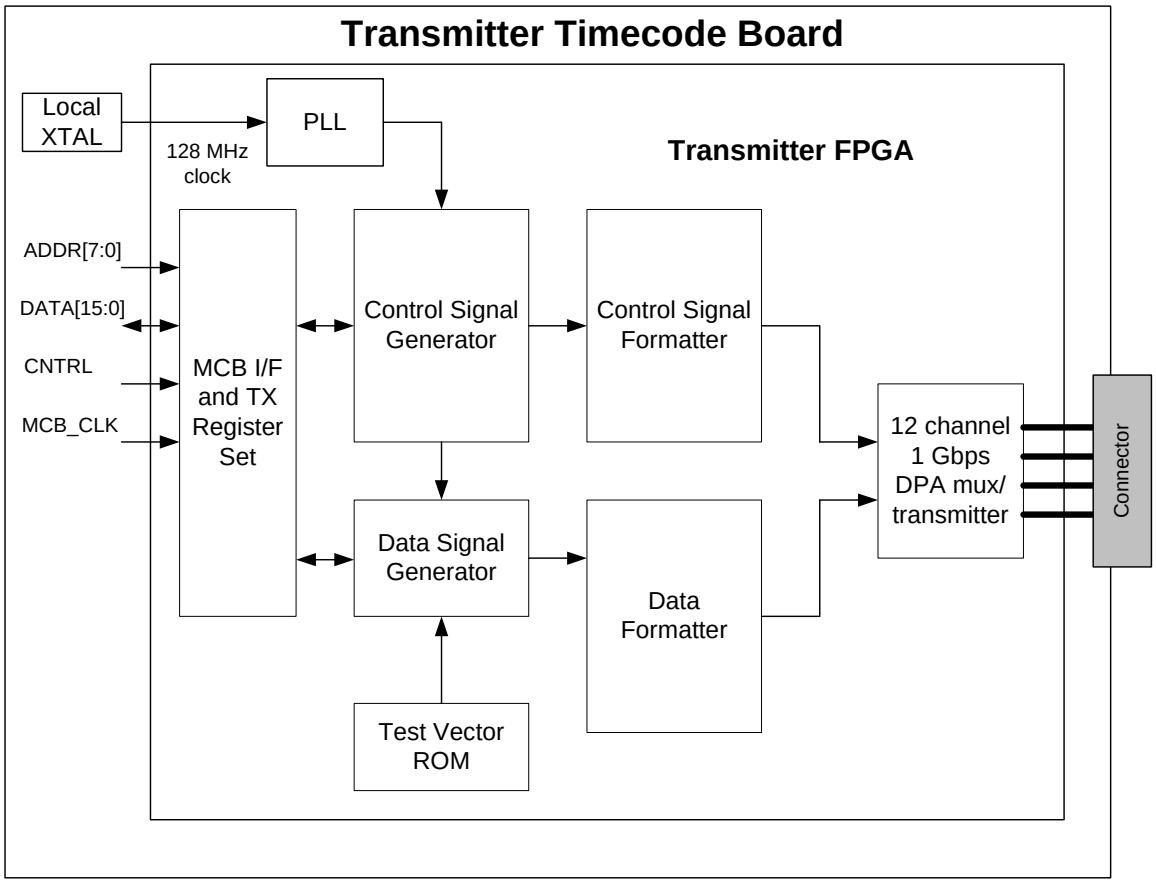


Figure 7-1 Timecode Board and transmit FPGA block diagram.

The FPGA can generate HM Gbps signals on 4 wafers. This is exactly the number of wafers that feed into a Recirculation Controller FPGA. The signals are fed to multiple Recirculation Controllers on the Baseline Board using one or more Fanout Boards [8]. The control signals (TIMECODE, DUMPTRIG, PHASEMOD) on each wafer are identical.

The transmit FPGA contains a number of registers that allow some variance in the character of the signals that get transmitted. Thus, in addition to the basic capability of transmission of control and data signals for the Gbit transmission test, it is possible to change the content and period of signals to facilitate testing of the Baseline Board and the correlator chip for prototype verification testing. This is most easily facilitated if a

simple GUI to write and read these registers are available, however some rudimentary capability is possible with simple command-line read/write access to these registers.

Since data from one transmit FPGA will be fed to both the X and Y inputs of the Baseline Board[5], the test vector ROM and PHASEMOD generators are designed so that it is necessary to cross-correlate unlike streams to fully exercise the correlator chip accumulators and phase rotators. To do this, streams should be correlated in adjacent pairs to produce both continuum and spectral-line correlations. For example, correlating streams 0,1 will produce 1 spectral line and a continuum correlation. Correlating streams 2,3 will produce 2 spectral lines etc. Correlating 0,2 will produce 1 spectral line and no continuum correlation. If like streams are correlated, then this is an autocorrelation—if phase rotation in the correlator chip is enabled, a phase-slope across the lag domain will still result, however, with the unusual and seemingly erroneous signature of alternating stretches of in-phase and quadrature accumulators being zero. This is completely normal and can be reduced to a pure autocorrelation (quadrature accumulators always zero) by disabling phase rotation in the correlator chip.

The test vector ROM is 4096 samples long. However, the first ~300 samples are set to data invalid (5), the next 2048 samples contain valid test vector data[6] with the occasional data invalid, and the remaining samples are also invalid. The FPGA repeats this data pattern, initializing the ROM address to zero[7] with every TIMECODE tick and generating a new data point every valid SE_CLK (see the SE_CLK_MOD register). When the ROM address reaches its maximum value, it stays there generating data invalids until the next TIMECODE tick comes along to reset the address.

With the TIMECODE tick period and DUMPTRIG period set to minimum values (or the reset values of 10 usec and 500 usec respectively), it is thus possible to perform a bit-exact comparison of the correlator chip correlation with the ‘C’ behavioural simulator since data valid is always blanked before, during, and after Dump Trigger events. (Note that it is possible to set the DUMPTRIG period so that this condition is violated using the Period Register.)

[5] It is not possible to use two different Timecode boards to generate independent data since there is no way to synchronize their clocks or ticks, and this will cause input FIFO overruns on the correlator chip.

[6] Generated by the same noise generator used for correlator chip design validation and verification. Thus, it has the same spectral signature. Only 4-bit samples are supported—no 7-bit data is available in this design.

[7] Synchronization guarantees that the data point at address 0 of the ROM is always generated at the same time as the first-half of the TIMECODE T bit (even though it is invalid data).

7.1 Master Control/Status Register (MCSR) R/W Addr=0x00 Reset=0x7F

7	6	5	4	3	2	1	0
DM-1	DM-0	ETG	DST	Dm-2	Dm-1	Dm-0	ETG

ETG (R/W) – Enable Tick Generation. If set (1) then a TIMECODE signal is generated. If reset (0) no TIMECODE signal is generated. Normally this should always be set.

Dm[2:0] (R/W) – Dump modulo. Dump period % $2^{DM[2:0]}$ indicates how often a non-"Dump Clear" command is sent on DUMPTRIG, ***only*** when in "speed dump" mode or when in "normal dump/accumulate" mode where an LTA bin contains the accumulation of only one frame (dump command "101"—"burst dumping"). These bits can be used to slow down how often correlator chip data is dumped. For example, if this is "2" (010), then a non-dump clear DUMPTRIG is sent every 4th time (for DATA-0; 8th time for DATA-1; 12th time for DATA-2 etc), and therefore data is dumped out of the correlator chip at ¼ the rate it normally would be.

DST (R/W) – DUMPTRIG Sync Test Frame Enable. If set (1), this enables the generation of "synchronization test frame" signaling in DUMPTRIG so that the receiver can properly lock DUMPTRIG to TIMECODE. If reset (0), no synchronization test frames are generated. Synchronization test frames are used to ensure that the Recirculation Controller properly synchronizes the timing of the DUMPTRIG "Dump Trigger" to TIMECODE. Normally this should always be set.

ETG (R/W) – Enable DUMPTRIG Generation. If set (1) then DUMPTRIG signals are generated. If reset (0), no DUMPTRIG signals are generated (i.e. the DUMPTRIG line alternates 101010…). Resetting and then setting this bit forces the transmit FPGA's internal DUMPTRIG sequence generator to start over.

DM[1:0] (R/W) – Dump mode. 00=speed frames; 01=normal dump/accumulate commands. In 01 mode, the number of frames that are accumulated into the LTA is controlled by the DTACCR register. In this case it is possible to exercise LTA accumulation functions on the Baseline Board.

7.2 Period Register (PEREG) R/W Addr=0x01 Reset=0x10

The Period Register allows some control over the TIMECODE tick period and DUMPTRIG period. The default, if this register is 0x01, is that TIMECODE is generated once every 10 usec, and DUMPTRIG is generated every 60 usec for DATA-0, and every 60 usec x 8 = 480 usec for DATA-7

Bits [2:0] are the TIMECODE tick period multiplier bits.

Bits [7:3] are the DUMPTRIG period multiplier bits.

For example, if Bits [2:0] are 110b (6), then the TIMECODE tick period is 10 usec x (6+1)=70 usec. If Bits [7:3] are 11000b (24), then the DUMPTRIG period is 20 usec x (24+1)=500 usec.

7.3 DUMPTRIG Accumulate Control Register (DTACCR) (R/W) Addr=0x02 Reset=0x00

This register controls how many frames are accumulated into the LTA when the dump mode (DM bits in the MCSR) is 01. If 1, then burst dumping occurs ***on all streams*** whereby frames are dumped, saved into the LTA, and then flagged as ready. If 0, then 255 frames are accumulated. In both speed dump and normal dump/accumulate modes, the FPGA generates sequential phase bins that eventually fully exercise all of the phase bin numbers in both banks and even cause dumping to illegal phase bins >999 and <1024, that should be detected by the LTA. The number of frames that are accumulated in the LTA for a particular DATA stream is multiplied by the *DATA stream # + 1* factor, if the contents of this register is not 1. For example, if this register is set to 53, then for DATA-7, the number of frames accumulated in the LTA is 53 x 8 = 424. Note that this design does not support pulsar phase binning, nor does it support DUMPTRIG generation for recirculation.

Observation: in the reset/default configuration for the DATA-0 stream, DUMPTRIG is generated every 60 usec, normal dump accumulate mode is used and 2 frames are accumulated in each LTA bin. For DATA-1, dumps occur every 120 usec and 4 frames are accumulated in each LTA bin. For DATA-7, dumps occur every 480 usec, and 16 frames are accumulated in each LTA bin.

7.4 CRC Error Generation Register (CRCERRGEN) R/W Addr=0x03 Reset=0x0000

This register contains 16 bits that allow the controlling microprocessor to force CRC errors on individual bit streams to test receiver CRC error detection capability. If bits in this register are set (1), then the CRC code that is transmitted is inverted. Note that there is no synchronization between when bits in this register might be set, and when the errored CRC code is generated, thus, when bits are set, they must be held for a sufficient period of time to ensure an error is generated.

Bit assignments to bit streams in this register are as follows:

B0-TIMECODE
B1-DUMPTRIG
B2-PHASERR0_1
B3-PHASERR2_3
B4-PHASERR4_5
B5-PHASERR6_7
B6-PHASEMOD
B7-none
B8…B15-DATA-0…DATA-7

7.5 Shift Enable Clock Modulo Register (SECMR) R/W Addr=0x07 Reset=0x00

This register sets the shift-enable clock rate for the data being transmitted on the DATA outputs. The clock rate is set according to the following table:

Bits (3 2 1 0)	Clock divide factor	Sample rate (Ms/s)
0000	1	256
0001	2	128
0010	4	64
0011	8	32
0100	16	16
0101	32	8
0110	64	4
0111	128	2
1000	256	1
1001	512	0.5
1010	1024	0.25
1011	2048	0.125
1100	4096	0.0625

The same sample rate is set on all output DATA streams. DATA is taken from the on-board ROM and transmitted whenever there is a valid shift clock. When the shift clock is low indicating it is not valid, the DATA value is set to an invalid "5". For example, if the content of this register is 1 (0001b), then the shift clock is 128 MHz and every other sample is invalid. The effect of this is to extend the absolute duration that data is present on the output by a factor of 2. To ensure that bit-exact comparisons can be done with the C simulator, the TIMECODE and DUMPTRIG periods in this example must be increased by a factor of 2, and the PHASEMOD phase rate must be reduced by a factor of 2 since the phase generators on the Baseline Board always operate at a constant 64 MHz clock rate.

7.6 DATA Stream IDs Registers 0-7 (DSIR-0…DSIR-7) R/W Addr=0x10…0x17

These 8, 16-bit registers are used to allow the embedded IDs in the DATA streams to be set to desired values. The breakdown of the register bits is as follows:

B[7:0]—8-bit SID

B[12:8]—5-bit SBID

B[15:13]—3-bit BBID

Reset values, in decimal numbers, are according to the following table:

DATA Stream	SID	SBID	BBID
0	105	3	0
1	115	5	1
2	125	7	2
3	135	9	3
4	145	11	4
5	155	13	5
6	165	15	6
7	175	17	7

7.7 PHASEMOD Coefficient Registers 0-7 (PMCR-0…PMCR-7) R/W Addr=0x20…0x3f

These 16-bit wide registers, implemented as DPSRAM on the chip, store the PHASEMOD coefficients that are sent every TIMECODE time tick on the PHASEMOD stream. The on-chip DPSRAM that contains these coefficients has been initialized to the coefficients required to perform phase rotation (i.e. stop fringes) on the test vector data in the 4096 on-chip ROM when the shift enable clock modulo register (SECMR) is reset (0) (i.e. the highest speed of 256 Mbits/sec/stream). If the SECMR register is changed for a different clock rate, then the phase rate (P1) coefficients must be changed accordingly for fringes to be properly stopped.

Addressing and initial values of the PMR register DPSRAM is according to the following table. For example, "D0-P0-MSW" is the most-significant word (16 bits) of the initial phase (P0) for the DATA-0 stream output.

Addr	Coefficient	Init Value	Addr	Coefficient	Init Value
0x20	D0-P0-MSW	0x0302	0x30	D4-P0-MSW	0x0864
0x21	D0-P0-LSW	0x6EFB	0x31	D4-P0-LSW	0x6C1B
0x22	D0-P1-MSW	0x0025	0x32	D4-P1-MSW	0x0068
0x23	D0-P1-LSW	0x9506	0x33	D4-P1-LSW	0xCD14
0x24	D1-P0-MSW	0x0D41	0x34	D5-P0-MSW	0x26CA
0x25	D1-P0-LSW	0x6D36	0x35	D5-P0-LSW	0x9FCF
0x26	D1-P1-MSW	0x00A5	0x36	D5-P1-MSW	0x01E4
0x27	D1-P1-LSW	0x8873	0x37	D5-P1-LSW	0x6BB2
0x28	D2-P0-MSW	0x05B3	0x38	D6-P0-MSW	0x0B15
0x29	D2-P0-LSW	0x6D8B	0x39	D6-P0-LSW	0x6AAB
0x2A	D2-P1-MSW	0x0047	0x3A	D6-P1-MSW	0x008A
0x2B	D2-P1-LSW	0x310D	0x3B	D6-P1-LSW	0x691B
0x2C	D3-P0-MSW	0x1A06	0x3C	D7-P0-MSW	0x338F
0x2D	D3-P0-LSW	0x0683	0x3D	D7-P0-LSW	0x391C
0x2E	D3-P1-MSW	0x0144	0x3E	D7-P1-MSW	0x0283
0x2F	D3-P1-LSW	0xFA12	0x3F	D7-P1-LSW	0xDD52

7.8 PHASERR Coefficient Register (PECR) R/W Addr=0x40 Reset=0x0000

This 16-bit register contains coefficients that are transmitted on all PHASERR output streams. Bit assignments in this register are as follows, where 'PHASERR_m' is the phase for streams 0, 2, 4, 6, and 'PHASERR_n' is the phase for streams 1, 3, 5, 7:

B[7:0]-PHASERR_m

B[15:8]-PHASERR_n

8 Appendix III: Transmit FPGA GUI panels

This appendix contains a basic description of the GUI panels that will enable driving the transmit FPGA on the Timecode Board. The layout may be slightly different than the actual implementation, although the functionality will be the same.

8.1 Timing and Dump Control

This GUI panel is shown in Figure 8-1. It allows for control of TIMECODE and DUMPTRIG via the FPGA control registers.

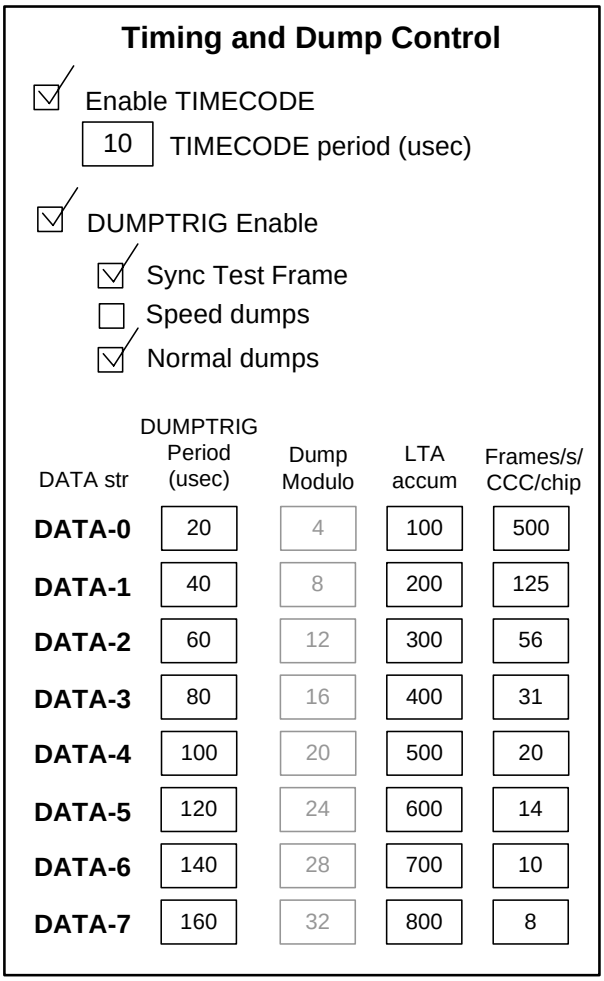


Figure 8-1 Timing and Dump Control GUI panel for the transmit FPGA.

A description of each element in the GUI follows:

- **Enable TIMECODE**. If checked on, then TIMECODE is generated. If not, then TIMECODE is not generated (in this case TIMECODE is repeating '1010' pattern).

- **TIMECODE period**. Sets the TIMECODE period in usec. The default of 10 usec is shown. There are restrictions on what the TIMECODE period can be—refer to the register set description.

- **DUMPTRIG Enable**. If checked on, then DUMPTRIG generation is enabled. If not, then DUMPTRIG is not generated.

- **Sync Test Frame**. If checked, then the generation of DUMPTRIG synchronization test frames is enabled. If this is the case, then the next Dump Trigger must occur coincident with the Timecode T-bit, or the Recirculation FPGA receiver will report an error.

- **Speed dumps.** If checked, then "speed dumps" will be generated by the correlator chip, whereby frames bypass the LTA RAM and are immediately routed to the output.

- **Normal dumps.** If checked, then normal dump/accumulate dumps are generated, whereby correlator chip data frames are accumulated into the LTA for a specified number of times.

Note: Since each stream has a DUMPTRIG associated with it, and the period that these DUMPTRIGs gets generated depends on the DATA stream number, the following parameters are specific to each stream number, but are always related (refer to sections 7.1 to 7.3.) DATA str 0 is the minimum dump period stream, and dump periods for all other streams are factors of this minimum dump period.

- **DUMPTRIG Period**. DUMPTRIG period in usec for the particular stream. Normally this is set to be some integer multiple of the TIMECODE period if it is desired to perform bit-exact comparisons of the correlator chip output with the 'C' behavioural simulator output. This value can be set to a maximum of 640 usec. Changing any stream's period, must re-calculate and reflect the change in all other streams' periods.

- **Dump Modulo**. This value sets how often a generated DUMPTRIG actually causes the correlator chip to dump data and generate one or more frames to the LTA Controller FPGA, when **LTA accum** is 1, or when in speed dump mode. For example, if this value is 4, then every 4^{th} dump actually causes frames to be generated; the intervening 3 dumps are "dump discard" that clear the accumulators but don't generate data frames. This value ranges from 2^0 to 2^7. This feature, along with the DUMPTRIG period, can throttle the correlator chip frame rate output over a considerable range, when burst dumping or speed dumping.

- **LTA accum**. This value sets how many correlator chip data frames are accumulated in the LTA, for the particular stream. If any one of these is set to 1, then they are all 1, and Dump Modulo takes effect.

- **Frames/s/CCC/chip**. This is a calculated value that indicates how many frames per second per correlator chip CCC exit the LTA to the backend, given the DUMPTRIG period and LTA accumulation, if the particular stream is correlated by a CCC. The total number of frames exiting the LTA depends on the setting of the Recirculation Controller FPGA and the correlator chip and therefore cannot be calculated here. However, this GUI function makes the total calculation simpler for the user. This number is calculated as 1/(period x LTA accum). If Dump Modulo is active, this number is calculated as 1/(period x Dump Modulo).

8.2 Data Stream Control

This panel is used to control the information content of output DATA streams as well as other miscellaneous items. This panel is shown in Figure 8-2.

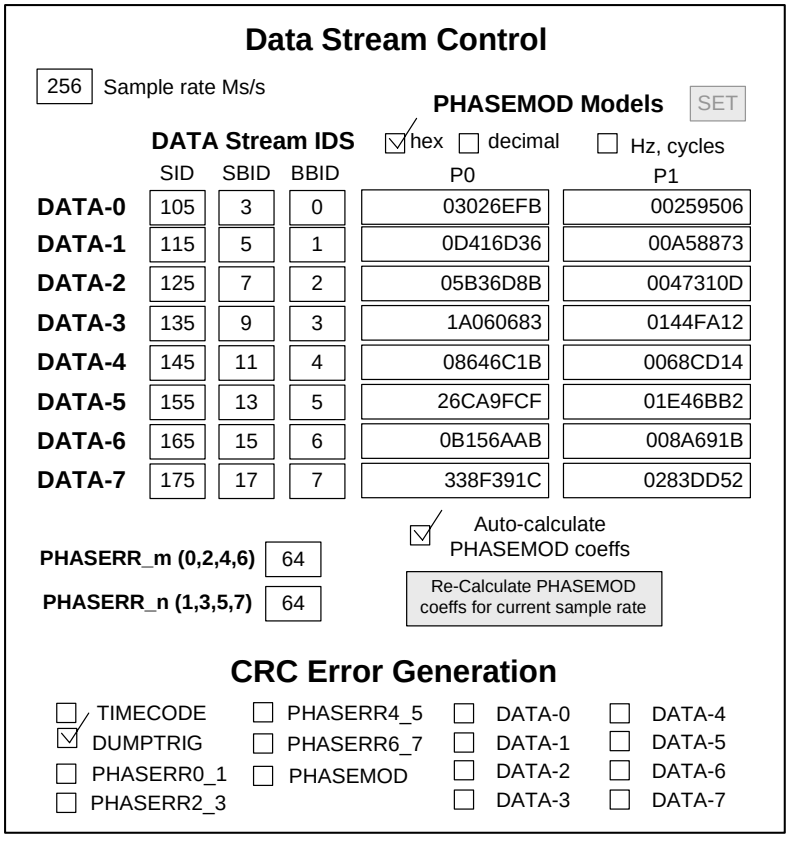


Figure 8-2 Data Stream control GUI panel.

The Data Stream Control GUI panel is used to control the content of DATA streams, PHASEMOD, the PHASERR streams, and to be able to set, on a stream-by-stream basis, CRC error generation. A description of each GUI panel element is as follows:

- **Sample rate Ms/s.** The DATA stream sample rate in Ms/s. This is the same for all DATA streams, and can range from 256 to 0.0625.

- **DATA Stream IDS.** This part of the panel allows the SID, SBID, and BBIB for each stream to be set independently.

- **PHASEMOD Models.** There are several components to this part of the panel which allows PHASEMOD models to be set. These components are as follows:

    - **SET –** This button is only enabled when the "Auto-calculate PHASEMOD coeffs" box is not checked. When pressed, it takes the P0 and P1 models and writes them into the PMCR registers. This allows the user the ability to tweak the models to see the effects on correlated data. Note that since the same phase models are generated every Timecode tick, changing P1 slightly will not result in a residual phase rate out of the correlator.

    - **hex –** Displays the coefficients in hexadecimal.

    - **decimal –** Displays the coefficients in decimal (normal number) format.

    - **Hz, cycles –** If checked, then phase (P0) is a number from 0.0 to 1.0-LSBit representing one cycle of phase, and phase rate (P1) is in cycles/sec. If not checked, phase (P0) is in actual register contents, and phase rate (P1) is in actual register contents.

    - **Auto-calculate PHASEMOD coeffs** – If checked, the P0 and P1 coefficients are automatically calculated and stored in the PMCR registers when the Sample rate is changed.

    - **Re**-calculate PHASEMOD coeffs for current sample rate – Pressing this button results in recalculation (and saving in the PMCR registers) of the PHASEMOD P0 and P1 coefficients for the current sample rate.

- **PHASERR_m, PHASERR_n.** Sets and stores the value stored in the PECR register anytime these are changed.

- **CRC Error Generation.** Checked boxes turn bits on in the CRCERRGEN register, resulting in the generation of CRC errors on checked data streams.

9 Appendix IV: Correlator Backend (CBE) Requirements

Correlator Chip lag frames, whether integrated in the LTA or not, ultimately end up in a CBE computer for further processing. Initially, the LTA RAM read feature is likely used to capture integrated lag frames to ensure that LTA RAM contents are correct. However, eventually the CBE is required to allow access to raw lag frames or integrated lag sets to detect intermittent errors that could indicate Correlator Chip timing problems.

The following is a list of CBE requirements to support Correlator Chip prototype testing.

1. Display of **Real and Imaginary vs. lags** for a lag set, after CBE integration. These are real-time 2-D displays that are refreshed every time CBE integration is complete. This display capability may already be rolled into the RTDD (Real-Time Data Display—A25280N0000), and thus if the RTDD is ready in time, may not be required as a separate item. The RTDD is superior in that it has an infinite persistence display capability.

2. Display of **Amplitude vs. lags** for a lag set, after CBE integration. If available, the RTDD may eliminate the need for this display capability in the CBE.

3. Display of **Amplitude and phase vs. frequency** for a lag set, after CBE integration. If available, the RTDD may eliminate the need for this display capability in the CBE.

4. Saving of **raw lag frames to ASCII files** before CBE integration. Each CCC's lag frames are saved to a separate file, to allow for direct comparison of files with each other or other chips. The format for a "Correlator Chip Output Data Frame" is as indicated in Figure 9-1, noting that slightly different formats are required for Correlator Chip lag frames (available with the "speed dump" capability of the Correlator Chip and the LTA) and LTA lag frames (refer to LTA RFS A25091N0000 Figure 5-1 and Figure 5-3). The first column is the 32-bit word in ASCII hex, and the rest of each line, 5 spaces to the right, is the decoded printout of the contents of the word, in human-readable form, the *exact* format of which is not formally defined, but should generally follow the form shown in Figure 9-1

5. Saving of **lag frame sets to ASCII files after CBE integration**. Here, the Real and Imaginary lag values for a complete lag set, in double-precision floating-point ASCII form[8], are saved to a file. Typically the CBE integration time is very long, and only one lag set is saved to each file. Comparison of files that are generated with the same repeating test vectors, and contain the same lag set, will likely be able to detect even very subtle and intermittent problems. Additionally, and most importantly, these files can be directly compared via an external plotting/comparison program (i.e. that already exists in MathCad), with files

[8] Converted to floating-point form be dividing integer lag frame values by the center lag data valid count.

generated by the reference software correlator. The first few lines of the format are shown in Figure 9-2. Each line is one lag; the left value is the Real component, and the right value is the Imaginary component. Lag numbering is in ascending order, starting with lag 0. Each number contains 16 significant figures and an exponent.

```
aaaaaaaa     W0: START SYNC WORD -- OK
e001006a     W1: B31=ASIC[1] Yin=1 Xin=1 YSyner=0 XSyner=0 ACC_OV=0 OVR=0 Rsrv=000000 NUM_CLAGS= 128 CCC=13 Cmmd=010
076423c8     W2: BBID-Y=0 SBID-Y= 7 SID-Y=100 BBID-X=1 SBID-X= 3 SID-X=200
d972e900     W3: LTA/Phase bin=55666 Recirc_blk-Y=233 Recirc_blk-X=  0
0d408a04     W4: TIMESTAMP-0= 222333444
211ecd59     W5: TIMESTAMP-1= 555666777
00000869     W6: DVCOUNT-Cntr=      2153
00000847     W7: DVCOUNT-Edge=      2119
0003a300     W8: DATA_BIAS   =    238336
0003a545     Lag    0 =     238917
0003a3e6     Lag    0 =     238566
0003a5f8     Lag    1 =     239096
0003a3d8     Lag    1 =     238552
0003a370     Lag    2 =     238448
0003a1fa     Lag    2 =     238074
0003a3a6     Lag    3 =     238502
0003a078     Lag    3 =     237688
0003a2af     Lag    4 =     238255
0003a1d2     Lag    4 =     238034
0003a2ce     Lag    5 =     238286
0003a268     Lag    5 =     238184
0003a3b6     Lag    6 =     238518
0003a256     Lag    6 =     238166
0003a1fc     Lag    7 =     238076
0003a1fa     Lag    7 =     238074
.
.
.
0003a20a     Lag  125 =     238090
0003a3fe     Lag  125 =     238590
0003a6cc     Lag  126 =     239308
0003a57c     Lag  126 =     238972
0003a550     Lag  127 =     238928
0003a418     Lag  127 =     238616
1c71c71c     W265: END SYNC WORD -- OK
b9fa8535     W266: Checksum calculated OK
```

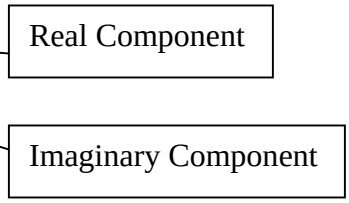


Figure 9-1 Raw lag frame ASCII file example.

```
1.675218060543869e-01    -1.723961005643920e-01
3.732683427398666e-02    -1.918932786044125e-01
-5.438686505900462e-02   -1.096716264751154e-01
1.282709081580298e-03     5.246280143663417e-02
.
.
.
```

Figure 9-2 ASCII Real and Imaginary floating-point lag file example.