CARMA Memorandum Series #58

Third-generation CARMA Correlator FPGA Configurations

Kevin P. Rauch (UMD)

November 11, 2013

ABSTRACT

This memo describes details of the FPGA signal processing and control logic used in the third-generation (3G) CARMA correlator. This hardware incorporates a new ultra-wideband A/D converter operating at 20 GHz, sampling the entire 1-9 GHz CARMA IF band for each of 46 inputs (23 antennas x 2 polarizations). Altera Stratix IV GT FPGAs are used to capture the digitized input signals, downconvert the IF into 8 individual sub-bands (with selectable bandwidth and center frequency), and compute cross-correlation lags in real time. Included here are descriptions of the cross-correlation baseline partitioning; input data pipelines; digital signal processing components and performance; FPGA memory map; and control register specification.

Change Record

Revision	Date	Author	Sections/Pages Affected
	Remarks		
0.1	2012-Jun-12	Kevin Rauch	
	Draft 1.		
0.2	2013-Mar-12	Kevin Rauch	
	Draft 2.		
0.3	2013-Apr-01	Kevin Rauch	
	Draft 3.		
0.4	2013-Oct-11	Kevin Rauch	
	Draft 4.		

1. Correlator Overview

The CARMA array achieved first-light in early 2006. At that time the array contained 15 antennas—six 10 m (formerly OVRO) dishes and nine 6 m (formerly BIMA dishes). The first-light correlator consisted of three 15-input bands of recycled COBRA hardware reprogrammed to support both wideband and spectral line operation, and offered a total bandwidth of up to 1.5 GHz. In 2008, eight 3.5 m (formerly SZA) dishes were moved to Cedar Flat for eventual integration into CARMA, initially operating as an independent array attached to a dedicated wideband correlator comprised of sixteen 500 MHz wide, 8-input bands of COBRA hardware. Each COBRA digitizer board contains two 1 GHz 2-bit ADCs and four Altera FLEX 10KE data-processing FPGAs with 5,000 logic cells each, for a total signal processing capacity of 10,000 logic cells per RF input. Each COBRA correlator board contains 10 Altera FLEX 10KE data-processing FPGAs, for a total of 50,000 logic cells per board (5,000 per cross-correlation baseline).

The second generation (2G) CARMA correlator was put into service in 2010 and offers eight 15-input bands of 500 MHz each (in wideband mode), for up to 4 GHz of total bandwidth. In 2011, complementary support for four 30-input bands of up to 500 MHz each was added to its capabilities, enabling both 15-antenna full-Stokes and 23-antenna single-polarization observations. The second generation CARMA boards are based on Altera Stratix II FPGAs. Each CARMA digitizer board contains two 1 GHz 8-bit ADCs and four Stratix II 90K FPGAs devoted to signal processing, a total of 180,000 logic cells per RF input. The more than order-of-magnitude increase in logic compared to COBRA allowed several digital signal processing algorithms to be performed in real time by the FPGAs, including both phase offset and fractional-sample delay corrections, as well as cross-correlation calculations with up to 4-bit samples (Rauch 2008). Each CARMA correlator board contains four Stratix II 130K FPGAs, for a total of 520,000 logic cells per board (32,000 per baseline in 15-input single-polarization observing modes), allowing up to a 6x increase in channel resolution compared to COBRA.

This document details the FPGA-based capabilities of the third generation (3G) CARMA correlator (also called the ‘fast-sampler’ or ‘MRI’ correlator). This hardware iteration focuses on exending the processed bandwidth up to 8 GHz with 23 inputs, with channel resolution comparable to that previously delivered over 4 GHz with 15 inputs by the second generation system.

2. Third-generation Hardware Design

The 3G hardware represents a major advance in RF digitization capabilities with the introduction of 20 GHz 10-level samplers, each digitizing the entire usable receiver IF of 1-9 GHz for a single antenna polarization. Unlike previous generations, two polarizations are available simultaneously from all 23 antennas. This corresponds to a raw data rate of nearly 4 Tbps, a four-fold increase over the second generation CARMA correlator. In contrast to previous designs, each ADC resides on an independent card directly connected to two FPGA boards through high-speed (10 Gbps) transceiver lanes. Each FPGA board contains a single Altera Stratix IV GT device with 530,000 logic elements.

The use of ultra-wideband ADCs alters the layout of FPGA signal processing to be more input-centric as opposed to band-centric, as was previously the case. In particular, the former analog downconversion process, which created eight analog 500 MHz IF sub-bands connected to parallel bands of correlator hardware, is now performed digitally by ‘bandformer’ FPGA boards—the boards directly connected to ADCs via high-speed serial links. These boards combine the signal processing functionality of the previous digitizer boards with flexible IF downconversion capabilities. A pair of bandformer FPGA boards both receive the full output from a single ADC and process it into four independent IF sub-bands each, for a total of eight observing bands. Each observing band is defined by its IF center frequency and bandwidth (the latter ranging from 1.25 GHz to 2 MHz). Restrictions on center frequency and bandwidth values are detailed in §5.2 and §5.3, respectively. Each sub-band has ≈130,000 FPGA logic elements available for signal processing, including the new digital downconversion function—significantly less than the ≈180,000 logic elements per input per band of the 2G correlator. However, the highly oversampled input data rate (relative to the sub-band output frequency) greatly reduces the computational complexity of certain functionality—in particular delay processing (§5.1)—and as a result the decimation filter quality (§5.3 and §5.4) remains unchanged compared to the 2G system.

The Stratix IV GT device on each FPGA card contains 32 full-duplex transceivers capable of operating at speeds up to 11.3 Gbps. Transceivers are grouped into blocks of four for external communication. The front panel contains a quad QSFP+ cable cage connected to the 16 left-side transceivers (each QSFP+ cable carrys the traffic for four send and receive lanes—i.e., one transceiver block); the 16 right-side transceivers are accessible through connectors on the chassis backplane via a rear transition module (RTM) or similar bridge card. Bandformer FPGA cards use two front-panel QSFP+ cables operating at 10 Gbps to receive the 80 Gbps (4-bit x 20 GHz) ADC input data from a digitizer card. The remaining two front-panel connectors transmit identical copies of processed, downconverted IF segments for four inputs (2 antennas x 2 polarizations, one per lane) to correlator FPGA cards for cross-correlation. The rear (right-side) transceivers are used to exchange processed input data between groups of four adjacent bandformer boards to allow each bandformer to output collated data for a single IF segment (see Fig. 1), as required by the correlation back-end. All inter-FPGA communication links run at either 6.25 Gbps (for the 1.25 GHZ and 625 MHz bandwidth modes) or 5 Gbps (for 500 MHz and below), including the 8-to-10 bit modulation used for link protection. The corresponding net input data rates are 5 Gbps and 4 Gbps, respectively—sufficient to support 2-bit samples x 1.25 GHz bandwidth operation and up to 4-bit sampling in all lower bandwidth modes (including full-Stokes in all cases).

Although there are only 46 ADC samplers in the final system, requiring $46 \times 2 = 92$ bandformer boards for IF band processing (four bands per FPGA), $48 \times 2 = 96$ bandformer boards are needed to complete the backplane data networks. Each IF band also contains 12 correlator FPGA boards devoted to calculation of the cross-correlation baselines, for a total of $12 \times 8 = 96$ correlator FPGA boards. Correlator FPGA boards are physically identical to bandformer FPGA boards, but differ in their FPGA configurations, backplane connections, and front-panel cabling arrangements. In total $2 \times 96 = 192$ FPGA boards are required to implement the 3G correlator (excluding spares).

Each 3G correlator FPGA contains four times the logic of a 2G correlator FPGA, but processes six times as many baselines (24 versus 4; cf. §3). The base expectation therefore is a 1.5x decrease in channel resolution compared to the previous system. As described in §7, however, the correlation logic has been redesigned in the 3G correlator to maximize FPGA resource usage and hence resolution, thereby maintaining similar channel counts relative to 2G hardware.

3. Baseline Partitioning

In 23-input correlation mode there are $(23 \times 24)/2 = 276$ correlation baselines per observing band (23 auto-correlations and 253 cross-correlations). To maximize channel resolution, these must be distributed as evenly as possible among the 12 correlator FPGAs allocated to each band. In addition, to support indepen-

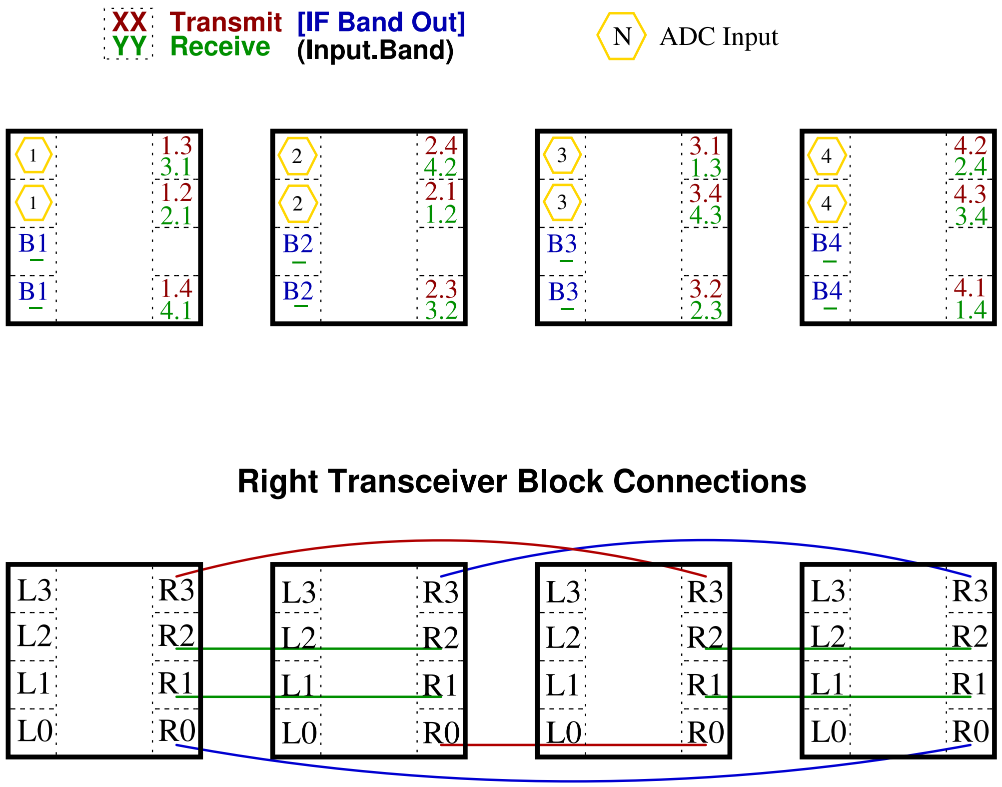

Fig. 1.— Bandformer board data fanout diagram. ADC input (gold) is received via two front-panel connections. Dual IF segment outputs (blue) containing four unique sample streams (2 antenna x 2 polarizations) are transmitted to correlator boards. Backplane traffic (red, green) redistributes input data for collation into front-panel output streams. Front-panel connections use the left-side FPGA transcievers, connections to the backplane use those on the right.

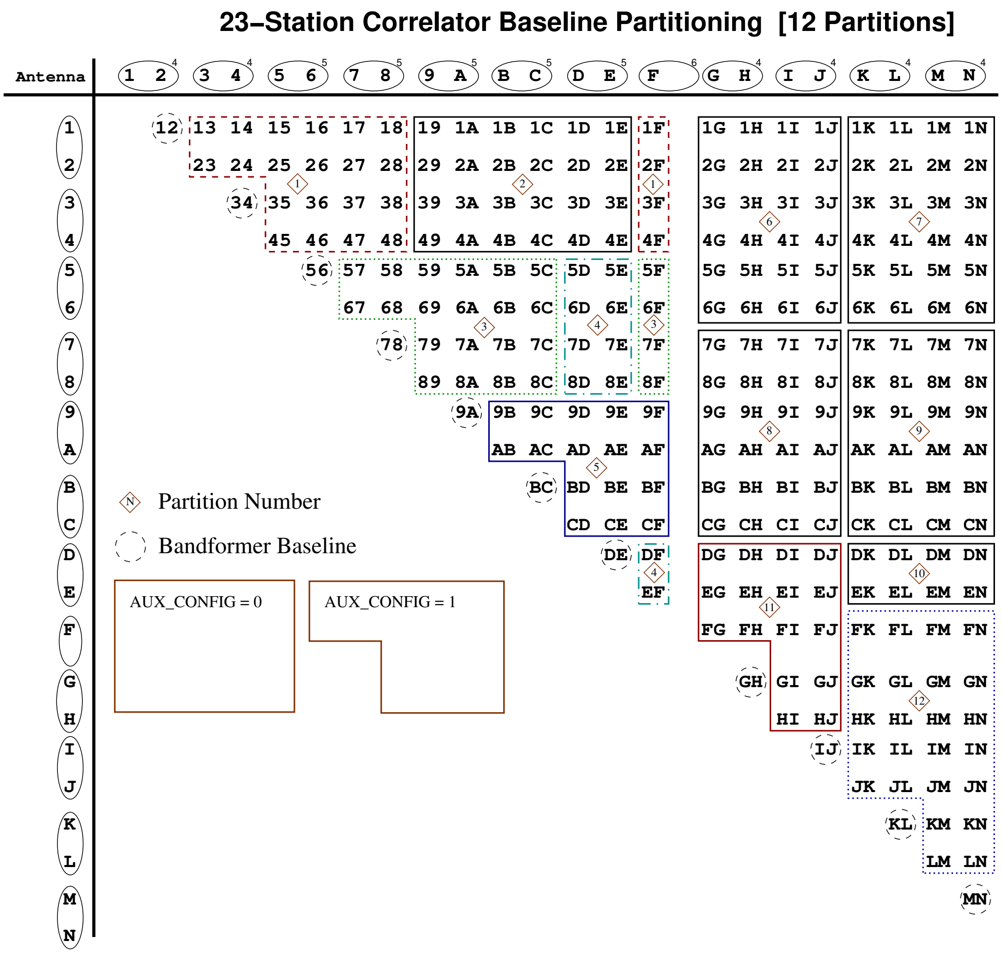

Fig. 2.— Single-polarization baseline-to-FPGA partition map for the 3G correlator. Each correlator FPGA processes up to 24 cross-correlation baselines; bandformer FPGAs also calculate a small number of baselines. This map supports both 23-input and independent 15-input/8-input operation. Total input fanout is indicated in small type above each cable bundle.

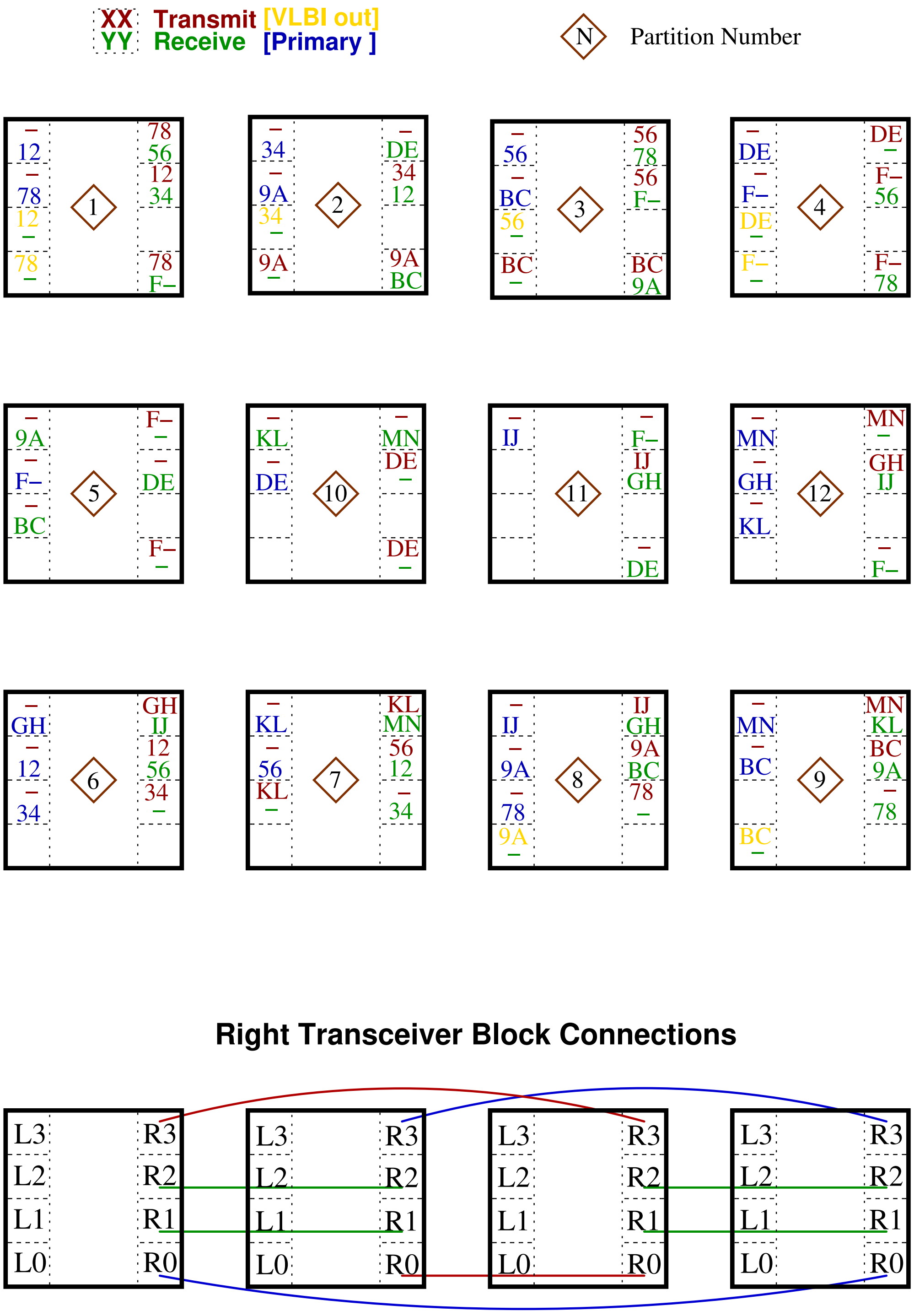

Fig. 3.— Correlator board cable fanout diagram. Primary inputs (blue) are received from bandformer boards; the remainder are passed between correlator boards. Front-panel connections use the left side transceivers, connections to the backplane use those on the right. The communication map is designed to support both 23-input and independent 15-input/8-input operation, as well as complete 15-element fanout to an external unit (e.g., a VLBI beamformer).

dent 15-input/8-input sub-array operation, no cross-communication between boards belonging to separate sub-arrays is allowed (excluding inputs ignored by the sub-array in 15+8 mode). To further improve density (and facilitate testing), bandformer FPGAs also compute a small fraction of the baselines, including the auto-correlations in particular. In contrast to the 2G system, bandformer boards supply only two copies of each input to the correlator boards; the latter is responsible for all remaining data fanout. Hence the viability of a baseline partitioning scheme depends not only on the number of baselines per FPGA, but also on the data fanout required to implement it, which must be compatible with the FPGA board I/O capabilities. It is equally important to maximize the regularity of the partitioning scheme in order to minimize the number of distinct FPGA configurations, which are labor-intensive to produce and maintain.

The final baseline partition derived for use with 23-input modes in the 3G system is shown in Figure 2. The fanout of any single input (shown in small type above the corresponding cable bundle) ranges from four to six. Each of the eight observing bands partitions baselines in this manner. Correlator FPGAs calculate up to 24 baselines arranged in one of two basic patterns. Bandformer FPGAs calculate two auto-correlations—one for each input of the observing band it outputs—as well as the asociated cross-correlation; this arrangement evenly distributes correlation logic among the bandformer FPGAs. A correlator board cable fanout map implementing Figure 2 is shown in Figure 3. For use with sub-arrays, not only must each correlator board calculate baselines for at most one sub-array, but also it *cannot provide active data fanout* for any other sub-array—otherwise they would no longer be independent (reconfiguring one would break data fanout and synchronization for the other). The fanout map was chosen to support both 23-input operation and independent 15-input/8-input sub-arrays (only). Note that the two sub-arrays consist specifically of the first 15 and final 8 inputs; no other mapping of inputs to sub-arrays is possible as data fanout is fixed by the cable topology.

In 46-input correlation mode (23 inputs x 2 polarizations), each single input in Figure 2 becomes a pair of inputs (e.g., $1 \rightarrow \{1L, 1R\}$) and each baseline expands into a 2x2 grid of baselines split between two FPGA boards—halving both the number of observing bands (from eight to four) and the channel resolution relative to the corresponding 23-input mode. In this case bandformers compute four auto-correlations and six cross-correlations per four outputs (2 inputs x 2 polarizations), split evenly between two bandformers. Dual-polarization observations, where only the diagonal of the 2x2 grid is needed, present a special case. In contrast to the previous correlator, the 3G hardware provides the option to either halve the resolution but maintain a full eight observing bands, or (as for the 2G system) maintain resolution but reduce the number of bands to four. Supporting both options does not increase the number of FPGA configurations; the former reuses the 46-input modes and the latter, the 23-input modes. The former option could not be supported previously due to a lack of digitizer hardware.

4. FPGA Memory Map

The FPGA memory map component unifies all externally-visible FPGA control registers and memory blocks into a single contiguous address space—from the viewpoint of the system bus bridge, it appears as a single large, fixed-latency RAM block. The memory map allocates 1 MB of address space arranged as 256K $\times$ 32-bit (big-endian) words, equivalent to the combined capacity of all M144K blocks in an EP4SGX530 device. (In a typical instantiation, not all of the available address space is populated with actual RAM.) The control register layout consists of 16 $\times$ 32-bit read-only registers in addresses 0x00000 to 0x0000F, followed by 48 read-write registers in addresses 0x00010 to 0x0003F. All control registers possess dedicated outputs to internal logic. Internally the control registers shadow the initial 64 locations of the first M144K RAM block, which is otherwise reserved for future use. This is followed by one M144K tap RAM block which can be used for (pre-configured) internal signal capture. One additional M144K block is allocated for each correlation calculated by the chip and used for lag dumps (in 46-input mode, two cross-correlations are dumped per RAM block).

Table 1 summarizes the contents of external FPGA memory. Details of individual control registers follows (any omitted bitfields are reserved and their contents unspecified):

- `MMAP_REG_VERSION` [Version (read-only)]
  This register contains board-level configuration settings. The bandwidth mode number encodes the spectral bandwidth; 0x0 to 0x9 for 1000 MHz to 2 MHz (1000×2^{-m} MHz) and 0xA to 0xF for 1250 MHz to 39 MHz ($1250 \times 2^{10-m}$ MHz). In this register the mode refers to the bandwidth processed by the correlation logic (corresponding to one of the four IF segments).
  - **bits 31-24:** *Reserved* (cleared)
  - **bits 23-20:** Hardware revision: 0x2 = 3G prototype
  - **bits 19-16:** FPGA type: 0xB = bandformer, 0xC = correlator
  - **bits 15-12:** Requantized sample width
  - **bits 11-8:** Correlation bandwidth mode number (local base)
  - **bits 7-0:** Configuration feature version
- `MMAP_REG_CORL_BASE` [Correlation configuration base (read-only)]
  Correlation logic settings shared by all correlations on the FPGA.
  - **bits 31-29:** *Reserved* (cleared)
  - **bit 28:** Auxiliary correlation input indicator (`USE_AUX`)
  - **bits 27-16:** Number of (32-bit) lag block locations written ($2 \times$ `DUMP_COUNT`)
  - **bits 15-12:** Metadata elements per lag stream (`NUM_META`)
  - **bits 11-0:** Number of zero+positive lags per stream (`1+NUM_LAGS`)
- `MMAP_REG_CORL_CONF1` [Correlation configuration register 1 (read-only)]
  Correlation logic settings unique to the first correlation set.
  - **bits 31-28:** *Reserved* (cleared)
  - **bits 27-24:** Number of baselines per RAM block (`NUM_PACK1`)
  - **bits 23-12:** Number of quantization state counters (`NUM_QCNT1`)
  - **bits 11-4:** Number of correlation baselines (`NUM_CORL1`)
  - **bits 3-2:** *Reserved* (cleared)
  - **bits 1-0:** Correlation type; 0 = auto, 1 = cross +lags only , 2 = cross -lags only, 3 = cross +lags and -lags (`CORL_TYPE1`)
- `MMAP_REG_CORL_CONF2` [Correlation configuration register 2 (read-only)] Correlation logic settings unique to the second correlation set (if present).

- **bits 31-28:** *Reserved* (cleared)
- **bits 27-24:** Number of baselines per RAM block (`NUM_PACK2`)
- **bits 23-12:** Number of quantization state counters (`NUM_QCNT2`)
- **bits 11-4:** Number of correlation baselines (`NUM_CORL2`)
- **bits 3-2:** *Reserved* (cleared)
- **bits 1-0:** Correlation type; 0 = auto, 1 = cross +lags only , 2 = cross -lags only, 3 = cross +lags and -lags (`CORL_TYPE2`)

• `MMAP_REG_STATUS_OLD` [Previous correlation status (read-only)]
This register contains the value of `MMAP_REG_STATUS` when the most recent lag dump completed. It can be used determine the final status of the previous correlation while the next is in progress.

• `MMAP_REG_STATUS_META1` and `MMAP_REG_STATUS_META2` [Correlation metadata status (read-only)]
Starting with bit 0 of `STATUS_META2` through bit 31 of `STATUS_META1`, contains one bit per correlation indicating whether erroneous pipeline metadata has been encountered, which may invalidate the corresponding lags. These registers are automatically cleared whenever the correlation status register (`MMAP_REG_STATUS`) is cleared.

• `MMAP_REG_STATUS_TRX` [Transceiver lane status (read-only)]
Contains detailed status information for a single transceiver lane (selected using a field in `MMAP_REG_TRX_CTRL`). Some byte-oriented status is delivered demux-by-4 as incoming bytes are demultiplexed to 32-bit parallel data into the FPGA core.

- **bits 31-28:** Transceiver state machine code
- **bit 27:** Transmit PLL lock indicator
- **bit 26:** Receive PLL lock indicator
- **bit 25:** Receive LTD (lock-to-data) indicator
- **bit 24:** Receive PMA (analog signal) active indicator
- **bits 23-20:** Receive word synchronization status (demux-by-4 output)
- **bit 19:** Transceiver block powerdown indicator
- **bit 18:** Transmit phase compensation FIFO error
- **bit 17:** Receive phase compensation FIFO error
- **bit 16:** Receive run-length violation error
- **bits 15-12:** Receive 8B/10B decode error (demux-by-4 output)
- **bits 11-8:** Receive 8B/10B disparity error (demux-by-4 output)
- **bits 7-4:** Receive control code detection (demux-by-4 output)
- **bits 3-0:** Receive alignment byte detection (demux-by-4 output)

• `MMAP_REG_CTRL_TAP` [Control bit readback (read-only)]

- **bits 31-24:** *Reserved* (cleared)

- **bits 23-20:** Center frequency update error (IF segments 3-0)
- **bits 19-16:** Phase offset update error (IF segments 3-0)
- **bit 15:** Input delay update error
- **bits 14-13:** *Reserved* (cleared)
- **bit 12:** Tap RAM write enable
- **bit 11:** Global PLL reset
- **bit 10:** Global logic reset
- **bit 9:** Global 1 PPS signal
- **bit 8:** Current 180 degree phase switch demodulation state
- **bit 7:** *Reserved* (cleared)
- **bit 6:** Lag dump FSM interruptN signal (active-low)
- **bit 5:** Correlation dump signal
- **bit 4:** Correlation active signal
- **bits 3-0:** Correlation control bits

• `MMAP_REG_CONFIG` [Data processing configuration]
This register contains FPGA-specific configuration settings. The IF mode numbers are defined as for `MMAP_REG_VERSION` and are generally read-only, except for the special narrowband configuration which supports a range of values (0x5 to 0x9). In the latter case, correlator configurations use the IF segment 0 mode; the others are meaningful only for bandformers. Invalid modes written to these fields are ignored (readback always presents the active value). The input number field determines the cross-correlation input label for band data produced by the board and is valid for banformers only (for 2G hardware this information was hard-wired into an initialization file).

- **bits 31-28:** Bandwidth mode number, IF segment 3
- **bits 27-24:** Bandwidth mode number, IF segment 2
- **bits 23-20:** Bandwidth mode number, IF segment 1
- **bits 19-16:** Bandwidth mode number, IF segment 0
- **bits 15-12:** Active IF segment count (read-only, 4 maximum)
- **bits 11-8:** *Reserved* (configuration subtype field)
- **bits 7-0:** Digitizer input number

• `MMAP_REG_STATUS` [Correlation status]

- **bits 30-4:** Counter indicating the number of clock cycles the correlation (specifically, multiply-adder) logic was active/enabled since the last correlation dump. Cannot be reset externally (cleared automatically).
- **bits 3-2:** *Reserved* (cleared)
- **bit 1:** Correlation error indicator; if set, the correlate signal went high before lag data was successfully transferred to FPGA RAM. Asserts the `corl_doneN` interrupt when set.

  - **bit 0:** Correlation done indicator; this bit is set once lag data (including metadata and quantization counts) for the most recent integration has been successfully transferred to FPGA RAM. Asserts the `corl_doneN` interrupt when set.

- `MMAP_REG_CORL_MODE` [Correlator operating mode]
  - **bit 2:** Decimation filter input samples replaced with a periodic delta function according to `MMAP_REG_IMPULSE` settings
  - **bits 1-0:**
    * Mode `"10"`: digitizer input samples replaced with ramp pattern 0x87780FF0, 0x96691EE1, ... (replicated to input sample width)
    * Mode `"01"`: prompt/delay correlation inputs replaced with test patterns based on `MMAP_REG_TEST_PATTERN`
    * Mode `"00"`: normal operation of the pipeline and correlation logic

- `MMAP_REG_DEMOD` [Phase-switch demodulation state]
  Encodes the 180 degree phase-switch demodulation sequence for up to 32 consecutive integrations (bits 31-0, with 0 being read first). A set bit indicates that samples should be negated during the corresponding integration cycle. Demodulation is applied to the raw 4-bit ADC input.

- `MMAP_REG_IMPULSE` [Impulse response pattern parameters]
  Determines the frequency of sample generation for impulse response filter testing (cf. `CORL_MODE`, bit 2). The (up to) 16-bit LSBs encode the impulse sample value (limited to `SAMP_WIDTH`); the 16-bit MSBs represent the number of (pipeline) clock cycles between impulse sample output in the pattern generator, with all zeroes output in between.

- `MMAP_REG_TEST_PATTERN` [Correlation sample test patterns]
  The 16-bit LSBs and MSBs define the test patterns fed to the prompt (`PIN`) and delay (`DIN`) cross-correlation inputs, respectively, in self-test mode (cf. `CORL_MODE`). Reset default patterns are `PIN` = 0xEB14 and `DIN` = 0x14EB.

- `MMAP_REG_PIPE_SEL` [Pipeline multiplexer selects]
  Sets the select lines for the pipeline data multiplexers, which determine the data input to the correlation logic and output to the transceivers. The inputs to these multiplexers can be deduced from Figs. 1-3 and are documented explicitly in the corresponding `pipeline` VHDL components. For bandformers the contents are as follows:
  - **bits 31-14:** *Reserved*
  - **bit 13:** Metadata select (manual)
  - **bit 12:** Input A/B select
  - **bits 11- 8:** IF segment select
  - **bits 7- 0:** IF delay control

  The metadata select bit is a manual trigger that replaces sample data (all IF segments) with their metadata in the pipeline. This is provided for testing; in normal operation this bit remains cleared as the substitution is automatically enabled during metadata verification each lag dump (cf. §6). Input A/B select swaps adjacent inputs on the bandformer's correlator outputs (swapping the two polarizations for each antenna). The (0-based) IF segment select specifies which segment is correlated/output by the FPGA (cf. Fig. 1); if the specified value is equal to or greater than the number of active IF segments (see MMAP_REG_CONFIG), behavior is undefined. The IF delay control is used to align internal IF segment data with transceiver input for correlation and band output.

For correlators the contents are:

- **bits 31-29:** *Reserved*
- **bit 28:** Correlation block baseline select (NUM_PACK = 2 only)
- **bits 27-26:** Correlation block auxiliary input select
- **bits 25-24:** Correlation block column 2 input select
- **bits 23-22:** Correlation block column 1 input select
- **bits 21-20:** Correlation block column 0 input select
- **bits 19-18:** Correlation block row 1 input select
- **bits 17-16:** Correlation block row 0 input select
- **bits 15-14:** Transceiver block 7 output select
- **bits 13-12:** Transceiver block 6 output select
- **bits 11-10:** Transceiver block 5 output select
- **bits 9-8:** Transceiver block 4 output select
- **bits 7-6:** Transceiver block 3 output select
- **bits 5-4:** Transceiver block 2 output select
- **bits 3-2:** Transceiver block 1 output select
- **bits 1-0:** Transceiver block 0 output select

The block baseline select determines which half of the 2x2 baseline block a NUM_PACK = 2 correlator configuration calculates (0 for the diagonal, 1 for the first row).

- `MMAP_REG_DIG_DELAY` [Digitizer sample delay (current)]
  Contains the current sample delay applied to the raw ADC input stream. Normally used for read-back in conjunction with `MMAP_REG_DIG_DELAY_1PPS` and `MMAP_REG_DIG_DELAY_STEP`. If written, the new value overrides the current delay setting during the next update cycle (after which it increments as usual). The delay is a 32-bit fixed-point value with 12-bit fractional component; hence the maximum possible delay is $\sim 2^{20}$ samples (52 μs @ 20 GHz).

- `MMAP_REG_DIG_DELAY_1PPS` [Digitizer sample delay (next 1 PPS)]
  The input sample delay to set at the next 1 PPS tick. After this time—until the next 1 PPS tick—the delay will increment by `MMAP_REG_DIG_DELAY_STEP` each update cycle (during correlation lag dumps). This register must be re-written prior to each 1 PPS tick to avoid assertion of a delay update error in `MMAP_REG_CTRL_TAP`. In the latter case, the delay is **not** reset on the 1 PPS tick, but continues to increment using the previous stride; this allows the value to "coast" via smooth extrapolation should updated coefficients fail to arrive.

- MMAP_REG_DIG_DELAY_STEP [Digitizer delay step (after 1 PPS)]
  The input sample delay increment to apply each update cycle (during correlation lag dumps) *after* the next 1 PPS tick. The delay value is reset to MMAP_REG_DIG_DELAY_1PPS on the 1 PPS tick itself. This register must be re-written prior to each 1 PPS tick to avoid assertion of a delay update error in MMAP_REG_CTRL_TAP.

- MMAP_REG_TRX_RESET [Transceiver reset]
  Each set bit holds the corresponding *receiver* lane in hard reset. If all four lanes in a single block are placed in reset, the entire block (including transmittters) is powered down. If all blocks on one side of the device are in reset, the associated ATX PLL is also powered down.

- MMAP_REG_TRX_CTRL [Transceiver control]
  Each byte selects the active transceiver lane or block for either MMAP_REG_STATUS_TRX or MMAP_REG_TRX_DELAY. Only the 5-bit LSBs are used for lane selects (maximum of 32 transceivers per FPGA); lanes are numbered lower-right to upper-right, then lower-left to upper-left (receive lane 0 is GXB_RX_R0 and lane 31 is GXB_RX_L15). Only the 3-bit LSBs are used for block selects (each block is four contiguous lanes); block 0 is lanes R0-R3 and block 7 is lanes L12-L15. The transceiver verification select bit allows manual control of symbol alignment verification; when enabled, all pipeline data is replaced by a fixed word alignment pattern expected by the receivers. In normal operation this bit remains cleared as pattern substitution is controlled automatically (cf. §6).

  - **bits 31-17:** *Reserved*
  - **bit 16:** Transeiver verification select (manual)
  - **bits 15-8:** Block select for MMAP_REG_TRX_DELAY
  - **bits 7-0:** Lane select for MMAP_REG_STATUS_TRX

- MMAP_REG_TRX_ERROR [Transceiver error status]
  Each set bit indicates detection of a transceiver link error on the corresponding lane since the register was last cleared. Bits stay set until cleared. Detailed link status for a particular lane can be found in MMAP_REG_STATUS_TRX after setting MMAP_REG_TRX_CTRL appropriately.

- MMAP_REG_TRX_DELAY [Transceiver input delay]
  Used for lane alignment; each byte sets the absolute delay (in receive clock cycles) to apply to one lane in a particular transceiver block $0 \leq b \leq 7$ (selected by MMAP_REG_TRX_CTRL). The delay is applied to the raw, 32-bit demultiplexed transceiver input and is unrelated to the digitizer sample rate or data demux. Typical transceiver transmit-to-receive latency is about 16 parallel clock cycles.

  - **bits 31-24:** Receive delay for lane $\ell = 4b+3$, block b
  - **bits 23-16:** Receive delay for lane $\ell = 4b+2$, block b
  - **bits 15-8:** Receive delay for lane $\ell = 4b+1$, block b
  - **bits 7-0:** Receive delay for lane $\ell = 4b+0$, block b

- MMAP_REG_IF0_FREQ [Center frequency 0 (current)]
  Reads back the current center frequency of IF segment 0, in units of the sampling frequency, as a 32-bit fixed-point value (with 32-bit fractional part). The current frequency is usually calculated automatically from MMAP_REG_IF0_FREQ_1PPS and MMAP_REG_IF0_FREQ_STEP; however,

writing to this register will override the frequency setting during the next update cycle (correlation lag dump), after which it will increment as usual.

- `MMAP_REG_IF0_PHASE` [Phase offset 0 (current)]
  Reads back the current phase offset of IF segment 0, in revolutions (units of 2π radians), as a 32-bit fixed-point value (with 32-bit fractional part). The current phase offset is usually calculated automatically from `MMAP_REG_IF0_PHASE_1PPS` and `MMAP_REG_IF0_PHASE_STEP`; however, writing to this register will override the phase offset during the next update cycle (correlation lag dump), after which it will increment as usual.

- `MMAP_REG_IF0_GAIN` and `MMAP_REG_IF0_OFFSET` [Sample gain and offset 0]
  These registers define a linear transformation applied to the (18-bit) output sample stream for IF segment 0 immediately prior to requantization into final 2-bit, 3-bit, or 4-bit band-limited samples (see §5.5). The gain is a specially encoded 20-bit limited-range floating-point value, bits 19-16 containing the binary exponent (as a signed nibble) and bits 15-0 the mantissa (a 17-bit unsigned, fixed-point mantissa with implied leading 1 and 16-bit fractional part). For example, GAIN $= 0x00000 \equiv 1$ and GAIN $= 0x23400 \equiv 4.8125$. In the current implementation, only exponents between -2 and 5 (inclusive) are supported. The offset is a 20-bit signed fixed-point value with 2-bit fractional part. Both factors occupy the 20-bit LSBs of the register.

- `MMAP_REG_IF0_FREQ_1PPS` [Center frequency 0 (next 1 PPS)]
  The IF center frequency to set at the next 1 PPS tick, in units of the sampling frequency, as a 32-bit fixed-point value (with 32-bit fractional part). Afterward the frequency will increment by `MMAP_REG_IF0_FREQ_STEP` each update cycle. See `MMAP_REG_DIG_DELAY_1PPS`, which operates similarly, for additional details.

- `MMAP_REG_IF0_FREQ_STEP` [Center frequency 0 step (after 1 PPS)]
  The center frequency increment to apply each update cycle (correlation lag dump) *after* the next 1 PPS tick for IF segment 0. The value is reset to `MMAP_REG_IF0_FREQ_1PPS` on the 1 PPS tick itself. This register must be re-written prior to each 1 PPS tick to avoid assertion of a center frequency update error in `MMAP_REG_CTRL_TAP`.

- `MMAP_REG_IF0_PHASE_1PPS` [Phase offset 0 (next 1 PPS)]
  The phase offset to set at the next 1 PPS tick for IF segment 0, in revolutions, as a 32-bit fixed-point value (with 32-bit fractional part). Afterward the offset will increment each update cycle by `MMAP_REG_IF0_PHASE_STEP`. See `MMAP_REG_DIG_DELAY_1PPS`, which operates similarly, for additional details.

- `MMAP_REG_IF0_PHASE_STEP` [Phase offset 0 step (after 1 PPS)]
  The phase offset increment to apply each update cycle (correlation lag dump) *after* the next 1 PPS tick for IF segment 0. The value is reset to `MMAP_REG_IF0_PHASE_1PPS` on the 1 PPS tick itself. This register must be re-written prior to each 1 PPS tick to avoid assertion of a phase offset update error in `MMAP_REG_CTRL_TAP`.

- `MMAP_REG_IF1_FREQ` to `MMAP_REG_IF1_PHASE_STEP` [IF segment 1 control]
  These registers are the equivalent of `MMAP_REG_IF0_FREQ` to `MMAP_REG_IF0_PHASE_STEP` for IF segment 1.

- MMAP_REG_IF2_FREQ to MMAP_REG_IF2_PHASE_STEP [IF segment 2 control]
  These registers are the equivalent of MMAP_REG_IF0_FREQ to MMAP_REG_IF0_PHASE_STEP for IF segment 2.

- MMAP_REG_IF3_FREQ to MMAP_REG_IF3_PHASE_STEP [IF segment 3 control]
  These registers are the equivalent of MMAP_REG_IF0_FREQ to MMAP_REG_IF0_PHASE_STEP for IF segment 3.

Table 1. CARMA 3G Correlator FPGA Memory Map

Symbolic Name[a]	Hex Address[b]	Description
MMAP_REG_VERSION	0x00000[c]	FPGA configuration version.
MMAP_REG_CORL_BASE	0x00001[c]	Correlation logic base parameters.
MMAP_REG_CORL_CONF1	0x00002[c]	Correlation logic set 1 parameters.
MMAP_REG_CORL_CONF2	0x00003[c]	Correlation logic set 2 parameters.
MMAP_REG_STATUS_OLD	0x00004[c]	Correlation status of last integration.
MMAP_REG_STATUS_META1	0x00005[c]	Baseline metadata status.
MMAP_REG_STATUS_META2	0x00006[c]	Baseline metadata status.
MMAP_REG_STATUS_TRX	0x00007[c]	Single transceiver detailed status.
MMAP_REG_CTRL_TAP	0x00008[c]	Control bit readback.
MMAP_REG_UNUSED_09	0x00009[c]	Reserved for future use.
MMAP_REG_UNUSED_0A	0x0000A[c]	Reserved for future use.
MMAP_REG_UNUSED_0B	0x0000B[c]	Reserved for future use.
MMAP_REG_UNUSED_0C	0x0000C[c]	Reserved for future use.
MMAP_REG_UNUSED_0D	0x0000D[c]	Reserved for future use.
MMAP_REG_UNUSED_0E	0x0000E[c]	Reserved for future use.
MMAP_REG_UNUSED_0F	0x0000F[c]	Reserved for future use.
MMAP_REG_CONFIG	0x00010	Data processing configuration.
MMAP_REG_STATUS	0x00011	Correlation status.
MMAP_REG_CORL_MODE	0x00012	Correlation/pipeline operating mode.
MMAP_REG_DEMOD	0x00013	Phase-switch demodulation state.
MMAP_REG_IMPULSE	0x00014	Impulse response cycles & sample.
MMAP_REG_TEST_PATTERN	0x00015	Delay/prompt input test patterns.
MMAP_REG_PIPE_SEL	0x00016	Pipeline multiplexer selects.
MMAP_REG_DIG_DELAY	0x00017	Digitizer input delay (current).
MMAP_REG_DIG_DELAY_1PPS	0x00018	Digitizer input delay (next PPS).
MMAP_REG_DIG_DELAY_STEP	0x00019	Digitizer input delay rate (next PPS).
MMAP_REG_TRX_RESET	0x0001A	Transceiver reset.
MMAP_REG_TRX_CTRL	0x0001B	Transceiver select control.
MMAP_REG_TRX_ERROR	0x0001C	Transceiver error status (persistent).
MMAP_REG_TRX_DELAY	0x0001D	Transceiver input delay (selected block).
MMAP_REG_UNUSED_1E	0x0001E	Reserved for future use.
MMAP_REG_UNUSED_1F	0x0001F	Reserved for future use.
MMAP_REG_IF0_FREQ	0x00020	IF segment 0 center frequency (current).
MMAP_REG_IF0_PHASE	0x00021	IF segment 0 phase offset (current).
MMAP_REG_IF0_GAIN	0x00022	IF segment 0 sample gain.
MMAP_REG_IF0_OFFSET	0x00023	IF segment 0 sample offset.
MMAP_REG_IF0_FREQ_1PPS	0x00024	IF segment 0 center frequency (next PPS).

Table 1—Continued

Symbolic Name[a]	Hex Address[b]	Description
MMAP_REG_IF0_FREQ_STEP	0x00025	IF segment 0 center frequency rate (next PPS).
MMAP_REG_IF0_PHASE_1PPS	0x00026	IF segment 0 phase offset (next PPS).
MMAP_REG_IF0_PHASE_STEP	0x00027	IF segment 0 phase offset rate (next PPS).
MMAP_REG_IF1_FREQ	0x00028	IF segment 1 center frequency (current).
MMAP_REG_IF1_PHASE	0x00029	IF segment 1 phase offset (current).
MMAP_REG_IF1_GAIN	0x0002A	IF segment 1 sample gain.
MMAP_REG_IF1_OFFSET	0x0002B	IF segment 1 sample offset.
MMAP_REG_IF1_FREQ_1PPS	0x0002C	IF segment 1 center frequency (next PPS).
MMAP_REG_IF1_FREQ_STEP	0x0002D	IF segment 1 center frequency rate (next PPS).
MMAP_REG_IF1_PHASE_1PPS	0x0002E	IF segment 1 phase offset (next PPS).
MMAP_REG_IF1_PHASE_STEP	0x0002F	IF segment 1 phase offset rate (next PPS).
MMAP_REG_IF2_FREQ	0x00030	IF segment 2 center frequency (current).
MMAP_REG_IF2_PHASE	0x00031	IF segment 2 phase offset (current).
MMAP_REG_IF2_GAIN	0x00032	IF segment 2 sample gain.
MMAP_REG_IF2_OFFSET	0x00033	IF segment 2 sample offset.
MMAP_REG_IF2_FREQ_1PPS	0x00034	IF segment 2 center frequency (next PPS).
MMAP_REG_IF2_FREQ_STEP	0x00035	IF segment 2 center frequency rate (next PPS).
MMAP_REG_IF2_PHASE_1PPS	0x00036	IF segment 2 phase offset (next PPS).
MMAP_REG_IF2_PHASE_STEP	0x00037	IF segment 2 phase offset rate (next PPS).
MMAP_REG_IF3_FREQ	0x00038	IF segment 3 center frequency (current).
MMAP_REG_IF3_PHASE	0x00039	IF segment 3 phase offset (current).
MMAP_REG_IF3_GAIN	0x0003A	IF segment 3 sample gain.
MMAP_REG_IF3_OFFSET	0x0003B	IF segment 3 sample offset.
MMAP_REG_IF3_FREQ_1PPS	0x0003C	IF segment 3 center frequency (next PPS).
MMAP_REG_IF3_FREQ_STEP	0x0003D	IF segment 3 center frequency rate (next PPS).
MMAP_REG_IF3_PHASE_1PPS	0x0003E	IF segment 3 phase offset (next PPS).
MMAP_REG_IF3_PHASE_STEP	0x0003F	IF segment 3 phase offset rate (next PPS).
MMAP_TAP	0x01000[c]	Start of signal tap RAM.
MMAP_TAP+MMAP_TAP_SIZE-1	0x01FFF[c]	End of signal tap RAM.
MMAP_LAGS	0x02000[c]	Start of lag data for set 1 correlation 0.
···	0x02000[c]	Lag 0.
···	0x02001[c]	Lag -1.
···	0x02002[c]	Lag 1.
···	0x02003[c]	Lag -2.
MMAP_LAGS + 2 × NUM_PACK × (NUM_LAGS+1)	A_{meta}[d]	Start of metadata for set 1 correlation 0.
···	$A_{\text{meta}}+0$	Prompt input metadata word 0.

5. Digital Signal Processing

The following provides in-depth information on the individual signal processing tasks present in the 'bandformer' FPGAs. They are applied to the raw digitized input signals to create the bandwidth-limited outputs later transmitted to the correlatior FPGAs, which compute a complete set of cross-correlation lags for the selected observing mode. The algorithms are described in order of application.

5.1. Delay Processing

The sample delay line corrects for time delay offsets between individual inputs. Using observations of the noise source as a reference, it is manipulated by the high-level phase flattening algorithm, which determines the input delay (and phase) offsets required for all cross-correlation baseline spectra to achieve zero phase. In normal operation the delay line also includes the bulk signal delay due to physical path length differences upstream of the noise source (which is injected directly into the IF). Only the relative delay between inputs is of consequence.

The delay line is split into two major components: a simple whole-sample delay, implemented in FPGA RAM, and a fractional-sample delay, consisting of a highly-demultiplexed asymmetric FIR filter with reloadable coefficients. The delay is adjusted on a fast time scale—the 64 Hz correlation lag dump rate—using delay coefficients computed by a small state machine inside the FPGA, based on a pair of control register settings (`MMAP_REG_DIG_DELAY_1PPS` and `MMAP_REG_DIG_DELAY_STEP`; cf. Table 1). The delay control registers, updated by the chassis CPU once per second, specify the requested total delay (in samples) and its average rate of change (in samples/s) for the following second, which the state machine double-buffers and begins to apply once the next 1 PPS signal is received. It incorporates logic to verify that the delay registers were loaded precisely once between each pair of 1 PPS ticks, and sets a persistent error indicator should this fail to be satisfied.

The maximum required delay is determined by the physical line length differences in the most extended array configuration. For CARMA this is the A array, with baselines up to 2 km ($7\,\mu$s geometric delay); including allowance for optical fiber length disparities and a safety margin, the total delay line requirement is $20\,\mu$s. The raw 4-bit @ 20 GHz digitized input is delivered to the FPGA core as a 320-bit (demux-by-80 x 4-bit) @ 250 MHz data bus; the total delay RAM required is 1.6 Mb or at least 12 Stratix IV M144K memory blocks. The FPGA configurations allocate 20 M144K blocks to the delay line, 30% of the blocks in an EP4S100G5 device, for a maximum possible delay of $32.7\,\mu$s. This delay is applied to the raw input prior to separation into individual complex baseband IF segments (of which there are four per bandformer FPGA); more delay RAM would be required in the latter case due to the complex vs. real data and the expansion of sample bit-width due to intervening signal processing.

Design of the fractional-sample delay filter in the 3G hardware is greatly simplified by the fact that raw input consistently undergoes decimation by an order of magnitude or more in creating each IF segment. Thus applying the sub-sample delay prior to all decimation, but after generation of complex baseband (which centers the IF segment on zero frequency), implies that the delay filter need only perform well over a small fraction of the quantized bandwidth. For the 3G hardware the mimimum decimation factor is eight (1.25 GHz bandwidth mode, assuming 20 GHz sampling); hence the highest possible complex baseband frequency is $f_{\rm N}/16$, where $f_{\rm N}$ is the Nyquist frequency. In this case, the nominal CARMA delay resolution requirement of 0.25 degrees (0.0007 samples) per output can be met with a simple 2-coefficient fractional-delay filter, specifically $y_i = (1-\delta)x_i + \delta x_{i-1}$, where $0 \le \delta < 1$ is the fractional sample delay and $\{y_i\}$ is

Table 1—Continued

Symbolic Name[a]	Hex Address[b]	Description
...	$A_{\rm meta}+1$	Delay input metadata word 0.
...	...	...
$A_{\rm meta}+2\times$ NUM_META	$A_{\rm qcnt}$[d]	Start of quantization counters for set 1 correlation 0.
...	$A_{\rm qcnt}+0$	Sample count for quantization state 0x00.
...	$A_{\rm qcnt}+1$	*Cleared.*
...	$A_{\rm qcnt}+2$	Sample count for quantization state 0x01.
...	$A_{\rm qcnt}+3$	*Cleared.*
...	...	...
$A_{\rm qcnt}+2\times$ NUM_QCNT1	$A_{\rm samp}$[d]	Start of sample dump area for set 1 correlation 0.
...	...	...
...	0x02FFF[c]	End of sample dump area for set 1 correlation 0.
MMAP_LAGS + 0x01000	0x03000[c]	Start of lag data for set 1 correlation 1.
...	...	...
MMAP_LAGS + NUM_CORL1 × 0x01000	$A_{\rm lags2}$[d]	Start of lag data for set 2 correlation 0.
...	...	...
MMAP_FPGA_SIZE - 1	0x3FFFF	Top of local FPGA memory.

[a]Defined in carmacorl CVS file fastsamp/share/src/fastsamp_components.vhd.

[b]As seen by the 32-bit external memory interface.

[c]These locations (and all tap/lag dump RAM) are read-only.

[d]Calculable using MMAP_REG_CORL_BASE and MMAP_REG_CORL_CONF1.

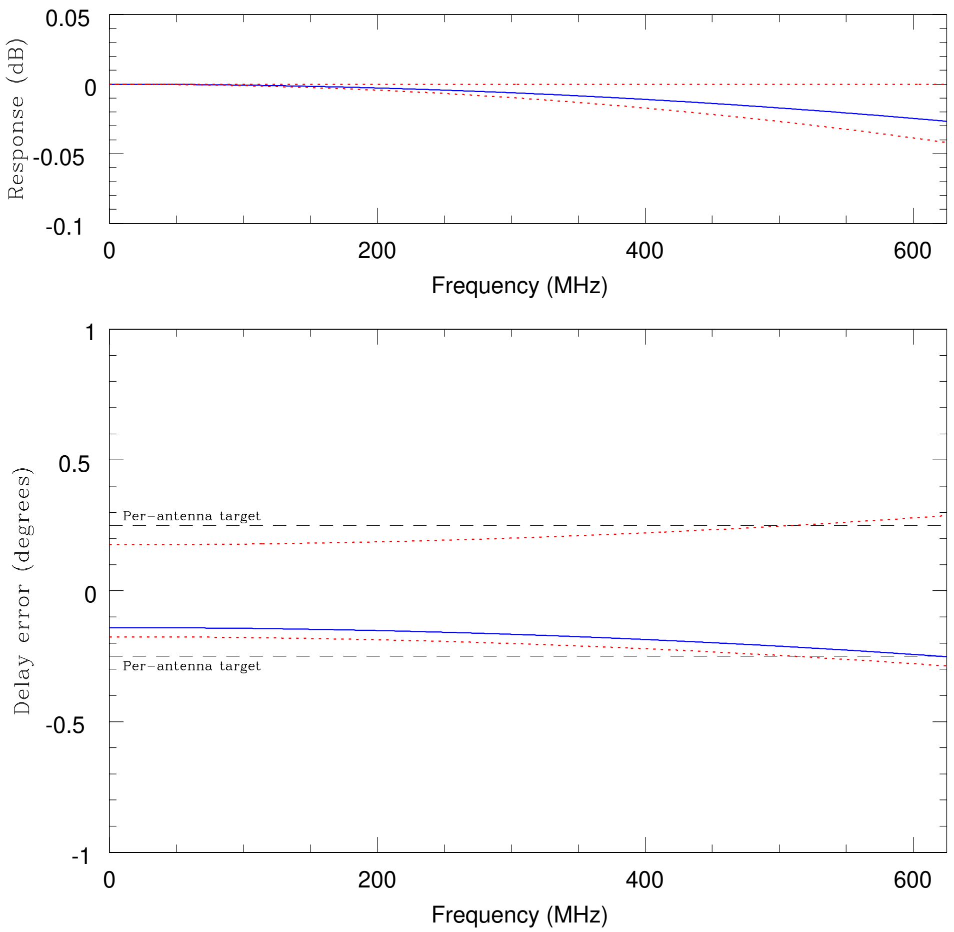

Fig. 4.— CARMA 3G correlator fractional sample delay filter performance. The top and bottom panels display wideband frequency and delay responses, respectively. Red, dotted lines show the worst-case errors over all possible delays; blue, solid lines show the response for $\delta = 0.2$ in particular.

the delayed version of the sample sequence $\{x_i\}$. The filter coefficients can be derived using the Lagrange interpolation method (TODO: ref). The simplicity of the coefficients makes the filter easy to implement in FPGA logic, and their symmetry offers additional resource savings—only a *single* multiplication per (baseband) sample is required to implement the filter! By way of comparison, the previous generation hardware required a large, 80-coefficient fractional-delay filter—partitioned into 128 individual sub-filters—consuming nearly 100% of the FPGA RAM blocks and fed by a multi-stage state machine interpreting a data stream containing encoded coefficients and other auxiliary values (Rauch 2008).

The corresponding delay filter response is shown in Figure 4, assuming 20 GHz sampling and 9-bit quantization for δ. Dotted lines indicate the worst-case error envelopes for amplitude and delay response; solid lines show the response for the concrete example $\delta = 0.200$. Most of the phase error—the residual at zero frequency—is round-off error in δ; for 8-bit and 12-bit quantization (not shown), the maximum phase error is 0.46 degrees and 0.11 degrees, respectively. Frequency response is not affected. To conserve logic, application of this filter is integrated into the creation of complex baseband as described in the following section.

5.2. Frequency Modulation and Phase Offset

Every bandformer FPGA produces band-limited sample streams for four independent IF segments, each with their own (possibly overlapping) center frequency and total bandwidth. To support Doppler tracking, the center frequencies can vary in real time, whereas bandwidth changes require reconfiguring the FPGAs. Total bandwidth per IF segment can range from 1.25 GHz to 2 MHz; bandwidth choices are based on power-of-two decimation from a common 1.25 GHz (or 1.00 GHz; TBD) complex baseband. Conceptually, the first step in creating the complex baseband is to center the IF segment on DC by multiplying the real samples by a rotating complex phasor; as in the previous correlator, this phasor includes a dynamic phase offset to enable real-time input phase correction. Due to the very high input data rate, however, practical implementation of the fractional-sample delay requires that it be integrated into the phasor calculation. This is quite reasonable both conceptually and operationally since the delay and phase corrections are closely related and updated simultaneously.

To center the desired passband on zero frequency and correct for any phase offsets, the raw input sequence $\{x_k\}$ is multiplied by a counter-clockwise rotating phasor $\{\Phi_k\}$, where

$$\Phi_k = e^{+i(\pi f_0 k - \phi)} = \cos(\pi f_0 k - \phi) + i \sin(\pi f_0 k - \phi) \equiv A_k + iB_k,$$

the center frequency $f_0 > 0$ is in units of f_N and ϕ is the antenna-based phase correction. In the CARMA band definition, which uses the e^{-i} sign convention for the forward FFT, this corresponds to centering the negative-frequency passband on DC while applying ϕ in the appropriate sense. The resulting complex baseband sequence is $\{z_k\} = \{A_k x_k + iB_k x_k\}$. Applying the fractional delay correction to $\{z_k\}$ yields the sample series $\{z'_k\}$, where

$$z'_k = [(1-\delta)z_k + \delta z_{k-1}] = 2^{-D}\left\{\left[dA_{k-1}x_{k-1} + 2^D A_k x_k - dA_k x_k\right] + i\left[dB_{k-1}x_{k-1} + 2^D B_k x_k - dB_k x_k\right]\right\},$$

D is the fixed-point precision (bit-width) of δ, and $d = 2^D\delta$ is an unsigned integer satisfying $0 \le d < 2^D$. In practice the scale factor 2^{-D} is ignored, and $-1 \le \{A_k, B_k\} \le +1$ are similarly replaced with quantized counterparts $-2^{Q-1} < \{a_k, b_k\} < 2^{Q-1}$, where a_k and b_k are Q-bit signed integers. Note that the FPGA numerically controlled oscillator (NCO) component, which calculates $\{a_k, b_k\}$, will not generate $\pm 2^{Q-1}$ to avoid saturating the Q-bit integers, even though the output is mathematically incorrect by a full LSB in those cases.

Support for Doppler tracking places two basic requirements on the digital downconversion. The need for a dynamic center frequency has already been mentioned. The timescale on which f_0 is updated for tracking purposes is long—currently 8 minutes in the real-time system (RTS). For consistency with the delay and phase update procedure, however, the FPGA downconversion component allows f_0 to change on the lag dump timescale, with frequency updates once per second and smooth interpolation in between. This requires little additional logic and permits useful lab tests such as fast (virtual) frequency sweeps. The second requirement is that the downconversion frequency error, $\Delta f_0 = f_{\rm actual} - f_0$, satisfy $|\Delta f_0| \lesssim 0.1$ channels in all bandwidth modes to ensure spectral line observations are not degraded by inter-channel frequency drift. Satisfying this for the 2 MHz 2-bit mode requires a frequency resolution of 500 Hz—a 24-bit dynamic range relative to $f_{\rm N} = 10\,{\rm GHz}$; hence the 32-bit frequency control registers are more than sufficient. Note that other than these requirements, the correlator knows nothing per se about Doppler tracking, which remains an RTS responsibility.

The incoming data at this stage is 4-bit × demux-by-80 @ 250 MHz. Since instantiating 80 NCOs in parallel would consume excessive FPGA resources, the design instead buffers one full period of NCO output into FPGA RAM blocks. This requires exacting care to implement in the case of demux-by-80 (and other non-power-of-two values) because the inherently binary NCO input frequency naturally produces only power-of-two output periods, incommensurate with the RAM layout required to support the data demux factor (in the present case, the RAM is 80 elements wide with an arbitrary depth). One way to overcome this would be to increase the RAM depth so that the total number of elements is the least common multiple of the demux factor and the NCO output period, so that the output period is an integral number of rows. The minimum RAM depth (rows) required, R, is determined by the combination of the NCO period in clock cycles, P, and the input demux factor, M. Conceptually, R is the number of unique values of $(M \cdot k) \bmod P$ evaluated over all integers k. The general solution is $R = P/G$, where $G = \gcd(P, M)$ is the greatest common divisor of P and M; hence for the simple case $G = M$, $R = P/M$. The RAM in turn contains $O = M/G$ repetitions of the NCO output. For example, for an 8-bit NCO frequency the (maximum) output period is $P = 256$, which for $M = 80$ implies $G = 16$, $R = 16$, and $O = 5$—repeating the 256-cycle output five times in succession precisely fills 16 rows of 80 columns each. Clearly, however, this solution significantly increases RAM usage (and the time required to fill it) as compared to a binary-compatible demux factor, by a factor of O.

A more efficient solution to this problem is to constrain the NCO input frequency such that the output period is *close* to an integral multiple of the demux factor. This amounts to instantiating an NCO with sufficient frequency precision such that the frequency error incurred to satisfy this constraint is negligible. For the 3G system, 32-bit NCO input frequencies are used, providing an ample safety margin beyond the 24-bit accuracy requirement mentioned above. Constraining the requested center frequency to the nearest supported frequency is done as follows. First the requested IF center f_0 is multiplied by the demux factor M and rounded to the nearest integer, $m = {\rm rnd}(M \cdot f_0)$. Next the decomposition $m = M \cdot q + r$ is calculated, where $q = k/M$ and $r = k {\rm mod} M$ are the quotient and remainder modulo M, respectively. Finally, given a total RAM size of $R * M$ elements, where $R = 2^s$ is the number of rows, the constrained 32-bit center frequency is $f_0' = 2^{32-s}\,[q + r/M]$. Note that the constraint calculation occurs in the FPGA itself through the use of components specially implemented to reduce computational complexity, allowing the frequency to be updated dynamically on a fast timescale (the same used for the delay and phase offset).

The need for high dynamic range in the center frequency combined with a high input demux factor requires a multi-stage approach for practical implementation. In the final design a second stage of modulation is applied for output bandwidths below 500 MHz to fine-tune the initial, coarser correction. Initial wideband downconversion uses a RAM cache allocating $R = 256$ rows for each of the $\{a_k\}$ and $\{b_k\}$ sequences, with individual a_k and b_k values rounded to 9-bit values (sufficient to limit distortion products to $\lesssim -50\,\mathrm{dBm}$)—thereby consuming (per IF segment) 40 Stratix IV M9K RAM blocks in total for the 9-bit a_k and b_k values and an additional 80 RAM blocks for the 18-bit da_k and db_k delay products. Each FPGA contains 1280 M9K blocks and processes four IF segments, in total a 38% M9K utilization by this stage. A state machine re-initializes RAM during each lag dump; to speed loading, four parallel NCOs are used, for a net reload time of $(R*M/4)/(250\,\mathrm{MHz}) = 20.48\,\mu\mathrm{s}$—just below the phase switch settling time of $21\,\mu\mathrm{s}$. The RAM is filled such that rows are read sequentially in operation. With these parameters, the maximum center frequency error at this stage is $\Delta f_0' = f_\mathrm{N}/(R*M) = 0.488\,\mathrm{MHz}$, which meets the resolution criterion for output bandwidths of 500 MHz and above. Below this bandwidth a second stage is applied, adjusting the center frequency by the residual $\delta f = f_0 - f_0'$. This stage occurs at low demux and is implemented using the direct output from high-precision NCOs; the maximum frequency error in these modes is $\approx 2\,\mathrm{Hz}$, a velocity error of ≈ 1 cm/s for observations at 3 mm.

5.3. Decimation

At this stage the complex baseband signal is reduced from 10 GHz to its final bandwidth—ranging from 1.25 GHz to 2 MHz—by applying multi-stage digital filtering and decimation. Each stage applies an anti-aliasing FIR filter to reject out-of-band signal power, followed by simple 2:1 or 5:1 decimation (depending on the filter). See Rauch (2003) for digital filter terminology and Crochiere and Rabiner (1983) and Mintzer (1982) for theoretical background. Filtering and decimation in the 3G system is designed to maintain the following criteria over the output bandpass (given ideal input):

- Peak-to-peak bandpass ripple below 0.1 dB.
- Peak aliased noise power below -40 dB.

More specifically, for an N channel spectrum, these criteria are satisfied in the central $N-4$ channels; hence the two half-width edge channels and one additional channel from each edge should be discarded for maximum signal integrity. The following filter sequences are used to achieve this goal:

$$
\begin{array}{rllll}
1.25\,\mathrm{GHz}: & & F2, & F4x, & F5a; \\
625\,\mathrm{MHz}: & & (F2)^2, & F4x, & F5a; \\
\vdots & & \vdots & & \\
2.5\,\mathrm{MHz}: & & (F2)^{10}, & F4x, & F5a; \\
 & & & & \\
1.00\,\mathrm{GHz}: & H4, & & & F5a; \\
500\,\mathrm{MHz}: & H1, & & F4x, & F5a; \\
250\,\mathrm{MHz}: & H1, & F2, & F4x, & F5a; \\
125\,\mathrm{MHz}: & H1, & (F2)^2, & F4x, & F5a; \\
62.5\,\mathrm{MHz}: & H1, & (F2)^3, & F4x, & F5x; \\
\vdots & & \vdots & & \\
2\,\mathrm{MHz}: & H1, & (F2)^8, & F4x, & F5x;
\end{array}
$$

where each Fn (half-band) and Hn (fifth-band) filter is implicitly followed by either 2:1 or 5:1 decimation, respectively, and $(Y)^M$ means the filter Y is applied M consecutive times. Coefficients for these filters are listed in Table 2. The $F2$ filter can be found in § 5.5.2 of Crochiere and Rabiner (1983); the remainder are optimized filters derived especially for CARMA use according to the listed criteria.

5.4. Band Edge Definition

5.5. Sample Gain and Offset Control

Sample processing includes dynamic digital gain and offset control. The gain and offset are specific to each IF segment and controlled by the `MMAP_REG_IF0_GAIN`–`MMAP_REG_IF3_GAIN` and `MMAP_REG_IF0_OFFSET` – `MMAP_REG_IF3_OFFSET` registers. Each `GAIN` register is a 20-bit binary floating-point value containing a 4-bit signed exponent (MSBs) and 16-bit unsigned mantissa fraction (LSBs). The complete 17-bit fixed-point mantissa consists of an implied leading one with 16-bit fractional part (the register LSBs). The `OFFSET` register is a 20-bit, signed fixed-point value with 2-bit fractional part. Changes to these registers take effect immediately. The gain values are used in the calculation of $T_{\rm sys}$ for each band.

Gain control occurs in two phases. As part of the sample requantization (bit reduction) occurring immediately prior to the final sharp-edged half band filtering stage, multiplication/division by a power of two—applying the binary exponent in `MMAP_REG_IFn_GAIN`—is performed on the multi-bit intermediate samples. The fixed-point gain and offset are applied immediately before final sample requantization (see § 5.6) to the incoming, 18-bit band-limited sample stream x as follows:

$$x \rightarrow x' = (\mathrm{GAIN} \times x + 2^{14} \times \mathrm{OFFSET})/2^{16}.$$

This two-stage approach maximizes both dynamic range and precision while limiting logic usage. (Note that to further conserve logic, the actual range and precision processed by the configuration may be less than that allowed by the register specification; see the register definition for the supported limits.) The gain and offset may be set as desired, but are normally used to achieve an optimal output sample mean and standard deviation.

The multi-bit input to these stages are normally close to Gaussian, as expected by the central limit theorem; however, detailed agreement will vary with the number of intervening stages and distribution of the raw input samples. The target mean, μ, is always zero; the standard deviation, σ, is set to maximize the weak-correlation detection efficiency of the requantized samples. The precise value of σ is not critical here as the multi-bit input ($\approx$ 18 bits are available) is requantized to many fewer bits following this stage; typically, multi-bit full-scale corresponds to $\sim 6\sigma$ once requantization thresholds have been optimized.

5.6. Sample Requantization

After gain and offset renormalization, the final band-limited ouput is requantized to 2-bit, 3-bit or 4-bit samples; each corresponds to a unique FPGA configuration and hence a reconfiguration is required to change it. Using fewer bits decreases cross-correlation detection efficiency but offers higher spectral resolution for a given bandwidth. The detection efficiencies are 87.24% (2-bit), 96.26% (3-bit), and 98.36% (4-bit); for details of the quantization schemes, see Appendix B of Rauch & Salter (2012). Spectral resolutions for each are given in Tables 3-5 below.

A set of quantization state counters is attached to the requantized sample stream and read out as part of the lag dump cycle. They are used to monitor sample statistics and to determine the optimal gain/offset values for each IF segment.

5.7. Phase Switch Demodulation

Following sample requantization, any 180-deg phase switching applied to the analog input signal is removed by negating sample values as appropriate. This step is performed after all other signal processing to allow not only analog asymmetries, but also any digital artifacts (such as rounding asymmetries) that may have been introduced to be mitigated by phase switching as well. The output of this stage comprises the data to be cross-correlated.

6. Data Transport and Synchronization

In the 2G correlator, data is transferred from digitizer to correlator boards via synchronous LVDS links. As a consequence, complete delay correction of a data stream to a reference can be done at a single location—at the source, in the digitizer—with the guarantee that all copies of the stream received by the correlator boards will arive with the zero relative delay (i.e., in alignment with all other delay-corrected data streams). Hence delay correction is a local, pair-wise operation.

In the 3G correlator, all transport of sample data (and metadata) occurs via asynchronous, point-to-point serial links. This adds another layer of complexity to delay alignment as correction at the source (the bandformer) is generally insufficient. Correction is now a distributed, global operation involving all data copies and reception points—alignment to a reference on one correlator board does not ensure alignment on any other. This consideration applies equally to inter-bandformer data exchange, required to collate fixed-input, multi-band data bundles into multi-input, fixed-band outputs (all correlations calculated by a single FPGA board being limited to one IF segment). This fact complicates the phase flattening algorithm in the 3G system as compared to the (already intricate) 2G implementation.

To help ensure data integrity the 3G system contains three levels of pipeline verification:

(1) The low-level transceiver links employ 8b10b encoding, whose status is monitored continuously to detect physical transmission errors such as broken links, loss of data lock and parity violations.

(2) During each phase switch settling interval, every link undergoes a period of symbol alignment verification to ensure that the 32-bit parallel data sent to the core arrives with the correct byte ordering (and value). The phase switch rate is 1024 Hz.

(3) Also every phase switch cycle, pipeline metadata received by the correlation logic is examined for self-consistency, providing an end-to-end data integrity check (metadata is populated by the bandformers at the head of the pipeline).

Failure of any of these checks leads to automatic invalidation of all affected correlation data during the next lag dump. The error status is reset automatically after the lag dump is read (when correlation status is cleared).

7. Correlation Logic

In the 3G system, each correlator FPGA calculates many baselines (up to 24; see §3), with 1/3 less FPGA logic allocated per baseline compared to the previous system. This translates directly into a 1/3 loss in channel resolution, all other things being equal. To reclaim as much resolution as possible, the correlation logic in the 3G system was significantly redesigned to increase its logic efficiency. The two major areas of improvement are (1) resource sharing between baselines; and (2) RAM-based lag counters. As shown in Figure 2, in the new system each input appears in several of the baselines calculated by a single FPGA. This allows some components to be shared between baselines, reducing overall logic usage. In particular, the input delay lines needed to compute high lag numbers were moved to a new high-level component, `corl_block`, which manages the resources for an arbitrary rectangular array of baselines. This arrangement saves approximately Q bits of registered logic per lag, where Q is the sample quantization bit-width. An input mux selecting either data samples or test samples for correlation was also raised to `corl_block`, on average reducing the resource usage for this feature by a factor of five.

The most significant change made to the correlation logic was to replace logic-based lag counters with RAM-based lag counters. In the narrowband modes (bandwidths less than 100 MHz), use of this feature reduces registered logic by over a factor of two, removing it as a resource bottleneck. This most benefits the 2-bit sample modes, which exhibit the highest ratio of registered to combinational correlation logic. In the new design (the `lag_cache` component), multiple lag counters are stored in a single medium-sized (M9K) FPGA RAM block managed like a ring buffer. A simple state machine continuously reads individual lags from RAM, increments them if their (buffered) carry-in bit is set, and then writes them back into RAM. For reliable results, the number of lags stored in a single RAM block cannot exceed the minimum period between carries. The latter is regulated by maintaining a small 'prescaler' lag counter in logic, which acculumates carries from the multiply-adder (which can emit them as frequently as every clock cycle). During lag dumps, the prescaler counter is read out along with the RAM-based counter, allowing counter bits to be split between them arbitrarily. Increasing the number of prescaler bits decreases the carry-in rate to the RAM-based counters, allowing more lags to be stored in a single RAM block (decreasing RAM requirements) at the expense of additional logic cell utilization. A 5-bit prescaler was found to provide the optimum balance.

A small change in 3G correlation logic behavior compared to previous iterations is the calculation of one additional positive (more precisely, non-negative) lag for cross-correlation baselines, as compared to the number of negative lags. This enforces full positive-negative lag symmetry which supports normal-ordering baseline inputs in lag space through exchange of positive and negative lags (with subsequent discard of the final, most positive lag), at the cost of $\sim 1\%$ increase in logic usage.

Estimated 23-input (single-polarization) channel resolution for each supported bandwidth mode and quantized sample bit-width are given in Tables 3-5. The listed mode bandwidth has been rounded to the nearest MHz. Spectral line bandwidths are based on power-of-two reductions from a common, wide bandwidth. Resolutions for the 46-input (full-Stokes) observing modes are precisely half that of the corresponding 23-input mode; resolutions in split 15-input/8-input subarray modes are identical to the 23-input values (the same FPGA configurations are used). Note that due to inter-FPGA communication limitations, the 1250 MHz mode is available with 2-bit quantization only.

Because one bandformer FPGA calculates four independent IF segments, each with a (possibly) different bandwidth, supporting an arbitrary mixture of the available bandwidth modes would require thousands of unique FPGA configurations and numerous months of CPU time to generate them. A broad but limited mixture of bandwidth choices was implemented to balance this consideration, as follows. First, each

Table 2. FIR Filter Coefficients

Filter	Expansion[a]	Coefficients
$F2$	2	{1, 2, 1}
$F4x$	5	{-1, 0, 7, 12, 7, 0, -1}
$F5a$	7	{1, 0, -4, 0, 17, 28, 17, 0, -4, 0, 1}
$F5x$	7	{1, 0, -6, 0, 29, 48, 29, 0, -6, 0, 1}
$H1$	5	{1, 2, 3, 4, 5, 4, 3, 2, 1}
$H4$	7	{-1, -1, -2, -1, 0, 5, 8, 15, 17, 20, 17, 15, 8, 5, 0, -1, -2, -1, -1}

[a]Bit expansion of filtered samples (output minus input bit-widths).

Table 3. Estimated CARMA 3G correlator spectral resolution [2-bit samples]

Bandwidth (MHz)	Channels (per sideband)	δF (MHz)	δV[3 mm] (km/s)	$V_{\rm tot}$[3 mm] (km/s)	δV[1 mm] (km/s)	$V_{\rm tot}$[1 mm] (km/s)
1250	57	22.3	67	3750	22	1250
625	97	6.51	20	1875	6.5	625
500	97	5.21	16	1500	5.2	500
250	161	1.56	4.7	750	1.6	250
125	257	0.488	1.5	375	0.49	125
62	353	0.178	0.53	188	0.18	62.5
31	353	0.089	0.27	93.8	0.089	31.2
16	353	0.044	0.13	46.9	0.044	15.6
8	353	0.022	0.067	23.4	0.022	7.81
4	353	0.011	0.033	11.7	0.011	3.91
2	353	0.006	0.017	5.86	0.006	1.95

Table 4. Estimated CARMA 3G correlator spectral resolution [3-bit samples]

Bandwidth (MHz)	Channels (per sideband)	δF (MHz)	δV[3 mm] (km/s)	$V_{\rm tot}$[3 mm] (km/s)	δV[1 mm] (km/s)	$V_{\rm tot}$[1 mm] (km/s)
625	65	9.77	29	1875	9.8	625
500	65	7.81	23	1500	7.8	500
250	97	2.60	7.8	750	2.6	250
125	161	0.781	2.3	375	0.78	125
62	289	0.217	0.65	188	0.22	62.5
31	289	0.109	0.33	93.8	0.11	31.2
16	289	0.054	0.16	46.9	0.054	15.6
8	289	0.027	0.081	23.4	0.027	7.81
4	289	0.014	0.041	11.7	0.014	3.91
2	289	0.007	0.020	5.86	0.007	1.95

Table 5. Estimated CARMA 3G correlator spectral resolution [4-bit samples]

Bandwidth (MHz)	Channels (per sideband)	δF (MHz)	δV[3 mm] (km/s)	$V_{\rm tot}$[3 mm] (km/s)	δV[1 mm] (km/s)	$V_{\rm tot}$[1 mm] (km/s)
625	25	26.0	78	1875	26	625
500	25	20.8	63	1500	21	500
250	41	6.25	19	750	6.3	250
125	81	1.56	4.7	375	1.6	125
62	145	0.434	1.3	188	0.43	62.5
31	145	0.217	0.65	93.8	0.22	31.2
16	145	0.109	0.33	46.9	0.11	15.6
8	145	0.054	0.16	23.4	0.054	7.81
4	145	0.027	0.081	11.7	0.027	3.91
2	145	0.014	0.041	5.86	0.014	1.95

Table 6. Supported Spectral Bandwidth Combinations

$\{W,W,W,W\}$	$\{I,I,I,I\}$	$\{N,N,N,N\}$
$\{W,W,W,I\}$	$\{W,W,W,N\}$	$\{W,I,I,I\}$
$\{I,I,I,N\}$	$\{W,N,N,N\}$	$\{I,N,N,N\}$
$\{W,W,I,I\}$	$\{W,W,N,N\}$	$\{I,I,N,N\}$
$\{W,W,I,N\}$	$\{W,I,I,N\}$	$\{W,I,N,N\}$

bandwidth mode is classified as either wide (1250 MHz), intermediate (625 MHz to 62 MHz), or narrow (31 MHz to 2 MHz). By special design all narrow modes are created by a single bandformer component offering dynamic (i.e., runtime) bandwidth selection—hence an arbitrary mixture of narrowband modes can be supported. The complete selection of (four) simultaneous bandwidth modes in a single FPGA consists of all possible combinations containing (at most) one specific intermediate bandwidth component, multiple copies thereof being permissible. Aside from that restriction, wide, intermediate, and narrow modes can be mixed arbitrarily. Table 6 lists these combinations systematically using W for wideband (1250 MHz), I for intermediate (any *single* bandwidth in the range 625 MHz to 62 MHz—each I is the same) and N for narrowband (any set of bandwidths from 31 MHz to 2 MHz—each N can be different). For example, $\{1250, 1250, 125, 31\}$, $\{250, 250, 250, 8\}$ and $\{1250, 62, 31, 8\}$ are all supported combinations, whereas $\{125, 62, 31, 8\}$ is not. For single-polarization observations eight simultaneous observing bands are available; although the first four and second four bands must separately satisfy the preceding restrictions, they are completely independent of each other.

Lags are transferred from the FPGA to the crate CPU at a 64 Hz rate. The maximum data volume per baseline is $384\,\mathrm{chan} \times 2\,\mathrm{lags/chan} \times 4\,\mathrm{bytes/lag} = 3\,\mathrm{KB}$ (in 23-input mode) and thus the maximum transfer rate per FPGA is $24\,\mathrm{bl} \times 3\,\mathrm{KB/bl} \times 64\,\mathrm{Hz} = 4.5\,\mathrm{MB/s}$. There are 8 FPGA boards per chassis; hence the data volume per CPU is ≤ 36 MB/s. For the correlator as a whole, the maximum possible correlation data rate into the pipeline is $8\,\mathrm{bands} \times 2\,\mathrm{sides} \times 276\,\mathrm{bl} \times 3\,\mathrm{KB/bl} \times 2\,\mathrm{Hz} \approx 26$ MB/s.

REFERENCES

Crochiere, R., and Rabiner, L. 1983, *Multirate Digital Signal Processing* (Englewood Cliffs: Prentice-Hall), Chapter 5.

Hawkins, D.W. 2012, *Terasic TS Series Development Discussion*, `http://www.ovro.caltech.edu/~dwh/wbsddc/terasic.pdf`

Mintzer, F. 1982, *On Half-Band, Third-Band, and Nth-Band FIR Filters and Their Design*, IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 30, no. 5, pp. 734-738

Rauch, K.P. 2003, *Digital FIR Filtering Options for CARMA Digitizers*, CARMA Memo 12.

Rauch, K.P. 2008, *Revised CARMA Correlator FPGA Configurations*, CARMA Memo 46.

Rauch, K.P., and Salter, D.M. 2012, *Blank Sky Analysis and Statistics*, CARMA Memo 57.