The hardware interfaces of the CCB

Martin Shepherd, California Institute of Technology

January 22, 2003

This page intentionally left blank.

Abstract

This document specifies the operational characteristics of the CCB hardware interfaces.

Contents

List of Figures

1 A definition of terms

The following terms are used throughout this document. It is recommended that in other documents that describe both the hardware and software, these terms be prefixed with the word *hardware* when referring to the FPGA.

- **A sample**
  A single A/D sample.
- **A measurement**
  The group of samples added within a single phase-switch state of a single cycle.
- **A cycle**
  A single phase-switch cycle is composed of between 1 and 32 samples, with each sample individually programmable to have 1 of 4 possible phase switch states.
- **integration period**
  The integer number of cycles who's measurements are to be coadded to form integrations.
- **An integration**
  The data accumulated for a single phase switch state during an integration period.
- **A scan**
  A group of one or more integrations associated with a single set of scan-specific configuration parameters.

2 Data-type conventions

2.1 Integer parameters

Integer valued parameters presented over the PCI bus are unsigned and arranged in little-endian byte order.

2.2 Boolean parameters

Boolean registers presented over the PCI bus are represented as single bits within larger data-types. The ordering of the bits within these data-types is defined such that performing a logical shift right by 1 bit, results in the successively next least significant bit appearing on the carry line of the shift register. The meanings of the values of these bits are as follows:

	0	1
Boolean	false	true
Switch state	off	on

Note that when referring to phase-switches, **off** corresponds to when the phase-shifter isn't inserting any phase-shift into the signal path, and **on** means that it is inserting a 180° phase shift.

3 The PCI interface between the computer and the FPGA

3.1 Configuration registers

With the exception of the control register, which is discussed in the next section, the values of all of the following registers, are examined by the FPGA just before starting each new integration, and are therefore written by the device driver in advance of the start of each new integration.

The device driver adopts the following strategies to ensure that the values of these registers are updated in advance of each new integration.

- The FPGA sends the device driver an interrupt just after ending one integration and loading the configuration of the following integration. On receiving this interrupt, the device driver updates the configuration registers with the configuration of the integration that follows the one that is just being started.

- Whenever the FPGA is reset, the device driver writes the configuration of the first integration before interrupts are enabled. Since the FPGA doesn't start its state machine until interrupts are first enabled, this ensures that the parameters of the first integration are configured before the FPGA loads them.

- When directing the FPGA to start a new scan, the driver first clears the start-scan bit in the control register, updates the configuration registers with the configuration of the first integration of the scan, then allows the new scan to start by asserting the start-scan bit.

Note that in practice, very few of the configuration registers change from one integration to the next. In fact, except at the start of a new scan, usually only the calibration diode registers change from one integration to the next.

Name	Addr	Description	Qty	Multiplier	Default	Range
control-register	40	The control register	32	1 bit	0	0 or 1
samples-per-cycle	44	Samples per cycle	1	1 sample	1	1-32
phase-switch-a	48	Phase switch A states	32	boolean	0	0 or 1
phase-switch-b	4c	Phase switch B states	32	boolean	0	0 or 1
cycles-per-integ	50	Integration period	1	1 cycle	40	1-65535
long-sample-dt	54	Long sample interval	1	100ns	250	1-65535
short-sample-dt	58	Short sample interval	1	100ns	250	1-65535
phase-switch-dt	5c	Phase-switching delay	1	100ns	0	1-255
analog-reset-dt	60	Integrator-reset delay	1	100ns	0	1-255
cal-diode-states	64	Cal diode on/off states	2	boolean	0	0 or 1
cal-diode-dt	68	Cal diode settling time	1	100ns	0	1-$2^{32}-1$

Note that in this table the *Addr* column specifies the hexadecimal address offset of each register with respect to the PCI base address of the CCB board. Since the PCI interface standard reserves the first 64 bytes following the base address for its own configuration header, the first CCB-specific register starts at address 0x40.

Also note that when refering to boolean or other single-bit sub-registers, the *Qty* column specifies how many of the least significant bits of the target register are used for this purpose.

3.1.1 Detailed descriptions

- **The control register** (control-register)

  This register is effectively many single-bit registers embedded in one 32-bit register. The section following this one is dedicated to discussing the function of each of these bits.

- **Samples per cycle** (samples-per-cycle)

  The states of the phase-switch within each stage of the phase-switch cycle are configured via a 32 stage state-machine, advanced by one state every time that a new sample is taken by the A/D. In practice only the first samples-per-cycle of these states is used. Thus the number of samples in a cycle is defined by this register.

  Note that while the direct specification of a 32-state state machine allows a great deal of flexibility, in practice the states are filled by the device driver from much higher level configuration parameters, such as the number of samples per measurement, the selection of one of 3 phase-switching modes (ie. no phase-switching, 1-diode phase-switching, 2-diode phase-switching), and the specified initial states of the phase switches at the start of each cycle.

- **The state of phase switch A** (phase-switch-a)

  Each bit of this 32-bit register specifies the state of phase-switch A in both radiometers, at one stage of the phase-switching state machine. This is designed to be loaded into a

32-bit shift register at the start of each phase-switch cycle, then right shifted just before the start of each new sample, to determine the required state of the phase switches in that sample. The number of shifts required to complete a cycle is determined by the samples-per-cycle register described above.

- **The state of phase switch B** (phase-switch-b)

  This register is the same as the phase-switch-a register, except that it is applied to phase-switch B instead of to phase-switch A.

- **Integration period** (cycles-per-integ)

  This is the integer number of complete phase-switch cycles who's measurements are to be integrated, before presenting the data to the computer. It was decided at the preliminary design review that 1ms is the minimum integration time that needs to be guaranteed. In practice, shorter times may also be possible, depending on the speed of the computer, the bandwidth of the PCI bus, and the bandwidth of the communications link between the CCB server and the CCB manager.

- **Long sample interval** (long-sample-dt)

  This is the time spent integrating individual A/D samples that aren't preceded by phase switch transitions, expressed in multiples of the 100ns FPGA clock. This interval does not include the time spent resetting the analog integrators.

  Although this interval is widely configurable, the analog electronics are designed for a sampling interval of 25μs, so longer intervals may result in the A/D saturating, and much shorter intervals may not be possible with the chosen ADC.

- **Short sample interval** (short-sample-dt)

  This is the time spent integrating A/D samples that are preceded by phase switch transitions, expressed in multiples of the 100ns FPGA clock. The device driver gives this register the value,

  $$\text{short-sample-dt} = \text{long-sample-dt} - \text{phase-switch-dt} \tag{1}$$

  As explained later, in cycles that don't contain any phase-switch transitions, phase-switch-dt is zero, so the value of short-sample-dt register is then be equal to long-sample-dt.

- **Phase switch blanking interval** (phase-switch-dt)

  This is the settling time needed by the phase-switches in the front-end whenever they change state. The FPGA inserts this delay, both whenever it changes the states of the phase switches, and for the first sample of each cycle. It is important to understand the interaction between this register and the phase-switch-b and phase-switch-a registers, since it is possible to generate flawed cycles. The basic constraints are:

1. The total integration time spent in each phase-switch state of a cycle must be identical. Otherwise the equations governing the operation of the differential radiometer would be broken.
2. Settling time delays must be inserted as needed to prevent phase-switch transitions corrupting the integrated data.
3. As few delays should be included as possible, to avoid wasting integration time.

The main difficulty in complying with these constraints is that the phase-switch states are un-predictable to the driver for the first sample of each new scan, and completely unknown by the FPGA before the first scan after a FPGA reset (eg. at power-on). Thus if one just went by the rule that delays should be included at phase-switch transitions, there might or might not be a delay removed from the first sample of a new cycle, and in the first scan there might not be a delay included when a delay was needed. This would randomly break the first and second of the constraints listed above. The way to correct these problems depends on the type of phase-switching cycle being commanded, as follows.

The basic approach is to require an equal number of samples in each of the phase-switch states of the cycle, and require that all samples within a given phase-switch state are contiguous in the cycle. This requirement is enforced by the device driver. The consequences of this strategy, and the minor adjustments that are required to work around its deficiencies are as follows.

- In cycles that include at least one phase-switch transition, the above strategy ensures that there is exactly one phase-switch transition per phase-switch state, and thus that the same amount of time is lost to phase-switching in each phase-switch state. A exception to this rule is at the start of a scan, where this strategy might or might not cause a phase-switch delay to be included, depending on what state the previous scan left the phase-switches in. For this reason, the FPGA state machine always includes a phase-switching delay at the start of each cycle, irrespective of whether a phase-switch transition occurred or not.
- In cycles where the phase-switch states are held in a single unchanging state, there are no phase-switch transitions, and thus no need for phase-switching delays. Since the FPGA always includes a phase-switch delay at the start of each cycle, it is thus necessary for the device-driver to set phase-switch-dt to zero for unswitched cycles. Again, depending on what state the previous scan left the phase-switches in, there may need to be a phase-switch transition before the first cycle can procede. In order not to break the first of the constraints, this delay needs to be removed before the first cycle starts, rather than being subtracted from the first sample of the first cycle. To handle this, the device driver lengthens, if necessary, the value of cal-diode-dt of the first integration of the scan, to include one phase-switch transition delay.

- **Integrator reset blanking interval** (analog-reset-dt)

This specifies how much time is needed to reset the analogue integrators to zero, expressed in multiples of the 100ns FPGA clock. The choice of this number is determined by the current-handling capacity of both the capacitors and the FET switches that are used to discharge them, and should be around 0.5μs. This delay is inserted after each A/D sample.

- **Cal diode states** (cal-diode-states)

  The 2 least-significant bits of this register specify the states that the noise diodes are to be set to at the start of the next integration period. A value of 1 means turn a diode on; a value of 0 means turn it off. This is updated by the device driver from the integration-done ISR.

- **Calibration diode settling time** (cal-diode-dt)

  The calibration diodes are potentially switched at the start of each new integration period. This register specifies how long the hardware must wait after commanding the diodes to their new states, before starting the first sample of the following integration. Its value, which is set by the integration-done ISR before each integration, and is expressed in multiples of the 100ns FPGA clock, is determined according to the following rules.

  – If neither calibration diode is changing state, then the driver writes zero in this register.

  – If one or both of the diodes is being switched on, and neither are being switched off, then the driver writes the cal-diode rise time in this register.

  – If one or both of the diodes is being switched off, and neither are being switched on, then the driver writes the cal-diode fall time in this register.

  – If one of the diodes is being switched off, and the other is being switched on, then the driver writes the longest of the cal-diode rise and fall times in this register.

  – If the integration is the first integration of a scan, then the driver writes the longest of the phase-switch and cal-diode rise and fall times in this register.

  Note that the 32-bit range of this register is bigger than that of any other timer. This is due to the uncertainty over how long the diodes need to settle to a level suitable for astronomical measurements.

3.2 The control register (control-register)

This register is a 32-bit register who's individual bits set flags in the hardware. The first thing that the FPGA does after being reset, is to zero the bits in this register. It then waits for the driver to set the interrupt-enable bit in this register before starting its state machine. This gives the driver the chance to initialize the configuration registers to suitable values for the first integration of the first scan.

Bit	Name	Description	Effect
0	start-scan	Start a new scan	Start a new scan
1	wait-1pps	Scan starts on 1-PPS	Synchronize scan starts with 1-PPS?
2	reset-fpga	Reload firmware	Reload the FPGA firmware from EPROM
3	interrupt-enable	Enable interrupts	Enable 1-PPS and integration interrupts
4	drive-phs-a	Drive phase-switch A	Drive phase-switch A's control line
5	drive-phs-b	Drive phase-switch B	Drive phase-switch B's control line
6	drive-cal-a	Drive cal-diode A	Drive cal-diode A's control line
7	drive-cal-b	Drive cal-diode B	Drive cal-diode B's control line

The operations listed in the above table are all asserted by setting the corresponding bits to 1 (ie. positive assertion logic). The response of the hardware to each of these bits is detailed below.

3.2.1 Detailed descriptions

- **Start scan** (start-scan)

  Clearing this bit immediately resets the integration state machine in the FPGA, thus returning it to its starting state. Subsequently asserting this bit starts the integration state machine. Between clearing and asserting this bit, the device driver sends the FPGA the configuration of the first integration of the scan.

  When the device driver finishes writing the new configuration, it asserts the start-scan bit. The FPGA then latches the integration configuration registers and starts to drive both the cal-diode and the phase-switch lines to their initial states. It then examines the wait-1pps bit of the control register, and if this is asserted, the FPGA waits for the next rising edge of the 1-PPS signal before proceding with the integration state machine.

  Within the device driver, this bit is used to implement the start-data-scan and start-intra-scan commands, as follows.

  start-data-scan:

  1. Clear the start-scan bit in the control register.
  2. Assert the wait-1pps bit in the control register.
  3. Update the hardware configuration registers to configure the first integration.
  4. Wait for the next 1-PPS interrupt.
  5. If the timestamp of the 1-PPS is more than one second before the requested start time, go back to step 4.
  6. Assert the start-scan bit in the control register.

  start-intra-scan:

1. Clear the start-scan bit in the control register.
2. Deassert the wait-1pps bit in the control register.
3. Update the hardware configuration registers to configure the first integration.
4. Assert the start-scan bit in the control register.

Since the states of the calibration diodes and the phase-switches aren't known by the device driver at the instant that the start-scan bit is cleared, and the timing of integrations needs to be predictable to the manager, the device driver sets the value of the cal-diode-dt register for the first integration to the longest of the phase-switch and cal-diode rise and fall times. This causes the FPGA to wait for this amount of time before starting to integrate the first sample, and thus gives the calibration diodes and the phase-switches time to settle.

- **Reload FPGA firmware** (reset-fpga)

  This bit is asserted by the driver at startup, and can also be used during initial debugging to reload the FPGA program from EPROM after a crash. After a reload, the FPGA zeroes the control register, then waits for the device-driver to assert the interrupt-enable bit of this register before initiating the first scan.

- **Enable interrupts** (interrupt-enable)

  Whenever the FPGA is reset, the generation of 1-PPS and integration interrupts is initially disabled. This bit is used to enable them once the driver is in a fit state to respond to them. Subsequently setting it to zero simply stops interrupts from being generated, without affecting the rest of the state machine.

- **Drive phase-switch A** (drive-phs-a)

  This bit turns on the output control line which drives phase-switch A.

- **Drive phase-switch B** (drive-phs-b)

  This bit turns on the output control line which drives phase-switch B.

- **Drive cal-diode A** (drive-cal-a)

  This turns on the output control line which drives calibration diode A.

- **Drive cal-diode B** (drive-cal-b)

  This turns on the output control line which drives calibration diode B.

3.3 Diagnostic monitoring registers

These registers, which are stored in PCI DMA memory, are updated by the hardware just before the integration-done interrupt is sent. The device driver thus reads them out on receipt of this interrupt.

Description	Qty	Units	Range
Power supply voltages	N/A	mv	N/A
A/D converter outputs	16	A/D sample	$0 \cdots 2^{32} - 1$

The ordering of the array of 16 A/D converter outputs is as follows.

Array Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Detector	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2
Band	1 . . .		2 . . .		3 . . .		4 . . .		1 . . .		2		3		4	
Radiometer	1								2							

3.3.1 Detailed descriptions

- **Power supply voltage**

  This records the voltages of all power supplies in the system. 8 bits should provide ample precision for these values.

- **A/D converter outputs**

  The last values read-out from the 16 A/D converters. This can be used to check for failing A/D converters and A/D saturation. Like the A/D converters themselves, these are 16-bit values.

3.4 Data output via DMA

Like the diagnostic monitoring registers, these registers are stored in PCI DMA memory, and are updated by the hardware just before the integration-done interrupt is sent. The device driver thus reads them from DMA memory on receipt of this interrupt.

Description	Qty	Units	Range
An integrated data value	64	–	$0 \cdots 2^{32} - 1$
Overflow mask	64	boolean	0 or 1

The overflow mask uses 64 bits of an array of 8 bytes to record which of the 64 data values suffered overflows during integration. Bits are numbered from zero, starting with the least significant bit of the highest memory location, and ending at 63 with the most significant bit of the lowest memory location.

The array of integrations is arranged as an array of 32-bit little-endian integers, indexed from zero, starting with the element in the lowest memory location.

The order of assignment of the 64 values and their corresponding overflow bits to array indexes and bit indexes is as follows.

Array index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Integration	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2
Detector	1				2				1				2				1	
Band	1								2								3	
Radiometer	1																	

3.5 Interrupts

The following interrupts share a single interrupt line.

Description	When the interrupt
Integration interrupt	When one integration ends and another has just been started.
1-PPS interrupt	On rising edges of the 1-PPS signal

3.5.1 Detailed descriptions

- **Integration interrupt**

  This interrupt serves two purposes.

  1. It tells the driver that the results of the last integration period have just been written to the DMA memory area, and that the driver should read them ASAP.
  2. It tells the driver that the configuration registers should be updated to configure the integration that follows the one that has just been started.

  Immediately before generating this interrupt, the FPGA sets the sent-integ-done register to 1.

- **1-PPS interrupt**

  Whenever a rising edge of the 1-PPS signal is seen, the FPGA first sets the 1-PPS sent-1pps register, then generates this interrupt.

3.6 The interrupt status registers

The following interrupt status registers are checked by the device driver whenever it receives an interrupt. It uses them to check which event or events generated the interrupt, and responds accordingly. Note that to prevent race conditions in the device driver when it comes to resetting these registers, it is essential that they be separate registers, rather than individual bits within a single register.

Name	Description
sent-1pps	1-PPS interrupt sent
sent-integ-done	End-of-integration interrupt sent

3.6.1 Detailed descriptions

- **End-of-integration interrupt sent** (sent-integ-done)

  Immediately before the FPGA sends an integration interrupt, it sets this register to 1. The device-driver's interrupt handler subsequently resets it to zero.

- **1-PPS interrupt sent** (sent-1pps)

  Immediately before the FPGA sends a 1-PPS interrupt, it sets this register to 1. The device-driver's interrupt handler subsequently resets it to zero.

4 John's FPGA state diagrams

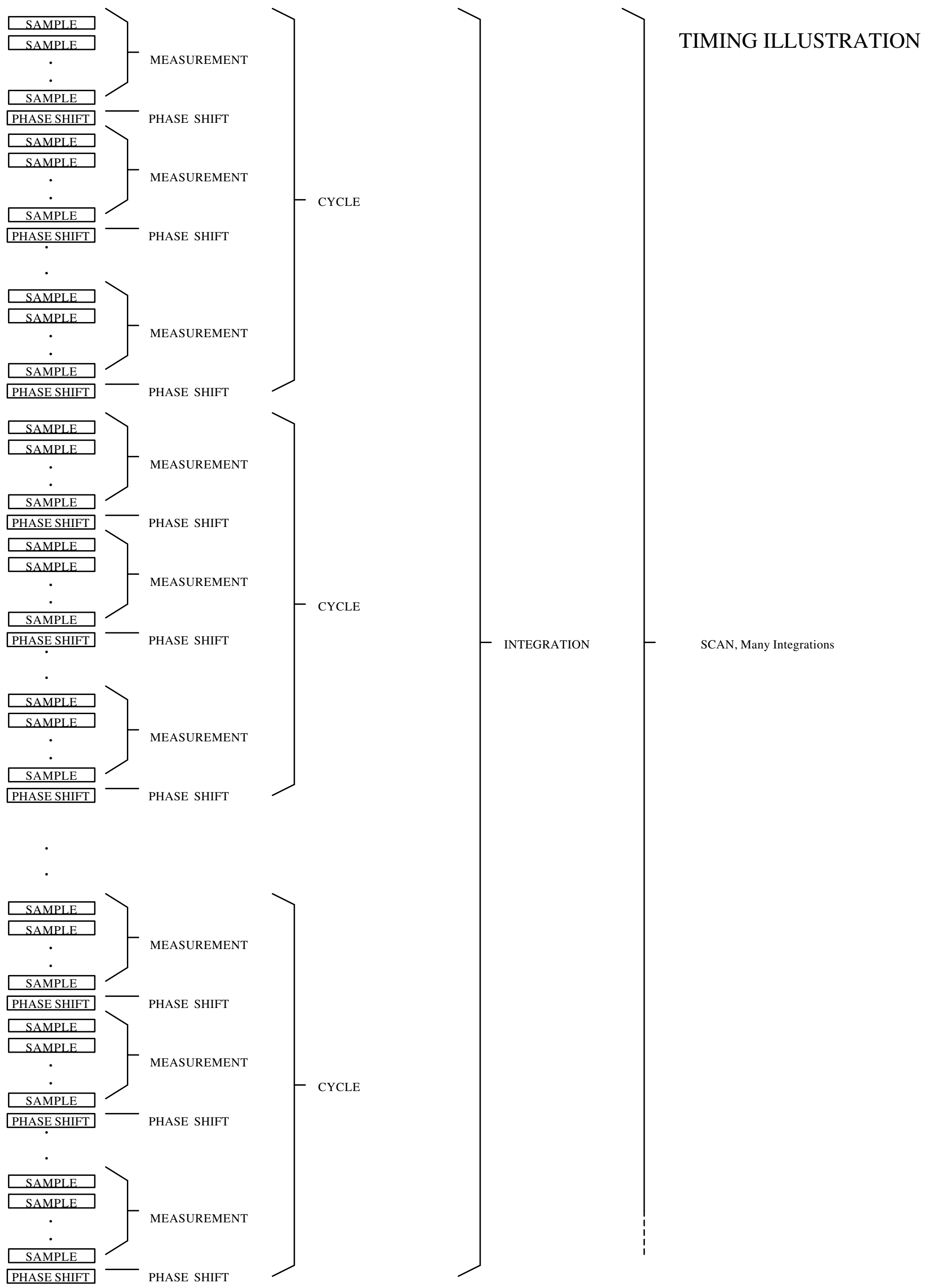
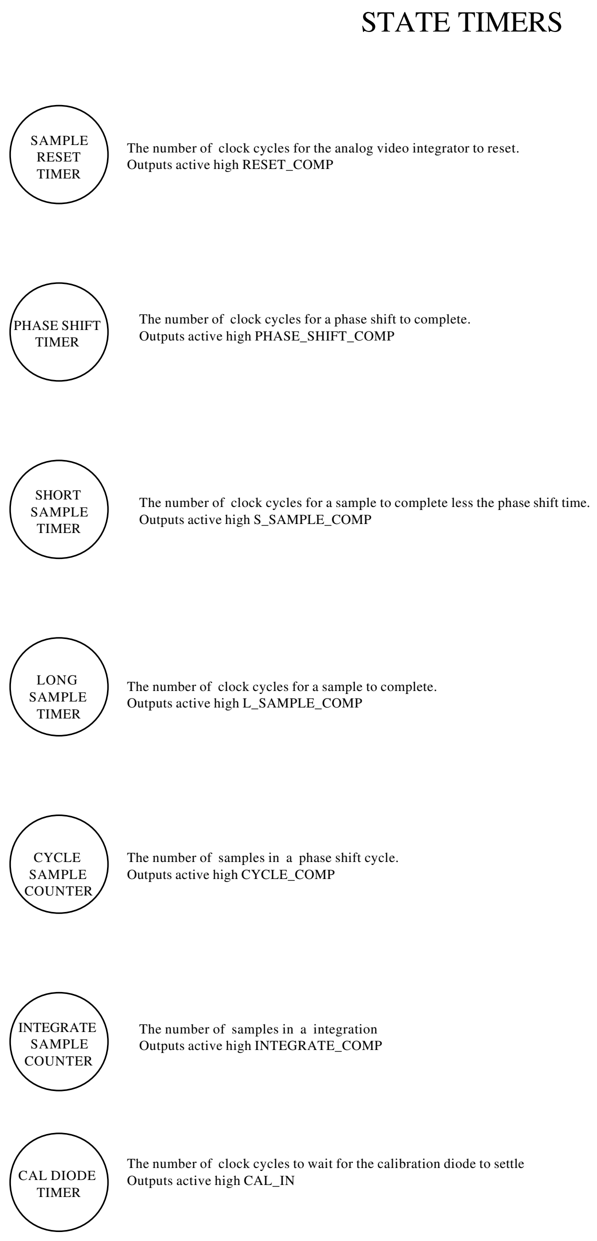


Figure 1: A summary of the structure of a scan



The number of clock cycles for the analog video integrator to reset.
Outputs active high RESET_COMP

The number of clock cycles for a phase shift to complete.
Outputs active high PHASE_SHIFT_COMP

The number of clock cycles for a sample to complete less the phase shift time.
Outputs active high S_SAMPLE_COMP

The number of clock cycles for a sample to complete.
Outputs active high L_SAMPLE_COMP

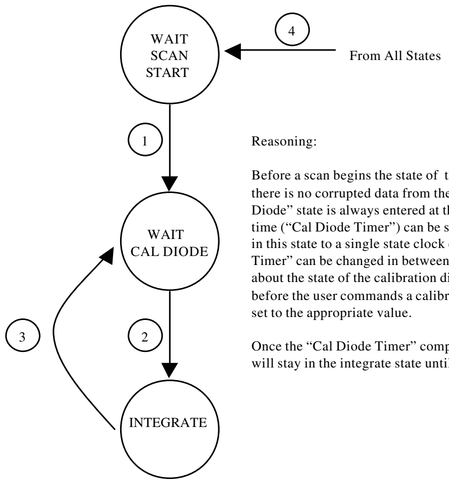
The number of samples in a phase shift cycle.
Outputs active high CYCLE_COMP

The number of samples in a integration
Outputs active high INTEGRATE_COMP

The number of clock cycles to wait for the calibration diode to settle
Outputs active high CAL_IN

Figure 2: The hardware timers

INTEGRATE
STATE MACHINE



Reasoning:

Before a scan begins the state of the calibration diodes is unknown. To ensure that there is no corrupted data from the calibration diodes settling in or out a "Wait Cal Diode" state is always entered at the beginning of a scan. The register controlling this time ("Cal Diode Timer") can be set to zero which would minimize the time that is used in this state to a single state clock cycle. This is possible because the "Cal Diode Timer" can be changed in between integrations. So if the user knows or doesn't care about the state of the calibration diodes then the timer could be set to zero and then before the user commands a calibration diode integration the "Cal Diode Timer" can be set to the appropriate value.

Once the "Cal Diode Timer" completes its count integrations can begin. The system will stay in the integrate state until a calibration is requested or a start of a new scan.

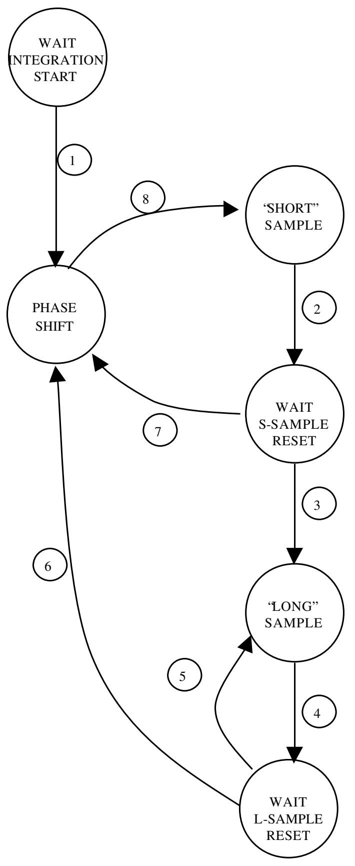
State Definitions:

WAIT SCAN START	Wait here until a scan start comes
INTEGRATE REG SOURCE	Integrating on source
WAIT CAL DIODE	Wait here until the Calibration Diode is settled IN or Out

STATE TRANSITIONS:
1. (START_SCAN &!START_1PPS) # (START_SCAN & START_1PPS & 1PPS)
2. CAL_DIODE_SETTLED
3. INT_COMPLETE & CAL_DIODE_CHANGE
4. !START_SCAN

Figure 3: The integration state machine

SAMPLE STATE
MACHINE



Reasoning:

At power up or before a scan begins the state machine waits for an INTEGRATE command generated by the integrate state machine. On reset or power up the machine will enter the WAIT INTEGRATION START and remain in this state until an INTERGRATE command is given. The number of 100ns periods that each state takes is controlled by the associated timer.

From the WAIT INTEGRATION START state the PHASE SHIFT state is entered to allow for any phase shift to complete. Waiting for phase shift to complete at the beginning of a scan or after a power up is necessary since the state of the phase shifters is unknown.

Depending on the phase cycle the PHASE SHIFT state will be entered from a either the SHORT or LONG sample reset states. A comparator is used to evaluate the current phase state with the next phase state generating the PHASE SHIFT signal indicating that a phase shift delay is required.

Once a phase shift delay is completed the state machine will enter the SHORT SAMPLE state. This is the"SHORT" integration time of the anal og integrator before the analog to digital conversion.

After SHORT SAMPLE state the machine enters the WAIT S-SAMPLE RESET. This is the time that is needed to reset the analog integrator before taking a new sample.

If a phase shift is indicated the PHASE SHIFT state as previously described will be entered.

If no phase shift is required and a integration hasn't been comp leted then the LONG SAMPLE state is entered. This is the"LONG" integration time of the analog integrator before the analog to digital conversion.

After LONG SAMPLE state the machine enters the WAIT L-SAMPLE RESET. This is the time that is needed to reset the analog integrator before taking a new sample.

From the WAIT L-SAMPLE RESET state the machine can either loop back to take another LONG SAMPLE or a PHASE SHIFT can be inserted.

After either SAMPLE RESET states a integration complete or phase cycle complete will force the machine to the PHASE SHIFT state regardless if a phase shift is required. This is necessary to ensure that all cycles integrations are equal in the amount of actual integration.

PHASE SHIFT state is entered regardless if there actually is a phase shift
Long and short sample times are necessary since a phase shift is not always required between analog integrations. With short and long analog integrations, overall integration time is maximized.

State Definitions:
WAIT INTEGRATION START--Wait here for the integration to start
"SHORT SAMPLE" ----------------Sample time minus time for phase shift
WAIT S-SAMPLE RESET--------Wait for reset time for "SHORT" sample.
PHASE SHIFT-----------------------Time for phase shift to complete
"LONG" SAMPLE ------------------Sample time no phase shift between samples
WAIT L-SAMPLE RESET---------Wait for reset time for "LONG" sample

State Transitions:
1.INTEGRATE
2.S_SAMPLE_COMP
3.RESET_COMP & ! PHASE_SHIFT
4.L_SAMPLE_COMP
5.RESET_COMP & ! PHASE_SHIFT
6.RESET_COMP & (PHASE_SHIFT # PHASE_CYCLE_COMP # INTEGRATE_COMP)
7.RESET_COMP & (PHASE_SHIFT # PHASE_CYCLE_COMP # INTEGRATE_COMP)
8.PHASE_SHIFT_COMP

Figure 4: The sample state machine