

OpenCPI HDL Infrastructure Specification

Authors:

Shepard Siegel, Atomic Rules LLC (Shepard.Siegel@atomicrules.com)

Jim Kulp, Mercury Federal Systems, Inc. (jkulp@mercfed.com)

Revision History

Revision	Description of Change	By	Date
1.01	Creation from previous internal documents	jkulp	2010-07-01

Table of Contents

1	References.....	4
2	Overview	5
3	PCIe Block Plus Endpoint (BPEP)	6
3.1	Configuration v1.13 (ISE11.4).....	6
4	uNOC: PCIe Multiple Access Provisioning	8
5	Control Plane (CP).....	9
5.1	BAR0 – 16 MB Memory for Administration and Workers	9
5.2	BAR0 – Administrative “Admin” Region Registers.....	10
5.3	BAR0 – Worker Control Operation Space	15
5.4	BAR0 – Worker Configuration Properties Space	18
6	Data Plane Unit (DPU)	20
6.1	BAR1 – 64KB Memory Space for Data Movement	20
6.2	Configuration Properties	21
7	HDL Time Service (HTS).....	26
7.1	How the HTS is used with Workers that desire Time	26
7.2	HTS Time Server Configuration Properties	27
7.3	Using the HTS Time Server.....	27

1 References

This document depends on several others. Primarily, it depends on the “OpenCPI Generic Authoring Model Reference Manual”, which describes concepts and definitions common to all OpenCPI authoring models, and the “OpenCPI HDL Authoring Model Reference” which describes how HDL workers must be written.

Title	Published By	Link
OpenCPI Generic Authoring Model Reference [AMR]	OpenCPI	Public URL: http://www.opencpi.org/doc
OpenCPI HDL Authoring Model Reference [HDL]	OpenCPI	Public URL: http://www.opencpi.org/doc

2 Overview

This document describes the OpenCPI infrastructure IP blocks and how they are used as the platform for OpenCPI applications on FPGAs. The on-chip architecture for OpenCPI defines a stack of IP/Gateway with the blocks relating to external (off-chip via pins) hardware at the bottom and application workers on top. As seen below, the bottom layer is divided between “platform” elements that are always required, and “device” elements that are present to support certain external hardware devices when the application needs them. This division (platform, devices) corresponds to the software notions of “hardware abstraction layer” (under an operating system kernel), and “device drivers.

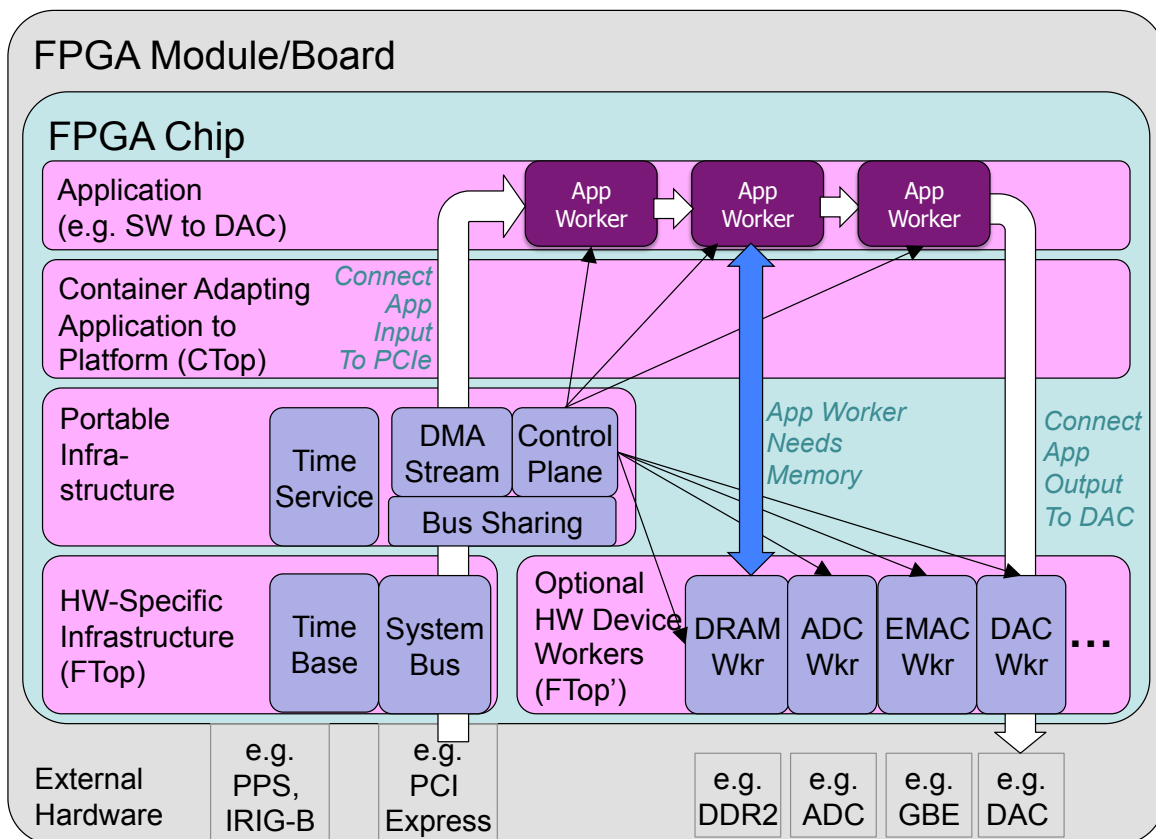


Figure 1: HDL Stack for OpenCPI on an FPGA

The white arrow above is simply a notional application data flow, with data arriving via a stream over the PCI Express fabric, and existing via a DAC device. Thus the DAC worker is a device worker that is included because the application needs it, while the PCI Express interface (called “system bus”) is always included because it is needed as part of the “platform”. Similarly, the blue arrow shows that the second “app worker” needs memory, and thus the DRAM worker is included in the design to support that need. The “control plane” is shown controlling both these “device workers” as well as the “application workers”. Not shown are other internal infrastructure elements that are also controlled by the control plane.

3 PCIe Block Plus Endpoint (BPEP)

The PCI Express fabric is frequently used as the system bus when software talks to processing FPGAs in an embedded system. This block is the configured/generated Xilinx block that allows OpenCPI FPGA applications to use this fabric. It is specifically configured for use by OpenCPI, and enables both control and dataplane traffic between the FPGA and software (or other FPGAs). It is represented by the “system bus” block in Figure 1. The control plane PCIe traffic is always between software and the HDL infrastructure, device, and application blocks on the FPGA. The data plane PCIe traffic is both between SW and FPGAs, as well as for peer-to-peer data plane traffic between different FPGAs attached to the PCI Express fabric.

The BPEP module is the Xilinx-supplied IP block that wraps their hard-core PCIe with an interface for use by FPGA applications. In this case the OpenCPI system bus block further (lightly) wraps the BPEP in order to normalize it for OpenCPI so that other vendors’ PCIe blocks can be used easily.

3.1 Configuration v1.13 (ISE11.4)

The table below lists the changes from default for the BPEP not related to the PCIe BARs (Bus Address Registers/Windows). There are several configurations used by OpenCPI for various hardware platforms (boards and chips).

Table 1 – Virtex5 PCIe Gen1 Parameters

Parameter	Value	Value
lane_width [1]	x8, x4, x1	x8, x4, x1
interface_freq	125 MHz	250 MHz
device_id	16'h_4243	16'h_4243
revision_id	02	03

Table 2 – Virtex6 PCIe Gen2 Parameters

Parameter	Value
lane_width	x4
interface_freq	250 MHz
device_id	16'h_4243
revision_id	02

Table 3 –Spartan6 PCIe Gen2 Parameters

Parameter	Value
lane_width	X1
interface_freq	250 MHz
device_id	16'h_4243
revision_id	02

[1] lane width set per board target.

BAR0 and BAR1, the two PCIe slave bus windows enabled for use by OpenCPI, are set to the following parameters; the other BARs are disabled. BAR0 is used for the control plane, and BAR1 is used for the data plane.

BAR0	BAR1
bar0_64bit=false bar0_enabled=true bar0_prefetchable=false bar0_scale=Megabytes bar0_size=16 bar0_type=Memory bar0_value=FF00_0000	bar1_64bit=false bar1_enabled=true bar1_prefetchable=false bar1_scale=Kilobytes bar1_size=64 bar1_type=Memory bar1_value=FFFF_0000

4 uNOC: PCIe Multiple Access Provisioning

The uNOC (micro-Network-On-Chip) provides for multiple independent functions sharing the PCIe interface (via the BPEP described above). This multiple access applies to both master and slave use of the PCIe. In the OpenCPI FPGA infrastructure the Control Plane block (one per FPGA) and Data Plane blocks (several per FPGA) all share the PCIe; the uNOC block is what implements this sharing.

This block corresponds to the “bus sharing” block in Figure 1. It can also be seen in the diagram below, where the “CP”, and multiple “DP” blocks all share the PCI Express fabric (via the BPEP). In this case the “OCInf” block (blue rectangle) contains portable infrastructure used in different hardware designs (boards and chips).

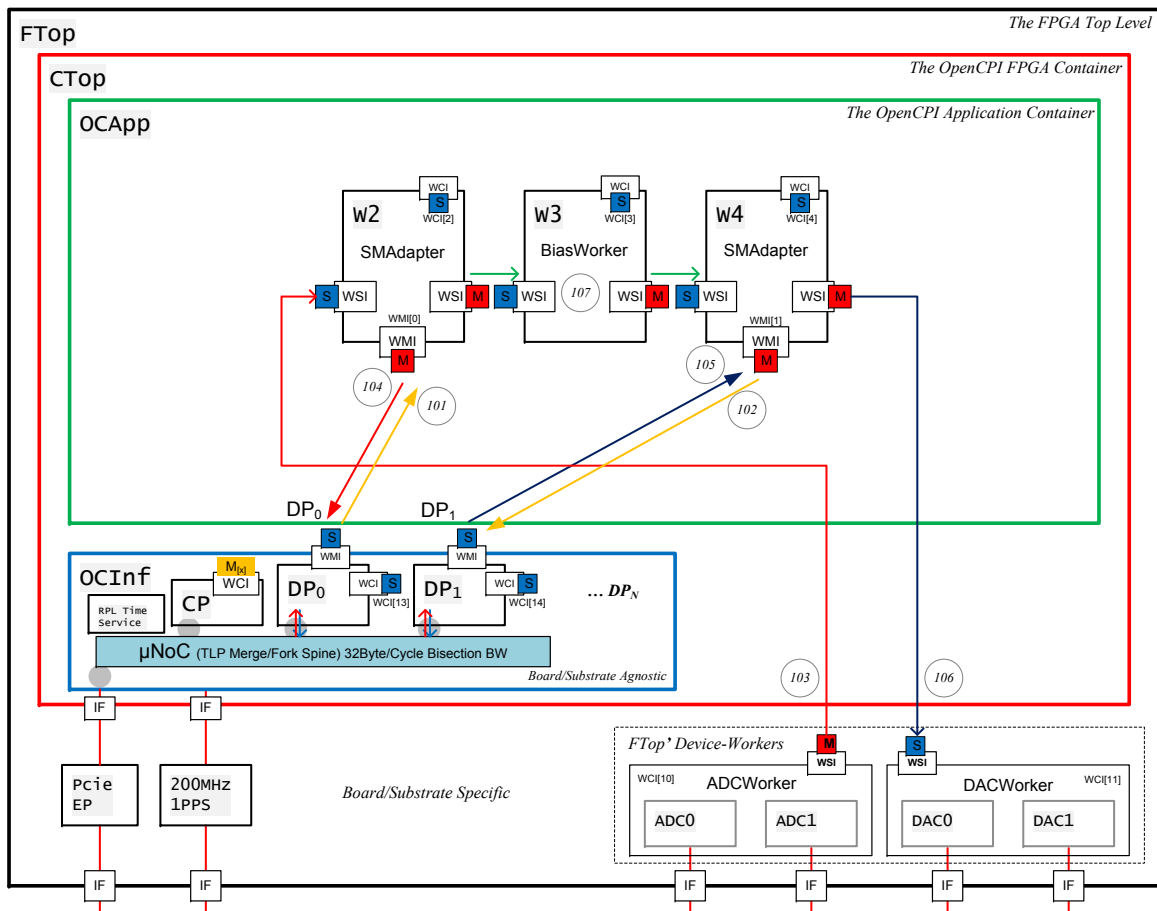


Figure 2: Typical OpenCPI FPGA contents

5 Control Plane (CP)

The Control Plane module presents various controllable functions via BAR0 memory mapped accesses via the uNOC and the BPEP. On the “back side” of the block, it surfaces 15 WCI master interfaces to control all three types of workers in the FPGA:

- Infrastructure blocks like the Data Plane units
- Device Workers like DRAM, ADC, and DAC.
- Application workers

The WCI interface is described in detail in [HDL]. Beyond providing for WCI-based control and configuration of other blocks, it has other administrative functions including:

- Controlling the Time Server, which is internal to the CP block, and supporting WTI time clients.
- Declaring Memory Regions used as buffer memory by the DPs.

5.1 BAR0 – 16 MB Memory for Administration and Workers

BAR0 is a 16MB space that is used for accessing a 1 MB Administrative area (for the CP itself), as well as fifteen identical 1MB Worker Configuration Property areas. The table below shows how the top four byte-address bits[23:20] of the 24b byte address in BAR0 are used to select one of the sixteen 1MB regions.

The 1MB Administrative (“Admin”) region contains a 64KB area that can affect the entire FPGA; along with fifteen identical 64KB worker control regions, which provide Control (and Status) of individual workers. Each of the 15 Worker Configuration Property regions provides a full, contiguous 1MB memory map to each worker’s configuration property space. The address map for BAR0 is below:

BAR0 Offset	Size	Region	In oc1001 bitstreams	In oc1003 bitstreams
+0x00_0000	64KB	Administrative (“Admin”)	admin	admin
+0x01_0000	64KB	Control, Worker 0	Gcd	
+0x02_0000	64KB	Control, Worker 1	Gcd	
+0x03_0000	64KB	Control, Worker 2	FCAdapter	SMAadapter
+0x04_0000	64KB	Control, Worker 3	Bias	Delay
+0x05_0000	64KB	Control, Worker 4	FPAdapter	SMAadapter
+0x06_0000	64KB	Control, Worker 5		
+0x07_0000	64KB	Control, Worker 6		
+0x08_0000	64KB	Control, Worker 7		
+0x09_0000	64KB	Control, Worker 8		EMAC
+0x0A_0000	64KB	Control, Worker 9		Flash
+0x0B_0000	64KB	Control, Worker 10	ADC	ADC

+0x0C_0000	64KB	Control, Worker 11	DAC	DAC
+0x0D_0000	64KB	Control, Worker 12	FtpUtil	DRAM
+0x0E_0000	64KB	Control, Worker 13	DP0	DP0
+0x0F_0000	64KB	Control, Worker 14	DP1	DP1
+0x10_0000	1MB	Configuration, Worker 0	Gcd	
+0x20_0000	1MB	Configuration, Worker 1	Gcd	
+0x30_0000	1MB	Configuration, Worker 2	FCAdapter	SMAadapter
+0x40_0000	1MB	Configuration, Worker 3	Bias	Delay
+0x50_0000	1MB	Configuration, Worker 4	FPAdapter	SMAadapter
+0x60_0000	1MB	Configuration, Worker 5		
+0x70_0000	1MB	Configuration, Worker 6		
+0x80_0000	1MB	Configuration, Worker 7		
+0x90_0000	1MB	Configuration, Worker 8		EMAC
+0xA0_0000	1MB	Configuration, Worker 9		Flash
+0xB0_0000	1MB	Configuration, Worker 10	ADC	ADC
+0xC0_0000	1MB	Configuration, Worker 11	DAC	DAC
+0xD0_0000	1MB	Configuration, Worker 12	FtpUtil	DRAM
+0xE0_0000	1MB	Configuration, Worker 13	DP0	DP0
+0xF0_0000	1MB	Configuration, Worker 14	DP1	DP1

In addition to 32b DWORD access, the BAR0 hardware respects multi-DWORD transactions (e.g. 64b 2-DW, or DMA N-DW) to this BAR.

Not all Workers need be connected to the WCIs.

5.2 BAR0 – Administrative “Admin” Region Registers

This section describes the administrative region registers, the 64KB space at the base of BAR0.

Admin Region Offset	Name	Access	Description
+0x0000	ByteSwap(0x4F70656E)	RO	0x4F70656E = ASCII "Open"
+0x0004	ByteSwap(0x43504900)	RO	0x43504900 = ASCII "CPI" [1]
+0x0008	cpRevision	RO	Control Plane Revision
+0x000C	cpBirthday	RO	Control Plane Birthday
+0x0010	cpConfig	RO	Control Plane Configuration
+0x0014	pciDevice	RO	PCIe Device ID
+0x0018	wrkAttn	RO	Worker Attention Register
+0x001C	cpStatus	RO	Control Plane Status
+0x0020	scratch20	Read/Write	GP Scratch Register 0x20
+0x0024	scratch24	Read/Write	GP Scratch Register 0x24
+0x0028	cpControl	RW	Control Plane Control
+0x002C		RO	Reserved
+0x0030	rplTimeStatus	RO	Time Status Register
+0x0034	rplTimeControl	RW	Time Control Register
+0x0038	rplTimeMS	[2] RW (stage)	RPL Time Integer Seconds
+0x003C	rplTimeLS	[2] RW (commit)	RPL Time Fractional Seconds
+0x0040	rplTimeCompareMS	[2] RW (stage)	Time Compare Argument MS
+0x0044	rplTimeCompareLS	[2] RW (commit)	Time Compare Argument LS
+0x0048	rplTimeRefPerPps	RO	Number of Ref Clocks Per PPS
+0x4C~78			Reserved
+0x007C	numDPMemRegions	Read-Only	Number of DP Mem Regions
+0x0080- +0x00BC	dpMemRegion 0-15	Read-Only	DP MemRegion 0-15 Info

1. Note that the ASCII characters "CPI" are followed by a NULL, not a Space.
2. Locations 0x38, 0x40, and 0x48 are 8B (64-bit) locations where the LS DWORD must first be staged, and the MS DWORD performs the commit. On a little-endian (e.g. x86) platform, this will happen automatically for a 64b store over PCIe, where the first DWORD of the 8B write goes to the lower address; and the second DWORD of the 8B write goes to the higher address.

5.2.1 cpRevision (Admin Base +0x0008)

This register indicates control plane revision number.

5.2.2 cpBirthday (Admin Base +0x000C)

This register indicates when the bitstream was built. (32b POSIX time)

5.2.3 cpConfig (Admin Base +0x0010)

This register indicates in bits [14:0] which of 15 possible WCI interfaces have been connected in the application. A logic '1' in a particular bit position means that a worker is connected to that interface; a logic '0' means that there is no WCI interface present, and thus no worker. The number of logic '1's in this register is the number of workers connected. Bit 0 corresponds to Worker 0.

5.2.4 pciDevice (Admin Base +0x0014)

This register indicates PCI device ID in bits [15:0]

5.2.5 wrkAttn (Admin Base +0x0018)

This register indicates in bits [14:0] which workers have non-zero sticky bit status. There is a one-to-one correspondence between worker number and bit position. If a particular bit is set, further details may be found by reading that worker's Worker Status Register. This register provides the utility of a single read, which can indicate if any worker has a sticky bit set. Bit 0 corresponds to Worker 0.

5.2.6 cpStatus (Admin Base +0x001C)

Control plane admin register. Bits [3:0] hold the number of "rogue" TLP packets received from PCI Express since reset.

5.2.7 scratch20/24 (Admin Base +0x0020/24)

These registers may be used for DWORD R/W tests with no side effect.

5.2.8 cpControl(Admin Base +0x0028)

The bit fields in this register control the functions described in the table below:

Bits	Name	Access	Description
31:24		Reserved	0
23:16		Reserved	0
15:8		Reserved	0
7:0		Reserved	0

5.2.9 rplTimeStatus (Admin Base +0x0030)

The bit fields in this Time Status register provide the status described in the table below:

Bits	Name	Access	Description
31	ppsLostSticky	RO	1 = An active PPS fell outside the $\pm 0.1\%$ (1000 PPM) window
30	gpsInSticky	RO	1 = At least one valid GPS time has updated rplTime
29	ppsInSticky	RO	1 = At least one valid PPS input has aligned rplTime
28	timeSetSticky	RO	1 = At least one valid CP set has been issued to rplTime
27	ppsOK	RO	1 = PPS fell within the valid window in the last second
26	ppsLost	RO	1 = PPS went missing from the valid window in the last second
25:8		RO	0
7:0	rollingPPSIn	RO	8b rolling count of (external) PPS Events

5.2.10 rplTimeControl (Admin Base +0x0034)

The bit fields in this Time Control register control the functions described in the table below.

Bits	Name	Access	Description
31	clearStickyBits	WO	Writing a 1 to this location clears time sticky bits
30:5		Reserved	
4	disableServo	RW	0 = Use CP, PPS, or GPS to discipline local XO timebase 1 = Disable servo, local XO runs w/o compensation
3	disableGPS	RW	0 = Acquire GPS Time from (external) GPS 1 = Disable (external) GPS dialog attempts
2	disablePPSIn	RW	0 = Observe and Synchronize to (external) PPS Input 1 = Disable observation of (external) PPS Input
1:0	drivePPSOut	RW	2'h0 = Time Server Integer Seconds (1 Hz) 2'h1 = PPS Input Loop-through (PPS Hz) 2'h2 = Local XO Reference / 2 (100 MHz) 2'h3 = Mute (output disabled)

5.2.11 rplTime MS, LS (Admin Base +0x0038, 3C)

This 8B location provides a method for the control plane (CP) fabric (e.g. software) to set and observe the FPGA master chip clock. Time is represented in unsigned fixed-point 32.32 format where the radix point is immediately to the right of integer seconds. Writes set the time; reads provide the time.

5.2.12 rplTimeCompare MS, LS (Admin Base +0x0040, 44)

This 8B location is a diagnostic instrument that measures the difference between the time written and the value of the time server's clock at this instant this register is written. There is no side-effect to the use of this instrument. The 8B signed 32.32 delta value is read from this location at any time after the measurement was made.

5.2.13 rplTimeRefPerPPS (Admin Base +0x0048)

This 4B RO location is a diagnostic instrument, literally a frequency counter, which measures the number of uncompensated local XO cycles that occur between successive (external) PPS events. It is updated after each PPS event. If the PPS signal is considered "exact"; then this count may be used to observe the second-to-second behavior of the local 200 MHz XO. Given a +/- 50 PPM XO, this count may be 200e6 +/- 10e3

5.2.14 Dataplane Memory Region Provisioning

The value conveyed in numDPMemRegions in conjunction with the subsequent data structure in dpMemRegion[] advertises the provisioned capabilities of this bitstream's dataplane memory regions. Dataplane memory is used for DMA buffering, and these regions may use different memories (BRAM, DRAM, etc.). Different data plane blocks are configured to use (and/or share) different memory regions.

5.2.15 numDPMemRegions (Admin Base +0x007C)

The bit fields in this register control the functions described in the table below:

Bits	Name	Access	Description
31:4		Reserved	0
3:0	numDPMemRegions	RO	The number of dataplane memory regions

5.2.16 dpMemRegion[x] Structure (Admin Base +0x0080, 84, ...)

Each 32b DW starting at AdminBase+0x80 describes a dataplane memory region. Only numDPMemRegions are populated. For example, if numDPMemRegions==2, then only the region structure members at 0x80 (for member 0) and 0x84 (for member 1) will be valid.

Bits	Name	Description
31:28	PCIe BAR	The PCIe BAR this region belongs to (usually 1)
27:14	Offset, 4KB pages	Offset into this is PCIe BAR, in 4KB pages
13:0	Size, 4KB pages	Size of this memory region, in 4KB pages

The PCIe BAR says in which BAR (never 0) this memory region is decoded. The Offset, specifies the offset from the base of the BAR, in 4KB pages. The Size, specifies the size of the memory region, in 4KB pages.

5.3 BAR0 – Worker Control Operation Space

Every worker has a fixed set of **Control Operations** (see [AMR]). The worker control aspect is a common model that allows all workers to be managed without having to customize the management software for each individual worker. (Do not confuse with **Configuration Properties**, described next, which allow components to be specialized at runtime.)

This section describes the worker control region registers, the 64KB space allocated to each worker, where the base address is $0x0W_A_0000$. Where the hex nibble $W_A = (W_N + 1)$ is one of the 15 worker ordinal numbers, $W_N = \{0, 1, \dots, 14\}$. For example, the first worker, worker W_0 , has its control region based at BAR0 offset $0x01_0000$. The second worker, worker W_1 , is at offset $0x02_0000$, and so on. The individual locations perform the indicated control operation on the worker when read.

Worker Control Offset	Access	Description
+0x00	Read-Only	Control-Op: Initialize
+0x04	Read-Only	Control-Op: Start
+0x08	Read-Only	Control-Op: Stop
+0x0C	Read-Only	Control-Op: Release
+0x10	Read-Only	Control-Op: Test
+0x14	Read-Only	Control-Op: BeforeQuery
+0x18	Read-Only	Control-Op: AfterConfig
+0x1C	Read-Only	Control-Op: Rsvd7
+0x20	Read-Only	Worker Status Register
+0x24	Read/Write	Worker Control Register
+0x28	Read-Only	lastConfigAddress
+0x2C	Write-Only	Worker Sticky Clear
+0x30-3C	Reserved	-

There are four possible responses (values returned) to Control-Op Reads, they are

Value	Description
0xC0DE_4201	OK
0xC0DE_4202	ERROR
0xC0DE_4203	TIMEOUT
0xC0DE_4204	WORKER IS RESET

The OK and ERROR values come from the worker itself, indicating whether the operation requested was performed successfully or not. The TIMEOUT value is returned when the worker did not return a read response in time to prevent a system bus timeout. In this case the control SW can poll the status register (for as long as it wants) to see if the worker eventually returns an OK or ERROR. The RESET value

indicates that the worker was in a reset state when the control operation was requested (when this register was read), which is usually a software error, since software controls deassertion of reset, per worker.

5.3.1 Worker Status Register (Worker Control Offset +0xFFE0)

Each worker has its own worker status register located 32B before the end of each worker's control space. The worker status register is located at an offset of 0xFFE0. If *N* is a hex nibble specifying the **worker ordinal plus one**, then that worker's status register is located at BAR0 address of 0x0*N*_FFE0.

The worker status register contains both status bits with diagnostic information of the last access and sticky bits, which capture events and require a specific action to clear. The worker status register's collection of sticky bits are initially set to zero at power up reset and set and held if a specific exception event occurs. The sticky bit remains set until the register is cleared.

- Clearing the sfCap bit: Write a '1' to bit position 9 (0x0000_0200) at the worker sticky clear address.
- Clearing the other sticky bits: Write a '1' to bit location 8 (0x0000_0100) at worker sticky clear address.

The "sFlag Captured" bit signals that the worker asserted sFlag for one or more clock cycles on its WCI. The nine other sticky bits serve to capture the cause(s) of error, should the worker not respond normally to a sequence of one or more requests. The sticky bits capture what kind of exception (timeout, fail, err) as well as what was the pending request when the exception occurred (cfgWt, cfgRd, ctlOp). Since these nine bits are sticky, they accumulate all exceptions that may have occurred since the status was reset.

Bit	Name	Access	Description
31-28		Reserved	0
27	lastOpWrite	RO	1=last op was write
26-24	lastControlOp	RO	Last control Op
23-20	lastConfigBE	RO	Last BE used
19		RO	lastOpWrite is Valid
18		RO	lastControlOp is Valid
17		RO	lastConfigBE is Valid
16		RO	lastConfigAddr is Valid
15-10		Reserved	0
9	sfCap	RO (Sticky)	sFlag Captured
8	reqTO.cfgWt	RO (Sticky)	Time-Out of a Config-Write
7	reqTO.cfgRd	RO (Sticky)	Time-Out of a Config-Read
6	reqTO.ctlOp	RO (Sticky)	Time-Out of a Control-OP
5	reqFAIL.cfgWt	RO (Sticky)	Resp-Fail of a Config-Write
4	reqFAIL.cfgRd	RO (Sticky)	Resp-Fail of a Config-Read
3	reqFAIL.ctlOp	RO (Sticky)	Resp-Fail of a Control-OP
2	reqERR.cfgWt	RO (Sticky)	Resp-Err of a Config-Write
1	reqERR.cfgRd	RO (Sticky)	Resp-Err of a Config-Read
0	reqERR.ctlOp	RO (Sticky)	Resp-Err of a Control-OP

5.3.2 Worker Control Register (Worker Control Offset +0xFFE4)

Each worker has its own worker control register located 28B before the end of each worker's control space. The worker status register is located at an offset of 0xFFE4. If *N* is a hex nibble specifying the **worker ordinal plus one**, then that worker's status register is located at BAR0 address of 0x0N_FFE4.

The wrkReset_n bit [bit 31] drives the MReset_n signal to each WCI. When clear, the worker is reset. At power on reset, this bit is clear for all workers. This bit must be set to disable reset before interacting with the worker.

The wrkTimeout bit field [4:0] sets the number of 8 ns cycles to wait after a command before a timeout condition is declared. This field is the log2 of that threshold. A setting of 4 (the power on default) results in $2^4 = 16$ cycle timeout (128 ns). The maximum setting of 31 corresponds to about 17 seconds. ($2^{31} \times 8$ ns/cy). The default timeout setting of 16 cycles (128 ns) may be inadequate for workers requiring more time than that to service their requests. Workers will return the "0xC0DE_4203" TIMEOUT indication if the access timer expires before the worker completes the control operation, or acknowledges the configuration property access. In such cases, it is recommended that the default timeout for that worker be increased, only as much as needed. Higher timeout values add to the upper bound on the maximum latency that can be expected for a response from the system.

The power-on reset value of this register is 0x0000_0004. This means the worker is reset. To take a worker out of reset, retaining the default 16-cycle timeout, write 0x8000_0004 to this register.

Bit	Name	Access	Description
31	wrkReset_n	RW	Worker MReset_n
30-5		RW	Spare
4-0	wrkTimeOut	RW	Log ₂ control cycles for Timeout

5.4 BAR0 – Worker Configuration Properties Space

Worker **Configuration Properties** are named storage locations of a worker that may be read or written. Their values indicate or control aspects of the worker's operation. Reading and writing configuration property values may or may not have side effects on the operation of the worker. Configuration properties with side effects can be used for custom worker control. Each worker may have its own, possibly unique, set of configuration properties. They may include hardware resource such registers, memory, and state.

This section describes the worker configuration properties, the contiguous 1 MB space allocated to each worker, where the base address is 0xW_A0_0000. Where the hex nibble W_A=(W_N + 1) is one of the 15 worker ordinal numbers, W_N = {0,1,...14}.

For example, the first worker, worker W0, has its configuration properties based at BAR0 offset 0x10_0000. The second worker, worker W1, at offset 0x20_0000, and so on.

Each worker has a contiguous 1MB address space, which maps directly to 2²⁰ Bytes of configuration properties.

The following table shows the BAR0 worker assignments in the OC1001 application:

BAR0 Offset	Region	OC1001 Assignment
+0x0_0000	Admin	Admin
+0x1_0000	Worker 0	GCD Worker 0
+0x2_0000	Worker 1	GCD Worker 1
+0x3_0000	Worker 2	FC Worker 2
+0x4_0000	Worker 3	Bias Worker 3
+0x5_0000	Worker 4	FP Worker 4
+0x6_0000	Worker 5	Unused
+0x7_0000	Worker 6	Unused
+0x8_0000	Worker 7	Unused
+0x9_0000	Worker 8	Unused
+0xA_0000	Worker 9	Unused
+0xB_0000	Worker 10	Unused
+0xC_0000	Worker 11	Unused
+0xD_0000	Worker 12	Unused
+0xE_0000	Worker 13	Data Plane 0 (FC)
+0xF_0000	Worker 14	Data Plane 1 (FP)

6 Data Plane Unit (DPU)

6.1 BAR1 – 64KB Memory Space for Data Movement

BAR1 is a (currently) 64KB memory space used by the data plane. Each data plane adapter occupies 32KB, in its own “memory region”. This 32KB space may be partially- or fully populated- with BRAM buffer memory. The 64KB space is suitable to hold the two 32KB memory regions for 2 DPUs.

BAR1 Offset	Region
+0x0000	Data Plane 0 Buffer Memory (16KB)
+0x4000	Data Plane 0 Buffer Memory Expansion (16KB)
+0x8000	Data Plane 1 Buffer Memory (16KB)
+0xC000	Data Plane 1 Buffer Memory Expansion (16KB)

We show below an example of a 16KB data plane memory-mapped region used to hold messages and metadata. Each DPU can be programmed for the following organization for a 7-buffered storage in its region. There are seven 2KB message buffers and seven 16B metadata buffers.

BRAM Offset	Region
+0x0000	Message Buffer 0 (2KB)
+0x0800	Message Buffer 1 (2KB)
+0x1000	Message Buffer 2 (2KB)
+0x1800	Message Buffer 3 (2KB)
+0x2000	Message Buffer 4 (2KB)
+0x2800	Message Buffer 5 (2KB)
+0x3000	Message Buffer 6 (2KB)
+0x3800	Metadata 0 (16B)
+0x3810	Metadata 1 (16B)
+0x3820	Metadata 2 (16B)
+0x3830	Metadata 3 (16B)
+0x3840	Metadata 4 (16B)
+0x3850	Metadata 5 (16B)
+0x3860	Metadata 6 (16B)
+0x3870~3FFF	Unused (16KB – 14KB – 112B)

The 16B data structure for the metadata is:

```
typedef struct {
    Bit#(32) length; // Message Length in Bytes
    Bit#(32) opcode; // Opcode in bits [7:0]
    Bit#(32) nowMS;
    Bit#(32) nowLS;
} MesgMeta deriving (Bits, Eq);
```

The “length” and “opcode” are associated with the accompanying message in the data buffer. The “now” fields represent a timestamp when the messages was provided to the DPU by the upstream application workers.

6.2 Configuration Properties

The DPU configuration properties allow specialization of the DPU behavior at runtime. Each DPU has its own configuration register set.

The color-coding of the rows in the table below indicates the initialization properties, from the message-advancing events, from the buffer status, from the side-effect-free status and debugging probes, from the registers directly used in active message and “doorbell” movement.

Property Offset	Property Name	Mode	Default	When Used	Description
+0x0000	lclNumBufs	RW	1	initialize	Number of Local Buffers (This Node)
+0x0004	fabNumBufs	RW	1	Initialize	Number of Fabric Buffers (Far Node)
+0x0008	lclMesgBase	RW	0x0000	initialize	Local Message Buffer Base Address
+0x000C	lclMetaBase	RW	0x3800	initialize	Local Metadata Buffer Base Address
+0x0010	lclMesgBufSize	RW	0x0800	initialize	Local Message Buffer Size
+0x0014	lclMetaBufSize	RW	0x0010	initialize	Local Metadata Buffer Size
+0x0018	fabDoneAvail	WO	-	MesgAdv	Fabric Done Available Scalar (See Table Following Describing Usage)
+0x001C	regionNum	RO		initialize	To which memory region this DP belongs
+0x0020	bufReady	RO	-	Poll	Lcl Buffers Committed/Freed (Number of buffers available for remote)
+0x0024	rsvd	RO	-	Debug	32'hF00D_FACE
+0x0028	bmlAccu	RO	-	Debug	{ Lcl Buffers Available/Ready[31:16], RemoteBuffersAvailable[15:0]}
+0x002C	bufIndex	RO	-	Debug	{remIndex [31:16] , lclIndex [15:0]}
+0x0030	lclCounts	RO	-	Debug	{lclStart [31:16] , lclDone [15:0]}
+0x0034	remCounts	RO	-	Debug	{remStart [31:16] , remDone [15:0]}
+0x0038	thisMesg	RO	-	Debug	{opcode [31:24] , length [23:0]}

+0x003C	lastMesg	RO	-	Debug	{opcode [31:24] , length [23:0]}
+0x0040	v[1]	RO	-	Debug	req/wrt Count
+0x0044	v[0]	RO	-	Debug	wrtData
+0x0048	rsvd	RO	-	Debug	32'hDADE_BABE
+0x004C	bufferExtent	RO	-	Debug	Size of Local Memory Buffer
+0x0050	fabMesgBase	RW	FFFF_0000	DMA	Fabric DMA Message Base Address
+0x0054	fabMetaBase	RW	FFFF_3800	DMA	Fabric DMA Metadata Base Address
+0x0058	fabMesgSize	RW	0000_0800	DMA	Fabric DMA Message Buffer Size
+0x005C	fabMetaSize	RW	0000_0010	DMA	Fabric DMA Metadata Buffer Size
+0x0060	fabFlowBase	RW	FFFF_0018	DMA	Fabric DMA Flow Control Base Address
+0x0064	fabFlowSize	RW	0000_0004	DMA	Fabric DMA Flow Control Base Size
+0x0068	dpControl	RW	0	DMA	Data Plane Control (See Text)
+0x006C	flowDiagCount	RO	0	Debug	Number of Flow Control Events Sent

Notes: All “sizes” and “addresses” in bytes unless otherwise stated.

6.2.1 fabDoneAvail (DPCP +0x0018)

This write-only DWORD location treats each write as signaling either a “done” or “avail” event. Depending upon the data plane dpDirection and dpRole set in the dpControl register (DPCP+0x68), the event signaled here will cause a specific internal action within the DPCP, as described in the table below:

Local Role	Fabric Producer (FP)	Fabric Consumer (FC)
Passive	(source) done – near buffer empty	(sink) done – near buffer full
Active Message	(pusher) avail – far buffer empty	(puller) avail – far buffer full
Active Flowcontrol	(pullee) done – near buffer empty	(pushee) done – near buffer full

This single-location for all inbound data plane event signaling can reduce the complexity of control software.

Internally to the DPCP, “done” events generate a “remoteDone” signal that is used to

- Increment LBAR
- Decrement LBCF
- Increment remBuf
- Increment fabBuf

Internally to the DPCP, “avail” events generate a “fabBufAvail” signal that is used to

- Increment FBA

6.2.2 regionNum (DPCP +0x001C)

This read-only location identifies to which memory region (as enumerated in the CP's admin register space) this DPU belongs



Bits	Name	Description
31:4	rsvd	'0
3:0	regionNum	DPU memory region

6.2.3 dpControl (DPCP +0x0068)

This register controls the behavior of the Data Plane. For each DPU in the infrastructure that is to be actively used, a choice of data flow *direction* and *role* must be made.

The dpDirection field requires a specific selection of Producer vs. Consumer, if the module is to act at all. The dpRole field refines that behavior into a Passive, Active Message, or Active Doorbell role.

Bits	Name	Access	Description
31:3		Reserved	0
7	dpMoveData	RW	0=Inhibit Active Message Movement 1=Enable Active Message Movement
6	dpMoveMeta	RW	0=Inhibit Active Metadata Movement 1=Enable Active Metadata Movement
5	dpSendTail	RW	0=Inhibit Transmission of Active Message Tail Event 1=Enable Transmission of Active Message Tail Event
4	dpSendInterrupt	RW	0=Inhibit Interrupt at end of Message Movement 1=Enable Interrupt at end of Message Movement
3:2	dpDirection	RW	00=Disabled DP Module 01= <u>Fabric Producer (FP)</u> 10= <u>Fabric Consumer (FC)</u> 11=Reserved
1:0	dpRole	RW	00= <u>Passive</u> 01= <u>Active Message</u> 10= <u>Active Flowcontrol</u> 11=Reserved

Local Role	 Fabric-Producer (FP)	 Fabric-Consumer (FC)
<u>Passive</u>	1. Partner polls status and reads message data by any means	2. Partner polls status and writes message data by any means
<u>Active Message Movement</u>	3. We receive fabric Flowcontrol for far-side “partner-buffer-free”; we produce message data by <u>Active-Push</u> DMA	4. We receive fabric Flowcontrol for far-side “partner-buffer-full”; we consume message data by <u>Active-Pull</u> DMA
<u>Active Flowcontrol Transmission</u>	5. We Transmit DMA “local-full” Flowcontrol to our partner; Partner reads message data by any means	6. We Transmit DMA “local-free” Flowcontrol to our partner; Partner sends message data by any means

These settings are applied to each DPU upon a control operation START event while in the INITIALIZED state.

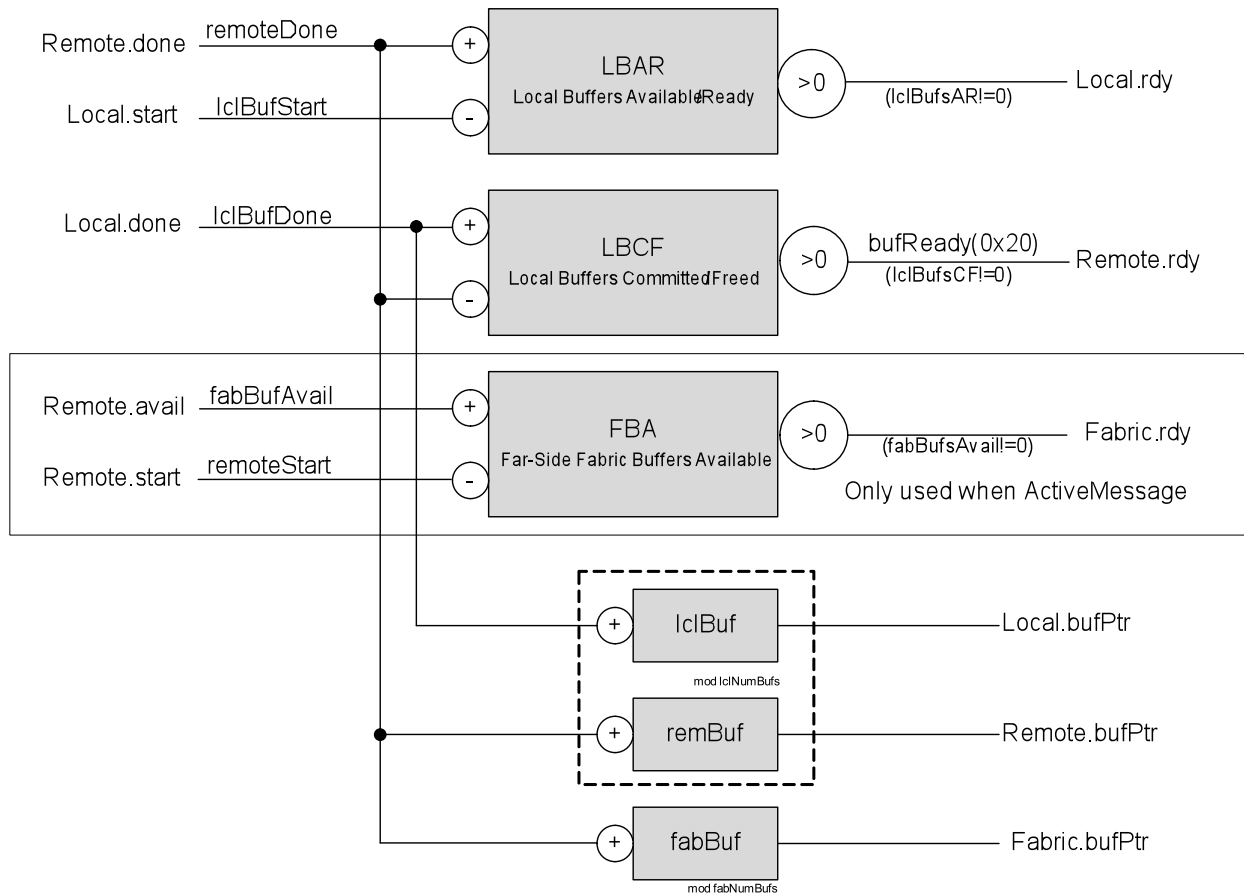
Unused DPUs will remain disabled due both to the default “disabled” in the dpDirection field as well as control system not initializing and then starting that component.

6.2.4 Multi-buffer Initial Conditions

The initial conditions of the multi-buffer accumulators have been chosen so that no precharge or priming is required to start data movement. Thus when started, a DPU assumes that local and far buffers are all initially empty and available for use.

6.2.5 Multi-Buffer Logical Block Diagram

The diagram below shows a simplified version of the logic connecting the various produce/consume events to the accumulators that track buffer availability and position.



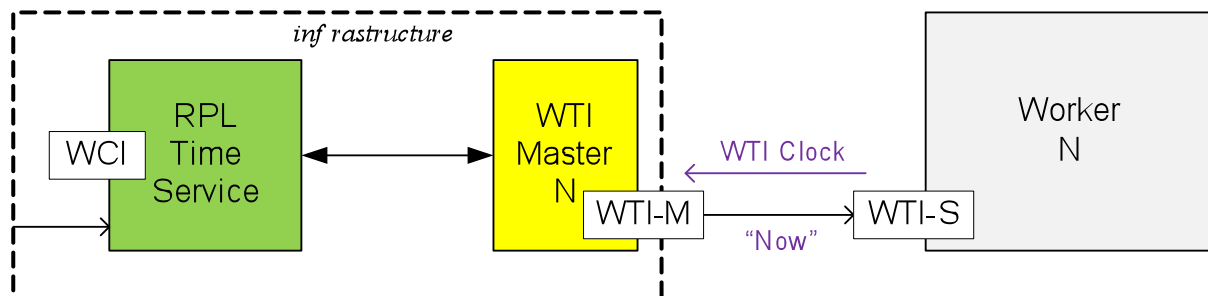
7 HDL Time Service (HTS)

HTS maintains a local time clock on an HDL device. It keeps time through a phase-accumulator whose increment is closely related to the frequency of a local crystal oscillator (XO). This XO is typically a timebase from a local, stable “brick”; distinct from other clocks that may have been recovered from a transport (such as a PCIe or Ethernet PHY), whose absolute stability is often less constrained.

The time on the HTS may be set (or observed) by writing (or reading) a configuration property. Select implementations of the HTS employ logic to assist in the IEEE1588 Precision Time Protocol (PTP).

The accuracy of the stable, uncompensated XO (typically +/- 50 PPM)¹ can be dramatically improved by connecting a GPS device with a 1 PPS output. Select implementations of HTS logic may automatically detect the presence or absence of the 1 PPS signal and, when available, digitally compensate for the local timebase frequency/phase error. The 1 PPS signal allows the stable, but not necessarily precise, local XO to deliver the long-term precision a GPS reference can provide.

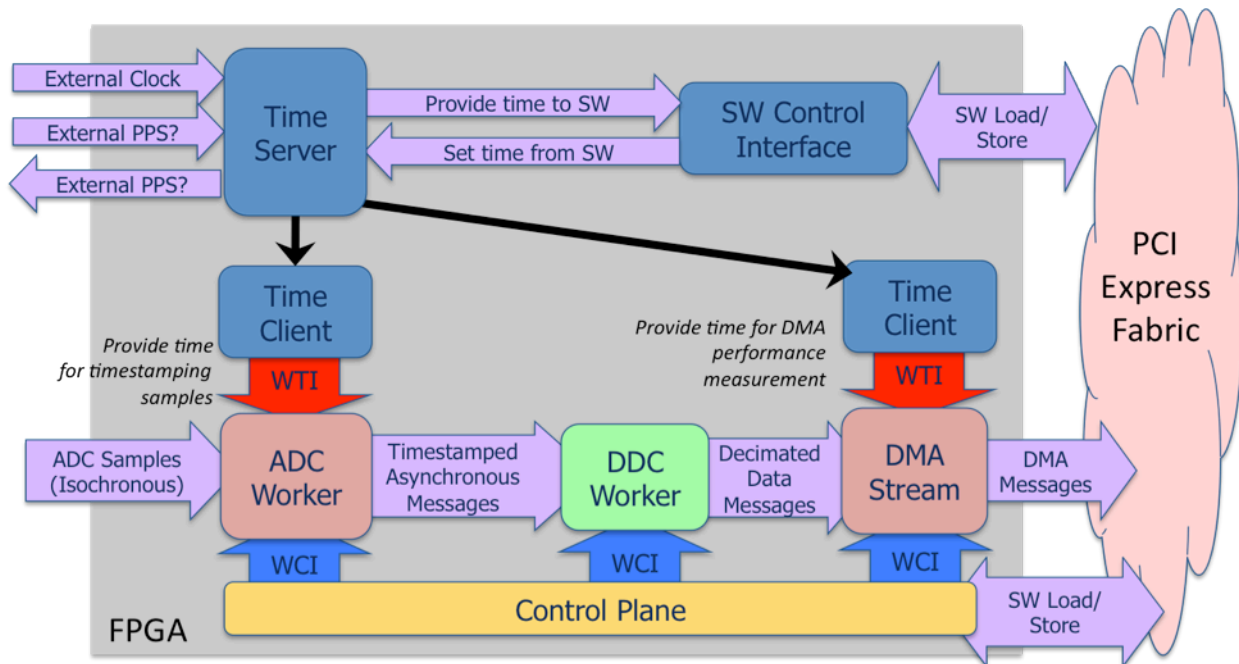
7.1 How the HTS is used with Workers that desire Time



Any worker that desires time can use the Worker Time Interface (WTI) defined in the WIP specification. The worker implementer chooses in which clock domain they want to observe time, and how many bits to the left and right of the “1 second” radix point they wish to observe. The infrastructure then provides each worker with pure-binary, continuous, monotonic, GPS time, in the clock domain and format specified.

The HTS clock supports multiple workers each asking for time in a different clock domain by instantiating as many WTI Masters as are required. Baseline implementations of WTI masters are area-efficient synchronizers that bring the HTS time into the WTI clock domain. Higher-accuracy implementations shadow the HTS time with a phase-accumulator in the WTI clock domain.

¹ Local XO frequency tolerance is board dependent. Contemporary platforms (e.g. Xilinx ML605) use a 200 MHz SAW XO with +/- 50PPM (Epson EG-2121CA)



7.2 HTS Time Server Configuration Properties

This section describes the properties of the HDL Time Service (HTS) from the admin space in BAR0 that is part of the Control Plane described earlier.

Admin Offset	Name	Access	Description
+0x0030	rpITimeStatus	RO	Time Status Register
+0x0034	rpITimeControl	RW	Time Control Register
+0x0038	rpITimeMS	[2] RW (stage)	RPL Time Integer Seconds
+0x003C	rpITimeLS	[2] RW (commit)	RPL Time Fractional Seconds
+0x0040	rpITimeCompareMS	[2] RW (stage)	Time Compare Argument MS
+0x0044	rpITimeCompareLS	[2] RW (commit)	Time Compare Argument LS
+0x0048	rpITimeRefPerPPS	RO	Reference Clocks per PPS

7.3 Using the HTS Time Server

Software can set the time by writing the rpITime register. It can subsequently “dial out” the time delay in setting the time by using the rpITimeCompare register, e.g., by determining a typical delay and then pre-adding that offset to the time value it is about to set. Software can also directly access precise time associated with the attached PPS signal.