

USB 3.0, PCI Express & Ethernet under a Transparent Hat

FPGA Manager / Stream Buffer Controller

ALL PROGRAMMABLE PLC2 Days

12 June 2013

Stuttgart

Martin Heimlicher

Enclustra GmbH



 **ALL PROGRAMMABLE**
PLC2 Days 2013

 **XILINX**
ALL PROGRAMMABLE™

- Enclustra Company Profile
 - FPGA Design Center
 - FPGA Solution Center
- FPGA-to-Host Communications
 - Multiplicity
 - Data Flow Semantics
 - Streaming & Memory-Mapped
 - Performance
 - Error Handling
 - PCIe specific
 - USB specific
 - Ethernet specific
- The Transparent Hat
- FPGA Manager
 - Overview
 - The Paradigms & Design Goals
 - UNISCP Packet Format
 - Host-side API
 - FPGA Block Diagrams
 - Xilinx Tool Integration
 - Licensing & Availability
 - C# Example
- Stream Buffer Controller
- Use Cases
- Further Information
- Questions

- **Enclustra Company Profile**
 - **FPGA Design Center**
 - **FPGA Solution Center**
- **FPGA-to-Host Communications**
 - Multiplicity
 - Data Flow Semantics
 - Streaming & Memory-Mapped
 - Performance
 - Error Handling
 - PCIe specific
 - USB specific
 - Ethernet specific
- **The Transparent Hat**
- **FPGA Manager**
 - Overview
 - The Paradigms & Design Goals
 - UNISCP Packet Format
 - Host-side API
 - FPGA Block Diagrams
 - Xilinx Tool Integration
 - Licensing & Availability
 - C# Example
- **Stream Buffer Controller**
- **Use Cases**
- **Further Information**
- **Questions**

- Quick Facts

- Founded in 2004
- Located in the Technopark of Zurich, Switzerland
- 11 FPGA Engineers, 1 technician, 1 office staff
- Vendor independent

- Business Units

- FPGA Design Center (services)
- FPGA Solution Center (products)

- FPGA Partnerships

- Xilinx Alliance Program – Certified member
- Altera Design Services Network – Certified member
- Lattice Semiconductor Leader – Member



Enclustra Company Profile

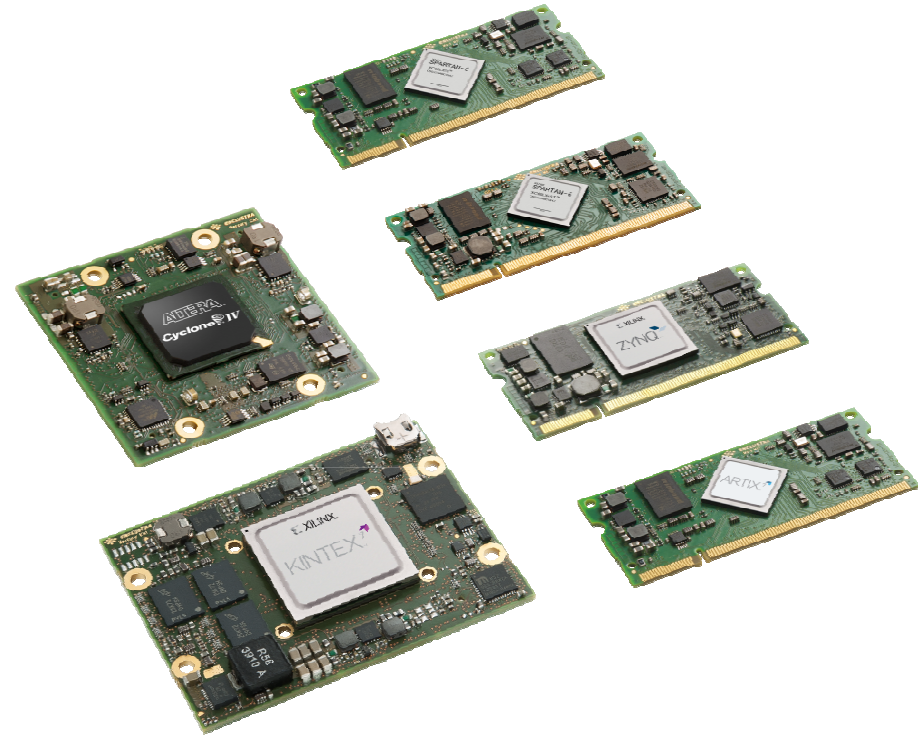
FPGA Design Center

- FPGA System Design
 - Hardware (High-Speed, Analog, RF)
 - HDL Firmware (VHDL, Verilog)
 - Embedded Software (for FPGA processors)
- Focus Areas
 - Embedded Computing
 - Communications
 - Software Defined Radio
 - Camera Systems
 - Motion & Drive Control
- Experience
 - >50 Person-years of FPGA hardware, firmware & software design
 - >50 Customers, >200 projects

Enclustra Company Profile

FPGA Solution Center – Modules

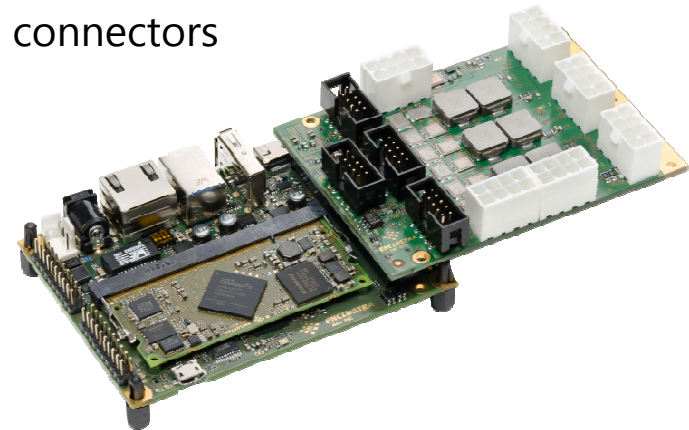
- Mars Family (MX1, MX2, AX3, ZX3)
 - SO-DIMM 67.6 x 30 mm
 - 96-108 User I/Os, 0-2 MGTs
 - 1-2 Ethernet Ports, 0-1 USB Port
 - 3.3V Single Supply Voltage
- Mercury Family (KX1, CA1)
 - 56 x 54 to 72 x 54 mm
 - 146-168 User I/Os, 0-4 MGTs
 - 1-2 Ethernet Ports, 1 USB Port
 - 5-15V Single Supply Voltage
- Designed & priced for integration into your product
 - Planned availability: 10 years
 - >250 Customers in >40 Countries
- Module Selection Guide & Roadmap available online



Enclustra Company Profile

FPGA Solution Center – Hw Solutions

- Base Boards & FMC Cards
 - Mercury Starter: USB 3.0, pin headers, Displayport & LVDS HDMI connectors
 - Mars Starter: USB 2.0, pin headers, LVDS HDMI connectors
 - Mars PM3: USB 3.0, FMC LPC
 - FMC DR2: Drive control card
 - FMC AN1: FMC breakout board
- Hardware & Quick Start Kits
- Design your Custom „Dream“ Module with us – and save!
 - Pay only around half the development cost
 - Get stable hardware
 - Get fully-tested modules
 - Get the production data
 - Produce yourself or buy from us with special pricing



- FPGA Manager IP Solution
 - PCIe 1X/2X/4X for Xilinx 7 Series FPGAs
 - Ethernet/UDP for Fast & Gigabit Ethernet
 - USB 2.0/3.0 Solution (FTDI FT2232H, Cypress EZ-USB FX3)
- IP Cores
 - Universal Drive Controller (DC/BLDC/SM)
 - 2D-Accelerated Display Controller (LVCMOS/LVDS/DVI/HDMI)
 - Camera Input Controllers (Camera Link/MIPI/LVDS)
 - UDP Streaming Ethernet Controller (10/100/1G/10G)
 - Resource-saving Memory Controllers (DDRx/QDRx/Flash)
 - Available with AMBA AXI4 and Avalon-compliant interfaces
- Easy SignOnce IP Licensing: Evaluation, Academic, Project, Site, Source
- Maintenance & Support available
- Design your Custom „Dream“ IP with us – and save!



Contact us with any FPGA challenge!



Klaus Schwan
Sales (Munich office)



Patrick Müller
Sales (Headquarter)



Christoph Glattfelder
Technical Support



Martin Heimlicher
CEO

- Enclustra Company Profile
 - FPGA Design Center
 - FPGA Solution Center
- **FPGA-to-Host Communications**
 - **Multiplicity**
 - **Data Flow Semantics**
 - **Streaming & Memory-Mapped**
 - **Performance**
 - **Error Handling**
 - **PCIe specific**
 - **USB specific**
 - **Ethernet specific**
- The Transparent Hat
- FPGA Manager
 - Overview
 - The Paradigms & Design Goals
 - UNISCP Packet Format
 - Host-side API
 - FPGA Block Diagrams
 - Xilinx Tool Integration
 - Licensing & Availability
 - C# Example
- Stream Buffer Controller
- Use Cases
- Further Information
- Questions

FPGA-to-Host Communications Multiplicity

- Multiple operating systems
- Multiple FPGA vendors
- Multiple programming languages (C++, .NET, MATLAB, ...)
- Multiple communication interfaces (PCIe, USB, Ethernet, ...)
- Multiple interface speeds & generations
 - PCIe: Gen1, Gen2, Gen3 (soon: Gen4)
 - USB: 2.0, 3.0 (soon: 4.0)
 - Ethernet: 100 Mbps, 1 Gbps, 10 Gbps
- Multiple customer hardware & software products of different generations interacting together in the lab and in the field

FPGA-to-Host Communications

Data Flow Semantics

- Multiplexing
 - A single communication interface for a multitude of dataflows
 - Multiple simultaneous connections to the same device (e.g. monitoring daemon, database logger, user GUI)
- Data flow semantics
 - Memory-mapped (register bank/control) & streaming (data)
 - Data push & data pull
 - Foreground & background transfers
 - Blocking & Non-blocking transfers

FPGA-to-Host Communications Data Streaming

- Stream Framing
 - Frame streams & byte streams
 - Meta-data association (e.g. timestamp, video resolution, video source)
- Frame fragmentation & re-assembly
 - Frame-to-packet
 - Frame-to-buffer
- Frame size
 - Constant (e.g. video frames)
 - Variable (e.g. network packets, object recognition results)
- Challenges
 - Frame size larger than buffer size (cut-through forwarding)
 - Frame poisoning (marking a frame as "bad" after transmission started)

FPGA-to-Host Communications

Memory-mapped Access

- Memory-mapped access types
 - Single word
 - Burst
 - Same-address bursts
 - Read-modify-write (set bit, clear bit, toggle bit)

- Challenges
 - Read from non-prefetchable targets (e.g. FIFOs)
 - Execution ordering under packet loss scenarios
 - Sub-word accesses (e.g. 1 byte read/write on a 32-bit interface)
 - Target-address boundary-crossing burst accesses

FPGA-to-Host Communications Performance

- Bandwidth (MBytes/sec) vs. frame size
 - Packets as long as possible
 - The least possible number of data copies in the host
- Latency (micro-seconds) vs. frame size
 - Packets as short as possible
 - Liveness & worst-case latency
- Throughput (Mops/sec) vs. frame size
 - Pipelining of operations
 - Multiple operations per packet
- CPU utilization
 - Event-driven instead of polling
- Scheduling
 - Minimize the impact of one stream's transfers on other streams

FPGA-to-Host Communications Error Handling

- Non-fatal error handling
 - Buffer overflows
 - Checksum error
 - Packet loss
 - Timeouts & recovery
- Fatal error handling
 - Illegal packet format
 - Callback returns exception
 - Stack overflow
- Out-of-memory / out-of-resources handling
 - Fatal or non-fatal?
- Prevent application restart & device power-cycle „work-around“ due to incomplete error handling

FPGA-to-Host Communications

Device Discovery

- Enumeration
 - Find „own“ devices on all interfaces (PCIe, USB, Ethernet, ...)
 - Retrieve device information (name, serial number, configuration, ...)
- Device Identification
 - FPGA/Flash/EEPROM unique IDs
 - Product & serial numbers
- Versioning
 - Protocol backward & forward compatibility
 - Firmware-to-software version backward & forward compatibility

FPGA-to-Host Communications Bitstream Configuration

- FPGA configuration
 - FPGA firmware download
 - Fail-safe FPGA firmware update
- Design Security
 - FPGA bitstream copy protection
 - Pay-option storage and retrieval
 - Encrypted data flows

FPGA-to-Host Communications

FPGA Architecture

- FPGA interfaces
 - AXI & Avalon & more
 - Master & Slave
 - Memory mapped & streaming
 - Multiple data widths
 - Multiple stream formats
 - Multiple clock domains
- FPGA data buffering
 - Virtual FIFO functionality (e.g. host or interface is not as fast as the data source)
 - Retransmission capability
- Local processor data pre-processing

FPGA-to-Host Communications

PCIe specific

- System
 - Kernel-to-user access protection (per stream)
 - User software version ⇔ FPGA Manager Library version ⇔ FPGA Manager Driver version ⇔ FPGA Manager IP version ⇔ User HDL version
 - Multiple devices may have different driver versions
 - Multiple applications may use different library versions
- Software
 - Windows Driver: 32-bit & 64-bit required with installation and uninstallation procedure, Microsoft driver signing
 - Linux Driver: Kernel and distribution „independence“?
 - Portability to other Operating Systems?
 - Memory-mapping to user process(es): Kernel bypass?
 - Contiguous-buffer allocation size limitation

FPGA-to-Host Communications

PCIe specific

- Address translation modes
 - Direct DMA
 - Contiguous DMA
 - Scatter/Gather DMA
- Command Queues
 - Circular buffer handling
 - Status updates
- Performance
 - Read-request-size limitation
 - TLP field width limitation
 - Completion reordering
 - Prefetching of commands & page-table entries
 - Kernel bypass

FPGA-to-Host Communications

PCIe specific

- Races between interrupts, target registers, host memory visibility and cache
- Device insertion & removal, hot plug
- Power management
- System reboot handling
- Live firmware download/update without reboot
- Error handling
 - Application crash handling
 - Stream closing/flushing
 - Segmentation fault handling (addresses specified in command queues)
 - Bus-error handling
 - Stream independence: An error in a single stream will allow the other streams to continue without interruption

FPGA-to-Host Communications

USB specific

- General USB topics
 - Prevent short packets
 - (Re-) Initialization difficulty (e.g. FPGA reconfiguration)
- FTDI FT2232H USB 2.0
 - Challenging external timing
 - FTDI library access not trivial
 - External circuitry required to support FPGA configuration, SPI, I2C
- Cypress EZ-USB FX3 USB 3.0
 - Challenging external timing
 - ARM firmware required for endpoint and DMA configuration
 - Additional endpoints for FPGA configuration, SPI, I2C

FPGA-to-Host Communications

Ethernet specific

- Hardware
 - A multitude of external PHY interfaces (MII, RMII, GMII, RGMII, ...)
 - Challenging external timing for RGMII
 - PHY specific MDIO initialization is often required for optimal timing margin
- TCP/IP on the FPGA-side
 - MAC & IP address setup (with local non-volatile storage)
 - Support of the ARP protocol is practical
 - Packet reordering at reception?
 - Data retransmission after packet loss?
 - (Adaptive?) bandwidth throttling for slower connections?
- TCP/IP on the host is usually challenge-free

FPGA-to-Host Communications

Ethernet specific

- MTU & fragmentation
 - Packet fragmentation may occur in intermediate routers
 - Large datagram size is useful for higher performance (at the host), but may require datagram-to-packet fragmentation and re-assembly (on FPGA and host sides)
 - Jumbo packet support necessary for higher performance
- Port multiplexing
 - Support of additional (non-host related) UDP ports may be required on the same physical Ethernet interface
 - Support of general MAC functionality (with DMA) may be required when a local (soft or hard) processor contains a TCP/IP stack

FPGA-to-Host Communications

Ethernet specific

- Reasons for a local processor with TCP/IP stack
 - Support of DHCP, DNS & Bonjour
 - Web interface with IP address setup
- Open/close
 - Reconnection from same or different host while connection is open
 - Connection from multiple hosts to different streams
- Timeouts
 - Connection to host lost: Transmission should stop
 - Last packet lost: Liveness must be assured

The Transparent Hat?

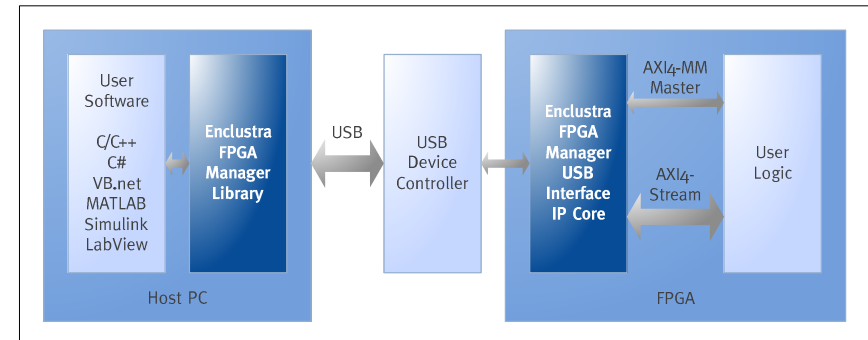
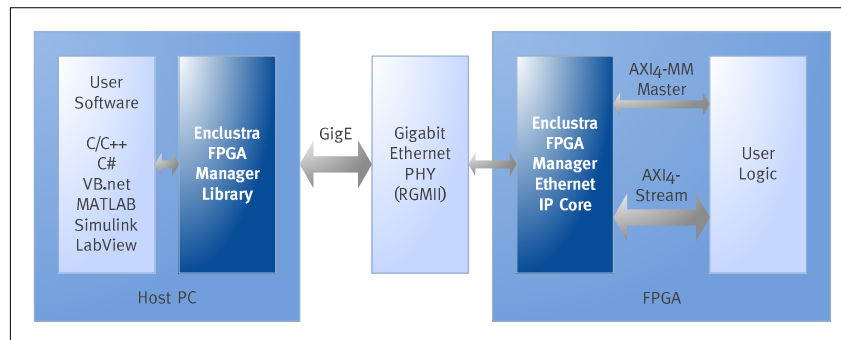
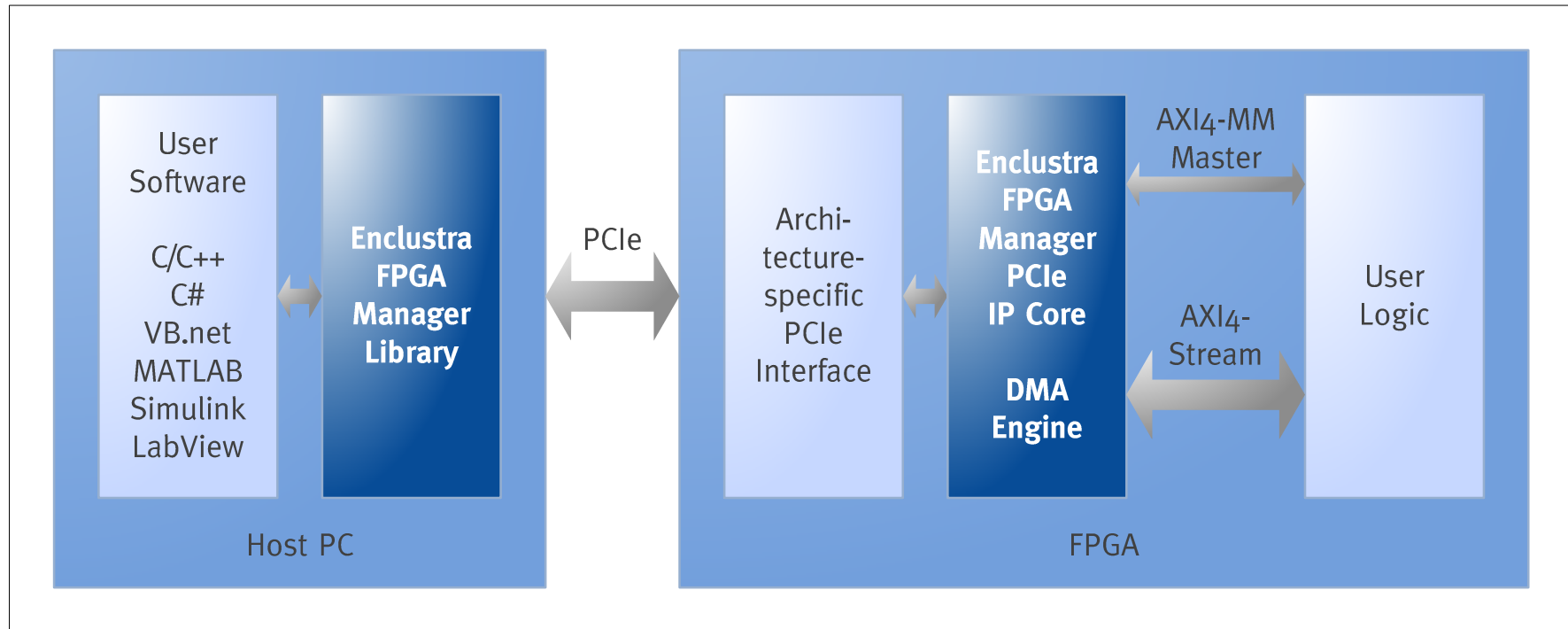
- Define a feature set that is as small as possible, but sufficient for most applications.
- Bring the FPGA as close as possible to the host. The software developer would like to have the FPGA integrated into his processor.
- Abstract the interface by defining a data model that works efficiently on all interfaces.
- Don't over-engineer!
- Test, test, test.



- Now: Let's have a look at FPGA Manager...

- Enclustra Company Profile
 - FPGA Design Center
 - FPGA Solution Center
- FPGA-to-Host Communications
 - Multiplicity
 - Data Flow Semantics
 - Streaming & Memory-Mapped
 - Performance
 - Error Handling
 - PCIe specific
 - USB specific
 - Ethernet specific
- The Transparent Hat
- **FPGA Manager**
 - **Overview**
 - **The Paradigms & Design Goals**
 - **UNISCP Packet Format**
 - **Host-side API**
 - **FPGA Block Diagrams**
 - **Xilinx Tool Integration**
 - **Licensing & Availability**
 - **C# Example**
- Stream Buffer Controller
- Use Cases
- Further Information
- Questions

FPGA Manager Overview



FPGA Manager

The Paradigms

- System
 - Up to 16 independent equivalent streams per direction
 - Unidirectional and/or bidirectional streams
 - Stream to memory-mapped converter (e.g. AXI4-S to AXI4-MM)
 - Round-robin arbitration between streams with optional per-stream bandwidth limitation
 - The user only receives correct data in the correct order.
- Frames
 - Data is organized into a sequence of frames of variable size.
 - A frame number and a frame byte offset is used in the packet header to associate packets to a specific frame and position
 - Frames received on the host may be shorter or longer than the pending receive buffers. Frames may span multiple receive buffers to support large dynamics in frame sizes without wasting large amounts of memory.

FPGA Manager

The Paradigms

- Buffering / Flow control
 - There is no explicit flow control. The user is required to ensure sufficient buffer space.
 - Transmission is started as soon as there is available data.
- Error handling
 - Data loss is tolerated.
 - The user is notified of all error conditions using callbacks and exceptions.
 - Liveness assured with timeouts on all operations.

FPGA Manager

The Paradigms

- FPGA side
 - The IP core contains asynchronous FIFOs to allow the user-side AXI interfaces to run at a user-selectable clock frequency.
 - The user can specify the receive buffer size of each stream at compile time.
- Host side software API
 - Multi-threaded API access (e.g. one user thread per data stream)
 - Any number of receive operations with individual receive buffers can be pending on each stream.
 - Any number of transmit operations with individual transmit buffers can be pending on each stream.
 - The user is notified when a transfer completes using callbacks and exceptions.

FPGA Manager Design Goals

- The solution shall be easy to learn and work out-of-the-box.
- The user shall see the least possible differences when using FPGA Manager on different interfaces, FPGA vendors, programming languages, and operating systems.
- The IP components shall integrate well with FPGA vendor tools (drag & drop).
- The licensing shall be as flexible as possible in order to keep license costs low for applications not requiring the full set of features.
- The highest possible bandwidth shall be achieved.
 - Bandwidth is more important than latency because bandwidth is more often a limiting factor in user applications. Latency is usually limited by OS performance.
 - A maximum of one user-space data copy shall be used for any data transfer.
 - PCIe shall support zero-copy operation when using one of the DMA modes: direct, contiguous buffer or scatter/gather

FPGA Manager Design Goals

- One stream shall not stall the other streams under any condition.
 - The user needs to ensure that enough buffer space is available before starting any data transfer. Data is discarded in order to prevent stalling and to guarantee liveness.
- Memory-mapped accesses shall support retransmission and respect execution order.
 - User application design will simplify substantially if this goal is achieved.
- Long link latencies shall be supported.
 - Worldwide Ethernet connections shall work as expected, although streaming channels must use low enough bandwidth and frame sizes in order to overcome occasional packet losses.
- All data transfers shall be specified at the byte (and not the word) level.
 - It shall be possible to read and write any number of bytes on each transfer. There shall be no alignment restrictions on the source and destination addresses.
 - The host software shall not need to know the stream width on the FPGA.

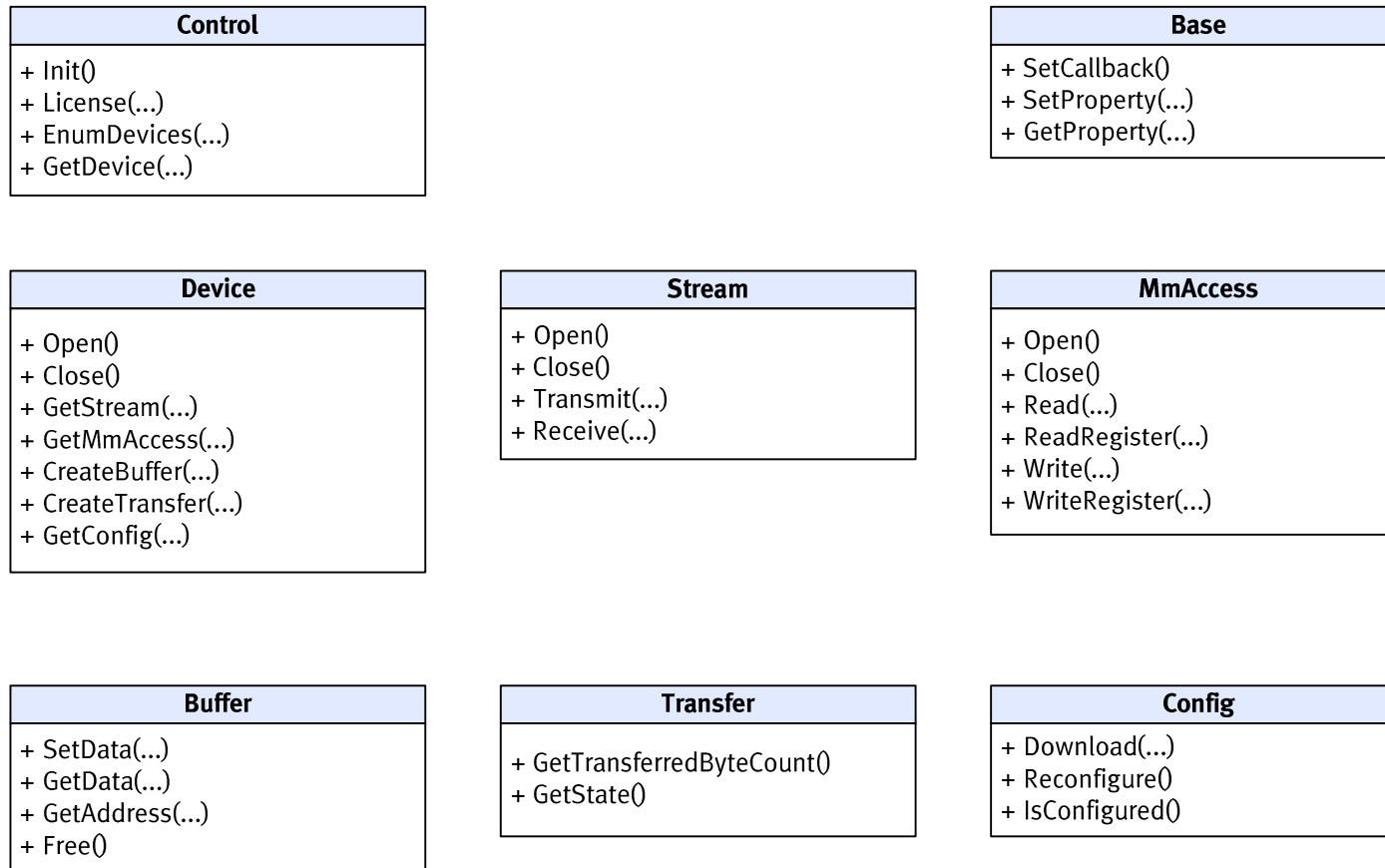
FPGA Manager

UNISCP Packet Format

Data Command																																	
Dword	Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		Byte 3 (transmitted last)								Byte 2								Byte 1								Byte 0 (transmitted first)							
0		CT				AB				FN								CL								IC							
1		FO																															
2..CL-1		PL																															

Field	Field name	Description
CT	Cmd type	3 = Data
AB	Reserved	Always 0
FN	Frame number	Frame number, wraps at 2^{12}
FO	Frame offset	Byte offset of payload in current frame, wraps at 2^{32}
CL	Command length	Command length in Dwords, including header
IC	Invalid count	Number of invalid trailing bytes
PL	Payload	Dword aligned payload. The two LSBs of the frame offset (FO) specify a number of leading invalid payload bytes. The invalid count (IC) specifies a number of trailing invalid payload bytes.

FPGA Manager API Class Diagram

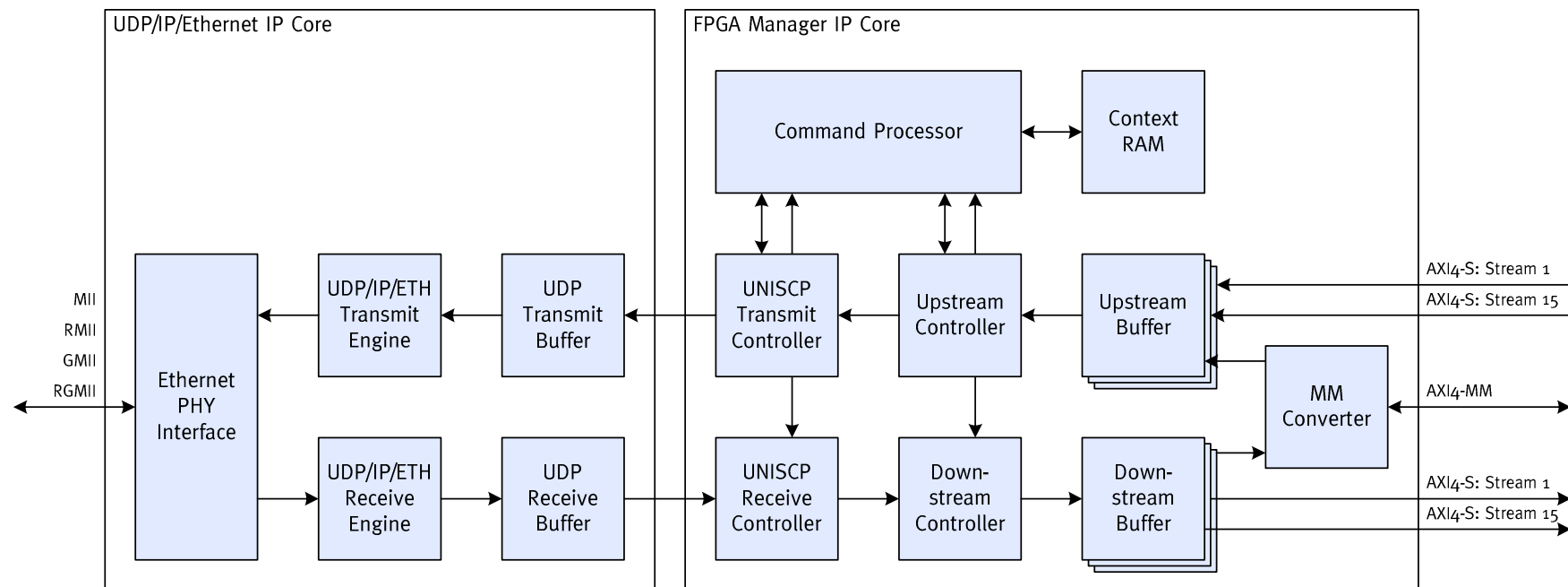


FPGA Manager

The API Classes

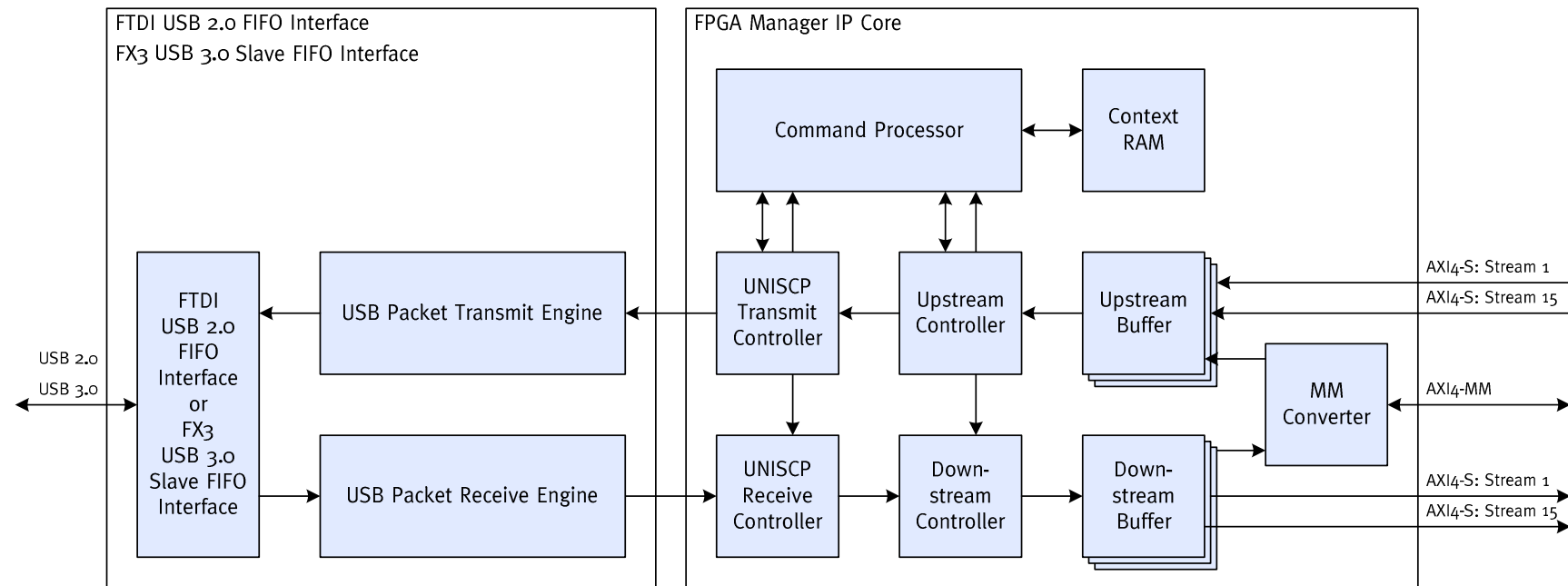
- All classes derive from **Base** which provides property set/get and callback functionality (completion & error reporting).
- There is only one instance of the **Control** class which is used to enumerate and open FPGA-Manager-capable devices.
- The **Device** class contains methods to create objects of the **Stream**, **MmAccess**, **Buffer**, **Transfer** and **Config** classes.
- The **Stream** class is used to send and receive stream data to/from the FPGA.
- The **MmAccess** class is used to read and write to a memory-mapped interconnect within the FPGA (e.g. accessing register banks, reading/writing external DDR3 memory).
- The **Buffer** class is used to allocate data buffers for more efficient data transfer (e.g. contiguous buffers for PCIe).
- The **Transfer** class is used to track the progress of non-blocking transfers.
- The **Config** class is used to download a FPGA bitstream.

FPGA Manager Block Diagram – Ethernet/UDP

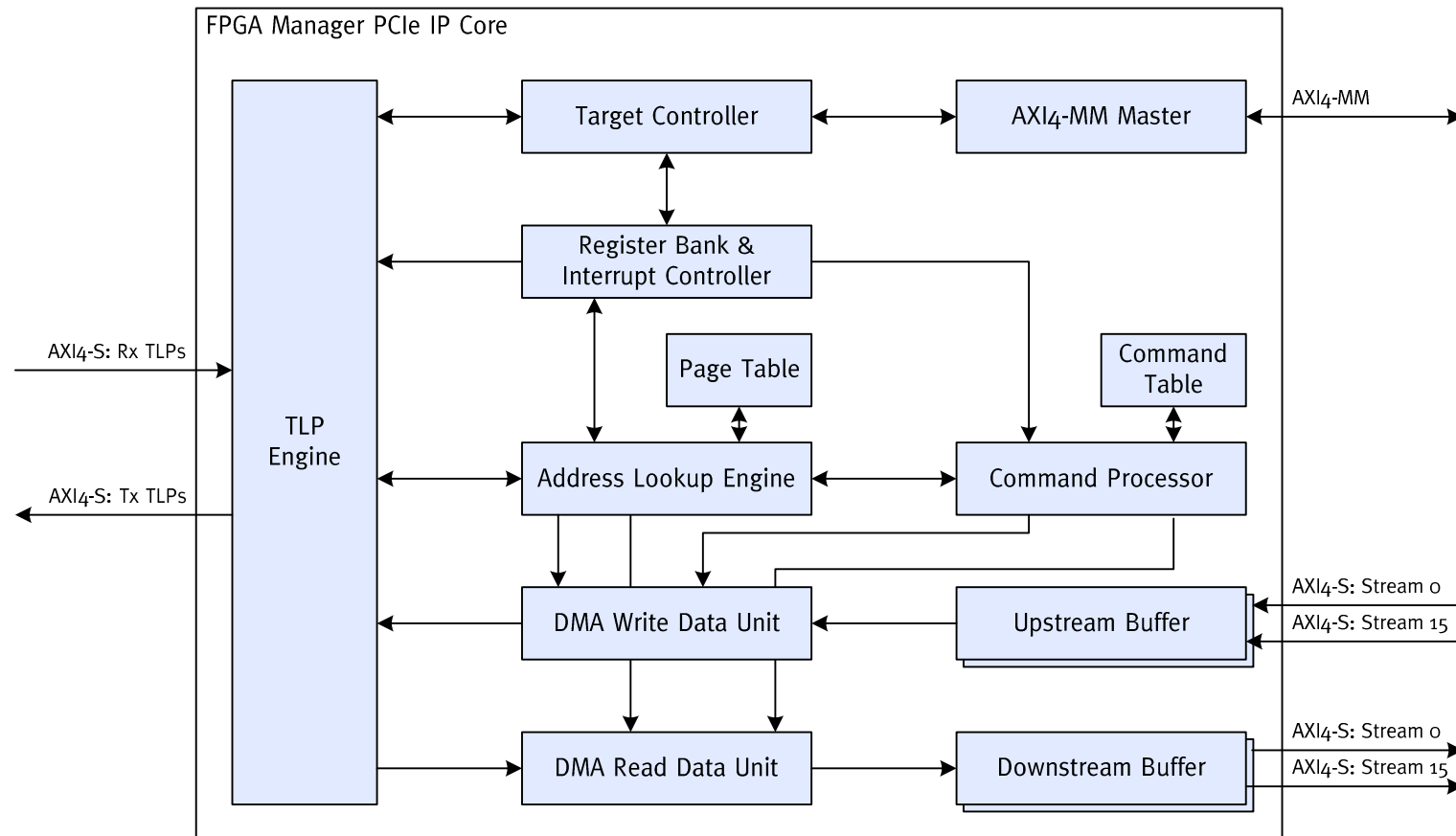


FPGA Manager

Block Diagram – USB 2.0 / 3.0



FPGA Manager Block Diagram – PCIe



FPGA Manager Integration into Xilinx Platform Studio

Xilinx Platform Studio (EDK_P.58f) - C:\Users\moberhol\Desktop\IpEdkComponents\edk\system.xmp - [System Assembly View]

File Edit View Project Hardware Device Configuration Debug Simulation Window Help

Navigator

Design Flow

Run DRCs

Implement Flow

Generate Netlist

Generate BitStream

Export Design

Simulation Flow

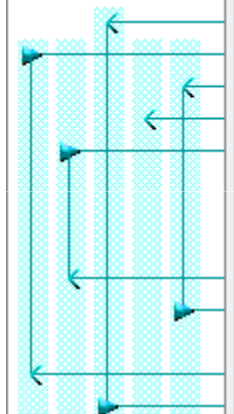
Generate HDL Files

Launch Simulator

IP Catalog

Description	IP Version	IP Type	Status
EDK Install			
+ Analog			
+ Arithmetic			
+ Bus and Bridge			
+ Clock, Reset and Interrupt			
+ Communication High-Speed			
+ Communication Low-Speed			
+ DMA and Timer			
+ Debug			
+ FPGA Reconfiguration			
+ General Purpose IO			
+ Interprocessor Communication			
+ Memory and Memory Controller			
+ PCI			
+ Peripheral Controller			
+ Processor			
+ Utility			
+ Verification			
+ Video and Image Processing			
Project Local PCores			
Enclustra FPGA Manager			
Enclustra FPGA Manager AXI4S to AXI4MM Converter IP Core	1.00.a	en_mgr_mm_conv	PRE_PRODUCTION
Enclustra FPGA Manager IP Core	1.00.a	en_mgr	PRE_PRODUCTION
Enclustra UDP/IP/Ethernet IP Core	1.00.a	en_udp_ip_eth	PRE_PRODUCTION
+ Video and Image Processing			

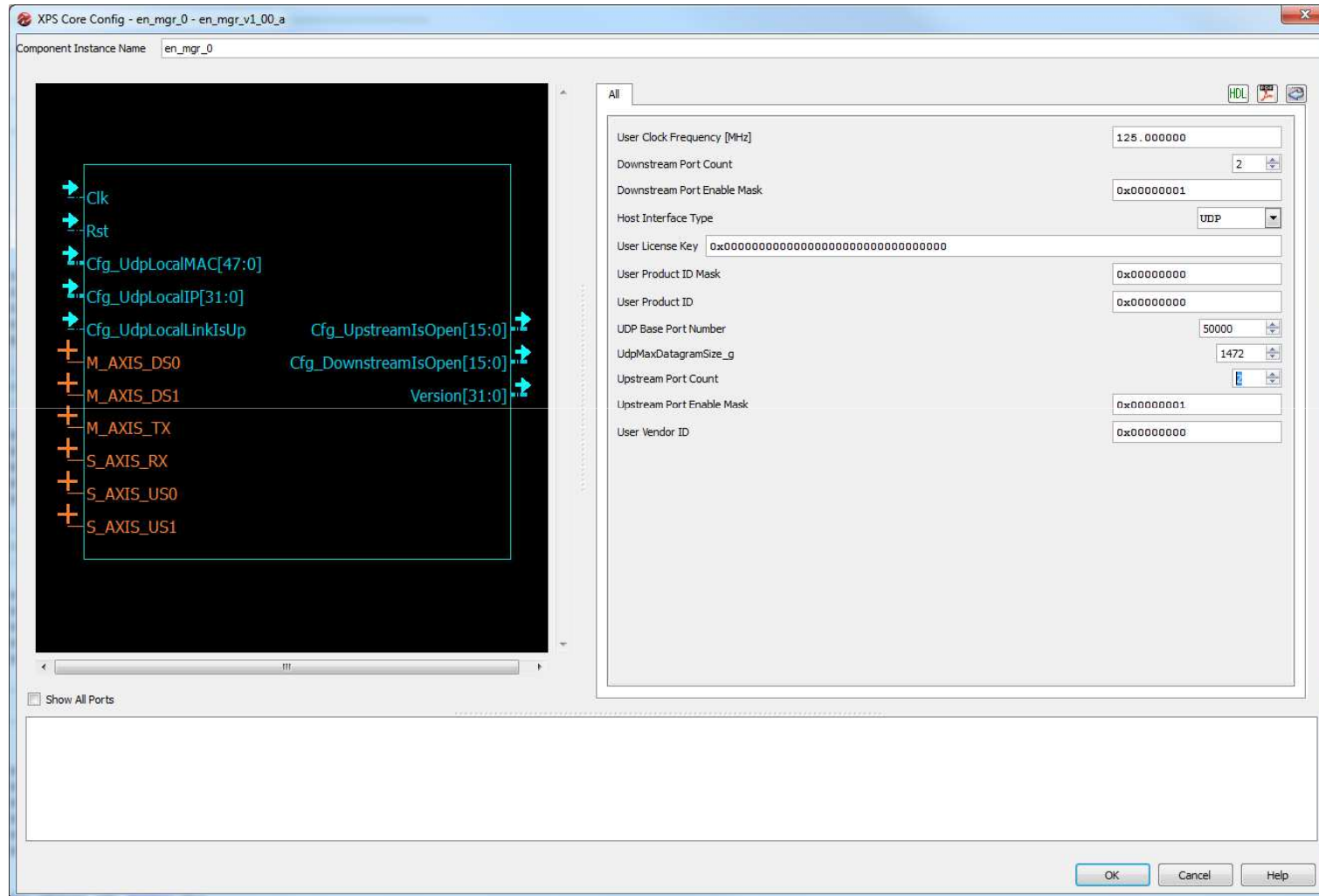
FPGA Manager Integration into Xilinx Platform Studio



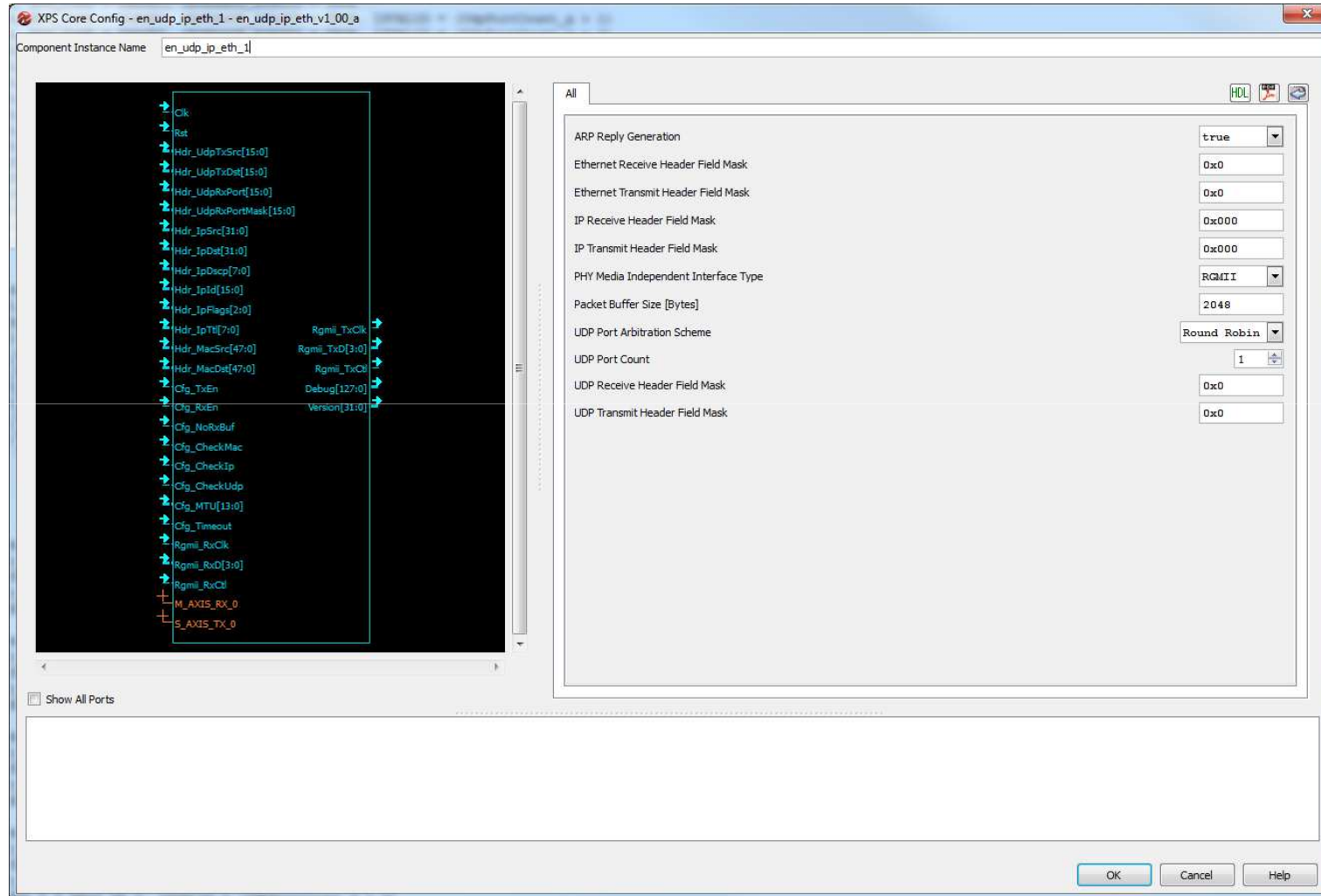
The diagram on the left shows a vertical stack of components with horizontal arrows indicating data flow between them. The components are: en_mgr_0, en_mgr_mm_conv_0, en_udp_ip_eth_0, clock_generator_0, and reset_0. The arrows show connections between the ports of these components as listed in the table.

Bus Interfaces		
Name	IP Version	Bus Name
en_mgr_0	1.00.a	
M_AXIS_TX		en_mgr_0_M_AXIS_TX
S_AXIS_RX		en_udp_ip_eth_0_M_AXIS_RX_0
M_AXIS_DS0		en_mgr_0_M_AXIS_DS0
M_AXIS_DS1		en_mgr_0_M_AXIS_DS1
S_AXIS_US0		en_mgr_mm_conv_0_M_AXIS
S_AXIS_US1		No Connection
en_mgr_mm_conv_0	1.00.a	
M_AXI		No Connection
M_AXIS		en_mgr_mm_conv_0_M_AXIS
S_AXIS		en_mgr_0_M_AXIS_DS0
en_udp_ip_eth_0	1.00.a	
M_AXIS_RX_0		en_udp_ip_eth_0_M_AXIS_RX_0
S_AXIS_TX_0		en_mgr_0_M_AXIS_TX
clock_generator_0	4.03.a	
reset_0	3.00.a	

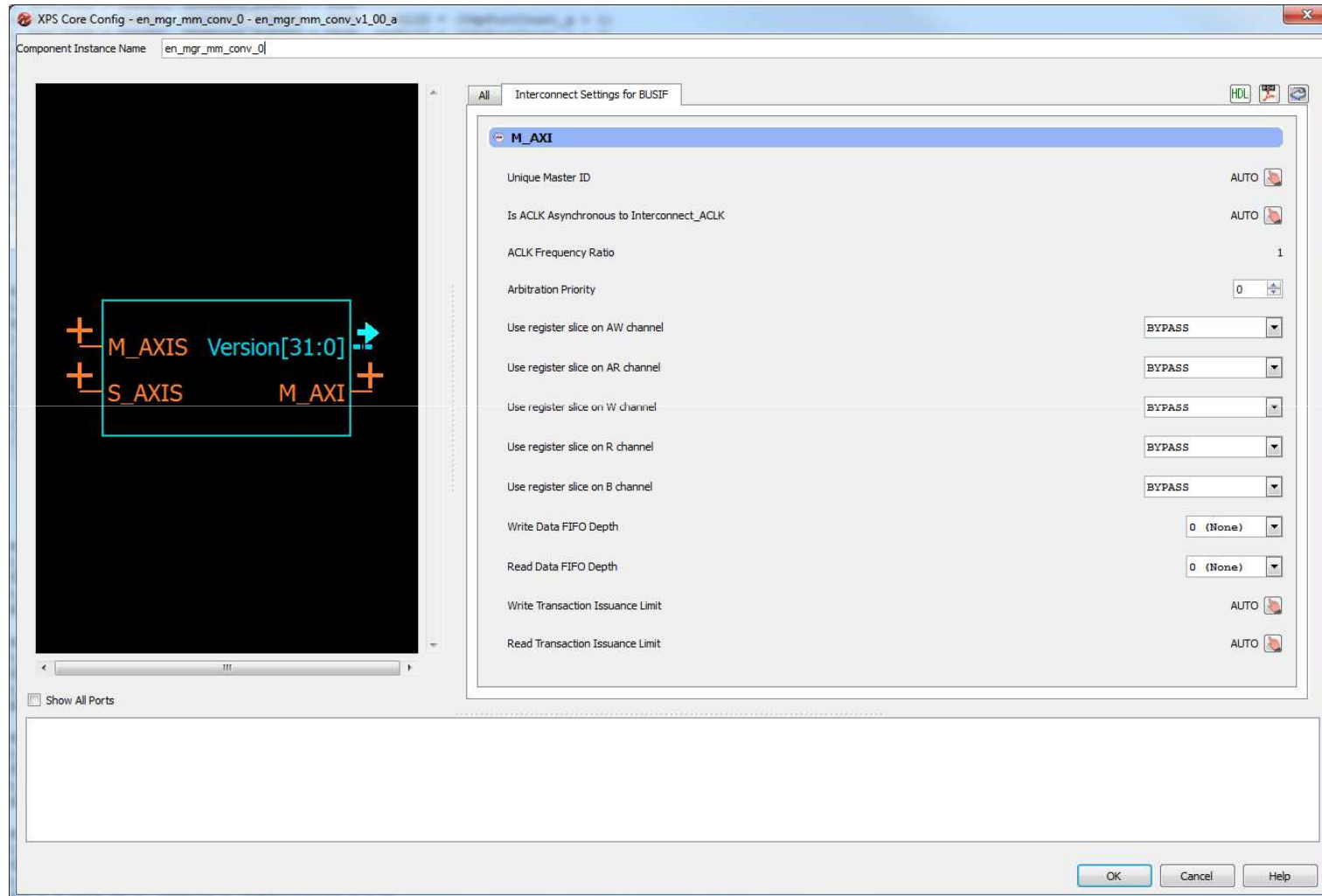
FPGA Manager Integration into Xilinx Platform Studio



FPGA Manager Integration into Xilinx Platform Studio



FPGA Manager Integration into Xilinx Platform Studio



FPGA Manager Licensing

Type	Ordering Code	Description
Base	EN-MGR	FPGA Manager 2 data streams, 1 memory-mapped interface C/C++/C# language support
Interfaces	EN-MGR-OPT-USB2FT	USB 2.0 FTDI support
	EN-MGR-OPT-USB3CY	USB 3.0 Cypress support
	EN-MGR-OPT-ETH1G	Fast & Gigabit Ethernet support
	EN-MGR-OPT-PCIE64	PCIe 64-Bit support
Targets	EN-MGR-OPT-XIL	Xilinx FPGA support
	EN-MGR-OPT-ALT	
Operating Systems	EN-MGR-OPT-WIN	Windows operating system support
	EN-MGR-OPT-LIN	Linux operating system support
Options	EN-MGR-OPT-ADV	Advanced feature pack 16 total streams, multiple memory-mapped interfaces, multiple stream widths
	EN-MGR-OPT-ML	MATLAB language support
Example (Set)	EN-MGR-USB3CY-XIL-WIN-BS EN-MGR-OPT-ADV-BS	FPGA Manager, C/C++/C# language support, USB 3.0 Cypress support, Xilinx FPGA support, Windows operating system support, Advanced feature pack, Binary site license
Example (Set)	EN-MGR-USB3CY-XIL-WIN-SS EN-MGR-OPT-ETH1G-SS	FPGA Manager, C/C++/C# language support, USB 3.0 Cypress, Fast & Gigabit Ethernet support, Xilinx FPGA support, Windows operating system support, Source-code site license

FPGA Manager Availability

	Xilinx		Xilinx	
	Spartan-6		7 Series	
	Beta	Production	Beta	Production
Ethernet	July 13	Dec 13	July 13	Dec 13
USB 2.0 (FTDI FT2232H)	July 13	Dec 13	July 13	Dec 13
USB 3.0 (Cypress EZ-USB FX3)	Sept 13	Dec 13	Sept 13	Dec 13
PCIe Gen1/2 1X/2X/4X (64-Bit)	TBD	TBD	Sept 13	Dec 13

	Windows XP x86 Windows 7 x86/x64		Linux Ubuntu LTS x64	
	Beta	Production	Beta	Production
Ethernet	July 13	Dec 13	Nov 13	Feb 14
USB 2.0 (FTDI FT2232H)	July 13	Dec 13	TBD	TBD
USB 3.0 (Cypress EZ-USB FX3)	Sept 13	Dec 13	TBD	TBD
PCIe Gen1/2 1X/2X/4X (64-Bit)	Nov 13	Feb 14	Sept 13	Dec 13

	Windows XP x86 Windows 7 x86/x64		Linux Ubuntu LTS x64	
	Beta	Production	Beta	Production
C (DLL)	July 13	Dec 13	Sept 13	Dec 13
C++	July 13	Dec 13	TBD	TBD
C#/VB.NET	July 13	Dec 13	TBD	TBD
MATLAB	Sept 13	Dec 13	TBD	TBD
Simulink	TBD	TBD	TBD	TBD
LabView	TBD	TBD	TBD	TBD

This preliminary release schedule can be re-prioritized according to customer demand.

FPGA Manager Performance

	Bandwidth		Latency	
	Upstream	Downstream	MM Write	MM Read
Fast Ethernet (MII, RMII)	10 MByte/s	10 MByte/s	1 msec	2 msec
Gigabit Ethernet (GMII, RGMII)	100 MByte/s	100 MByte/s	1 msec	2 msec
USB 2.0 (FTDI FT2232H)	30 MByte/s	30 MByte/s	2 msec	4 msec
USB 3.0 (Cypress EZ-USB FX3)	300 MByte/s	200 MByte/s	1 msec	2 msec
PCIe Gen1 1X	200 MByte/s	200 MByte/s	0.01 msec	0.01 msec
PCIe Gen1 2X	350 MByte/s	350 MByte/s	0.01 msec	0.01 msec
PCIe Gen1 4X	700 MByte/s	700 MByte/s	0.01 msec	0.01 msec
PCIe Gen2 1X	350 MByte/s	350 MByte/s	0.01 msec	0.01 msec
PCIe Gen2 2X	700 MByte/s	700 MByte/s	0.01 msec	0.01 msec
PCIe Gen2 4X	1300 MByte/s	1300 MByte/s	0.01 msec	0.01 msec

All numbers are preliminary and host system dependent.

FPGA Manager C# Example

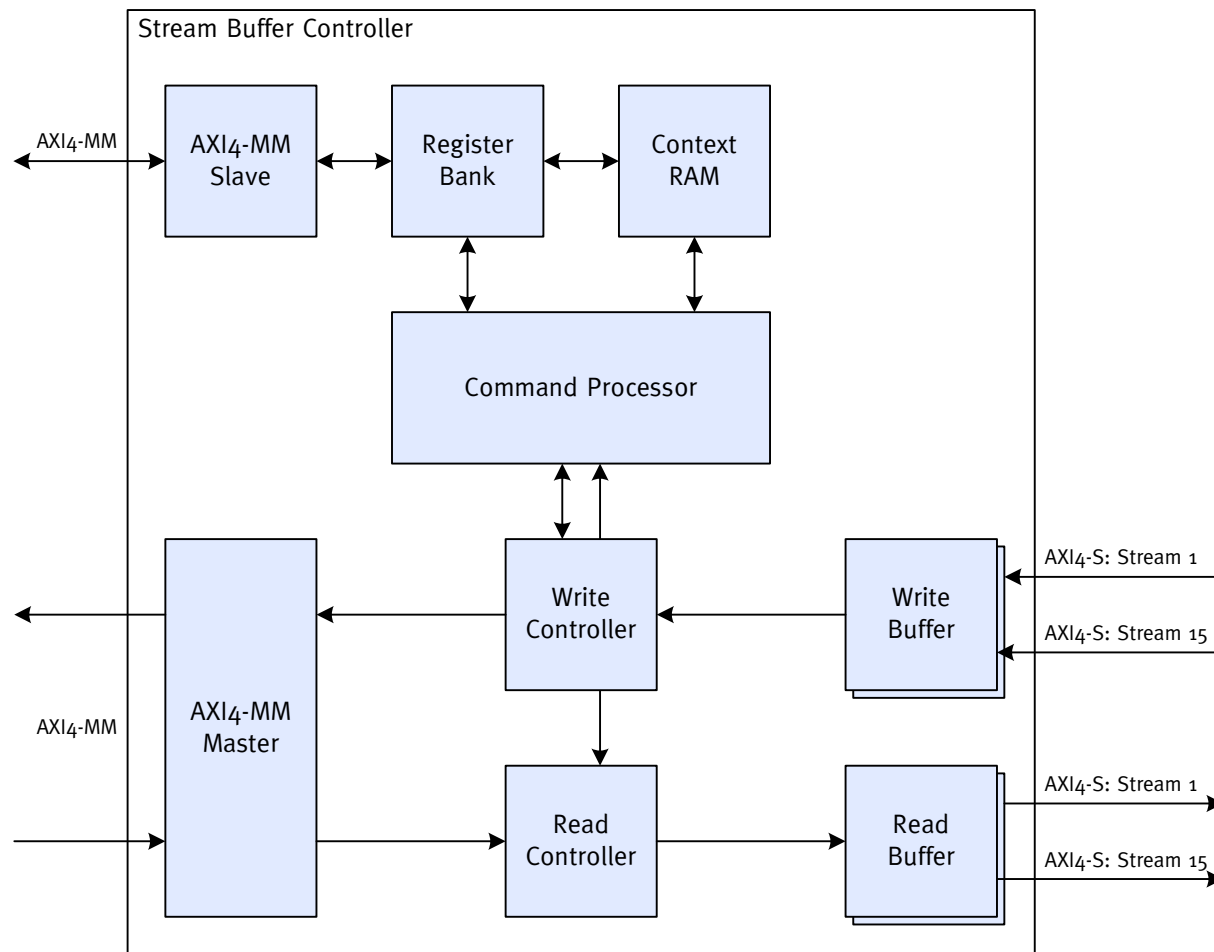
```
using Enclustra.Manager;
namespace my_namespace {
    class my_class {
        const char license[] = "<license-string>";
        int width = 1920; int height = 1080; int bytesPerPixel = 4;
        const int FPGA_MM_VIDEO_CAPTURE = 0x20000000;
        Device dev; MmAccess mm0; Stream st1; Buffer buf;
        public void main() {
            Control.Init(); Control.License(license);
            dev = Control.Open("uniscp://192.168.1.42/");
            mm0 = dev.GetMmAccess(0);
            st1 = dev.GetStream(1); st1.Open();
            buf = dev.CreateBuffer(width*height*bytesPerPixel);
            mm0.WriteRegister(FPGA_MM_VIDEO_CAPTURE, 1);
            st1.Read(buf);
            my_draw_image(width, height, buf.GetByteArray());
            st1.Close(); mm0.Close(); dev.Close();
        }
    }
}
```

- Enclustra Company Profile
 - FPGA Design Center
 - FPGA Solution Center
- FPGA-to-Host Communications
 - Multiplicity
 - Data Flow Semantics
 - Streaming & Memory-Mapped
 - Performance
 - Error Handling
 - PCIe specific
 - USB specific
 - Ethernet specific
- The Transparent Hat
- FPGA Manager
 - Overview
 - The Paradigms & Design Goals
 - UNISCP Packet Format
 - Host-side API
 - FPGA Block Diagrams
 - Xilinx Tool Integration
 - Licensing & Availability
 - C# Example
- **Stream Buffer Controller**
- **Use Cases**
- **Further Information**
- **Questions**

Stream Buffer Controller Features

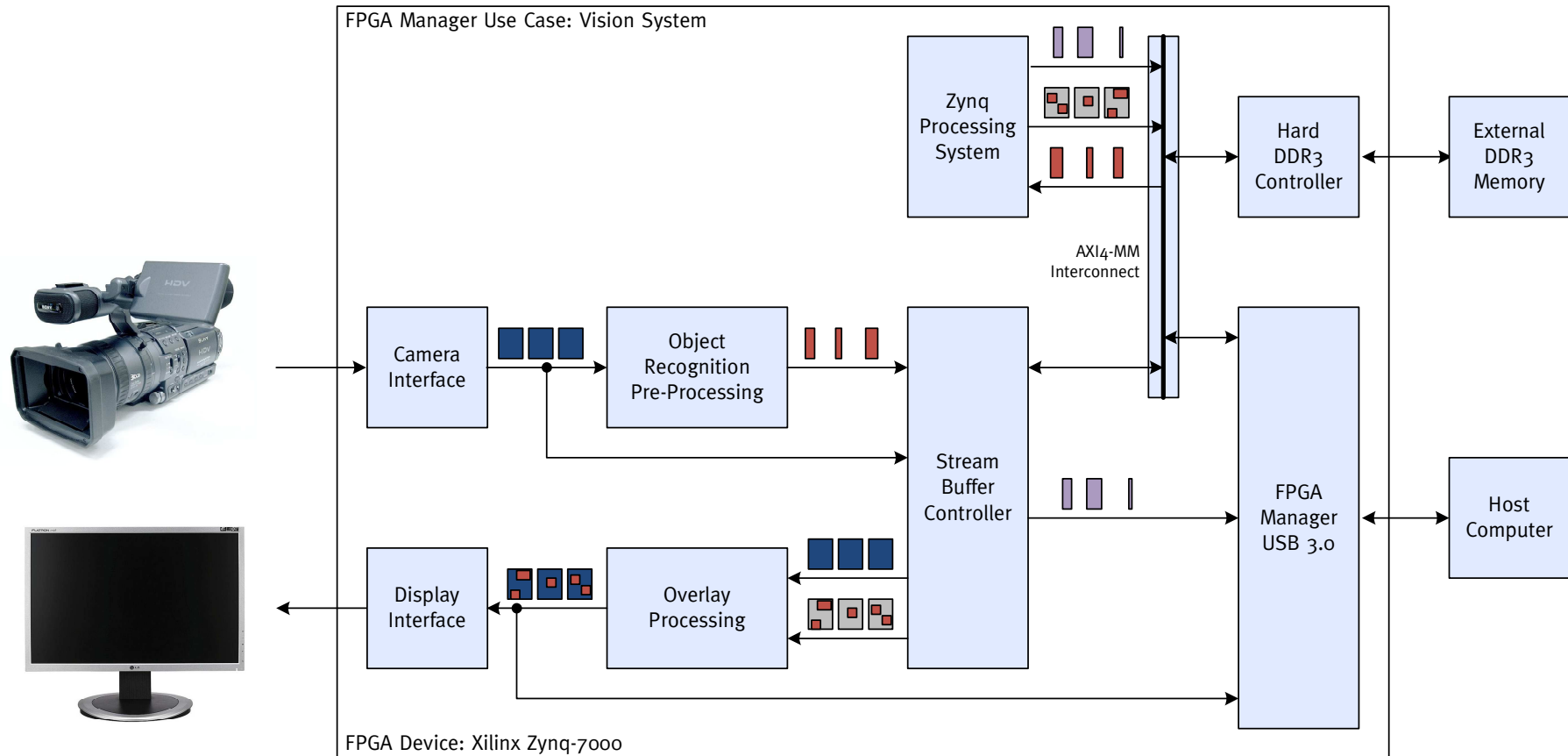
- Versatile AXI-Stream to AXI-MM DMA bridge with 16 independent streams
- Provides stream writing capability to external memory (e.g. DDR3)
- Provides stream reading capability from external memory (e.g. DDR3)
- A DMA write and read stream can be linked to provide virtual FIFO capability of up to 4 GByte
- Each stream uses a contiguous buffer in external memory in a circular fashion
- Each stream buffer can be independently configured (address, size, mode)
- A register bank allows to access the read/write pointers and configuration registers
- A software API is provided to configure and read/write data from/to the circular buffers from a soft or hard CPU within or outside the FPGA device

Stream Buffer Controller Block Diagram



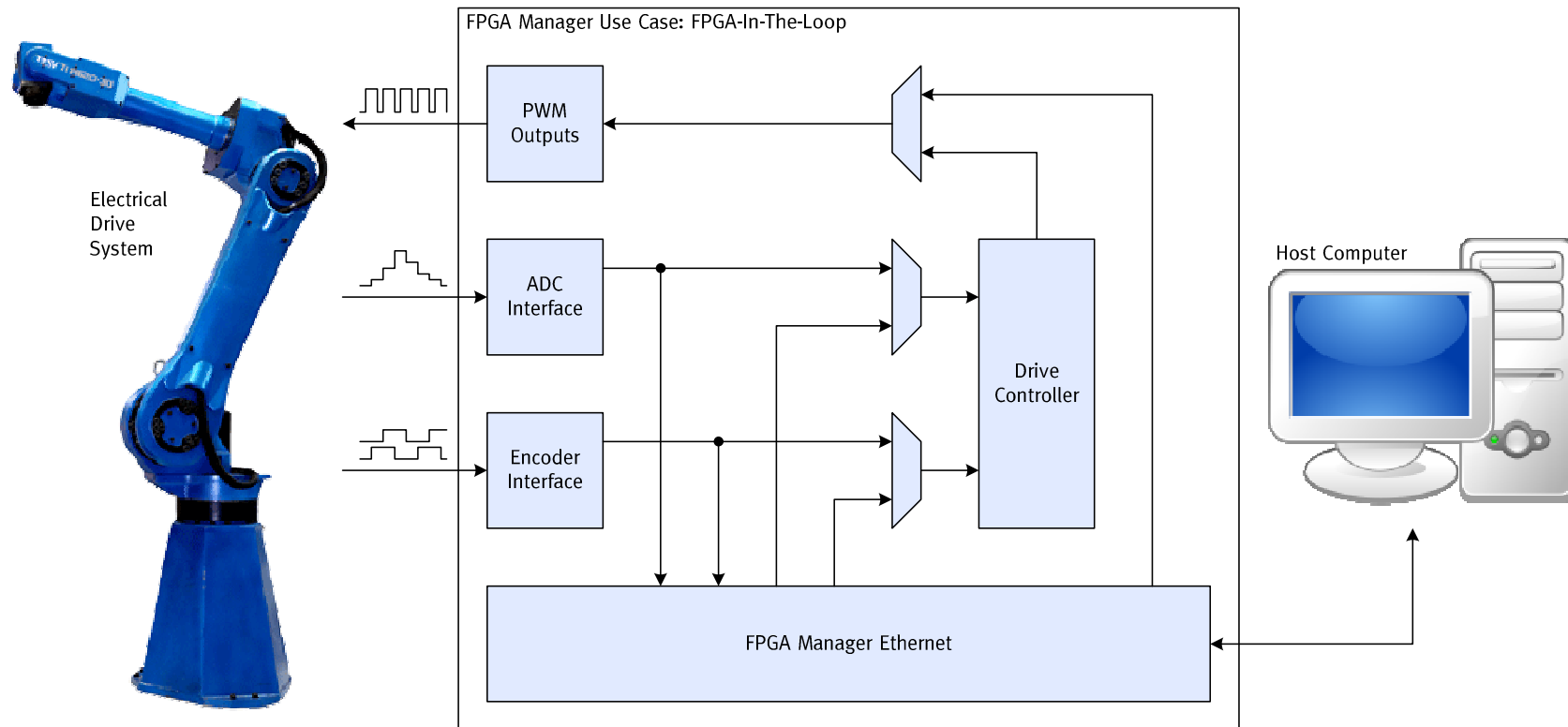
FPGA Manager

Use Case: Vision System



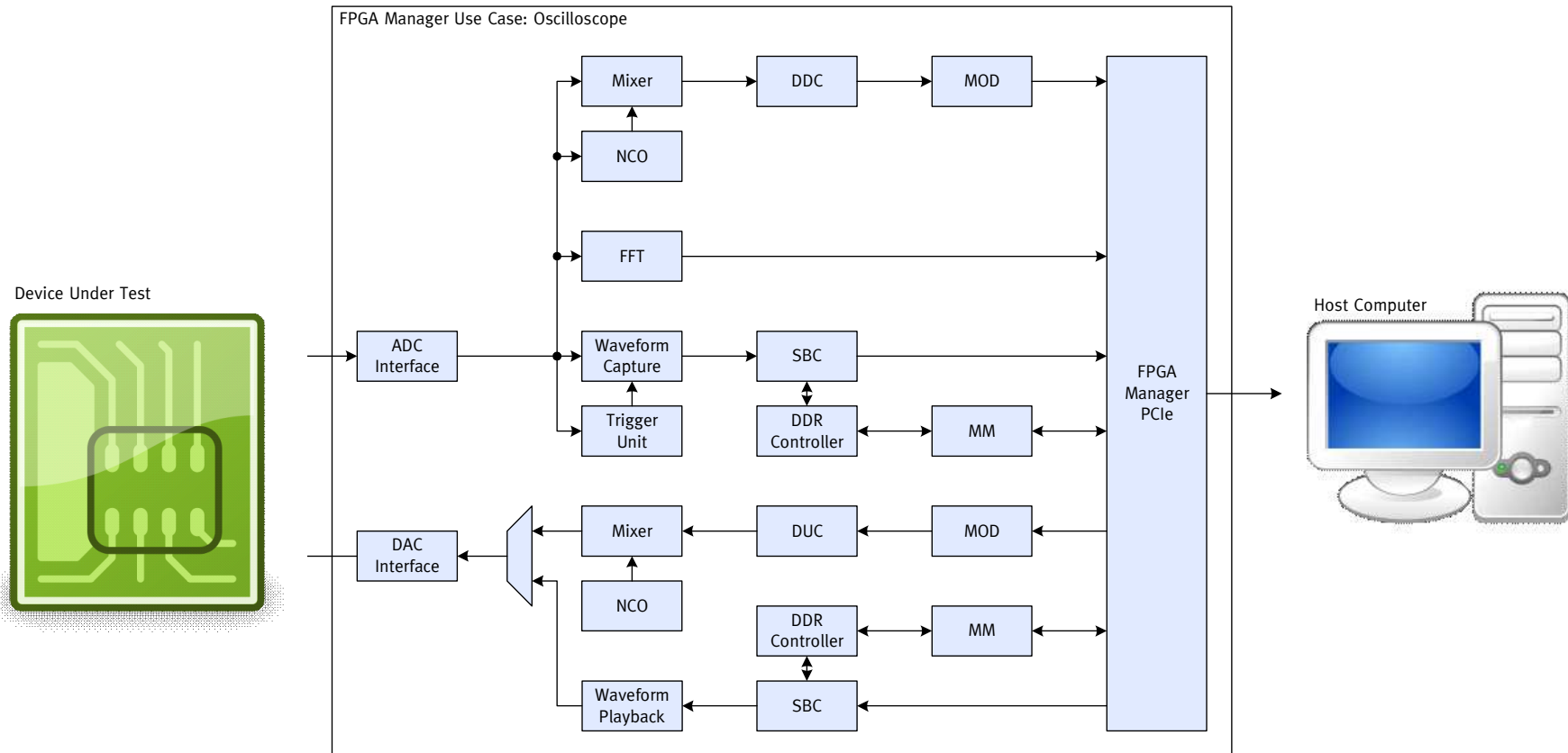
FPGA Manager

Use Case: FPGA-In-The-Loop



FPGA Manager

Use Case: Oscilloscope



- Enclustra
 - Mars ZX3 PM3 Kit: Zynq-7020 SoC FPGA with USB 3.0 and Gigabit Ethernet
 - Mars AX3 PM3 Kit: Artix-7 FPGA with USB 3.0 and Gigabit Ethernet
 - Mercury KX1 Starter Kit: Kintex-7 FPGA with USB 3.0, Gigabit Ethernet and PCIe
- Xilinx
 - AC701: Artix-7 FPGA with PCIe and Gigabit Ethernet
 - KC705: Kintex-7 FPGA with PCIe and Gigabit Ethernet
 - ZC702: Zynq-7020 SoC FPGA with Gigabit Ethernet
 - ZC706: Zynq-7045 SoC FPGA with PCIe and Gigabit Ethernet
- Digilent
 - Zedboard: Zynq-7020 SoC FPGA with Gigabit Ethernet
 - Atlys: Spartan-6 FPGA with Gigabit Ethernet

Information Sources & Further Information

- Xilinx
 - XAPP1052: Bus Master Performance Demonstration Reference Design for the Xilinx Endpoint PCI Express Solutions (Spartan-6 & Virtex-6)
 - UG482: 7 Series FPGAs GTP Transceivers User Guide
 - UG477: 7 Series FPGAs Integrated Block for PCI Express
- Enclustra
 - FPGA Manager Product Page:
www.enclustra.com/en/products/fpga-manager/
 - UDP/IP Ethernet IP Core:
www.enclustra.com/en/products/ip-cores/udp-ip-ethernet/
 - Mars PM3 USB 3.0 Quick-Start Kit:
<http://www.enclustra.com/en/products/quick-start-kits/mars-pm3-usb3-kit/>



Martin Heimlicher
Enclustra GmbH
heimlicher@enclustra.com
Tel. +41 43 343 39 43

Quarterly newsletter: subscribe@enclustra.com

Slides in PDF format:
<http://www.enclustra.com/en/company/publications/>

Next Events:

- Embedded World 2014
25. - 27. Februar 2014
Messezentrum Nürnberg

