

Recent Study on High Speed Serial Links for Multifunction Digital Array Receivers and Processors

Hernán A. Suárez

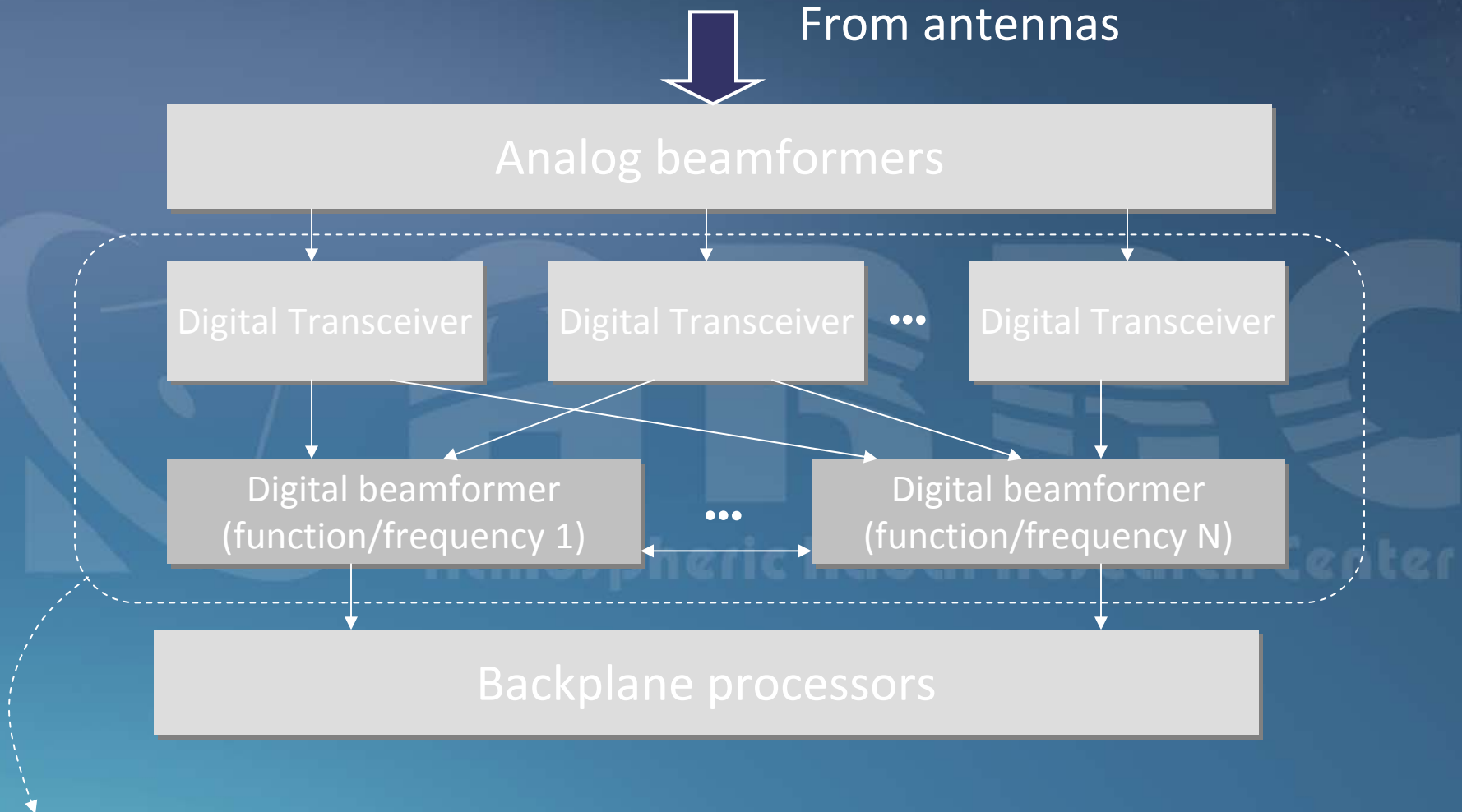
**School of Electrical and Computer Engineering
Radar Innovations Laboratory**

UNIVERSITY OF OKLAHOMA

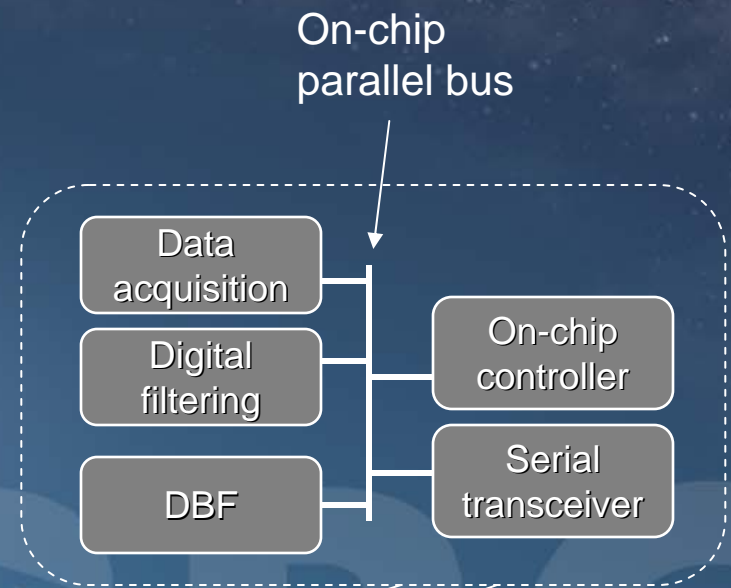
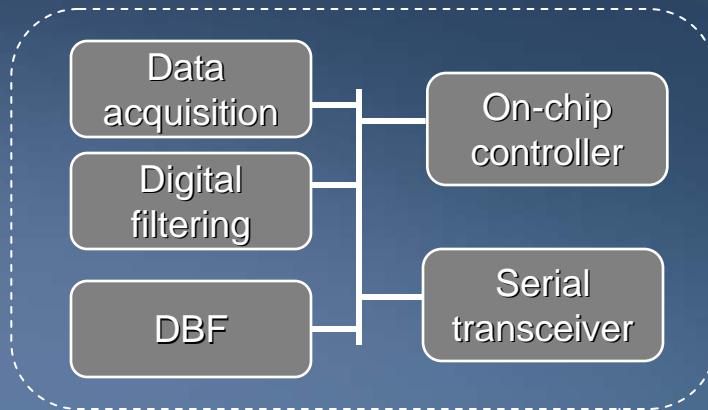
The Bottleneck of Digital Radar

- The bottleneck of digital array radar and its processors: associated with the massive data handling and transportation requirements (GBytes for future MPAR) and simultaneous multiple channel processing capabilities.
- Also in advanced signal processing (such as Space-Time Adaptive Processing) which are invoking the embedded distributed computing schemes, data need to be sent to right processing point on time, no information loss, and low-cost.
- IO power consumption >> processing power consumption
- Interconnection and interface are the bottleneck of scalability
- It is an urgent need to study the solutions for data transportations in embedded Systems to make the future multi-functional radar possible.

Overview of Future MPAR Architecture

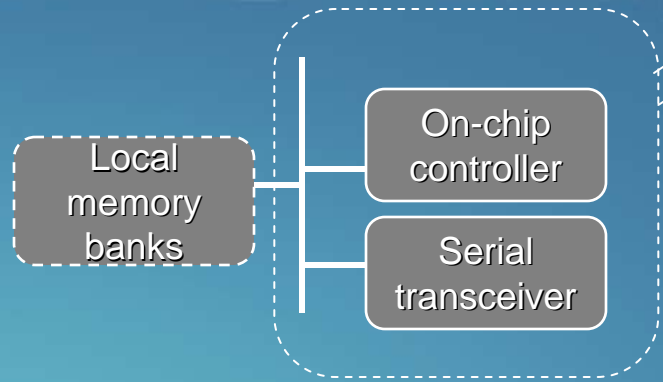


A mesh network model



DREN can be reconfigured or do self-reconfigure during operation

Smaller, cheaper and faster because Operations are within chip-boundary

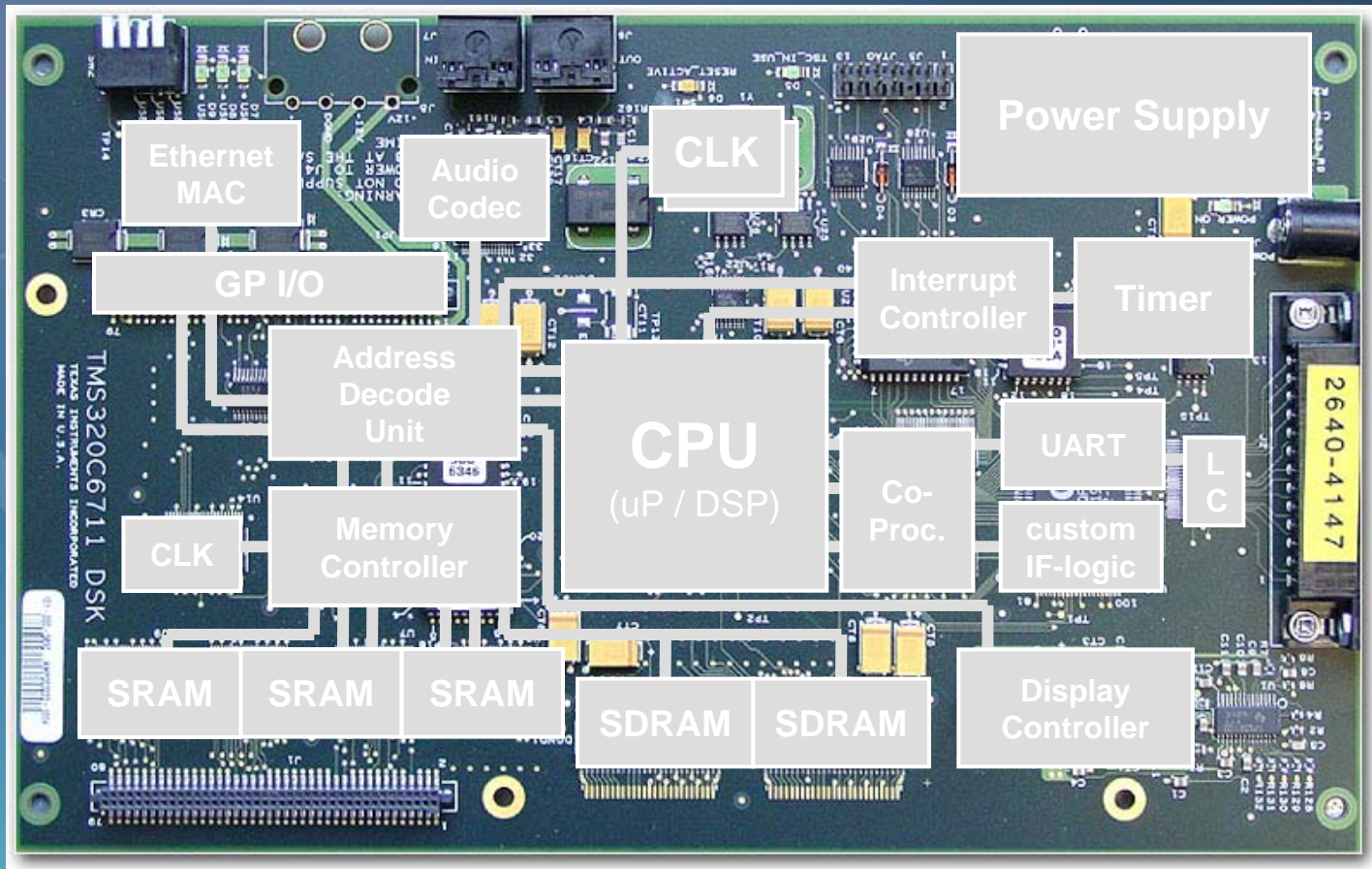


Digital Radar End Node (DREN) concept

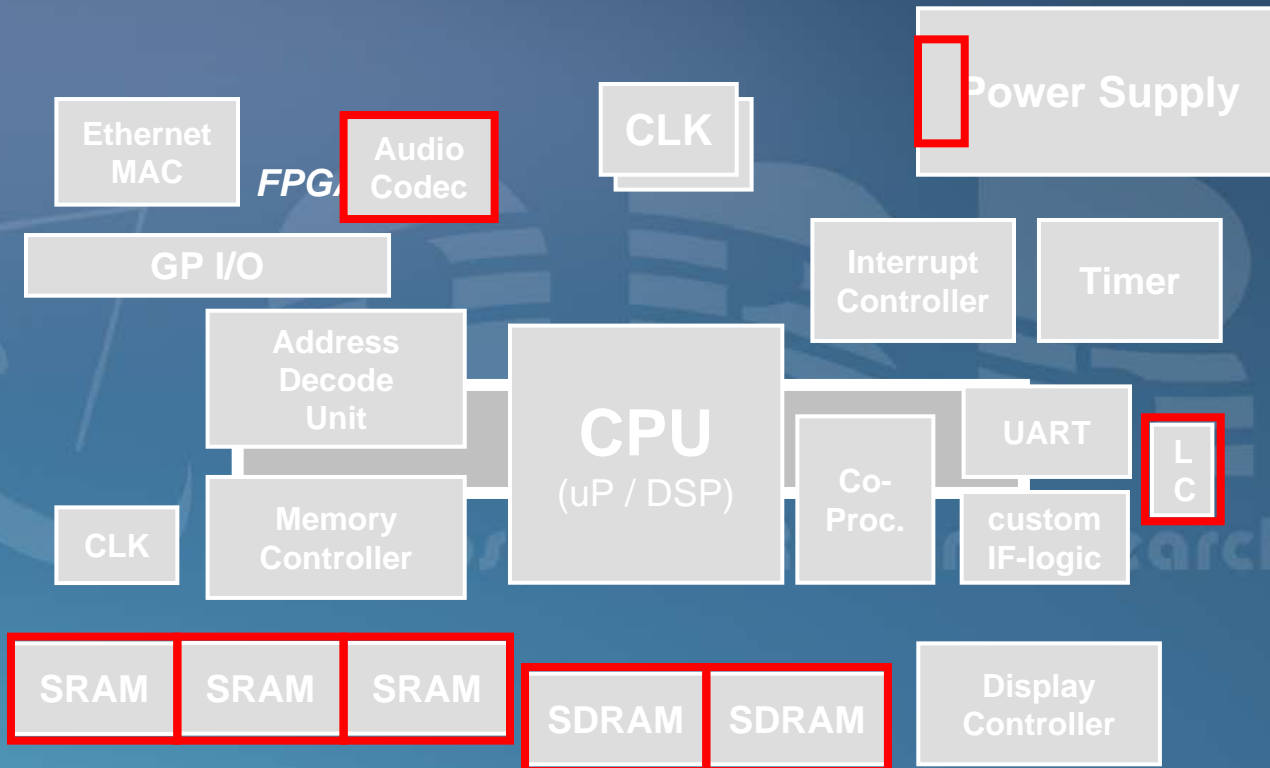
EMBEDDED SYSTEMS

- An embedded system is a computing system (other than a general-purpose computer) with the following general characteristics
 - Single-functioned
 - Typically, is designed to perform predefined function
 - Tightly constrained
 - Tuned for low cost
 - Single-to-fewer components based
 - Performs functions fast enough
 - Consumes minimum power
 - Reactive and real-time
 - Must continually monitor the desired environment and react to changes
 - Hardware and software co-existence

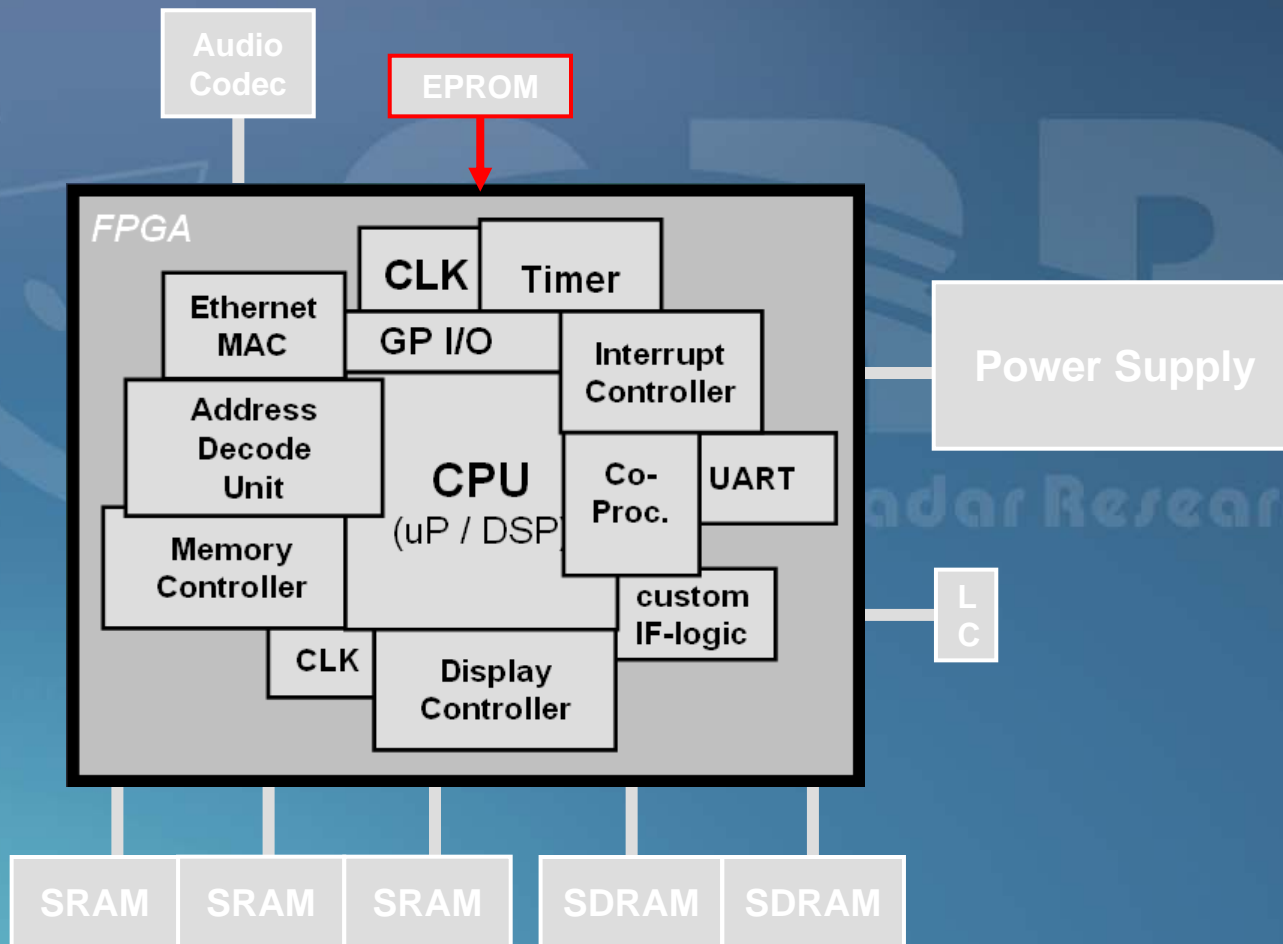
TRADITIONAL EMBEDDED SYSTEM



NEXT STEP



RECONFIGURABLE SYSTEM ON A CHIP

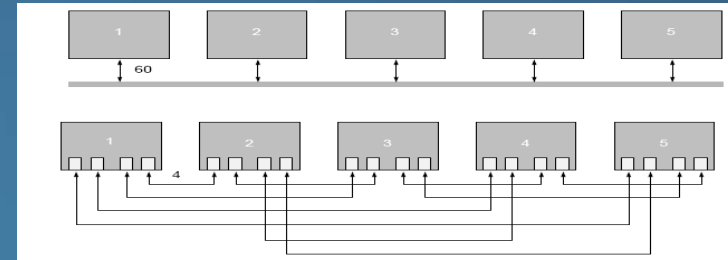


HIGH SPEED SERIAL LINKS

Why serial links?

- More flexible, thinner cabling
- Topologies that promise to scale to the needs of the end user
- Reduced system costs because of fewer PCB traces, and lower pin/wire count.
- Predictable and reliable signaling schemes
- Exceptional bandwidth per pin.

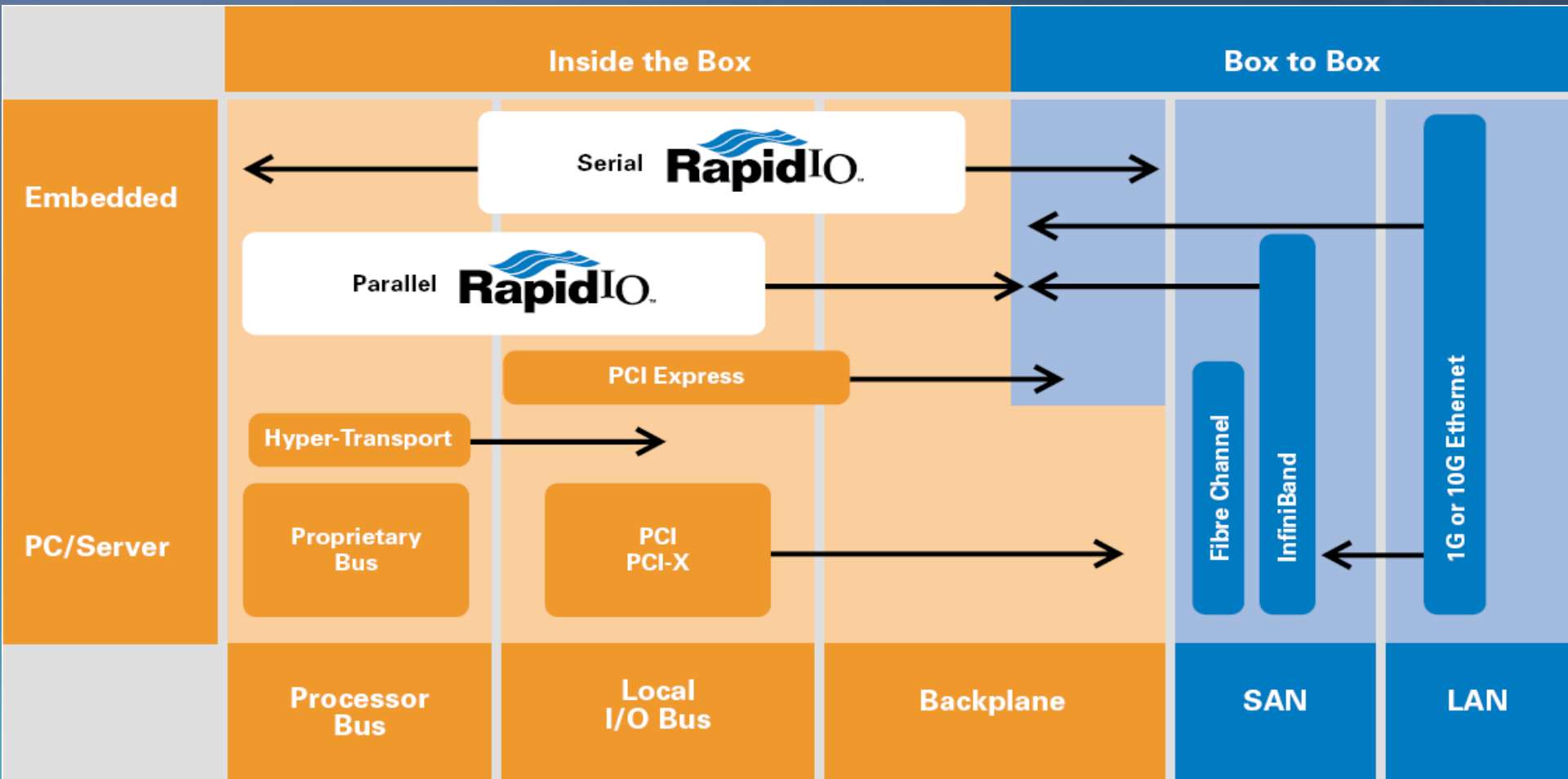
Currently studying:
1. Serial RapidIO
2. Aurora



RapidIO Overview

- A high-performance Point-to-point, packet-switched, interconnect technology
- Controlled by The RapidIO Trade Association (Xilinx, Freescale, Tundra, Mercury, Altera, Ericsson, Curtiss-Wright, etc.)
- Compatible with the most popular integrated communications processors, host processors, and networking digital processors
- Chip-to-chip and board-to-board communications at performance levels scaling to ten Gigabits per second and beyond.
- Networks of any topology possible (generally using switches)

Where Does Serial RapidIO Fit?



Advantages of Serial RapidIO(SRIO)

➤ *SRIO vs PCI, PCI-X and PCI – Express*

- Lower-pin number
- Higher protocol efficiency
- Better for embedded application

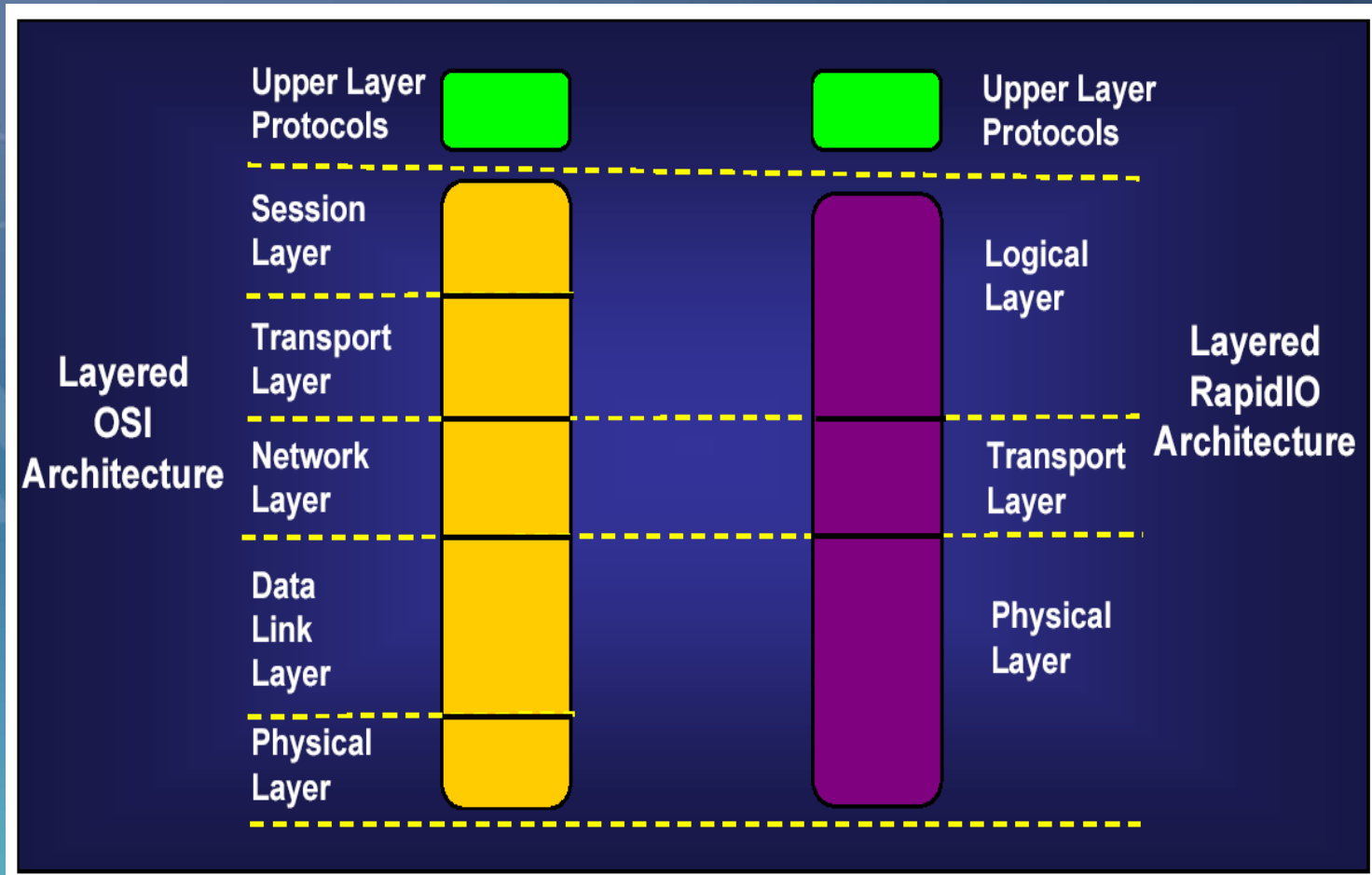
➤ *SRIO vs Ethernet and Gigabit Ethernet*

- Provides 9.3 Gbps of effective data rate vs 795 Mbps for GigE
- Lower latency and faster speed
- Better efficiency for small form factor

➤ *SRIO vs 10 Gbit (1000 Base-T) Ethernet:*

- Provides 6.7 Gbps using small 64 bytes PDU vs 3.3Gbps for 10 GigE.
- 95% efficient
- Save ~80% power consumption, lower cost, better efficiency .

RapidIO Layered Architecture



SRIO Layers: Logical and Transport

- Logical Layer- contains protocols necessary for **end points** to process a transaction
 - I/O Logical Layer
 - Memory mapped remote reads/writes
 - No direct support for cache coherence
 - Message Passing Logical Layer
 - Uses sends and receives with explicit processor ID supplied as sender/receiver ID
 - Data messages- “normal” sends/receives
 - Doorbell messages- very short, low overhead “special” message (no payload)
 - Globally Shared Memory Logical Layer
 - Hardware-based cache coherence
 - Memory-based mechanism for directory-based coherence
 - RIO is targeted towards the other two logical layers
 - All three logical layers may co-exist on the same network
- Common Transport Layer
 - All logical layers use the common transport layer
 - Simple spec provides information to route a packet from source to destination

SRIO Layers: Physical

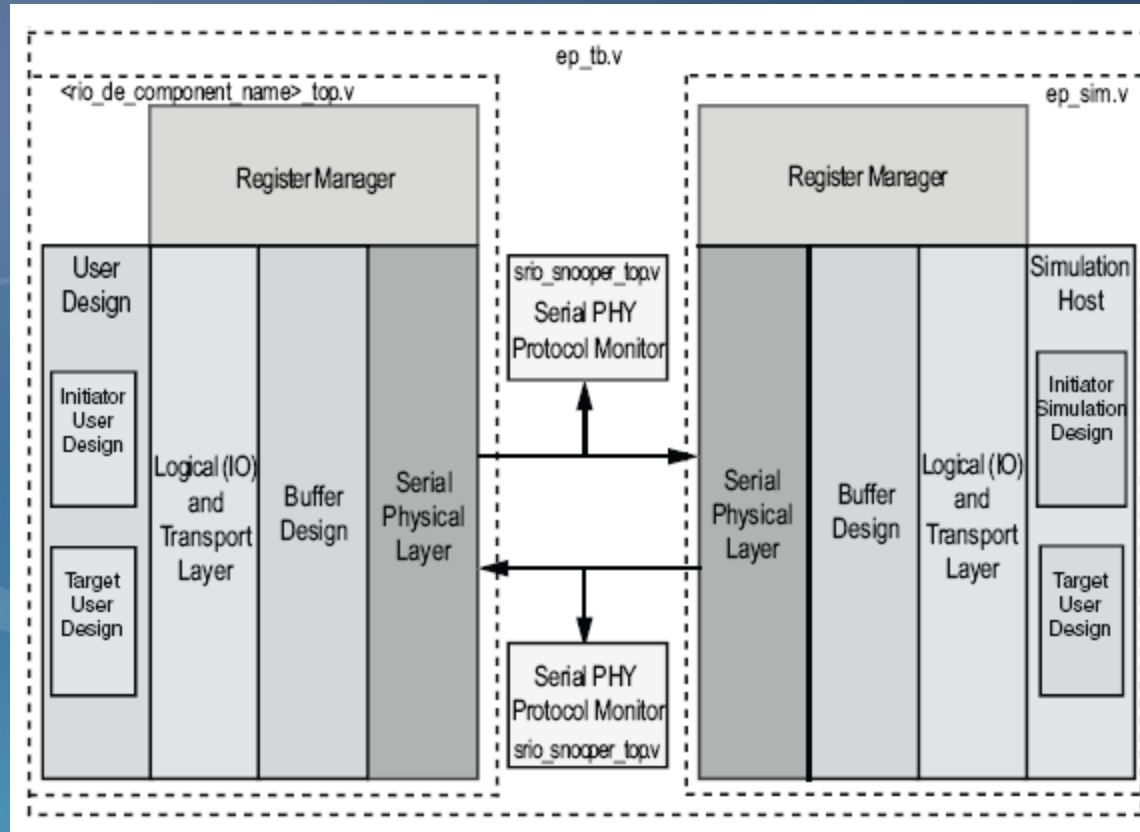
- Used for link-level communication
 - LP-Serial 1x-4x
 - 1x provides 1 data lane, 4x provides 4 data lanes
 - Embeds clock with data
 - Both variants use differential signaling
- Many services provided by both physical layers
 - flow control, error management, and signal acknowledgement (all between linked devices)

Serial RapidIO						
Clock Rate	1-bit Wide			4-bit Wide		
	PEAK	Sustained 32 byte Op	Sustained 256 byte Op	PEAK	Sustained 32 byte Op	Sustained 256 byte Op
1.25 GHz	2 Gb	1 Gb	1.8 Gb	8 Gb	4 Gb	7.2 Gb
2.5 GHz	4 Gb	2 Gb	3.6 Gb	16 Gb	8 Gb	14.4 Gb
3.125 GHz	5 Gb	2.5 Gb	4.5 Gb	20 Gb	10 Gb	18 Gb

Tools Used for Serial RIO Implementation

- Xilinx software tools (ISE, IP core generator, Impact, ChipScope)
- Two Xilinx Virtex-II Pro XC2VP30-7 FPGA boards
- Agilent Logic Analyzer 16902A
- Two Agilent differential flying lead sets E5381A
- Two Diligent FX2-BB boards
- Two Serial ATA cables

SRIO Testbench



Synthesis Results

p30_host.ise - [Design Summary]

Timing Summary:

Speed Grade: -7

Minimum period: 5.590ns (Maximum Frequency: 178.883MHz)
 Minimum input arrival time before clock: 4.862ns
 Maximum output required time after clock: 12.427ns
 Maximum combinational path delay: 6.531ns

Timing Detail:

All values displayed in nanoseconds (ns)

Timing constraint: Default period analysis for Clock 'sys_clkp'
 Clock period: 4.695ns (frequency: 212.982MHz)
 Total number of paths / destination ports: 166596 / 16678

Delay: 4.695ns (Levels of Logic = 6)
 Source: rio_de_wrapper/phy_wrapper/phy_1x_ser/U0/phy_1x_ser_gen.phy_ser/u_oplm_top
 Destination: rio_de_wrapper/phy_wrapper/phy_1x_ser/U0/phy_1x_ser_gen.phy_ser/u_oplm_top
 Source Clock: sys_clkp rising 0.3X
 Destination Clock: sys_clkp rising

Data Path: rio_de_wrapper/phy_wrapper/phy_1x_ser/U0/phy_1x_ser_gen.phy_ser/u_oplm_top/u_oplm_top

Cell:in->out	fanout	Gate	Net	Delay	Delay	Logical Name (Net Name)
FDC:C->Q	123	0.370	0.937	U0/phy_1x_ser_gen.phy_ser/u_oplm_top/u_oplm_pcs_t		
LUT3_L:I0->L0	1	0.275	0.118	U0/phy_1x_ser_gen.phy_ser/u_oplm_top/u_oplm_pcs_t		
LUT3:I2->O	1	0.275	0.349	U0/phy_1x_ser_gen.phy_ser/u_oplm_top/u_oplm_pcs_t		
LUT4_L:I2->L0	1	0.275	0.118	U0/phy_1x_ser_gen.phy_ser/u_oplm_top/u_oplm_pcs_t		
LUT4:I2->O	8	0.275	0.576	U0/phy_1x_ser_gen.phy_ser/u_oplm_top/u_oplm_pcs_t		
LUT2:I1->O	1	0.275	0.369	U0/phy_1x_ser_gen.phy_ser/u_oplm_top/u_oplm_pcs_t		
LUT4:I3->O	1	0.275	0.000	U0/phy_1x_ser_gen.phy_ser/u_oplm_top/u_oplm_pcs_t		
FDC:D		0.208		U0/phy_1x_ser_gen.phy_ser/u_oplm_top/u_oplm_pcs_t		
Total		4.695ns (2.228ns logic, 2.467ns route)		(47.5% logic, 52.5% route)		

Suite 10.1 | Design Summary | srio_xvp30_v1_host_top.v | Boundary Scan

Others : 18
 # icon2 : 1
 # OPAD : 17

Device utilization summary:

Selected Device : 2vp30ff896-7

Number of Slices:	6629	out of 13696	48%
Number of Slice Flip Flops:	7281	out of 27392	26%
Number of 4 input LUTs:	9032	out of 27392	32%
Number used as logic:	8490		
Number used as Shift registers:	66		
Number used as RAMs:	476		
Number of IOs:	63		
Number of bonded IOBs:	51	out of 556	9%
Number of BRAMs:	10	out of 136	7%
Number of GCLKs:	3	out of 16	18%
Number of GTs:	1	out of 8	12%
Number of DCMs:	1	out of 8	12%

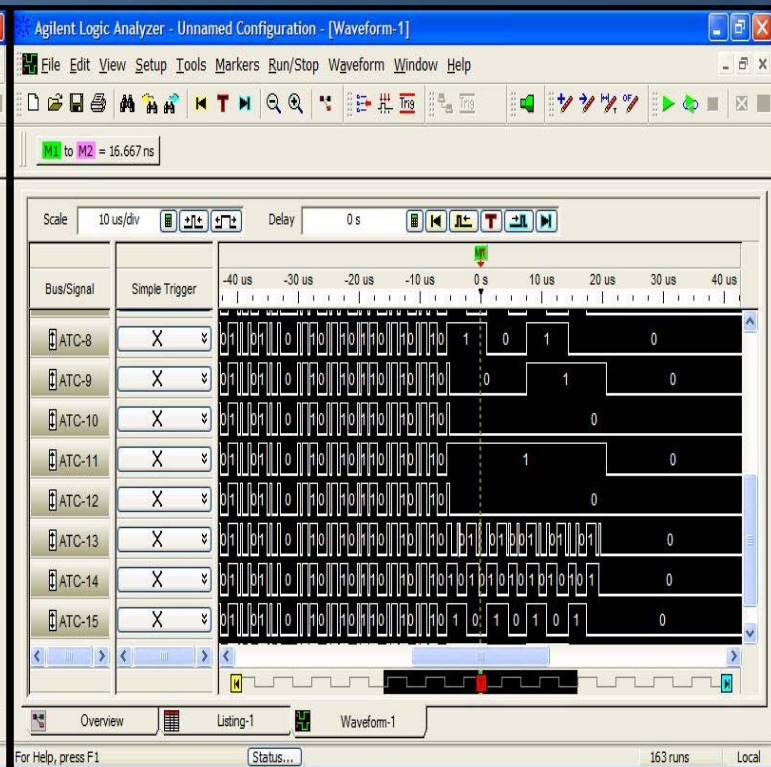
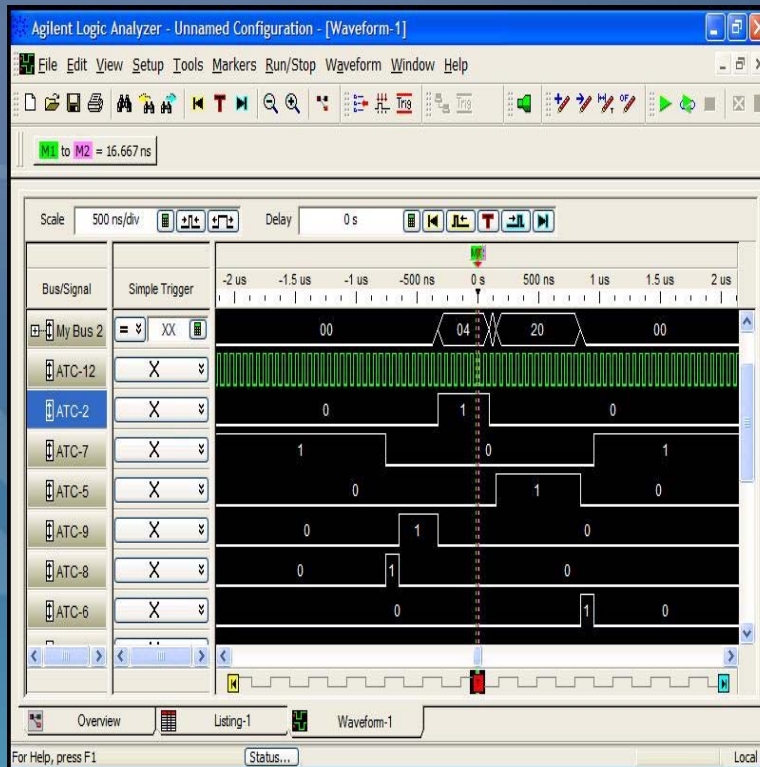
Partition Resource Summary:

No Partitions were found in this design.

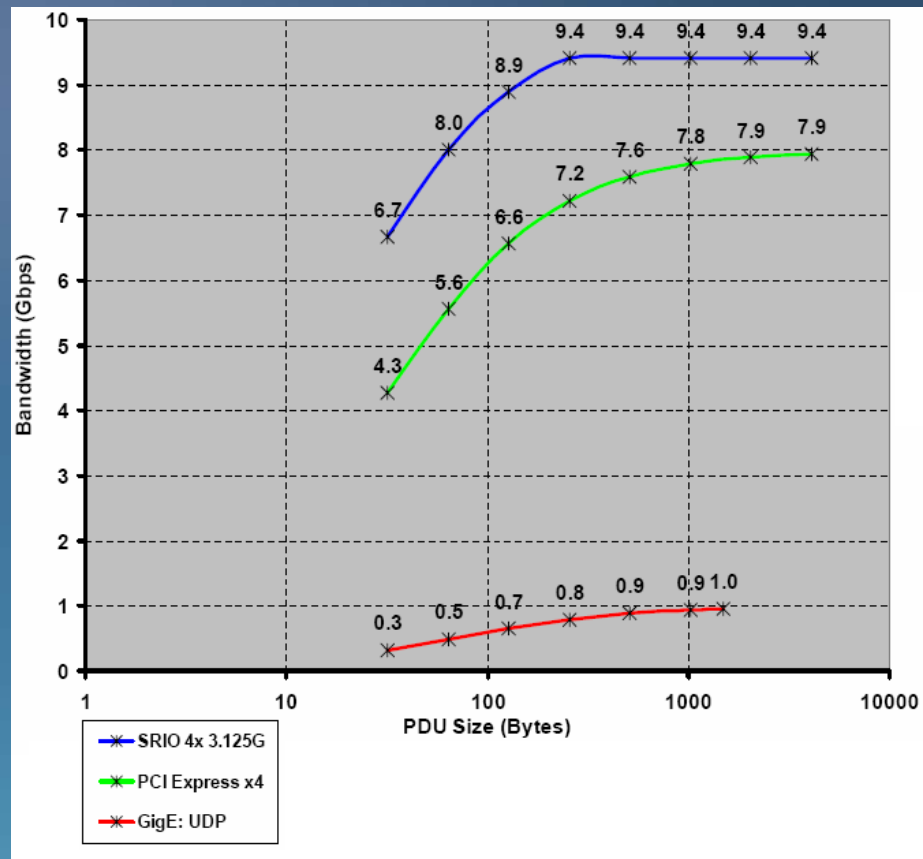
TIMING REPORT

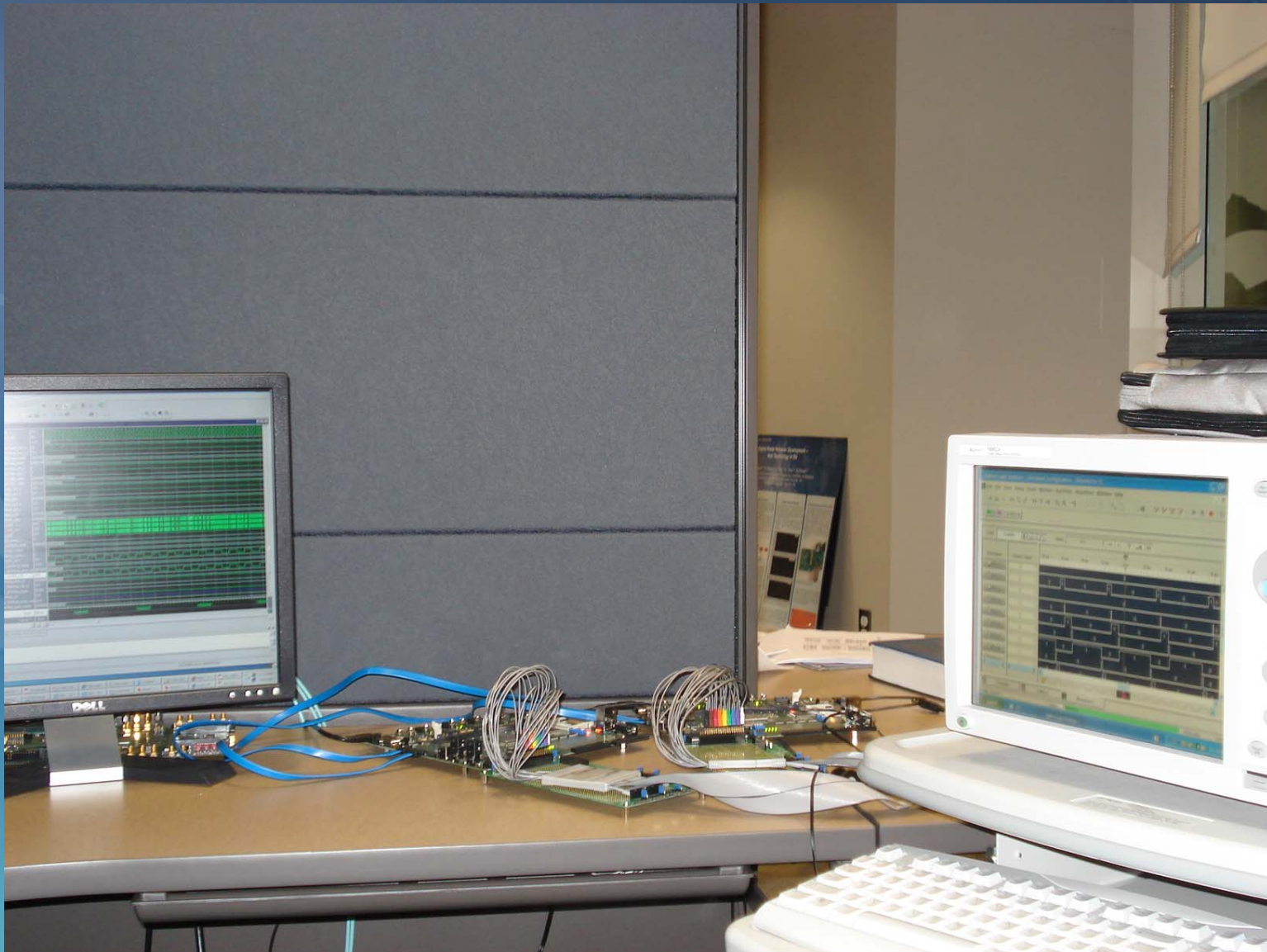
NOTE: THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
 FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE REPORT
 GENERATED AFTER PLACE-and-ROUTE.

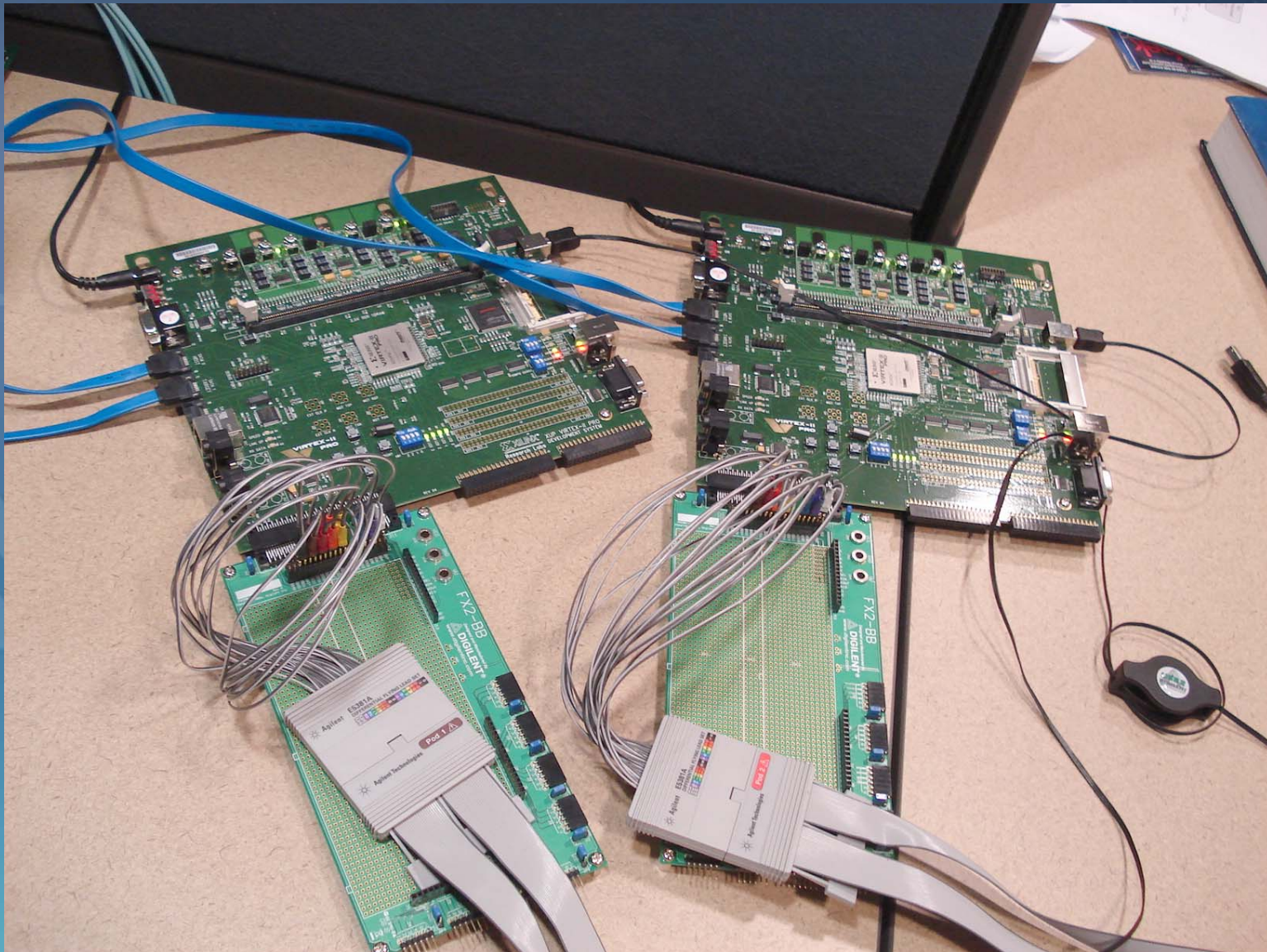
SRIO Signals Captured by the Logic Analyzer



SRIO Bandwidth Performance







AURORA PROTOCOL

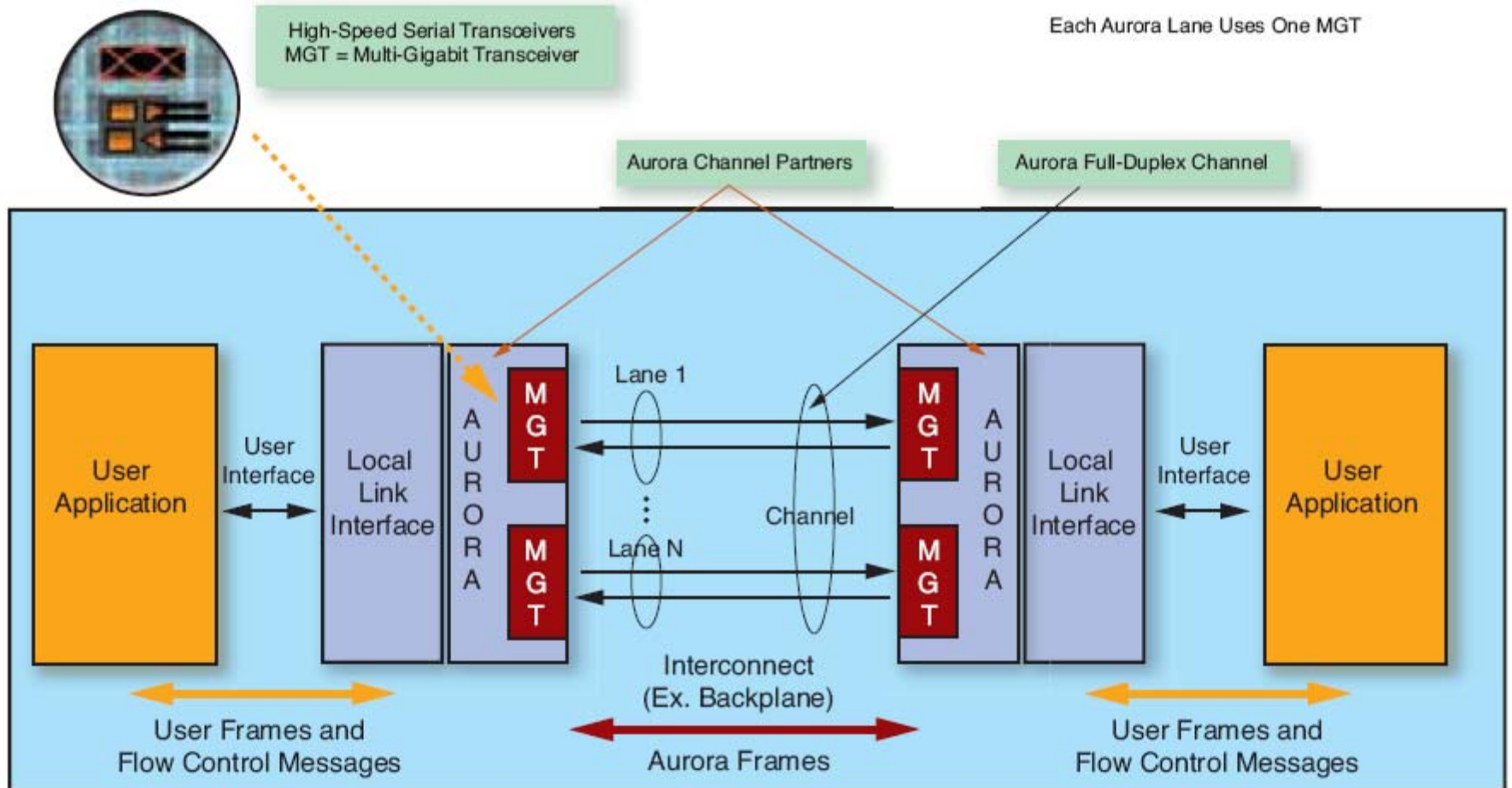
DEFINES: (point to point only)

- Physical layer interface: electrical level and symbol coding
- Initialization and error handling
- Data stripping
- Link layer
- Flow control

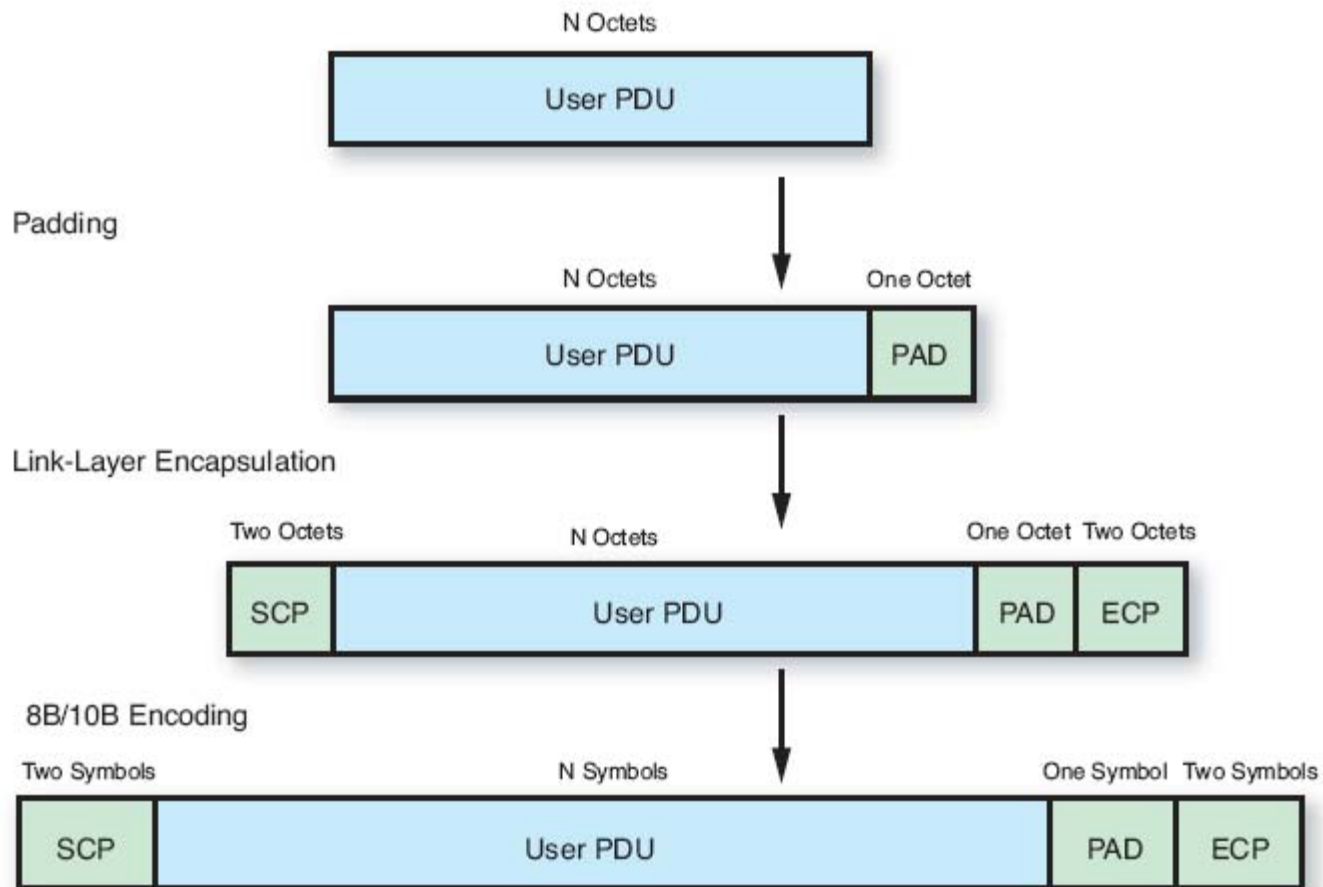
DOES NOT DEFINE:

- Error detection and recovery : does not define a mechanism for detecting error in users PDUs.
- Data switching: does not define an address scheme.

SERIAL CONNECTIVITY WITH AURORA IP



SERIAL CONNECTIVITY WITH AURORA IP



SERIAL CONNECTIVITY AURORA IP

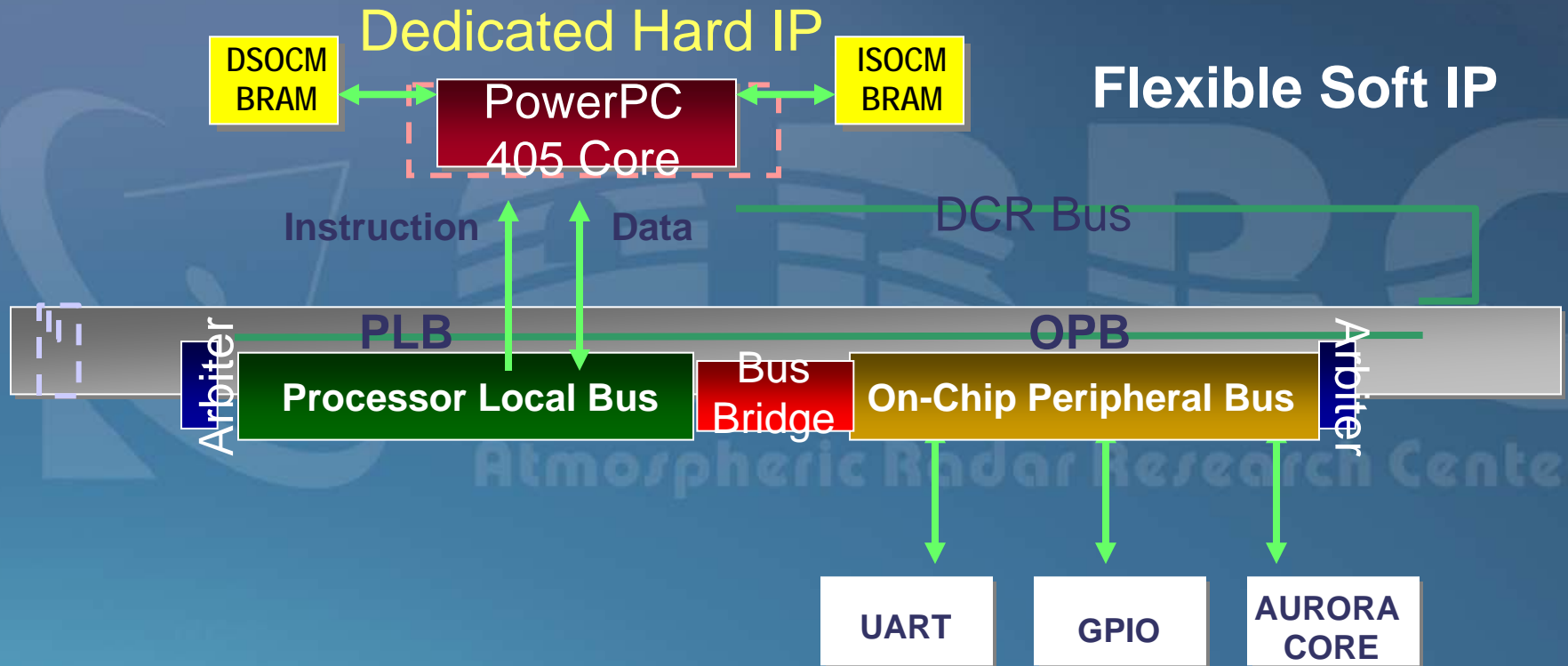
Configurable parameters :

- Streaming / framing interface .
- Simplex / full duplex data flow
- Single / multiple MGTs
- Reference clock value
- Line rate
- MGT location based on number of MGTs selected
- Reference source clock

Tools Used for Serial RIO Implementation

- Xilinx software tools (ISE, EDK, IP core generator, Impact)
- Two Xilinx Virtex-II Pro XC2VP30-7 FPGA boards
- Two Diligent FX2-BB boards
- Two Serial ATA cables
- National Instruments LabView 8.5

SERIAL CONNECTIVITY WITH AURORA IP



SERIAL CONNECTIVITY WITH AURORA IP

The screenshot shows the Xilinx Platform Studio interface for a project named 'TestApp_Peripheral'. The 'System Assembly View' is active, displaying a tree of components and their connections. The components are organized into a hierarchy, with the 'aurora_mgt' components highlighted in green. The table below provides a detailed view of the components and their connections.

Name	Bus Connection	IP Type	IP Version
ppc405_0		ppc405	2.00.c
MDCR	No Connection		
DPLB	plb		
IPLB	plb		
DSOCM	No Connection		
ISOCM	No Connection		
JTAGPPC	No Connection		
plb		plb_v34	1.02.a
SDCR	No Connection		
opb		opb_v20	1.10.c
plb2opb		plb2opb_bridge	1.01.a
SDCR	No Connection		
SPLB	plb		
MOPB	opb		
onewire_0		opb_onewire	1.00.a
RS232_Uart_1		opb_uartlite	1.00.b
SOPB	opb		
plb_bram_if_cntlr_1		plb_bram_if_cntlr	1.00.b
SPLB	plb		
PORTA	plb_bram_if_cntlr...		
aurora_mgt_0		aurora_mgt	1.00.a
SOPB	opb		
aurora_mgt_1		aurora_mgt	1.00.a
SOPB	opb		
mgt_dcm_0		mgt_dcm	1.00.a
SOPB	opb		
reset_block		proc_sys_reset	1.00.a
plb_bram_if_cntlr_1_bram		bram_block	1.00.a
PORTA	plb_bram_if_cntlr...		
PORTB	No Connection		
dcm_0		dcm_module	1.00.a

R232 INTERFACE

DATA RECEIVED

VISA resource name

COM1

baud rate

9600

data bits

8

parity

None

stop bits

1.0

timeout (ms)

10000

termination char
(0xA = '\n' = LF)

A

bytes to read 2

500

Input
buffer size

4096

End write with
termination
character?



End read on
termination
character?



STOP

```
0x0032 0x0031 0x0030 0x002F 0x002E 0x002D 0x002C 0x002B 0x002A 0x0029 0x0028 0x0027
0x0026 0x0025 0x0024 0x0023 0x0022 0x0021 0x0020 0x001F 0x001E 0x001D 0x001C 0x001B
0x001A 0x0019 0x0018 0x0017 0x0016 0x0015 0x0014 0x0013 0x0012 0x0011 0x0010 0x000F
0x000E 0x000D 0x000C 0x000B 0x000A 0x0009 0x0008 0x0007 0x0006 0x0005 0x0004 0x0003
0x0002 0x0001
```

```
0x0032 0x0031 0x0030 0x002F 0x002E 0x002D 0x002C 0x002B 0x002A 0x0029 0x0028 0x0027
0x0026 0x0025 0x0024 0x0023 0x0022 0x0021 0x0020 0x001F 0x001E 0x001D 0x001C 0x001B
0x001A 0x0019 0x0018 0x0017 0x0016 0x0015 0x0014 0x0013 0x0012 0x0011 0x0010 0x000F
0x000E 0x000D 0x000C 0x000B 0x000A 0x0009 0x0008 0x0007 0x0006 0x0005 0x0004 0x0003
0x0002 0x0001
```

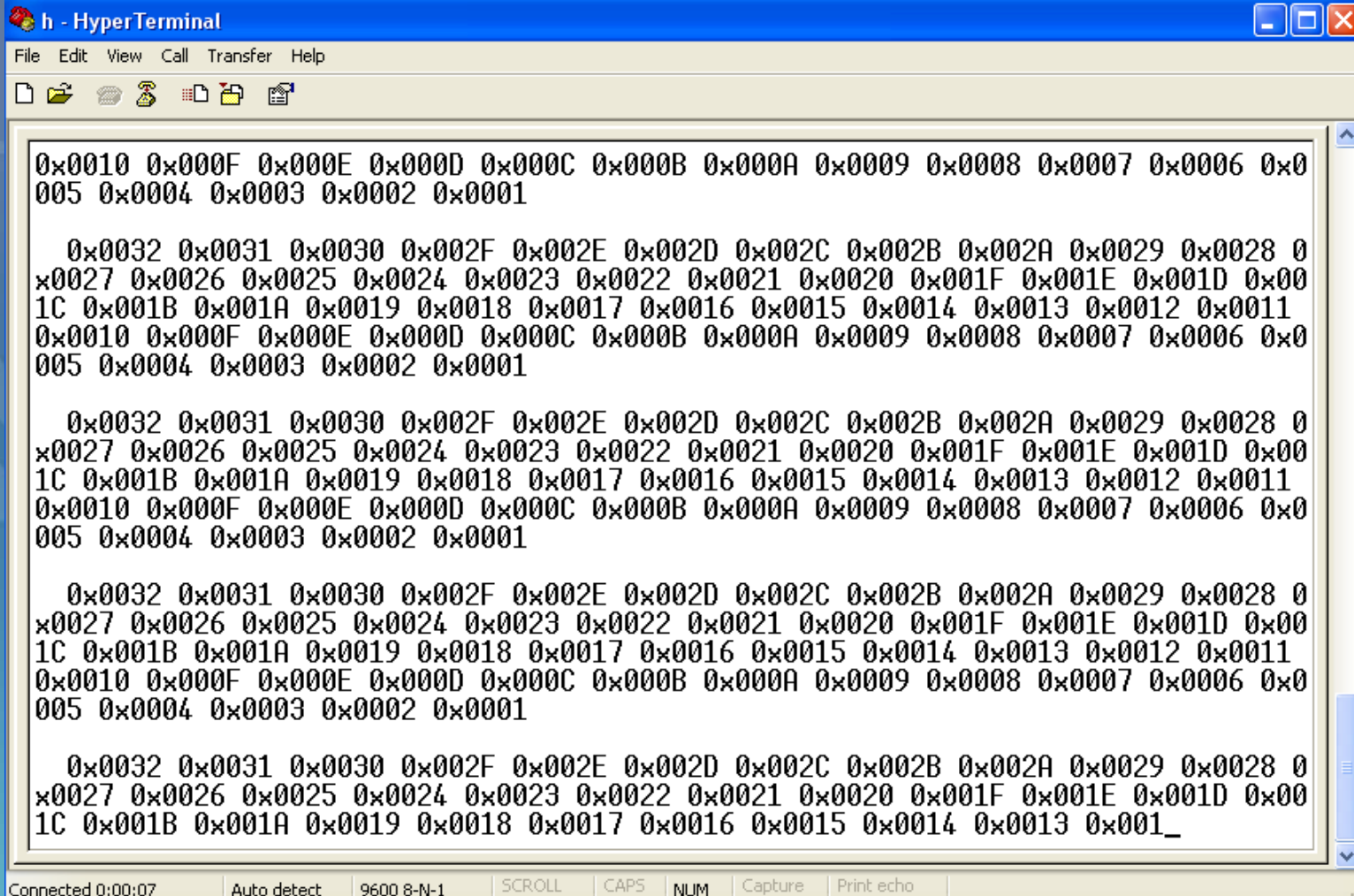
```
0x0032 0x0031 0x0030 0x002F 0x002E 0x002D 0x002C 0x002B 0x002A 0x0029 0x0028 0x0027
0x0026 0x0025 0x0024 0x0023 0x0022 0x0021 0x0020 0x001F 0x001E 0x001D 0x001C 0x001B
0x001A 0x0019 0x0018 0x0017 0x0016 0x0015 0x0014 0x0013 0x0012 0x0011 0x0010 0x000F
0x000E 0x000D 0x000C 0x000B 0x000A 0x0009 0x0008 0x0007 0x0006 0x0005 0x0004 0x0003
0x0002 0x0001
```

```
0x0032 0x0031 0x0030 0x002F 0x002E 0x002D 0x002C 0x002B 0x002A 0x0029 0x0028 0x0027
0x0026 0x0025 0x0024 0x0023 0x0022 0x0021 0x0020 0x001F 0x001E 0x001D 0x001C 0x001B
0x001A 0x0019 0x0018 0x0017 0x0016 0x0015 0x0014 0x0013 0x0012 0x0011 0x0010 0x000F
0x000E 0x000D 0x000C 0x000B 0x000A 0x0009 0x0008 0x0007 0x0006 0x0005 0x0004 0x0003
0x0002 0x0001
```

```
0x0032 0x0031 0x0030 0x002F 0x002E 0x002D 0x002C 0x002B 0x002A 0x0029 0x0028 0x0027
0x0026 0x0025 0x0024 0x0023 0x0022 0x0021 0x0020 0x001F 0x001E 0x001D 0x001C 0x001B
0x001A 0x0019 0x0018 0x0017 0x0016 0x0015 0x0014 0x0013 0x0012 0x0011 0x0010 0x000F
0x000E 0x000D 0x000C 0x000B 0x000A 0x0009 0x0008 0x0007 0x0006 0x0005 0x0004 0x0003
0x0002 0x0001
```

```
0x0032 0x0031 0x0030 0x002F 0x002E 0x002D 0x002C 0x002B 0x002A 0x0029 0x0028 0x0027
0x0026 0x0025 0x0024 0x0023 0x0022 0x0021 0x0020 0x001F 0x001E 0x001D 0x001C 0x001B
0x001A 0x0019 0x0018 0x0017 0x0016 0x0015 0x0014 0x0013 0x0012 0x0011 0x0010 0x000F
0x000E 0x000D 0x000C 0x000B 0x000A 0x0009 0x0008 0x0007 0x0006 0x0005 0x0004 0x0003
0x0002 0x0001
```

SERIAL CONNECTIVITY AURORA IP



The image shows a HyperTerminal window titled "h - HyperTerminal" with a menu bar (File, Edit, View, Call, Transfer, Help) and a toolbar. The main display area contains several lines of hexadecimal data, grouped into five distinct blocks. Each block consists of two lines of 16 hex characters each, separated by a blank line. The data appears to be a repeating sequence of values. The status bar at the bottom indicates a connection to "9600 8-N-1" with various control options like SCROLL, CAPS, NUM, Capture, and Print echo.

```
0x0010 0x000F 0x000E 0x000D 0x000C 0x000B 0x000A 0x0009 0x0008 0x0007 0x0006 0x0005 0x0004 0x0003 0x0002 0x0001

0x0032 0x0031 0x0030 0x002F 0x002E 0x002D 0x002C 0x002B 0x002A 0x0029 0x0028 0x0027 0x0026 0x0025 0x0024 0x0023 0x0022 0x0021 0x0020 0x001F 0x001E 0x001D 0x001C 0x001B 0x001A 0x0019 0x0018 0x0017 0x0016 0x0015 0x0014 0x0013 0x0012 0x0011 0x0010 0x000F 0x000E 0x000D 0x000C 0x000B 0x000A 0x0009 0x0008 0x0007 0x0006 0x0005 0x0004 0x0003 0x0002 0x0001

0x0032 0x0031 0x0030 0x002F 0x002E 0x002D 0x002C 0x002B 0x002A 0x0029 0x0028 0x0027 0x0026 0x0025 0x0024 0x0023 0x0022 0x0021 0x0020 0x001F 0x001E 0x001D 0x001C 0x001B 0x001A 0x0019 0x0018 0x0017 0x0016 0x0015 0x0014 0x0013 0x0012 0x0011 0x0010 0x000F 0x000E 0x000D 0x000C 0x000B 0x000A 0x0009 0x0008 0x0007 0x0006 0x0005 0x0004 0x0003 0x0002 0x0001

0x0032 0x0031 0x0030 0x002F 0x002E 0x002D 0x002C 0x002B 0x002A 0x0029 0x0028 0x0027 0x0026 0x0025 0x0024 0x0023 0x0022 0x0021 0x0020 0x001F 0x001E 0x001D 0x001C 0x001B 0x001A 0x0019 0x0018 0x0017 0x0016 0x0015 0x0014 0x0013 0x0012 0x0011 0x0010 0x000F 0x000E 0x000D 0x000C 0x000B 0x000A 0x0009 0x0008 0x0007 0x0006 0x0005 0x0004 0x0003 0x0002 0x0001

0x0032 0x0031 0x0030 0x002F 0x002E 0x002D 0x002C 0x002B 0x002A 0x0029 0x0028 0x0027 0x0026 0x0025 0x0024 0x0023 0x0022 0x0021 0x0020 0x001F 0x001E 0x001D 0x001C 0x001B 0x001A 0x0019 0x0018 0x0017 0x0016 0x0015 0x0014 0x0013 0x0012 0x0011 0x0010 0x000F 0x000E 0x000D 0x000C 0x000B 0x000A 0x0009 0x0008 0x0007 0x0006 0x0005 0x0004 0x0003 0x0002 0x0001_
```

Connected 0:00:07 Auto detect 9600 8-N-1 SCROLL CAPS NUM Capture Print echo

SERIAL CONNECTIVITY AURORA IP

```
h - HyperTerminal
File Edit View Call Transfer Help
[Icons]

Reading data from the board_1 :
W H A T S - U P ? -

Transmit data from Board 0 to Board_1
-----

Writing data to the board_1 :
0x0001 0x0002 0x0003 0x0004 0x0005 0x0006 0x0007 0x0008 0x0009 0x000A

Reading data from the board_1 :
0x000A 0x0009 0x0008 0x0007 0x0006 0x0005 0x0004 0x0003 0x0002 0x0001

Transmit text from Board_0 to Board_1
-----

Writing data to the Board 1 :
A U R O R A 0 0 0 -

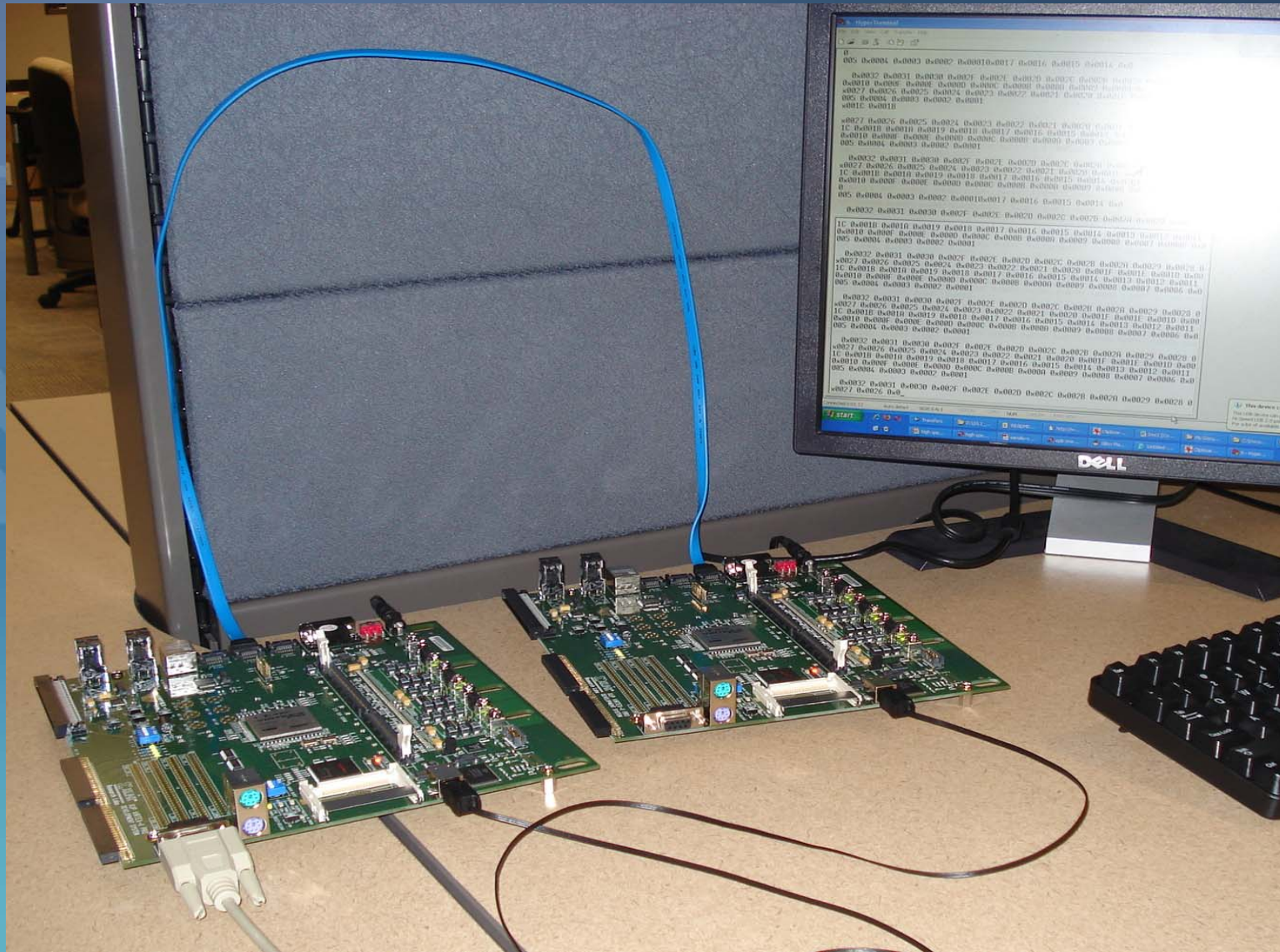
Reading data from the board_1 :
W H A T S - U P ? -

Transmit data from Board 0 to Board_1
-----

Writing data to the board_1 :
0x0001 0x0002 0x0003 0x0004 0x0005 0x0006_

Connected 0:01:04  ANSIW  9600 8-N-1  SCROLL  CAPS  NUM  Capture  Print echo
```

SERIAL CONNECTIVITY AURORA IP



ACKNOWLEDGMENT

This Study was made possible thanks to a donation of tools and IP cores from Xilinx