FPGA+CPU Architectures for Trigger Applications

Kristian Hahn – Northwestern

CMS Software/Computing R&D Meeting
10/5/15

Two questions for discussion:

– Would it be possible (and beneficial) to involve a hybrid MPSoC directly in the L1 processing path?

– How could an MPSoC accelerate HLT processing?

Outline:

– Some technical background

– Initial considerations

– A rough plan of study and some initial attempts

Essentially just thoughts on paper at this point ...

Background: Zynq MPSoC

- Xilinx / Altera bread & butter are (were) FPGAs

- 2011: Xilinx introduced Zynq-7000 family SoCs

  - Hybrid "Programmable Logic" (FPGA) + ARM CPU "Processing System"

- Eclipse-based software development kit integrated with Xilinx's newer version synthesis tools (ie: Vivado)

- Devices available today across performance ($) spectrum
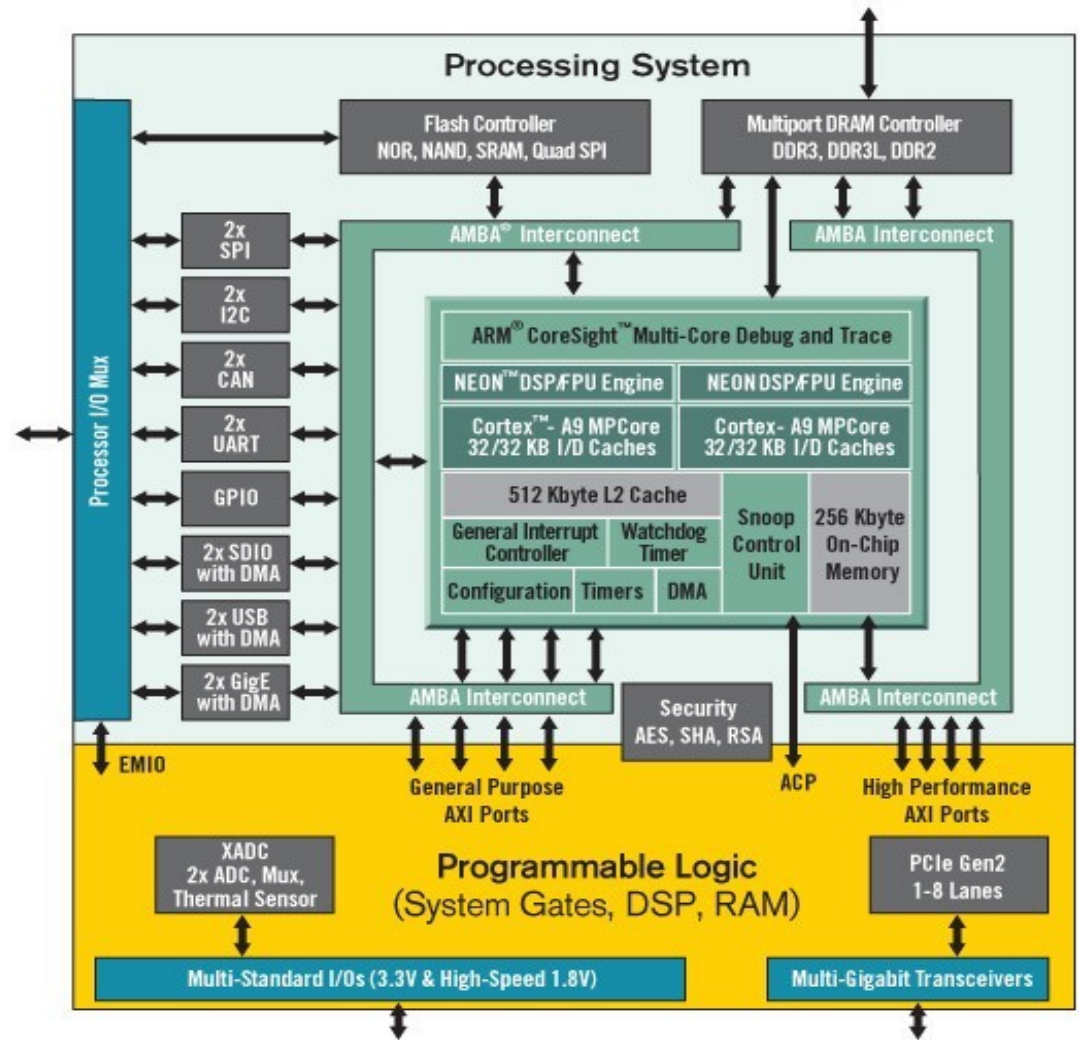
More Background: Zynq @ CMS

Zynq already in the (Phase-1) CMS trigger … UWisc. CPT7
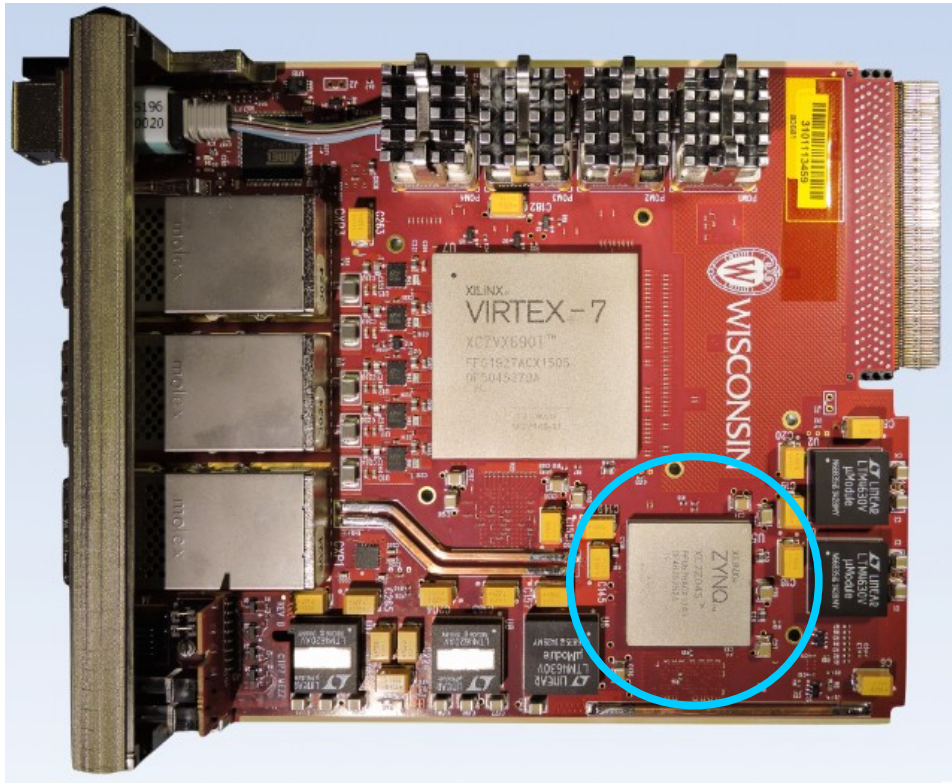
- Embedded Linux

- GbE TCP/IP slow control & monitoring

- Virtual cable / Linux driven FPGA reprogramming

- MGT monitoring (tuning?) via non-invasive, integrated eye scan

See poster from TWEPP 2015:

https://indico.cern.ch/event/357738/session/10/contribution/210/attachments/1160834/1671224/Svetek_TWEPP_Poster.pdf
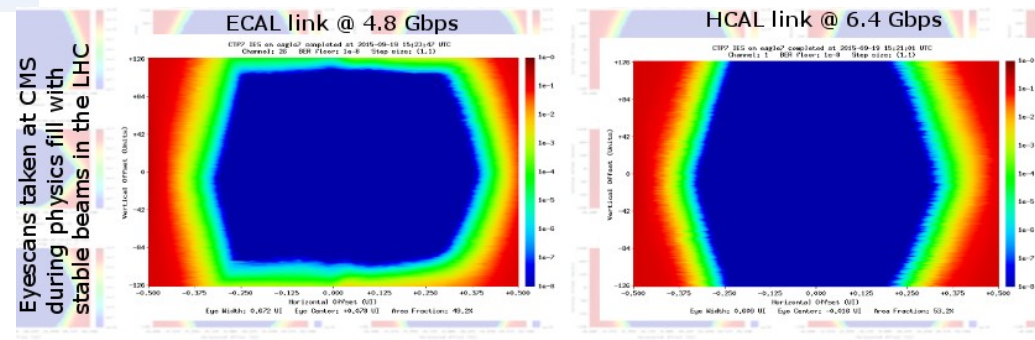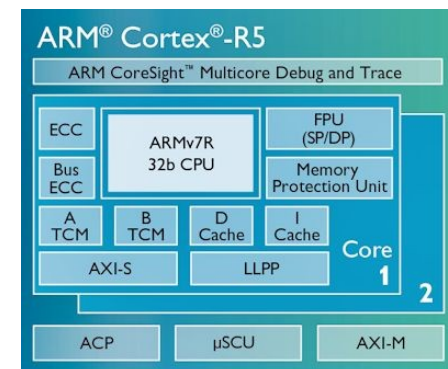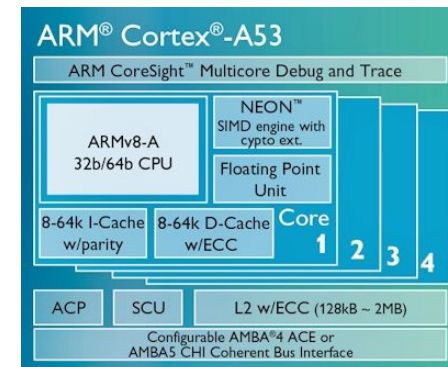
"The future is hybrid"

Xilinx Ultrascale+ Zynq

Zynq® UltraScale+™ MPSoCs

		Smarter Control and Vision					Smarter Network					
	Device Name[1]	ZU2EG	ZU3EG	ZU4EV	ZU5EV	ZU7EV	ZU6EG	ZU9EG	ZU15EG	ZU11EG	ZU17EG	ZU19EG
Application Processor Unit	Processor Core	**Quad-core** ARM® Cortex™-A53 MPCore™ up to 1.3GHz										
	Memory w/ECC	L1 Cache 32KB I / D per core, L2 Cache 1MB, on-chip Memory 256KB										
Real-Time Processor Unit	Processor Core	**Dual-core** ARM Cortex-R5 MPCore™ up to 600MHz										
	Memory w/ECC	L1 Cache 32KB I / D per core, Tightly Coupled Memory 128KB										
Graphic & Video Acceleration	Graphics Processing Unit	Mali™-400MP up to 466MHz										
	Memory	L2 Cache 64KB										
External Memory	Dynamic Memory Interface	x32/x64: DDR4, LPDDR4, DDR3, DDR3L, LPDDR3										
	Static Memory Interfaces	NAND, 2x Quad-SPI										
Connectivity	High-Speed Connectivity	PCIe® Gen2 x4, 2x USB3.0, SATA 3.0, DisplayPort, 4x Tri-mode Gigabit Ethernet										
	General Connectivity	2xUSB 2.0, 2x SD/SDIO, 2x UART, 2x CAN 2.0B, 2x I2C, 2x SPI, 4x 32b GPIO										
Integrated Block Functionality	Power Management	Full / Low / PL / Battery Power Domains										
	Security	RSA, AES, and SHA										
	AMS - System Monitor	10-bit, 1MSPS - Temperature, Voltage, and Current Monitor										
PS to PL Interface		11x 32/64/128b & 1x 32/64b AXI Ports										
Programmable Functionality	Effective LEs[2] (K)	100	150	185	245	485	450	575	715	625	890	1,100
	Logic Cells (K)	83	124	154	205	403	376	480	597	522	741	915
	CLB Flip-Flops (K)	94	141	176	234	461	429	548	682	597	847	1,045
Memory	Max. Distributed RAM (Mb)	1.2	1.8	2.8	3.8	6.2	6.9	8.8	11.3	9.1	8.0	9.8
	Total Block RAM (Mb)	5.3	7.6	4.5	5.1	11.0	25.1	32.1	26.2	21.1	28.0	34.6
	UltraRAM (Mb)	-	-	14.0	18.0	27.0	-	-	31.5	22.5	28.7	36.0
Integrated IP	DSP Slices	240	360	728	1,056	1,728	1,973	2,520	3,528	2,928	1,590	1,968
	Video Codec Unit (VCU)	-	-	1	1	1	-	-	-	-	-	-
	PCI Express® Gen 3x16 / Gen4x8	-	-	2	2	2	-	-	-	4	4	5
	150G Interlaken	-	-	-	-	-	-	-	-	2	2	4
	100G Ethernet MAC/PCS w/RS-FEC	-	-	-	-	-	-	-	-	1	2	4
	AMS - System Monitor	1	1	1	1	1	1	1	1	1	1	1
Speed Grades	Extended[2]	-1 -2L -3					-1 -2L -3					
	Industrial	-1 -1L -2					-1 -1L -2					

ARM® Cortex®-A53

ARM CoreSight™ Multicore Debug and Trace

ARMv8-A 32b/64b CPU

NEON™ SIMD engine with cypto ext.

Floating Point Unit

8-64k I-Cache w/parity — 8-64k D-Cache w/ECC — Core 1 2 3 4

ACP — SCU — L2 w/ECC (128kB ~ 2MB)

Configurable AMBA®4 ACE or AMBA5 CHI Coherent Bus Interface

ARM® Cortex®-R5

ARM CoreSight™ Multicore Debug and Trace

ECC — ARMv7R 32b CPU — FPU (SP/DP)

Bus ECC — Memory Protection Unit

A TCM — B TCM — D Cache — I Cache — Core 1 2

AXI-S — LLPP

ACP — µSCU — AXI-M

"The future is hybrid"

Altera & Intel (see also previous talk)

Would it be possible (and beneficial) to involve a hybrid MPSoC directly in the L1 processing path?

Would it be possible (and beneficial) to involve a hybrid MPSoC directly in the L1 processing path?

- Why? Physicist-accessibility …

  - Our community has much more experience w/ SW than w/ HDL

  - A SW-enhanced L1 would promote wider physicist participation in the trigger, spurring development of novel trigger algorithms

  - Related, a likely reduction of development and maintenance costs

  - Performance gains? Possibly … maybe only for special cases

Would it be possible (and beneficial) to involve a hybrid MPSoC directly in the L1 processing path?

- **Why?** Physicist-accessibility …

  - Our community has much more experience w/ SW than w/ HDL

  - A SW-enhanced L1 would promote wider physicist participation in the trigger, spurring development of novel trigger algorithms

  - Related, a likely reduction of development and maintenance costs

  - Performance gains? Possibly … maybe only for special cases

- **Why not?**

  - Latency!

    - Significant overhead incurred by a general purpose CPU, eg: cache & memory references, branch prediction, interrupts, etc.

    - Would at the very least require RTOS / bare-metal / RT core

    - Possible this erases any gains in "accessibility" ...

Could a hybrid MPSoC accelerate HLT processing?

Could a hybrid MPSoC accelerate HLT processing?

- Why?

    – The familiar potential upside: improve performance by offloading parallel operations to a co-processor

    – Dynamic reconfiguration of the programmable logic …

    – Lower power / more efficient use of silicon

Could a hybrid MPSoC accelerate HLT processing?

- **Why?**

  - The familiar potential upside: improve performance by offloading parallel operations to a co-processor

  - Dynamic reconfiguration of the programmable logic …

  - Lower power / more efficient use of silicon

- **Why not?**

  - Would make for a more complicated HLT

  - Development for heterogeneous systems not yet at the same level of maturity / ease-of-use as plain SW

Thoughts on Evaluation

1) Assess feasibility for L1 by studying performance of basic hybrid operations ← have a first attempt at this

- Latency & bandwidth for PL to PS data transfer

- From PS to PL

- Real-time performance (ie: latency distribution) of basic operations on the PS (eg: sorting)

2) Algorithm partitioning for L1 & HLT ← not considered yet

- Explore algorithms that might benefit from a split :

FPGA	CPU
– Integer operations	– Floating point operations
– Many relatively small memories	– Complex control
– Possibilities for fine-grained parallelism and deep pipelining	– Inherently sequential / iterative algorithms

Latency Studies

Have started to explore low level timing using Xilinx ZC706 evaluation kit

- – Zynq-7000 XC7Z045 FFG900-2
- – Dual ARM A9 @ 800 MHz
- – Kintex 7 equivalent PL

http://www.xilinx.com/products/boards-and-kits/ek-z7-zc706-g.html

http://www.xilinx.com/support/documentation/data_sheets/ds190-Zynq-7000-Overview.pdf

Latency Studies

- Worked through and built upon "Zynq-7000 All Programmable SoC: Concepts, Tools and Techniques (CTT)" with undergrads

  - http://www.xilinx.com/support/documentation/sw_manuals/xilinx14_6/ug873-zynq-ctt.pdf

  - Port designs from ZC702 board (slower Zynq) to ZC706

- Basic test setup

  - Toggle an input to the PL (SW5/7 … later, IO pin drive by oscillator)

  - Send this bit to PS over high priority interconnect

  - Poll on this bit in the PS

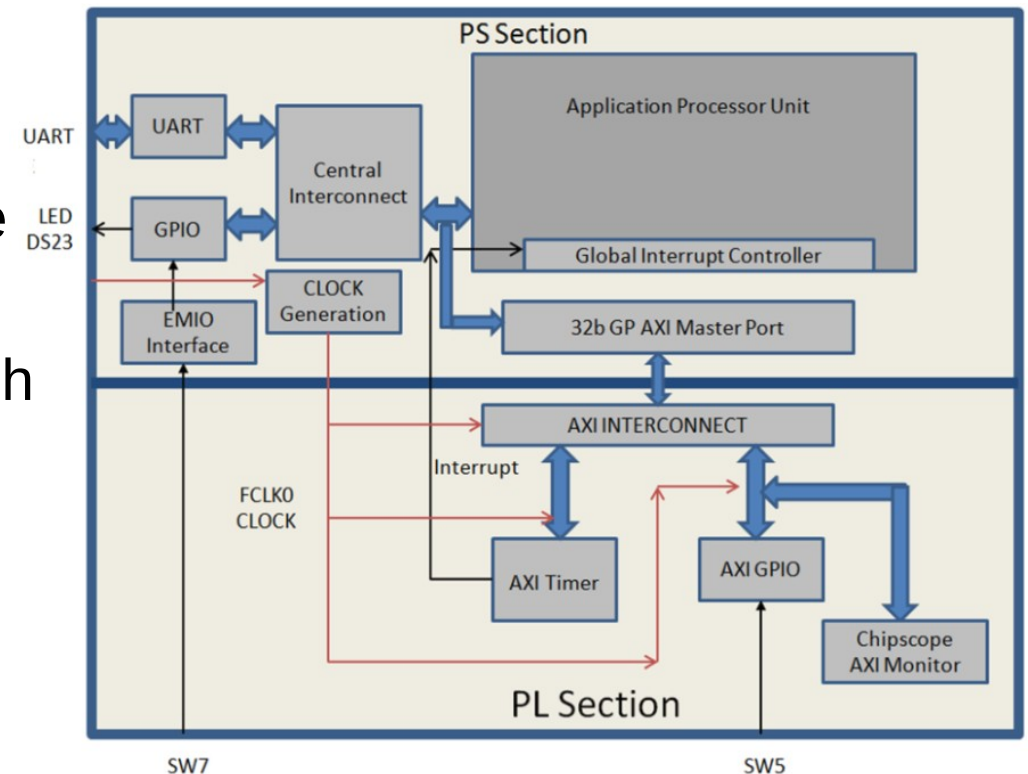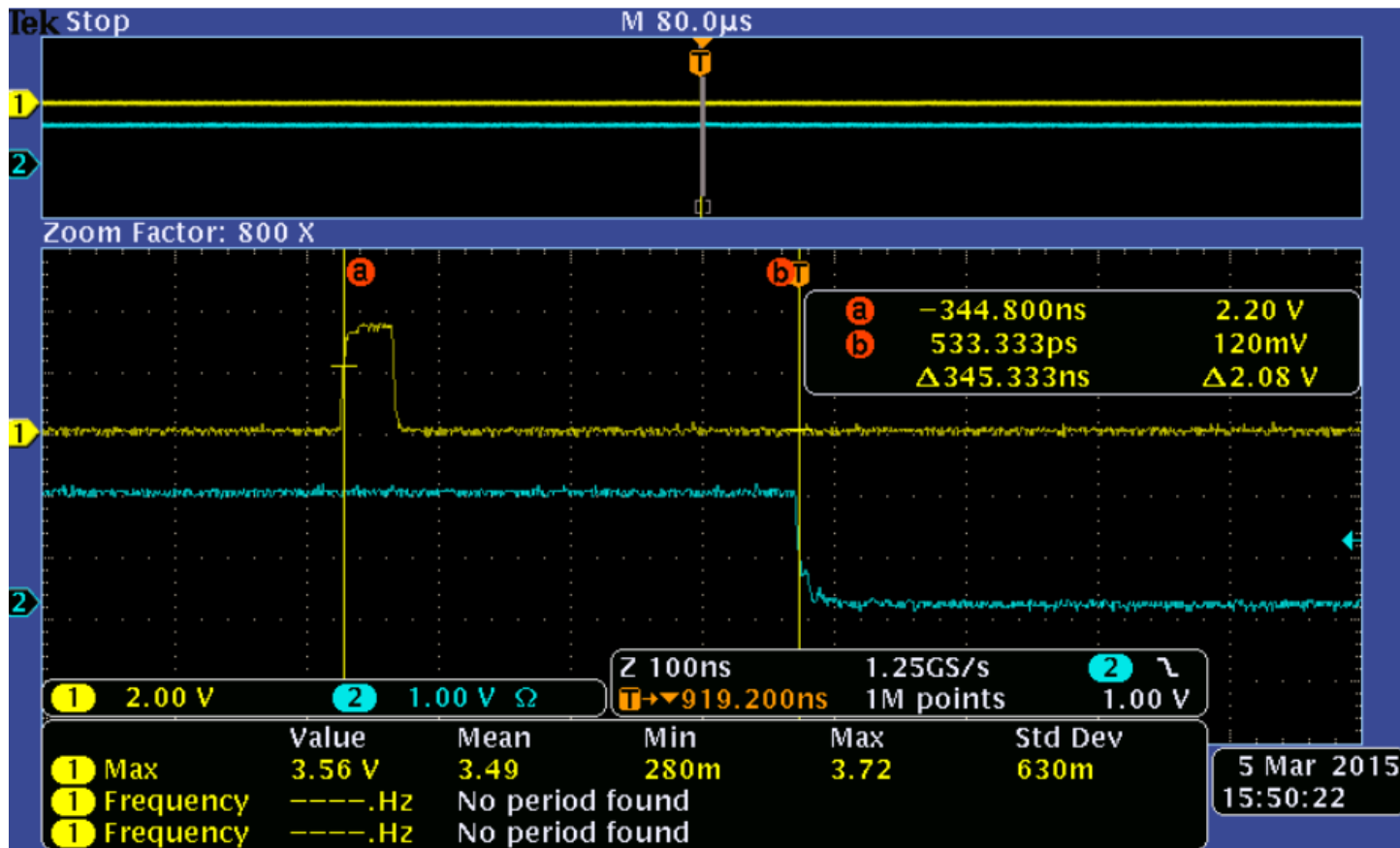 - Send back to PL over another high prio IO line

Figure 3-1: **Block Diagram**

- Measured ~350 ns RTT …
  - Seems quite large for a tightly coupled SoC
  - For comparison: have measured 80 ns for P2P communication between Xilinx FPGAs over a backplane

- Found a more closely matched example design from Xilinx (for the ZC702)

  – Latency for shared memory between the PS & PL

  – http://www.xilinx.com/support/answers/47266.html
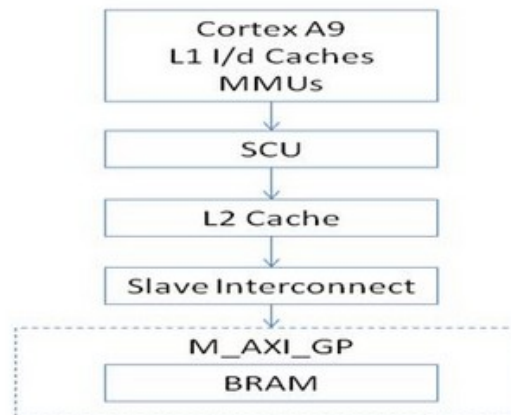
  – Advertised (unidirectional) latency more consistent with expectations for an SoC
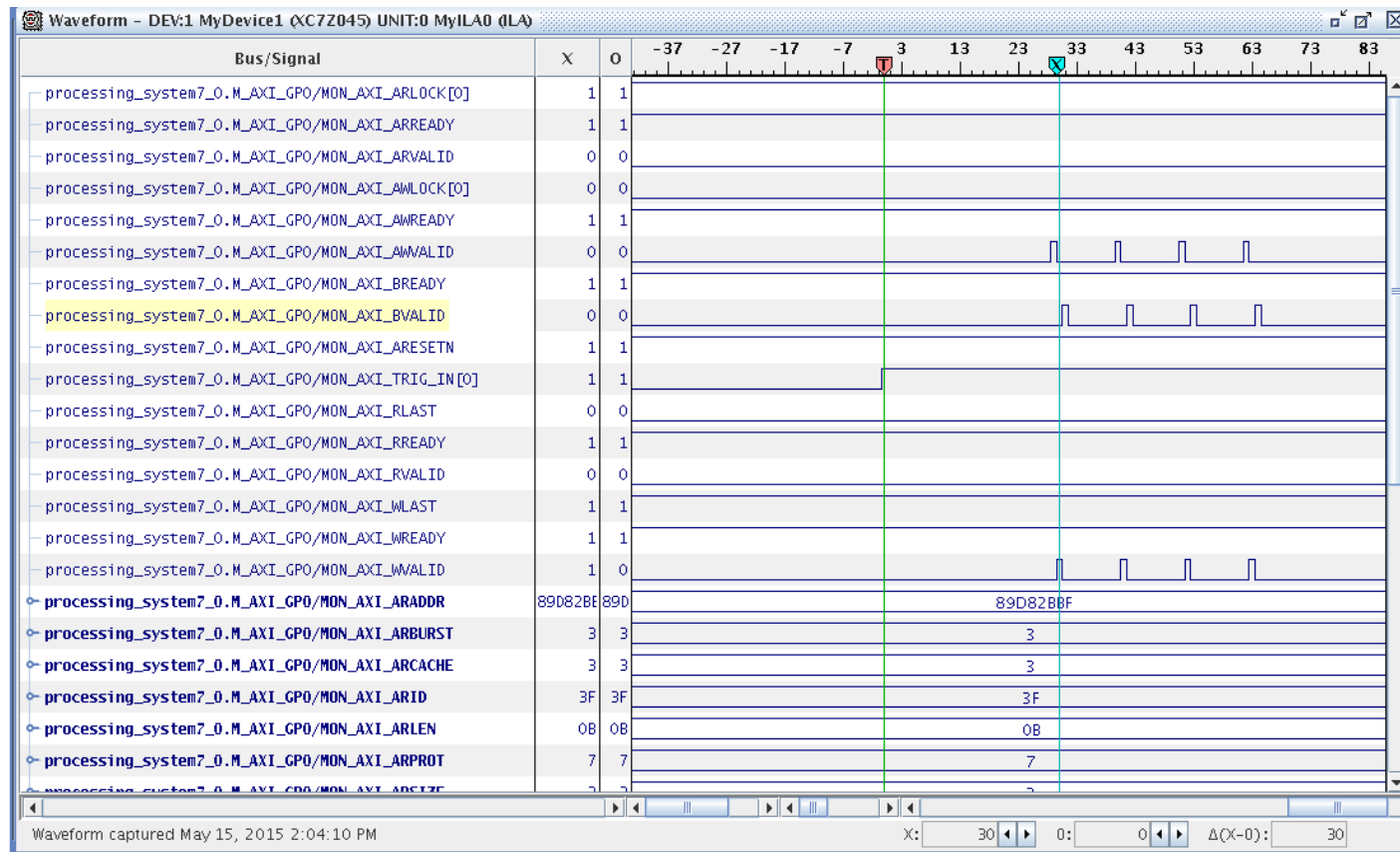
Expected Results

Strongly-ordered or Shareable device does not change the LATENCY.
Enabling the CACHE (L1 and L2) affects the LATENCY.

Type	Cache	FCLK cycles	Latency CPU cycles	Time (nS)
Strongly-ordered	Disabled	11	53	74
Strongly-ordered	Enabled	6	29	40
Shareable device	Disabled	11	53	74
Shareable device	Enabled	6	29	40

- **Obtaining much large latencies than advertised**
  - Unidirectional >200 ns ...

- Likely some simple misconfiguration …
  - Opened help ticket with Xilinx, has not converged

https://forums.xilinx.com/t5/Zynq-All-Programmable-SoC/PS-PL-latency-example-on-the-zc706/td-p/607829

Summary / Outlook

- Ruminations on the possible use of hybrid FPGA+CPU devices in the trigger
  - Tech moving to hybrid architectures … can we benefit?
  - HLT is the more likely target, maybe only a support role possible for L1
  - Should at least understand limitations for L1, as future devices might resolve these

- Have some very basic ideas about what this might entail, and as how to assess feasibility
  - No attention paid yet to algorithm classification / partitioning
  - Attempts at characterizing low level latency for hybrid communication started but not yet converged

- Happy to work others interested in these or related studies
  - Possibility for cycles from the occasional undergrad, but at this point not much more