Real-Time Development System for Image Processing Engines

Sergio Goma[a], Radu Gheorghe[b], Milivoje Aleksic[b]

a – Qualcomm, San Diego, US

b- AMD, Markham, Canada

7244-8/EI 2009

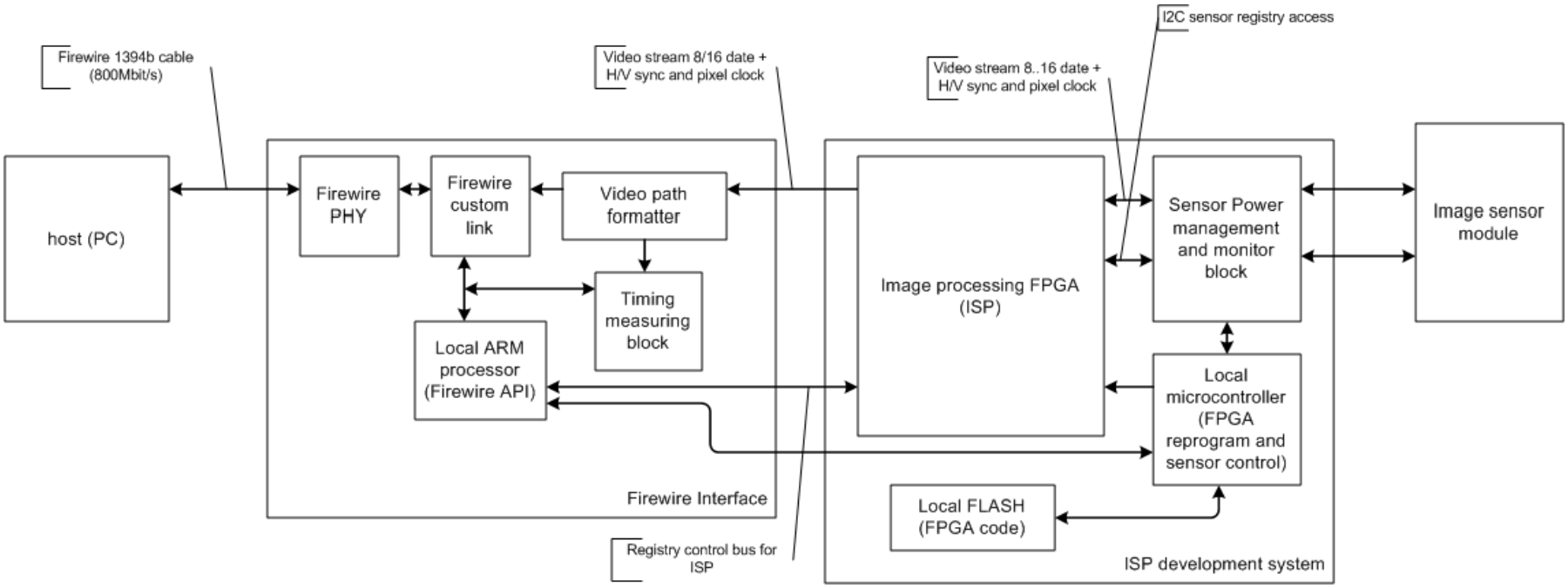
Outline

- Project motivation
- System block diagram (hardware architecture)
  - 1394 bus workarounds
  - Time domains and sequencing
- Software architecture
- Multithread timing and sequencing
- Control loop example, timing and sequencing
- Performance/Results:
  - Corner case conditions
  - Streaming bandwidth (isosynchronuos traffic)
  - Control loop latency (asynchronuos traffic)
- Conclusions

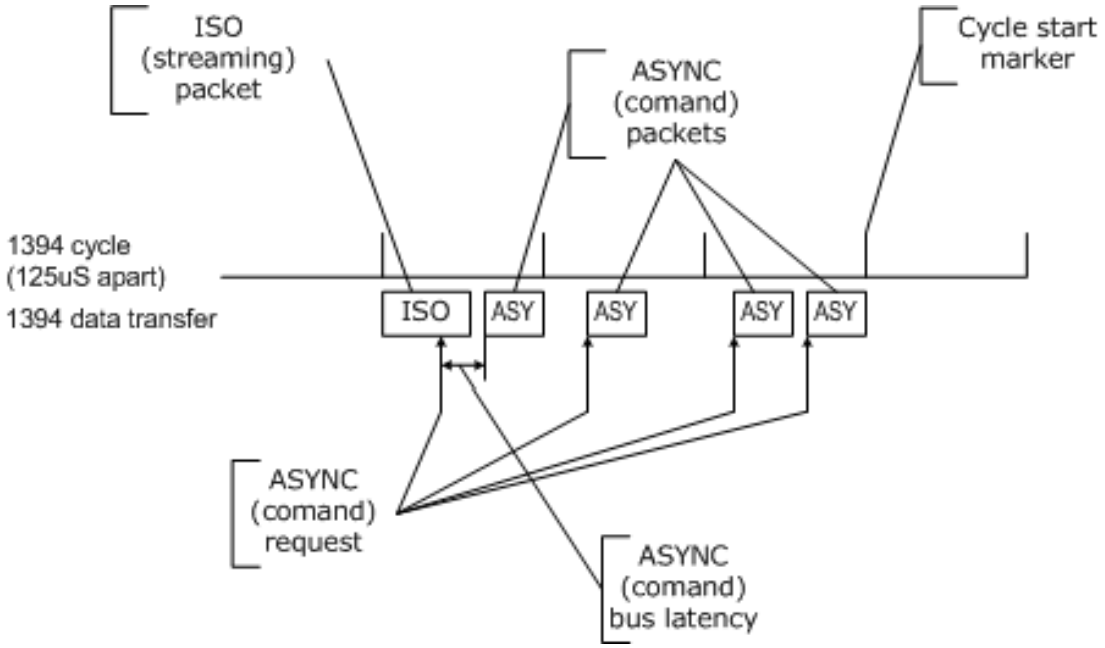
Project motivation

- To develop internal image processing algorithms and test a hardware image processing engine, real time capabilities are needed.
- Ideally, the software control loop would be developed on PC, but this requires low-latency interaction with the hw engine.
- To display a full image, high data bandwidth is required.

System block diagram

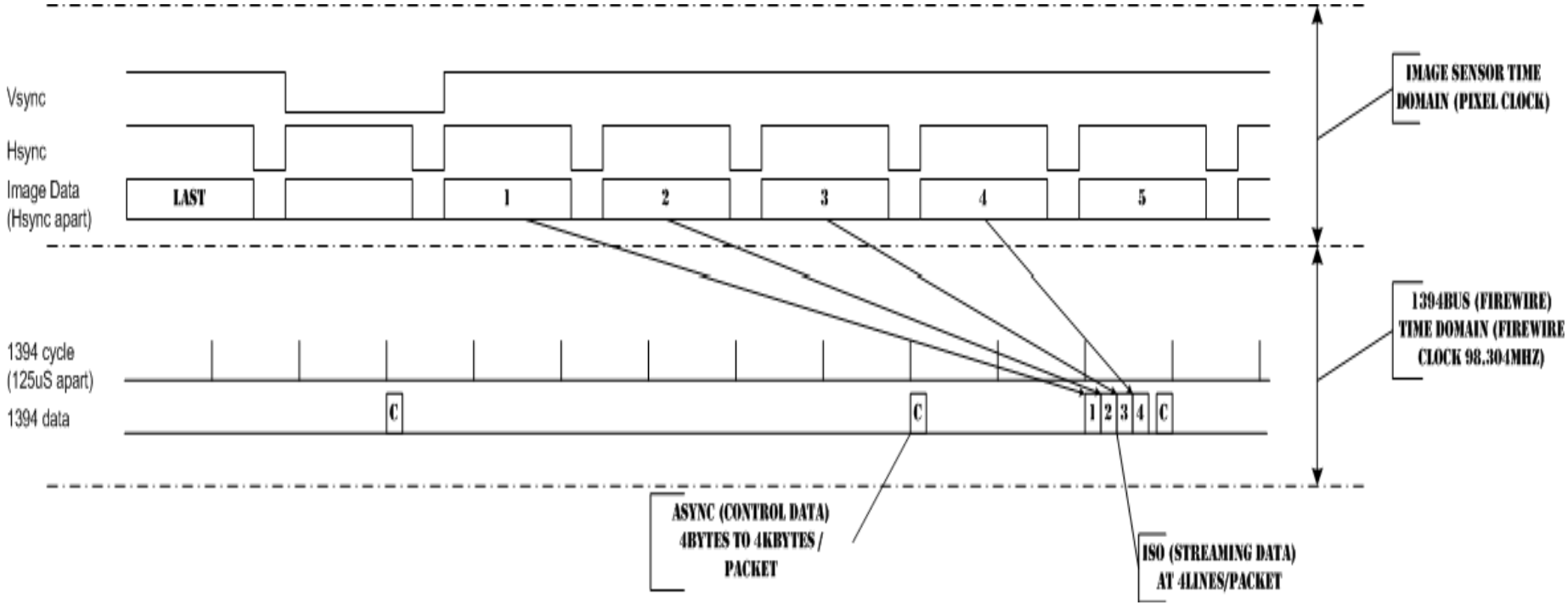


1394 async latency workaround



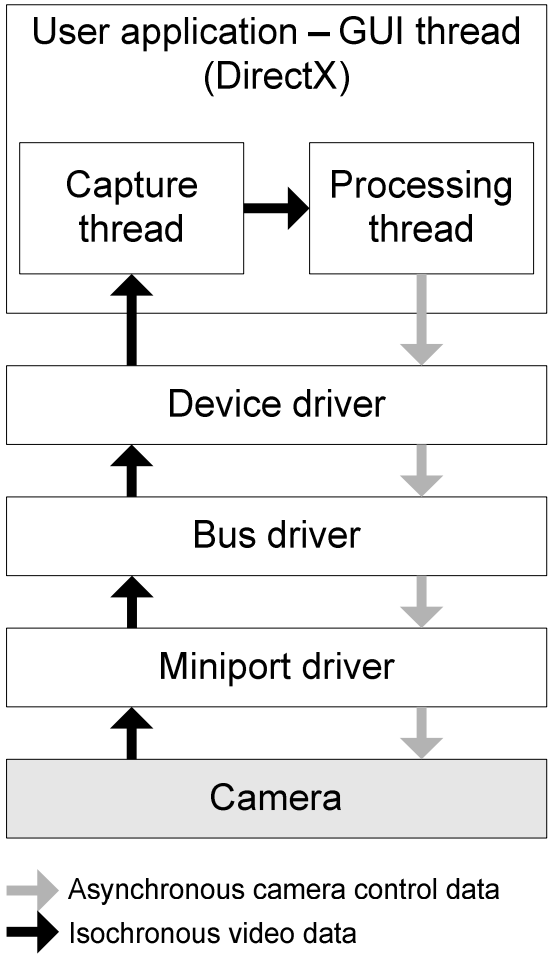
- As the IEEE1394 cycle time is 125us, and in each cycle, up to 80% of time can be used for isochronous traffic. To guarantee latency on iso, each cycle iso packet is sent first, followed by async packets – that do not have guaranteed latency. To workaround this problem, we used only one node other then the bus manager (host - PC). In this case, the worst case latency for asynchronous traffic will be less than 100us, which meets our requirements.

Time domains and sequencing

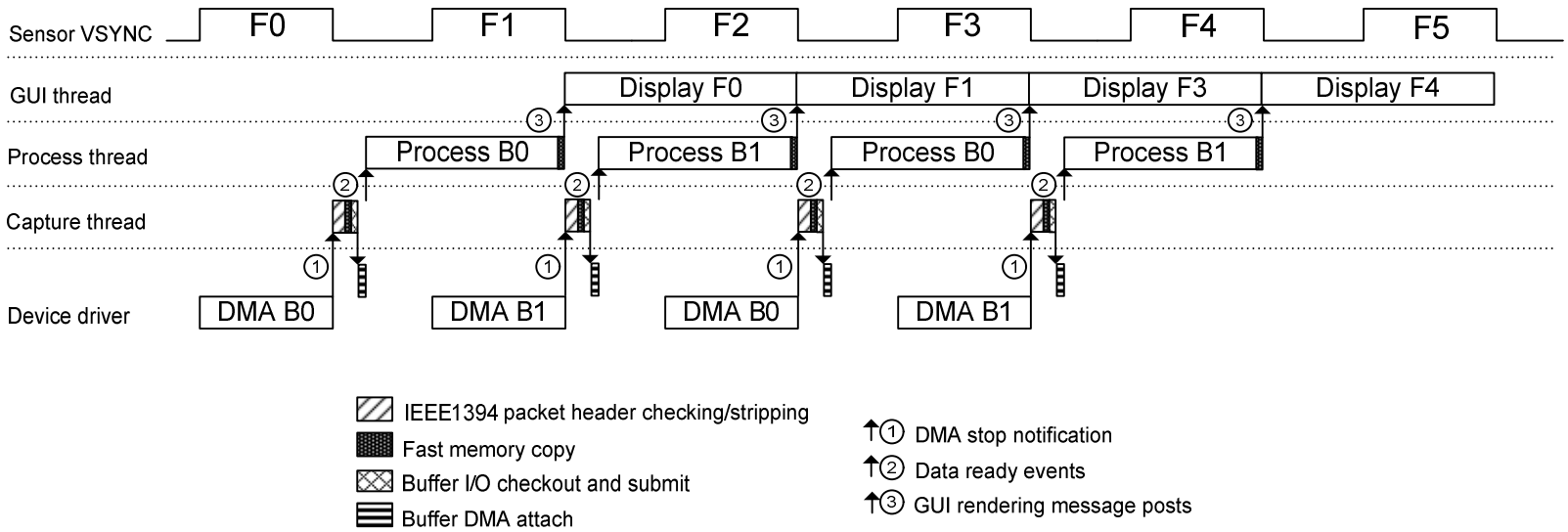


Software architecture

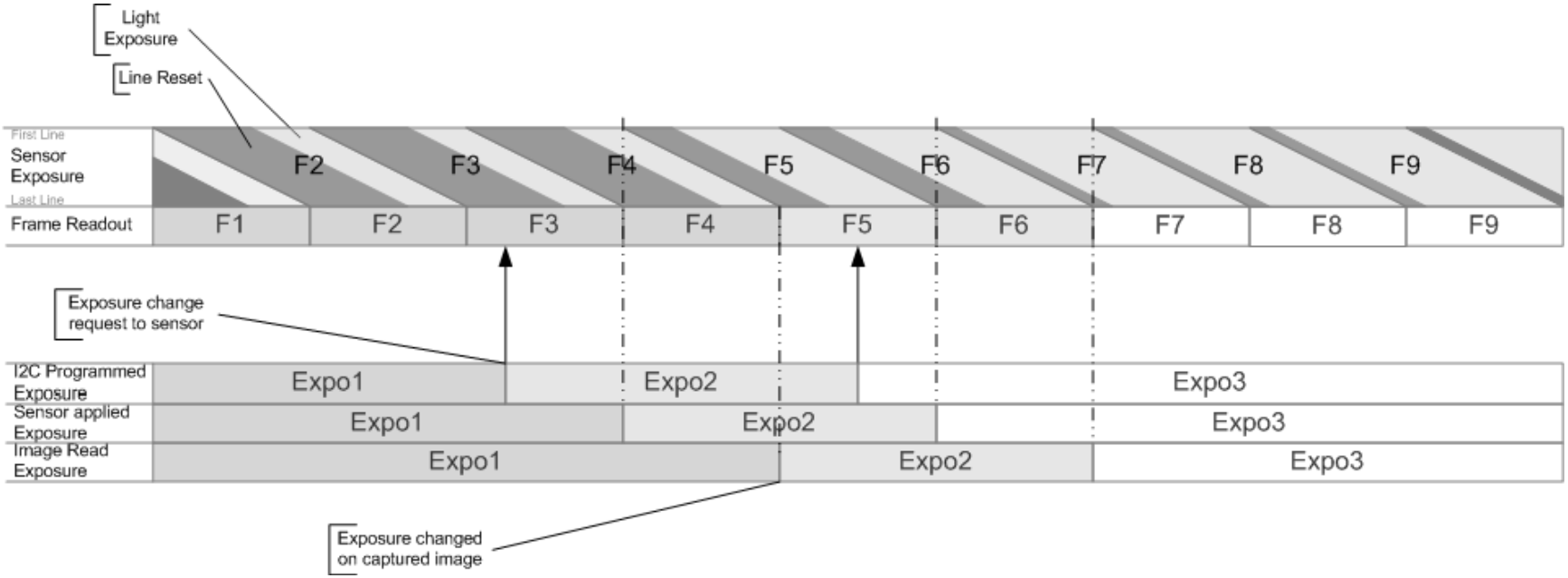
- For asynchronous camera control transfers, the driver performs overlapped I/O, allowing a calling thread not to block until the transfer is completed.
- For isochronous video data transfers, the driver shares a circular buffer I/O queue with the capture thread.
- Capture thread and processing thread are sync to camera vsync, while GUI thread is sync to display vsync. Elasticity of connection accomplished through a custom message interface.



Multithread timing and sequencing

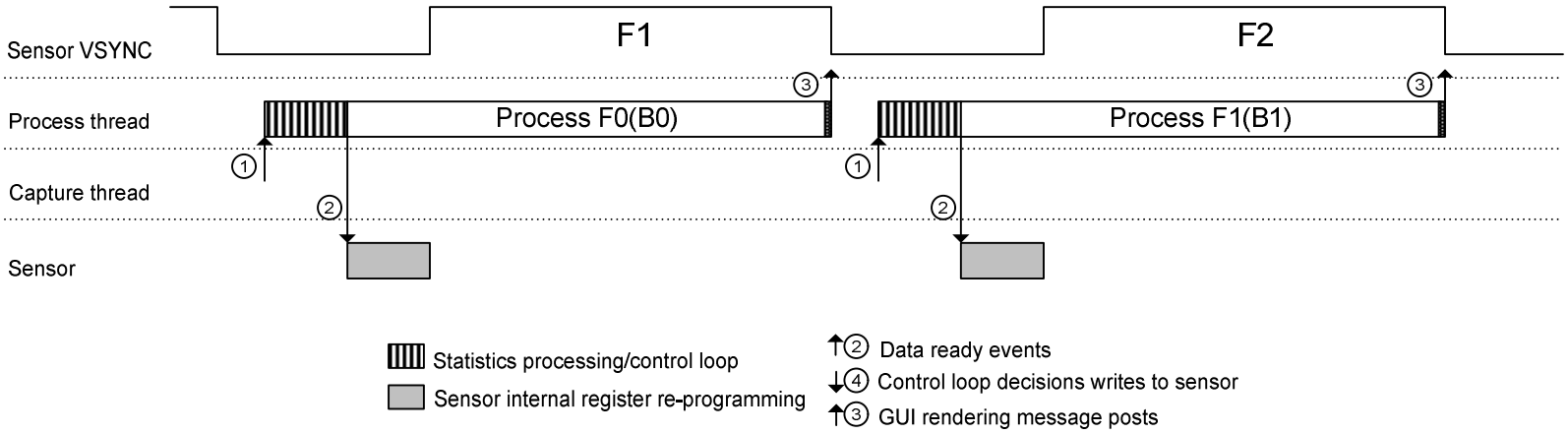


Control loop timing

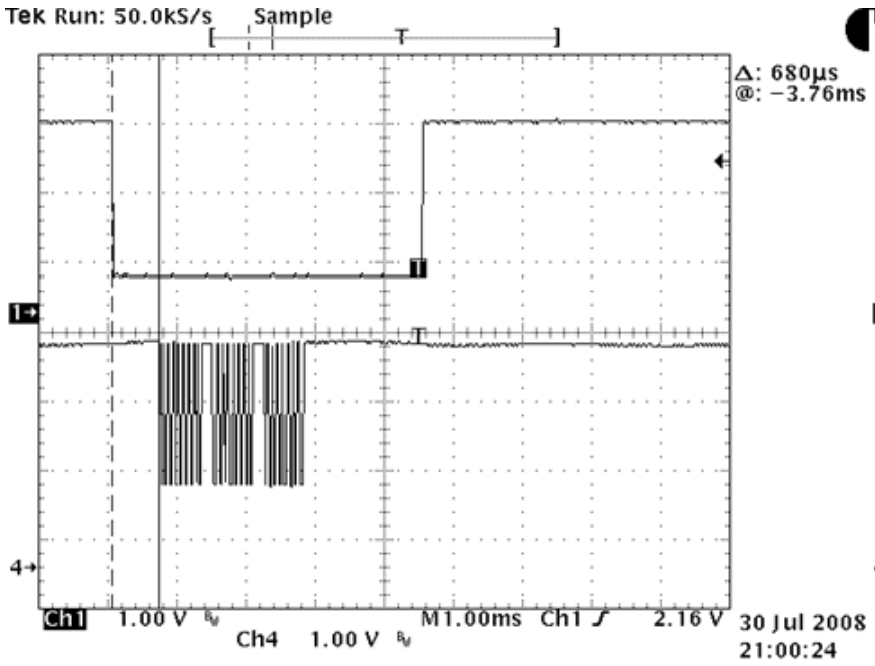


- captured image is analyzed and statistics relevant to the implemented control loop algorithm are collected. The earliest time this can be accomplished is after the full frame was captured on host side.
- the algorithm analyzes statistics data and decides an update of timing/exposure registry of the sensor (usually through I2C interface that connects the image sensor to the ISP development system

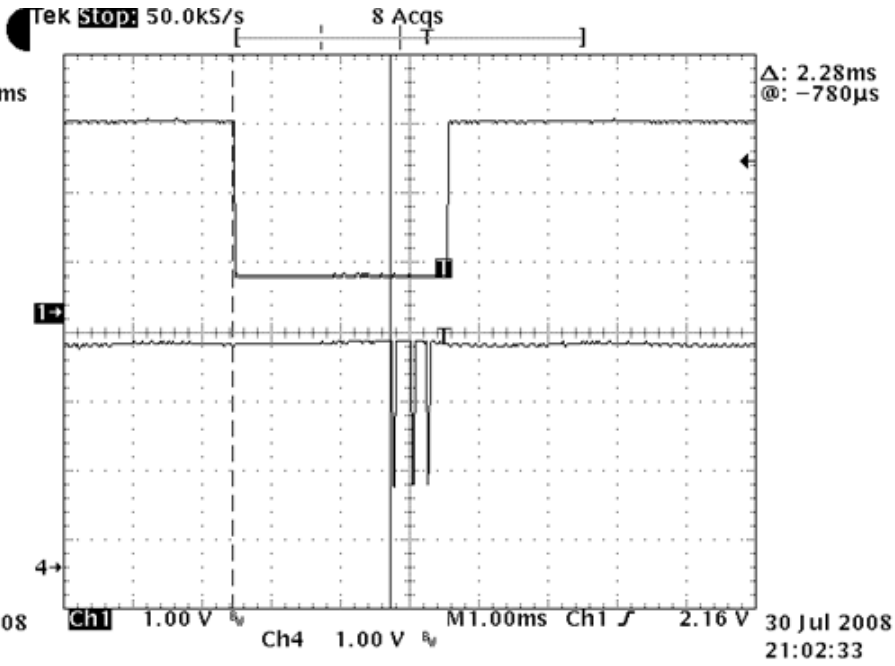
Control loop sequencing



Control loop vsync alignment



(a)

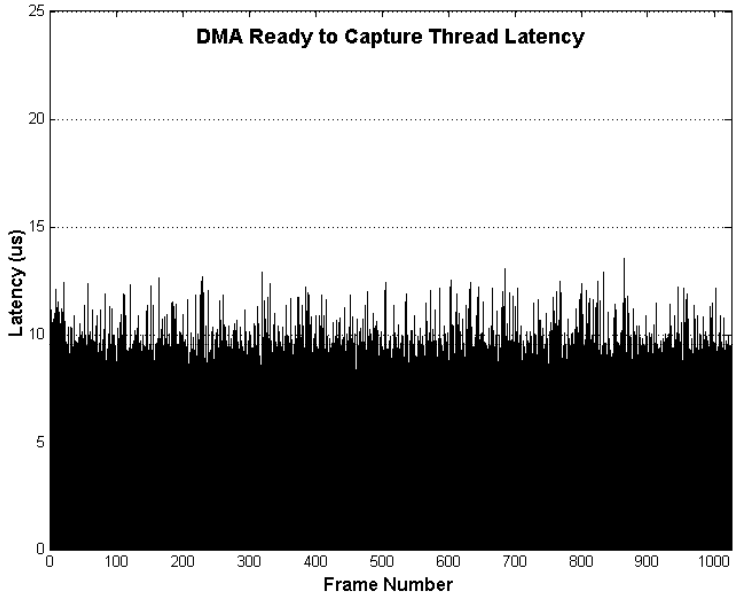


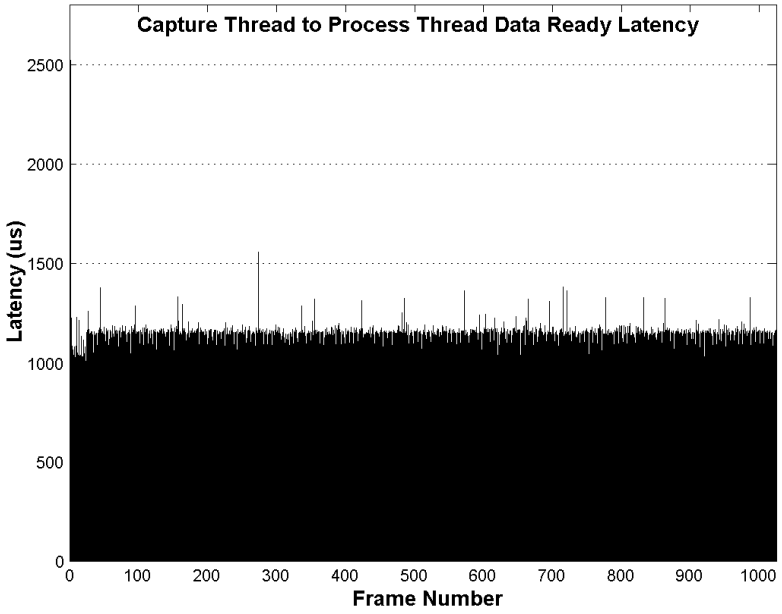
(b)

Results: corner case

- Conditions: Windows XP on a P4 3.2GHz PC with 1Gbyte of RAM
- Maximum theoretical isosynchronuos bandwidth achieved (8192bytes/cycle) or 62.5Mbytes/s (500mbps)
  - Pixel clock = 98.304MHz (1394 bus clock)
  - Horizontal active data = 1024 pixels
  - Horizontal line = 1536 clocks or 15.63us
  - Vertical active data = 1024 lines
  - Frame time = 1028 lines or 16.06ms, which yields to 62.26 frames/second
  - Pixel size = 8bits/pixel
- To this condition we added asynchronuos camera traffic up to 2048bytes/packet, achieving total bandwidth of 78.125Mbytes/s (625mbps data or **781mbps** on the cable)

Results: latency iso

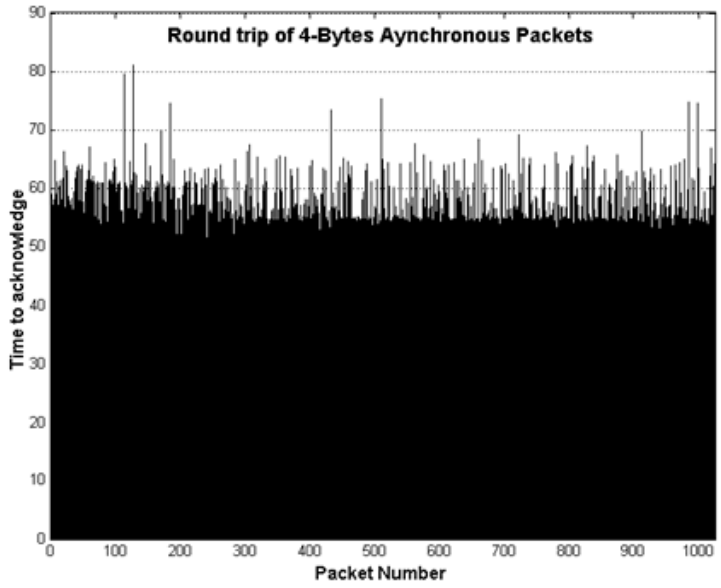




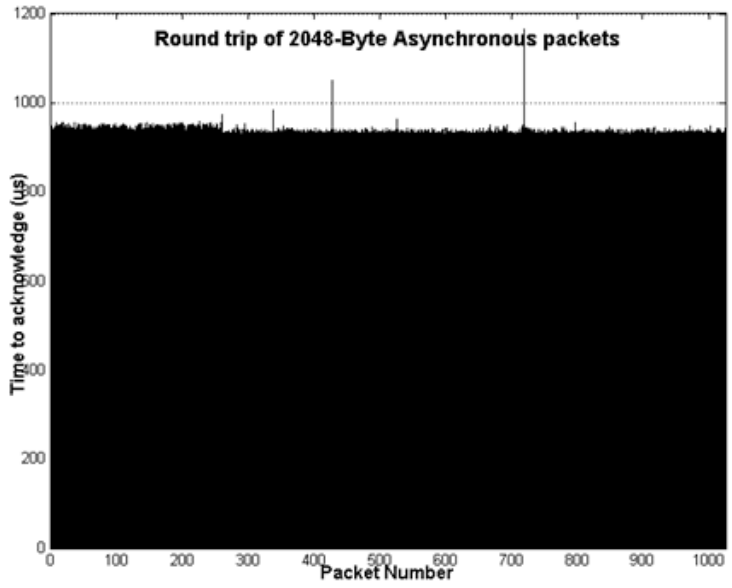
-Left: DMA ready to capture thread;

-Right: capture thread from point measured above to process thread awakening (includes IEEE1394 packet header check/strip and memory copy)

Results: latency async



(a)



(b)

-Left: 4 byte packet ;

-Right: 2048 bytes packet; in both cases data is copied to the host (ARM) before an acknowledge is generated;

Results: latency

Streaming latencies

Statistic	DMA to Capture (us)	Capture to Process (us)
Average	10.205	1154.792
Minimum	8.390	1011.545
Maximum	13.522	2780.255
Standard Deviation	0.900	77.974

Control loop latency (round trip total a control command is send from the process thread, is received over the IEEE1394 bus, the local processor (ARM) reads the packet and writes the result to the image processing FPGA and after that a acknowledge is sent back)

Statistic	2048 bytes (us)	4 bytes (us)
Average	940.421	58.711
Minimum	1164.783	51.560
Maximum	930.139	81.696
Standard Deviation	11.278	4.223

Conclusions

- A real-time image processing development system was presented, and using a non real-time operating system (WindowsXP) consistent performances satisfying real-time requirements were achieved. Plus:
  - Control loop latency in us range for short commands.
  - Streaming bandwidth of up to theoretical limit of the bus.
  - Added additional traffic up to the total limit of the bus.