Embedded Systems Capabilities

NLC Controls Architecture

JJRussell, Stanford Linear Accelerator Center

JJRussell
russells@slac.stanford.edu
(650) 926-2583

Motivation

What's the reason for this talk

- SLD experience, in particular with

  VXD2, a 120Mpixel device,

  VXD3, a 360Mpixel device

  What allowed us to go from a 60 -> 16 board system?
- Experience with BaBar DAQ, another generation forward.
- How does this apply to NLC, i.e. gazing into the future?

JJRussell
russells@slac.stanford.edu
(650) 926-2583

The SLD Experience

VXD2

- System was architected around 1988, built in 1990
- Implemented as a standard FASTBUS board
- One 20MHz 68020 along with 8 semi-custom ASICs to filter the incoming data
- Total of 60 boards to service the 120 Mpixel detector
- Cost of the board was around $3K, not including the support boards

VXD3

- System architected around 1994-95, built in 1995
- Implemented as a standard FASTBUS board
- One 33/66MHz 68040 along with 6 standard Xilinx FPGAs
- Total of 16 boards to service the 360 Mpixel detector
- Cost of the boards was around $2K

Cost, roughly $200K => $40K for 3 times the channel count.

- Perhaps a better way to think of this is that it would have cost $600K to instrument VXD3 if implemented in the same way.
- This is the method we have been using to do NLC estimates.

JJRussell
russells@slac.stanford.edu
(650) 926-2583

What made this possible?

- Faster CPU
- More efficient CPU
- Bigger, cheapier memories
- More efficient memory access
- Use of cheaper, more highly integrated parts, i.e. not as much glue logic, e.g. Xilinx's instead of custom ASICs
- Better understanding of the problem.

  Don't underestimate this last one.

JJRussell
russells@slac.stanford.edu
(650) 926-2583

CPU Trends, Past to Present

CPUs

- Clock speeds have moved from 16->50->500MHz
- More registers, generally speaking 16 -> 32 registers
- Better compilers
- Instruction execution is roughly 1 cycle per integer operation
- Advent of superscalar architectures capable of retiring multiple instructions per cycle
- Vector instructions, MMX on Pentimums, Alti-Vec on Power PCs (future)

Performance numbers

	SpecInt95	SpecFp95	MIPs
68020 @ 33MHz			5
68040 @ 40MHz			44
68060 @ 75MHz			110
PowerPC (750) @ 400 MHz:	18.8	12.2	720
Pentimum III @ 450 MHz	18.6	13.6	
Alpha (21164) @ 500 MHz	15.4	21.1	

JJRussell
russells@slac.stanford.edu
(650) 926-2583

BaBar Experience

Power PC 604 Overview

- Available in speeds up to 500MHz

  BaBar is using a 300MHz version
- Very useful integer instruction set, particularly nice bit manipulation instructions which are great for DAQ operations.
- Has 7 execution units

  2 integer

  1 Multiple

  1 Load/Store

  1 Condition Unit

  1 Branch Unit

  1 Floating Point Unit
- Capable of retiring 4 instructions per cycle
- Branch prediction and speculative execution

Observations

- Easy to keep the two integer units busy
- 3 cycle multiple is very useful
- Still useful to avoid branches

JJRussell
russells@slac.stanford.edu
(650) 926-2583

Power PC 604 - Memory

Basic Features

- Harvard architecture, i.e. separate instruction/data cache and busses
- Cache line size of 32 bytes
- L1 caches sizes of 16K for each instruction and data, with better associativity.
- Provisions for large L2 caches, but this is too board dependent to draw any general conclusions about. Not real important in DAQ applications.
- Posted writes, prefetchs, copy-back or write-through caching. This is important. Bus-snoopers allow copy-back, the fastest caching method, to be used simply.

Why is this so important?

- While CPU speeds are at 2nsec...
- ...bulk memory speeds are still in the 50-60 nsec range, not much better than the 70-80 nsecs SLD used on its original boards, circa 1990.

  This is a bit of an overstatement, since memory access (efficiency) has improved; burst modes, etc.
- However, the basic rule for RISC processors is,

  Stay away from memory

JJRussell
russells@slac.stanford.edu
(650) 926-2583

Vector Additions

Both the Pentimum (MMX) and the PowerPC (Alti-Vec) have added or will be adding special vector instructions.

- Both deal with a maximum vector length of 128 bits that can be broken into 16 bytes, 8 shorts, 4 longs or (possibly 2 doubles).
- The PowerPC seems the better thought-out of the two.

  Pentimum has a context switching time of 50 cycles

  PowerPC has a full complement of 32 128 bit registers vs. 8 for the Pentimum which are only 64 bits for integer operations.
- Improvements in DSP type operations (i.e. multiple and accumulate) are in the 4-12 range.
- If BaBar's CPUs had this capability, one could significantly reduce the number of boards in the system.

  Almost all the CPU time is in what's called feature extraction, which main times consists of manipulating an array of about 32 1 byte or 2 byte quantities.
- These instructions should be perfect for RF waveform processing and to implement the calculational phase of feedback loops.

JJRussell
russells@slac.stanford.edu
(650) 926-2583

Interrupt Latency

This is harder to quantify and, is, to some extent overrated, in importance.

- 68K interrupt latency was in the 2-10 usec range, depending on your exact definition.
- PowerPC interrupt latency is in the 4 usec range.

  If one is dealing with many interrupts, as might happen in a packetized serial network, this could be important.
- Part of the failure to scale comes from:

  Larger context to save

  More primitive hardware interrupt model, i.e. usually only 1 hardware level interrupt. Dispatching is then done in software.

  VxWorks has chosen to emulate the old 68K interrupt structure, further bogging down the process.
- The same general arguments apply to task context switching times.

  This is further complicated by the use of non-reentrant code in many of the UNIX style libraries employed by VxWorks. This means that local context must be copied at each task switch.

  Typically the system libraries are optimized to mitigate this by bundling all the non-reentrant context under a common pointer.

  The penalty for this approach is modularity and accesses that may require an additional level of indirection.

JJRussell
russells@slac.stanford.edu
(650) 926-2583

Busses and Interconnects

Private memory busses

- Usual bandwidth available in these memory subsystems is in the 400-500 Mbyte/sec range. This is an important feature of a DAQ board. One of its primary functions is to move data.

Auxiliary Busses

- Almost always this means some variation of a PCI bus.
- The PCI is an economical 32 bit, 33 MHz bus which can be used as an auxiliary bus or as the main system bus.
- Current through-put is a maximum of 133Mbytes/sec.
- Because it can be used both as a system bus or as an auxiliary bus, one can have multiple CPUs per PCI bus or multiple PCI busses per board.
- This last point can be important when trying to balance IO input costs with CPU needs. Multiple CPUs per board may be the solution.
- Its widespread usage makes it easy and cheap to interface to other standard IO, like Ethernet and Firewire.

JJRussell
russells@slac.stanford.edu
(650) 926-2583

External Interconnects/Busses

- This has been beaten about in this forum but include

  ATM

  Ethernet

  Firewire

  1355

- While ATM and Ethernet can span large distances, the last two are meant to be used locally, i.e. 10-50 meters.
- With Firewire and 1355, what one gains is not so much bandwidth, but reduced latency.
- Both behave more like a bus, with IEEE 1355 capable of supporting a large number of nodes.
- In 1355

  The routers are worm-hole routers, i.e. more like cross-bars, not store and forward. This means much reduced latency in going through the router.

  In a CERN test, Ethernet latencies were measured from a minimum of 6 usecs to a maximum of 150 usecs.

  There is hardware flow control, adding a degree of determinancy.

JJRussell
russells@slac.stanford.edu
(650) 926-2583

Ethernet, TCP/IP

No discussion would be complete without considering the effects of TCP/IP on performance.

- It is an expensive way to move bytes.

  BaBar DAQ is now moving to using the VME backplane instead of TCP/IP wherever possible.

- It takes almost 30% of 300MHz PowerPC 604 to keep a 100Mbit Ethernet link full.

  This is a highly environment dependent number. Solaris seems better.

  Because of this, ATLAS may use Ethernet to implement a level 2 trigger, but likely will not use TCP/IP.

- The model of transmitting and receiving on embedded systems is through a dedicated ‘network task’.

  On VxWorks, every pended read operation is going to cost at least two task switches.

  It’s not clear whether this is true of write operations.

  This means there is an overhead which will translate directly to a latency.

- One of my favorite ways to evaluate the cost of an operation is to consider the number of instructions/cycles per byte.

  At 30% of 300 MHz to keep 10Mbytes/sec moving this is

  100MHz/10Mbytes/sec => 10 instructions/byte

  or perhaps more relevant => 40 instructions/32bits

JJRussell
russells@slac.stanford.edu
(650) 926-2583

What Got Lost on the Way to Nirvana

Mainly, what got lost happened in the translation of bus protocols to packet switched networks and serial type busses.

- Deterministic response time.

  QOS may help, but it will never be as effective as priorities enforced at the hardware level.

- Multicast functionality. In SLD we used multicast for 3 distinct and sometimes overlapping reasons:

  1. Ability to address a ‘logical’ collection of modules

  2. Efficiency

  3. Synchronization, i.e. all members receive the message within a time characteristic of the bus speed.

- In Ethernet, TPC/IP what remains is:

  1. Multicast groups provide the ability to address a logical collection, so this still exists, but...

  2. The efficiency of multicasts is impacted in two ways

  i. It is dependent on the interface chip. Most chips can only filter perfectly for a small number of addresses (16). After that they rely on a hashing algorithm (typically modulo 512). This means that some unwanted multicasts may leak into the CPU.

  ii. The manner in which the switch implements the broadcast function. How does it fan the packets out?

  3. Synchronization functionality is completely lost.

JJRussell
russells@slac.stanford.edu
(650) 926-2583

- The absence of a real read.

  Socket reads and the like, which are common on serial networks, are not real hardware reads. Any needed data must be solicited by requesting it from a source. This increases latency by an unknowable amount since one depends on another CPU to process the request and ship back the response.

  The programming model is a push model, never a pull.

  Pipelining techniques can help, but is not always possible to do.

JJRussell
russells@slac.stanford.edu
(650) 926-2583

But some of what got lost happened within the CPU

- Have already discussed interrupt problems.
- CPU synchronization primitives are more primitive

  These almost always translate into some sort of memory barrier, which, in practice, behaves like a semaphore.

  As such, this method can be more error prone, leaving dangling locks.

  It is almost impossible for a user to implement these correctly. The RTOS must provide them, which is probably good because of the previous reason, except, semaphores, which are invariably provided by the RTOS, have exactly the same problem.

  In contrast, SLD implemented interlocked queues and interlocked bit lists etc, both locally and over the bus, without any help from the OS.
- RTOSs seem sloppier. The convenience of standard libraries (read UNIX) has imported a lot of code whose style is not fully compatible with an asynchronous, real-time environment.

  Unknown and uncontrollable latencies

  Poor memory strategies resulting in memory fragmentation problems.

  Unknown stack requirements

  Just plain bad interfaces

  Inadequate error reporting/handling
- While none of these are killers, many times the only practical workaround is to over-capacity the board and bulkier code.

JJRussell
russells@slac.stanford.edu
(650) 926-2583

Future Improvements

This is just a projection of past trends with a little reality thrown in about what's already in the pipe.

- CPUs speeds will continue to improve, certainly 1GHz is within sight, although there are warnings that things are getting more difficult.
- 64 bit processors for the embedded environment.
- Floating point performance will start to catch up with integer performance.
- Memory sizes will continue to expand, likely to the point that it will no longer be an issue in embedded systems.
- Don't look for big gains in bulk memory speeds.
- Memory management/protection will be a part of RTOSs.
- Internal L1 cache sizes are likely to continue to grow from the current 32Kbytes.

  Some worry here about the radiation hardness.

  The 603e's used by the Iridium project have had to turn the caches off.

  For the DC feature extraction code in BaBar, turning the caches off decreased performance by a factor of 20.
- PCI bus improvements are a doubling of the data width to 64bits and a doubling of the bus speed to 266MHz.

  This may not be without penalty, the rumor is that the number of slots that can be supported at the double width/speed may be limited.

JJRussell
russells@slac.stanford.edu
(650) 926-2583

How Does This Translate in Real Life

As an example take the often cited example that EPICs is capable of monitoring 2000 channels a board (167 vintage).

- Let's take this to be 2000 channels at 120 Hz.
- Now compute the number of instructions available per channel for a 300 MHz PowerPC

  @500 MIPS/100Hz/2000 channels

  = 5MIPS/beamcrossing/2000chanels

  = 2500 instructions per channel.
- This seems at least 10x too many, i.e. one should be able to monitor 20K channels per CPU, even at 120 Hz.

Another example is the DC feature extraction of BaBar

- This consists of adding up to 32 bytes and determining if the result is over some threshold (well not quite that simple).
- This must be done for ~400 hits at 2KHz, i.e. about 1.5-2.0usecs per hit.
- The basic cost is 32*3nsec or about 100nsecs per operation that is done for each sample.
- This translates to a budget 15-20 cycles per sample.
- Accomplished the task in .7-1.3 usecs. Original was >5 usecs.

JJRussell
russells@slac.stanford.edu
(650) 926-2583

Recommendations/Observations

- Would concentrate on the high end applications, i.e. those applications that define our needs, i.e. those that need

  High bandwidth

  Low latency

- Would avoid implementing too much policy in the field

  Have the CPUs which are geographically distributed along the length of the accelerator do only what is absolutely necessary.

  Using high bandwidth, but not necessarily ultra low latency networks, move the data to a central processing farm where...

  A low latency, local bus can be used as an interconnect.

- This accomplishes a couple of goals

  The need to upgrade the field CPUs is reduced.

  This need is transferred to the central processing farm where it likely easier to do.

  Follows the industry trends by employing what they do efficiently and cheaply where appropriate.

  The penalty is one round trip delay, ~30usecs.

- With the RISC processors, the difference between the average performance and the peak performance has grown.
- What this is really saying is, if we do our homework and understand our problem, we can save a bundle by getting closer to the peak capability.

JJRussell
russells@slac.stanford.edu
(650) 926-2583