PCI Coprocessor Expansion Card

Alex Barabanov
Shawn Crabb
Thomas Gongaware
David Palchak
Overview

Embed a microprocessor on a PCI expansion card
Allow the host system to offload repetitive computations
Specialize embedded tasks for maximum performance
Allow for easy task reconfiguration
Hardware Components

- Intel 960 HA MIPS processor
- PLX Technologies 9030-RDK prototyping board
- Flash ROM (1MB) for firmware
- Dual port SRAM (4 MB) for data buffers
- 33 Mhz 32-bit PCI interface
Software Components

Linux 2.4 kernel driver for hardware/software interaction
Modified libcrypto.so, libssl.so
Embedded subroutines to perform MD5, RSA algorithms
EEPROM reprogramming utility
Details I - Hardware

PCI/Processor synchronized using interrupts
SRAM used for data storage
  Dual ported
Address segment multiplexing
  Handled by 9030 PCI controller
Reconfigurable via dedicated EEPROM
Details II - Firmware

Stores power-on initialization code
Contains driver interaction routines
  Synchronizes with PCI controller
  Manages memory
  Marshalls processor interrupts
  Schedules worker thread execution
Requires documented hardware interface
Details III – User Code

- Embedded optimized subroutines for specific tasks
  - Modular exponentiation
  - Bit parity
- Operates on buffered data blocks
- Code must be reentrant (thread-safe)
- Potentially performance critical
PCI Interface

Handled by on-board PLX 9030 controller
DMA based data transfer/control signaling
Supports burst block transfers
Provides a generic interface to hardware on card
Hardware/Driver Interface

Most difficult aspect of project
Indirect interface (through PCI controller)
Provide set of common control tasks

Task Examples:
- Status Query
- Receive data/Request data
- Begin/Pause/End task execution
- Write to EEPROM
Firmware Interface

Firmware invokes embedded routines
Need predefined assembly conventions
  Argument passing/Return values
  Caller/Callee saved registers
Independent control threads
  Memory Manager
  Thread scheduler
  EEPROM programmer
  Status Monitor
Power-On Initialization Routine
User Process API

Hardware access routines used by library functions
EEPROM control routines for firmware update utility
Primary hardware abstraction
  DMA based data transfer/control
  General purpose routines
Multithreaded
Object-oriented API
  Resource Access Policies
Obstacles

Hardware
- Mounting SMT parts on PCB
- Swapping out SRAM chips for higher capacity parts
- Formalizing interaction between 9030 I/O controller and microprocessor

Software
- Linux driver development
- PCI / DMA protocols
- MIPS cross-compiling
Bill of Materials - I
<table>
<thead>
<tr>
<th>Part Number</th>
<th>Description</th>
<th>Quantity</th>
<th>Price</th>
<th>Vendor</th>
</tr>
</thead>
<tbody>
<tr>
<td>CY7C019-12AC</td>
<td>SRAM</td>
<td>5</td>
<td>$7</td>
<td>Arrow, Mouser</td>
</tr>
<tr>
<td>AT29C010A-12JC</td>
<td>Flash ROM</td>
<td>5</td>
<td>$58</td>
<td>Atmel</td>
</tr>
<tr>
<td></td>
<td>TOTAL</td>
<td></td>
<td>$58</td>
<td>Compass</td>
</tr>
</tbody>
</table>
Basic Schedule - I

Summer
- Save $$$ and purchase parts
- Acquire documentation and literature

September
- Build the PCI card (Alex, Dave)
- Write Linux driver

October
- Write firmware
- Write and simulate cryptography assembly (Shawn, Tom)
Basic Schedule - II

November
Modify Linux libraries to export work
Integration, testing, debugging

December
Integration, testing, debugging
Presentation
Questions?