Overview

- Raspberry Pi
- Enclosure
- Daughter Card
- Monitor Node
- Power Supply
- LDAP
- Slurm
- NFS
- OrangeFS
Hardware Overview

Figure 1: Hardware Diagram
Raspberry Pi 3

- Broadcom BCM28377 SOC
  - Quad Core Cortex A53 (AArch64, ARMv8 compatible)
  - VideoCore IV GPU
- 1GB LPDDR2 RAM
- 10/100 Ethernet, 4 USB 2.0 Ports

Figure 2: Raspberry Pi 3
Enclosure

- Rack Enclosure on Casters
- 2 custom Plexiglass boxes with 3D printed corner brackets house the worker nodes (dimensions: 17in × 9.25in × 8.75in).
- Multiple rack shelves to hold switches and power supplies.
- Pull-out shelf for computer keyboard.
Enclosure

Figure 3: Custom Box

Figure 4: Rack with Casters
Enclosure

Figure 5: 3D Printed Parts
Enclosure

**Figure 6:** Filled Enclosure Front

**Figure 7:** Filled Enclosure Rear
Daughter Card

- Needed a way to measure power input to nodes.
- Convert analog measurements to digital packets.
- Send information to a Monitor Node.

Figure 8: Current Sense Technique
Figure 9: Daughter Card Schematic
Daughter Card

Figure 10: Daughter Card 3D Render

Figure 11: Sixteen Populated Daughter Cards

- Altium Designer
- OSH Park
- Tweezers and Patience
• Adafruit has a C++ library for the INA219 that was originally used

• Someone had created a Python port that was much easier to integrate into our usage scenario
  – getShuntVoltage_mV()
  – getBusVoltage_V()
  – getCurrent_mA()

• Originally daughter cards connected through Python Paramiko library ssh connections for ease of access
  – This was slow and not the most easily expandable solution

• Now the monitor node handles communication with all daughter cards
Monitor Node Backend

- Separate Raspberry Pi with Touchscreen
- Python monitoring script runs as a service on each node
- Each node sends:
  - CPU temperature
  - CPU load
  - CPU frequency
  - SoC core voltage
- Nodes with daughter cards also send:
  - Supply current
  - Supply voltage
- Information is sent via UDP packets
- Information sent from nodes every 2 seconds
Monitor Node GUI

- Monitoring GUI implemented in Python and GTK and Glade
- GUI can show a map of any of the monitored metrics
- Shows min/max of the measurement metrics

Figure 12: Monitor Gui
Monitor Node GUI

- Monitoring GUI implemented in Python and GTK and Glade
- GUI can show a map of any of the monitored metrics
- Shows min/max of the measurement metrics

Figure 13: Monitor Map
• Discovered power delivery issue when HPL tests showed CPU frequency throttling
• Voltage dropout was primarily due to cable loss from cables and board-level power management devices (up to 1.5 Ω)
• Power distribution system has 10 Switch Mode Power Supply (SMPS) units rated at 20 A current capacity at 5 V (100W)
• Each SMPS drives a single 7-port USB hub, modified to accommodate approximately 4A per port.
USB Hub Mods

- 100mil trace only sufficient for up to 4A current load.
- Find optimum location to split trace in order to fan out current load.
- Slice trace with repeated scoring of Exacto-knife.

Figure 14: Sliced Power Bus
USB Hub Mods

Figure 15: Power Rail Connections

- Choose fanning locations and solder 16AWG stranded wire to points.
USB Hub Mods

Figure 16: Ground connection and Heat-shrink

- Remove solder mask by scoring from Printed Circuit Board (PCB) to reveal Ground (GND) Plane.
- Solder 16AWG stranded wire to exposed GND plane.
- Twist pairs to establish electromagnetic coupling between transmission and return path.
USB Hub Mods

Success!

Figure 17: Power Supply
Software Overview

Figure 18: Software Stack
LDAP

- Used to store Users and Groups
- Made easier to use with LDAP Scripts

Figure 19: LDAP Structure
Slurm

- Used for resource management

Figure 20: Slurm Architecture
NFS

- Network File System
  - Distributed file system Protocol
- Native component of the Linux Kernel
  - Current version NFSv4
- MPI and NFS
  - Message Passing Interface only supported under no attribute caching disabled
- NFS on BOB
  - Only used for storing users home directories
  - does not support MPI
What is OrangeFS?

- Parallel Virtual File System
- Object based design
- Client/Server Architecture
- Metadata and data services
- BMI - TCP/IP network communication

Figure 21: File Distribution
How OrangeFS Works

How OrangeFS Works

1. A program running on a compute node requests access to a file.
2. The OrangeFS client looks up the file’s locations in the metadata servers.
3. The client accesses the file’s multiple parts in parallel.

Figure 22: File Distribution
OrangeFS on B.O.B.

- **1 Server**
  - Mounted to 4 TB external hard drive
  - Provides highest throughput given 100Mbit/s Ethernet
  - 3 more servers available to integrate

- **64 clients**
  - FUSE
  - MPICH2
  - 262,144 Byte transfer buffers with 8 buffers per bulk transfer
  - Metadata and data files synchronized with every write operation
  - Uses a thread based implementation of Asynchronous IO
  - Additional 64 GB of unused storage space available on SD card of each client

- Provides no methods of security
## OrangeFS vs NFS

5.6 GB file, 1 MB block, 8 Nodes/32 Tasks, POSIX API

<table>
<thead>
<tr>
<th>Xfer Size</th>
<th>NFS</th>
<th>OFS</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 KB</td>
<td>10.41</td>
<td>10.78</td>
</tr>
<tr>
<td>100 KB</td>
<td>9.16</td>
<td>11.46</td>
</tr>
<tr>
<td>1 MB</td>
<td>9.65</td>
<td>11.12</td>
</tr>
</tbody>
</table>

Table 1: Read Rate (MB/s)

<table>
<thead>
<tr>
<th>Xfer Size</th>
<th>NFS</th>
<th>OFS</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 KB</td>
<td>11.42</td>
<td>9.73</td>
</tr>
<tr>
<td>100 KB</td>
<td>11.25</td>
<td>11.31</td>
</tr>
<tr>
<td>1 MB</td>
<td>11.34</td>
<td>11.44</td>
</tr>
</tbody>
</table>

Table 2: Write Rate (MB/s)

5.6 GB file, 1 MB block, 64 nodes/256 tasks, POSIX API

<table>
<thead>
<tr>
<th>Xfer Size</th>
<th>NFS</th>
<th>OFS</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 KB</td>
<td>8.4</td>
<td>5.64</td>
</tr>
<tr>
<td>100 KB</td>
<td>8.7</td>
<td>6.96</td>
</tr>
<tr>
<td>1 MB</td>
<td>8.5</td>
<td>7.85</td>
</tr>
</tbody>
</table>

Table 3: Read Rate (MB/s)

<table>
<thead>
<tr>
<th>Xfer Size</th>
<th>NFS</th>
<th>OFS</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 KB</td>
<td>11.59</td>
<td>10.05</td>
</tr>
<tr>
<td>100 KB</td>
<td>11.57</td>
<td>11.51</td>
</tr>
<tr>
<td>1 MB</td>
<td>11.64</td>
<td>11.54</td>
</tr>
</tbody>
</table>

Table 4: Write Rate (MB/s)