Hera-B DAQ System and its self-healing abilities

V.Rybnikov, DESY, Hamburg

1. HERA-B experiment
2. DAQ architecture
   - Read-out
     - Self-healing tools
   - Switch
     - SLT nodes isolation
3. Run control system
4. Self-healing tools (software)
   - Releasing resources
   - Process recovery
HERA-B experiment (sub-detectors)

<table>
<thead>
<tr>
<th>Detector</th>
<th>Technology</th>
<th>Number of Channels</th>
<th>Bias per BX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vertex detector</td>
<td>Si-strip</td>
<td>136 k</td>
<td>≈ 0.05</td>
</tr>
<tr>
<td>Tracker</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inner (2-6 cm)</td>
<td>Si-strip</td>
<td>40 k</td>
<td>≈ 0.02</td>
</tr>
<tr>
<td>Inner (6-19 cm)</td>
<td>microstrip gas-chamber (GEM)</td>
<td>135 k</td>
<td>≈ 0.04</td>
</tr>
<tr>
<td>Outer (&gt;19 cm)</td>
<td>honeycomb-DC</td>
<td>120 k</td>
<td>≈ 0.15</td>
</tr>
<tr>
<td>High-pT trigger</td>
<td>gas pixel/straw</td>
<td>26 k</td>
<td>≈ 0.05</td>
</tr>
<tr>
<td>B^0 \to \pi^+\pi^-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RICH</td>
<td>CuF_39 radiator</td>
<td>32 k</td>
<td>≈ 0.1</td>
</tr>
<tr>
<td>Kaon iden.</td>
<td>PMT photon detector</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Small-Angle TRD</td>
<td>Fiber radiator</td>
<td>15.7 k</td>
<td>≈ 0.1</td>
</tr>
<tr>
<td>Electron iden.</td>
<td>Straw chambers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EM calorimeter</td>
<td>W/Pb scint.</td>
<td>5.3 k</td>
<td>≈ 0.2</td>
</tr>
<tr>
<td>Electron iden.</td>
<td>Shashlik</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Muon system</td>
<td>gas pad + pixel</td>
<td>31.3 k</td>
<td>&lt; 0.01</td>
</tr>
<tr>
<td>Muon iden.</td>
<td>Prop. tubes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>≈ 550 k</td>
<td></td>
</tr>
</tbody>
</table>
DAQ architecture

Critical Points

~1100 SHARC nodes
240 SLT nodes
100 x 2 4LT CPUs

~2000 processes on ~1500 nodes

Event Rate

10 MHz
50 MHz < 30
13 Gb/s

500 Hz 200
165 MB/s

50 Hz 150
22 MB/s

5 May 2003
V. Rybnikov, DESY
DAQ architecture (SHARC board)

- 6U VME card (MSC, Stutensee, Germany)
- 6 ADSP-21060 (Analog Devices), 40 MHz
- ADSP chip holds 512 KB on-chip memory
- Global memory bus (240 MB/s in 48bit words)
- External memory 256K x 32
- 10 DMA controllers / chip
  - 6 for 4 bit parallel links (40 MB/s)
  - 4 for global memory communication
- VME interface to write/read ADSP and global memory

<table>
<thead>
<tr>
<th>SWITCH</th>
<th>44</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLBs</td>
<td>140</td>
</tr>
<tr>
<td>Event Controller</td>
<td>2</td>
</tr>
<tr>
<td>FCS interface</td>
<td>1</td>
</tr>
</tbody>
</table>
Total: ~2070 FEDs
(32-1024 channels)

Push-down system
No missing clock allowed
No hardware recovery

27-40 MHz
0.5 – 60 m
Self-healing tools (read-out recovery)

- FED error threshold
- min period between consecutive recoveries
- max number of consecutive recoveries

FED expert

- ACTION

Monitors
- check event header information for every FED w.r.t. errors

DATA STREAM

common monitor

monitor ITR

monitor SVD
Self-healing tools (read-out recovery)

- stop triggers
- reset FEDs
- Re-chain (initialize) event buffers and Event Controller
- start triggers

Action takes $< 5$ sec
Run re-initialization $\sim 2$ min
Run re-start $\sim 8 - 10$ min
DAQ architecture (switch)

Routing tables server

- reads the switch connection data base
- creates routing tables in memory
- pushes down the tables into every SHARC node after the boot-up

SHARC to PCI interface boards are used to connect Second Level PCs to the SWICTH
Self-healing tools (SLT nodes isolation)

Distributor tasks:
- to send calibration constants to all Second Level Trigger (SLT) nodes
- to check status of the SLT nodes (processes) via ping-pong messages

Problem:
Accumulating messages addressed to a dead node (process) blocks the switch
Self-healing tools (SLT nodes isolation)

Routing table server

SLT process expert

Distributor

SLT process

Process server

Interconnections

Ping-pong

SLT process died

Terminate SLT process

Change routing

Routing information
Run control system

BASICS

- the process information for all runs is stored in the DAQ data base
  - list of processes
  - how to start them (args, env, etc)
  - where to start them
  - etc.
- all the processes are started remotely by means of process servers and managers
- clean-up of shared resources (shared mem, semaphores, etc) carried out during the start-up and stop procedures
Run control system (process service)

Features

- Process creation and termination on any ‘ONLINE’ machine
- Process status monitoring and notification about its change
- Monitoring the node resources utilization (CPU, memory, etc)

Implementation

- proserv commands
- proserv interface
- start interface
- inetd
- process server
- start
- stop
- kill
Run control system (process management)

“SYSTEM” Process Managers

Data Taking
Slow Control
Standalone
Test
Reprocessing
MC

Run Watch is the very first process for every run

Boot up procedure supporters

Data Base
Run control system (DAQ data base)

process configuration

process template
Run control system (run boot-up)

- Run Control GUI
  - “SYSTEM” Process Manager
    - global processes

Run Watch
- Checking process servers on all machines
- Restarting them if required
- Freeing resources by launching ‘fini’ scripts

Process manager
- COMP 1
  - comp 1 processes
- COMP N
  - comp N processes
Self-healing tools (process recovery)

Process Manager

Can be restarted?

Yes

No

Process server reports on process termination

Checks processes

Yes

No

restart

Can be restarted?

Forget

Critical?

Yes

No

5 May 2003 V.Rybnikov, DESY
HERA-B is a big complex experiment developed and built up by hundreds of scientists, engineers and technicians. The major developments are complete. Problems effecting data taking efficiency are being fixed by introducing self-healing tools.
Appendix (ONLINE expert tools)
Switch performance

Fig. 4. a) Maximum message rate (solid line) and throughput (dashed line) routing messages through the SHARC Board bus as a function of message size. b) Maximum message rate (solid line) and throughput (dashed line) routing messages from the ADSP link to SHARC Board Bus as a function of message size. The data points are measurements in a test bench and the lines are fitted numerical models.
Switch performance

The traffic pattern on the ADSP network can be modified in different ways to remove bottlenecks in the switch. Link connection can be redistributed to avoid the collision of high traffic links into a single ADSP router. Another option is to force static routing paths for messages from a defined origin to an specific destination. This is done at the level of the routing table computation. Finally, the network can be scaled adding additional switch blocks to increase the bandwidth. Overall the ADSP switch is not as flexible as the configuration of Ethernet switches where local bottlenecks can be controlled dynamically by a port grouping.

Another advantage of standard Ethernet switches is that the backplane is able to handle the maximum throughput in all ports simultaneously. The bandwidth of the SHARC board bus is equivalent to the total bandwidth of 4 link connections. In an uniform system (18 links input, 18 links output), the total number of messages routed through the bus is 5/6 of the total messages (15 links at each time). The switch bottleneck is in the SHARC board bus since it is just able to handle \( \approx 1/4 \) of the link transfer capabilities. This is reflected in Table 1, where the total bandwidth is the number of switch blocks times the switch block bandwidth.

<table>
<thead>
<tr>
<th>Bandwidth</th>
<th>Message rate</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2 GB/s</td>
<td>2.6 MHz</td>
<td>110 %</td>
</tr>
</tbody>
</table>

Table 1
Maximum performance for the switch topology described in section 7. Large (Bandwidth) and small message (Message rate) sizes are computed to estimate the contribution of Event Building and SLT RoI protocol. The performance numbers are compared with the SLT reference requirements as described in section 3.
Switch routing

```
ask_scn > routing SLP_/5 RICH_0
Destination address is 241

SLP_/5(1)->(2)SMCPC_7[BUS]SMCPC_9(5)->(2)SHCSB_54[BUS]SHCSB_36(5)->(3)SMCRICH_0[BUS]RICH_0
ask_scn > routing RICH_0 SLP_/5
Destination address is 5

RICH_0[BUS]SMCRICH_0(5)->(2)SHCSB_39[BUS]SHCSB_36(1)->(4)SMCPC_3[BUS]SMCPC_1(2)->(2)SLP_/5
```