Infiniband Event-Builder Architecture Test-beds for Full Rate Data Acquisition in LHCb

Enrico Bonaccorsi, Juan Manuel Caicedo Carvajal, Jean-Christophe Garnier, Guoming Liu, Niko Neufeld and Rainer Schwemmer

CERN

Computing in High Energy and Nuclear Physics, 2010

Introduction

- Upgrade of the experiment involving a multi-terabit readout
- Interesting technologies
  - 10 Gigabit Ethernet : successor of Gigabit Ethernet
  - InfiniBand : Challenger
- Is InfiniBand an interesting alternative to 10 Gigabit Ethernet ?
- What software implementation for the Event-Builder ?

Outline

1. LHCb Event-builder and its upgrade
2. InfiniBand
3. Test-bed

Outline

1. LHCb Event-builder and its upgrade

2. InfiniBand

3. Test-bed

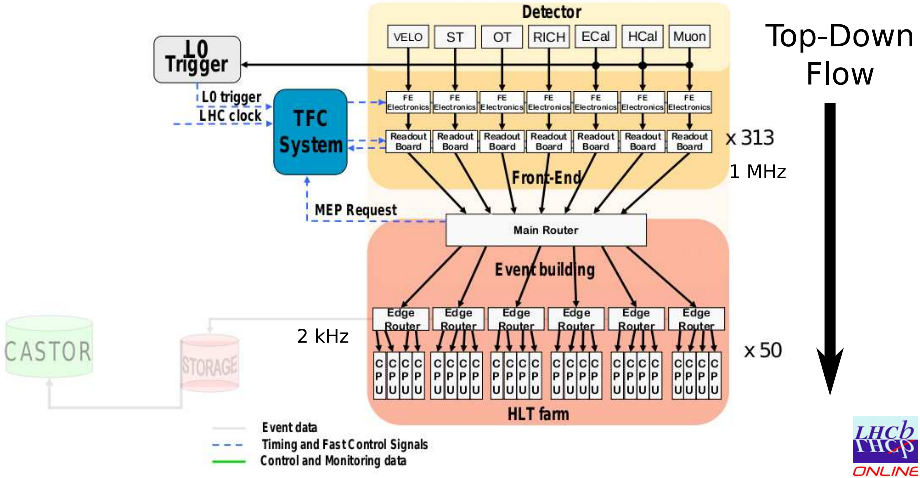
Outline

1. LHCb Event-builder and its upgrade
2. InfiniBand
3. Test-bed

Outline

1. LHCb Event-builder and its upgrade
2. InfiniBand
3. Test-bed

Protocols



Top-Down Flow

x 313

1 MHz

2 kHz

x 50

Figures

	Current	Upgrade
Event size	35 kB	100 kB
Read-out rate	1 MHz	30 MHz
Sources	313	1000
Sinks	1500	> 3000
Event-rate to storage	2 kHz	10 kHz

Aims

- Study the two main multi-gigabit technologies
- Study different network topologies
  - Cf. PS10-3-338 from Guoming Liu
- Study various software stacks
  - Push and pull protocols
  - Non-oriented connections protocols: Unreliable (UDP) and Realiable (RDS) datagrams
  - Oriented connections protocols: TCP, SDP
- Sources are FPGA, simple protocols are more interesting

Outline

1. LHCb Event-builder and its upgrade
2. InfiniBand
3. Test-bed

Overview

- Distributed computing
- Low latency
- Very promising bandwidth
- Data rate (and not signal rate):

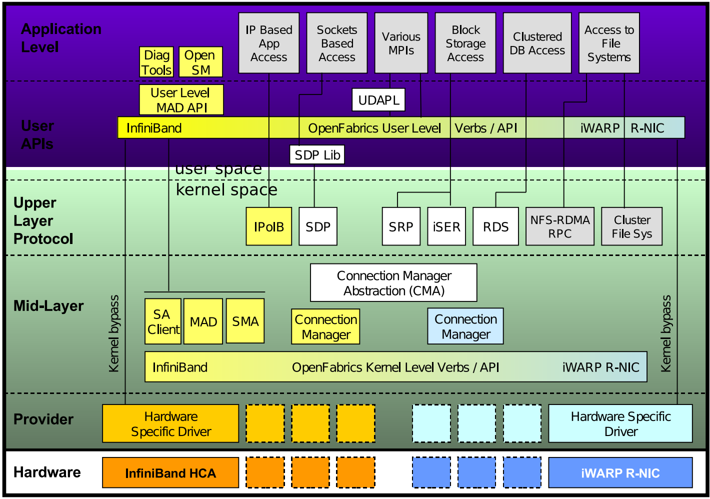
	SDR	DDR	QDR	FDR	EDR
1X	2 Gbit/s	4 Gbit/s	8 Gbit/s	14 Gbit/s	25 Gbit/s
4X	8 Gbit/s	16 Gbit/s	32 Gbit/s	56 Gbit/s	100 Gbit/s
12X	24 Gbit/s	48 Gbit/s	96 Gbit/s	168 Gbit/s	300 Gbit/s

- SDR, DDR and QDR use 8B/10B encoding
- Aggregation of links in units of 4
- Only a few vendors
- Huge software stack

Software stack: OpenFabrics Enterprise Distribution

OPENFABRICS ALLIANCE

OpenFabrics Software Stack



Application Level

Diag Tools | Open SM | IP Based App Access | Sockets Based Access | Various MPIs | Block Storage Access | Clustered DB Access | Access to File Systems

User APIs

User Level MAD API | UDAPL

InfiniBand | OpenFabrics User Level Verbs / API | iWARP R-NIC

SDP Lib

user space

kernel space

Upper Layer Protocol

IPoIB | SDP | SRP | iSER | RDS | NFS-RDMA RPC | Cluster File Sys

Mid-Layer

Kernel bypass

Connection Manager Abstraction (CMA)

SA Client | MAD | SMA | Connection Manager | Connection Manager

InfiniBand | OpenFabrics Kernel Level Verbs / API | iWARP R-NIC

Kernel bypass

Provider

Hardware Specific Driver | Hardware Specific Driver

Hardware

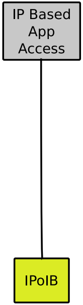
InfiniBand HCA | iWARP R-NIC

SA	Subnet Administrator
MAD	Management Datagram
SMA	Subnet Manager Agent
PMA	Performance Manager Agent
IPoIB	IP over InfiniBand
SDP	Sockets Direct Protocol
SRP	SCSI RDMA Protocol (Initiator)
iSER	iSCSI RDMA Protocol (Initiator)
RDS	Reliable Datagram Service
UDAPL	User Direct Access Programming Lib
HCA	Host Channel Adapter
R-NIC	RDMA NIC

Key: Common | InfiniBand | iWARP | Apps & Access Methods for using OF Stack

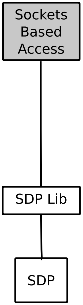
IPoIB

- Standard IP encapsulation over InfiniBand fabrics
- Relies on 2 modes of InfiniBand
  - Unreliable datagram: Max MTU = link MTU = 4096 B
  - Connected mode: Max MTU = $2^{31} B$
- Implementation using the libc socket
- No changes to your current source code
- No RDMA



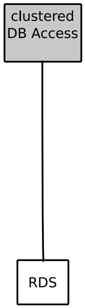
Sockets Direct Protocol (SDP)

- Defines a standard wire protocol over IB fabrics
- Local IP assignments and IP resolution using IPoIB
- Supports only stream sockets (*SOCK_STREAM*)
- 2 ways to use
  - Minimal reimplementation
  - LD_PRELOAD=libsdp.so
- Minor changes to your current source code
- Good use of RDMA



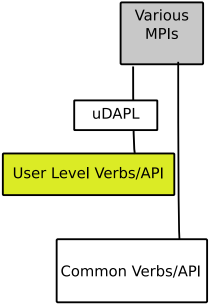
Reliable Datagram Sockets (RDS)

- Developed by Oracle and added to the OFED
- Not restricted to InfiniBand
- 1 socket to talk to N destinations
- Included in Linux kernel 2.6.30
- New source code
- Minimal use of RDMA



Verbs, MPIs and standard limits

- Allows full use of RDMA
- List of standard verbs, syntax is vendor specific
- About 1200 pages of verbs out of the 1700 page InfiniBand Architecture Specification
- If used correctly, maximum performances
- Each vendor brings its own non standard optimized MPI implementation



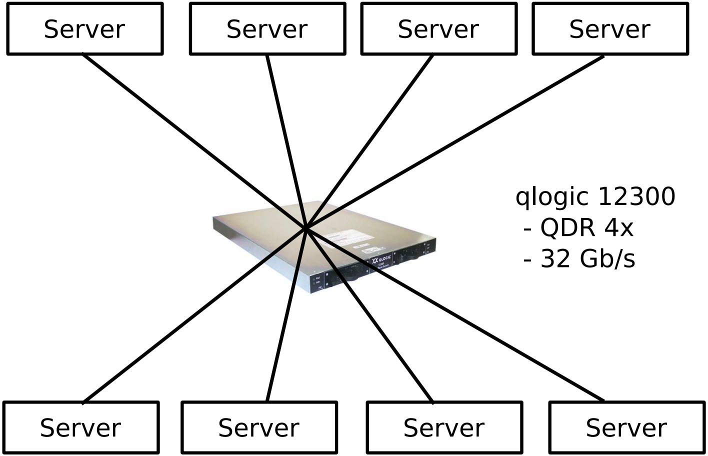
Outline

1 LHCb Event-builder and its upgrade

2 InfiniBand

3 Test-bed

Configuration



qlogic 12300
- QDR 4x
- 32 Gb/s

Server configuration

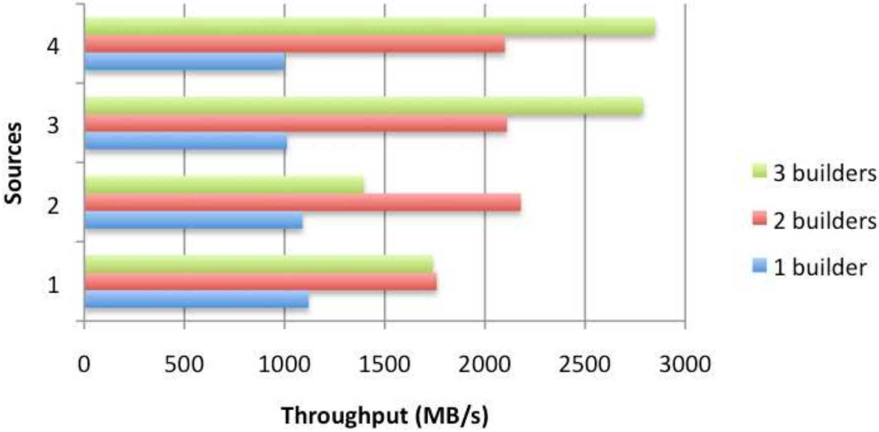
Processor type	Intel Xeon E5520
Processors x cores x clock (GHz)	2 x 4 x 2.27
RAM (GiB)	3
HCA	qle7340 4x QDR
kernel	2.6.18

net.core.rmem_default	16777216
net.core.wmem_default	16777216
net.core.rmem_max	16777216
net.core.wmem_max	16777216
net.core.netdev_max_backlog	250000

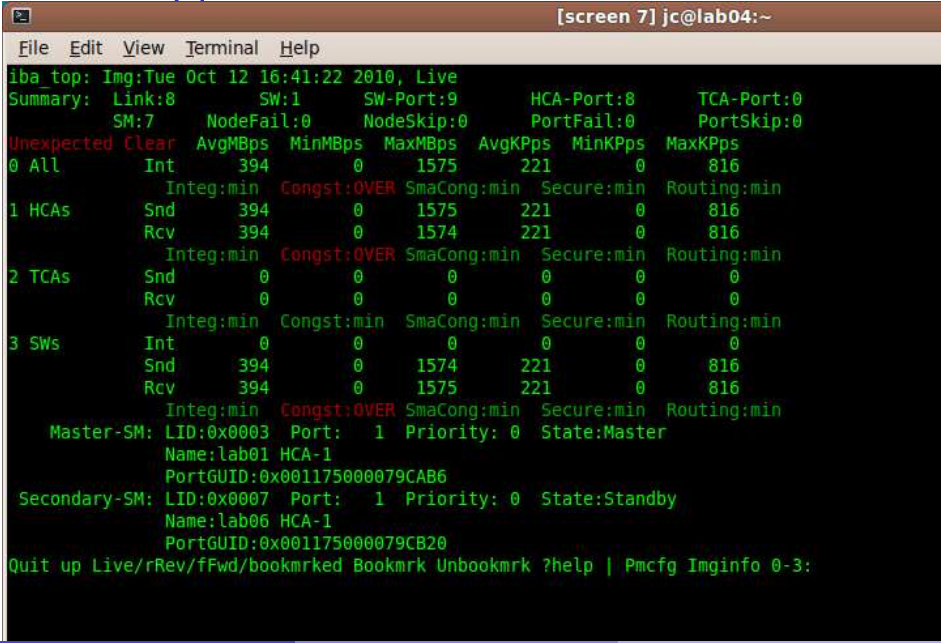
Implementation

- Node synchronization via Precise Time Protocol (ptp)
- From scratch, plain C/C++ for Linux, focusing on network performance and troubleshooting
- Portability InfiniBand/10GbE
  - TCP/IP stack done at kernel/API level, except RAW Ethernet/IP
  - SDP only related to InfiniBand
  - RDS fully portable
  - RDMA: InfiniBand verb syntax used by some 10GbE suppliers

Preliminary performance results: PUSH over UDP



Preliminary performance results



```
iba_top: Img:Tue Oct 12 16:41:22 2010, Live
Summary:  Link:8        SW:1      SW-Port:9       HCA-Port:8      TCA-Port:0
          SM:7     NodeFail:0     NodeSkip:0      PortFail:0      PortSkip:0
Unexpected Clear  AvgMBps  MinMBps  MaxMBps  AvgKPps  MinKPps  MaxKPps
0 All        Int      394        0     1575      221        0      816
               Integ:min  Congst:OVER SmaCong:min  Secure:min  Routing:min
1 HCAs       Snd      394        0     1575      221        0      816
             Rcv      394        0     1574      221        0      816
               Integ:min  Congst:OVER SmaCong:min  Secure:min  Routing:min
2 TCAs       Snd        0        0        0        0        0        0
             Rcv        0        0        0        0        0        0
               Integ:min  Congst:min  SmaCong:min  Secure:min  Routing:min
3 SWs        Int        0        0        0        0        0        0
             Snd      394        0     1574      221        0      816
             Rcv      394        0     1575      221        0      816
               Integ:min  Congst:OVER SmaCong:min  Secure:min  Routing:min
    Master-SM: LID:0x0003  Port:   1  Priority: 0  State:Master
               Name:lab01 HCA-1
               PortGUID:0x001175000079CAB6
 Secondary-SM: LID:0x0007  Port:   1  Priority: 0  State:Standby
               Name:lab06 HCA-1
               PortGUID:0x001175000079CB20
Quit up Live/rRev/fFwd/bookmrked Bookmrk Unbookmrk ?help | Pmcfg Imginfo 0-3:
```

First experience feedback

- A few issues experienced at the beginning
  - Performance drop over time
  - Negotiation problems
  - Congestion management to be investigated
  - Far from line rate
- Lack of online literature
- Always go to support, they are willing to work with you
- IPoIB not optimized according to the QLogic support
- MPIs are the best !
  - Dedicated implementations should allow maximum performance

Summary

- Push protocol over unreliable datagrams established
  - Optimization possible: aggregation of several IB interfaces over one IB link
- Future work
  - Implement PUSH over RDS and open MPI
  - Implement PULL over RDS, TCP, SDP and open MPI
    - TCP, SDP and MPIs might be too complex for FPGAs
    - Still interesting to know their performance