1. Readout software
CURRENT READOUT SYSTEM

I. data check by data-handler process
   I. Calculate CRC16 and compare CRC value attached by FEE
   II. XOR checksum calculated by software on COPPER
II. Data size reduction
   merging redundant header/trailer attached by b2link and COPPER
   Reduction by 15MB/s/ROPC at 30kHz trigger rate (<= 5COPPERs/ROPC, 4HSLB/COPPER)
III. Collect data from several COPPERs and do partial event-building and send data to High level trigger unit.

Electronics Hut

COPPER CPU

- HSLB
- Readout process #0

Network switch

ROPC (Readout PC)

- Readout process #0
- Readout process #1
- Readout process #n

Partial Event builder

Network switch

Event builder 1Rx/
High level trigger

Other ROPCs

GbE

GbE

GbE

10GbE

GbE

GbE
I. data check by data-handler process
   I. Calculate CRC16 and compare CRC value attached by FEE
   II. XOR checksum calculated by firmware on new readout hardware
II. Data size reduction
   merging redundant header/trailer attached by b2link and COPPER
   Reduction by 15MB/s/ROPC at 30kHz trigger rate (<5COPPERs/ROPC, 4HSLB/COPPER)
III. (Collect data from several COPPERs and do partial event-building and send data to High level trigger unit.)
Current format of RawCOPPER hdr/trl

1. Number of total words
2. 0x7f7f | Format ver.(8bit) | Number of words in this block
   (8bit)
3. exp no. (10bit=1024), run no.(22bit=4194304 including subrun)
4. event number(32bit)
5. From B2link FEE header 2 (TT-ctime | Trig-type)
6. From B2link FEE header 2 (TT-utime)
7. Node ID
8. b2link CRC error bit (4) | truncation mask (truncated or not) /
type of data (compressed,calibration, …)
9. offset to 1st block of user's data(FEE data)
10. offset to 2nd block of user's data(FEE data)
11. offset to 3rd block of user's data(FEE data)
12. offset to 4th block of user's data(FEE data)

Reserved (maybe used for checksum)
2. termination word of this block = 0x7fff0006

Use this version number to distinguish
Different data format.
Ver.0 : to 2014. June(including DESY test)
ver.1 : from June.2014
Ver.2 : from Apr. 2015 (rev.17269)
Ver. 3: For new readout data

1word=32bits

upPocketDAQ_4p1_RawCOPPERDataFormat.pdf
New data format for RawCOPPER hdr. and trl. (ver. alpha-190306)

- Remove COPPER header/trailer (which is currently removed on a readout PC.)
- Position in header will be removed to reduce header size.
- Error information in trailer (which link, what kind of error)
- XOR checksum (less CPU power to re-calculate on host servers or HLT)

1. Number of total words
2. 0x7f7f (Format ver.(8bit)) Number of words in this block (8bit)
3. exp no. (10bit=1024), run no.(22bit=4194304 including subrun)
4. event number(32bit)
5. From B2link FEE header 2 (TT-ctime | Trig-type)
6. From B2link FEE header 2 (TT-utime)
7. Node ID
8. b2link CRC error bit (4) | truncation mask (truncated or not) / type of data (compressed, calibration, ...)

1 word = 32 bits

B2link (FEE/HSLB) header and trailer

Detector buffer(slot A)
Detector buffer(slot B)
Detector buffer(slot D)

1. Error bit
2. Error slots
3. XOR checksum
4. termination word of this block = 0x7fff0006

➢ Remove COPPER header/trailer (which is currently removed on a readout PC.)
➢ Position in header will be removed to reduce header size.
➢ Error information in trailer (which link, what kind of error)
➢ XOR checksum (less CPU power to re-calculate on host servers or HLT)
Current format of HSLB/B2L hdr/trl

➢ FEE sends “B2L hdr + FEE data + B2L trl”.
➢ hslb_receiver.vhd adds HSLB hdr and trl.
➢ Some header entries (blue broken lines) are copied in RawCOPPER header and removed on ROPCs.

Figure 5: Data format as read out by the COPPER. The header and trailer words labelled with HSL are attached by HSLB, the words with B2L are attached by the belle2link component, and the words with FEE are those written into the belle2link component by the frontend firmware.
New data format for B2L/HSLB hdr. and trl. (ver. alpha-190306)

- Basically unchanged
- Add length and ID (link number = 0, 1, 2, ... 47)

“ID(0-47) : 8bits” + “# of words : 24bits”

Figure 5: Data format as read out by the COPPER. The header and trailer words labelled with HSL are attached by HSLB, the words with B2L are attached by the belle2link component, and the words with FEE are those written into the belle2link component by the frontend firmware.
Branch: feature/for-DAQ-readout-upgrade

Todo
- Test with dummy data source
- Replace the current recv() function which receives data from COPPER with another function which is specific for new readout hardware.
- Packer module for offline software test. (not urgent)
2. Slow control software
Slow control daemon processes on the new setup

Current setup

Readout PC

rocontrold <hostname> -d
runcontrold <hostname> -d

cprcontrold <hostname> -d

COPPER

New setup

Readout PC

runcontrold <hostname> -d
rocontrold <hostname> -d
cprcontrold <hostname> -d

FELIX/PCIe40

CPPF
It is a daemon process which communicates with the run-control system and readout hardware ( = COPPER(+HSLB(+FEE) ) ).

```
cprcontrold
```

```
CprCallback.cc
- virtual void initialize(const DDBObject& obj);
- virtual void configure(const DDBObject& obj);
- virtual void load();
- virtual void abort();
- virtual void term();
- virtual void monitor();
- virtual void monitor_th();
```

```
Configuration via b2link
- SVDFEE.cc
- CDDFEE.cc
- ...
```

```
Access to HSLB register
- HSLB.cc
```
Slow control on COPPER and HSLB

Detector experts developed codes.
: daq_slc/copper/arich

DAQ group developed this core functions in DAQ slc library

User software

Driver software (hslb.ko)

HSLB registers -> new RO registers

FEE registers -> unchanged

ARICHMerger.cc
➢ m_hslb.writefee32(b2adr, b2val);
➢ b2val = m_hslb.readfee32(b2adr);

copper/HSLB.cc
➢ readfn(), readfn32()  
➢ writefn(), writefn32()  
➢ readfee8(), readfee32()  
➢ writefee8(), writefee32()  

release/daq/slc/extra/hsprogs/libhslb.c
a. readfn : ioctl(fd, HSLBIO_GET(adr), &val)
b. writefn : ioctl(fd, HSLBIO_SET(adr), val)  
c. readfee8(int fd, int adr) : For HSLB, same as readfn(fd, adr)  
d. writefee8(int fd, int adr, int val) : For HSLB, same as writefn(fd, adr, val)  
e. readfee32(int fd, int adr, int *valp)  
f. writefee32(int fd, int adr, int val)  
g. writestream : writing a file

./daq/slc/extra/hsprogs/hslb.h
int ioctl(int fd, unsigned long request, ...);

./hsdriver/hslb.c
static int hslb_ioctl(struct inode *inode, struct file *filp, unsigned int cmd, unsigned long arg)
fngeneric_set/getreg()
New SLC software (in the PCIe40 case)

- Replacing read/write function in the current SLC software
  - Daq_slc repository: branch: feature/slc_wo_COPPER_for_DAQupgrade

Copying Patrick-san’s PCIe40 library in daq_slc/extra/hsprogs/PCle40

```c
int writefee8(int fd, int adr, int val)
{
    #ifdef USE_PCIE40
    return pcie40_writefee8(fd, adr, val);
    #else
    if (devtype == FIN_HSLB) {
        if (adr <= 0 || adr > 0x7f) return -1;
        if (ioctl(fd, HSLBFEE8_SET(adr), val) < 0) {
            sprintf(errmsg, "cannot write %s: %s\n", DEVICE, strerror(errno));
            return -1;
        }
        return 0;
    }
    else {
        int ret = 0;
        ret |= writefn(fd, HSREG_CSR, 0x05); /* reset read fifo */
        ret |= writefn(fd, adr, val & 0xff);
        ret |= writefn(fd, HSREG_CSR, 0x0a); /* parameter write */
        return ret;
    }
    #endif
}
```

- Compiling (make sl6) was successful after replacing read/write/open/close function.
- To be modified
  - HSLB register access part
  - Increase the max number of link per readout board
    (currently 4/COPPER -> 48 /new RO, for example)

➢ Same strategy (replacing just read/write functions in SLC repository) can be used for other proposals.
Summary

➢ For new readout hardware,
  ➢ DAQ process
  ➢ SLC process
  need to be updated.

➢ DAQ program
  ➢ Remove DAQ process on COPPER
  ➢ New unpacker for new format

➢ SLC program
  ➢ Cprcontrold will be moved to ROPC
  ➢ Try to minimize the modification (especially sub-detector dependent part.)
    ➢ Basic idea is replacing read/write functions for new hardware.

➢ Git branches are made.
  ➢ Development is ongoing.
  ➢ These software is necessary for TOP B4 demo by proponents.
/hsdriver/hslb.c
static int hslb_ioctl(struct inode *inode, struct file *filp, unsigned int cmd, unsigned long arg)
_IOC_TYPE(cmd) are defined in hsprogs/hslb.h.
#define HSLB_MAGIC 0xe3
#define HSLB_FEE8_MAGIC 0xe4
#define HSLB_FEE32_MAGIC 0xe5
#define HSLB_FEESTREAM_MAGIC 0xe6
The functions below are called depending on above values.
- fee8_getreg(), fee8_setreg()
- fee32_getreg(), fee32_setreg()
- hslb_setreg(), hslb_getreg() : // streaming

./hsdriver/hslb.c
** fee8
- getreg : fee8_getreg(slot, offset, *valp);
  ret = fngeneric_setreg(slot, HSLBADDR_CSR, 0x05); /* reset read fifo */
  ret = fngeneric_setreg(slot, offset, 0x0c); /* dummy value */
  ret = fngeneric_setreg(slot, HSLBADDR_CSR, 0x07); /* 8-bit read */
Waiting until the return value becomes 0x11,
  ret = fngeneric_getreg(slot, HSLBADDR_STAT, &stat);
  if (ret != 0) return ret;
  if (stat == 0x11) break;
Then start getting a value
  ret = fngeneric_getreg(slot, HSLBADDR_D32D, valp);

- setreg
  ret = fngeneric_setreg(slot, HSLBADDR_CSR, 0x05); /* reset read fifo */
  ret = fngeneric_setreg(slot, offset, val & 0xff);
  ret = fngeneric_setreg(slot, HSLBADDR_CSR, 0x0a); /* 8-bit write */

hslb_feecontrol.vhd
FD32R <= '1' when LeControl = '1' and LDsync = x"0c" else '0';
** fee32

- getreg
  ret = fngeneric_setreg(slot, HSLBADDR_CSR, 0x05); /* reset read fifo */
  ret = fngeneric_setreg(slot, HSLBADDR_CSR, 0x0c); /* 32-bit read */
  ret = fngeneric_setreg(slot, HSLBADDR_SERIAL, (offset >> 8) & 0xff);
  ret = fngeneric_setreg(slot, HSLBADDR_SERIAL, (offset >> 0) & 0xff);
  ret = fngeneric_setreg(slot, HSLBADDR_CSR, 0x08); /* end of stream */

Waiting until the return value becomes 0x11,
  ret = fngeneric_getreg(slot, HSLBADDR_STAT, &stat);
  if (ret != 0) return ret;
  if (stat == 0x11) break;

Then start getting a value
  ret = fngeneric_getreg(slot, HSLBADDR_D32D, &bytes[3]);
  ret = fngeneric_getreg(slot, HSLBADDR_D32C, &bytes[2]);
  ret = fngeneric_getreg(slot, HSLBADDR_D32B, &bytes[1]);
  ret = fngeneric_getreg(slot, HSLBADDR_D32A, &bytes[0]);

- setreg
  ret = fngeneric_setreg(slot, HSLBADDR_CSR, 0x05); /* reset read fifo */
  ret = fngeneric_setreg(slot, HSLBADDR_CSR, 0x0b); /* 32-bit write */
  ret = fngeneric_setreg(slot, HSLBADDR_SERIAL, (offset >> 8) & 0xff);
  ret = fngeneric_setreg(slot, HSLBADDR_SERIAL, (offset >> 0) & 0xff);
  ret = fngeneric_setreg(slot, HSLBADDR_SERIAL, (val >> 24) & 0xff);
  ret = fngeneric_setreg(slot, HSLBADDR_SERIAL, (val >> 16) & 0xff);
  ret = fngeneric_setreg(slot, HSLBADDR_SERIAL, (val >> 8) & 0xff);
  ret = fngeneric_setreg(slot, HSLBADDR_SERIAL, (val >> 0) & 0xff);
  ret = fngeneric_setreg(slot, HSLBADDR_CSR, 0x08); /* end of stream */