Minidaq app Tutorial

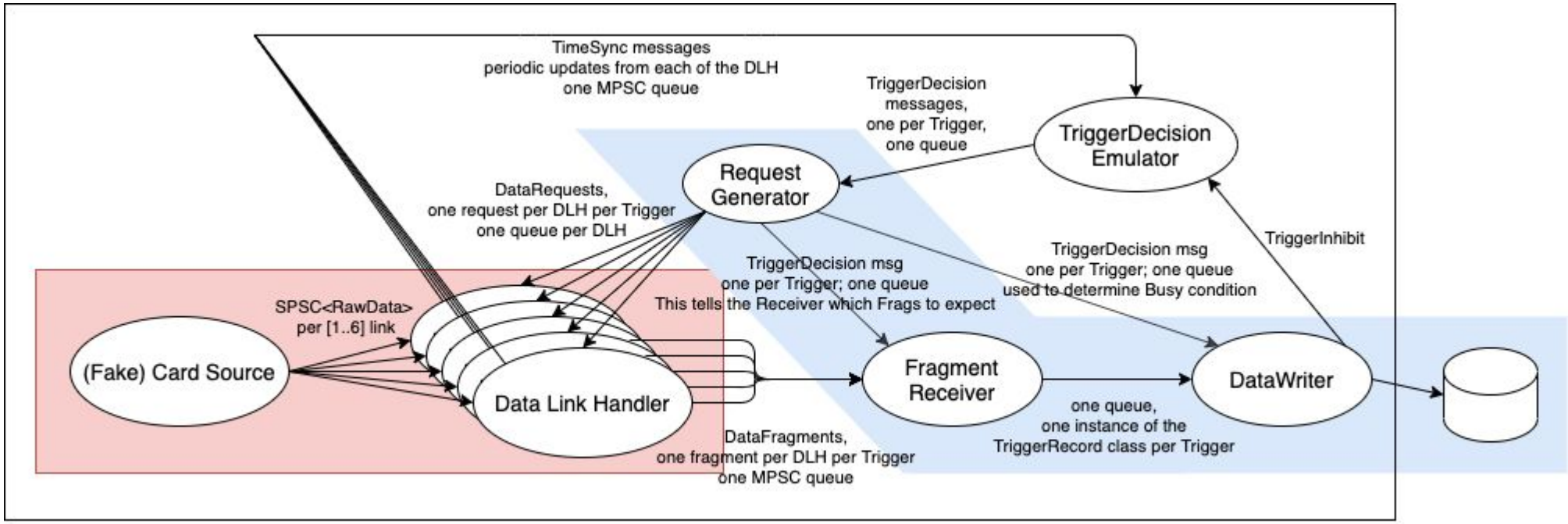
Based on **dunedaq-v2.2.0** release

Dr. Carlos Chavez Barajas

UNIVERSITY OF LIVERPOOL

DAQ/SC meeting, 1st February 2021

Minidaq components



MiniDAQApp Snapshot, 05-Jan (B), UD & DF Scope Highlighted

ReadOut **DataFlow**

TrigerDecision emulation and TimeSync updates

Preparation to run minidaq

1) Checkout daq-buildtools package for **dunedaq-v2.2.0** release:
   - git clone https://github.com/DUNE-DAQ/daq-buildtools.git -b dunedaq-v2.2.0
     - Only needs to be run once
     - Allows to setup as many work areas as you want
2) Prepare your “dbt” environment
   - source daq-buildtools/dbt-setup-env.sh
     - Access to scripts: **dbt-create.sh, dbt-build.sh**
     - Access to aliases: **dbt-setup-build-environment, dbt-setup-runtime-environment**
     - Needs to be run every time you want to work in one of your release areas
3) Create your release working area “work” (“MyTopDir” in our wiki pages)
   - mkdir work ; cd work ;
   - dbt-create.sh dunedaq-v2.2.0
     - Only needs to be run once
     - Creates a directory structure for running applications and code development
     - Creates file “dbt-settings” specifying the dune_products and their versions to be used
4) Prepare your runtime environment
   - dbt-setup-runtime-environment
     - Sets all necessary environment changes to access all parts of dunedaq-v2.2.0 release, access to executable and libraries, utilities of ups dune products
     - Needs to be run every time you want tor run minidaq app or any other app built as part of the relese

More details in the [Compiling-and-running-under-v2.2.0 wiki page]

Running minidaq application

1) Download the raw data file and rename it:
   - curl https://cernbox.cern.ch/index.php/s/VAqNtn7bwuQtff3/download -o frames.bin
     - If you change the name or location of this file, then change accordingly in config generation
2) Generate your configuration:
   - python -m minidaqapp.fake_app_confgen -d ./frames.bin -o '.' my_minidaq_config.json
   - python -m minidaqapp.fake_app_confgen --help
     - If your system doesn't have many cores available, use a downscaling factor.
3) Run the application:
   - daq_application -c stdin://./my_minidaq_config.json
     - The sequence of commands "init, conf, start, resume, stop, scrap"
     - Triggers will not be generated until after a resume command is issued
     - Default configuration will generate trigger records with 2 links each, at 1 Hz.
4) Running the application with REST command interface:

   Open 2 terminals and setup running environment (assume on the same host)

   In terminal one run:
   - daq_application -c rest://localhost:12345;

   In terminal two, go to your release area and run:
   - source daq-buildtools/dbt-setup-env.sh ; cd work ;
   - dbt-setup-runtime-environment
   - send-restcmd.py --interactive --file ./my_minidaq_config.json ;
     - Send sequence of commands "init, conf, start, resume, stop, scrap"
     - To change trigger rate, create a new configuration,run again, and issue "pause, resume"

More details in the Simple instructions for running the minidaq app

Peeking at the hdf5 output

1) Output file generated "**fake_minidaqapp_run000333_file0000.hdf5**"
    - Default run number "000333"

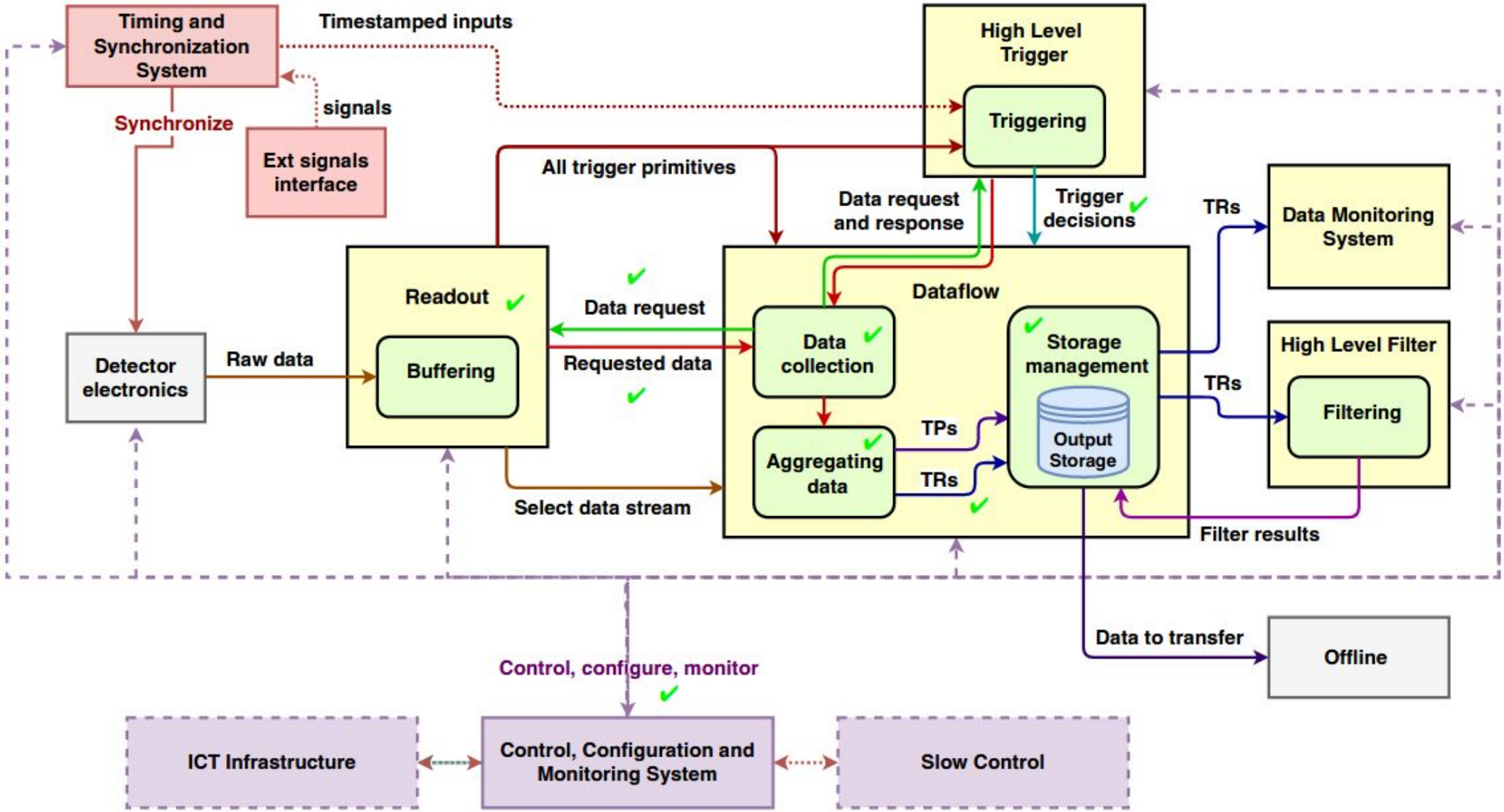
2) Using **h5dump**:
    - h5dump -H fake_minidaqapp_run000333_file0000.hdf5
        - Brute force dump, use more/less to search internally for TriggerRecord/Header

3) Use **hdf5_dump.py** tool (provided by Adam):

    For TriggerRecordHeaders:
    - python3 $DFMODULES_FQ_DIR/dfmodules/bin/hdf5dump/hdf5_dump.py -TRH -f fake_minidaqapp_run000333_file0000.hdf5

    For FragmentHeaders:
    - python3 $DFMODULES_FQ_DIR/dfmodules/bin/hdf5dump/hdf5_dump.py -H -f fake_minidaqapp_run000333_file0000.hdf5

DAQ System Components



Questions ?