Multi-purpose FC7-based test board: factorizing firmware

2016-11-01
Electronics for 2S and PS-Pt modules
Stefano Mersi

Uses of a multi-purpose test board

- System integration tests
  - Hybrid testing
  - Module integration
  - Larger structures
- System tests
  - Signal, noise measurements
  - Beam tests
- Production QC

- Variety of front-ends to readout & control
  - CBC3/MPA(+SSA)
  - With/without CIC
  - Electrical/optical
  - mini/full hybrid
- Potential issue:
  - large effort in firmware development
  - difficult to maintain/support

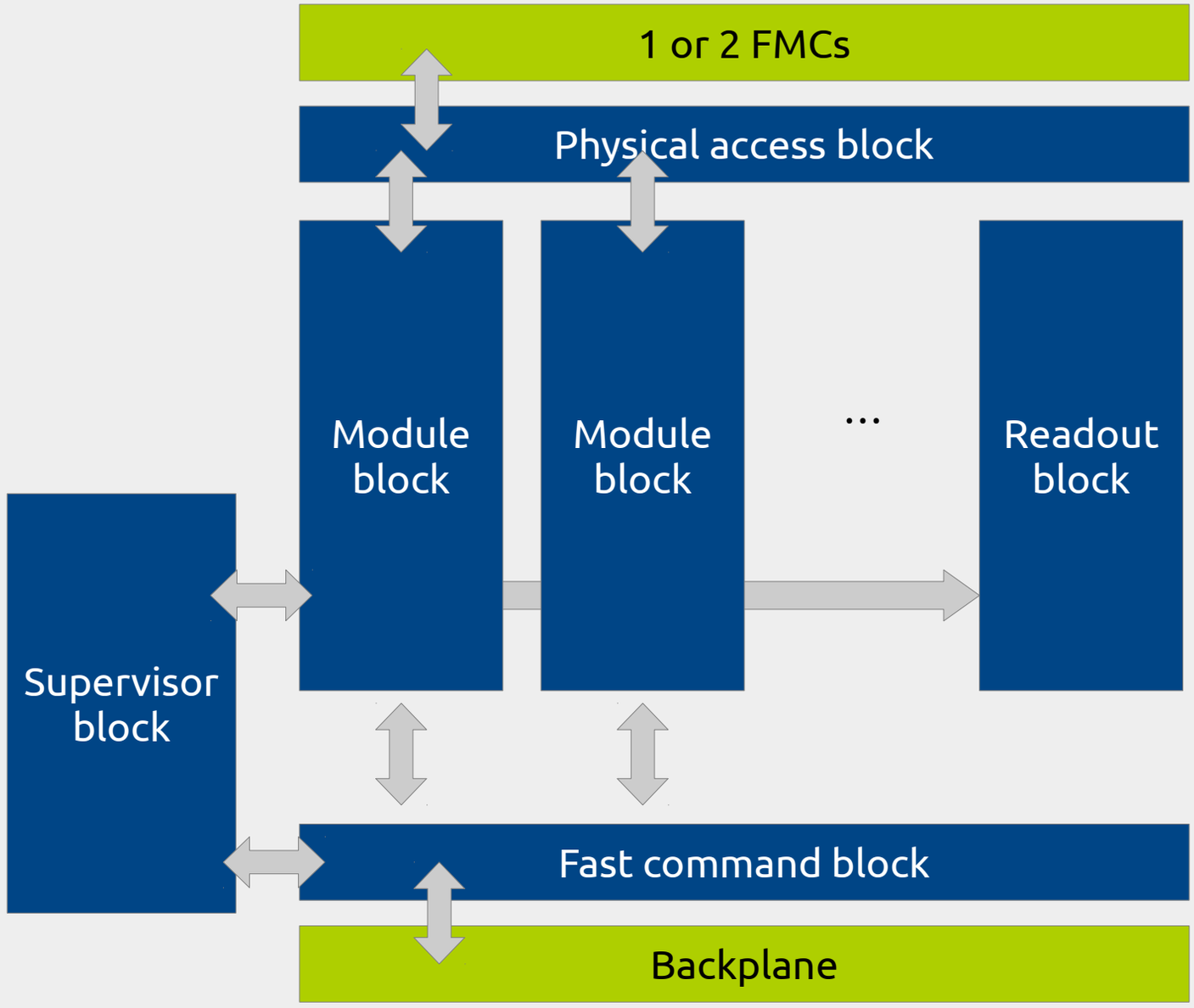
Firmware modular design

- High-level blocks
  - well defined functions (see next slides) and interfaces
  - switch between compatible blocks via compilation
- Block implementation
  - assigned to a developer or group
  - should fit the assigned resource use
- All code base included in the same repository
  Proposal: gitLab – advantages are
  - well defined code integration work-flow
  - code review prior to inclusion in official release
  - can run test benches automatically when new code is pushed

Organizational aspects

- The idea was circulated and found positive feedback
- Interest in contributing expressed by 9 institutes
  - Boston, CERN, DESY, FNAL, IC, IPHC, NUST, Rutgers, ULB/VUB
  - Groups with interest in both PS and 2S modules are present
  - Still need to strengthen expertise or gather enough documentation on: GBT links, CIC data format, hybrid connections
- A mailing list was formed
  - cms-tracker-phase2-testboard, 27 members (some potential developers, most interested observers)
- Time-line:
  - A series of meetings during November to define architecture & interfaces and assign block development (next one on Nov 7th)
  - Allowing development to start before Christmas break

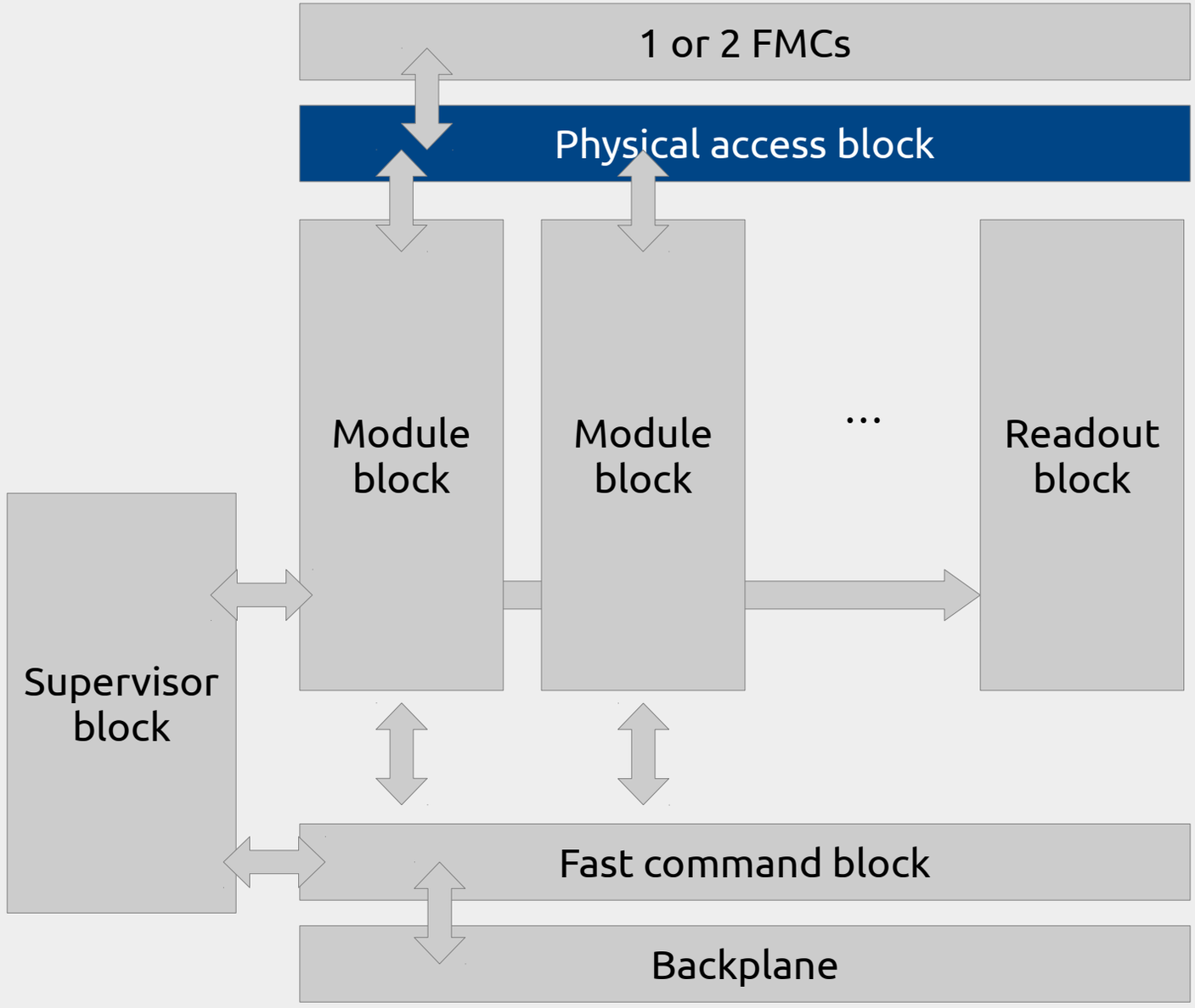
Firmware design proposal



Top level

- Blocks to be implemented in the user code
- Relies on the standard FC7 system firmware
  - IPbus
  - Memory access
  - …

Firmware design proposal

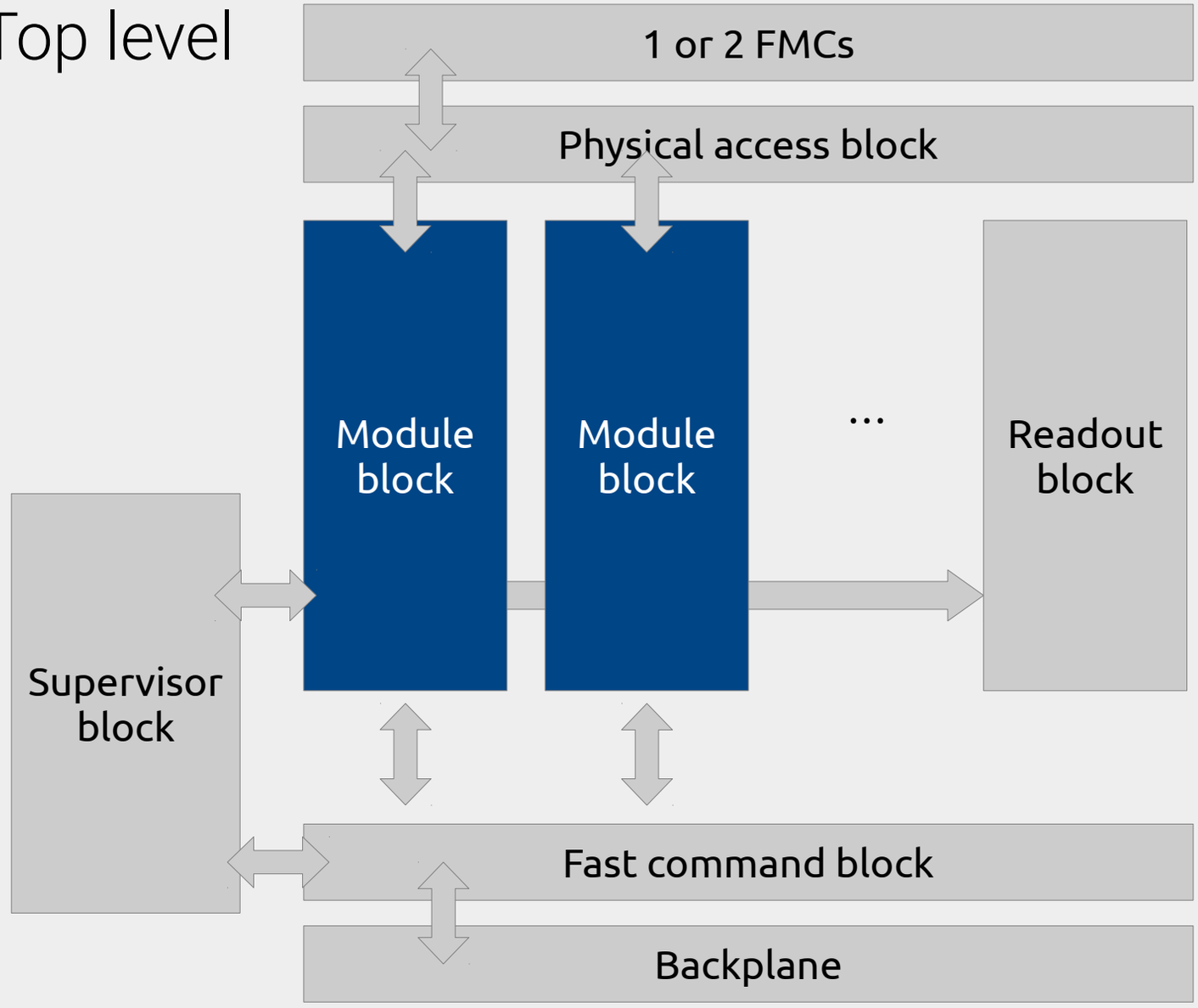


Physical access block

- Two working modes:
  - Optical: wrapping around N×GBT-FPGA blocks → SFP+ FMC
  - Electrical → interface FMC (must implement I²C master)
- Presents the same interface to the module block
  - Module does not know in which mode the system is working
  - The only difference is the number of available links
- Detects FE power-on and sends command to module

Firmware design proposal

Top level



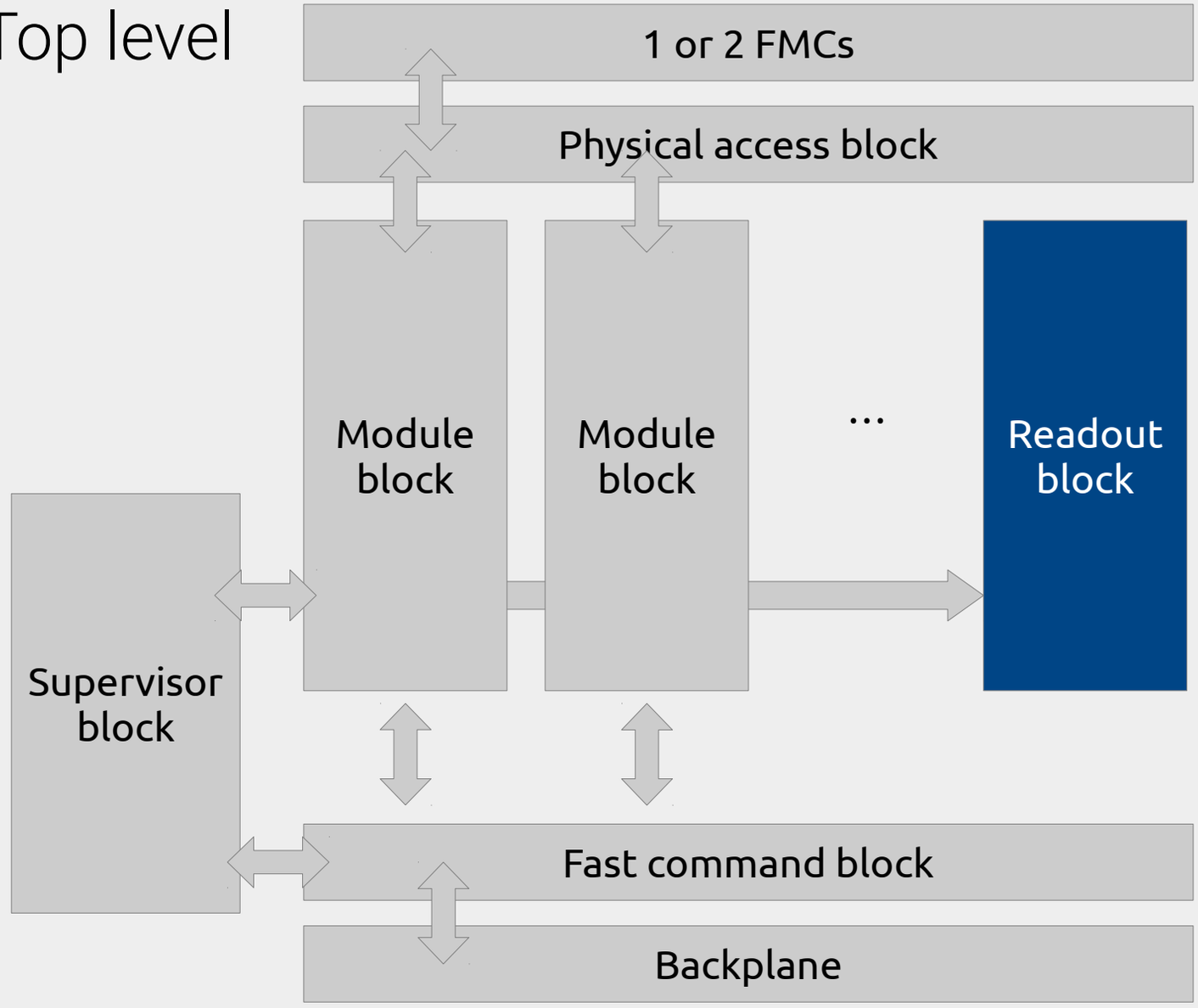
Module block

- Transmits slow and fast commands
- Receives and unpacks data
- Syncs trigger and stub data
- Can forward stub detection to fast command block
- Sends data to the readout block
- Can pre-process hits data for calibration (hit or stub counting)

(more details further on)

Firmware design proposal

Top level

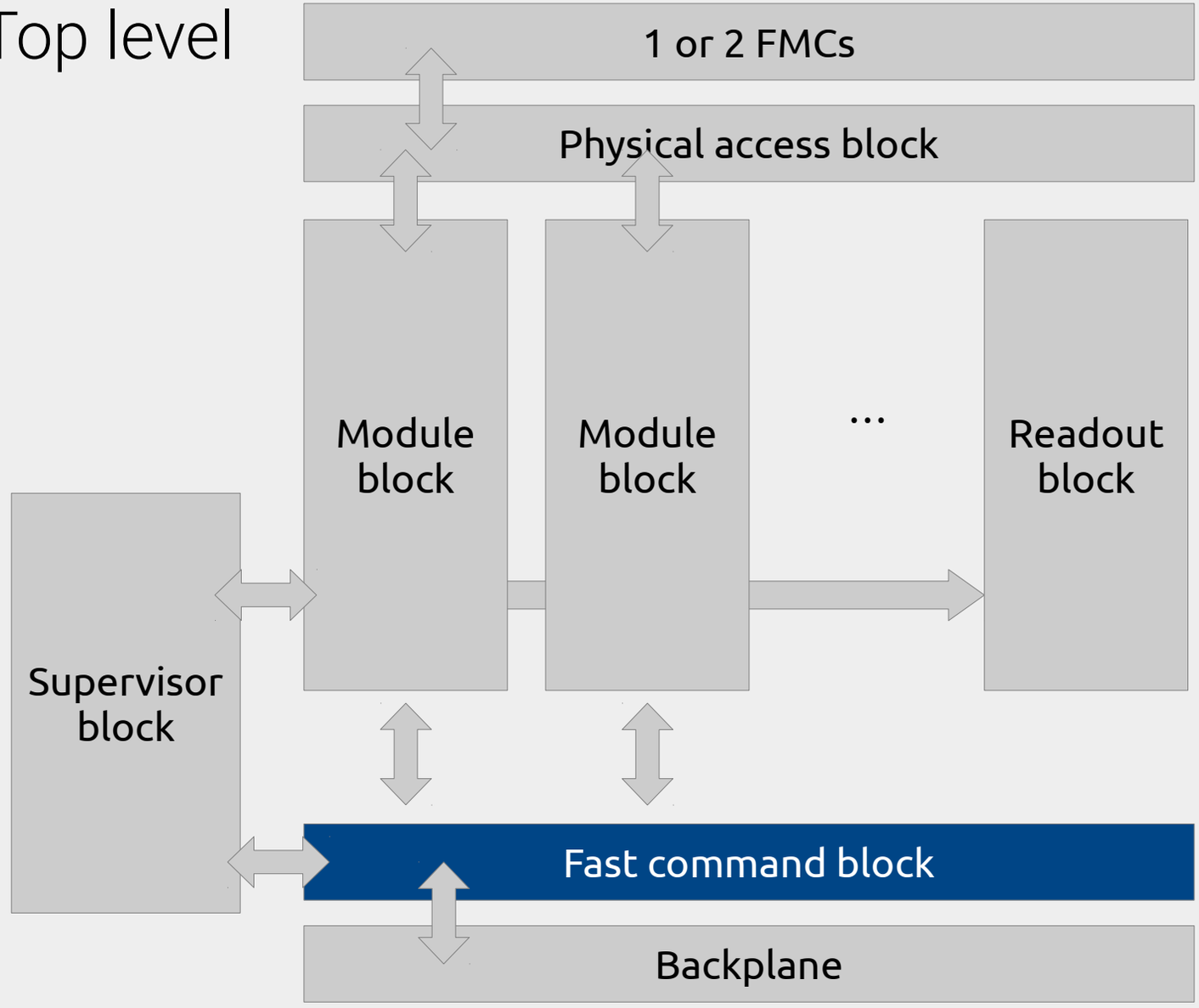


Readout block

- Data mode:
  - Manages memory readout and back-pressure
  - Optionally could stream data to SFP+ connectors on the FMC
    (iif optical readout is present, one connector can be dedicated to fast streams)
  - Data formatting for the DAQ, according to payload specs
  - Performs post-scaling (to exercise fast trigger rates with limited bandwidth)
- Count mode:
  - Used for commissioning: holds hit/cluster/stub counts *per strip*
  - Takes care to add counters properly – as many counters as conditions (e.g. thresholds)

Firmware design proposal

Top level

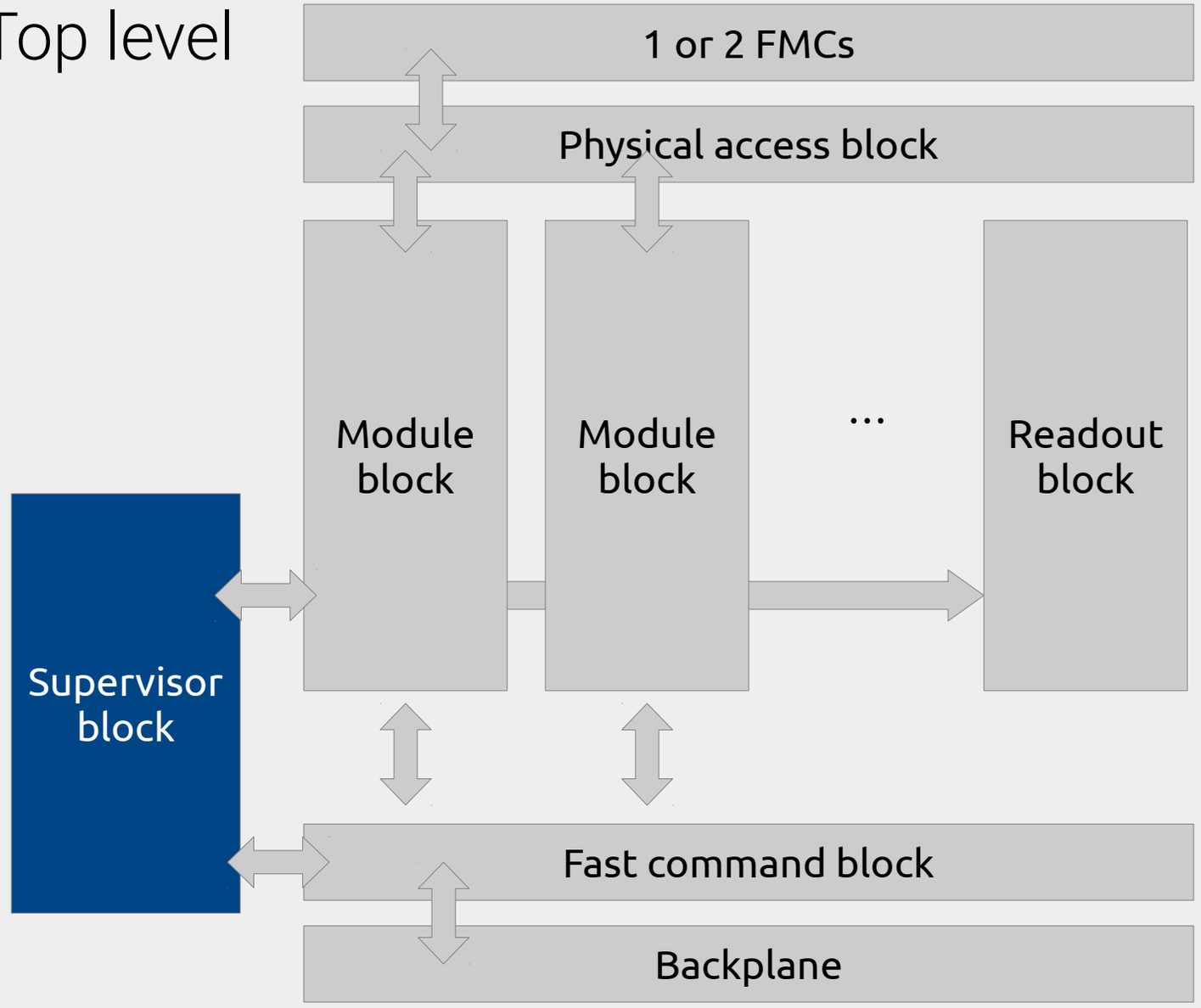


Fast command block

- Recovers clock and trigger from backplane
- Can generate clock and periodic triggers
- Holds a trigger counter: it can be programmed to accept next N triggers and then hold the next incoming triggers
- Can issue a trigger upon reception of a stub from modules

Firmware design proposal

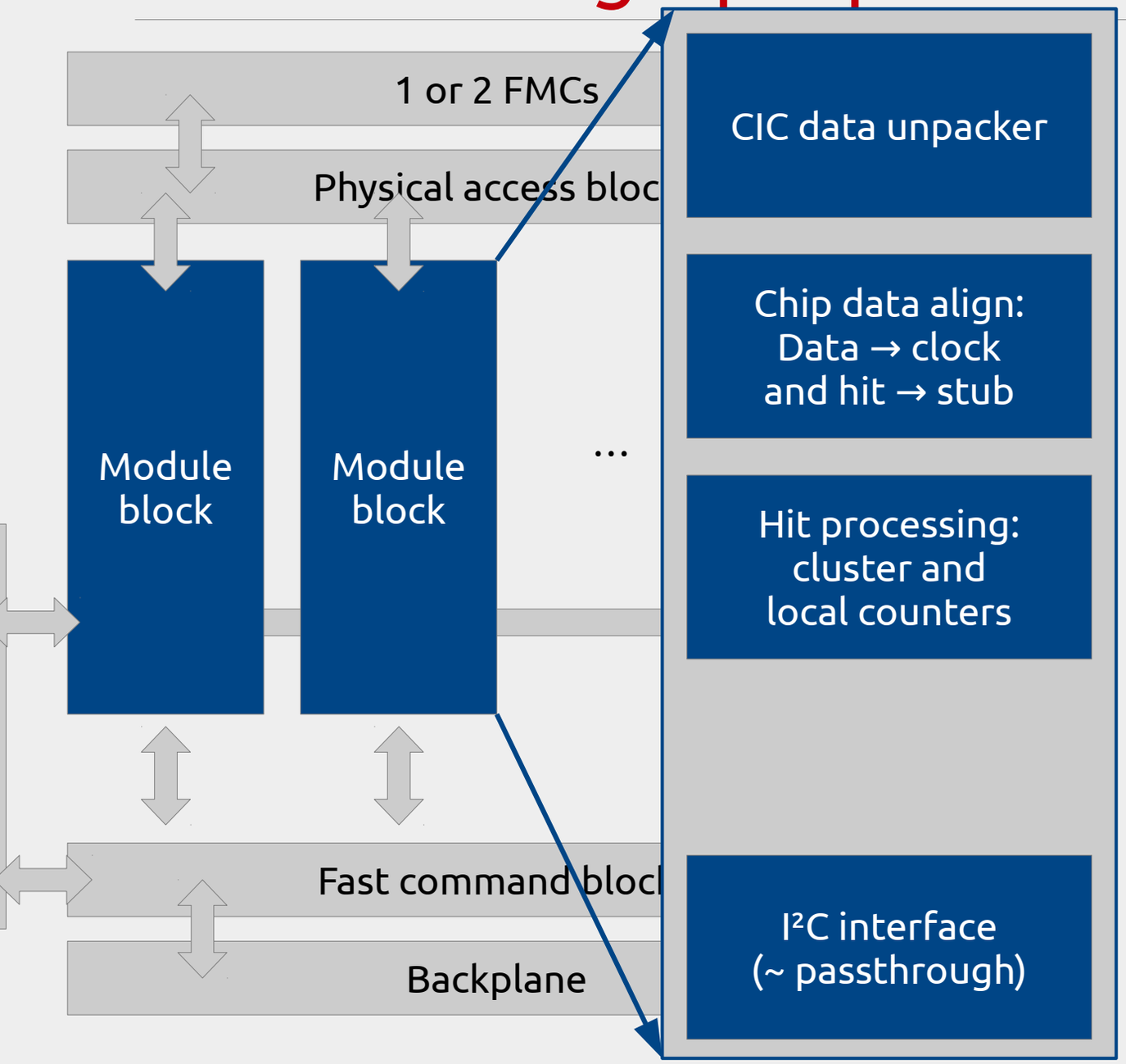
Top level



Supervisor block

- Holds a local stack of operations to be executed sequentially:
  - I²C transactions for F.E. (via module)
  - Generate (or accept next) N triggers
  - Push hit counters to appropriate memory location
- Significant reduction of IPBus transactions needed for a calibration operation (à la pixel DTB)

Firmware design proposal



Module block (detailed)

- **CIC unpacker**: (if needed) splits stub & hits streams, otherwise ~ passthrough
- **Chip data align**
- **Hit processing**:
  - **Hit count** can be activated to count hits (or clusters or stub) per strip locally
  - Always sends stub to trigger with fixed latency
  - When requested, queues data to readout block

Some links

- Mailing list cms-tracker-phase2-testboard
  https://e-groups.cern.ch/e-groups/EgroupsSubscription.do?egroupName=cms-tracker-phase2-testboard

- Shared document to collect information
  https://www.authorea.com/users/95964/articles/133719/_show_article