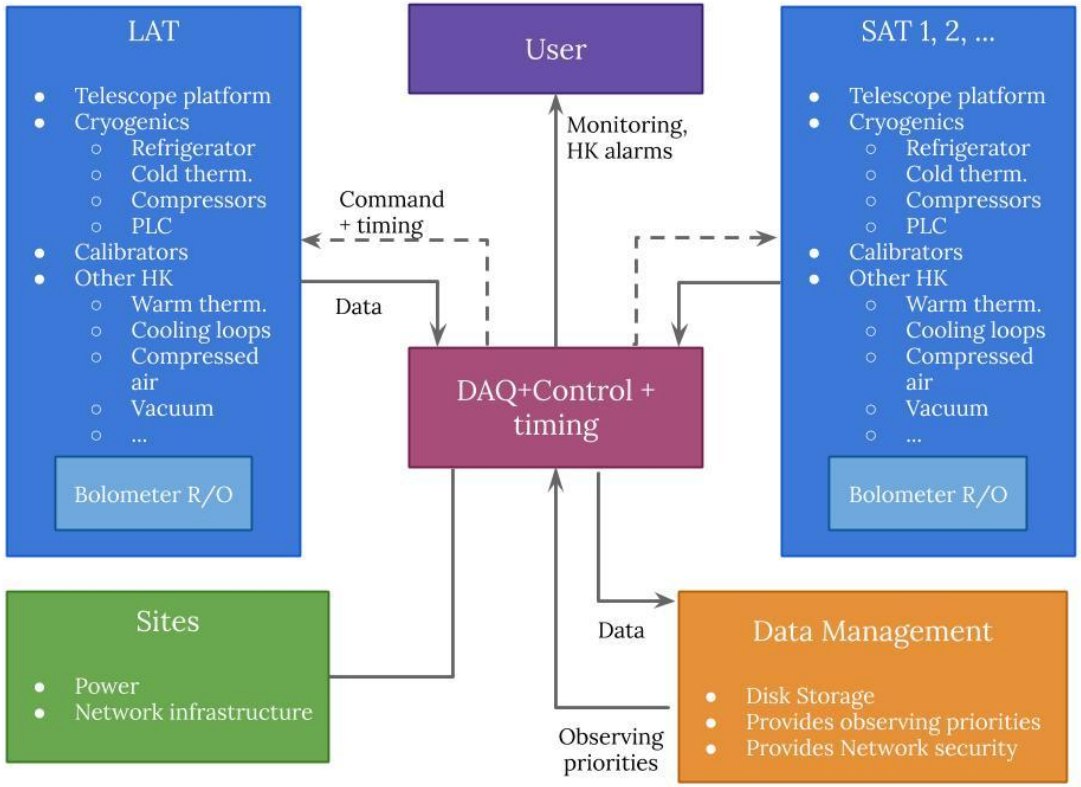
DAQ PBDR/Near-term

WBS 1.08.03

Nathan Whitehorn
Michigan State

DAQ Private Network Design

- Commodity 10 Gbit/s Ethernet trunks forming **private-fiber DAQ-only** network
- Synchronous Ethernet + PTP for timing (details later)
- IP as universal protocol
- On order 100-200 nodes per broadcast domain (dominated by estimated number of readout boards on LAT)
- Segregated broadcast domains for high-speed and low-speed DAQ



Technical Design Goals: DAQ

Key goals:

- Leverage the existing stage-3 DAQ designs where applicable
  - **SO and SPT-3G designs show no architectural obstacles to scaling to CMB-S4 requirements**
  - Community familiarity
- Portability/minimal setup requirements
  - **Allows use in university and national-lab detector testing labs from the beginning**
    - Prevents second-system syndrome, shaking out bugs early
    - Limits duplicated development costs for “lab” and “real” DAQ
    - Key feature in success of SPT-3G deployment
  - Lowers deployment risk
- Use COTS hardware and IP-based digital signalling everywhere: lowers cost, parts and schedule risk
- Use well-known (in community) languages, shared with DM: C++ and Python

Moving from stage-3 to stage-4: Detector Readout DAQ

Problem scope:

- Data rate increasing from ~20k detectors per site to 300k: order-of-magnitude in data rate
- Readout technology in flux from fMux (3G) and μMux (SO) to as-yet-undesigned TDM hardware

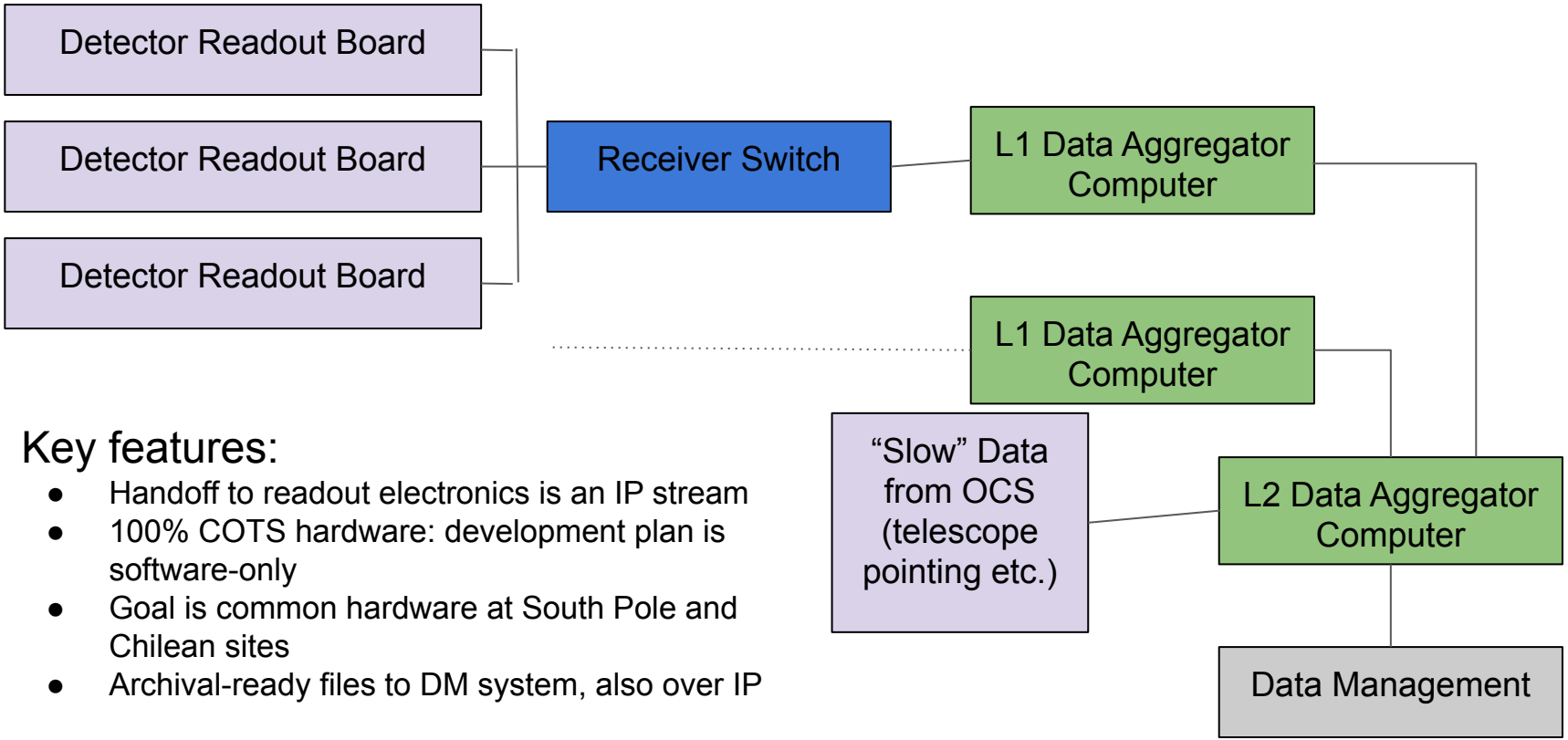
Reusability:

- Order-of-magnitude increase in data rate and new readout electronics mean stage-3 code is probably not useful.
- Architecture seems scalable however; synthetic testing of order-of-magnitude higher sampling rates on SPT-3G readout meet S4 throughput requirements

Summary:

- Stage-3 DAQ systems **demonstrate required throughput**, can serve as an architectural and resource-use guide for S4
- Limited applicability of stage-3 DAQ code and hardware for detector readout
- Substantial development effort required, but **little risk of unknown problems**

Technical Design Overview: DAQ



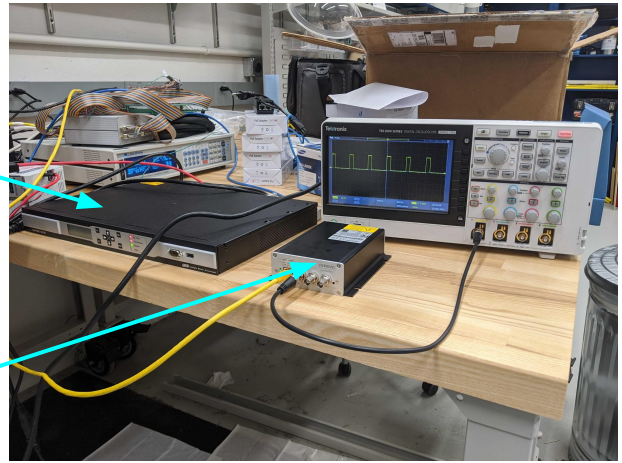
Key features:

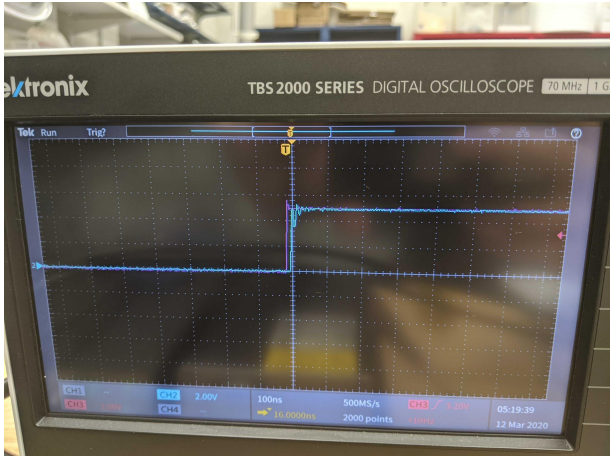
- Handoff to readout electronics is an IP stream
- 100% COTS hardware: development plan is software-only
- Goal is common hardware at South Pole and Chilean sites
- Archival-ready files to DM system, also over IP

Technical Design Overview: Timing System

Meinberg Grandmaster GPS Time card (outputs PTP, IRIG, 10MHz, PPS, … highly configurable

Boundary clock: PTP timing to IRIG, 10MHz, PPS (configurable)



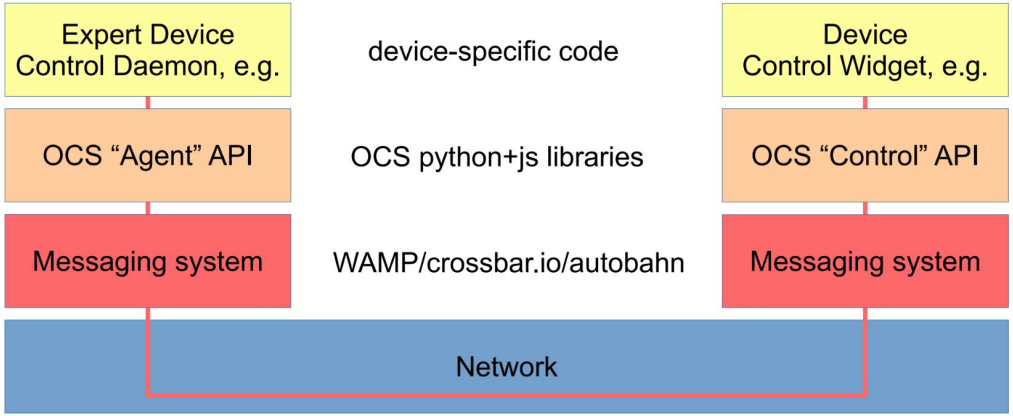


Boundary clock and Grandmaster aligned to 16ns (fluctuates within 100ns spec)

Key features:

- Low- and high-level timing signals: support electronics of varying complexity
- **Single timing domain per site**
- Meets requirements from known readout: Following design from SPT-3G (10 MHz + IRIG) and SO (near clone of this + PTP), limiting risk
- Assumption is that we will need to support asynchronous data (at minimum between housekeeping objects), and thus time stamps will be required for each data field

Technical Design: Low-Rate DAQ



Key features:

- Diverse collection of low-rate, low-data-volume sensors (structure thermometers, pressure gauges, power supply voltages, etc.)
- Handoff point in software
- Uses same OCS system as control, pushing small amounts of data through pub/sub system
- Design follows SO, substantial code reuse
- Designed for in-lab deployment as well as at-site
- **Low barrier-to-entry agent production**

Choices and Planning

Interaction with Readout:

- Need to have a defined interface basically ~ now (plan to have prototype readout boards in ~ 6 months)
- Do we want to support MCEs?
- How do we ensure that we get an emulator for test design?

Support for labs:

- Need to start defining a hardware/software package for use (post-downselect) in test labs
- Integration testing / support process??

Integration test stand:

- Building at MSU (contact for account, still some hoops to jump through with university IT and firewalls)
- Software development testbed for multiple users
- Full system: switch, clock,
- Can plug in your widget locally or (post-COVID) happy to host people for plug-fest testing