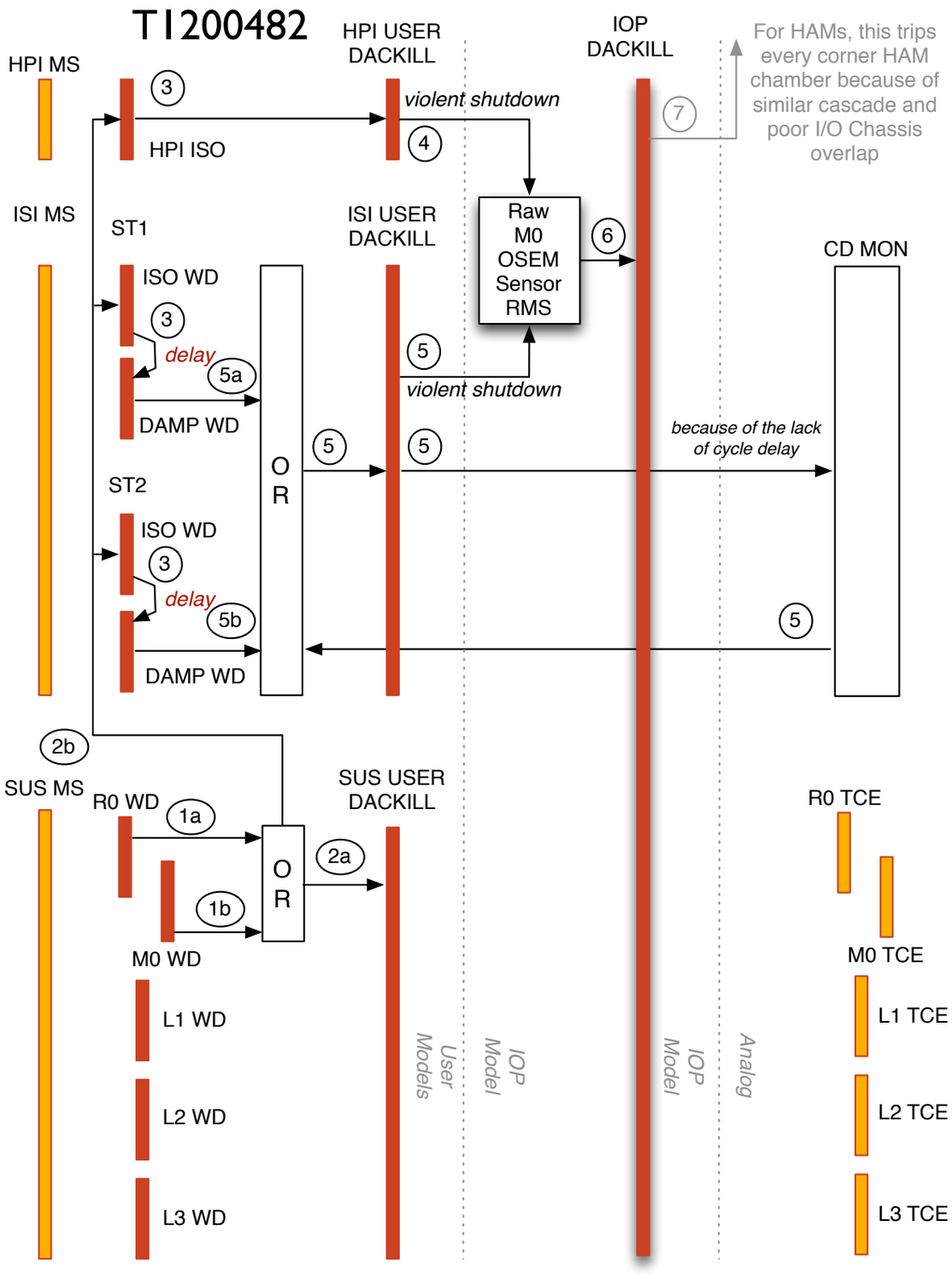
Software Watchdogs for SUS + ISI + HEPI

Brian Lantz, Jeff Kissel, for SEI, SUS, and CDS
Nov. 6, 2013, G1301210-v1

- Current Watchdogs are amalgamation of ideas which each seemed good at the time.
- Lots of them, interlocked with software triggers, jolts to common hardware, machines which control multiple chambers.
- Everyone is irritated with current 'system' because one unplugged OSEM can take down all HAM ISIs, HAM HEPIs and HAM suspensions in the corner station.
- Need a more integrated design. SUS user watchdog involves: ISC drives, SUS feedback loop, and SEI platform control.
- We suggest immediate actions and near-term actions.
- We propose an idea about SUS User Watchdog.

Watchdog Cascade

Skip most detail on this slide unless there are questions



To stop the cascade:

1) Have the IOP DAQKILL only kill the relevant DAC cards, not all cards in the chassis.

2) Only kill HEPI with slow DAQKILL, not fast SUS WD. This should avoid most trips so we can keep alignments.

Suggest Immediate Changes

Suggested Phasing of the changes list in the wiki:
https://awiki.ligo-wa.caltech.edu/aLIGO/SeiSusWatchDogCommissioning

- HEPI
  - Put the checker script functions into the HEPI frontend
  - Detach the SUS WD from HEPI User Watchdog and rely on (existing, or soon new) IOP WD
  - Install 4 state User Watchdog (like ISI)
    - run, rampdown isolation, hold bias only, full shutdown
      - No functional change, yet, becuase there is no separate 'hold bias' function. It enables a later change to let us hold offsets when the isolation feedback is disabled.
- ISI
  - Fix several bugs in User WD code:
    (Delay in coil-driver trip, tie resets together, remove redundant CPS level from USER-DACKILL, put FE rate into code.)
  - Modify ISI User Watchdog to allow 'several' saturations before tripping the watchdog. 'several' is 4-10 cycles (1 - 2.5 msec.)

Suggest Immediate Changes

- SUS
  - modify SUS user WD inputs:
    - Continue to trip on OSEM Band Limited RMS ,
    - Disconnect the DC level of OSEM (biased OSEM not dangerous, but many OSEMs are near the edge of range for alignment reasons).
    - Disconnect the RMS of the Actuators - actuators are too weak to cause damage, except driven oscillations - which OSEMs will see. Often driven to saturation by ISC - this is not a hazard per-se, and failure to catch IFO lock shouldn't trip SUS WD.
  - Remove the DC path from the SUS IOP Watchdog (but leave it as an indicator of unplugged or broken OSEMs)
  - Display the PUM driver's independent hardware watchdog (*that is already present*) on the QUAD overview screen and make reset button simpler to find.

Suggest Near Term Changes

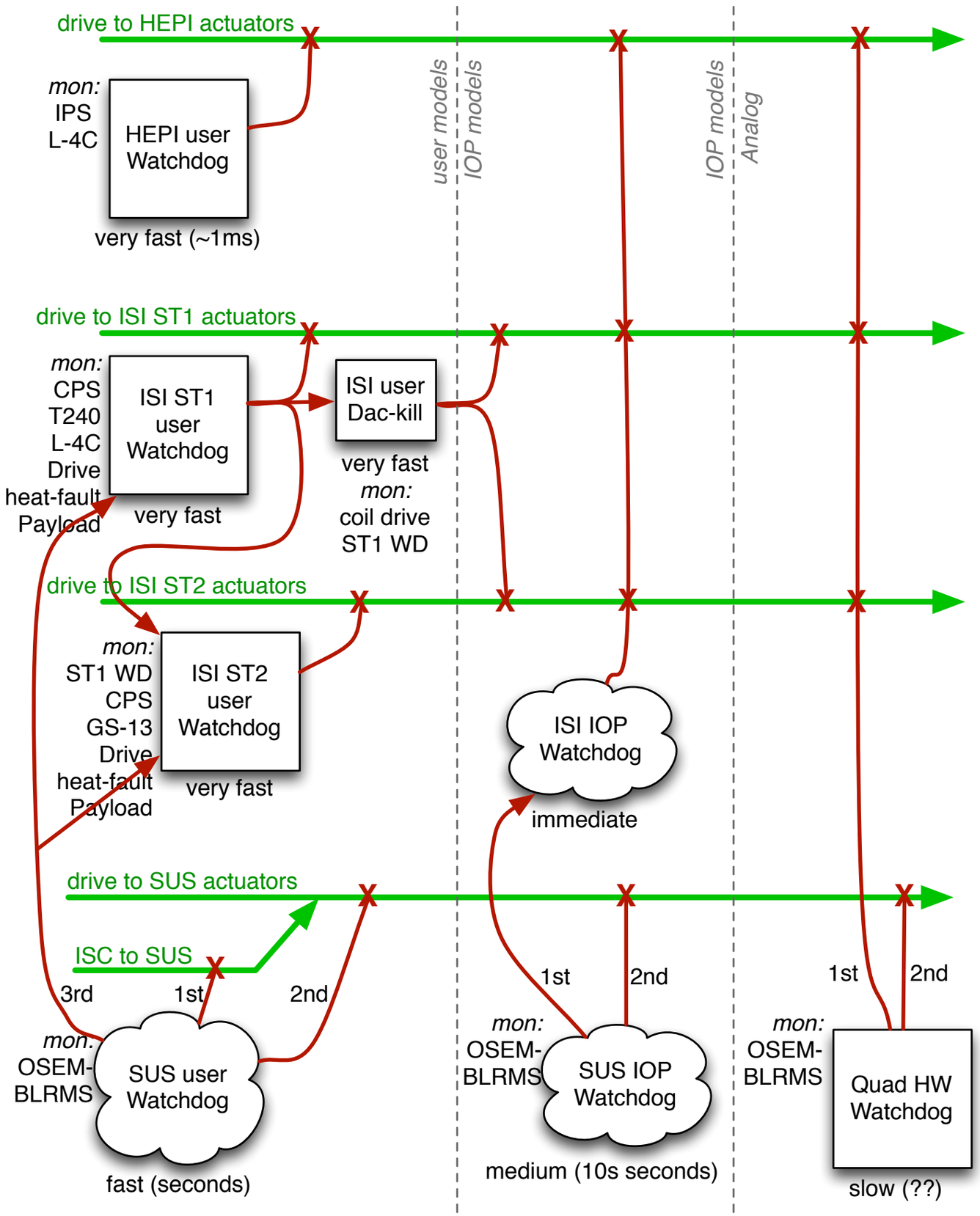
- System
  - Install new IOP Watchdog with new Daq-kill parts

- HEPI
  - Install Saturable Low-Frequency Integrators into drive chain which will hold the DC part of the drive signal even after the servo is disabled. This new features needs testing at LASTI.

New Daq-kill part (summary)

- Put a new part into the SUS - IOP model
- It monitors the SUS OSEMs, if the BLRMS is 'too big' and stays that way 'for long enough' then it sends a signal to the SEI IOP and disables the ISI and HPI drives (using a similar part).
- If the SUS OSEM signal stays 'too big' 'for a while longer' then it also disables the SUS IOP drives.
- Need to define 'too big' and 'long enough', 'for a while longer' in a smart way. Probably several minutes (Jeff?)

Suggested Interaction



- Simplify Interactions
- Suggest New SUS Watchdog to sequentially disable:
  1) ISC input to SUS
  2) Feedback from SUS
  3) ISI
  4) maybe HEPI?
- Quads get hardware watchdog
- Do we remove the IOP WD for quads as redundant?

Actuator signals in Green
Watchdog signals in Red

Details of new DK part

DK-IOP Implementation (pg 1) - *Copied from Rolf's original email without consult - details may have evolved!*
1) Allow multiple of these parts within an IOP model. In the case of the h1iopsush34 model, it would allow one for connection of HAM3 and another for connection of HAM4 suspension watchdogs. For h1iopseih23, one would have input from SUS HAM3 and one with input from SUS HAM2.

2) Allow DAC module definition within the Description field of the part to indicate which DAC modules are to be disabled (zero all outputs) on a fault. Examples:
- card_num=0, where card_num is the DAC module number. In this example, a DK trip will only zero out the outputs of the first DAC module. - card_num=1,2,3: Multiple DAC cards affected by trip.

3) Add a new timer input, similar to present Bypass Time input, specifying time that a fault indication at the Sig input must persist before the DK part takes action:
- Until first timeout, DK takes no action. Removal of fault indication at Sig input will reset timer.
-After first timeout:
- WD output of DK part would would go to zero (Fault), but outputs to DAC modules continue to operate as normal. In the case of a SUS to SEI IOP, this output is sent via an IPC, and would cause SEI to shutdown, but not shutdown drive signals to SUS.
Second timer is started if still fault at Sig input.
- Removal of fault indication, prior to timeout, at Sig input will reset timer. - Removal of fault indication at Sig input will NOT clear the WD output fault indication.
- After second timeout, outputs of assigned DAC modules would all go to zero. Essentially, if whatever action was to be taken by the WD output connection did not fix the problem, go ahead and shutdown the DAC modules.
- Neither the WD output nor the DAC trip will be cleared if fault indication at Sig input is removed.
- Once DK faults have been latched, a DK RESET must be issued to clear the fault(s).

Details of new DK part

DK-IOP Implementation (pg 2)

4) Modify the DK STATE word returned to EPICS.

- Present STATE definition:
  0 = TRIPPED or PANIC mode 1 = OK 2 = Bypass Mode
- New STATE definition (Bit pattern, where 1 in each bit indicates OK/Normal):
  Bit 0 = WD output from DK part (0= FAULT, 1 = OK) Bit 1 = DAC module trip ((0= FAULT, 1 = OK) Bit 2 = Bypass mode set (0 = Bypass Set (Not normal), 1 = Bypass not set (normal) Bit 3 = Fault timer running (due to fault at Sig input) (0 = running, 1 = stopped (normal))

5) Optional: Add EPICS output record to show present value to fault timer (in seconds, down counter)

- Could just send value to present Bypass timer reading channel.