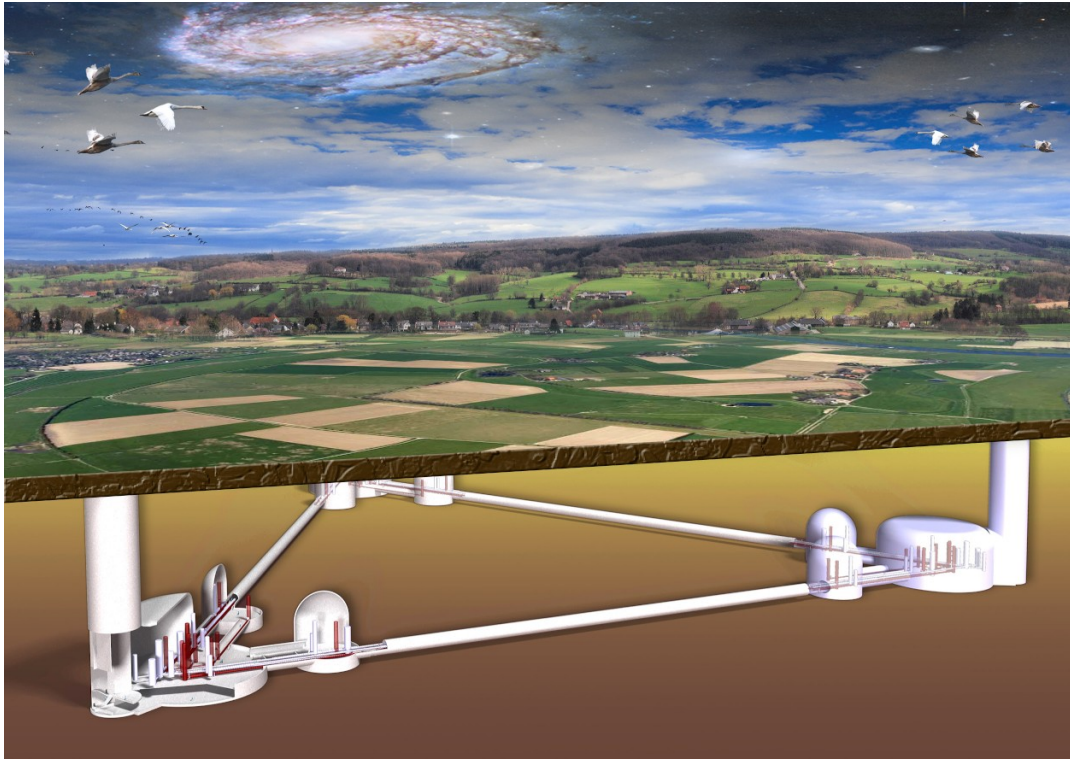


Online computing for ET



Bas Swinkels 

Loïc Rolland 

for the 'Data acquisition and real time control' work package

'Interferometer Division' of the 'ET Instrument Science Board'



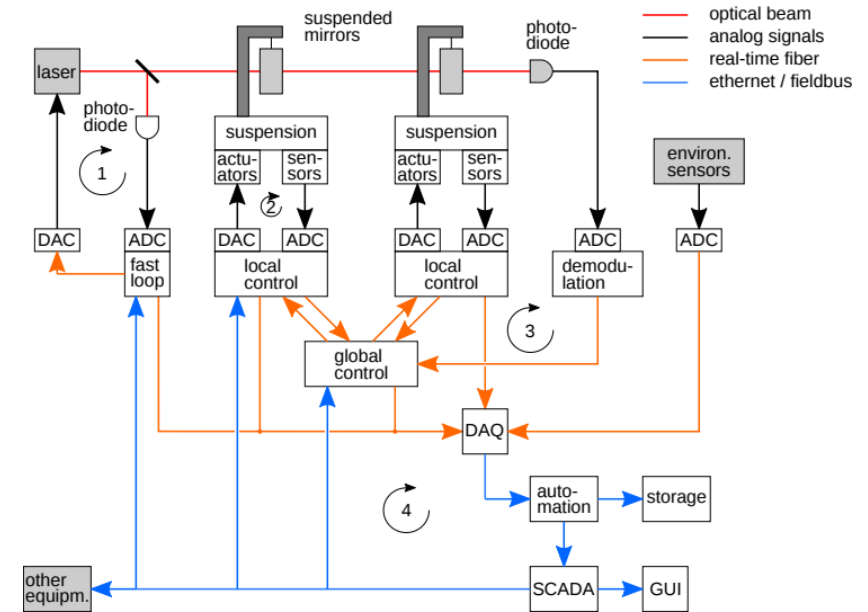
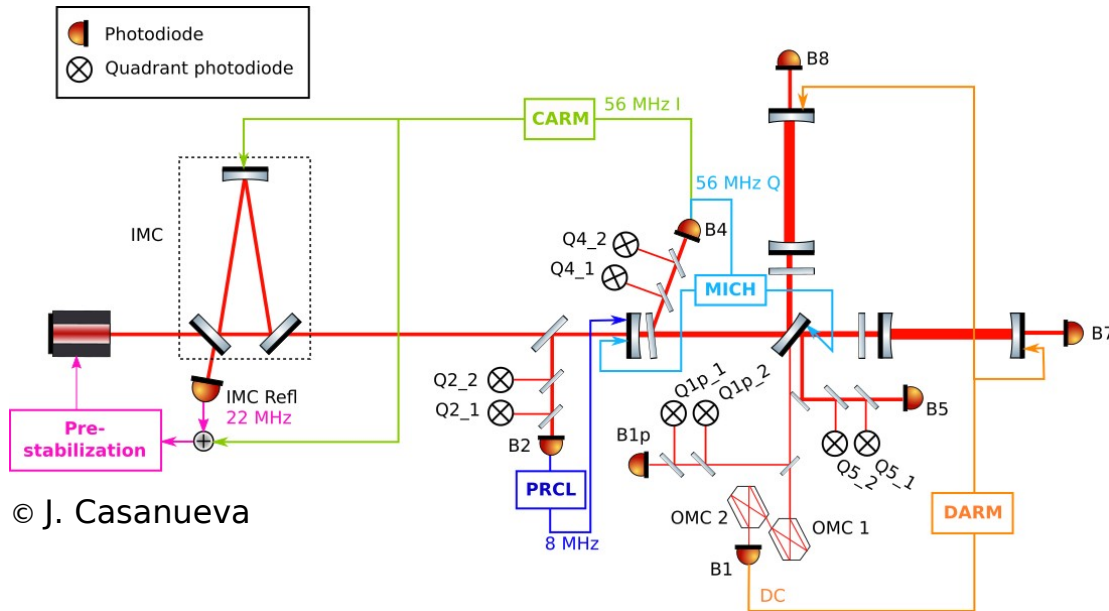
EIB kick-off meeting
30/11/2021

Outline

- Overview of various types of online computing needed to control and operate a GW interferometer
- Explain Data Acquisition (DAQ) chain
- First attempt at defining what is in scope for DAQ/control WP and what for the EIB
- Disclaimer: I am not an expert on computing or data analysis, but have interacted with all software needed to run the Virgo interferometer



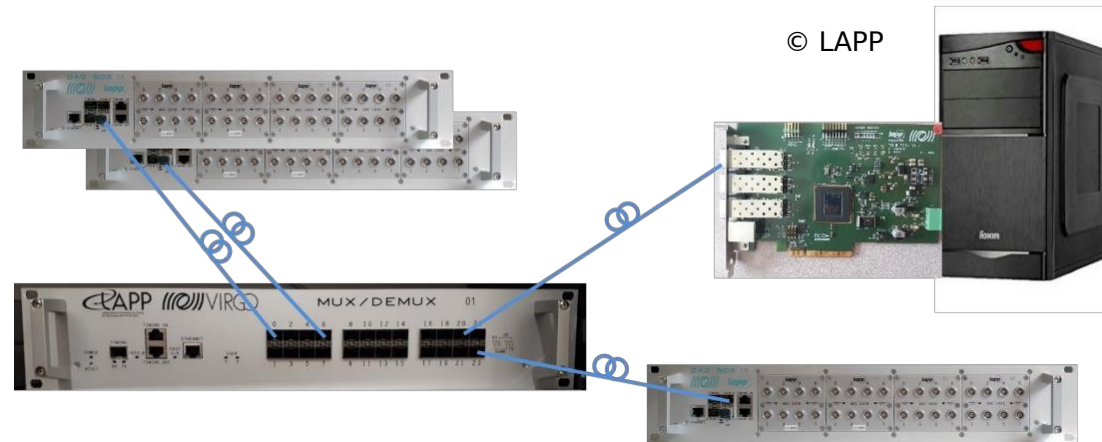
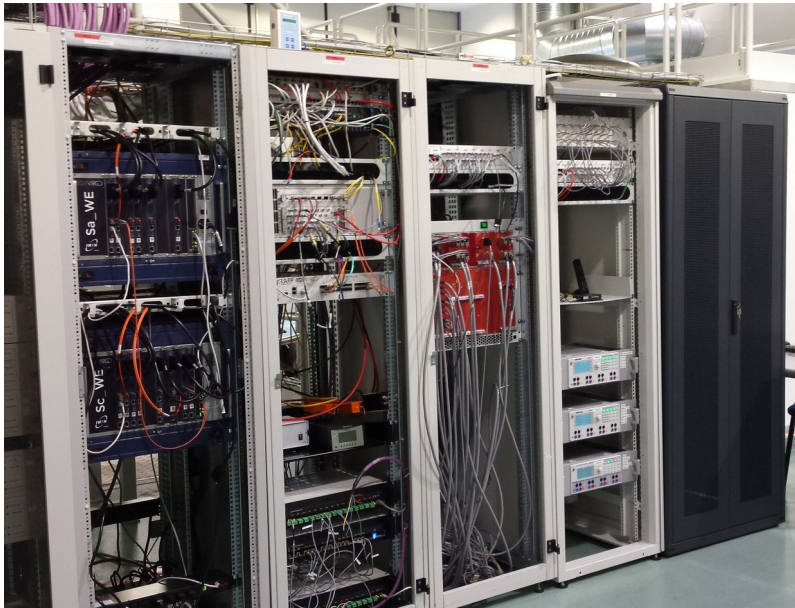
Instrument control



- Control system is an integral part of a GW interferometer: keep mirrors quiet and cavities on resonance
- Various levels of control:
 - 1) very fast analog/digital loops (\sim MHz)
 - 2) fast local control of suspensions (\sim 10 kHz)
 - 3) fast global control of whole interferometer (\sim 10 kHz)
 - 4) slow automation: lock acquisition (\sim 1 Hz)
 - 5) 'human-in-the-loop' monitoring and operation (minutes)
- Hard real time, distributed, hierarchical control
- All signals from control system and environmental monitoring recorded by data-acquisition (DAQ) chain



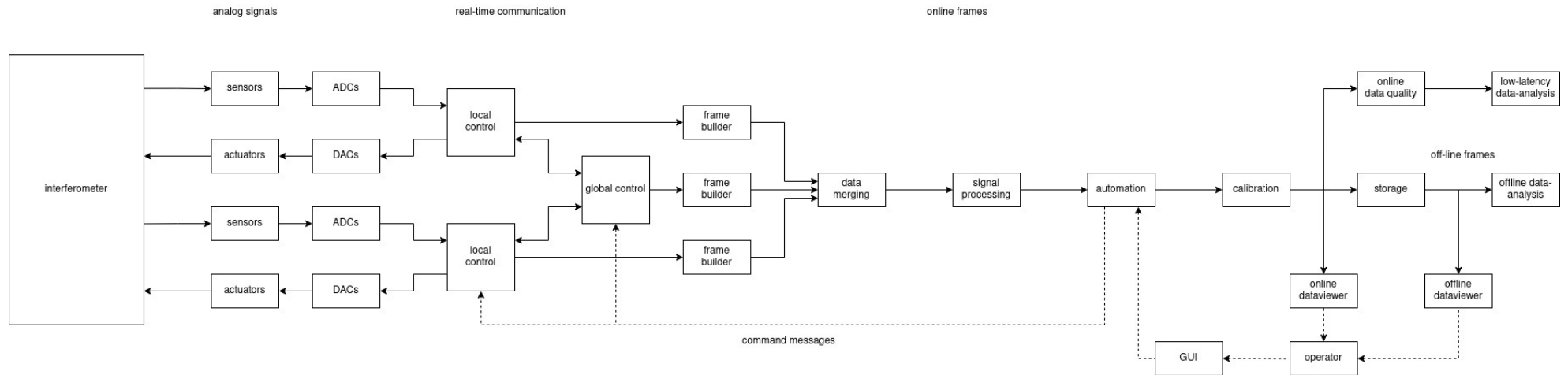
Control hardware



- Historic progression of control electronics: analog -> barely working custom digital -> comfortable custom digital -> off-the-shelf
- Current generation: hard real-time digital control consisting of ADCs, DACs, DAQ-boxes and controllers (real-time Linux PCs, DSPs). Typical sample frequency 10-20 kHz, control algorithm programmed by end-user in Simulink or similar
- Distributed system, sensors and actuators separated by kilometers: need real-time fiber communication
LIGO uses 'Reflected Memory' by Dolphin, Virgo uses home built fiber system (TOLM), maybe >10 Gbit Ethernet in future?
- Fastest loops are mostly analog, but some recent examples of digital loops at ~1 MHz
More flexible, might go completely digital in future



DAQ chain



- *'frame builders'* which collect the data of the fast real-time processes in 1-sec chunks of data
- merging of various data streams
- additional signal processing (decimation, image processing, ...)
- automation nodes
- forward frames to storage machines and low-latency pipelines
- provide data viewers, GUIs for human interaction with the interferometer
- data flux relatively modest compared to e.g. CERN, but we care about latency



Automation

The image shows a Python IDE with a script for a state machine. The code defines a class `PROCDERR_TUNING` with methods for state transitions and a `main` function. The state transition diagram shows the following flow:

```

    graph TD
      INIT((INIT)) --> DOWN((DOWN))
      DOWN --> GALVO_CLOSE((GALVO_CLOSE))
      GALVO_CLOSE --> REC_CLOSE((REC_CLOSE))
      REC_CLOSE --> IMC_RESTORED((IMC_RESTORED))
      IMC_RESTORED --> FMODERR_TUNING((FMODERR_TUNING))
      FMODERR_TUNING --> FMODERR_TUNED((FMODERR_TUNED))
      FMODERR_TUNED --> FMODERR_TUNING
  
```

The image shows the Virgo Online Control Room interface. It displays a hierarchical view of system components, including:

- Control Room:** TelescreenLeft, TelescreenRight, TelescreenDET, TelescreenArt.
- ADC7674:** EER_ADC7674, CEB_ADC7674_0, CEB_ADC7674_1, DET_ADC7674.
- Minitowers:** TIMING_dbox, SUSP_SBE, SIB2, SPRB, SDB and EDB, SDB2.
- Newton Noise:** NNC servers, Data Collection, Cameras, Fast Front End DAQ, Front End DAQ, Main Frame Builder, Alp Frame Builder, Storage Frame Builder, DAO to Spare, Storage - StoI01, StoI01 Input, Raw Computing.

A log window at the bottom shows system events and process status.

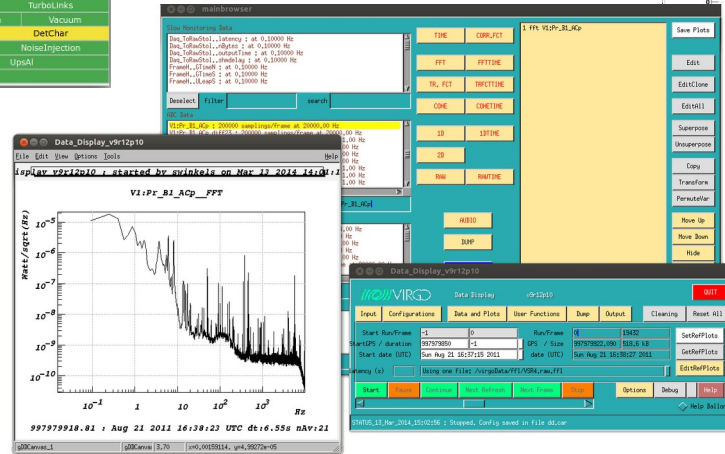
- Automation processes are embedded in the DAQ chain, so they have access to all data (with a latency of a few frames). Responsible for sequencing lock acquisition, various slow loops, basic safety checks
- Currently Python-based hierarchical set of state machines: Guardian/Metatron, see arXiv:1604.01456
- Needs a SCADA-like framework/communication protocol to change parameters of the fast processes (change gains/offsets/change filters, switch on loops): EPICS, TANGO, Cm, ...
- Process monitoring: Virgo Process Monitoring (VPM)



Monitoring



DMS		ITF Mode: Calibration (0x00 0x00 0x00)										ITF State: LOW_NOISE_2 (0x10 0x10 0x10)										UTC: 2017-08-29 13:41:02																										
Injection	IB_IP	IB_BRNCH	IB_BR	IB_Vert	IB_TE	IB_Guard	IB_Electr	MC_IP	MC_PAY	MC_BR	MC_Vert	MC_TE	MC_Guard	MC_Electr	Laser	LaserAmpli	LaserChoker	SL_TempController	RFC	MC_Power	PSTAB	IMC_AA	IMC_AA_GALV0	MC_F0_z	BPC	BPC_Electr																						
	PD	QPD_B1p	QPD_B2	QPD_B3	QPD_B4	OMC	PicoDisable	SDB1_IP	SDB1_LC	SDB1_BR	SDB1_Vert	SDB1_TE	SDB1_Guard	SDB1_Electr																																		
Detection																																																
ISC	B1p_DC					B4_5c_phi					SSFS_DiqNoise					UNLOCK																																
	B2_8_phi																																															
Suspensions	BS_IP	BS_F7	BS_PAY	BS_BR	BS_Vert	BS_TE	BS_Guard	BS_Electr	NL_IP	NL_F7	NL_PAY	NL_BR	NL_Vert	NL_TE	NL_Guard	NL_Electr	PR_IP	PR_F7	PR_PAY	PR_BR	PR_Vert	PR_TE	PR_Guard	PR_Electr	SR_IP	SR_F7	SR_PAY	SR_BR	SR_Vert	SR_TE	SR_Guard	SR_Electr	WL_IP	WL_F7	WL_PAY	WL_BR	WL_Vert	WL_TE	WL_Guard	WL_Electr	WE_IP	WE_F7	WE_PAY	WE_BR	WE_Vert	WE_TE	WE_Guard	WE_Electr
	CB_Hall	MC_Hall	TCS_zenes	NE_Hall	WE_Hall	WindActivity	SeisMon	INA_Area	DET_Area	EBRoom	External	DeadChannel	Lights	SeaActivity																																		
	ACS_CB_Hall	ACS_TB	ACS_DAQ_Room	ACS_EE_Room	ACS_MC	ACS_INJ	ACS_DET	UPS_TB	UPS_MC	UPS_NE	UPS_WF	Flatchannel	ExitChannel	ACS_WE																																		
	SDB2_SBE	SDB2_LC	SNEB_SBE	SNEB_LC	SWEB_SBE	SWEB_LC	SPRB_SBE	SPRB_LC	LargeValves	Clean_Air	TubeServers	TubePumps	TowerServers	MiniTowers	RemDrxPMP	Pressure	CompressedAir	CryoTrap	O2_Sensors	TankLN2	TurbLinks	Vacuum																										
VPM	DetectorEnvironment	ControlRoom	Minibowers	ISC	Injection	TCS	Suspension	Vacuum	DetectorMonitoring	DataCollection	Storage	DataAccess	Automation	DetChar																																		
DAQ-Computing	Latency	SoftwareAI	Diak	Timing	ADCs_Temperature	Daq_Boxes_Temperatures	NoiseInjection																																									
ITFOnCall																																																
Triggers																																																



- GUIs, data viewers for live monitoring of instrument status
- Web-pages with predefined plots for monitoring on scales of hours to days
- Supervising process to check that thousands of parameters are at their expected values, send alarm messages via mails/SMS when something breaks at night: Detector Monitoring System (DMS)



In scope for DAQ/control WP

- Exact boundaries TBD, but roughly “all specialized computing hardware and software that directly interfaces with the experimental hardware”: DAQ front ends, data collection, monitoring, data visualization
- Interface with hardware subsystems: monitor/provide analog and digital signals from sensors and actuators, provide computing for their control loops
- Interface with EIB: infrastructure (hardware, software, network), data storage
- Estimate total data flux from front ends to storage input
- Choose the hardware architecture for the real-time detector control and data acquisition:
 - general purpose front-ends with ADCs, DACs
(as far as they are not built into the sensors/actuators)
 - fast ADCs for digital demodulation
 - low phase noise timing distribution
 - real-time communication network (TOLM, reflective memory, ...)
 - fast computing for controls (real-time PC, DSP, FPGA, ...)
- Provide the software tools/architecture for the detector slow control
- Provide the software tools/architecture for the data acquisition pipeline



In scope for EIB

- Provide general purpose computing hardware for running the DAQ chain and slow controls
- Integrate the hardware for the real-time control in the general computing environment
- Provide hardware for control room and other needs of the commissioners (user management ...)
- Provide network hardware (fibers, patch-panels, ...) and management (DHCP, HTTP, ...) used by all experimental devices
- Install and maintain operating systems for DAQ system machines
- Long term storage, distribution and backup of data
- Design the data flow and architecture for online alert generation system
- All off-line computing



Grey areas

- Shared between EIB/DAQ
 - chose frameworks for low-latency frame distribution (Fd like)
 - chose protocol for inter-process communication (Cm/TANGO/EPICS like)
 - define file format for data exchange (gwf, hdf5, ...)
 - common package management?
- Done by other subsystems
 - provide space, power, cooling for underground control hardware (Infrastructure)
 - software for online data quality (Detector Characterization)
 - software for online/offline data analysis (Data analysis)
 - hardware/software for safety critical systems (laser interlocks, PLCs for vacuum system ...).
Best if this is provided by the corresponding subsystem, but all status data should be provided read-only to the DAQ system



Scaling up from Virgo to ET

- No major technological breakthroughs needed to control ET, could be evolution of current hardware
 - data flux might increase by order of magnitude: 4 TB/day now, but we will have multiple interferometers that are more complex
 - would like ADCs, DACs with a bit less noise
 - slightly faster digital loops (replace some more analog loops)
 - slightly better timing: lower phase noise
 - upgrade software to state-of-the-art
- Go more from custom built hardware and software to off-the-shelve where possible



the end...

