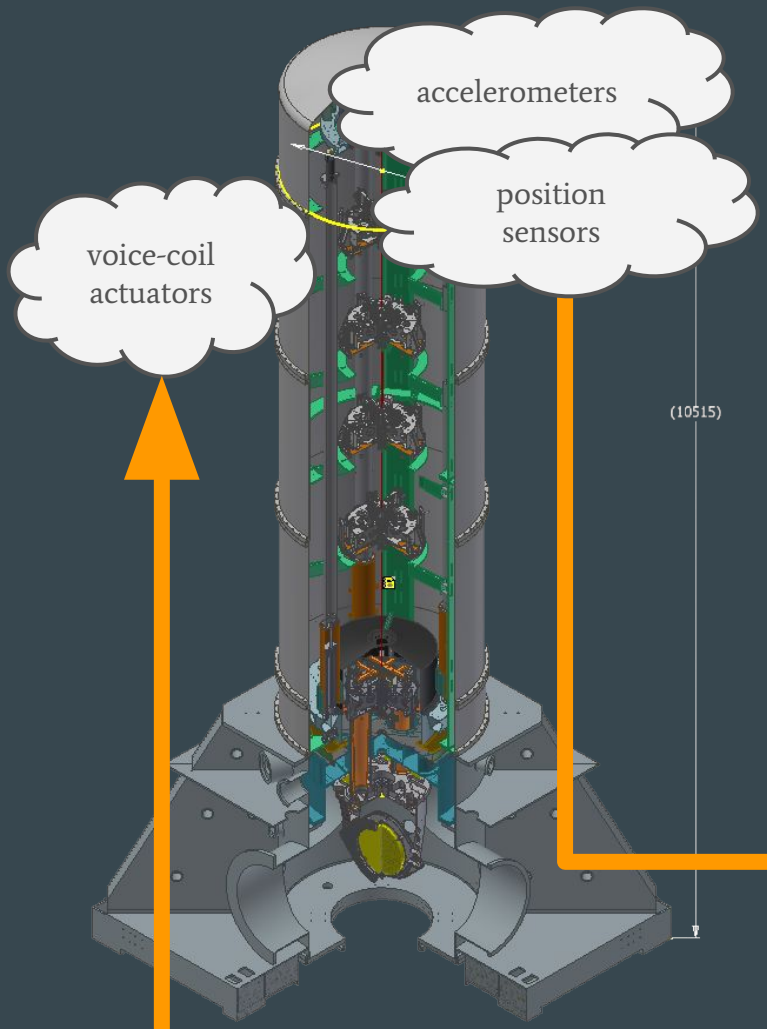
Software for the Suspension Control System of Advanced Virgo

...

Giovanni Cerretani
Università di Pisa & INFN Pisa

28/09/2016

The Suspension Control System



The Advanced Virgo mirror suspensions insulate the optical elements of the interferometer from the seismic noise. They consist of a five-stage pendulum.

The Control System allows to keep the suspension stable enough also in the low-frequency band, where the seismic noise is amplified by the resonances of the chain.

DSP-based processing

UDSPT Boards & co.

- 10 suspensions in AdV
- 2 crates per suspension
- 10 DSP boards per crate
- 1 µTCA controller per crate

>200 devices

Each UDSPT board is shipped with:

- 8-core DSP, up to 60 GFLOPS in double precision
- 512 MB of DDR3 RAM
- 2 Gbps Ethernet ports
- 1 Gbps optical fiber link to the Virgo DAQ

100 TFLOPS
in double precision

The Software Supervisor

A software supervisor is needed to control the large number of devices.

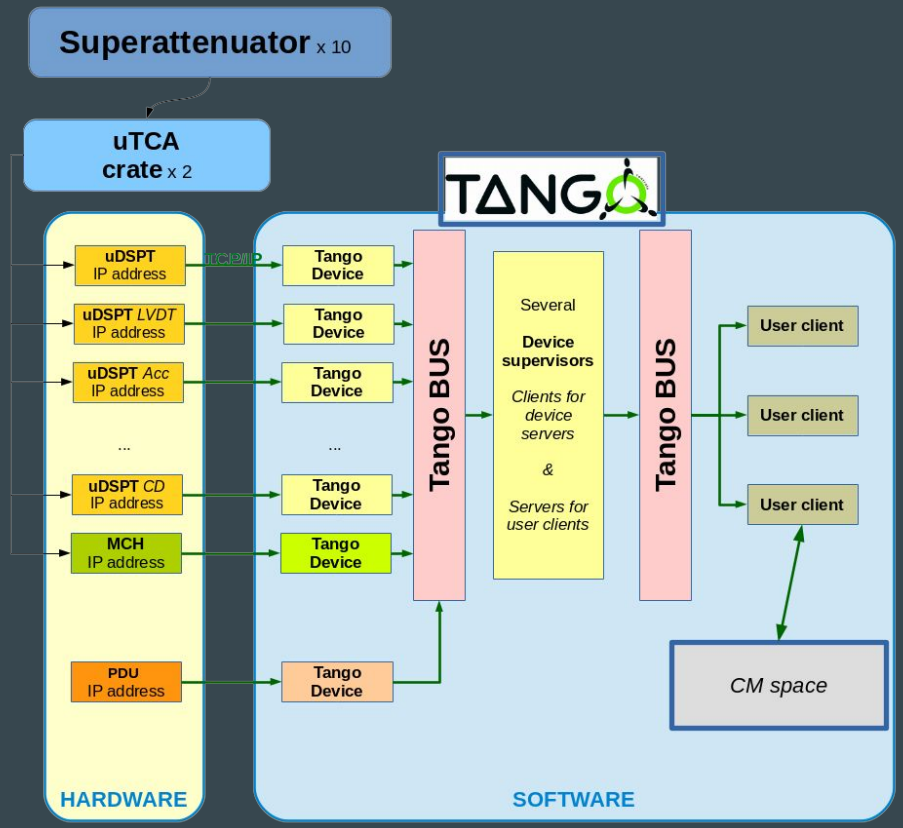
It has to:

- Provide the current state of the suspension control system, alerting the users if there are problems somewhere
- Provide commands to interact with the electronics, for maintenance and tasks that don't require hard-real-time performances
- Perform automatically some maintenance tasks
- Keep the history of the actions and of the states

TANGO is the toolkit chosen to develop the system. It is used by other Virgo subsystems and in over 20 physics experiments on the world.

A hierarchical system

TANGO is based on the concept of Devices. A Device is an instance of a class that describes its attributes and methods that acts as a server.



Device classes are coded in C/C++ and Python

In our architecture, Devices are organized in two levels:

1. A low-level TANGO Device for each physical device
   - UDSPT boards
   - µTCA controller,
   - ...
2. Devices that collects information of some low-level devices that are related
   - a crate, composed by the board plugged inside
   - a suspension, made by 2 crates

Clients

In TANGO, a clients is a piece of software that accesses to the services provided by the Device servers.

Graphical User Interface

Several GUI clients have been developed using **Python** and **Qt**, to

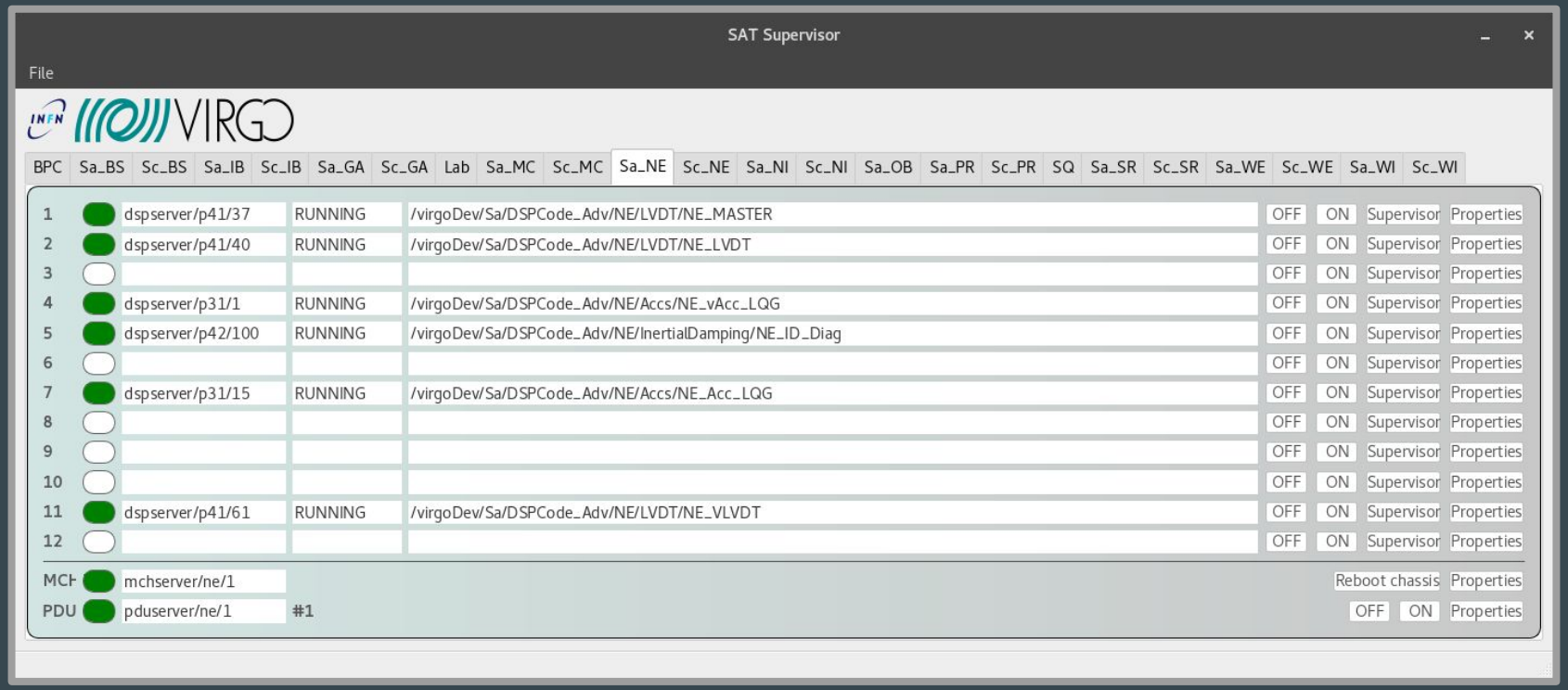
- Easily access to the information
- Service the system with the provided actions

Command Line Interface

Clients can be integrated in any Python script. This is useful to create tools for maintenance and debug. CLI scripts are used to

- Validate the analog frontend of UDSPT boards.
- Validate the 5 Gbps point-to-point RapidIO link between boards in the same crate.

GUI Clients



This client allows to:

- Turn on/off or reboot a single board or the whole crate
- Set the default hard-real-time routine to be loaded at the startup in each board
- Read temperatures and other values provided by sensors on the printed circuit boards.

GUI Clients

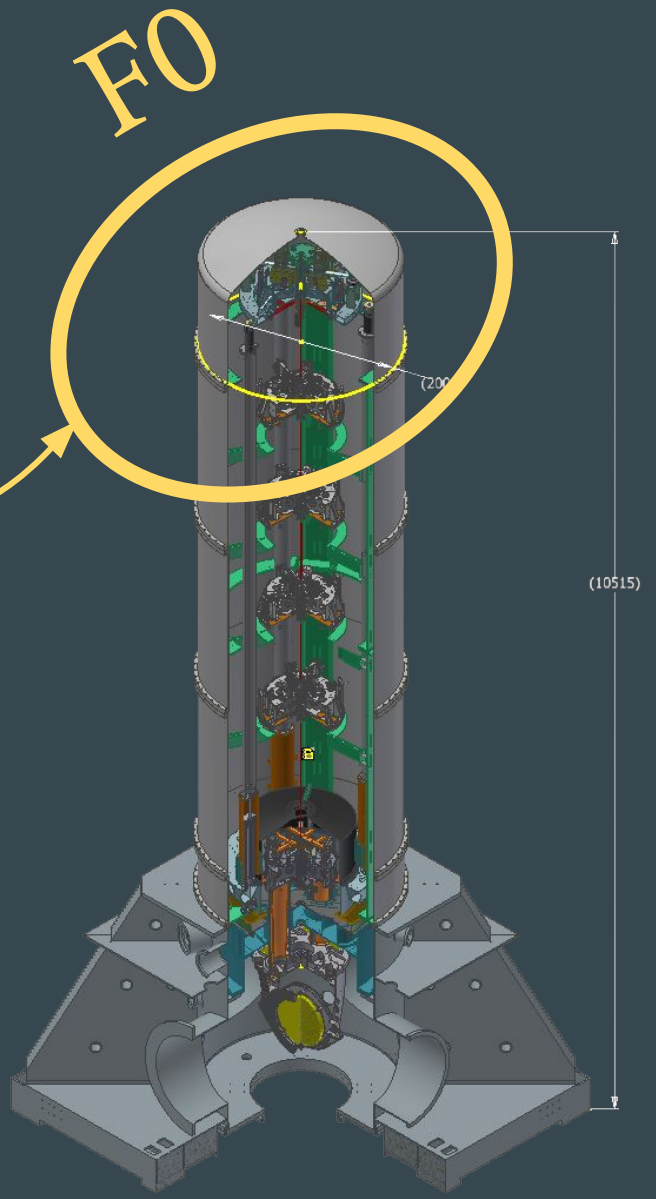
This client allows to change some variables in the DSP hard-real-time routines at runtime.

Mainly this is used to:

- Open and close control loops of the suspension
- Read and set the reference positions of the suspension and of the mirror

Top stage position control tool

The Software Supervisor can be used not only for monitoring, but also to develop applications that don't require low latencies, for example a tool to set the reference position of the suspension top stage, called **F0**.

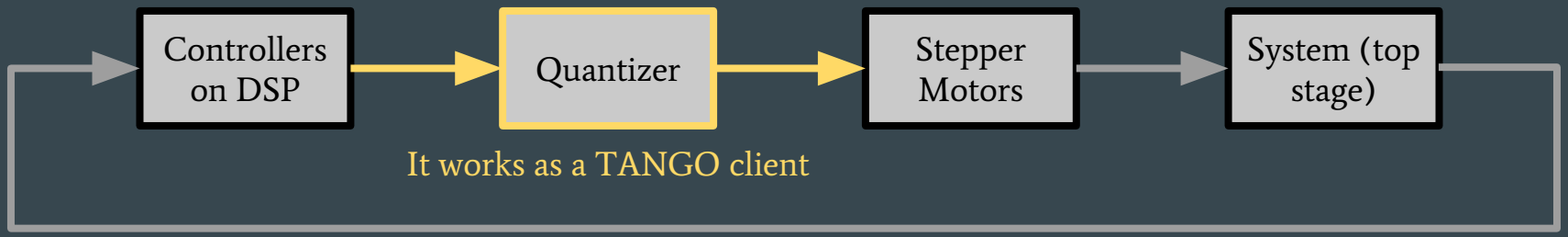


The F0 can be moved by:

- 3 coil actuators placed at 120°
  - relatively small dynamic (millimeters)
  - used for the feedback of the hard-real-time suspension control system
- 3 stepper motors placed at 120°
  - bigger dynamic (centimeters)
  - noisy, used only during maintenance periods to reduce the DC voltage on the coils

The last task is performed by hand, but can be easily done with a slow control system!

Top stage position control tool



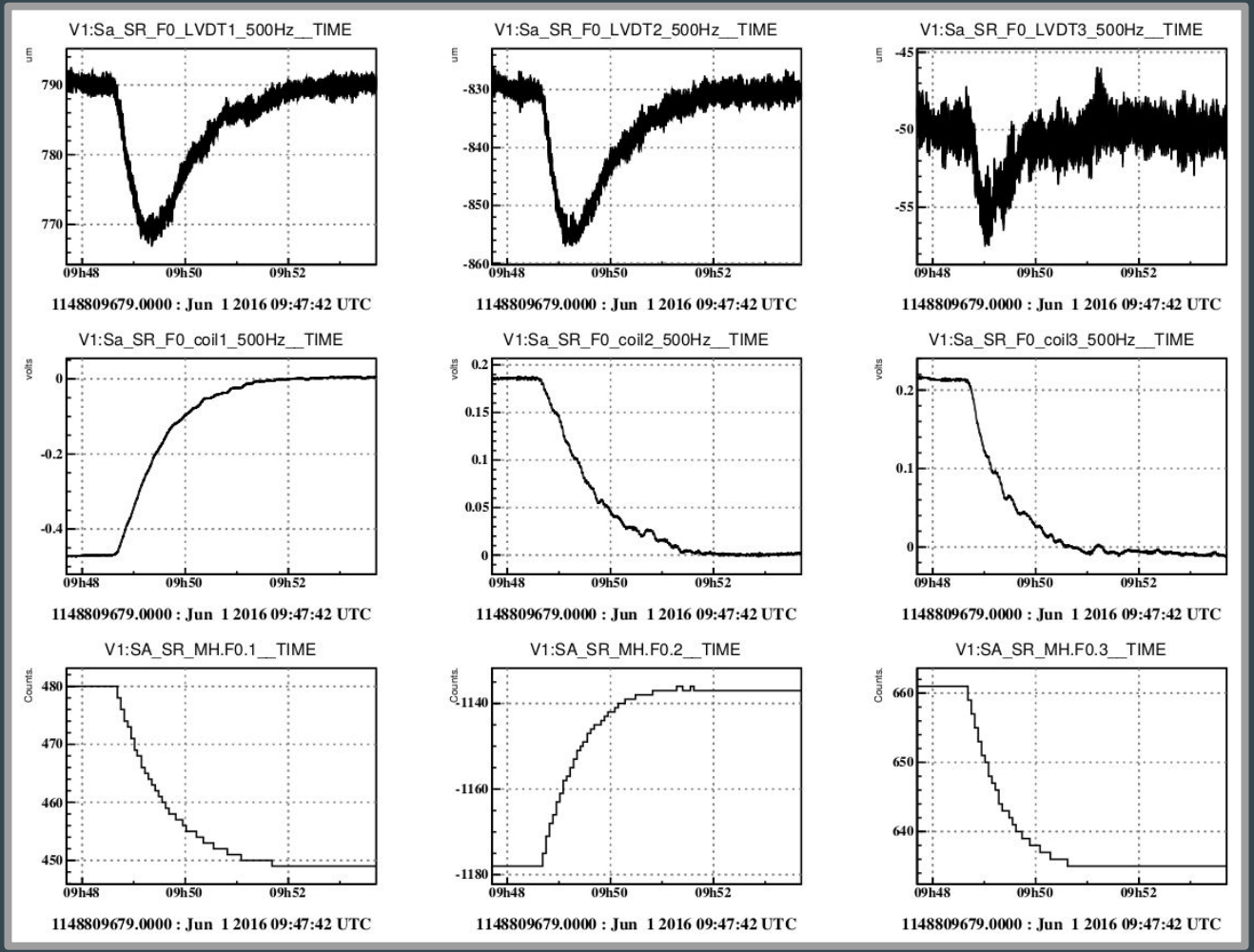
The idea of the control is simple:

- The position of the F0 is controlled by the DSP using the coil drivers on a arbitrary reference position position
- Another controller in the DSP computes the number of steps to minimize the corrections on the coils

In these conditions,

- A Python script reads the numbers of steps from the DSP, quantizes them and sends them to the motors. This is done as a client of the software supervisor!
- The corrections applied on the F0 coil drivers goes to zero (within the resolution of a step) in less than 100 s

Top stage position control tool: results

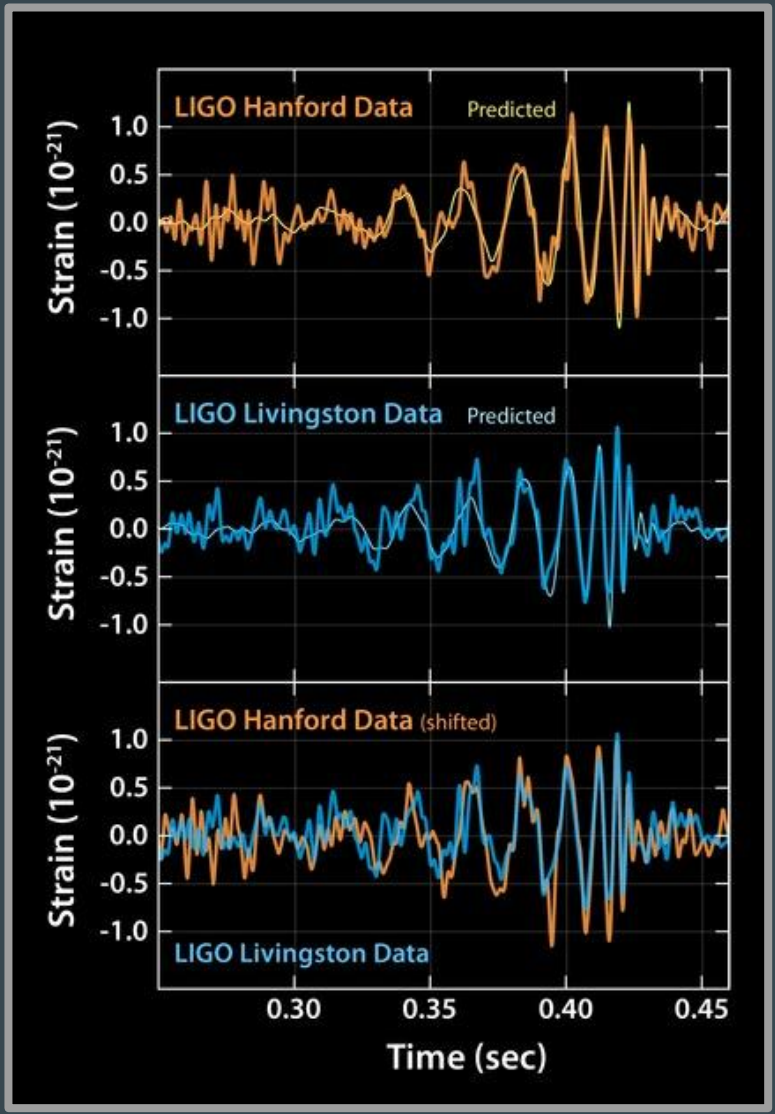


Top stage position

Voltages on coil

Cumulative steps

The detection of Gravitational Waves



On 2015 the LIGO observatories detected two signals from binary black hole mergers.

Fast **online** analyses are fundamental if we want to be able to look for possible electromagnetic counterparts of the event:

their latency must be low enough to trigger electromagnetic follow-up observations by some astronomical partners

especially for binary neutron stars and neutron star/black-hole mergers.

Low-latency pipelines

Currently two families of online pipelines exist:

Matched filter searches

- A matched filter is obtained by correlating a known signal, or template, with an unknown signal
- Mathematically it is the optimal way to to detect the presence of a template in the data of a GW detector
- Use of a bank of search templates to cover the parameter space of expected signals
- The searches are done separately on each single detector, then results are combined to find coincidence events

Generic transient searches

- Identification of time-correlated short duration transients in multiple detectors
- The searches is done using the data of all the detectors together
- No assumption of any particular signal morphology, origin, direction or time, i.e. no templates

Low-latency pipelines

Currently two families of online pipelines exist:

GstLAL

- Time-domain matched filtering to compute the SNR using ~250.000 templates in O1
- Peaks are identified in the matched-filter SNR time series and used to generate discrete triggers
- Each trigger is then checked for time coincidence (< 15 ms) with triggers from the same template in the other detectors
- The false alarm probability is associated to the candidate event.

~ 65 s

cWB

- Based on wavelet transformation and time-delay filters
- Combines data streams of two or more detectors into one coherent statistic
- A likelihood is associated to the candidate
- A job is launched every 60 s, that analyzes the last 180 s of data
- Takes 2 minutes to process a job

2-5 min

Data analysis with DSP

The existing pipelines run natively on standard CPU. Several ways to improve their performances have been investigated in the last years:

- Using the new instruction sets introduced by the CPU manufacturers (for example AVX and AVX2)
- Porting the most expensive calculations to other platforms, like GPU and DSP.

DSPs are processors specific for real time computations. There could be some advantages:

- Architecture optimized for the operations typically used in the pipelines (FFT, digital filters, upsampling/downsampling, ...)
- DSP used in the suspension control system can access the Virgo data with almost zero latency

GstLAL

GstLAL is based on GStreamer.

It has been designed around the possibility that some signal processing might be done using hardware acceleration:

- Switching between hardware accelerated element of the pipeline and a pure software implementation is completely transparent
- No need to rewrite the other parts of the pipeline

It is a pipeline-based multimedia framework that links together a wide variety of media processing systems to complete complex workflows.

GStreamer is free and open-source software.

It is commonly used as audio/video codec in Linux.

Future work

In the next months we'll try to:

- Identify the most computationally expensive parts of the GstLAL matched filtering
- Perform a feasibility study of porting that parts to a DSP-based architecture
  - Can we use the UDSPT boards? No PCIe link to CPU available on them.
  - Do we need to look for other boards?

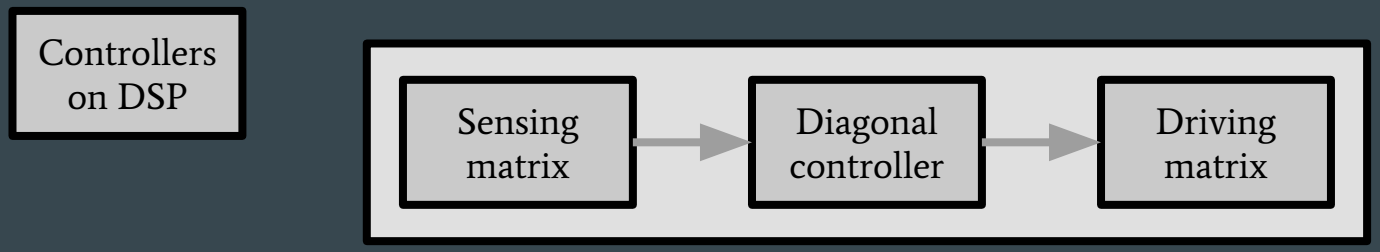
According to what we've understood so far, the candidates are:

- Audio resamplers
- Matrix mixers
- Channel adders

In particular, the audio resamplers takes >30% of the computing time of the GstLAL pipeline.

Thanks…!

Top stage position control tool: nonlinearities



The system has 3 inputs (the 3 stepper motors) and 3 outputs (the coil voltages to be minimized), and is not diagonal. Furthermore there are nonlinearities in the system:

- The top stage could be near to the external frame, or also touching it. In this zone, the coils and the motors could not act linearly on the top stage.
- While the DSP compute a continuous number of steps, stepper motors can apply only a finite number of steps: the quantizer is a nonlinear operator.

These reasons suggest to diagonalize the system, and to apply a diagonal controller on it. In this way, the controller consists of three simple PI, one for each diagonalized d.o.f.