INTRODUCTION TO ESIPAP COMPUTING SESSIONS

WEDNESDAY 9 – THURSDAY 10 FEBRUARY 2022

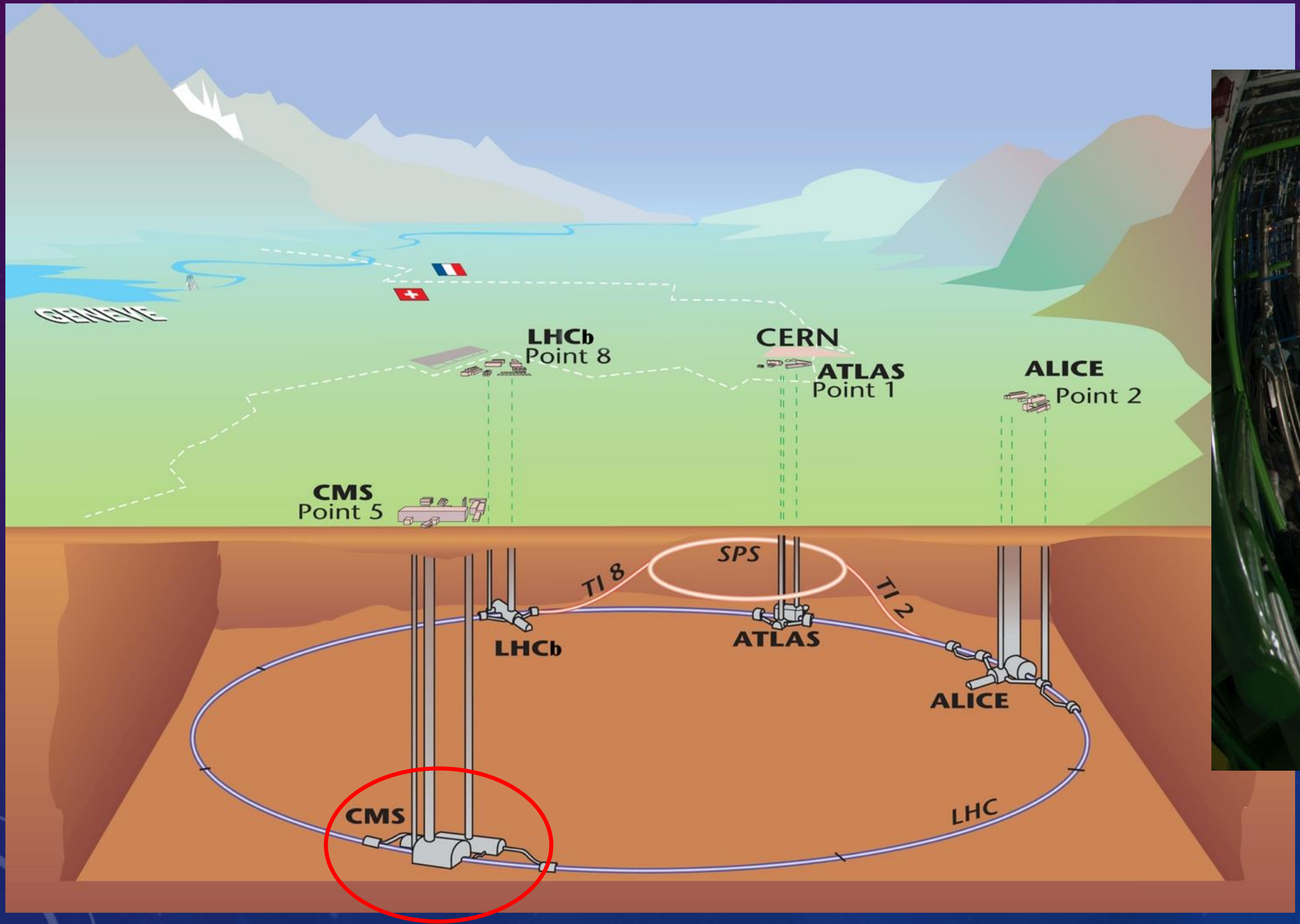
ERIC CHABERT - ERIC CONTE

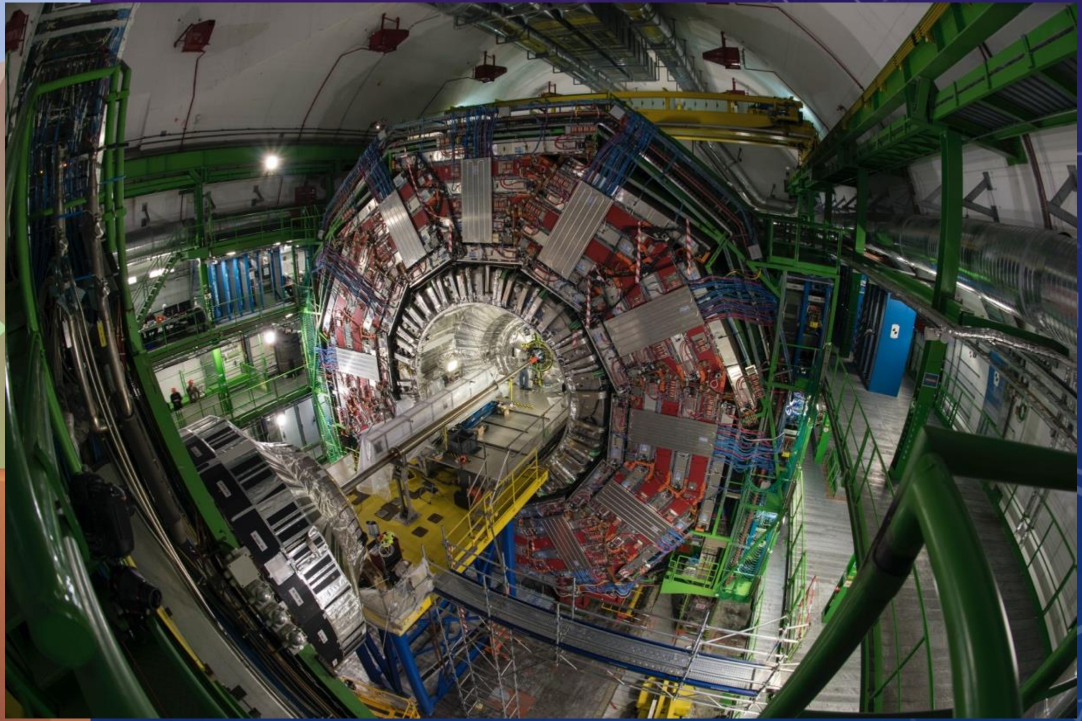
GOALS OF THE COMPUTING SESSIONS

- Computing is required for instrumentation purposes:
  - Simulation of sensor
  - Data acquisition
  - Data analysis
  - Algorithm and reconstruction of physics objects
- Computing sessions target to apply your theoretical knowledge:
  - Instrumentation
  - Software programming in C++
  - Using specific tools of high energy physics: ROOT
- Working by yourself and experimenting
- Getting the good practice

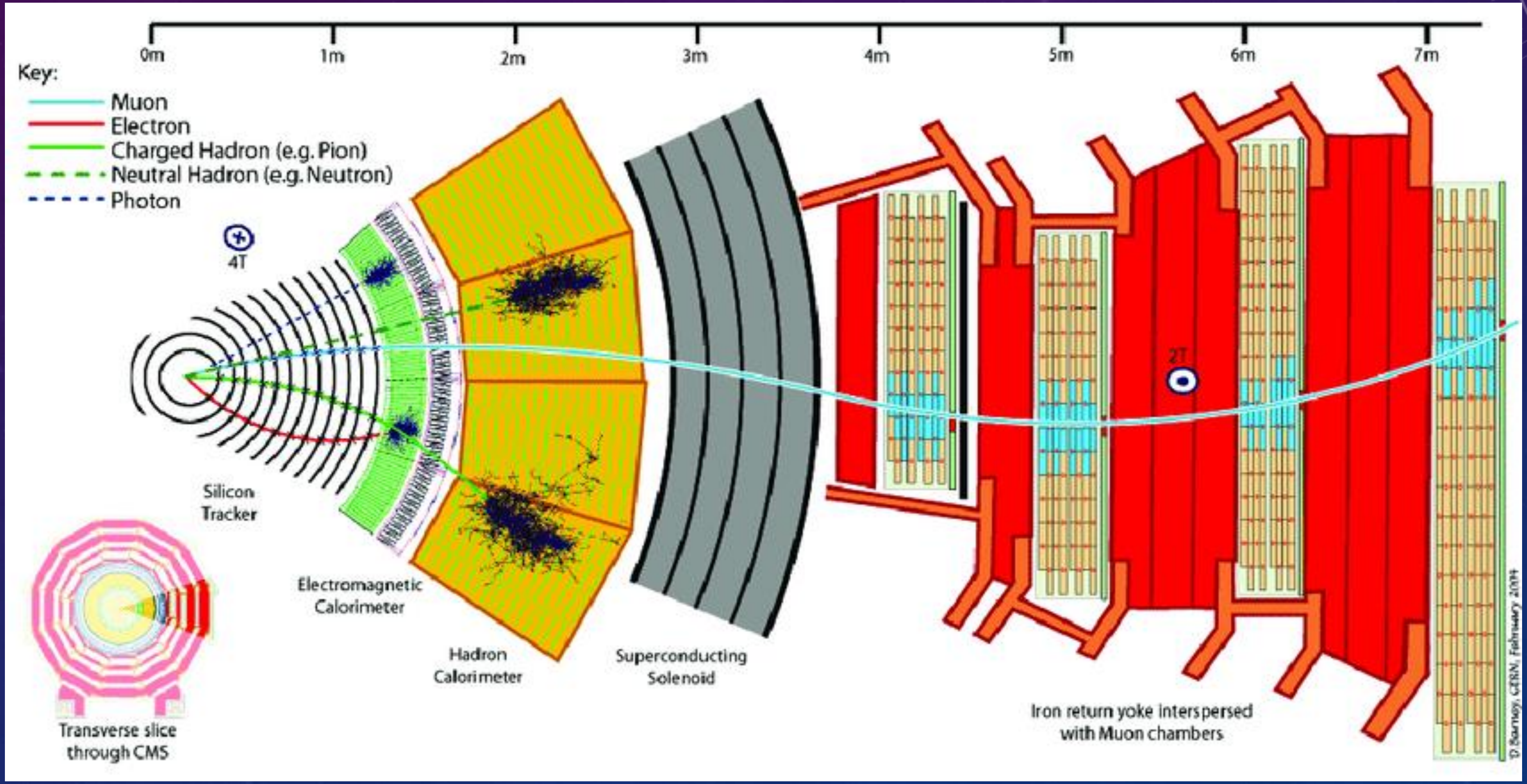
PHYSICS CONTEXT

THE CMS (COMPACT MUON SOLENOID) DETECTOR



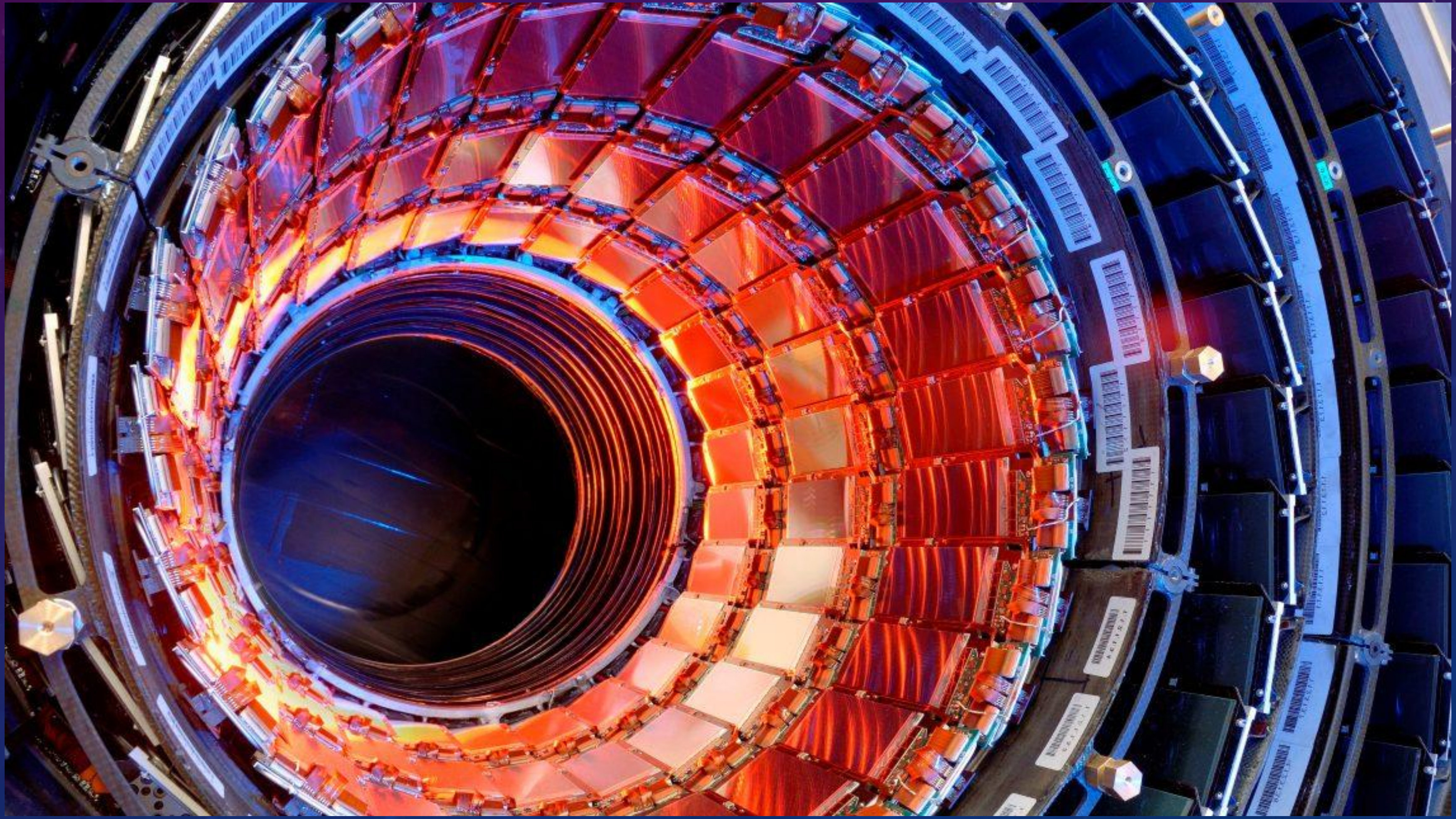


THE CMS (COMPACT MUON SOLENOID) DETECTOR


0m
1m
2m
3m
4m
5m
6m
7m
Key:
Muon
Electron
Charged Hadron (e.g. Pion)
Neutral Hadron (e.g. Neutron)
Photon
4T
2T
Silicon Tracker
Electromagnetic Calorimeter
Hadron Calorimeter
Superconducting Solenoid
Iron return yoke interspersed with Muon chambers
Transverse slice through CMS

D Barney, CERN, February 2004

SILICON STRIP TRACKER



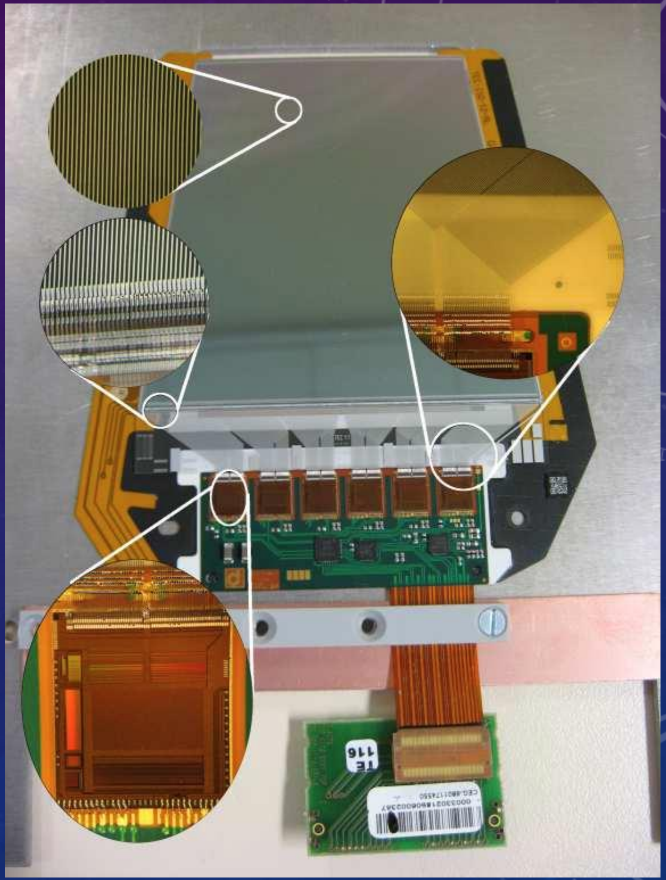
SILICON STRIP TRACKER

Instrumental activities

- R&D
- Construction
- Operation (online)
- Alignment & calibration
- Offline analyses
- Simulation
- Radiation damages evaluation
- …

CMS silicon strip tracker in few numbers:

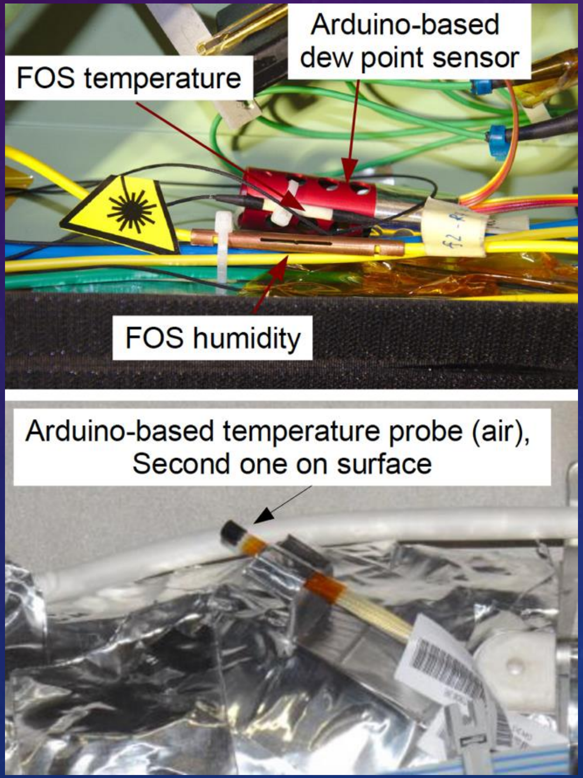
- 15 000 modules
- Surface: ~ 200 m^2
- 10^6 channels



Performances:

- Hit resolution: 20-40 µm
- Hit efficiency > 98% (at high Pile-Up)
- Timing alignment accuracy: 1ns
- …

SILICON STRIP TRACKER


Arduino-based dew point sensor
FOS temperature
FOS humidity
Arduino-based temperature probe (air), Second one on surface

During its operation it is important to monitor environment conditions:

- Temperature
  - Leakage current
  - Noise
  - Thermal dissipation
  - Radiation damages
  - …
- Humidity
  - Dew points & condensation
  - Front End electronics
  - …

Monitoring tools

Several probes are used to monitor that:

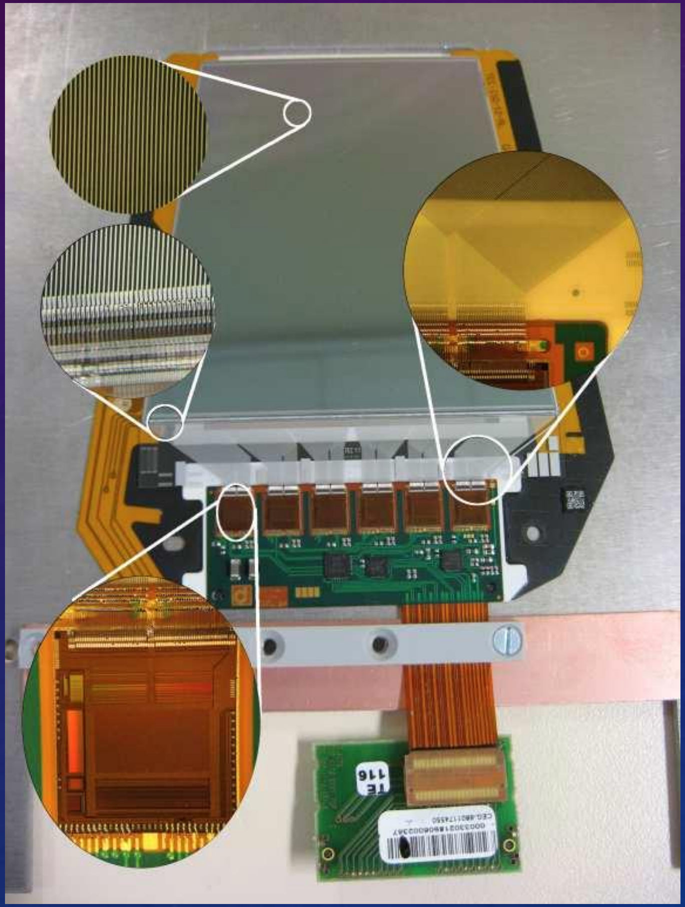
- On-board sensors
- External sensors

→ Some are ARDUINO-based!

COMPUTING SESSION AIMS

Instrumental activities

- R&D
- Construction
- **Operation (online)**
- Alignment & calibration
- **Offline analyses**
- **Simulation**
- Radiation damages evaluation
- …



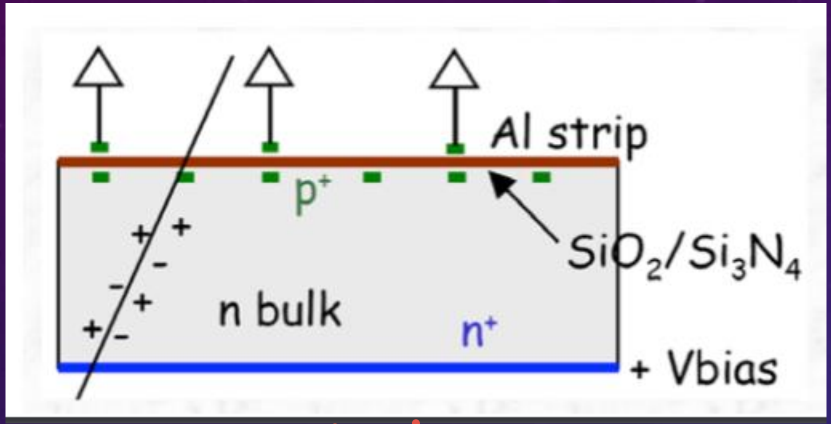
1. **Slow control**
   - Using a dedicated electronic board (Sense Hat) read by a Raspberry
     - Monitor the temperature & humidity
     - Send warning when conditions are not fulfilled

2. **Offline analyses**
   - Calibration of the temperature sensors
   - Evaluation of the sensor resolution

3. Simulation
   - Basic simulation with the GEANT4 package of a CMS silicon strip sensor

DATA USED IN THE COMPUTING SESSIONS

SENSORS TO STUDY



CONDITIONNING
Analogic Front-End + ADC + Signal treatment

2 channels
= collected charged
= energy (0 to 255)

- Temperature
- Relative humidity

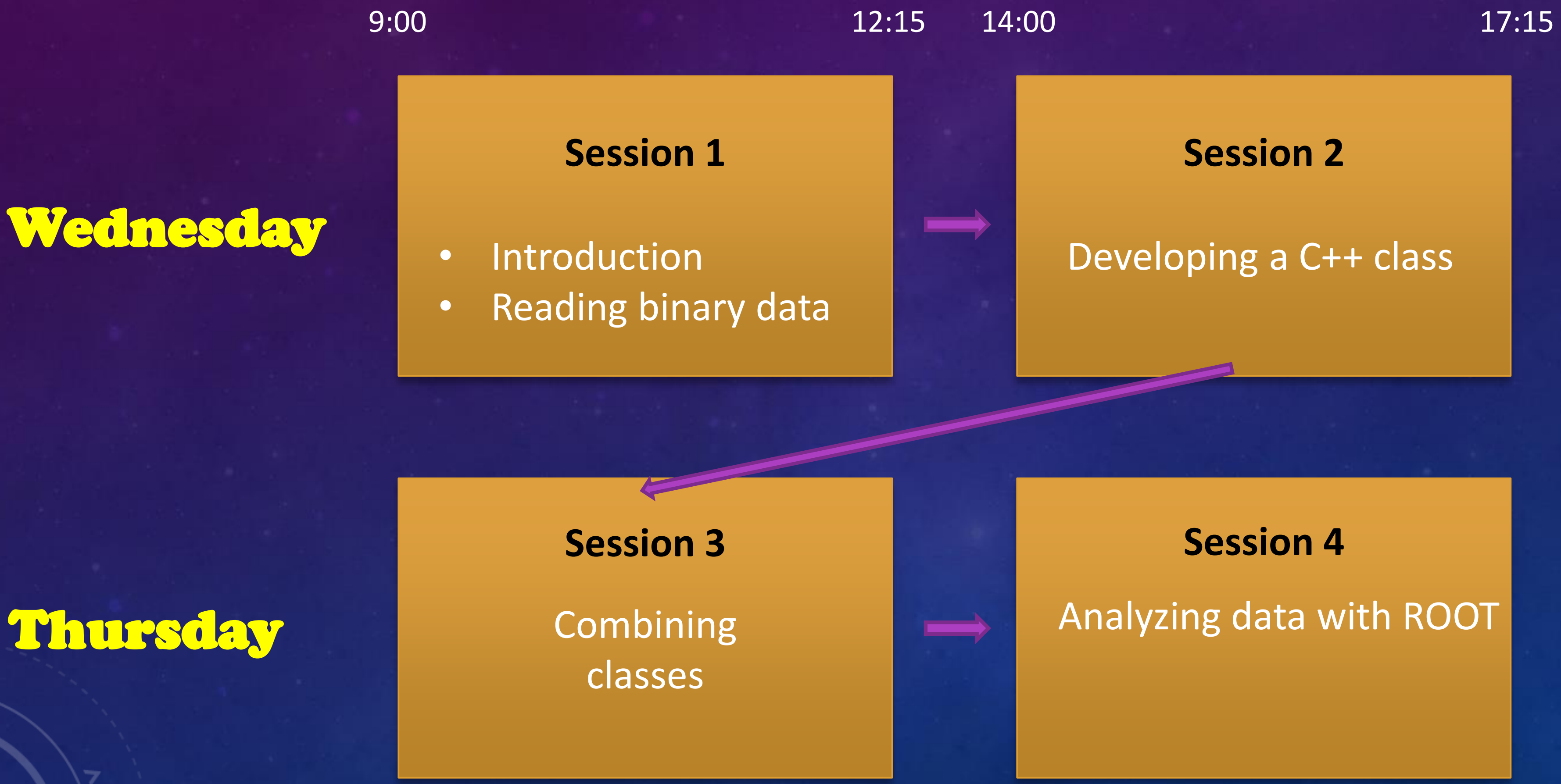
Pressure

SUMMARY ON ADC SENSITIVITY

	Pressure	Temperature	Humidity
Full scale	13.25 hPa to 2013.25 hPa	-20°C to +100°C	0% to 100%
ADC resolution	12 bits	12 bits	8 bits
Sensitivity	0,49 hPa	0,029 °C	0,39 %

ORGANIZATION

ORGANIZATION IN SESSIONS



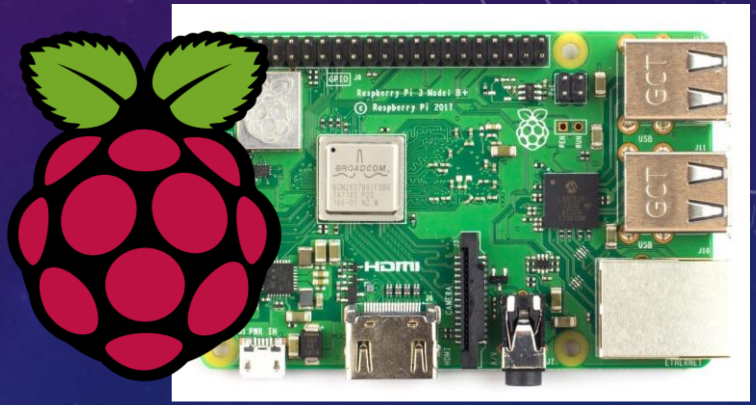
	9:00 – 12:15	14:00 – 17:15
Wednesday	**Session 1** - Introduction - Reading binary data	**Session 2** Developing a C++ class
Thursday	**Session 3** Combining classes	**Session 4** Analyzing data with ROOT

MULTI-PLATFORM DEVELOPMENT

Windows

Linux

Mac OS X



Raspberry board
(ARM architecture)

TOOLS TO USE

C++

ROOT
Data Analysis Framework

git

- Saving and preserving code on the internet: site github
- Sharing codes with others.

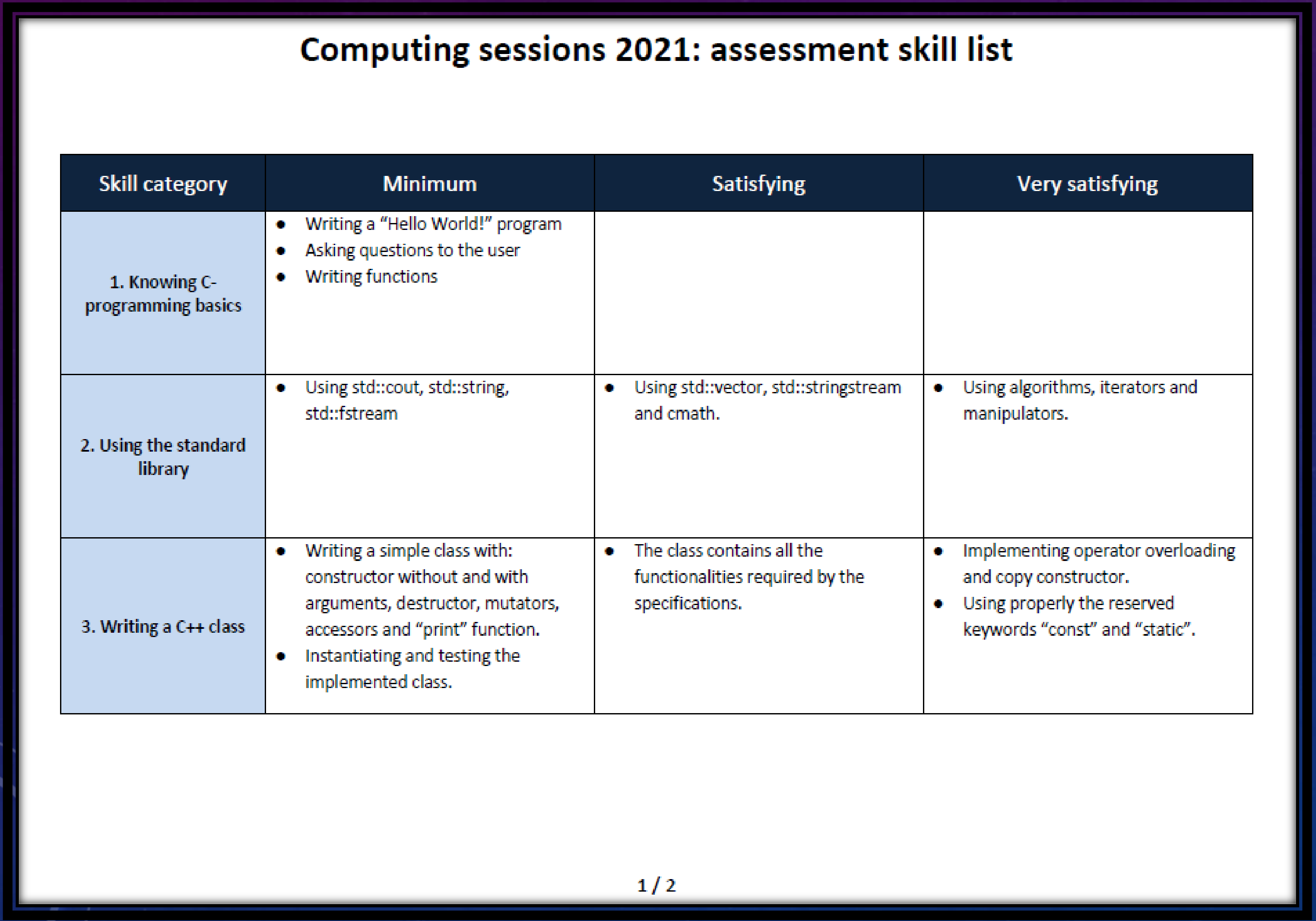
Doxygen

Generating automatically documentation of your code (in HTML and LaTex)

GNU Make

Building a C++ project witch several files (Linux / MacOSX only)

SKILL ASSESSMENT



Computing sessions 2021: assessment skill list

Skill category	Minimum	Satisfying	Very satisfying
1. Knowing C-programming basics	• Writing a “Hello World!” program • Asking questions to the user • Writing functions		
2. Using the standard library	• Using std::cout, std::string, std::fstream	• Using std::vector, std::stringstream and cmath.	• Using algorithms, iterators and manipulators.
3. Writing a C++ class	• Writing a simple class with: constructor without and with arguments, destructor, mutators, accessors and “print” function. • Instantiating and testing the implemented class.	• The class contains all the functionalities required by the specifications.	• Implementing operator overloading and copy constructor. • Using properly the reserved keywords “const” and “static”.

1 / 2

- Individual work is required
- Evaluation over 8 categories
- For validating the module
  - Minimum level must be reached for all the 8 categories
  - Satisfying level for at least 4 categories