CMS HGCAL: ADC Driver
Using $I^2C$ for Hexaboard
Sensor Development

Margot Lockwood
André David
December 14th, 2022
Margot Lockwood

Physics student at the University of Kansas interested in interdisciplinary scientific research in order to study the fundamental nature of the universe and to better the planet through technological advances.

Hobbies: Climbing, Art, Fermentation, Travel
1. Background information  
   a. HL-LHC and HGCAL  
   b. Project Goal  
   c. Project Specific Terminology  
2. Project Details  
   a. Chip Overview  
   b. Driver Implementation  
   c. Driver in Action  
   d. What’s Next?  
3. Reflection
Background: High Granularity Calorimeter (HGCAL) Endcap Upgrade

- High Luminosity (HL)-LHC will increase beam density
  - $2 \times 10^{34} \text{s}^{-1}\text{cm}^{-2} \rightarrow 5 – 7.5 \times 10^{34} \text{s}^{-1}\text{cm}^{-2}$
    - More collisions
    - More data (rare physics! Better precision)
    - More pile-up (40 → 140-200)

- HGCAL Upgrade will enhance particle identification, energy resolution, and pile up rejection
  - Harsh radiation environment
  - High precision spatial, energy, and timing information
  - 20ps timing precision required
High Granularity Calorimeter (HGCAL) Endcap Upgrade: Electromagnetic Section (CE-E)

To upgrade the EM section of the calorimeter 620 m² of silicon sensors are being installed (26 layers)
My Project

Create driver to readout voltages on ROC using ADS112C04 chip, an ADC, with I^2C protocol
Terminology

ROC - Read Out Chip
Reads data from the silicon sensor
Contributes to trigger system and DAQ system
3-5 ROCS per hexaboard divided into two halves
**Terminology**

**ADC** - Analog to Digital Converter

Allows analog sensors (light, sounds, pressure, voltage, etc) to communicate with microcontrollers

---

Example:

Let 1 LSB = 0.03125°C

We can then convert a binary number to a sensible piece of data:

00 0011 0010 0000 → 25°C

For a negative number the far left bit represents sign. We subtract 1 and flip all bits:

11 1100 1110 0000 - 1 = 11 1100 1101 1111

flip(11 1100 1101 1111) = 00 0011 0010 0000

Thus, 11 1100 1110 0000 → -25°C
Terminology

\( \text{i}^2\text{C} \) - Inter-Integrated Circuit

Uses two lines (SDA and SCL) for bidirectional communication

Can use System Management Bus (SMBus)

(similar to DNA translation with start codon [AUG], information, then stop codon [UAA/UAG/UGA])
My Project

Create driver to readout voltages on ROC using ADS112C04 chip, an ADC, with I^2^C protocol
ADS112C04

- Uses I²C
- ADC converter
- Has built-in internal temperature sensor
- 4 registers with various configurations
- Can be connected to a variable temperature resistor (schematic pictured left)
- Can be used to probe dozens of internal voltages on the ROC
  - This will help problem solve issues with the silicon sensor

Connected to trophy board (which looks like a trophy I guess?) which is connected to hexaboard
My physical set up: 640257
My physical set up: 640257
My physical set up: 640257
Driver Implementation

- trophyADC.py: Object oriented!
  - Classes: TrophyADC
    - RDT: temperature dependent resistor readings
    - ADC1: probing ROC
  - Methods
    - Configure
    - Basic commands
    - Reading raw data
    - Converting data to voltages or temperatures
    - Basic statistics
    - Calibration (by shorting PGA to mid supply voltage)

- config.yml

- checkBus.py: scans I²C Bus for acknowledging addresses with configurable devices

- readme.md

Can control data rate, gain, operating mode, reference voltage, current setting, etc.

Register configuration makes physical changes!
Driver In Action

- Currently using my ADC driver: measure noise in coulombs to calibrate ROC
  - Translate ROC ADC into ROC DAC injection value
  - Setting different ROC DAC injection values and reading voltages (my ADC)
- Reading data from both halves of each of the 3 ROCS for different voltages to make calibration curve
What’s Next?

- Use the driver
  - Do the data values make sense?
  - Modifying other scripts to use my driver
- I would love to continue to work on HGCAL
  - Make more SWAMP objects
  - Remotely in January
  - Gradschool?
  - Continue work this spring/summer?
Outtakes

- I have learned a lot...
  - Programming, detector development, electronic interfacing, cultural exchange
  - 2 monitors >> 1 monitor
  - CERN is the best
  - I need to learn another language
  - I like cities???

Andre David Tinoco Mendes 2:12 AM
Nothing like putting the right bits in the right place.
Acronyms and Abbreviations running list…

CERN :)
LHC
CMS
ATLAS
ALICE
HGCAL
HCAL
LUCY
HEP
TRENZ
ToT
PS
toA
PL
PA
SDA
TDC
SCL
CM
VHDL
BX
ECON
I'C
IpGBT
SMBus
PS (but different than the earlier PS)
FPGA
SPS
DAQ
CM
ROC / DR
HGCROC
TS
ADC
YAML (YAML Ain't Markup Language) -- A recursive acronym
DAC
PGA
L1
SPS (different than earlier SPS…)
PCB
IDAC
ZYNQ PL
RTD
ZYNQ PS
FIFO
MAD - Material Access Device
AXI
RF Cavity
SWAMP
LEIR
TOF
LINAC
LEAR
TWA (three word acronym)

It can be hard to communicate (with computers and people) but communication is important and useful. Miscommunication (often unknowingly) is common. Successful communication is rewarding.

PCMCIA - People Can't Memorize Computer Industry Acronyms

Personal Computer Memory Card
International Association
Outtakes

- Highlights:
  - Beam test
  - 1st time communicating with my ADC
  - Fedor’s excitement seeing a Dyson vacuum
  - Orsay and Versailles
  - Sparking water on tap at CERN
  - Cheese (munster!)
  - Andre giving chocolate
  - Asking people from different locations “do you have this/ what do you call this” and about stereotypes
  - Being excited to work Monday mornings

- Travel:
  - Chamonix, Lyon, Paris, Barcelona, London, Prague (tomorrow)
Acknowledgments

- André David, Arnaud Steen, and the HGCAL CMS team for mentorship and help
- Steve and Tom for organizing this experience
- US State Department in coordination with University of Michigan for funding this experience

Thank You!
Questions?
Extra slides…
Outtakes:
Program changes suggestions

- Do not go to CERN directly from airport, but wait so we can rest and look alive in our ID photos
- More group lectures
- Be registered with CMS instead of ATLAS
Overview of Device Operation

8.4 Device Functional Modes
Figure 54 shows a flow chart of the different operating modes and how the device transitions from one mode to another.

```python
def start_sync():
    """Start or restart conversions"""
    bus.write_byte(address, 0x8)

def read_data():
    """Read data: waits for read bit to be set to high indicating data is ready."""
    configure_register=0
    start_sync()
    bus.write_reg(address, 0x8)
    start = time.time()
    while check_data():
        time.sleep(0.001)
        print("wait", time.time() - start, " seconds for data")
    conversion = bus.read_word_data(address, 0x6)
    return conversion

def power_down():
    """Enter power-down mode""
    bus.write_byte(address, 0x2)

1. Any reset (power-on, command, or pin) immediately resets the device.
2. The conversion mode is selected with the CM bit in the configuration register.
3. The POWERDOWN command allows any ongoing conversion to complete before placing the device in power-down mode.
**Terminology**

**ROC** - Read Out Chip

Reads data from the channels of the silicon sensor

Contributes to trigger system and DAQ system

3-5 ROCS per hexaboard divided into two halves

**ADC** - Analog to Digital Converter

Allows analog sensors (light, sounds, pressure, voltage, etc) to communicate with microcontrollers

**I²C** - Inter-integrated circuit

uses two bidirectional lines (SDA and SCL)

Example: °C

1 bit=0.03125°C

00 0011 0010 0000 → 25°C

11 1100 1110 0000 → -25°C