Caribou: A Versatile Data Acquisition System Based on Programmable Hardware

7th Beam Telescopes and Test Beams Workshop,
17 January 2019

Tomas Vanat, CERN
tomas.vanat@cern.ch
On behalf of the CLICdp collaboration
Motivation

• A similar concept of readout, control and power is used in most silicon pixel detectors
  • Differs in voltage levels, number of channels (data/voltage) or protocol

• A new detector-specific DAQ system is usually developed for each new detector (or an existing one is modified)
  • Time-consuming process of HW/FW/SW development
  • No innovative functionality

• A versatile DAQ system can speed-up development
  • Common HW and SW core and interface
  • For detector development and tests
Caribou

• Open source hardware, firmware and software for laboratory and high-rate beam tests

• Maintained by collective effort of developers from:
  • Brookhaven National Lab
  • University of Geneva
  • CERN

• Minimizes device integration effort
  • Reduces time to get first data from a new detector
**Caribou hardware architecture**

- **FPGA/SoC board** (e.g. Xilinx ZC706)
  - An embedded CPU runs DAQ and control software
  - An FPGA runs custom hardware blocks for data processing and detector control

- **Control and Readout (CaR) interface board**
  - Provides physical interface from the FPGA/SoC to the detector chip
  - CaR – FPGA connection extendable by a cable (3 m tested)

- **Application-specific detector carrier board**
  - Detector chip and passive components only
FPGA/SoC board

• Xilinx ZC706 (Zynq) is currently supported

• Runs a Yocto-based Linux on an embedded ARM CPU
  • Standalone machine connected via Ethernet
  • Remote SSH connection available
  • Runs DAQ software (Peary)
  • Can run data analysis (quality monitoring) locally
  • Data can be stored locally (on SD card) or to a network-mounted storage (NFS, …)

• CPU is interfaced to FPGA fabric
  • Runs lower layers of communication protocols
  • Can (pre)process data in hardware
  • DMA is available
Control and Readout (CaR) board

- 8 adjustable power supplies with monitoring (0.8 – 3.6 V, 3A)
- 32 adjustable voltage references (0 – 4 V)
- 8 adjustable current references (0 – 1 mA)
- 8 voltage inputs to slow (50 kSPS) 12-bit ADC (0 – 4 V)
- 16 analog inputs to fast (65 MSPS) 14-bit ADC (0 – 1 V)
- 4 programmable injection pulsers
- 8 full-duplex high-speed GTx links (<12 Gb/s)
- 17 LVDS links (bidirectional)
- 10 output and 14 input links, adjustable level (0.8 – 3.6 V)
- Programmable clock generator
- External TLU clock reference and trigger input
- External HV input
- FEAST module compatible
- FMC interface to FPGA board (extendable by ~3 m cable)
- 320-pin SEARAY interface to detector chip
Detector carrier board (chip board)

• Detector specific
  • The **only** physical *hardware* to be made for a new detector
  • Passive and detector-specific components only

• Already supported detectors:

  ![C3PD](image1)
  ![CLICpix2](image2)
  ![ATLASPix](image3)
  ![ATLASPix2](image4)
  ![H35Demo](image5)
  ![FEI4](image6)
Caribou hardware architecture schematic

ZC706 board
- SD card
- ZYNQ SoC
  - SDIO
  - Ethernet
- Memory Controller
- DDR RAM

ZYNQ SoC
- ARM CPU
- I²C

FPGA
- Detector-specific firmware
  - GPIO
  - GTx
  - SERDES

CaR board
- Clocking
- Power Supplies
- Bias Voltages
- Pulser
- ADC
- GPIO
- Detector

External TLU (optional)

Chip board
- ZC706 board
- CaR board
- ZYNQ SoC

3 m + ↔
Caribou Firmware

- Based on custom and Xilinx IP cores
- An **interface** between CPU (SW) and a detector (HW)
- User needs to create or adjust detector-specific modules
  - Data transfer
  - Detector control
- Modules are connected to ARM with AXI bus
  - Registers are mapped to CPU memory space
- DMA is available
  - Direct transfer of data to RAM, disk drive or network interface
Caribou software

- **Custom** Yocto-based **Linux** distribution (meta-caribou)
  - Common **Linux tools** and packages are **pre-installed** (ssh, python etc.)
  - **FPGA** firmware registers are **accessible** through `/dev/mem`

- **DAQ software framework** (Peary)
  - Hardware Abstraction Layer (HAL)
    - Interface between SW and HW
    - Allows handling peripherals as objects in C++
  - Functions to control CaR board and detector
  - Device management (multiple devices/detectors in parallel)
  - Command line interface (CLI)
  - Client interface for integration with a superior DAQ
How to implement a new detector

• Hardware:
  • Design and produce a **chipboard** – route chip signals to suitable pins on CaR connector

• FPGA firmware:
  • Create or modify **FW blocks**:
    • to handle the control signals for the detector
    • to receive detector data and push it to FIFO

• Software (Peary):
  • Define **mapping** of:
    • Generic names of **CaR board peripherals** to detector-friendly names (Vout_3 → VThrPix)
    • Addresses of your **detector-specific registers** in FPGA to variable/object names
  • Create a module with **detector-specific functions** (configure, startDAQ, ...)
The complete lab setup

- Power supply (12 V)
- CaR board
- Detector (CLICpix2)
- FMC cable (optional, 3 m +)
- FPGA/SoC board
- Ethernet
Test beam setup

- CLIC Telescope in North Area (SPS, CERN)
  - Caribou used for DUT
  - Telescope uses SPIDR readout for Timepix3
  - Common 40 MHz clock and $T_0$ from TLU
  - Counting clock cycles $\rightarrow$ timestamps
Future plans

- Support for new detectors:
  - CLICTD, ATLASPix3, RD50, CLIPS
- Support for ZCU102 (Zynq UltraSCALE) board
  - Faster 4-core CPU, SATA for a larger local storage, HDMI for a local screen
  - Possible to connect 2 CaR boards at the same time
Caribou summary

• Caribou is a versatile DAQ system for silicon pixel detector
• Open source, Linux-based
• Standalone – no additional PC with a special software required
• Can run online data analysis locally
• Aimed for prototyping – laboratory and beam tests
• Focused on fast and simple implementation of a new detector
• New users and developers are welcome
• https://gitlab.cern.ch/Caribou