

AGATA - EDAQ V2

Pre-Processing F/W

Christian BONNIN
Laurent CHARLES

Outline

- IPBusBridgeIP
- GTS Leaf
- Energy & Trigger Blocks
- EventBuilder = EventBuffer + EventFilter
- EventSender = Readout
- Slow Control by IPBus
- Global Integration
- Schedule/Plan

IPBusBridgeIP

■ Context

- In the final architecture (couple PACE + STARE)
 - ✓ Only one Ethernet link will be used for the global slow control by IPBus
 - ✓ IPBus should be shared between the 2 boards
 - For configuring (acquisition, boards, peripherals, digiOpt12)
 - For monitoring (internal registers, sensors, peripherals)
 - ✓ PACE will be the master
- Nothing existed before for expanding IPBus from one board to another
- IPHC has developed an IP

■ Spec

- The link should be robust and reliable because used for configuring the DAQ and the boards
- Not necessary to be fast
- Firmware flexible through the use of generic parameters in VHDL

IPBusBridgeIP

■ Dev

- Use of AXI Chip2Chip + Aurora64b66b IPCores from Xilinx
 - ✓ Pros
 - ❑ Integrates internal errors detector/corrector
 - ❑ Plug & play
 - ✓ Cons
 - ❑ Take more logic resources compared to simple Aurora link
- A full link designed comprising
 - ✓ Master (sender) connected to the IPBus Controller
 - ✓ Slave (receiver) connected to remote IPBus slave(s))
- Compliant with 3 FPGA families from Xilinx (Ultrascale, Ultrascale+ and 7-series) in order to build a project on different hardware platforms
- Demo and test benches with the
 - ✓ KCU105 in loopback (through external fibers)
 - ✓ KCU105 + kintex-7 board from CMS
- Tests done : fonctionnal

■ Sources

- Clear, cleaned and deposited in a gitlab in2p3 repository
- Shared to IJCLab

IPBusBridgeIP

Overview of the git repo

CHARLES Laurent > ipbusBridgeIP







ipbusBridgeIP 
Project ID: 12094




☆ Star 0

Y Fork 0

🔗 2 Commits  1 Branch  1 Tag  59.7 MB Files  59.7 MB Storage

ipbusBridgeIP




Auto DevOps ✕

It will automatically build, test, and deploy your application based on a predefined CI/CD configuration.


Learn more in the [Auto DevOps documentation](#)

master  ipbusbridgeip /  



tag v1.0
CHARLES Laurent authored 1 month ago

ede5174c 

-
-
-
-
-
-

IPBusBridgeIP

Name	Last commit	Last update
boards	Initial commit	1 month ago
core/src	Initial commit	1 month ago
doc	Initial commit	1 month ago
extern	Initial commit	1 month ago
tests	Initial commit	1 month ago
.gitignore	Initial commit	1 month ago
README.md	tag v1.0	1 month ago

README.md

IpbusBridgeIP

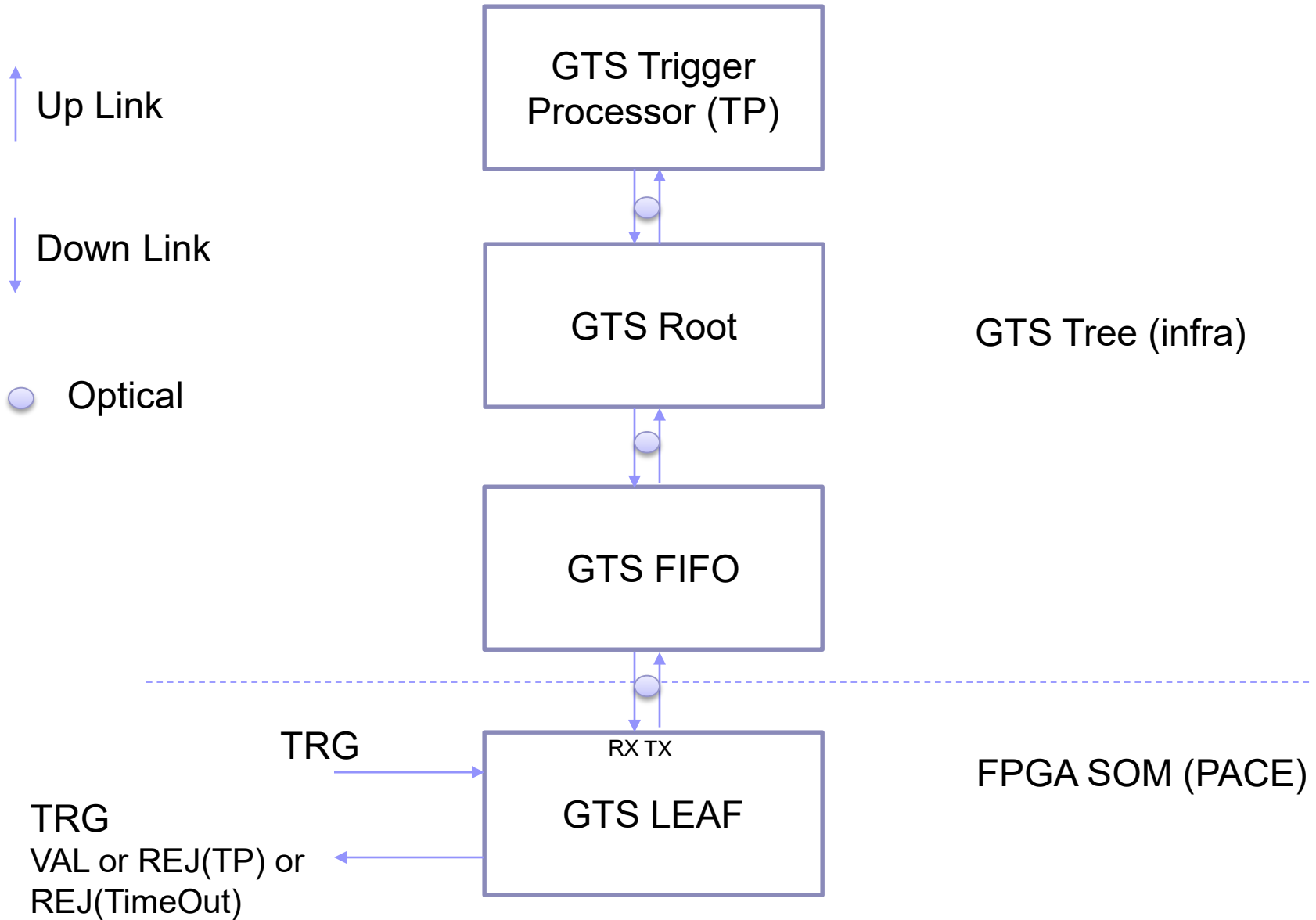
Description

- IPBus Bridge IP using AXI Chip2Chip link from Xilinx
- Core with the HDL sources (Master and Slave mode) and the IPCores from Xilinx
- Compliant with 3 FPGA families from Xilinx (Ultrascale, Ultrascale+ and 7-series) in order to build a project on different hardware platforms (boards, kits)
- Proposed with example boards
- Shared to IJCLab for Agata Phase 2 Development

Authors & Contributors

- **Laurent CHARLES** - CNRS/IPHC Strasbourg

GTS Leaf



GTS Leaf

- Re-work of the sources from Ganil
- Features
 - MGT targeted UltraScale & Ultrascale+ for the moment
 - ✓ Generic parameters in VHDL for selecting the right Xilinx primitives
 - Resets done automatically at power-up or through IPBus
 - ✓ Initially, at power-up, if everything is well connected, the byte-alignment should be done and the GTS Leaf ready to run
 - ✓ The MGT status is anyway readable by IPBus
 - Alignment done?
 - Re-alignment seen again? (normally should not occur)
 - Most of the commands may be generated by soft for testing
 - ✓ Command 'Trigger Request'
 - ✓ Command 'Start of backpressure' (emulating the DAQ'buffers going almost full)
 - ✓ Command 'End of backpressure' (emulating the DAQ'buffers going almost empty)
 - Command 'Idle' available and configurable by IPBus (enable/disable ; occurrence frequency)
 - ✓ Useful for debug through online debugger on GTS Root (ChipscopeAnalyser)

GTS Leaf

- GTS Root emulator (down encoder) for testing the GTS Leaf core in external loopback (TX fiber => RX fiber)
 - ✓ Instead of transmitting “up frames” to GTS tree, the TX is configured in such a way to send “down frames” as if it was the GTS Root
 - ✓ This avoids having the whole GTS infra for testing the core functionalities (dev, debug, testing of new features)
- Counters for diagnostics readable by IPBus useful for burn-in testing
 - ✓ Counters in the block ‘down decoder’
 - ❑ Counters of hamming errors
 - Should be 0 if no failure(s) in the whole hardware of the GTS tree
 - ❑ Counter of frames received having command errors
 - Should be 0 if no failure(s) in the whole hardware of the GTS tree
 - ❑ Counter of frames received with address = Crystal_ID
 - ❑ Counter of frames received with address = Broadcast_ID



GTS Leaf

- ✓ Counters in the block 'matching and timeout controller'
 - ❑ Counter of Trigger Requests sent to TX (up) and pre-buffered in local memory $\Leftrightarrow C1$
 - ❑ Counter of all Trigger Requests returned back in RX (down) $\Leftrightarrow C2$
 - Correspond to the Trigger Requests REJ or VAL by the Trigger Processor (TP)
 - Normally we should have $C1 = C2$ if the GTS tree is able to absorb and handle all the Trigger Requests initially transmitted
 - But currently, the up (top) of the GTS doesn't send a backpressure request to the down when the buffers are full
 - ❑ Counter of Triggers Requests sent rejected due to timeout $\Leftrightarrow C3$
 - When the Trigger Requests sent to up are not returned back to RX (down)
 - ❑ Counter of Triggers Requests returned back in RX (down) which match Triggers Requests sent initially and pre-buffered in local memory $\Leftrightarrow C4$
 - ❑ Normally we should have
 - $C1 = C3 + C4$

GTS Leaf


Sources

- Clear, cleaned, deposited in a gitlab in2p3 repository and regularly updated
- Shared to Ganil

 **gtsleaf**  Project ID: 12409 🔔 ★ Star 0 🍴 Fork 0

🔗 4 Commits 🌿 1 Branch 🏷️ 3 Tags 📁 11.9 MB Files 💾 11.9 MB Storage

GTS Leaf project (Agata Phase 2 Development)





Auto DevOps ✕

It will automatically build, test, and deploy your application based on a predefined CI/CD configuration.

Learn more in the [Auto DevOps documentation](#)

master ▼ gtsleaf / + ▼ History Find file Web IDE 📄 ▼ Clone ▼

 **readme updated** 1e839a91 
CHARLES Laurent authored 2 weeks ago

📄 README ➕ Add LICENSE ➕ Add CHANGELOG ➕ Add CONTRIBUTING ➕ Add Kubernetes cluster ➕ Set up CI/CD

Name	Last commit	Last update
------	-------------	-------------

GTS Leaf

Name	Last commit	Last update
boards	gtsleaf_downdec updated + scripts updated ...	2 weeks ago
core/src	gtsleaf_downdec updated + scripts updated ...	2 weeks ago
doc	Initial commit	3 weeks ago
extern/lc_lib	Initial commit	3 weeks ago
tests/kcu105_gtsleaf	gtsleaf_downdec updated + scripts updated ...	2 weeks ago
.gitignore	Initial commit	3 weeks ago
README.md	readme updated	2 weeks ago

README.md

gtsleaf

Description

- The GTS Leaf is the last stage of the GTS Tree (TP <=> ROOT <=> FIFO <=> LEAF <=> DAQ) connected to the DAQ (ADC Channels Processing)
- Re-work of initial sources from Ganil
- Compliant with 3 FPGA families from Xilinx (Ultrascale, Ultrascale+ and 7-series) in order to build a project on different hardware platforms (boards, kits)
- Pilotable over GbE/IPBus
- Plenty diagnostics for debugging purposes
- Proposed with example boards
- Shared to Ganil for Agata Phase 2 Development

Authors

- **Laurent CHARLES** - CNRS/IPHC Strasbourg

Board & Synthesis Tool

GTS Leaf

■ Tests ongoing at Ganil since 3 weeks

- With the help of Abderrahman Boujrad
- Use of the same KCU105 board than dev at IPHC
 - ✓ Optically connected to GTS Root via SFP & fibers
 - ✓ Differential Reference Clock coming from GTS Root and connected to SMA_MGT_REFCLK connectors from the board
- Link not established = no frames received by the GTS Root !!!
- Even the replicated test bench from IPHC (board configured in external loopback + use of GTS Root Emulator path instead of normal path) doesn't work
 - ✓ Optical components were changed
- The KCU105 board from Ganil seems broken
- Alternatives
 - ✓ Move to Ganil with the board from IPHC (but pandemic situation)
 - ✓ Migrate dev to the VC707 board already used for the Trigger Processor Upgrade
 - ❑ Need to evaluate the time it will take
 - ❑ Still a migration losing time

Energy & Trigger Blocks

- Blocks ready
- But lack of informations on
 - Latencies to be compensated/adjusted
 - Range of the delay functions
 - Current calibration procedures allowing to define the hidden functions behind the soft
 - ✓ Maybe an essential function should be kept but this function has to be known
 - What about the baseline restoration of segment channels if they are related to the trigger issued from the core channel ?
 - ✓ Unless it is tolerated to get wrong calculated energies from time to time in the current DAQ
 - Etc.
- Google doc in preparation based on my ideas and my current understanding
 - Will be shared for correction and agreement before coding
 - To know
 - ✓ The needs of Agata compared to other projects in nuclear physics
 - ✓ The degree of flexibility

EventBuilder = EventBuffer + EventFilter

■ EventBuffer

- In charge of buffering trace and energy for each trigger sent to GTS tree
- Architecture related to the architecture of the GTS Leaf well advanced now

■ EventFilter

- Discard the events (trace, energy) when receiving TRG_REJ (TP or TimeOut)
- Move the accepted events (trace, energy, timestamp, event counter) to EventSender when receiving TRG_VAL

■ First version ready in April

EventSender = Readout

- The events coming from the EventBuilder should be
 - First encapsulated/formatted by respecting the DataFormat (not really clear for me for the moment)
 - Then buffered in memory
 - Finally transmitted to STARE board via the Aurora64b66b link
- A solution of Readout by IPBus is feasible
 - By keeping the same DataFormat
 - Debug & backup solution
- Progress status
 - In standby the time to finish my current work

Slow Control by IPBus

- Need to propose a common registers table shared between PACE/STARE
 - Usefull for software developpers
 - Progress status : in standby the time to finish the current work
- Develop IPBus-based I2C Master Controller to make the interface with Valencia firmware
 - For configuring and monitoring the PACE board, peripherals, digi-Opt12
 - Progress status
 - ✓ A preliminary block exists; still to be completed
 - ✓ In standby the time to finish the current work

Global Integration



- Will be started when

- All previous blocks will be ready
- Hardware will be ready
- Firmware from Valencia will be shared

Schedule/Plan

■ Planning

- Unit test between labs, remotely? on site?
 - ✓ SlowControl(IPHC) ⇔ SlowControl(Valencia)
 - ✓ SlowControl(IPHC) ⇔ SlowControl(IJCLab) through IPBusBridgeIP
 - ✓ GTS Leaf IPHC ⇔ Ganil (burn-in tests)
 - ✓ DataTransfer IPHC ⇔ IJCLab (stress-tests, emulated data, physics data)
 - ✓ Etc.
- Tests in June

■ Availability LC / firmware : 50% Agata – 50% CMS

■ Software / IPHC : Christian Bonnin (CMS too)

- Status : no major contribution because of the successive delays
- Short term: tests in June
 - ✓ Need to evaluate what could be done in sw-side from python scripts and from the tools existing in Strasbourg for other projects (eg. readout by IPBus)
 - But scripts and registers table are not ready for the moment
 - ✓ Support Eric Legay team (methodologies, libraries, frameworks, git repos, tools, languages, etc.)
- Mid & long term
 - ✓ Available (monitoring, visualisation, support for Eric Legay Team)



Science & Technology
Facilities Council

UK Research
and Innovation

AGATA FPGA Processing Upgrades

Ian Lazarus

Rationale

- Signal Processing in the FPGAs- no changes for 20+ years
 - Can we gain from improvements in technology?
 - More powerful FPGAs
 - Higher bandwidths
 - Better ADCs
 - Maybe new algorithms?
- Where are limitations?
 - Spectrum Quality (resolution)
 - Throughput (pileup and count rate)
 - Reliability

Timeline

- UK funding for AGATA 2020 to 2024
 - *Detectors, Mechanics, PSA, Simulation, FPGA algorithms*
- Discussions to establish priorities (Dec 2020 to April 2021)
- Proposal to AMB (May/June 2021)
- Implementation schedule
 - Approval of proposal to AMB- July 2021
 - Coding starts August 2021
 - Prototype code ready to test offline Sept 2022
 - Prototype code ready to test in AGATA from February 2023
 - Code ready to use in AGATA Summer 2023

Initial discussion ideas

- Spectrum Quality and resolution
 - DNL correction (working with AP for digitizer)
 - Better characterization of preamp transfer function
 - Not really simple exponential as we assume
 - Baseline tracking
- Rate/Pileup
 - Higher singles rate without pileup losses
 - Usually limited by PSA except coincidence with VAMOS etc
 - Adaptive processing? Data Quality factor?
- Pre-processing for PSA
 - Identify and tag multiple interactions