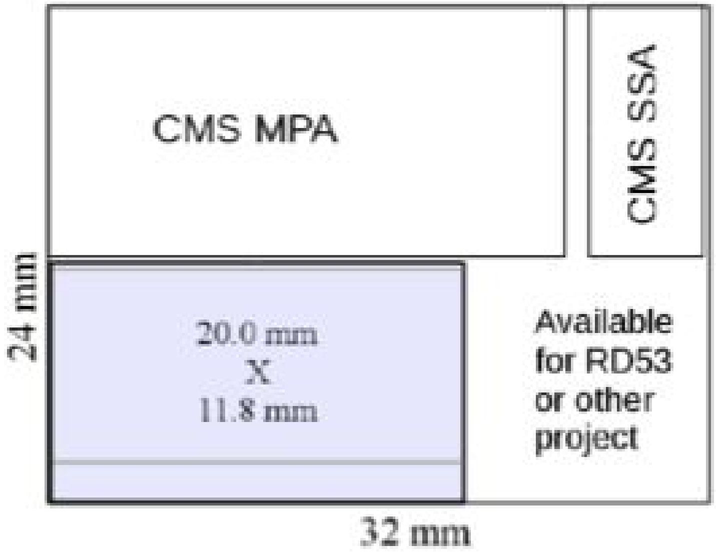
RD53A: a Large Scale prototype of a New generation Pixel Readout ASIC for the HL_LHC experiment

Vertex2017 - Las Caldas, Asturias, Spain - September 13th 2017

Roberto Beccherle, INFN - Pisa on behalf of the RD53 Collaboration:
[Bonn, CERN, FNAL, INFN (Bari, Milano, Padova, Bergamo-Pavia, Pisa, Perugia, Torino), LBNL, Marseille CPPM, New Mexico, NIKHEF, Paris LPNHE, Prague IP-FNSPE-CTU, RAL-STCF, Seville]

RD53: An ATLAS - CMS collaboration



RD53:

A collaboration amongst 18 different institutes from all over the world (plus additional guests) focused on pixel chips for ATLAS/CMS phase2 upgrades.

RD53 Goals:

- Detailed understanding of radiation effects in cmos 65nm to obtain guidelines for radiation hardness
- Design of a shared rad-hard IP library
- Design and characterization of full sized pixel array chip
- Development of tools and methodology to efficiently design large complex mixed signal chips

Has to address all challenges that next pixel upgrades will have to face:

- Small pixels:	50x50 um^2 (25x100um^2) and larger pixels
- Large chips:	~2 cm x 2 cm (~10^9 transistors)
- Hit rates:	3 GHz/cm^2
- Radiation:	1Grad, 2x10^{16} neu/cm^2 over 10 years
- Trigger:	1MHz, 10us (~100x buffering and readout)
- Powering:	Serial Powering

Last year was focused on the first full scale Pixel Chip prototype, **RD53A**.

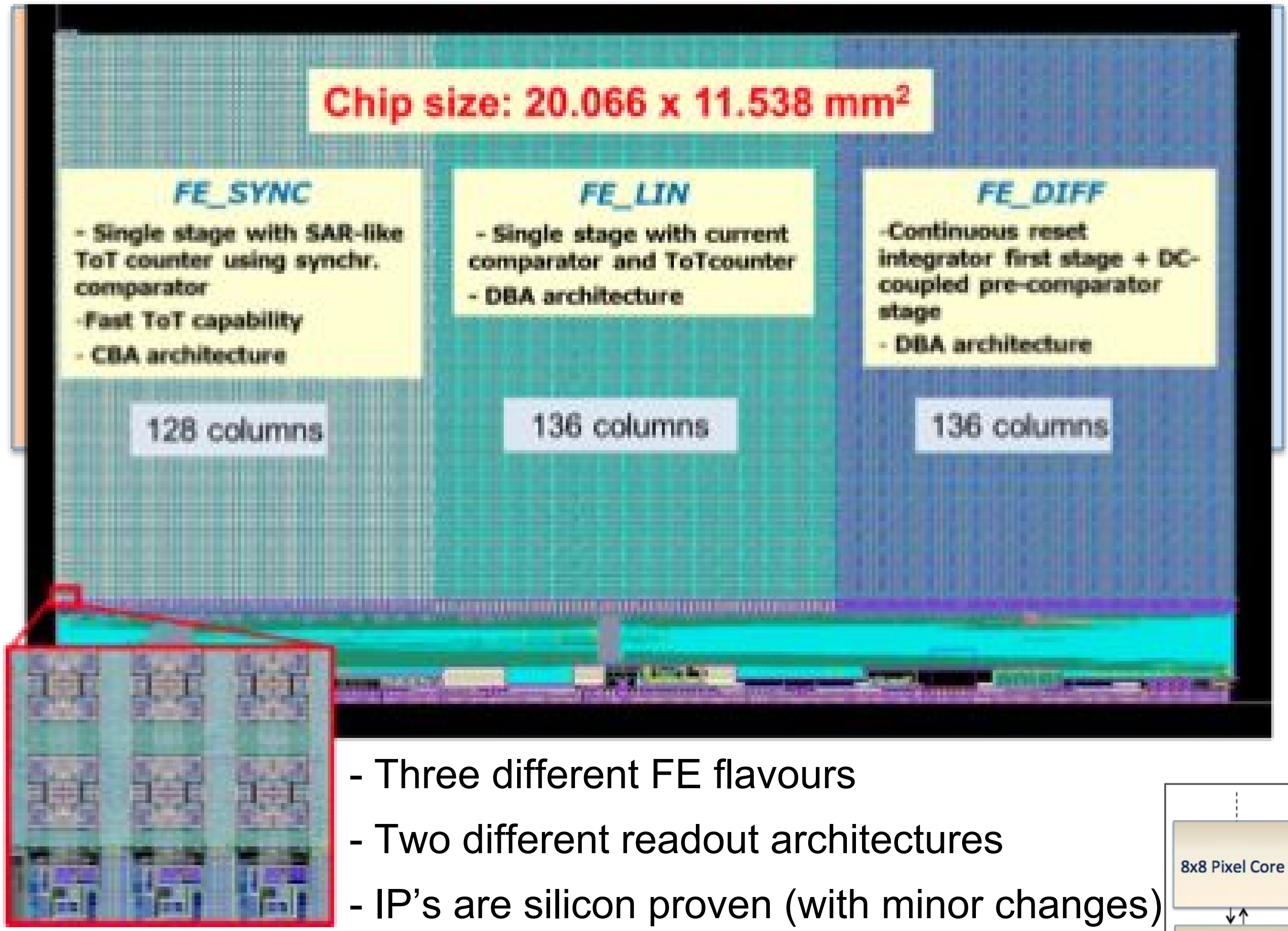
Common engineering run with CMS MPA to share a full reticle and therefore costs.

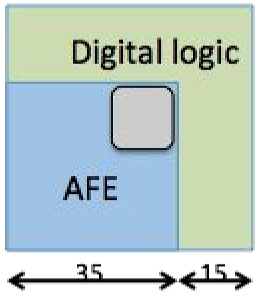
RD53A Chip submitted on August 31th

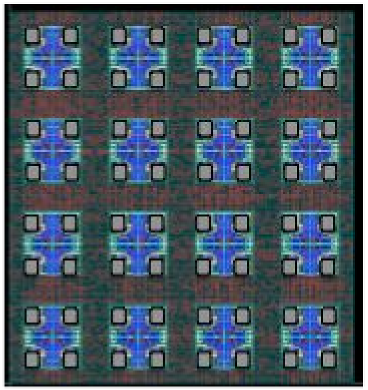
RD53A: Specifications [http://cds.cern.ch/record/2113263]

Technology	65 nm CMOS
Pixels	**Size**: 50 x 50 um^2 **Number**: 192 x 400 = 76800 (50% of production chip)
Detector capacitance	Capacitance: < 100fF (200fF for edge pixels) Leakage: < 10nA (20nA for edge pixels)
Detection threshold	< 600 electrons
In-time threshold	< 1200 electrons
Noise hits	$< 10^{-6}$
Hit rate	< 3 GHz/cm^2 (75 kHz avg. pixel hit rate)
Trigger	**Rate**: Max 1 MHz **Latency**: 12.5 us
Hit loss at max hit rate (in-pixel pile-up)	≤ 1%
Charge resolution	≥ 4 bits ToT (Time over Threshold)
Readout data rate	1-4 links @ 1.28Gbits/s = max 5.12 Gbits/s
Radiation tolerance	500Mrad, 1 x 10^{16} 1Mev eq. n/cm^2 at -15 ºC
SEU affecting whole chip	< 0.05/hr/chip at 1.5GHz/cm^2 particle flux
Power consumption at max hit/trigger rate	< 1W/cm^2 including SLDO losses
Pixel analog/digital current	4 uA / 4 uA
Temperature range	-40ºC ÷ 40ºC

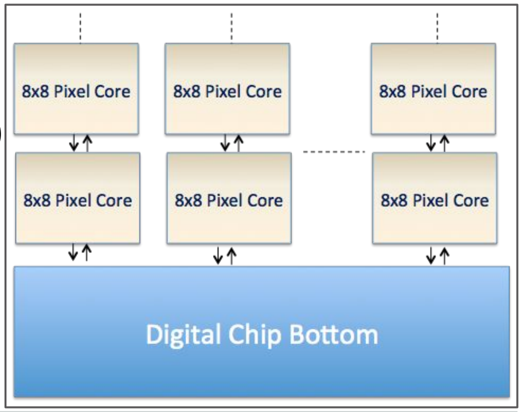
Analog FE's and Pixel organization


Chip size: 20.066 x 11.538 mm²
FE_SYNC
- Single stage with SAR-like ToT counter using synchr. comparator
-Fast ToT capability
- CBA architecture
FE_LIN
- Single stage with current comparator and ToTcounter
- DBA architecture
FE_DIFF
-Continuous reset integrator first stage + DC-coupled pre-comparator stage
- DBA architecture
128 columns
136 columns
136 columns





- Three different FE flavours
- Two different readout architectures
- IP's are silicon proven (with minor changes)

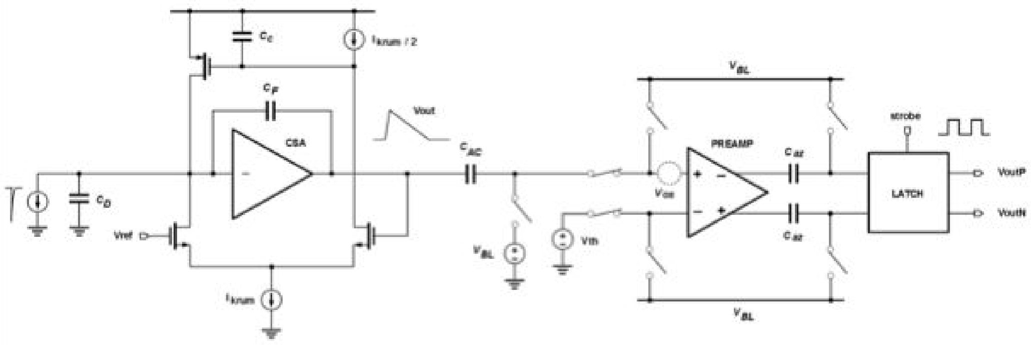


- Common calibration circuitry:
  - Calibration pulses generated via commands starting from 2 voltages.
  - Different operation modes (step/pulse, single pixel and neighbours).
  - Pulses can be phase shifted with a global fine delay (640 MHz clk).

Three different Front End versions

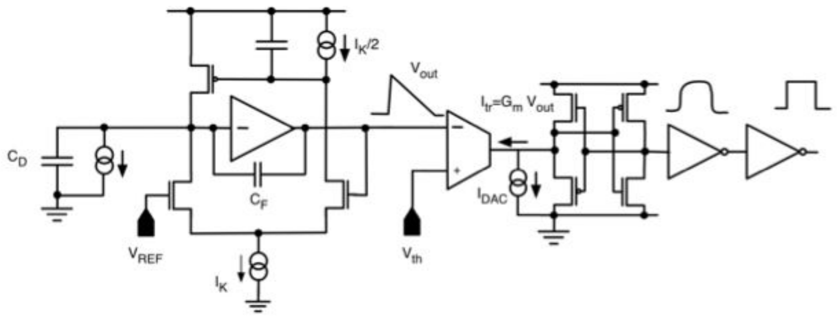
Common to all Front Ends:

- Low-noise, low-power, fast, small
- Threshold adjusting with global 10-bit DAC



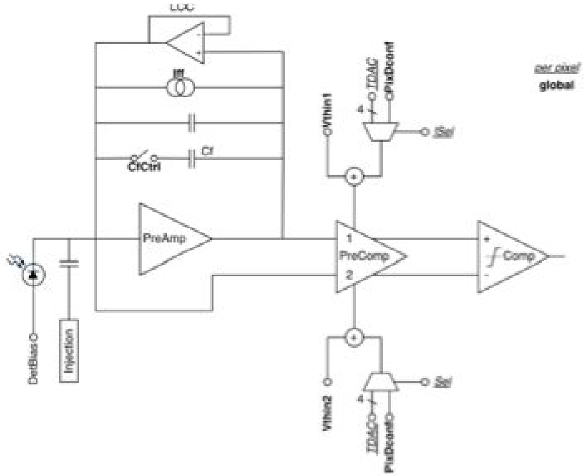
Synchronous Front End:

- telescopic cascode CSA (Krummenacher feedback) for linear Time Over Threshold (ToT)
- Synchronous hit discriminator with track-and-latch comp
- Autozeroing circuitry for threshold trimming
- Variable 40-to-320 MHz Fast ToT counter using a latch



Linear Front End:

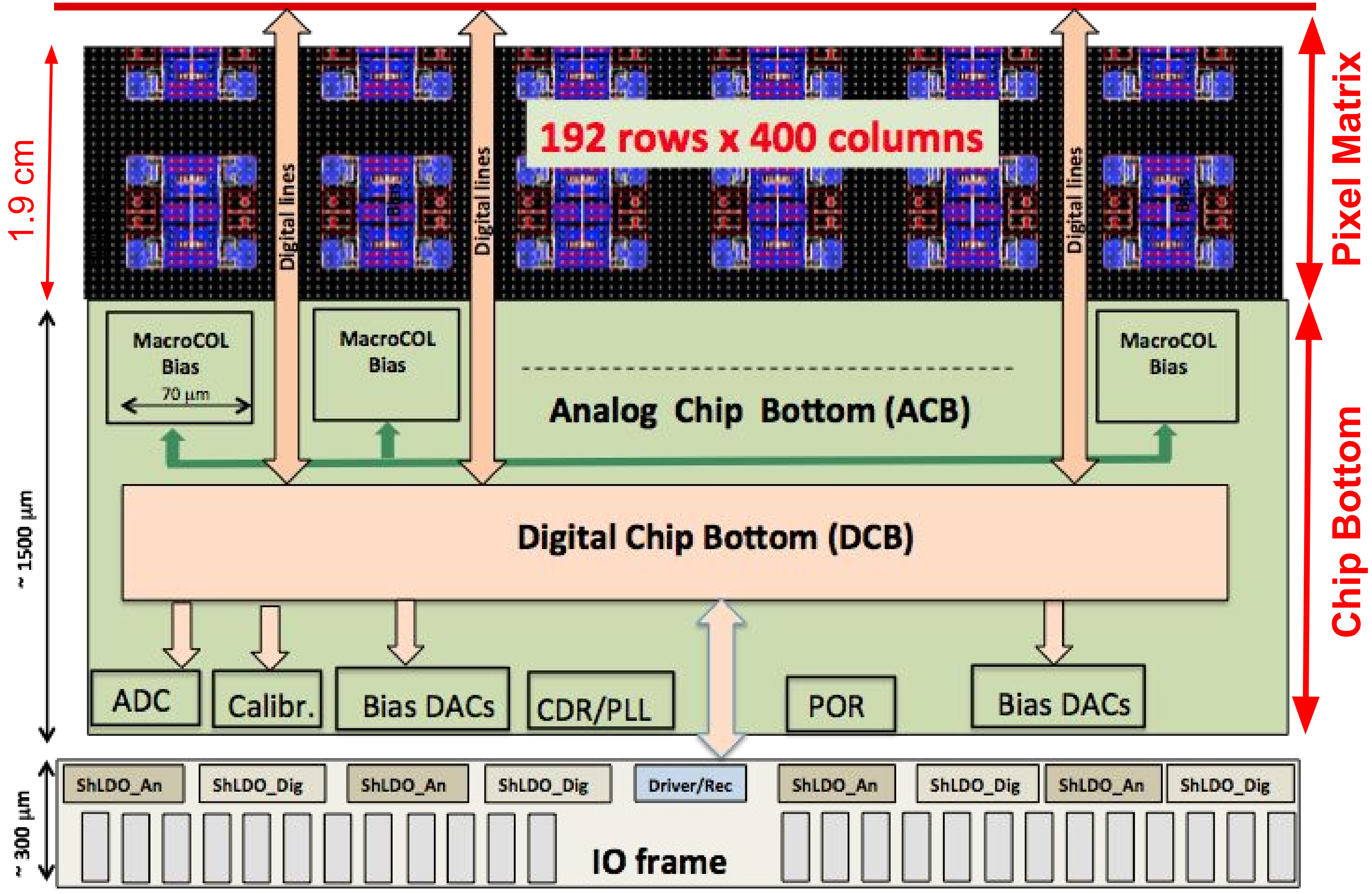
- Single stage amplification for minimum power dissipation
- Krummenacher feedback to comply with the expected large increase in the detector leakage current
- High speed, low power current comparator
- 4-bit local DAC for threshold tuningDAC



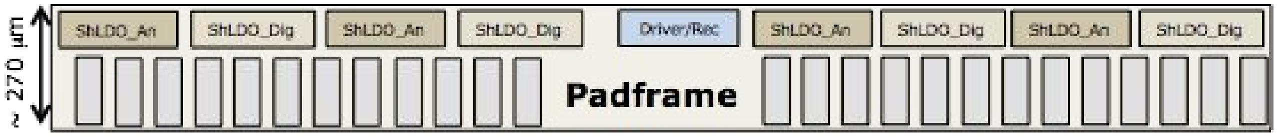
Differential Front End:

- Continuous reset integrator first stage with DC-coupled pre-comparator stage
- Two-stage open loop, fully differential input comparator
- Leakage current compensation like in FE-I4 (ATLAS IBL FE chip)
- Threshold adjusting with two per pixel 4-bit DACs

RD53A: Functional Floorplan



PadFrame



- **MacroBlock containing**:
  - I/O Pads + ESD protections
  - compatible with Through Silicon Vias (TSV)
  - Shunt LDO voltage regulators
  - Drivers and Receivers for Input/Output signals

- **Main characteristics are**:
  - 198 pads with 100 um pitch
  - 58 um x 86 um passivation openings for reliable wirebonding
  - SLVS drivers and receivers
  - One input port with Clock and Data encoded on one 160 Mbit/s line
  - 1, 2 or 4 1.280 Gbit/s output lines, totalling 5.12 Gb/s output bandwidth

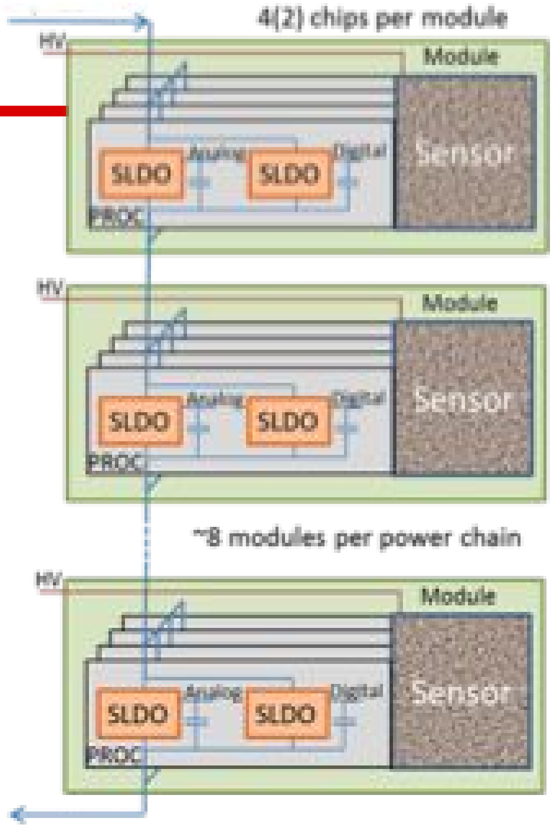
- **Top padframe**:
  - Will not be present in production chip
  - Contains analog pads for testing and monitoring different FE features
  - All different FE architectures have some dedicated test/monitoring points

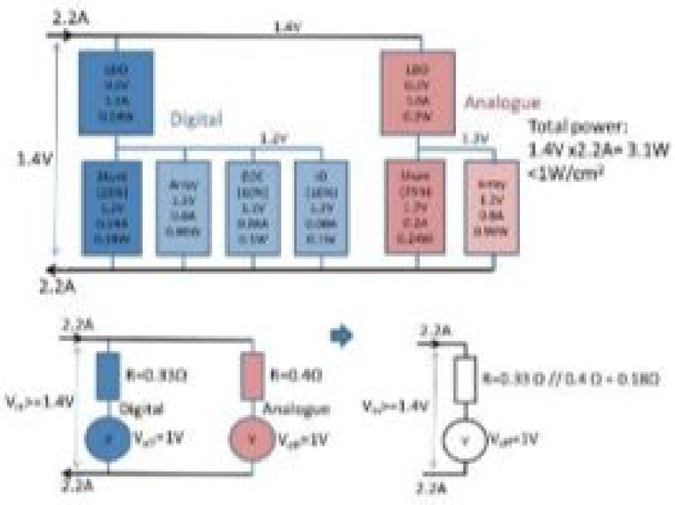
Serial Powering

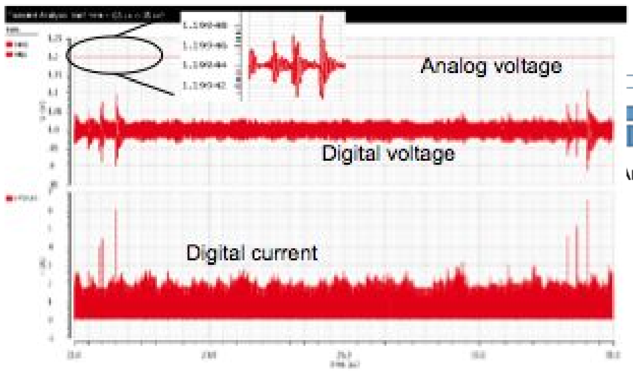
- **Serial Powering** is needed for low mass power distribution:
  - Constant current to power the chips in series
  - Based on ShuntLDO (sized for final production chip)
  - 2 ShuntLDOs (one for analog and one for digital internal rails) with 4 active pads associated with each block
- **Three operation modes**:

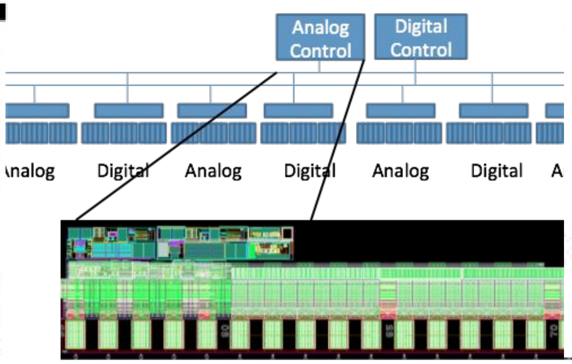
- ShuntLDO:	constant input current	-> locally regulated VDD
- LDO (shunt off):	external voltage	-> locally regulated VDD
- External VDD:	ShuntLDO completely bypassed	

- **Based on prototypes:**
  - They work as simulated for radiation, noise and failure modes
  - Configurable voltage offset of SLDO impedance
  - Improved control loop for stability
  - Full chip power model





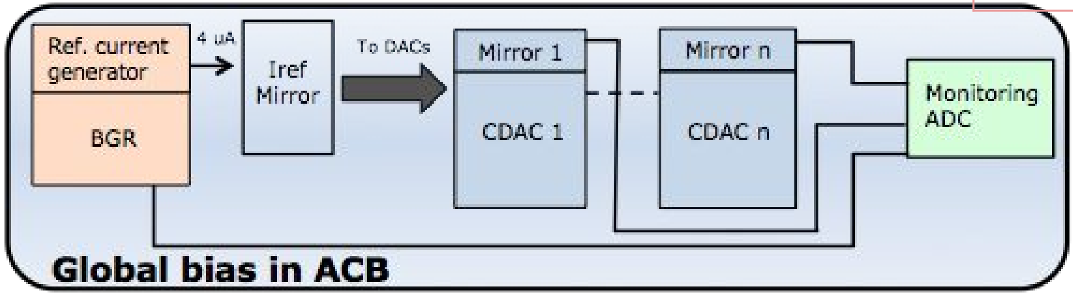




Analog Chip Bottom (ACB)

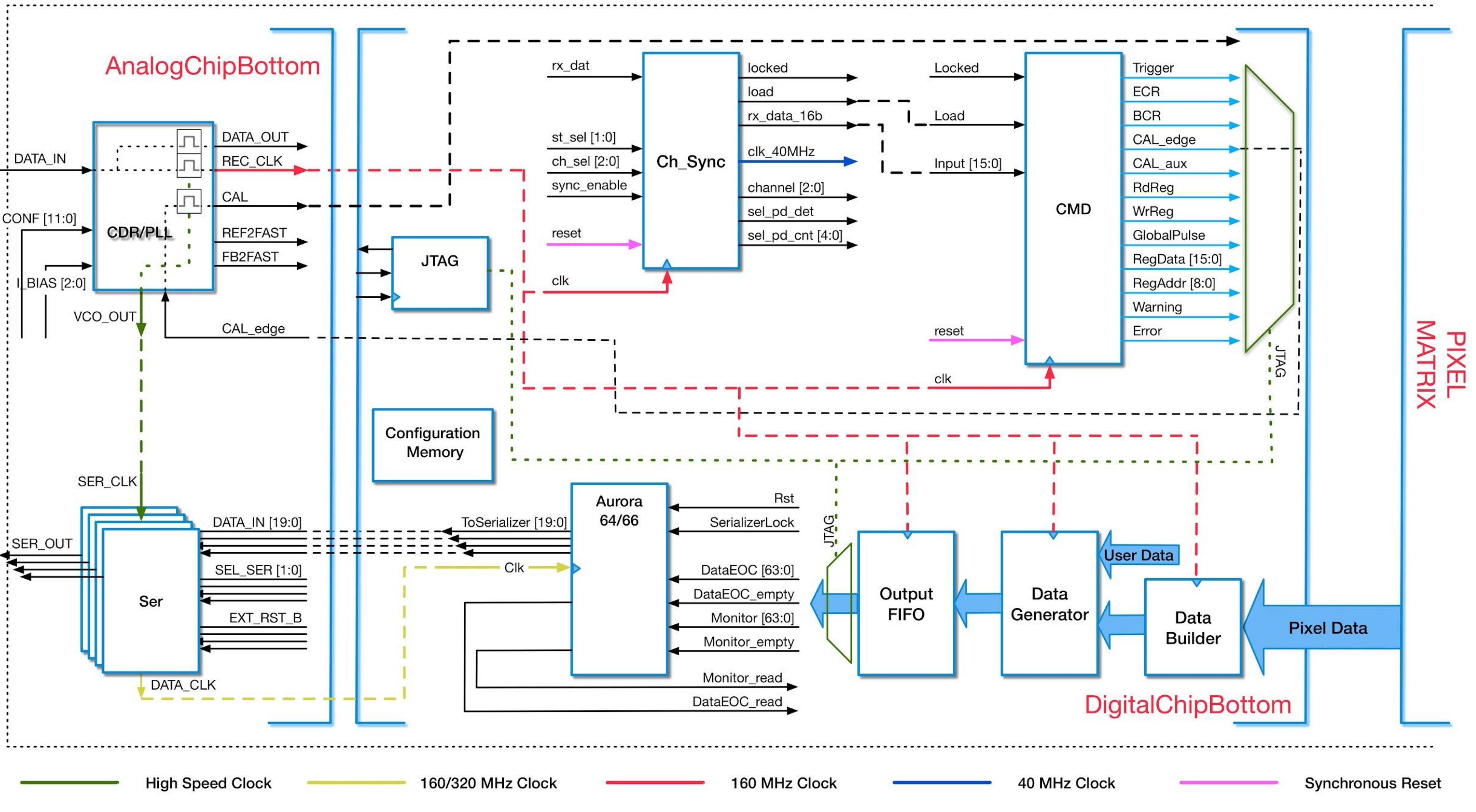
- **MacroBlock containing**:
  - All analog IPs are grouped in this block
  - Each block is prototyped, tested and irradiated
- **Global Biasing**:
  It is a two stage structure:
  - Global V/I provided by 10-bit current steering DACs in ACB
  - Current scaling biases are then distributed to MacroColumnBias cells placed abutted to each double column structure
  - They are distributed in parallel to all pixels
- **Monitoring block**:
  - All voltages and currents can be monitored
  - Selection and readout via dedicated command

Building Blocks	Function
12-bit Monitoring ADC	Monitoring
10-bit current DAC	Bias
12-bit voltage DAC	Calibration
Bandgap reference	Bias
POR	Power On Reset
CDR (PLL)	Clock and Data Recovery
Temp. and rad. Sensors	Monitoring
Analog buffer	Calibration/Monitoring
Ring Oscillators	Monitoring

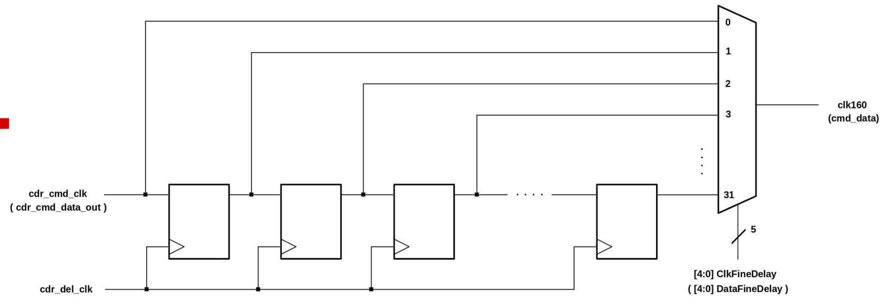


- **Simulation**:
  - All blocks were simulated using Mixed signal simulation
  - Stimuli via simulation framework
  - Analog simulation of the block

Data flow: Block Diagram



Fine Delays, Reset Strategy & Bypass circuitry

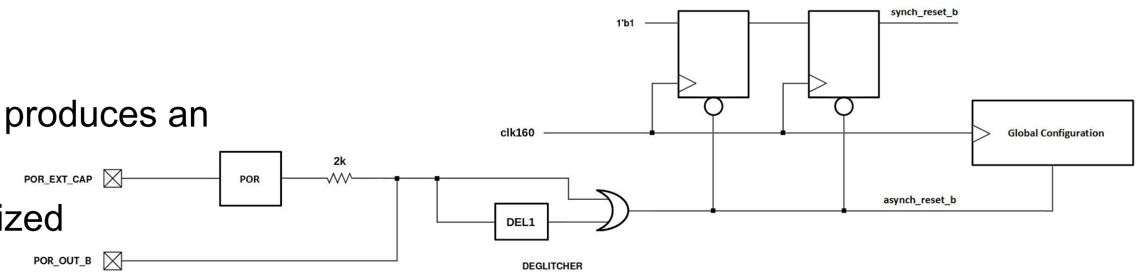


FineDelays:

Input Data, main Clock and the Cal Edge pulse are routed close to the CDR block in order to be able delay them in 16 steps of ~1.5ns using the 640 MHz clock. After being delayed the clock can also be inverted before being sent to the Channel Synchronizer.

- 40 MHz clock is generated, and aligned with the correct phase of the 160 MHz clock, in the Channel Synchronizer.

Reset:

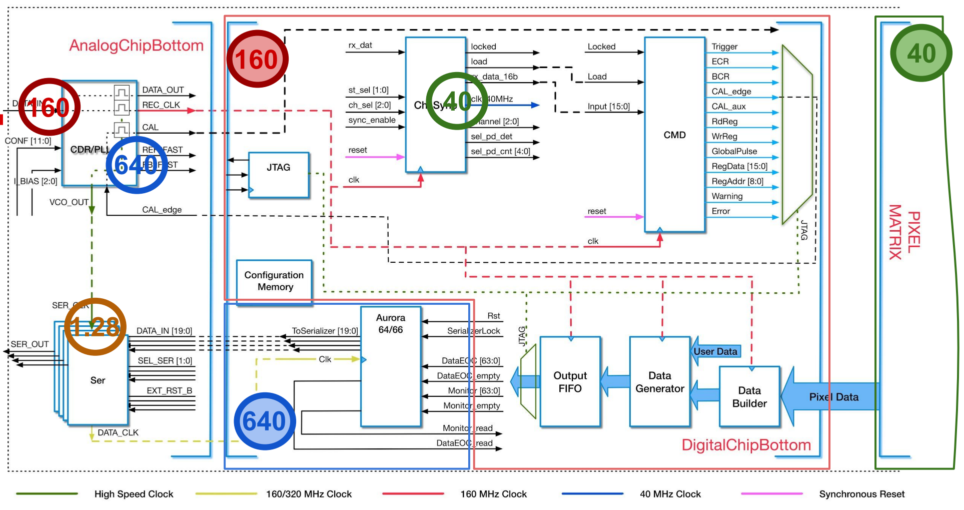


- We have a PowerOnReset circuit that produces an asynchronous active low reset.
- The signal is deglitched and synchronized before being used in the chip.
- Only GlobalConfiguration uses the asynchronous Reset.

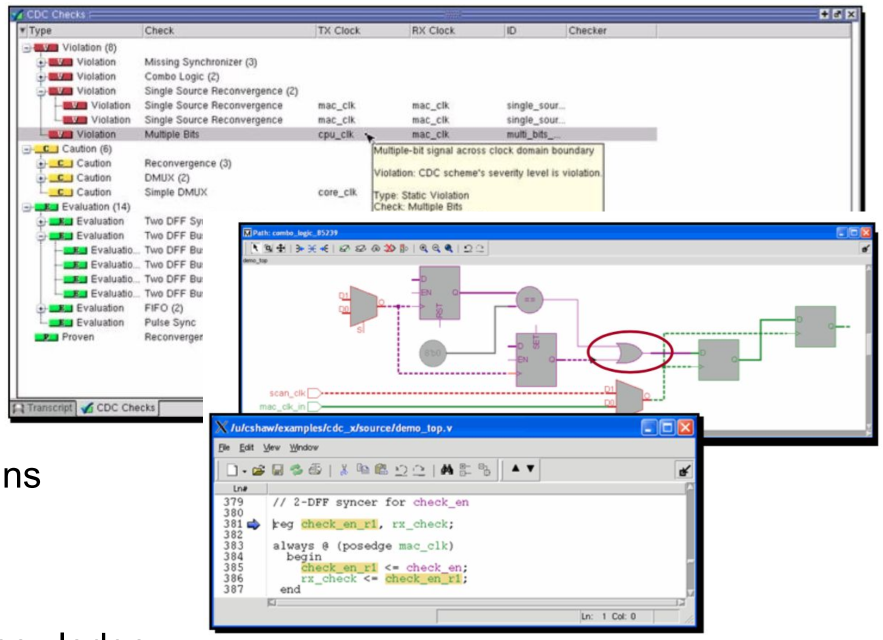
Bypass circuitry:

- All major blocks can be individually bypassed in order to be able to “fully” operate the chip even if they are bypassed.
- Bypass is always performed via dedicated pads, whilst different operation modes are configurable.

Clock Domains



- **System Clock**:
  Data / clock are encoded on a single line
  - 160 MHz is used to encode data
- **PLL / CDR**:
  Recovers 160 MHz clock from Data.
  - Generates 160 MHz System Clock.
  - Generates 640 MHz clock used by Aurora, and to delay Data, Clock and Calibration Pulse.
  - Generates 1.28 GHz clock used by Serializers
- **Channel Synchronizer**:
  Aligns Data with 40 MHz machine clock.
  - Generates 40 MHz Clock, in sync with machine clock, distributed to the Pixel Matrix.
- **Clock Domain Crossing**:
  Critical part of the design.
  - All Reset, Data and Control lines crossing clock domains have to be carefully designed with dedicated circuitry.
  - Asynchronous dual clock fifo, single-bit synchronizer, multi-bit synchronizer with (and without) feedback acknowledge.
  - All blocks have been tested with Mentor CDC formal verification tool.


CDC Checks

Type	Check	TX Clock	RX Clock	ID	Checker
Violation (8)					
Violation	Missing Synchronizer (3)				
Violation	Combo Logic (2)				
Violation	Single Source Reconvergence (2)				
Violation	Single Source Reconvergence	mac_clk	mac_clk	single_sour...	
Violation	Single Source Reconvergence	mac_clk	mac_clk	single_sour...	
Violation	Multiple Bits	cpu_clk	mac_clk	multi_bits_...	
Caution (6)					
Caution	Reconvergence (3)				
Caution	DMUX (2)				
Caution	Simple DMUX	core_clk			
Evaluation (14)					
Evaluation	Two DFF Sy				
Evaluation	Two DFF Bu				
Evaluatio...	Two DFF Bu				
Evaluatio...	Two DFF Bu				
Evaluatio...	Two DFF Bu				
Evaluatio...	Two DFF Bu				
Evaluation	FIFO (2)				
Evaluation	Pulse Sync				
Proven	Reconverger				

Multiple-bit signal across clock domain boundary
Violation: CDC scheme's severity level is violation.
Type: Static Violation
Check: Multiple Bits

Transcript | CDC Checks

Path: combo_logic_85239

/u/cshaw/examples/cdc_x/source/demo_top.v

```
// 2-DFF syncer for check_en

reg check_en_r1, rx_check;

always @ (posedge mac_clk)
  begin
    check_en_r1 <= check_en;
    rx_check <= check_en_r1;
 end
```

Ln: 1 Col: 0

Input Protocol

- **Frame based Custom Protocol**:
  Allows for DC balancing, short run length and efficient bandwidth usage
- **Continuous stream of frames**:
  A Frame is 16-bit long and is made out of two 8-bit (dc balanced) symbols
- **Priority list of commands built in**:
  Allows to continuously send configuration data, and interrupt the data stream in order to send Triggers or Sync frames, without having to restart sending configuration data.
- **Different type of Frames**:
  - Sync: Used to keep the channel locked to the 40 MHz machine clock
  - Trigger: Encodes up to 4 consecutive triggers plus a 5-bit Tag to identify each trigger
  - Commands: BCR, ECR, Cal, GlobalPulse, WrReg, RdReg, Null
    There are single frame commands and commands that accept Data
    All commands that carry data contain a ChipId field so only addressed chip or, via a broadcast bit, all chips connected to the same input line will respond.
  - Data Frames: 2 consecutive 8-bit symbols that allow to encode 5 + 5 bits of information
- **Data redundancy**: Allows for minimal Error detection/correction.
- **Channel locking**:
  At chip startup one has to lock the channel sending Sync frames. After a successful lock commands are passed to the Command Decoder.
  In case of Sync loss one has to reestablish the channel lock.

Global and Pixel Configuration

- **GlobalConfiguration** (~2000 configuration bits):
  All chip configuration is stored in a 16-bit wide Global Register Bank.
  - There are 137 addresses
  - Each register can be written with a WrReg command and read back with a RdReg command.
  - The whole Global Configuration Register has a default startup value that is selected by the POR circuitry. In order to perform this operation the Reset line of this block is asynchronous.
- **Special Registers**:
  Some registers have a special meaning:
  - Diagnostic counters that are ReadOnly and hold all Error/Warning conditions that might occur inside the chip. They are automatically erased if you write to them.
  - Monitoring Register: It's an interface to the Monitoring block and allows to readout the sampled data coming from the currently selected monitoring voltage or current.
  - Ring oscillator counters, used just to read out the value of the selected Ring Oscillator.
  - Pixel Register (#0). Allows to access Pixel Configuration bits, both for writing and reading.
- **Pixel Configuration** (more than 600k configuration bits):
  Each Pixel can have, based on flavour, up to 8 configuration bits.
  - Default Pixel configuration (for low power startup) is selectable via a multiplexer.
  - Two pixels (a column pair) are read and written at the same time.
  - Rd/Wrt to pixels is performed selecting in GlobalConf what Row/Column one wants to access.
  - There is an AutoIncrement mode that allows fast writing/reading of consecutive pixel values
- **Full chip configuration**:
  Configuring the whole chip takes ~ 30 ms, allowing continuous configuration refresh

Self generated SystemVerilog code

- **Running Specifications**:
  As always happens we had to write the code to correctly implement the register table with all it's default values, but the register definition was something changing rather often as people added features or chose different default values based on simulations.

  Having over 100 register definitions it would have been a nightmare.

  Decided to have a google spreadsheet with all the register definitions, names, addresses and default values.

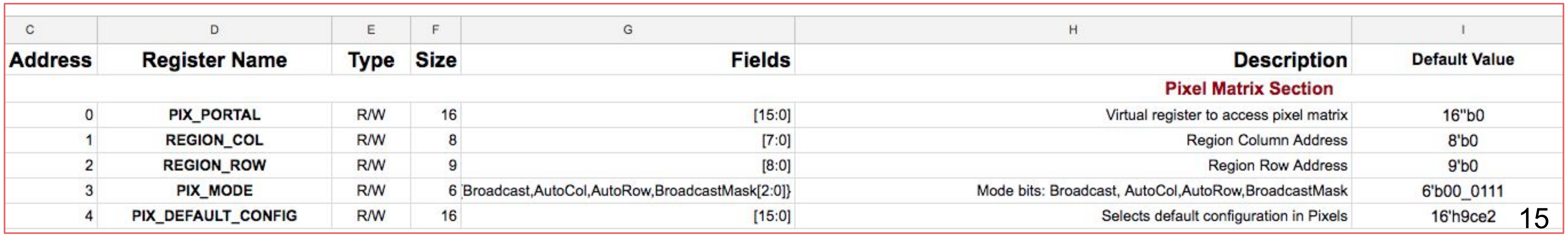
  All the designer team had access to it and it was constantly updated.

  Structured the SystemVerilog code of the Global Configuration to include two files

  - Actual register definition
  - Mapping of the Registers to signals used inside the chip

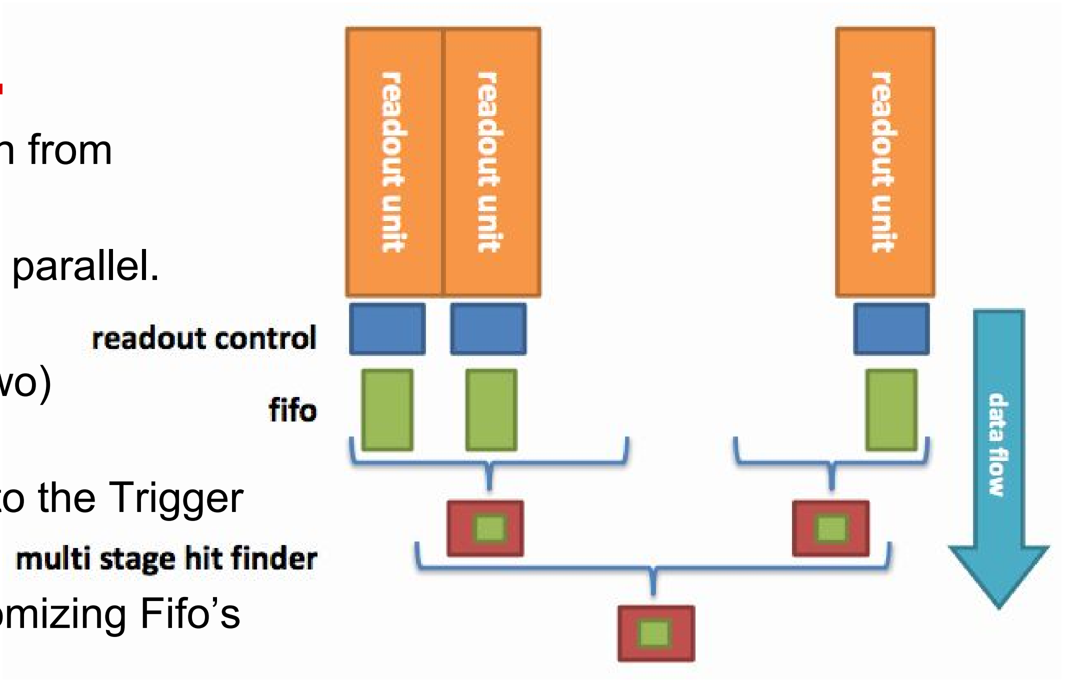
  Developed a small python script that could automatically access the online spreadsheet, get all the data contained in it, and automatically generate the SystemVerilog code of the two included files!

  Only minimal user interaction was needed to recreate a working and up to date code.



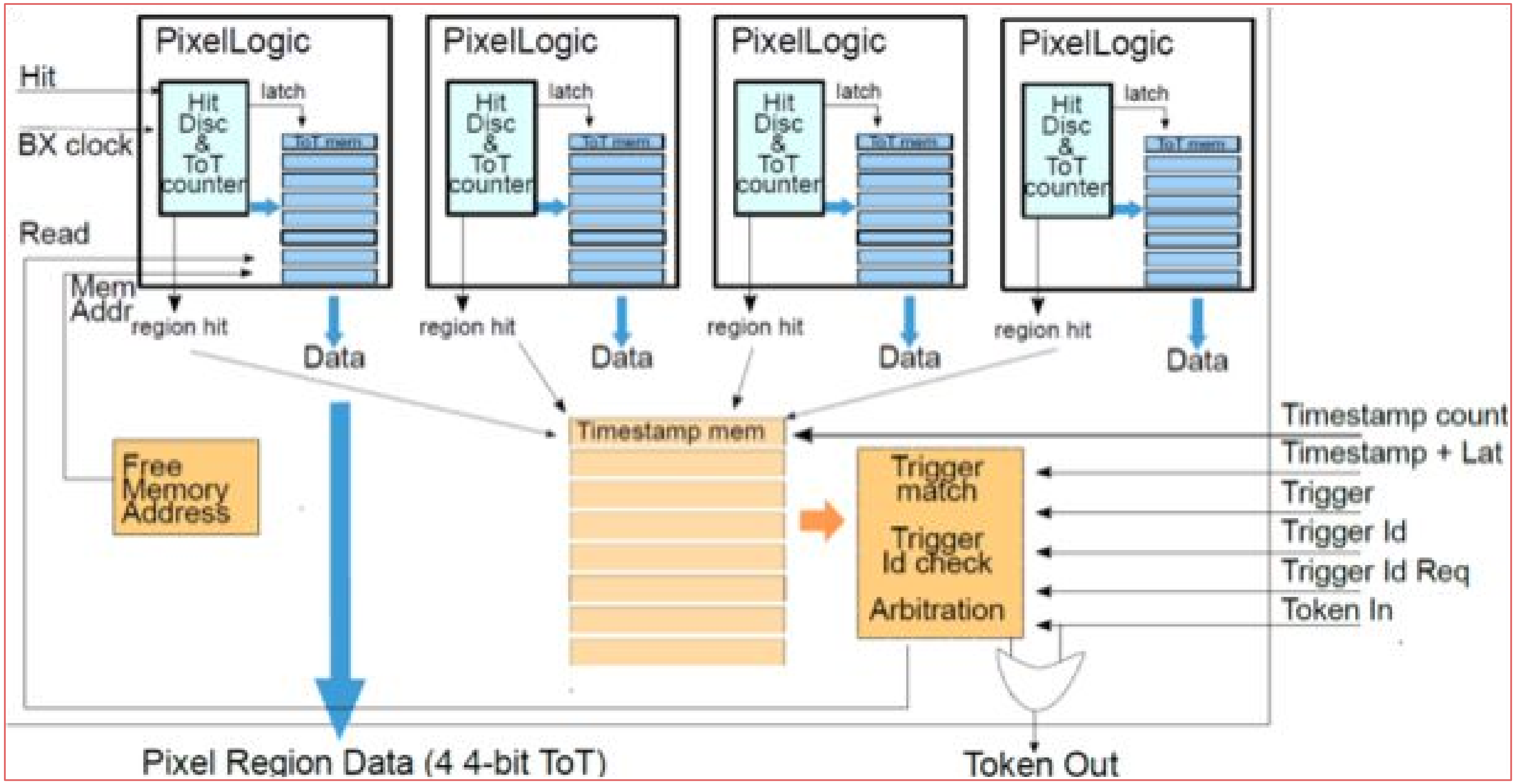
C	D	E	F	G	H	I
Address	Register Name	Type	Size	Fields	Description	Default Value
					Pixel Matrix Section	
0	PIX_PORTAL	R/W	16	[15:0]	Virtual register to access pixel matrix	16"b0
1	REGION_COL	R/W	8	[7:0]	Region Column Address	8'b0
2	REGION_ROW	R/W	9	[8:0]	Region Row Address	9'b0
3	PIX_MODE	R/W	6	Broadcast,AutoCol,AutoRow,BroadcastMask[2:0]}	Mode bits: Broadcast, AutoCol,AutoRow,BroadcastMask	6'b00_0111
4	PIX_DEFAULT_CONFIG	R/W	16	[15:0]	Selects default configuration in Pixels	16'h9ce2

Data Builder



- This block performs Data Collection from the Pixel Matrix.
- All Readout Units are processed in parallel.
- The used mechanism is a token based readout that takes one (or two) 40 MHz clock cycles to complete.
- Each time a new Trigger is added to the Trigger Fifo a new token is generated.
- Read out Data is stored in derandomizing Fifo's at the EndOfColumn.
- Trigger is removed from the Fifo.
- Data is then gathered together using a multistage hit finder.
- Hit Finder performs an event based hit processing that allows to have all data for a particular event to be ordered by column and then by row.
- Data ordering is not strictly required unless one wants to perform some sort of data compression (this feature did not go into the RD53A prototype).
- Data collection and ordering does not affect the readout speed, but only the Fifo's size that has to be chose according to expected hit rate.
- All collected data, sorted by TriggerId, is stored in a Fifo, already formatted according to the Output protocol, waiting to be read out.

Distributed Buffer Architecture



- The 64x64 core is composed of 4 Pixel Regions, each of them managing 16 independent front-ends.

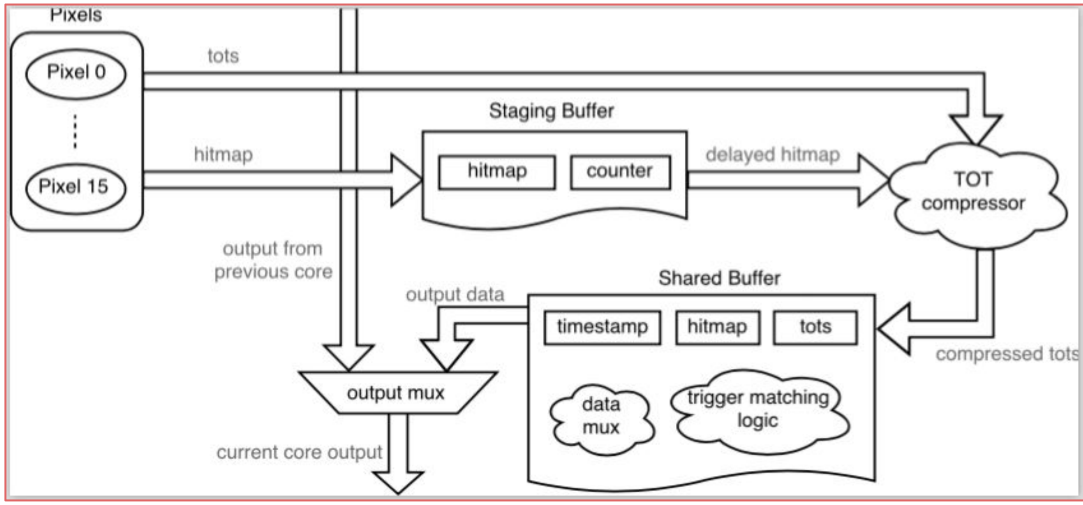
- Each Pixel Region has its own Memory storage elements (a latch based memory)

- A global TimeStamp memory is used to store Hit Timestamps waiting for Trigger match.

- This architecture uses more area but is more efficient in terms of signal routing.

Centralized Buffer Architecture

- The Centralised Buffer Architecture is based on the pixel regions of CHIPIX65.
- The 64x64 core is composed of 4 Pixel Regions, each of them managing 16 independent front-ends.
- Designed to maximise resource sharing among the pixels, the Pixel Region reads, stores and elaborates the TOTs of each Pixel.



- Pixels have independent logic: if hit, TOT of the event is evaluated, latched, and self-reset.
- If any of the pixels in the region is hit, the Pixel Region stores the hit map in a Staging Buffer.
- At evaluation time, TOTs in the latches are compressed and stored in a shared buffer.
- Compression is lossy and includes only the first 8 TOTs according to a priority queue. This ensures negligible TOT losses (0.01%)
- The Compression stage needs all TOTs to be ready. This is done by the synchronisation stage.
- Trigger matching and output multiplexing has been adapted to match the RD53A format, in order to minimise changes at the Chip Periphery.
- An adapter stage unpacks the compressed data and rebuild the packets, which are then processed by the Data Building stage.

User Generated Data

In addition to Pixel Data RD53A produces many different types of User Generated Data:

- **Monitoring Data**:
  All sampled data of the monitoring block have to be sent out of the chip.
- **Warning / Error conditions:**
  During operation all Warning / Error conditions are recorded.
- **Configuration Registers**:
  Check values of configuration registers.
- **Pixel Configuration Registers**:
  Check values of single pixel configuration registers.

All this data is mapped to dedicated configuration registers that can be accessed via RdReg.

We have 4 output links and 2 register values can be read out in each data frame.

We implemented 8 different data fifo's to store the user generated data waiting to be read.

Each time a new value is generated the corresponding Fifo is updated and erased on readout.

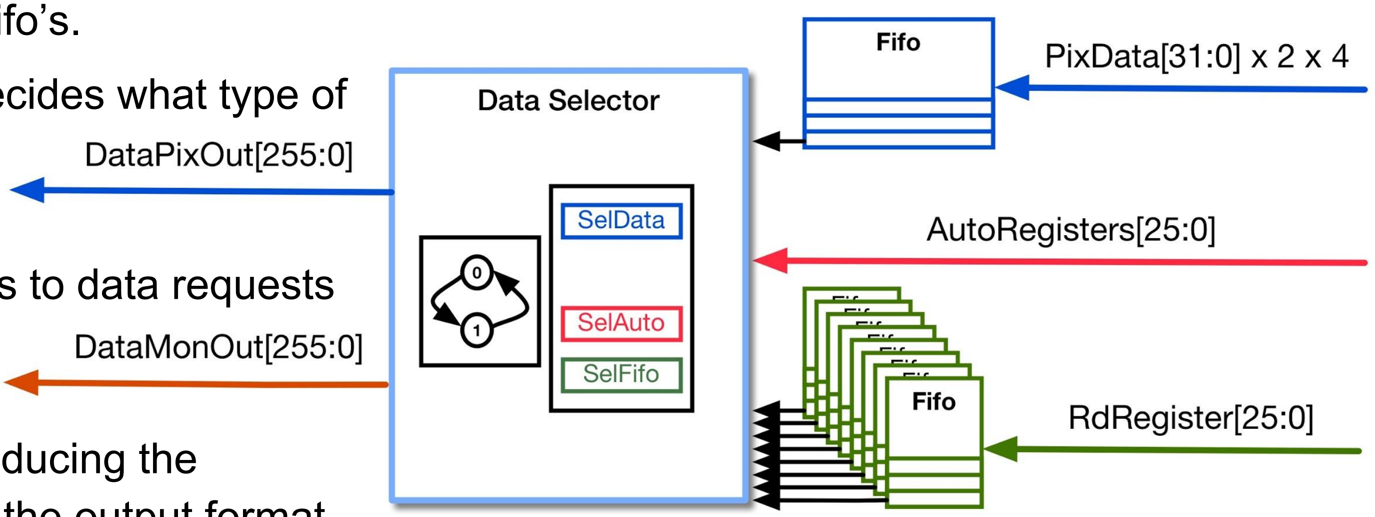
A direct RdReg command stores its data in each Fifo using a round-robin table.

In addition there are some Registers (AutoRegisters) that can be programmed to point to the registers that have to be automatically read out in absence of direct read commands.

This allows to always have data to be read out even without supplying a RdReg command.

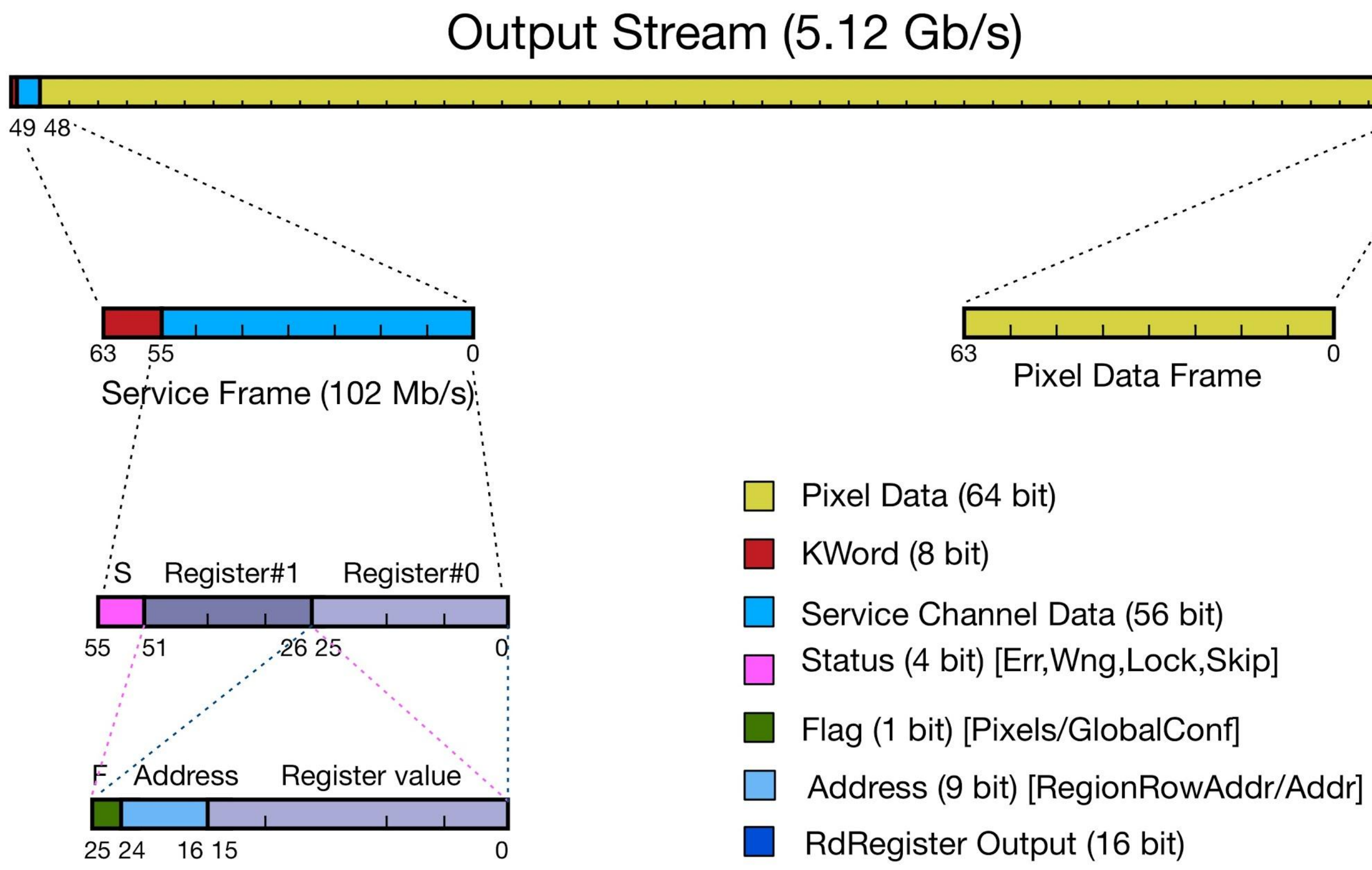
Data Generator

- Data coming from Pixels and User generated data types [Configuration/Error/Warning/Monitor] are stored in dedicated Fifo's.
- A Data Selector circuit decides what type of Data to send out at each clock cycle.
- A state machine responds to data requests and chooses between Pixel Data and User Generated Data producing the correct data according to the output format.



- When User Data is selected and the Fifo's containing RdReg command responses are not empty this data is chosen and formetted according to the output format.
- If all Fifo's are empty output data is formatted using the values contained in the AutoRegisters.
- This mechanism allows to always have data to be sent on the User Generated Data channel.
- The output data depends on the number of chosen output links:
    - If one channel is selected only DataPixOut[63:0] and DataMonOut[63:0] are read.
    - If two channels are selected DataPixOut[127:0] and DataMonOut[127:0] are read.
    - If all channels are selected full DataPixOut and DataMonOut are read.

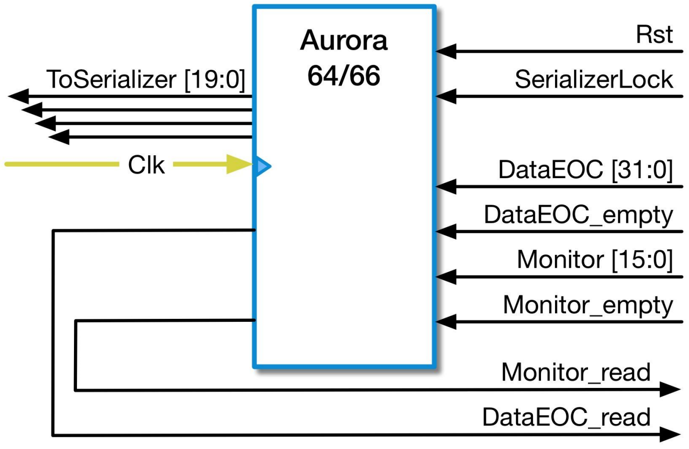
Output Data Stream


Output Stream (5.12 Gb/s)
49 48
0
63 55
0
Service Frame (102 Mb/s)
63
0
Pixel Data Frame
S
Register#1
Register#0
55
51
26 25
0
F
Address
Register value
25 24
16 15
0
Pixel Data (64 bit)
KWord (8 bit)
Service Channel Data (56 bit)
Status (4 bit) [Err,Wng,Lock,Skip]
Flag (1 bit) [Pixels/GlobalConf]
Address (9 bit) [RegionRowAddr/Addr]
RdRegister Output (16 bit)

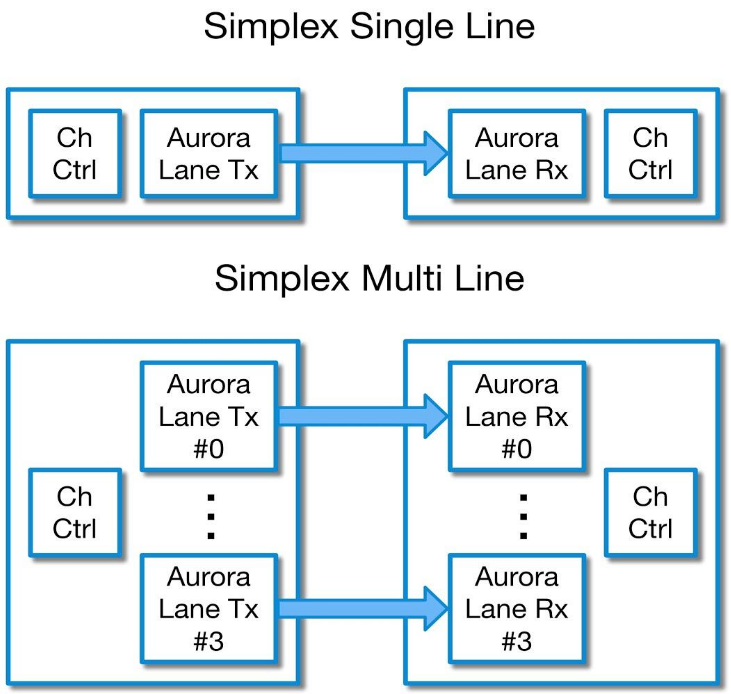
The Output Data Stream is splitted in two separated channels:

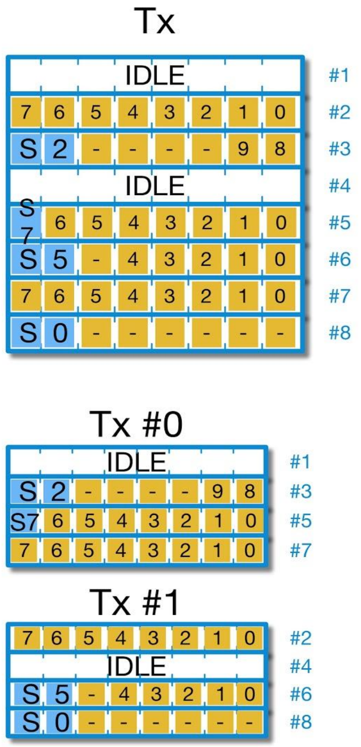
1. Pixel Data: 98% of 5.12 Gb/s (default value). Data coming from Pixels.
2. User Data: ~ 102 Mb/s. Data from RdReg commands, Warning/Errors, Monitor Data.
3. Ratio between Pixel Data and User Data is user programmable.

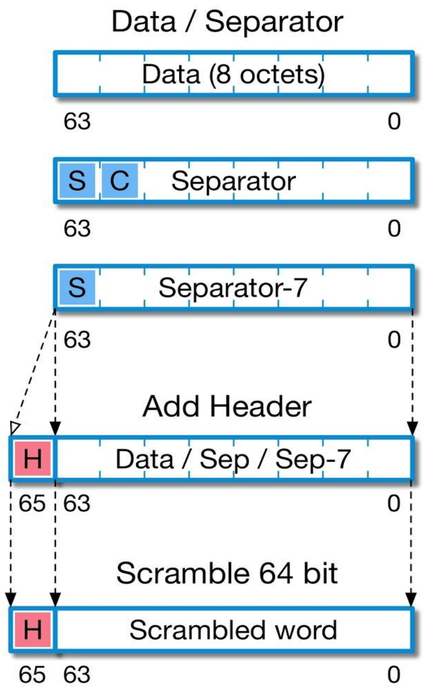
Output Protocol



- **Aurora 64B/66B**:
  Xilinx “standard” protocol with low overhead with Enough flexibility to fit our needs:
  - Being a standard it works “out of the box” on receiving end!
  - Very robust and allows single/multi line communication.
  - Allows to send User Defined Data types that are identified in the data stream by the different Header and by the use of a specific 8-bit K-Word.
  - Strict alignment (same data type is always sent on each active link) is used.
  - Data is then sent to custom Serializers.







Functional Redundancy and JTAG

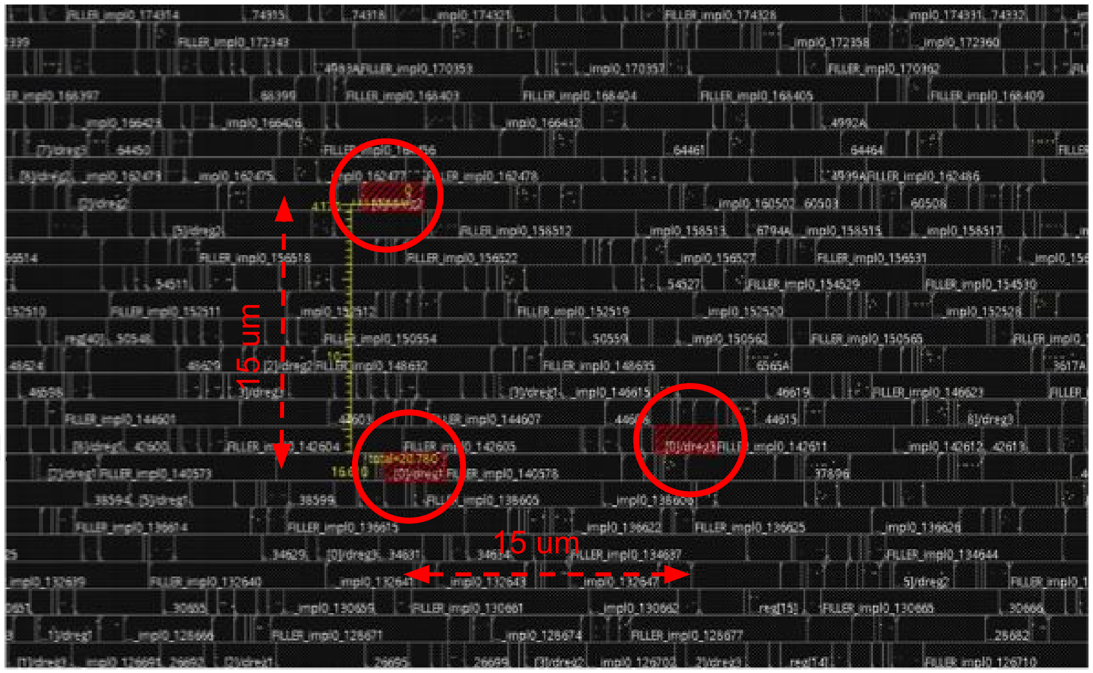
Being RD53A a test chip that will also be used by the sensor community to test detectors and modules that are being developed for both ATLAS and CMS one of the most critical issues we have to face is that we really need to make sure that the chip is able to operate even if some critical component have some major flaw.

We must be able to bypass all critical blocks, and still be able to "fully" operate the chip.

This chip uses JTAG to provide a backup solution for IO and debugging capabilities.

- JTAG is a slow control protocol that can/could be used for:
  "boundary scan", "chip configuration" and "wafer testing" using DFT techniques.
- The idea is to be able to "fully" operate the chip even bypassing completely:
  CRD/PLL, Channel Sync, Command Decoder, Output stage
- JTAG will be disabled by default (via wire bonding) but will eventually allow to:
  1. Bypass the Command Decoder, allowing to configure the whole chip, and to send Commands (replacing the Command Decoder outputs)
  2. Bypass Output Data generation before the Aurora block.
  3. Bypass Output Serializers and use a simple std cell based backup solution.
- Using a dedicated Trigger input it could still be possible to "fully" operate the chip.
- A minimal Scan Chain interface was built in.
- Boundary Scan is also implemented in order to allow spying on some critical regs.

SEU protection of critical blocks



Not a lot of effort has been done,
both in hardware and in simulation,
for addressing SEU issues in this chip.

Global strategy:

- All digital global signals that could cause SEU problems at the chip level are deglitched before being routed.
- No signal triplication has been performed

PixelRegion:

- Triple Modular Redundancy (TMR) cannot be used due to area constraints.
- RD53A relies on fast and continuous configuration refresh in order to mitigate SEU effects.

Digital Chip Bottom:

- All critical signals are deglitched.
- TMR is used for Global Configuration. Automatically added to the design via script.
- New Virtuoso version allows to spread triplicated FF's by a user defined distance (15 um were chosen).

The new input protocol allows for fast and continuous writing of all configuration bits.
The whole chip (including Pixel configuration bits) can be reconfigured in ~30 ms.

SEU tolerant DICE latches, with Interleaved layout, have been characterized and tested and provide a factor 9 protection over standard latches, but did not make it in the RD53A chip.

System verification: SVerilog + Python

- **SystemVerilog**:

  All digital blocks were verified using dedicated tb's done by the Individual designers.

  - Easy for I/O blocks
  - Hard for internal blocks

  Allows for quick checking of the code and first bug squashing.

- **Python framework**:

  Cocotb:

  - Open source, coroutine based cosimulation library for writing VHDL and Verilog testbenches in Python.

  Basil:

  - modular data acquisition system and system testing framework in Python
  - modular readout framework intended to allow simple and fast data acquisition systems (DAQ) design. It consists of different hardware components, FPGA firmware modulus and a Python based control software

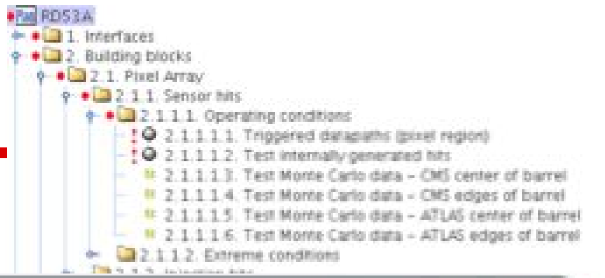
  Allows fast, and yet complex, design debugging due to the very simple python interface.

  Has all needed FPGA firmware components so it allows an easy transition between a simulation environment and the test system that will be used to characterize the chip.

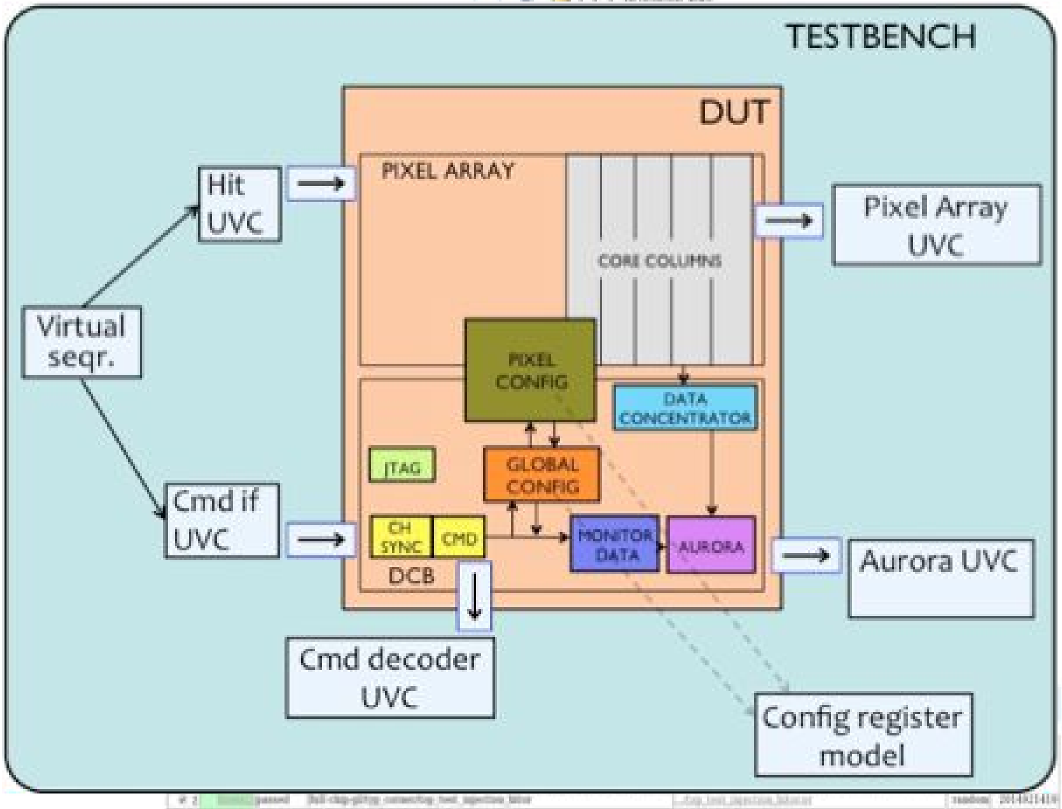
  It was developed for FE-I4 development and testing, and is an upgrade of this project.

System verification: UVM

- **VEPIX53**: CERN developed an UVM based simulation and verification environment supporting design from initial architectural modeling to final verification
  - different kinds of input stimuli (internally generated or Monte Carlo)
  - automated verification features
  - directed test pattern generation support
- Interface verification components (UVCs):
  - Hit (sensor pixel hit data)
  - Command (custom input protocol for control and trigger)
  - Aurora (pixel chip output protocol)
- Module UVCs:
  - Pixel array UVC (reference model, scoreboard, l
  - Command Decoder UVC (monitor, checker)
  - Configuration register model (automatic configuration register verification)
- Rather complex but very effective (regression tests)
- Also used to generate stimuli for Analog simulations



RD53A
1. Interfaces
2. Building blocks
2.1. Pixel Array
2.1.1. Sensor hits
2.1.1.1. Operating conditions
2.1.1.1.1. Triggered datapaths (pixel region)
2.1.1.1.2. Test internally generated hits
2.1.1.1.3. Test Monte Carlo data - CMS center of barrel
2.1.1.1.4. Test Monte Carlo data - CMS edges of barrel
2.1.1.1.5. Test Monte Carlo data - ATLAS center of barrel
2.1.1.1.6. Test Monte Carlo data - ATLAS edges of barrel
2.1.1.2. Extreme conditions

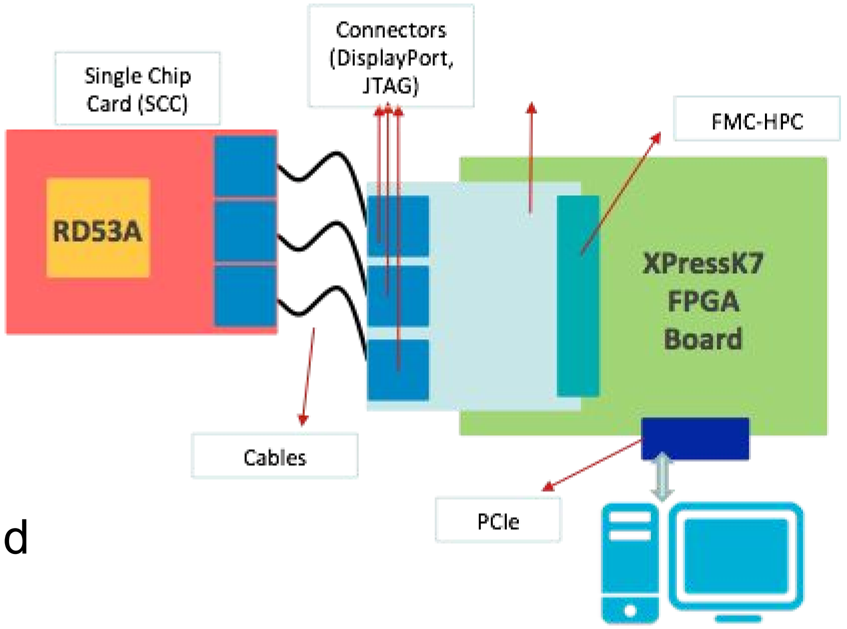


2	[illegible]	passed	[illegible]	[illegible]	random	[illegible]
3	[illegible]	passed	[illegible]	[illegible]	random	[illegible]
4	[illegible]	passed	[illegible]	[illegible]	random	[illegible]
5	[illegible]	passed	[illegible]	[illegible]	random	[illegible]
6	[illegible]	passed	[illegible]	[illegible]	random	[illegible]
7	[illegible]	passed	[illegible]	[illegible]	random	[illegible]
8	[illegible]	passed	[illegible]	[illegible]	random	[illegible]
9	[illegible]	passed	[illegible]	[illegible]	random	[illegible]
10	[illegible]	passed	[illegible]	[illegible]	random	[illegible]
11	[illegible]	passed	[illegible]	[illegible]	random	[illegible]
12	[illegible]	passed	[illegible]	[illegible]	random	[illegible]
13	[illegible]	passed	[illegible]	[illegible]	random	[illegible]
14	[illegible]	passed	[illegible]	[illegible]	random	[illegible]
15	[illegible]	passed	[illegible]	[illegible]	random	[illegible]
16	[illegible]	passed	[illegible]	[illegible]	random	[illegible]
17	[illegible]	passed	[illegible]	[illegible]	random	[illegible]
18	[illegible]	passed	[illegible]	[illegible]	random	[illegible]
19	[illegible]	passed	[illegible]	[illegible]	random	[illegible]

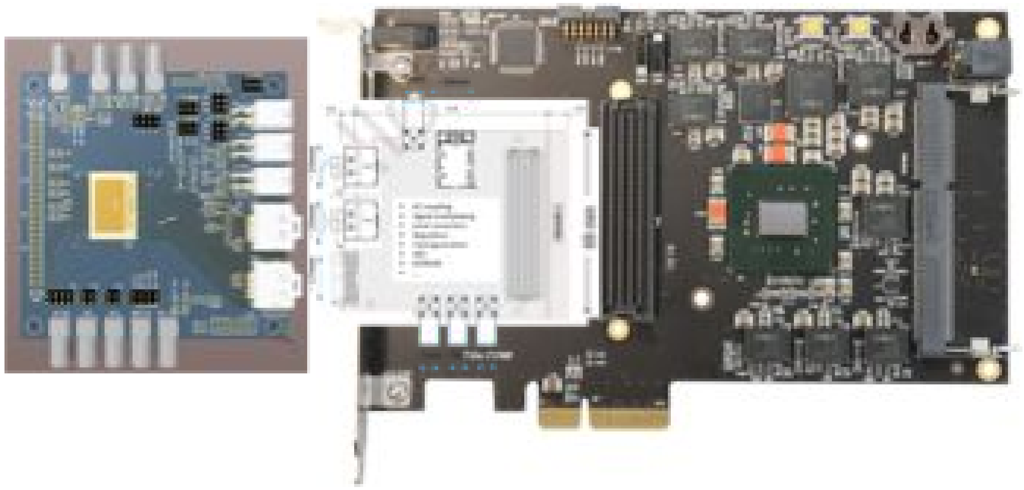
Test Systems (under development)

- Based on PCIe FPGA board (XPressK7) and custom adapter boards.

- Adapter Card to drive/receive FPGA generated signals via DisplayPort Connectors and Jtag.
  Allows for different test scenarios in which one can test a single chip or up to 4 chips.
  - Single chip fully testable
  - 4 chips driven by one command line and read out using 1 output/chip



- Analog monitoring pins



- Firmware already in advanced development.
  - Cocotb + Basil based
  - Complete C++ SW model Of the RD53A chip

- Will be available for all institutes that want to perform chip/sensor characterization

Conclusions

After lot of work the RD53A chip has finally been submitted on August 31st

Chip is expected to be back by beginning of November
(we will have a better estimate in the coming week)

Test systems are almost ready, so immediate testing should start immediately after receiving the chips.

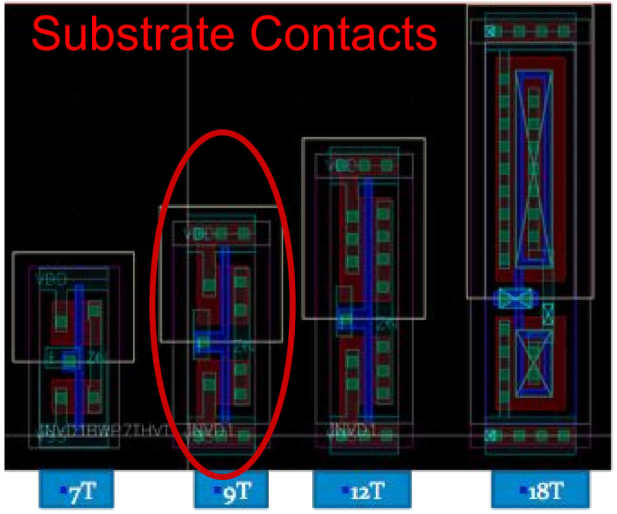
Setting up a Design Team with designers from both ATLAS and CMS, and being able to work all together in one place for 3/4 month for chip finalization, has proven to be a key component for this submission.

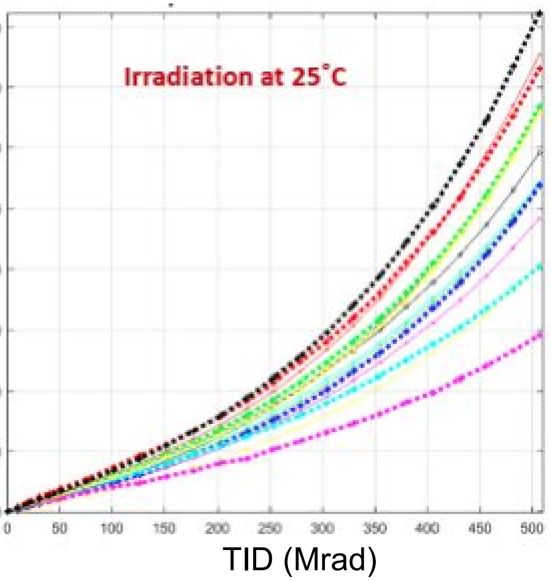
Based on this, the RD53 collaboration, is proposing a common design framework for the final version of ATLAS and CMS Pixel chips.

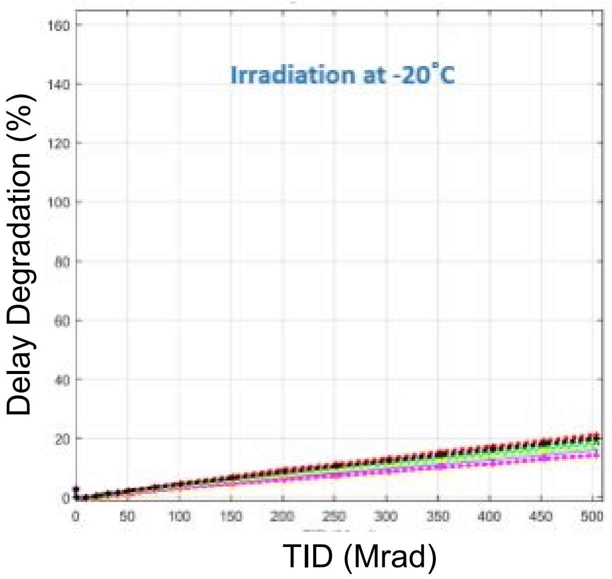
Thanks for your attention!

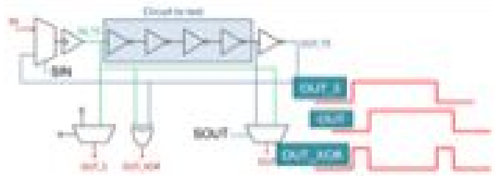
Backup slides

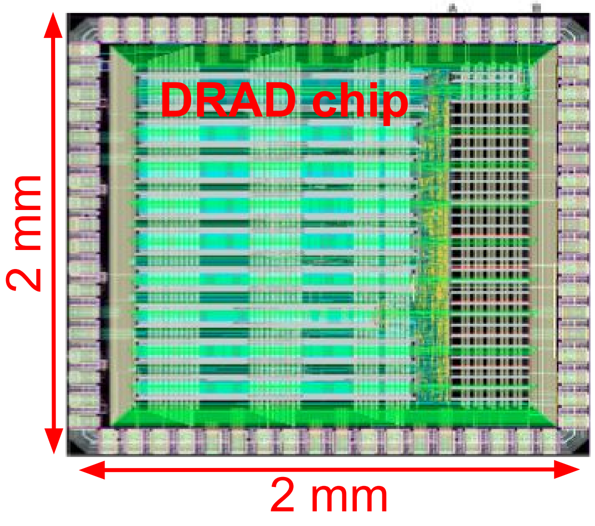
Digital speed degradation

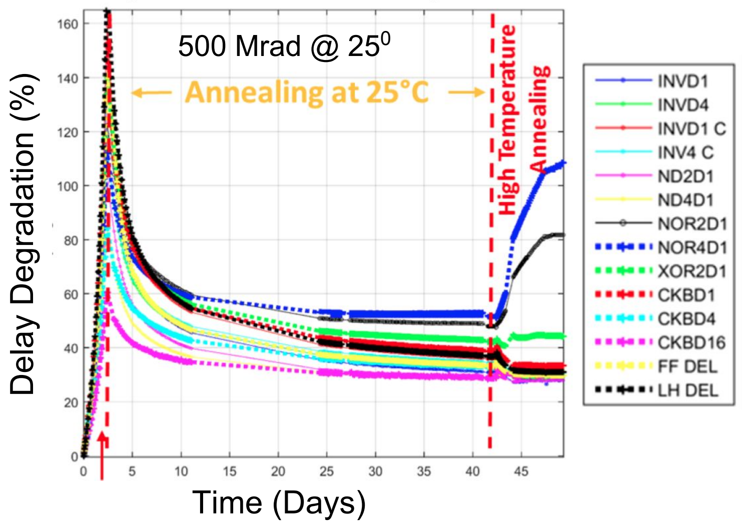


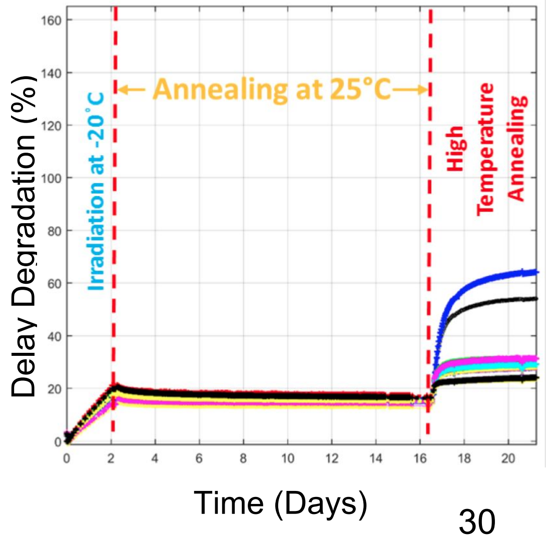




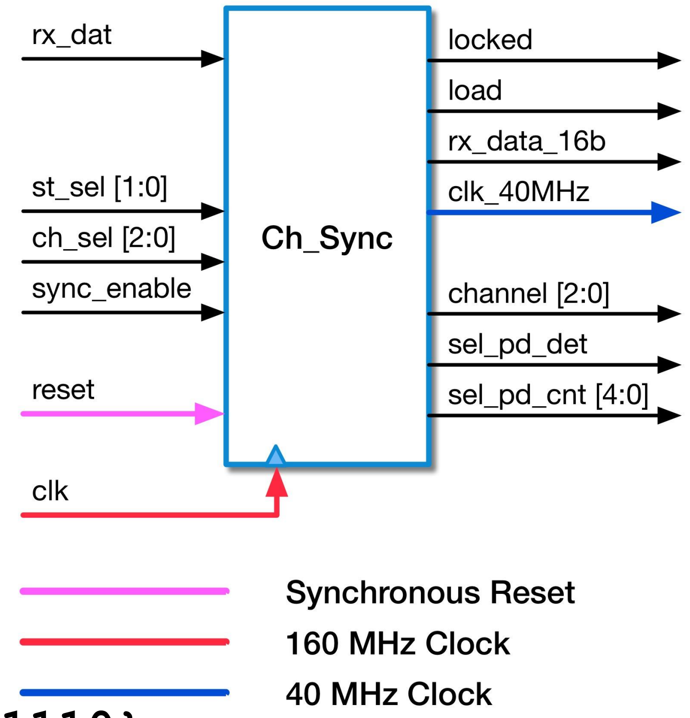








Channel Synchronizer



- All Commands are sent in a single line that encodes Clock + Data.
- Downlink clk 160 MHz, machine clk 40 MHz.
- All commands are in sync with machine clk.
- Need to detect correct phase of clock.
- Data frames are aligned using an unique pattern in Data called Sync `{1000_0001_0111_1110}`.
- Sync symbols are counted in all possible frame alignment channels and as soon as channel N reaches a user programmable threshold (Ch_Thr) N becomes the correct channel.
- The locked signal is set, all counters are reset and subsequent frames are correctly aligned with the phase of the machine clock.
- Counting continues on all channels and as soon as a different channel counter reaches the programmable unlock threshold (Ch_Unlock) sync is considered lost.
- Sync Symbols have to be sent at least every 32 Data symbols to ensure locking.

Trigger command

- Trigger commands have highest priority, so they can always interrupt any other command sequence, without affecting it.

- A Trigger command is defined as:
  - {TriggerNN,DataXX} where DataXX implements the Tag Trigger mechanism.

- Up to 4 consecutive triggers can be sent in a Trigger command. This is needed as a command frame takes 4 LHC bunch crossings to be sent to the chip.

- In order to identify in which bunch crossing a Trigger happened, we have defined 15 different commands (Trigger01 [000T], …, Trigger15 [TTTT]) that cover all possibilities.

- The Tag Trigger, the Tag (a 5 bit number written to “identify” an event) is used internally by the chip to associate events to the user defined Tag.

- This allows, in principle, to send independent data packets that are self identified by the Tag associated to the Data inside each data packet, without having to ensure event ordering inside the chip (this was not implemented in RD53A).

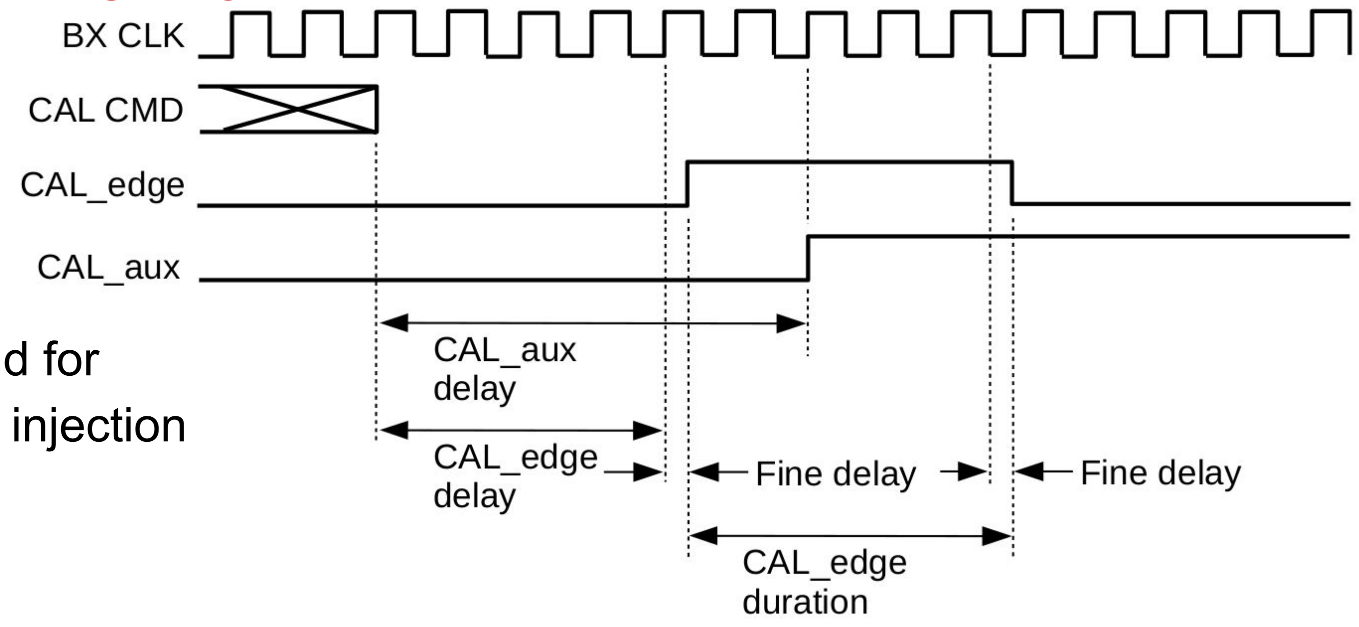
- Up to 16 Triggers are stored inside a Fifo of the chip and a Fifo word is erased once a full event belonging to that Trigger has been transmitted.

ECR, BCR, GlobalPulse commands

- Bunch Counter Reset is a dataless command that resets the BCId counter inside the chip. All other datapath and configuration related structures are unaffected by this command. (Used to synchronize BCId with DAQ)
- Event Counter Reset is a dataless command that clears all data path structures (Memory, Fifo pointers, Counters, Pending Triggers, etc) inside the chip leaving the configuration of the chip unaffected.

- GlobalPulse: {GlobalPulse,GlobalPulse}{ChipId[3:0],0,PulseLength[4:0]}
  It is used to generate, only for the addressed chips, a variable length pulse (ranging from 1 to 32) that can be used for triggering particular behaviours inside the chip (automatic test procedures, different levels of reset, etc)
- Each chip is uniquely identified via three wirebond pads (geographical address)
- The ChipId, associated to commands with data, allows to send data only to the chip that match their geographical address. (ChipId[3] stands for Broadcast)
- All chips that do not match by ChipId simply discard the received command.

CAL command



- One CAL command for Analog and Digital injection

{Cal,Cal}{ChipId[3:0],0,E_Mode,EdgeDly[3:0]}{EdgePulse[4:0],A_Mode,AuxDly[3:0]}

- Allows to inject Digital Pulse, Analog Single Pulse, Analog Double Pulse.
- Cal_aux signal is routed to the CDR/PLL block in order to provide a user programmable fine delay.
- E_Mode selects between step and pulse mode
- A_Mode selects between the level to which CAL_aux is switched to, after applying the Delay (always a step and never a pulse)
- All delays are counted in Machine Clock Cycles (40MHz) clock.

RdRegister command

{RdReg,RdReg}{ChipId[3:0],0,Addr[8:4]}{Addr[3:0],0,Null}

- Reading an address will return the 16 bit value of the addressed register.
- Only chips that match ChipId[2:0] will return some data as commands for non addressed chips will be ignored by the Command Decoder.
- Writing ChipId[3]=1 will enable all chips (Broadcast)
- Register #0 has a special meaning, being mapped to the user programmable row/col of the Pixel Matrix.
- Auto increment will allow to read the full matrix configuration just accessing this register location.
- As not all possible 512 registers are implemented, if you try to readback one inexistent register you will be returned with an all zero data word.

WrRegister command

{WdReg,WdReg}{ChipId[3:0],0,Addr[8:4]}{Addr[3:0],Data[15:10]}{Data[9:0]}

- A WrRegister command will write a 16 bit data in the addressed register if the chip is matched by ChipId[2:0] or if ChipId[3]=1.

- Register #0 has a special meaning, being mapped to the user programmable row/col of the Pixel Matrix.
- Addressing this register one can write one (or all) double column row.

- In addition to the “single mode” WrRegister there is a “burst mode” variant of this command that is accessed appending nine Data only Frames after a regular WrRegister command. In this case all Data following the first will be interpreted as new Data words to be sent to the selected pixel column pair.
- As not all addressable registers are implemented nothing happens if one tries to write a value in a non existent Register (no bound check implemented).

Warning / Error handling

- A Warning message is generated if:

  1. A BitFlip detection and correction in a Sync or Command frame.

- There are two different types of Error messages:

  1. A BitFlip in a Symbol.
  2. A not recognised Symbol (like if one is expecting Data and receives a Command instead, or a completely corrupted Symbol)

- There is no way to correct for single BitFlips while sending Trigger or Data frames

Other Warning / Errors can be generated and added to the output stream.

Warning / Error message types can be individually turned off via configuration.