AGH University of Science and Technology

SALT – digital part

Krzysztof Świentek

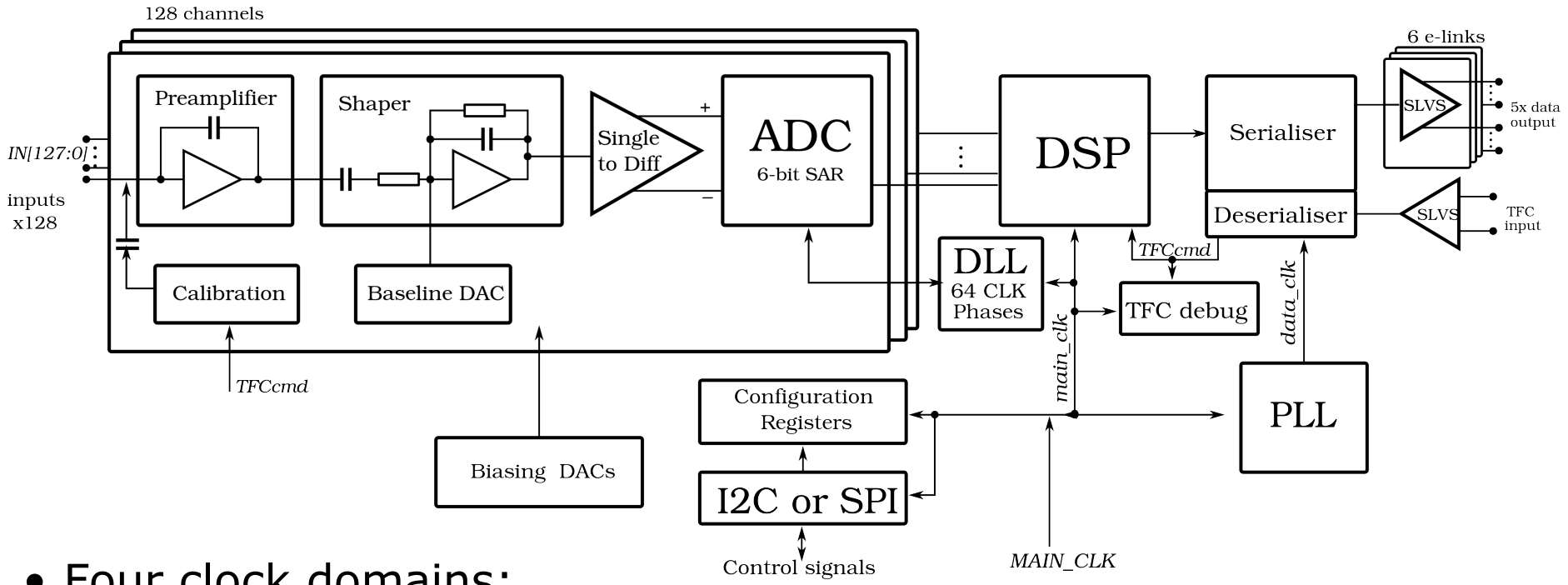
Faculty of Physics and Applied Computer Science
AGH University of Science and Technology

LHCb UT Workshop, Milan
17-19 May 2016

Outline

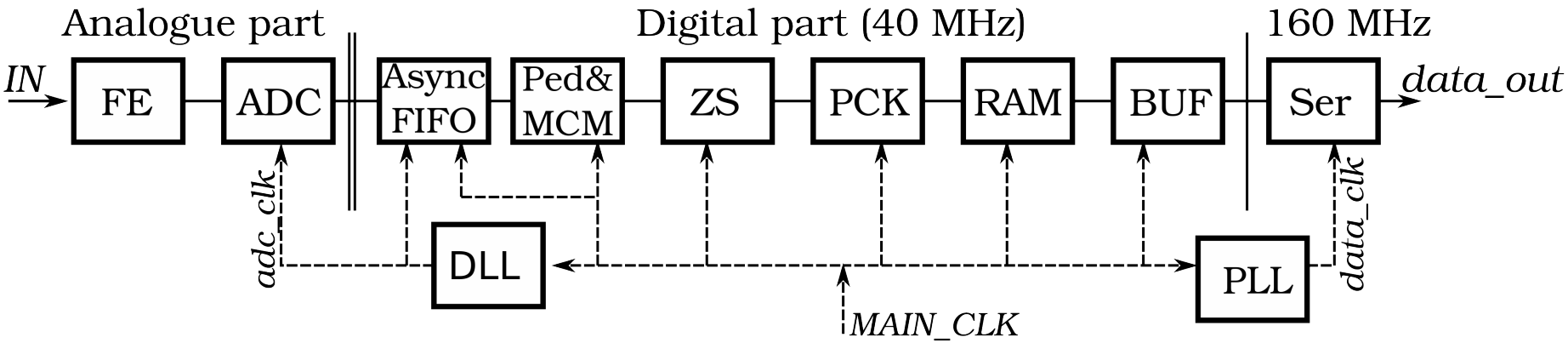
- Introduction
- RTL design:
  - DSP & back-end data processing
  - New features (TMR, reset, test pulse)
  - Verification
- Physical implementation
- SALT issues

Introduction
SALT block diagram



- Four clock domains:
  - *main_clk* (input clock)
  - *adc_clk* (generated by DLL)
  - ***calib_clk*** (generated by DLL)
  - *data_clk* (generated by PLL, 4 times faster)

Introduction
SATL Data processing chain

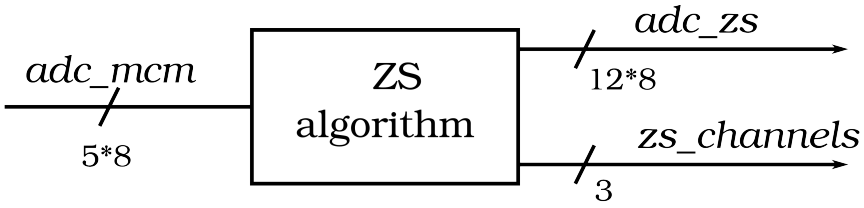


- Analogue processing: FE and ADC
- **Asynchronous FIFO** of depth 3
- DSP: Ped, MCM and ZS
- Back-end data processing: PCK, **RAM and BUF**
- Serializer: Ser

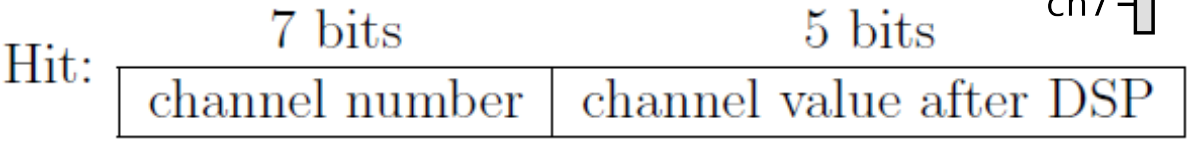
Outline

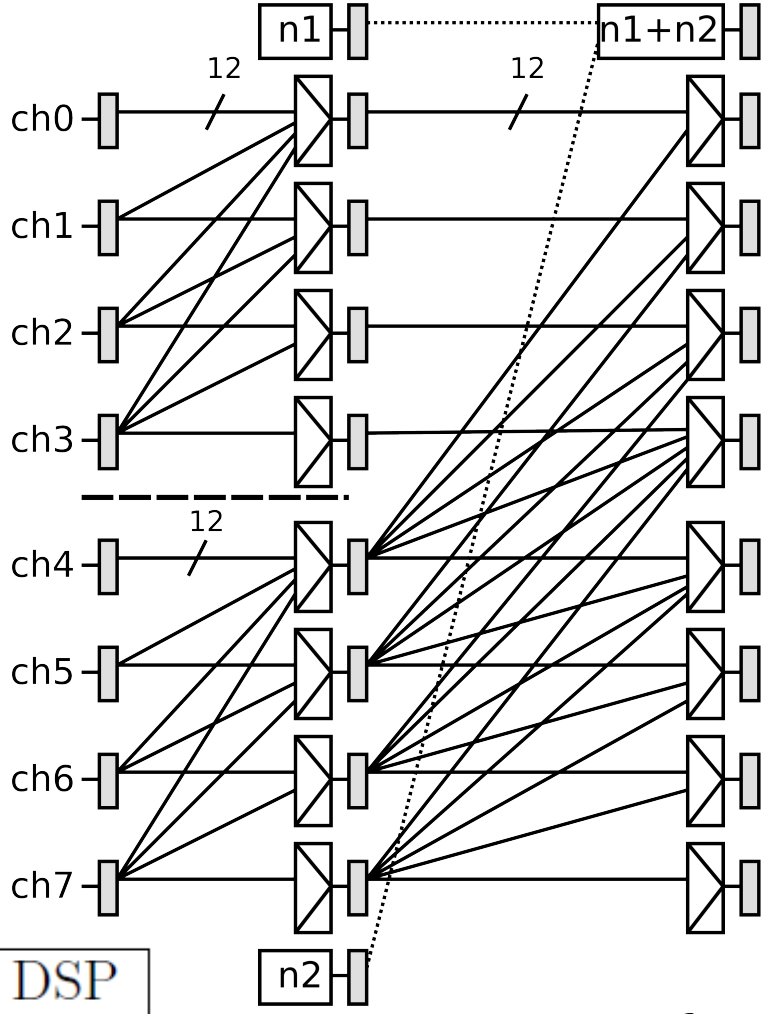
- Introduction
- RTL design:
  - **DSP & back-end data processing**
  - New features (TMR, reset, test pulse)
  - Verification
- Physical implementation
- SALT issues

DSP
Zero suppression block



- Input values 5-bit unsigned
- Constant latency 5
- Data element (hit): 7 bit channel number + 5 bit data = **12 bits**
- Output size is **0 – 63** hits
- The maximum data size is: 12*63 = 756 bits





Back-end data processing
TFC commands

- NZS – send NZS data packet
- BXReset – reset BXID counter
- HeaderOnly, BXVeto – send Header data packet
- Synch – empty buffers and send Sync data packet
- FEReset – empty buffers & resets TFC counters
- **Calib** – send calibration impulse to the analogue front-end input
- **Sanpshot** – write TFC counters to snapshot registers

- **Each TFC command has its own counter and counter snapshot available via I2C interface**

Back-end data processing
Data packet format (current one)

Packet name	Header (8-bit) BXID	NoData	IsTrunc	IsNZS	Parity	Length (8bit) Len	Parity	Data
Header	4-bit	1	0	0	1-bit	—	—	—
Idle	0000	1	1	0	0	—	—	—
Normal	4-bit	0	0	0	1-bit	7-bit	1-bit	Hits
Truncated	4-bit	0	1	0	1-bit	7-bit	1-bit	—
NZS	4-bit	0	0	1	1-bit	1111111	1	Values
Sync	12-bit BXID + 12, 20 or 28-bit sync data							

- Header, Normal, Truncated, and NZS created in packet building block (PCK) and go to RAM
- Idle and Sync are created in BUF block
- Hits: channel number (7b)+ channel value (5b)
- Values: mcm_value, mcm_channels, Chn7, Chn6, …, Chn0
- Packet length is filled up with zeros to full bytes (8bit)

Back-end data processing
Final data packet format

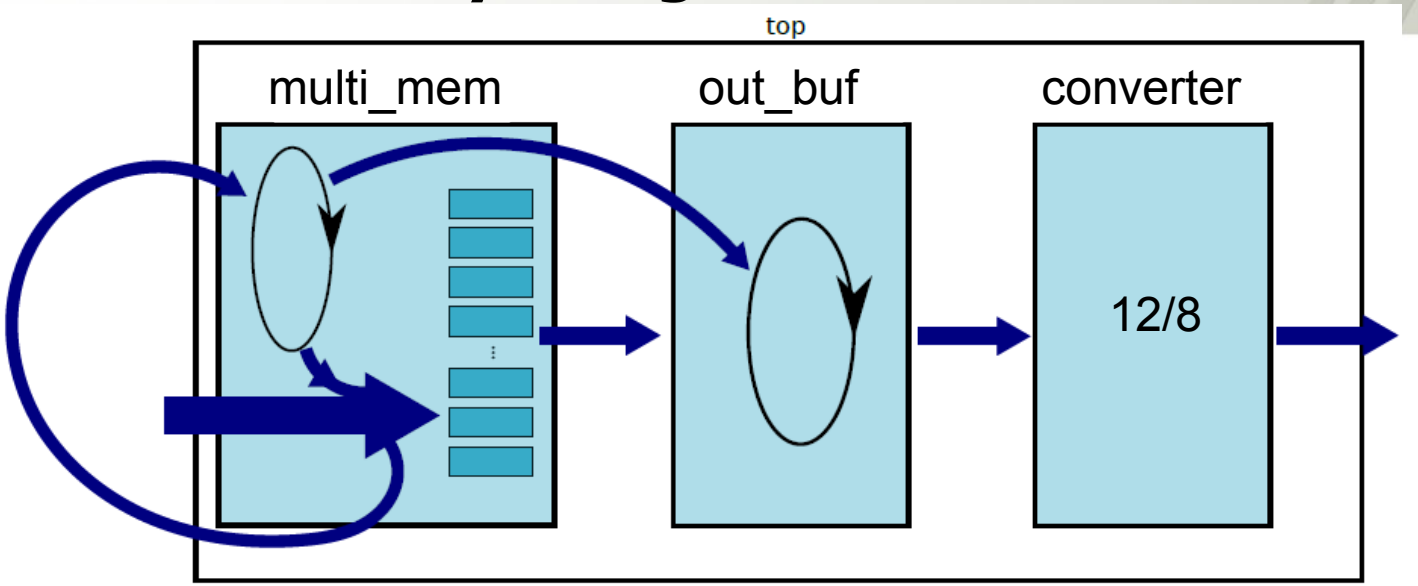
Packet name	Header (12-bit) BXID 4 bits	Parity 1 bit	Flag 1 bit	Length 6 bit	Data n·12 bits	Comment
Idle	0000	1	1	'b11_0000	—	no enough data
BxVeto	BXID[3:0]	*	1	'b01_0001	—	BxVeto in TFCcmd
HeaderOnly	BXID[3:0]	*	1	'b01_0010	—	HeaderOnly in TFCcmd
BusyEvent	BXID[3:0]	*	1	'b01_0011	—	$nHits > 63$
BufferFull	BXID[3:0]	*	1	'b01_0100	—	no space in memory
BufferFullN	BXID[3:0]	*	1	'b01_0101	—	no space in memory
NZS	BXID[3:0]	*	1	'b00_0110	Values	NZS in TFCcmd
Normal	BXID[3:0]	*	0	$nHits$	Hits	Normal event
Sync	BXID[11:0]				pattern	fill one whole frame

- Packet base element is 12 bit – complicated serialisation because 8-bits e-links
- More packet types – better ASIC control
- Sync fills whole frame (may be **not** n*12 bits)

Back-end data processing
Memory challenge

- Packet size range in SALT is extremely wide:
  - 1 word = 12 bits (e.g. HeaderOnly)
  - 2–64 words (ZS)
  - about 67 words (NZS)
- Memory size based on 12 bit word
- Output data stream changes from ASIC to ASIC: from 3 – 6 bytes (active e-ports) in single clock cycle
- Output data size 24, 32, 40 or 48 bits do not fit to memory word in general

Back-end data processing
Memory design



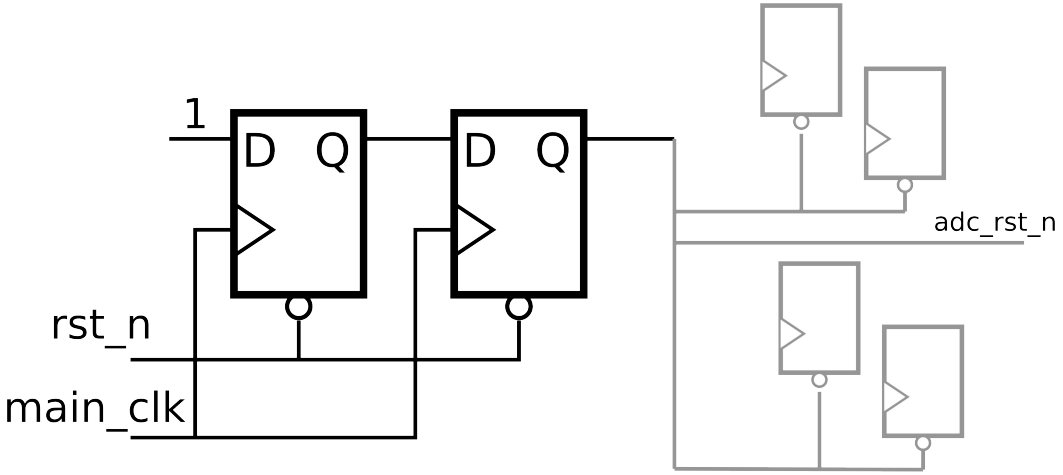
Parametrized modules, so right now base data size is 8.

- Many RAM instances – the smaller element the smaller block and power consumption
- Input circular buffer size: RAM instance size -1
- Data goes directly to RAM except small part which stays in buffer
- RAM width = 4 or 8 elements (power&area)
- Output circular buffer because variable number of e-links (Idle creation)
- Converter generates also Sync packets

Outline

- Introduction
- RTL design:
  - DSP & back-end data processing
  - **New features (TMR, reset, test pulse)**
  - Verification
- Physical implementation
- SALT issues

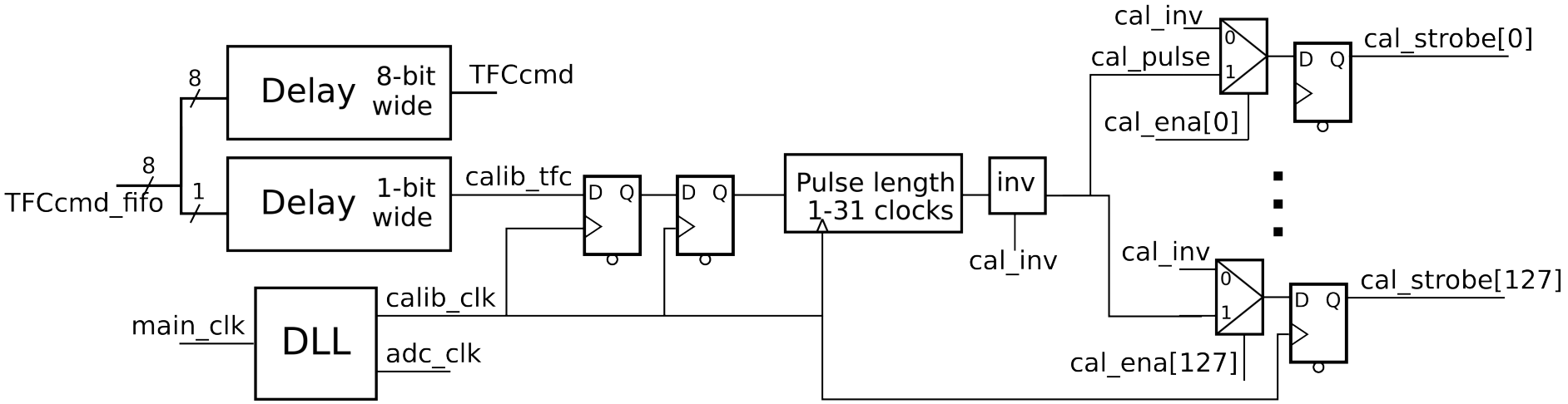
New features
Reset with synchronous removal



- Asynchronous reset
- Synchronous reset removal (clock needed)
- Two DFFs creates a synchronizer, so the relation between *main_clk* and *rst_n* may be undefined

- All DFFs in the ASIC reset asynchronously by signal from synchronizer

New features
Test pulse generator



- Independent delay then rest of TFC, but only for test pulse generation
- Clock phase controlled by DLL
- Variable pulse length and polarisation
- Masking different then in DSP

New features
Triplications

- All triplicated registers and counters are self corrected – SEU counter in each WishBone block treated as ordinary register, but corrected by +1
- Triplicated elements
  - configuration register
  - BXID counter
  - TFC counters
  - Snapshot registers
- Although referees suggested to tripicate clock, reset (both partially) and pointers in memory, FSM state registers, etc. it will not be done in this release

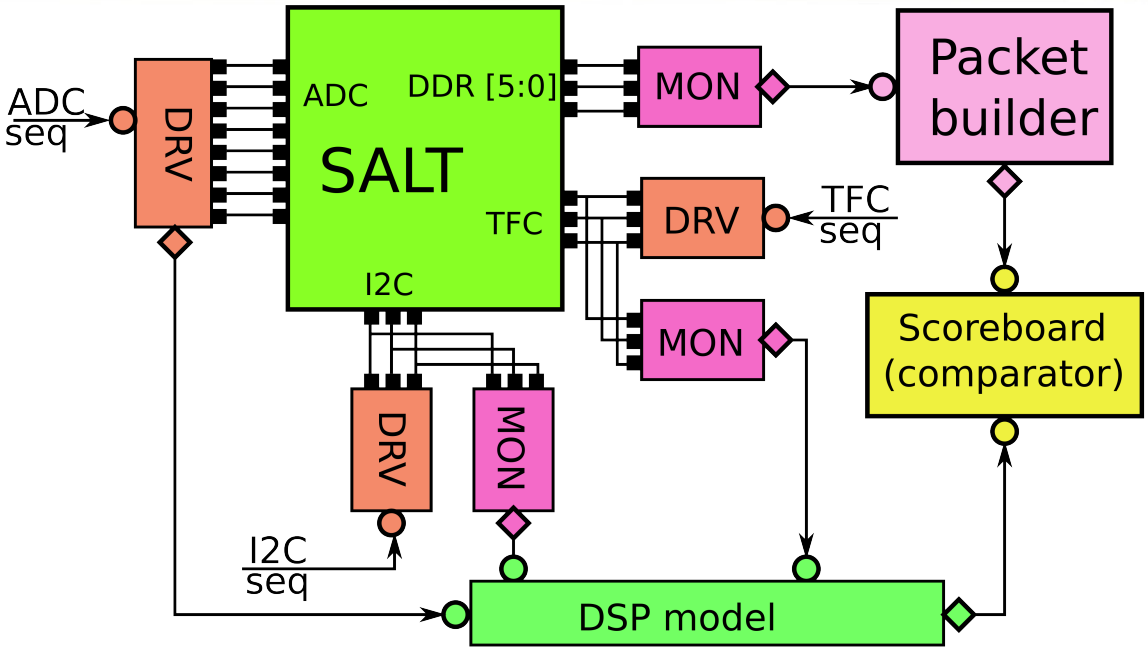
Outline

- Introduction
- RTL design:
  - DSP & back-end data processing
  - New features (TMR, reset, test pulse)
  - **Verification**
- Physical implementation
- SALT issues

Verification Introduction

- Extensive checks of full design functionality
- High level of abstraction – object oriented language SystemVerilog (similar to C++)
- UVM (Universal Verification Methodology) – verification library
- Best verification is automatic
  - Self-checking of outputs
  - random input
- In fact, never complete...

Verification DSP



- TFC and ADC data are randomised with constraints
- DSP model creates exactly the same packets as ASIC
- Packet builder (ready for 8-bit packets)
  - catches sync packet (at least 3) to synchronize data
  - assembles data packet

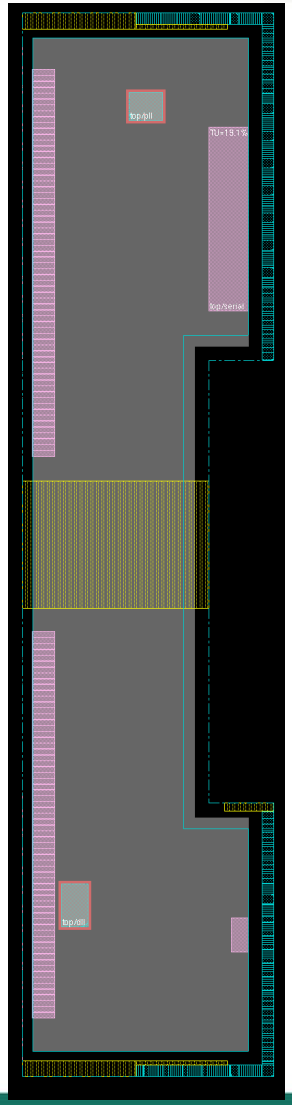
Outline

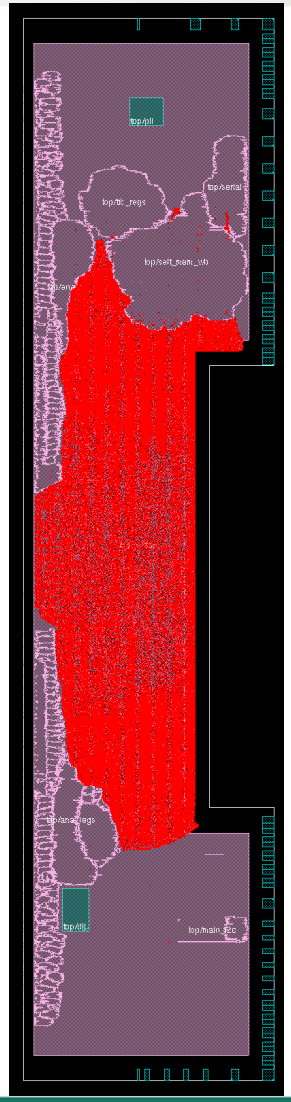
- Introduction
- RTL design:
  - DSP & back-end data processing
  - New features (TMR, reset, test pulse)
  - Verification
- **Physical implementation**
- SALT issues

Physical implementation
Digital implementation flow

- Custom blocks characterisation (SLVS driver & receiver, DDR SLVS driver, PLL, DLL)
- Synthesis
  - four clock domains (multiplication at PLL output)
  - scan chain added (for wafer tests)
- Place & route
  - placement constraints (FIFO, test pulse output)
  - power pads added (separate supply and ring supply)
- Analog-on-Top methodology for ASIC closing
  - add pad openings
  - create schematic from Verilog netlist
  - DRC & LVS

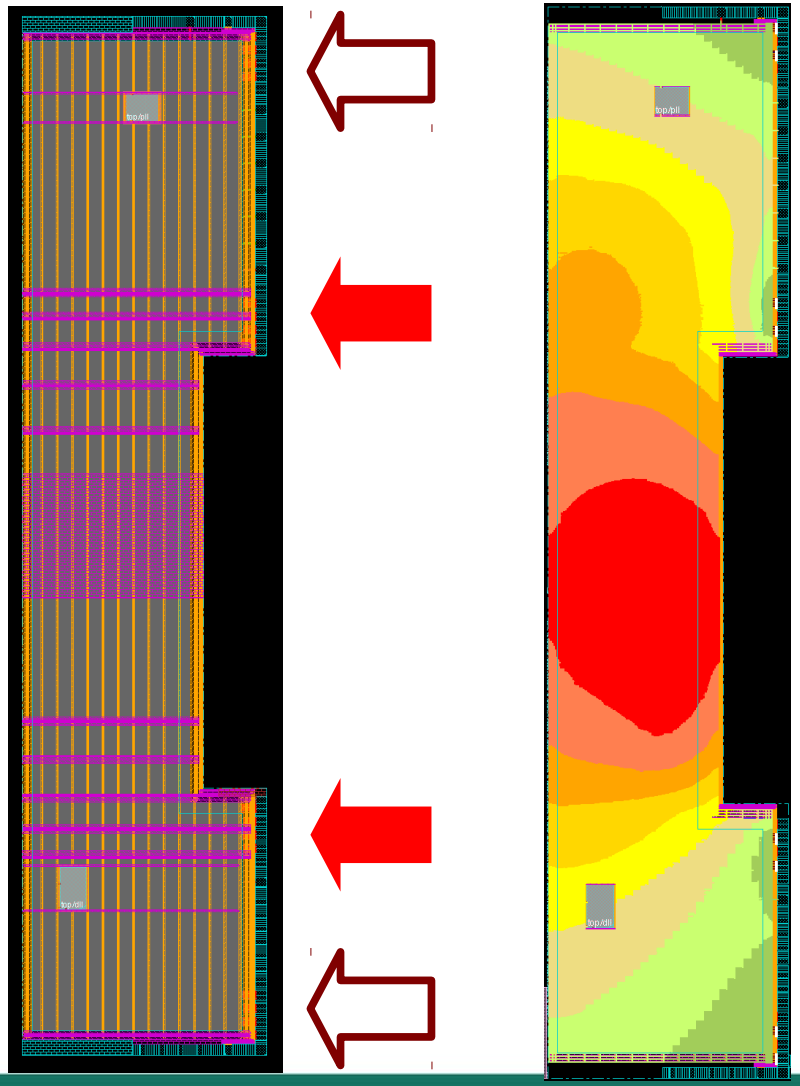
Physical implementation
Floorplan





- PLL and DLL are preplaced
- Guided placement of elements:
  - input FIFO
  - I2C master and serializer
- ZS is the biggest element in layout located in the middle

Physical implementation
Power distribution



- 4 supply points, 2 of them delivers most of current
- The largest voltage drop (middle) is about 30 mV
- Slow corner library use 1.08 supply, so 120 mV less then nominal (and 125ºC)
- Fast corner library is for 1.32 V (and 0ºC)

Physical implementation
Power consumption (static analysis)

- Methodology
  - only DSP is active
  - input switching activity form simulation e.g.
    - 20 Mps adc_data[]
    - 80 Mps clk
    - 218 Mps TFC
  - probability propagation
- Post synthesis/PNR simulations needed for higher precision

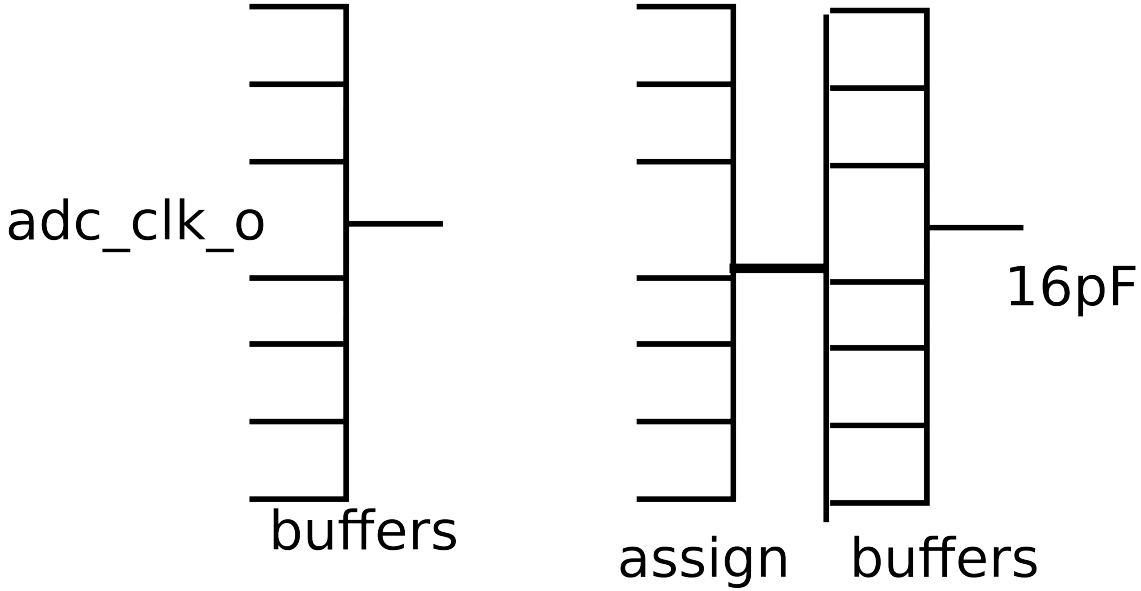
Module	Power [mW]	%
salt_dig	405	100
top	312	77
DSP	216	53
ZS	138	34
MCMS	27	6.6
mem	23	5.7
memm	21	5.2
ana_regs	20	4.9
tfc_regs	7.1	1.8
serial	6.0	1.5

Outline

- Introduction
- RTL design:
  - DSP & back-end data processing
  - New features (TMR, reset, test pulse)
  - Verification
- Physical implementation
- **SALT issues**

SALT issues

- Clock Tree Synthesis issue



- Memory issue – after some time (for ZS packets only) output packet is different then input one

SALT status I

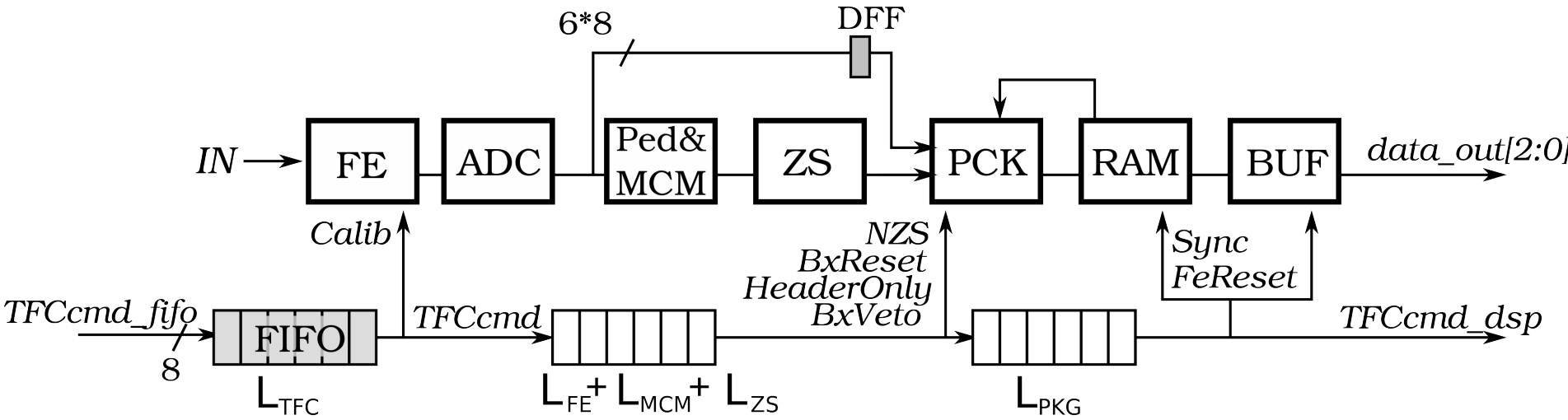
- DSP with new memory subsystem and variable number of e-links is finished and verified
- Memory issue *to be done*
- New packet format 12-bit base *not done*, but
  - memory is ready for 12-bit packets
  - verification is based on packets recognition
- SEU robustness
  - triplication of configuration registers and counters (TFC, BXID) done
  - I2C master corrected, so watchdog is active
- Test pulse generator (digital part) done

SALT status II

- Synthesis done
- PNR is very advanced – almost finished
- HVT and NORMAL libraries used to be more SEU robust
- Thick oxide decoupling cells used – to avoid leakage
- CTS issue have ***to be solved***
- LVS and DRC to be done
- Post-synthesis and post-PNR simulations ***to be done***

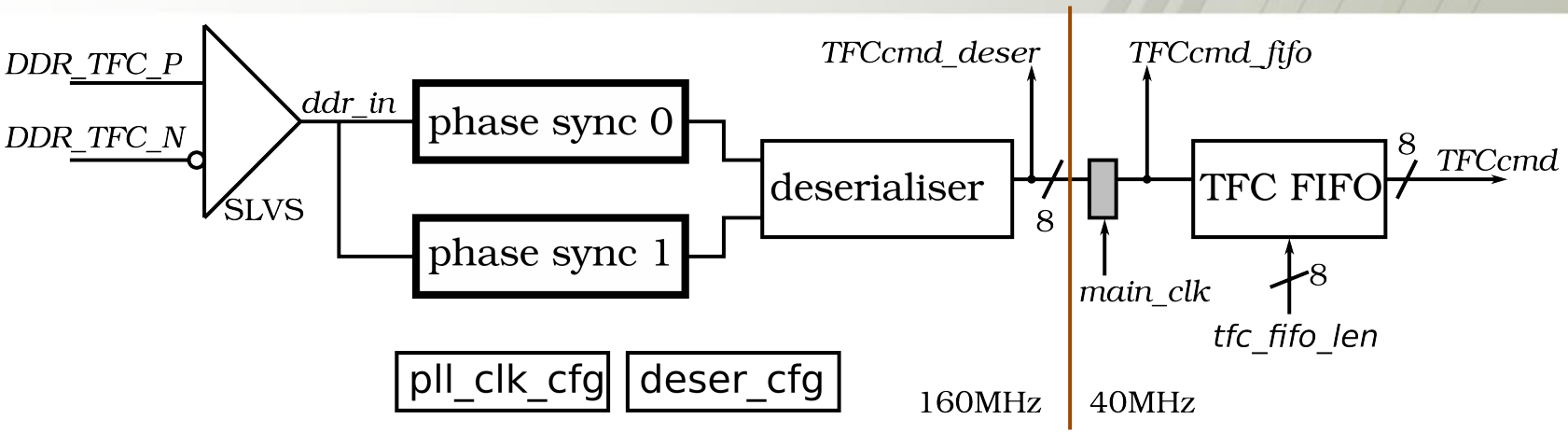
BACKUP SLIDES

Back-end data processing
TFC synchronisation



- TFC is received every clock cycle and it should be keep synchronous with specific BX and with DSP pipeline
- Input TFC FIFO located in serializer block (variable latency)
- **TFC counters** located just after TFC FIFO
- NZS TFC command changes data flow

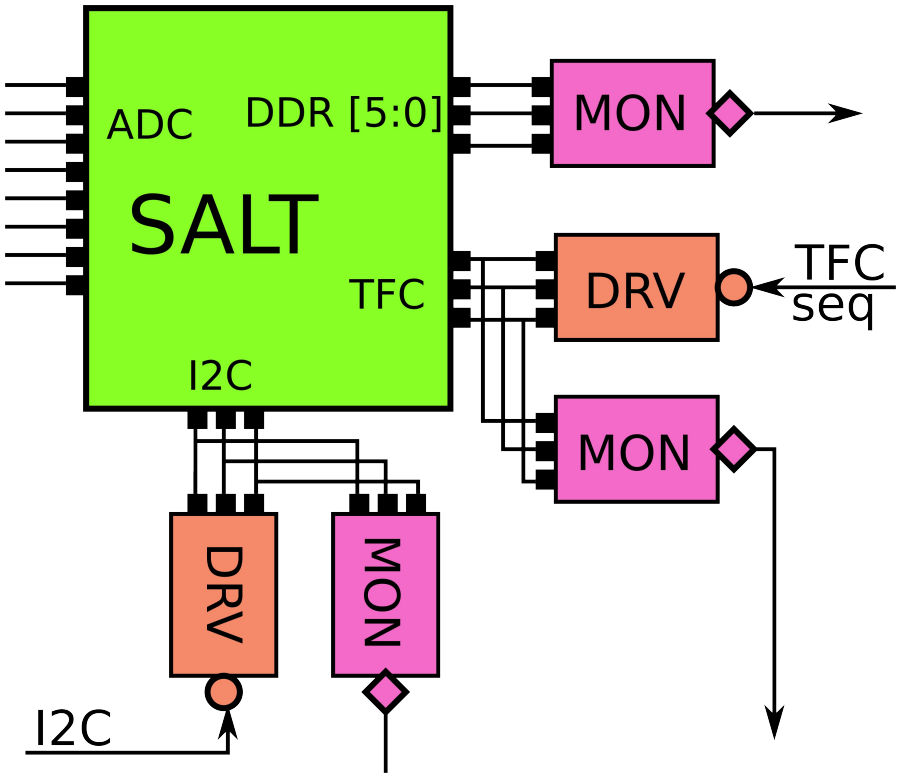
Data/TFC interface
Deserializer



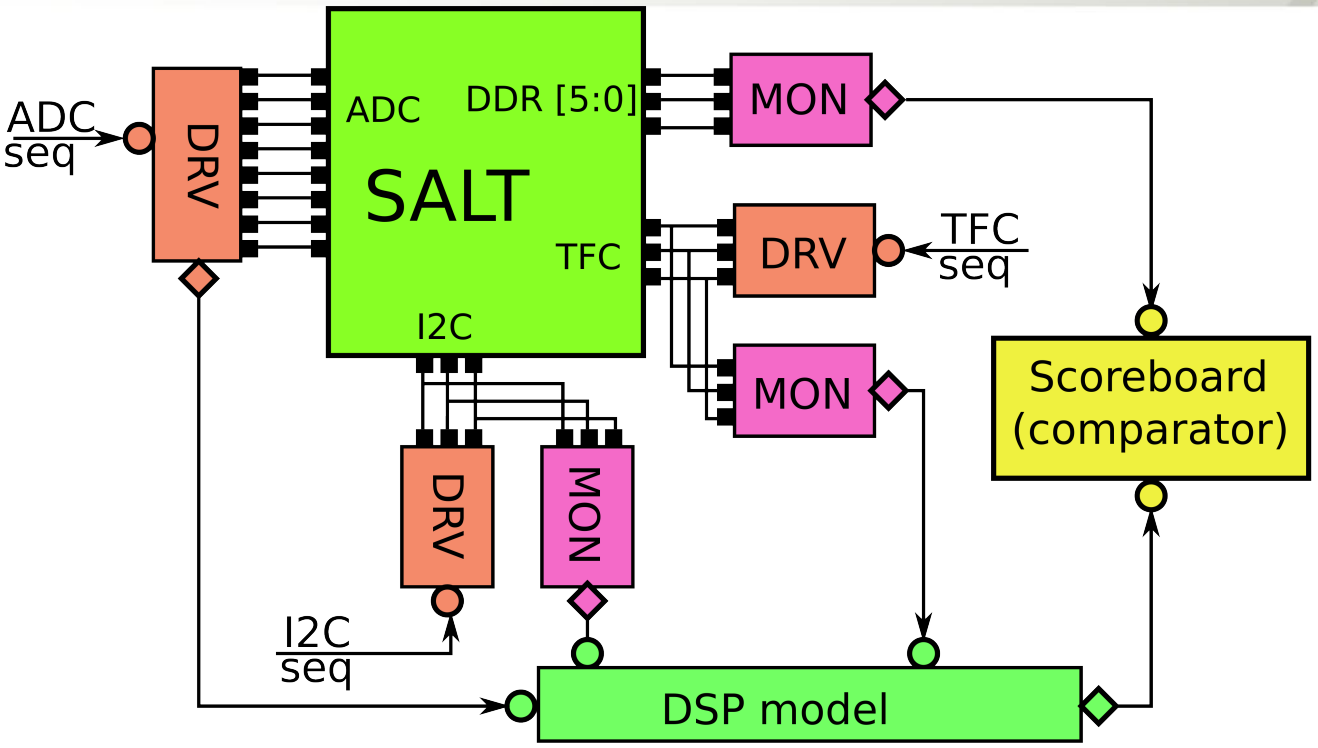
- Two phase sync (shift) blocks for two clock edges – flexible solution
- 1st bit position in a byte may be chosen in deser_cfg register
- Three step connection procedure:
  1. Set delay in FPGA/GBT receiver using serializer pattern reg.
  2. Phase set for both edges – receiving constant pattern and sending it back through serializers
  3. Choice of 1st bit position

Verification Introduction

- Based on UVM methodology
- Each interface has its own agent
  - driver converts transaction to signals
  - monitor converts signals to transaction
- Stimulus based on sequences of transaction
- Single virtual sequencer to coordinate individual interface sequences



Verification
DSP old scheme

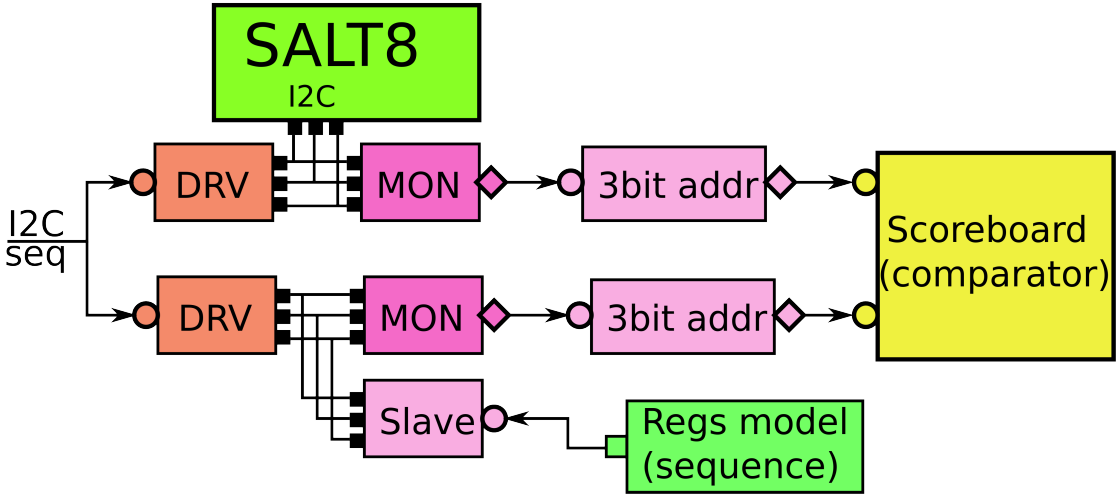


- Serializer (TFC to DDR, pattern, internal counter) is verified analogously

- DSP model creates exactly the same output data as ASIC
- TFC and ADC data are randomised with constraints

Verification I2C

- Protocol converter from I2C to SALT-I2C
- Two independent agents (DRV/MON/Slave)
- Register modelled in a slave sequence
- Simple scoreboard – comparator only
- Full I2C test:
  - multi-byte read&write
  - address ASIC change
  - auto incrementation
  - write+read



- Component not shown – constant ASIC input **address** driver