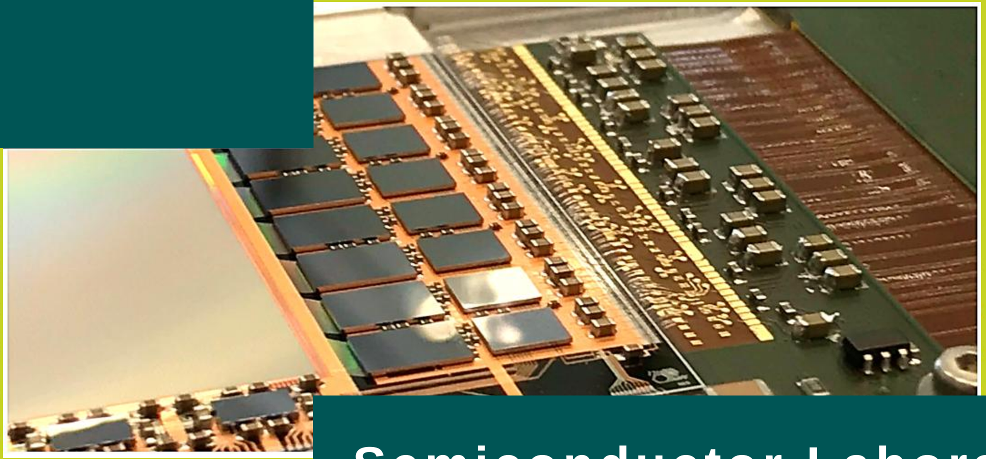
EDET INSTRUMENT FIRMWARE - STATUS AND PLANS



Semiconductor Laboratory of the Max Planck Society

Mishal Rizwan

AGENDA

Introduction

DMC overview and Operation Modes

Old and New Sequencer IP

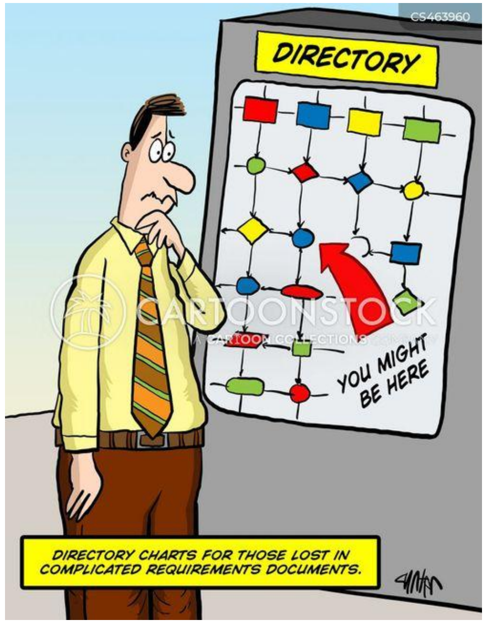
Data capturing

Summary and Outlook

INTRODUCTION

What we want:

- Implement various Operation Modes (OpModes) envisioned for the EDET Instrument (previous talk by J. Treis)
- Use the different features of the DMC-65 chip to accomplish this
- Well-designed sequences required by the DMC to realize the OpModes
- Formatting the data frames received from the DMC
- Specialized Firmware essential!


CS463960
DIRECTORY
YOU MIGHT BE HERE
CARTOONSTOCK
DIRECTORY CHARTS FOR THOSE LOST IN COMPLICATED REQUIREMENTS DOCUMENTS.

What we already have:

- DHPT command sequencer

Can it do what we want?

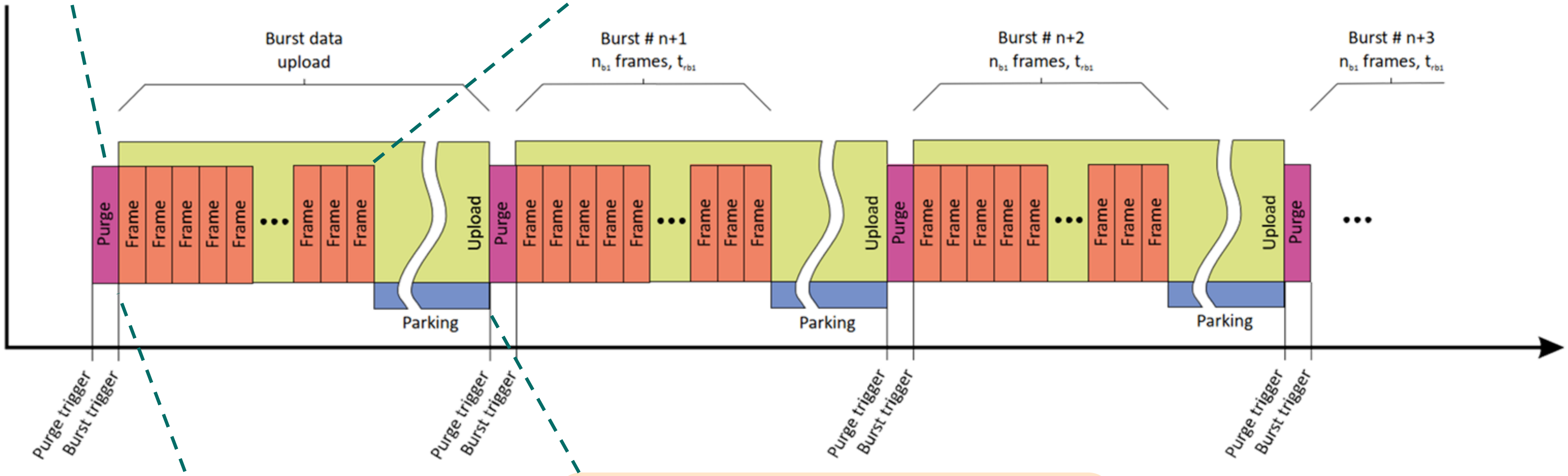
DMC OPERATION MODES

EXAMPLE 1: Standard Purge/Burst

PURGE: Clearing matrix of the charges accumulated while waiting for the next burst trigger. Can be done at high speed to reduce latency. No data recorded.

Sequencer's goal:
- **Response to external Triggers**
- **Implement Purge, Read, Stop modes**
- **Adjustable Parking duration**

Burst READ: Reading Arbitrary number of frames, maximum 50 possible (48 static + 2 dynamic frames with DMC memory capabilities)



Reacting to External Triggers

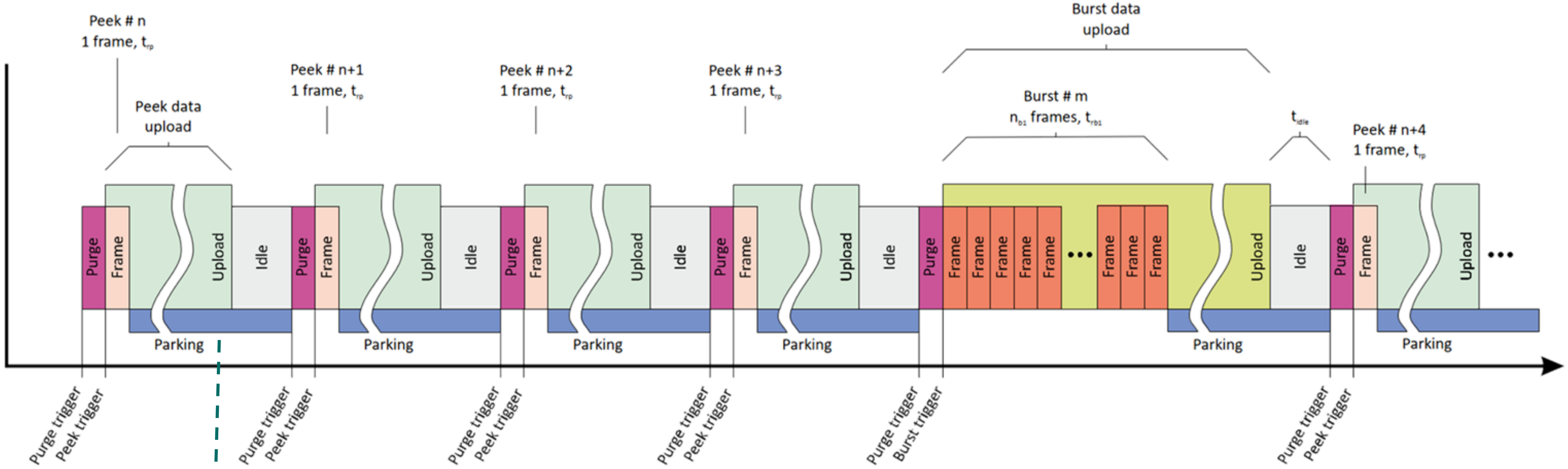
PARKING: Matrix stopped in a defined state while waiting for purge trigger

DMC OPERATION MODES

EXAMPLE 2: Peek/Burst

PEEK frame for observing some process
Purge → Read → Wait → Repeat

Burst READ after observing something scientifically interesting in the Peek frame



Adjustable Parking duration while waiting

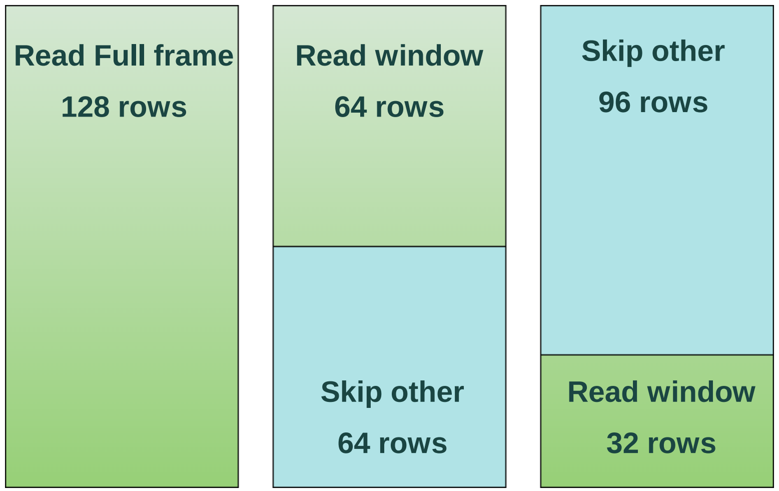
Sequencer's goal:
- **Short Exposure before frame read**
- **Response to external Triggers**
- **Implement Purge, Read, Stop modes**
- **Adjustable Parking duration**

DMC OPERATION MODE

EXAMPLE 3: Window Read

- Instead of reading the full frame, only read Region-of interest (ROI) rows
- Fast forward through the non-ROI rows
- Boost in time resolution
- More frames captured per burst

Window Read Example



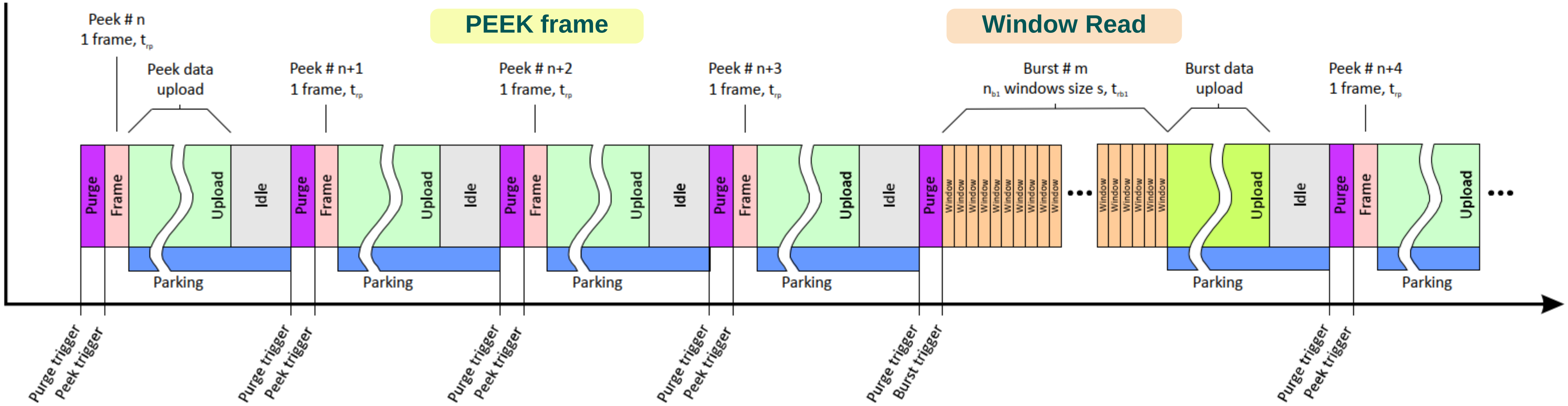
Read Full frame 128 rows

Read window 64 rows

Skip other 64 rows

Skip other 96 rows

Read window 32 rows



DEPFET MOVIE CHIP(DMC 65) – OVERVIEW

COMBINED SEQUENCER / DIGITAL BACKEND / DATA BUFFER IC FOR EDET

Sequencer

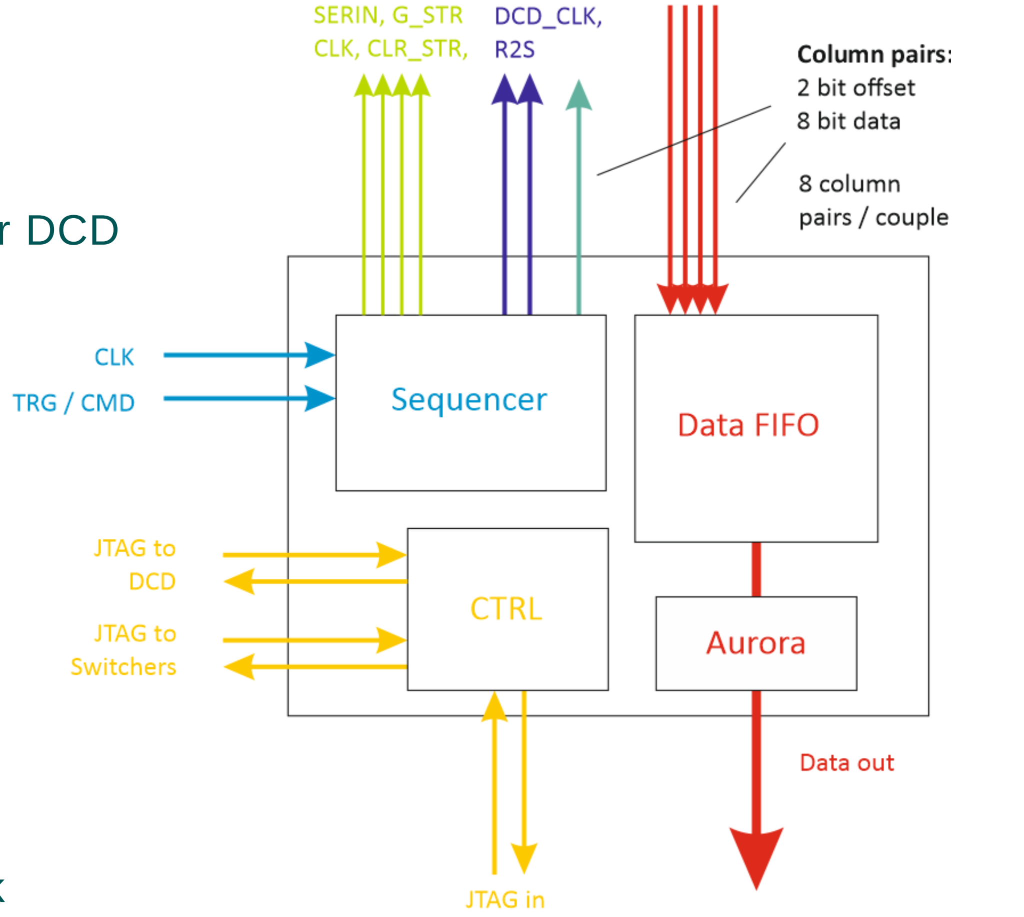
- Memory banks with selectable memory increment
- Store sequence configurations for Switchers and Offset data for DCD
- Sync signal (R2S) to synchronize DCD digitization
- Decoding of TRG/CMD stream

Control block

- JTAG configuration and PLL generation

Data FIFO

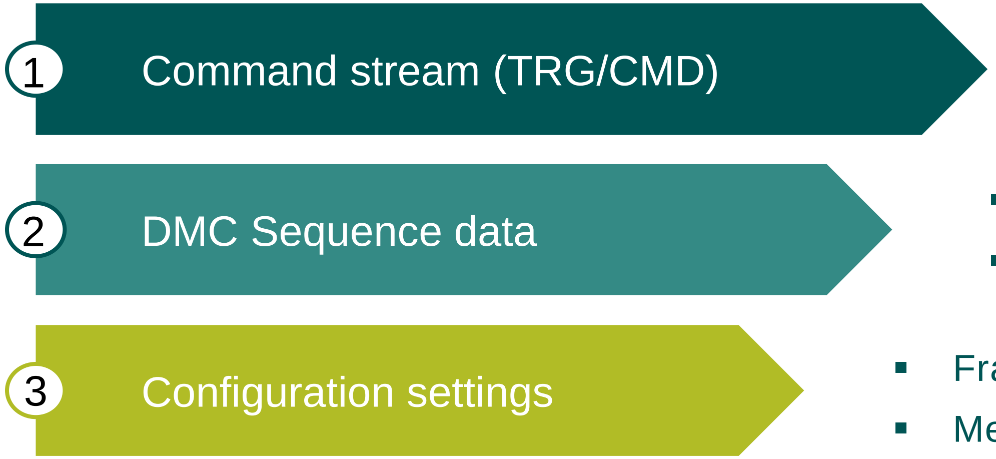
- Data storage in FIFO and Serial Transmission via AURORA link



DMC - Basic block diagram

DMC OPERATION MODE DATA

- Implement some simple and complex OpModes using different features of the DMC 65
- 3 constituents of OpMode configuration data



1 Command stream (TRG/CMD)

- Sequence of instructions from XU1 SoC
- Switches between different modes and memory banks

2 DMC Sequence data

- Patterns for Switcher tracks
- Memory addresses and Offset data

3 Configuration settings

- Frame size
- Memory increment and start addresses

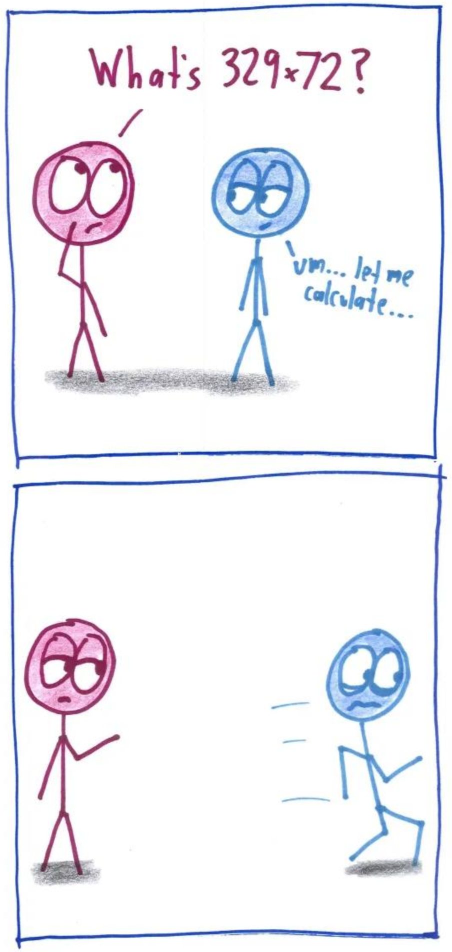
DHPT SEQUENCER - BELLE HERITAGE

What we already have:

DHPT Sequencer

What it can do:

- Basic “Pattern Generator” approach
- Implements sequences as written in its memory
- Only deterministic and repetitive sequences possible
- Memory block is read and translated to TRG/CMD



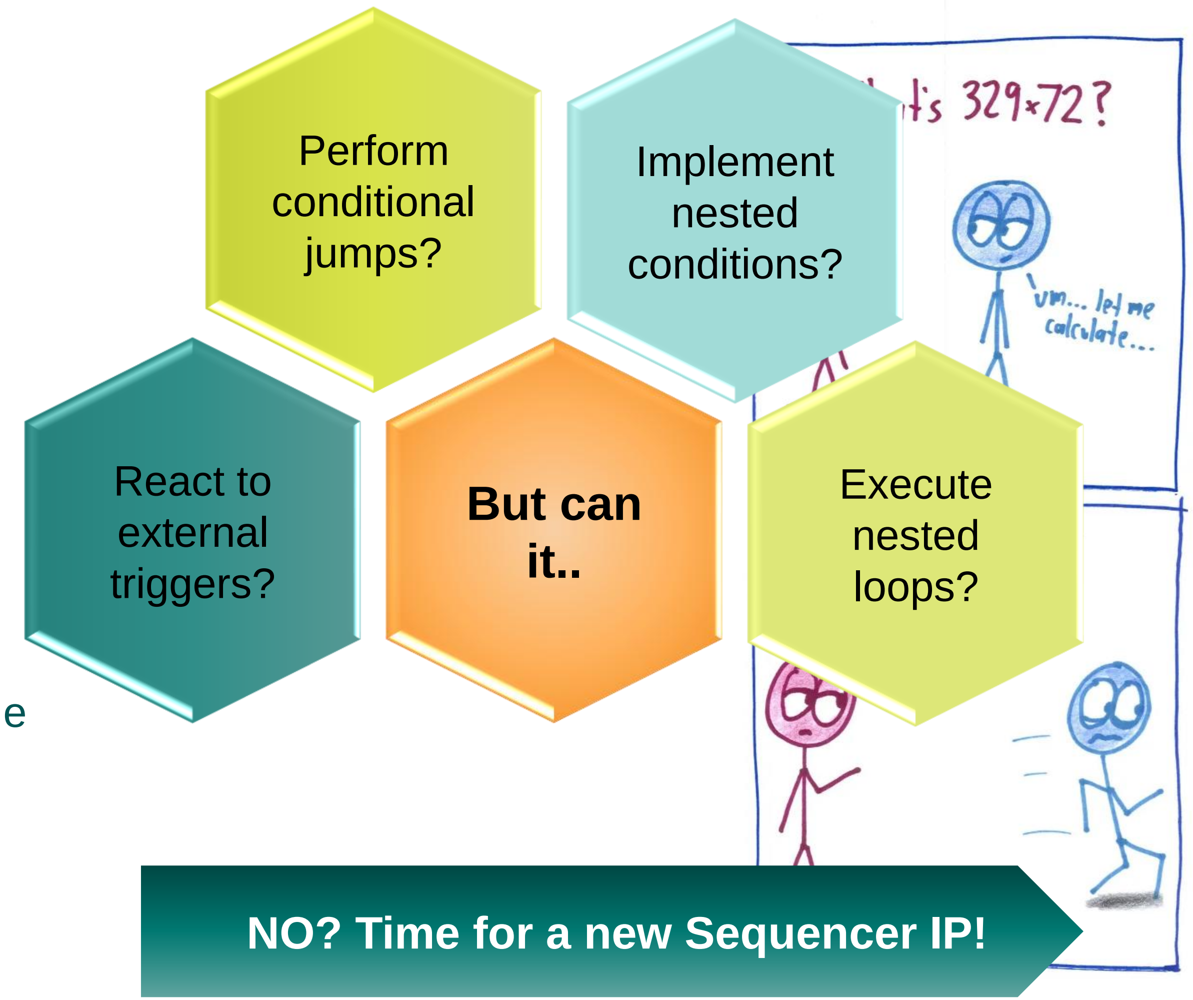
DHPT SEQUENCER - BELLE HERITAGE

What we already have:

DHPT Sequencer

What it can do:

- Basic “Pattern Generator” approach
- Implements sequences as written in its memory
- Only deterministic and repetitive sequences possible
- Memory block is read and translated to TRG/CMD



Perform conditional jumps?

Implement nested conditions?

React to external triggers?

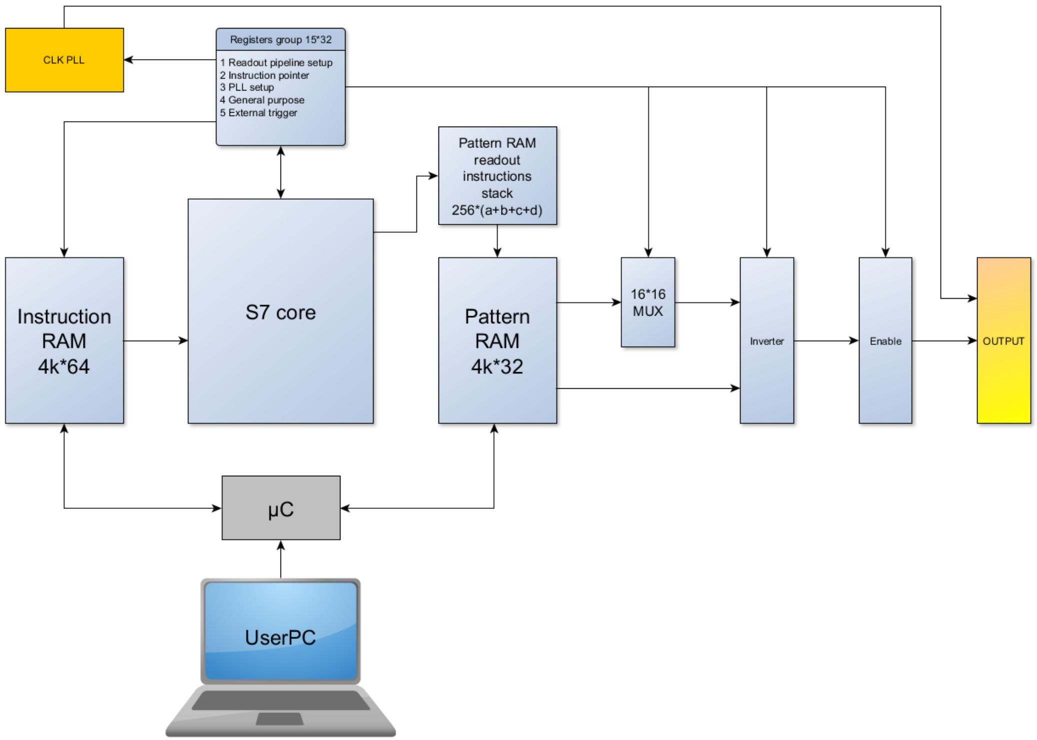
But can it..

Execute nested loops?

NO? Time for a new Sequencer IP!

SEQUENCER IP DEVELOPMENT

S7 SEQUENCER



- Already being used in different projects: FSP, PXD-13..
- Digital scheme with memory blocks, registers, PLL clock
- Outputs data from a Pattern RAM as described by the Instruction RAM

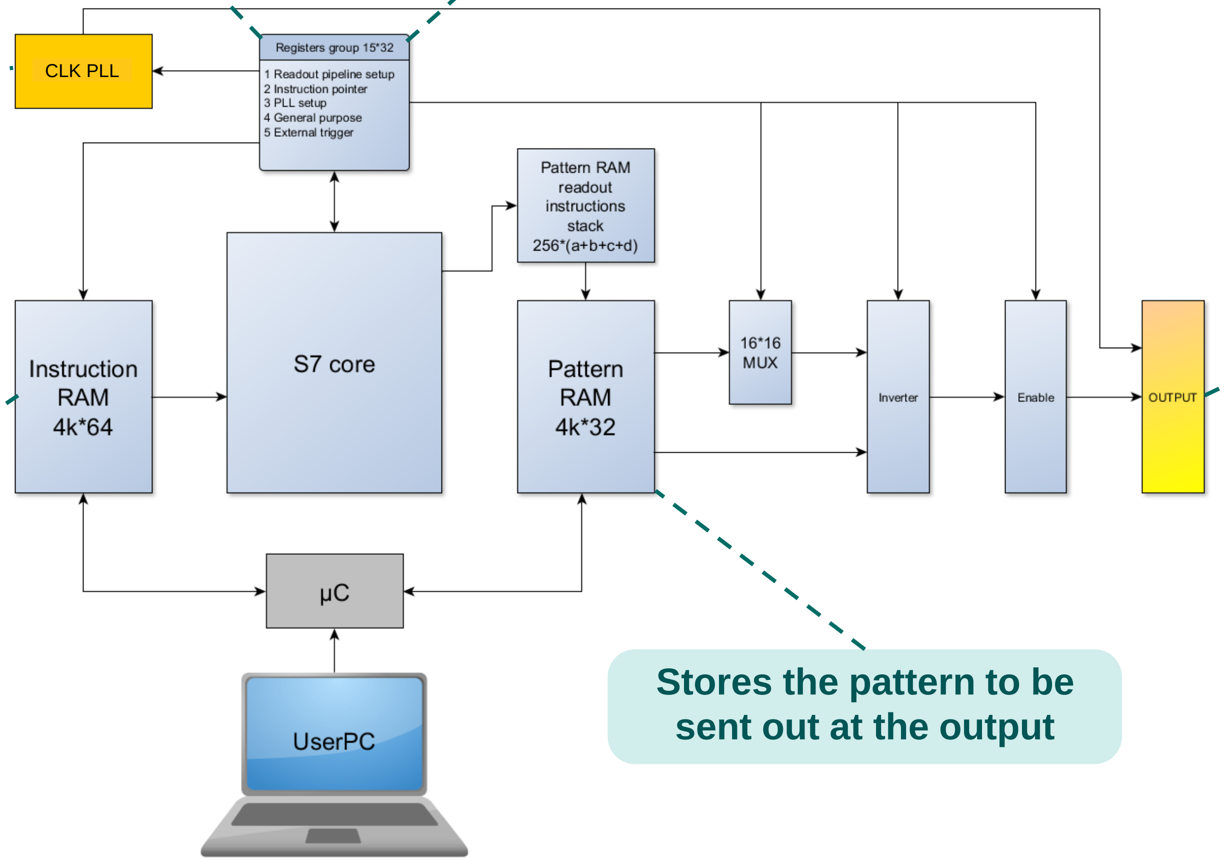
- Various abilities:
  - *Respond to multiple external triggers*
  - *Implementation of loops/conditional jumps*
  - *Customizable clock/ operating frequency*
  - *Increased flexibility*

S7 SEQUENCER

4 inputs for External Triggering

Registers to set the operational state of the Sequencer

Custom PLL clock (10MHz-200MHz)



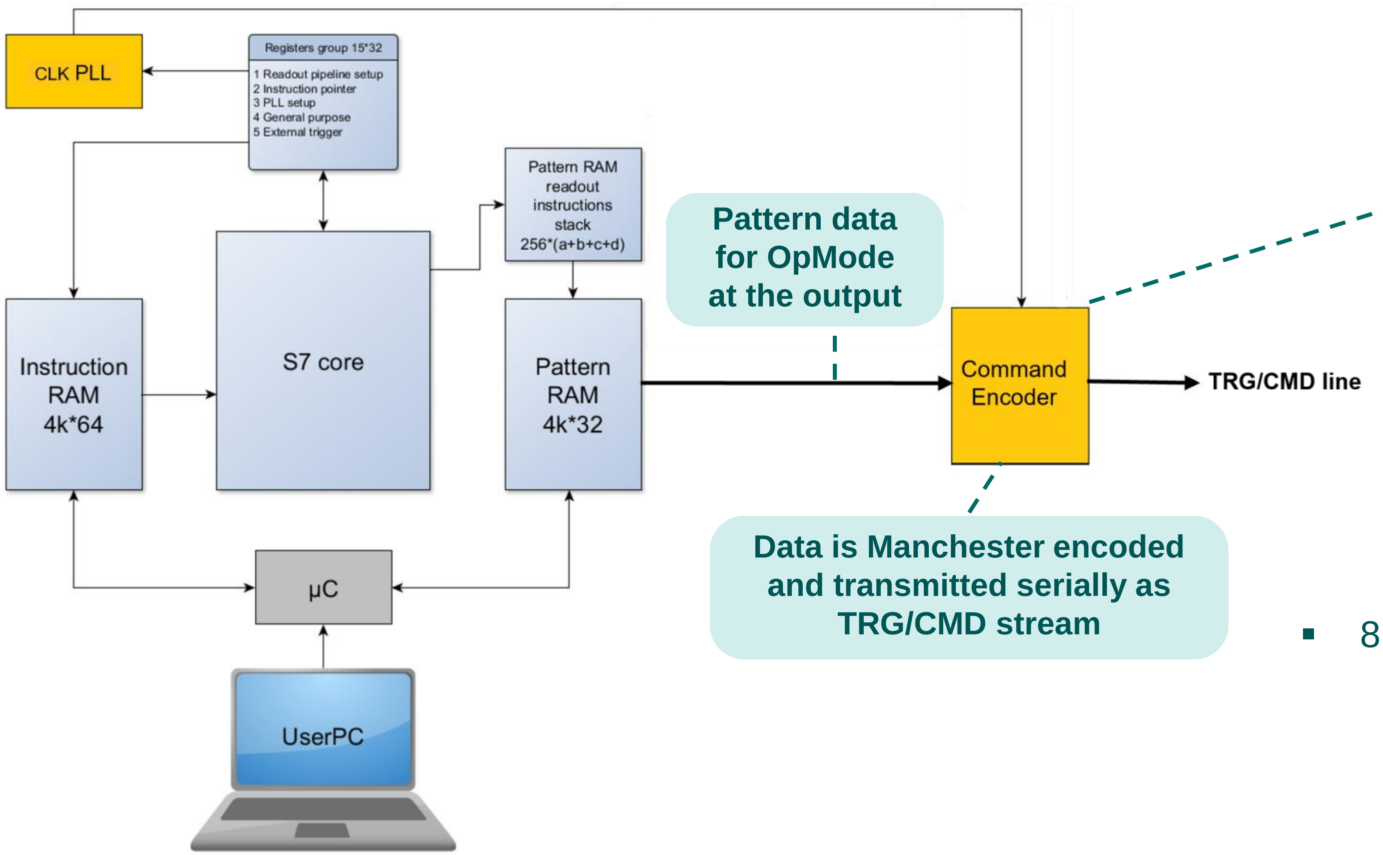
32 Output channels

Stores instructions on how to send out the sequences stored in the pattern RAM at the output

Stores the pattern to be sent out at the output

S7 SEQUENCER- Modifications for EDET

TRG/CMD SEQUENCER



Pattern data for OpMode at the output

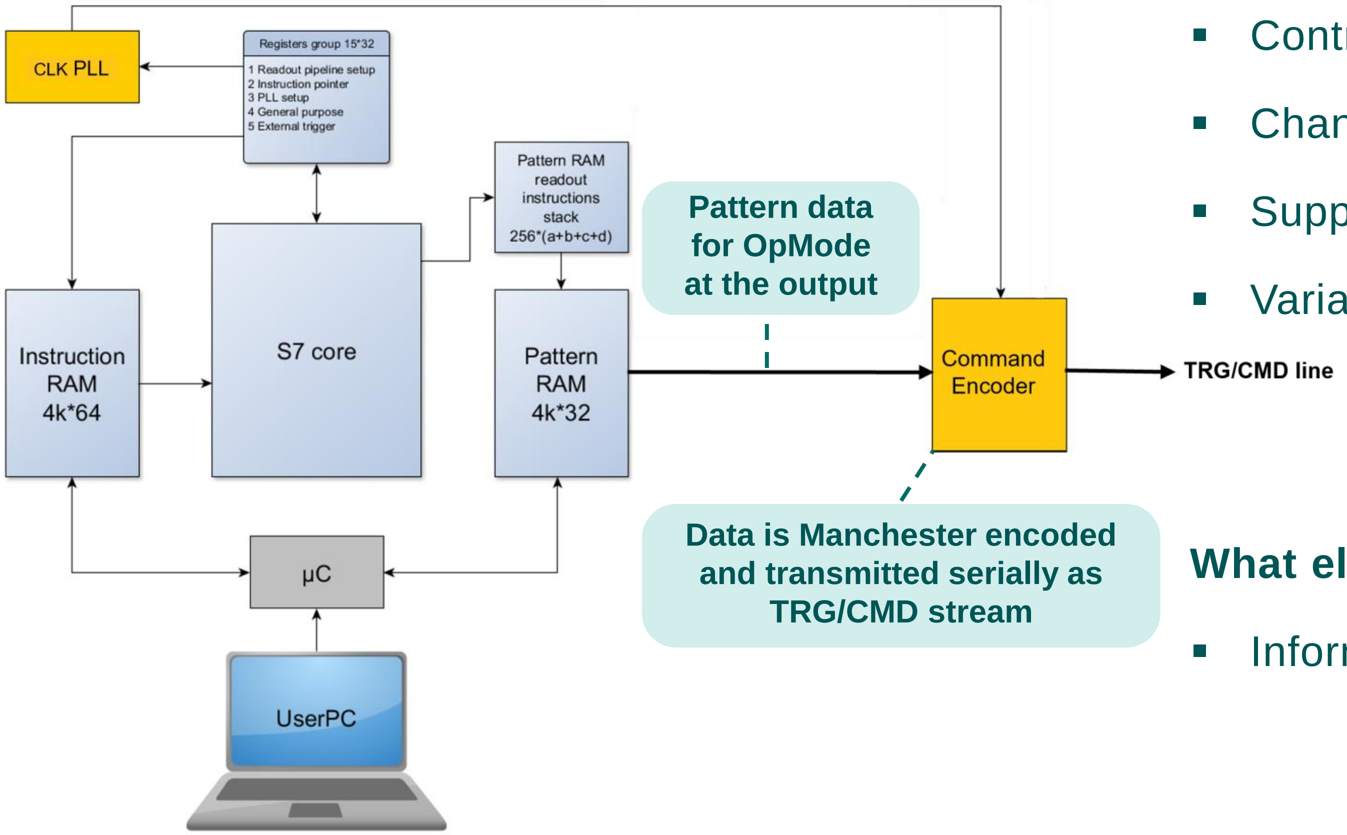
Data is Manchester encoded and transmitted serially as TRG/CMD stream

Coding scheme for TRG/CMD line

S7's Output (Hex code)	MODE	SEQ	Mnemonic	TRG/CMD
00	00	00	Stop A	**1010 1010**
01	00	01	Stop B	**1001 1010**
02	00	10	Stop C	**0110 1010**
03	00	11	Stop D	**0101 1010**
10	01	00	Idle A	**1010 1001**
11	01	01	Idle B	**1001 1001**
12	01	10	Idle C	**0110 1001**
13	01	11	Idle D	**0101 1001**
20	11	00	Run A	**1010 0101**
21	11	01	Run B	**1001 0101**
22	11	10	Run C	**0110 0101**
23	11	11	Run D	**0101 0101**
Others			Sync	**0001 1101**
AA			Sync	**0001 1101**
FF			Reset	**1010 0110**

- 8 output channels used to generate the pattern data

TRG/CMD SEQUENCER



Pattern data for OpMode at the output

Data is Manchester encoded and transmitted serially as TRG/CMD stream

What it can do:

- Controls repetitions, sequence and duration of modes
- Change its ouput in reaction to external triggers
- Supports parking, purging, exposure, windowing
- Variable durations of each state possible

What else do we need:

- Information for formatting primary stream of data frames

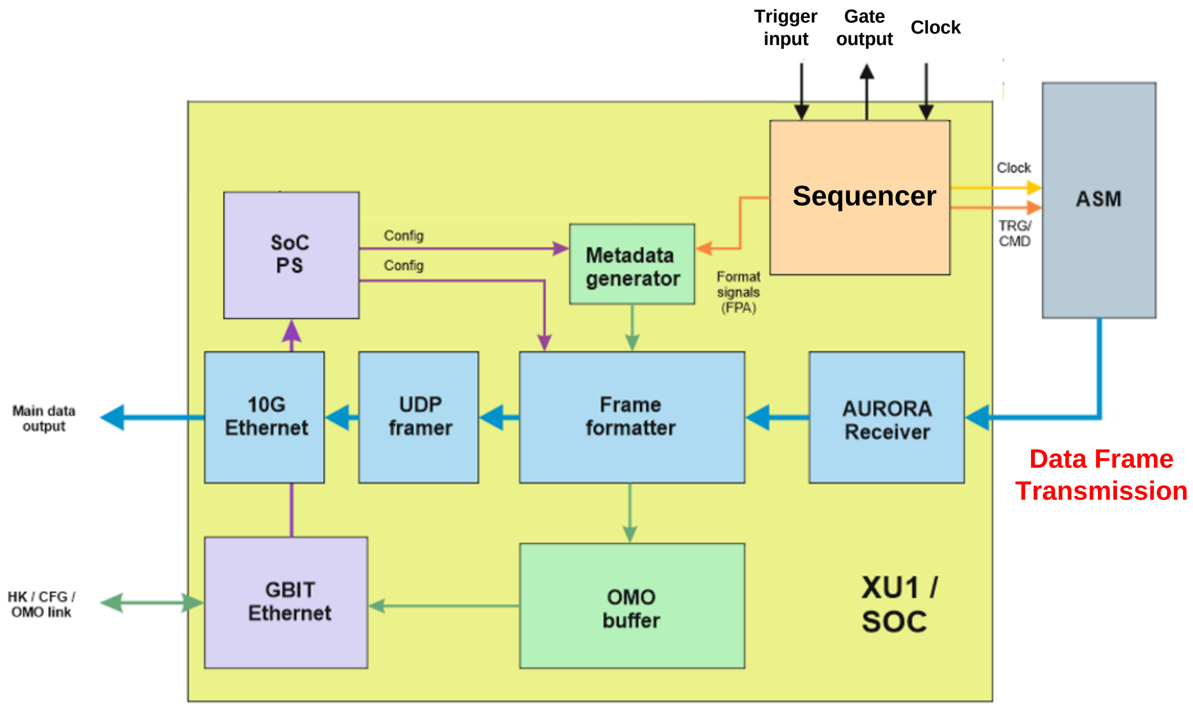
MODULE IP AND DATA CAPTURING

- DMC provides data in the form of AURORA data frames

Data Frame
- A single row
- A complete “Detector frame” (128 rows)
- A Burst of 50 detector frames
- Or any size in between

Data Frame Header
- DMC chip ID
- Timestamp value
- FIFO Overflow Tag

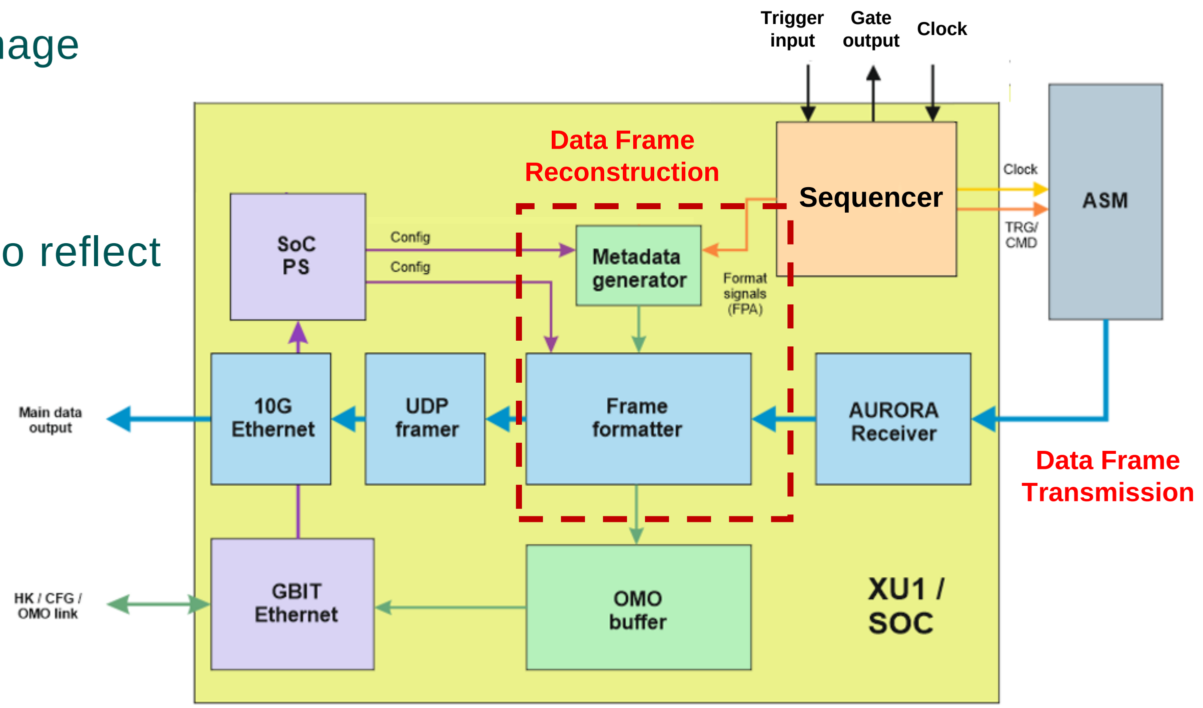
No information about
- Window size
- Burst size
- Timestamp of windows



Data Frame Transmission

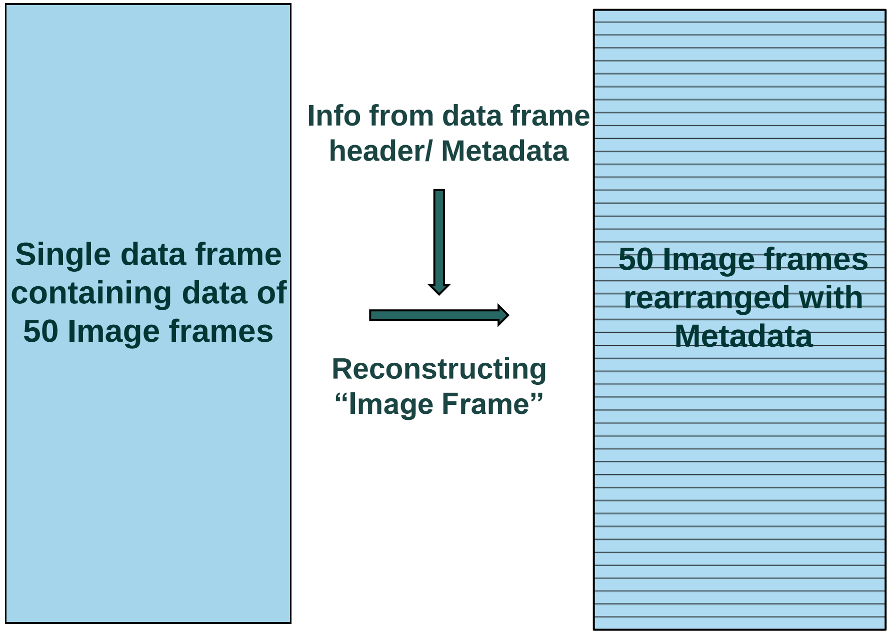
MODULE IP AND DATA CAPTURING

- DMC provides data in the form of AURORA data frames
- After data acquisition by the SoC, reconstruction of image frames from data frames is necessary
- Data from primary aurora stream is to be rearranged to reflect actual image frames and processed with Metadata



DATA CAPTURING

EXAMPLE


Single data frame containing data of 50 Image frames
Info from data frame header/ Metadata
Reconstructing "Image Frame"
50 Image frames rearranged with Metadata

Standard burst of 50 frames

Frame formatter have some information from data frame header

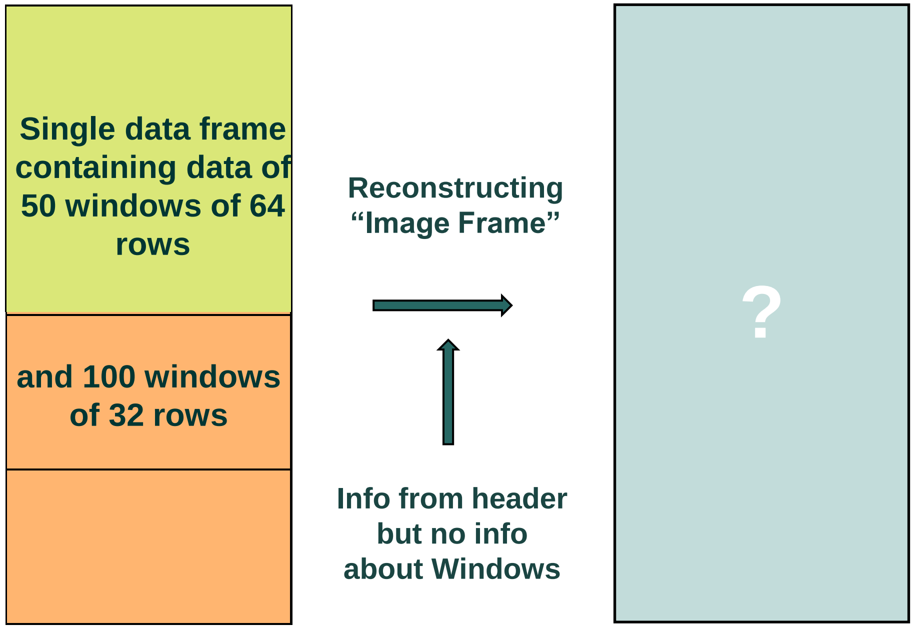
Comparatively simple to rearrange data frames to image frames

Window burst of 150 frames

Same number of rows are read as for the standard burst

Receiver does not have any information about windows

Individual windows are not distinguished as discrete frames


Single data frame containing data of 50 windows of 64 rows
and 100 windows of 32 rows
Reconstructing "Image Frame"
Info from header but no info about Windows
?

DATA CAPTURING AND RECONSTRUCTION

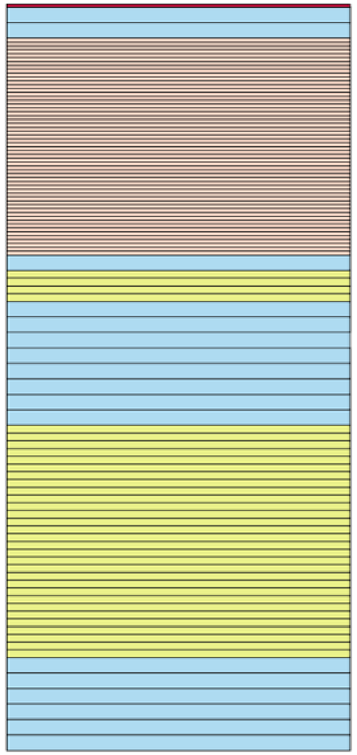
Added complexity to reconstruct a burst with variable window sizes

Sequencer should be able to:

- Keep track of the operation mode
- Follow any changes in the sequence in response to an external trigger
- Convey this information to the Frame Formatter
- Provide information about frame/window/burst size

Accurate reconstruction of primary stream of **data frames** to stream of **image frames** requires processing using the **format data** from the Sequencer

This is also used to generate the **Metadata** to be added to the image frames



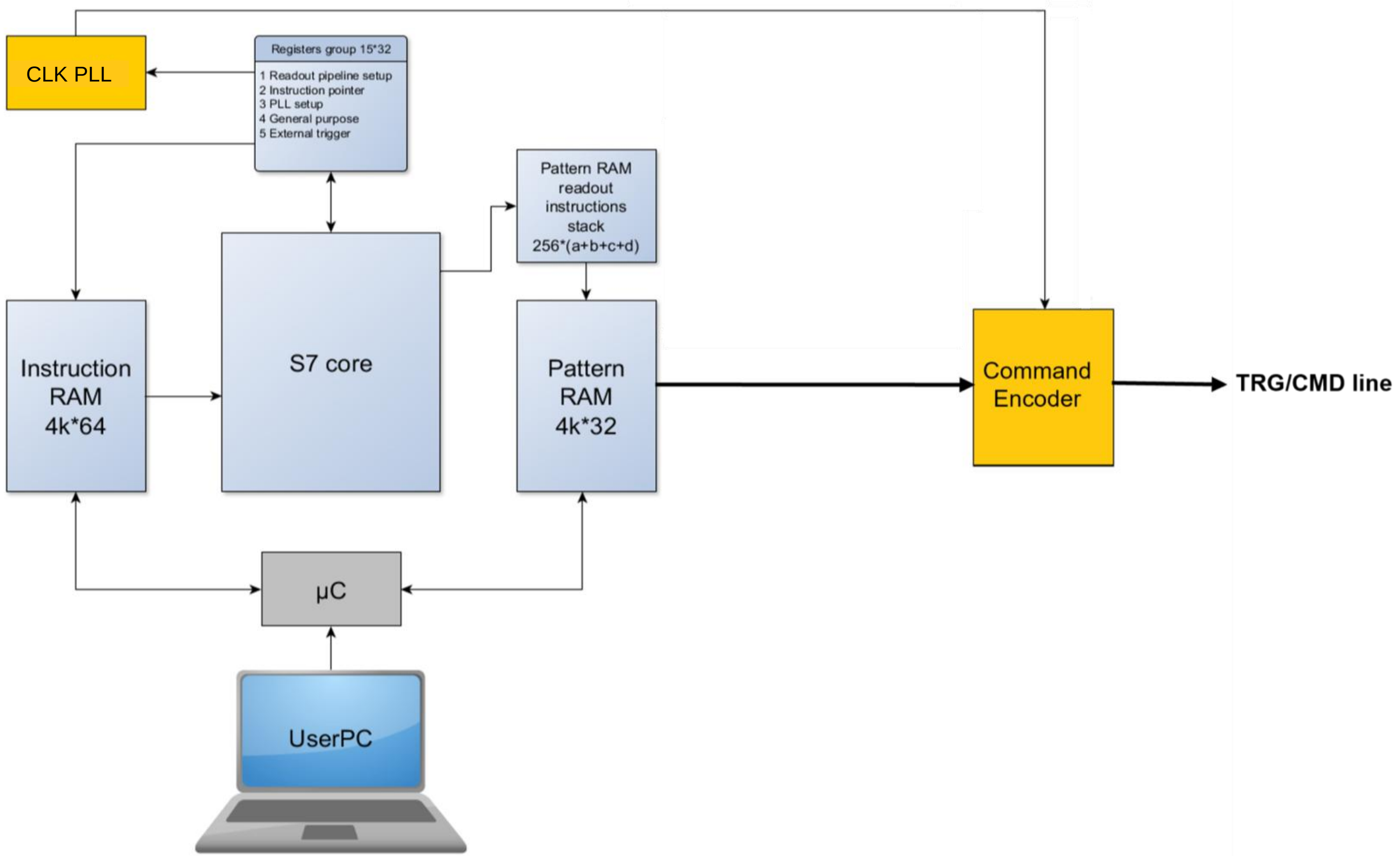
Window burst with

17 frames with 128 rows

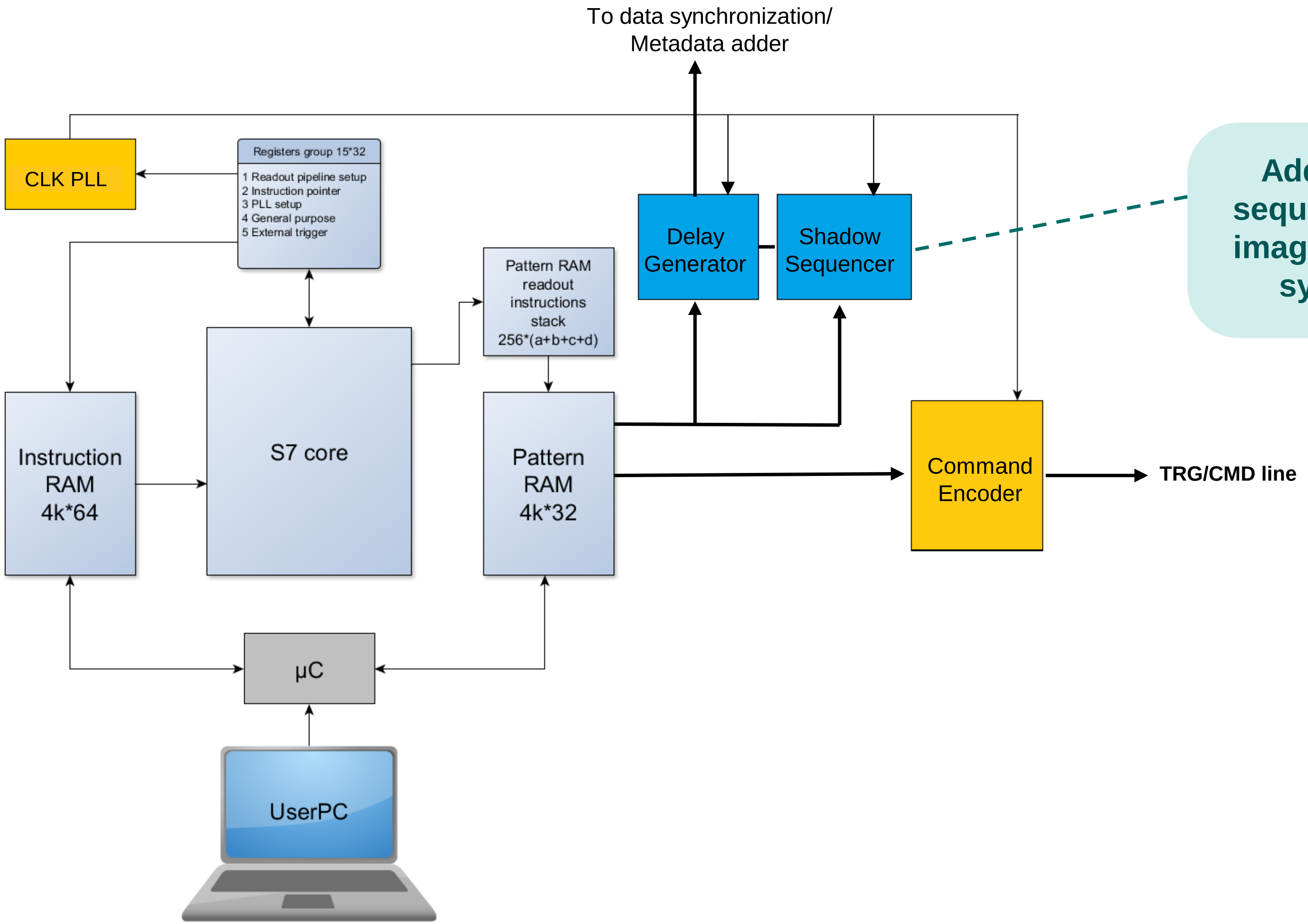
34 window frames with 64 rows

56 window frames with 32 rows

TRG/CMD SEQUENCER – MODIFICATIONS FOR EDET



TRG/CMD SEQUENCER – MODIFICATIONS FOR EDET



Additional blocks for tracking the sequences, providing information for image reconstruction, adding delays, synchronization with the FPGA

SUMMARY AND NEXT STEPS

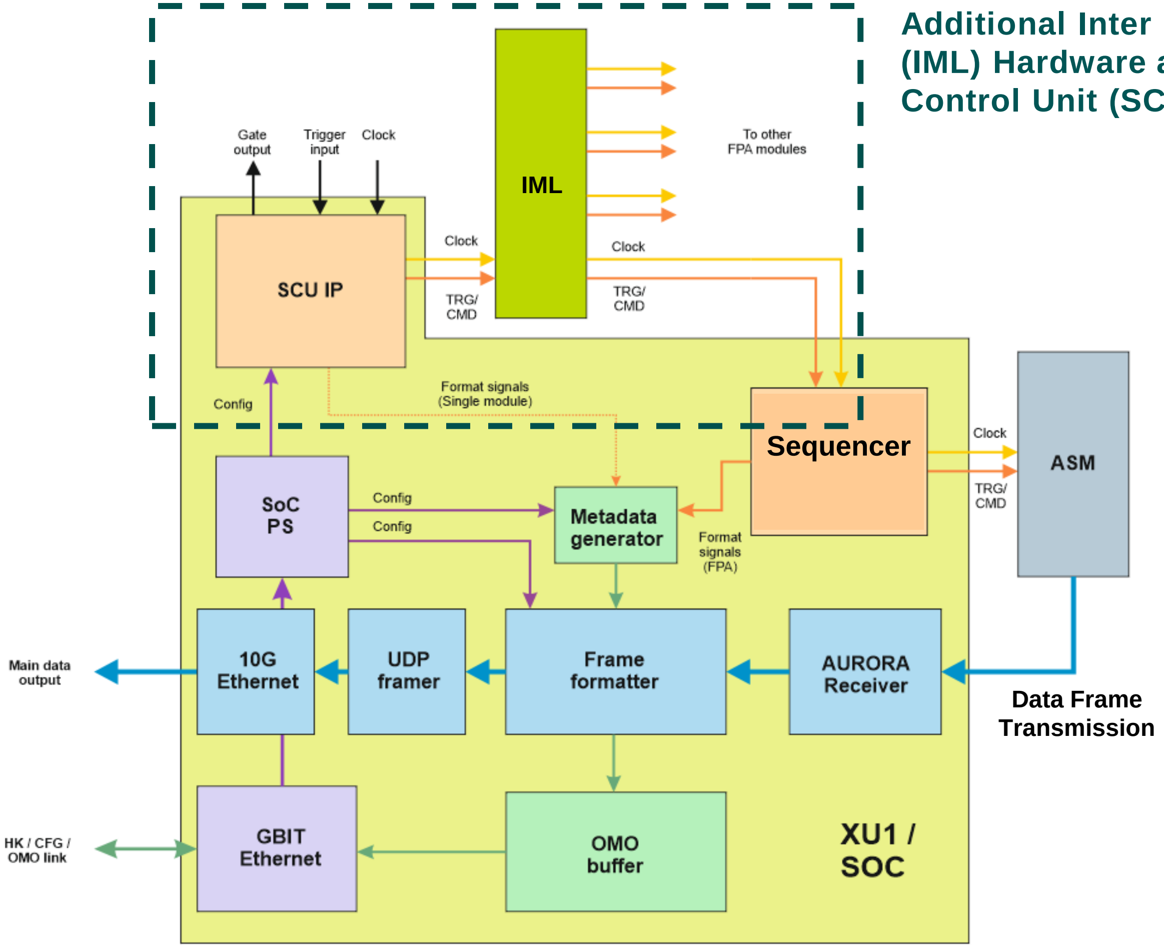
- Upgraded Sequencer IP to support DMC 65 features
- Implementation of different OpModes possible
- Supports parking, exposure, purging windowing
- Reaction to external trigger signals

Looking ahead:

- Experimental testing of the sequencer firmware on the small prototype system
- Addition of Metadata to fast data stream during data acquisition
- Synchronization of firmware and data capturing routine

OUTLOOK – SYSTEM IP

Expansion of the firmware to operate all four quadrants



Additional Inter Module Link (IML) Hardware and System Control Unit (SCU) IP required

"We'll say it's a 10 year plan. We'll all be long gone by the time they try to figure it out."

THANK YOU FOR YOUR ATTENTION

Halbleiterlabor der Max-Planck-Gesellschaft

Mishal Rizwan

Isarauenweg 1

85748 Garching bei München

Tel.: +49 (0)89 839400-23

E-Mail: mir@hll.mpg.de

Internet: www.hll.mpg.de

Comments/ Questions?

BACKUP SLIDES

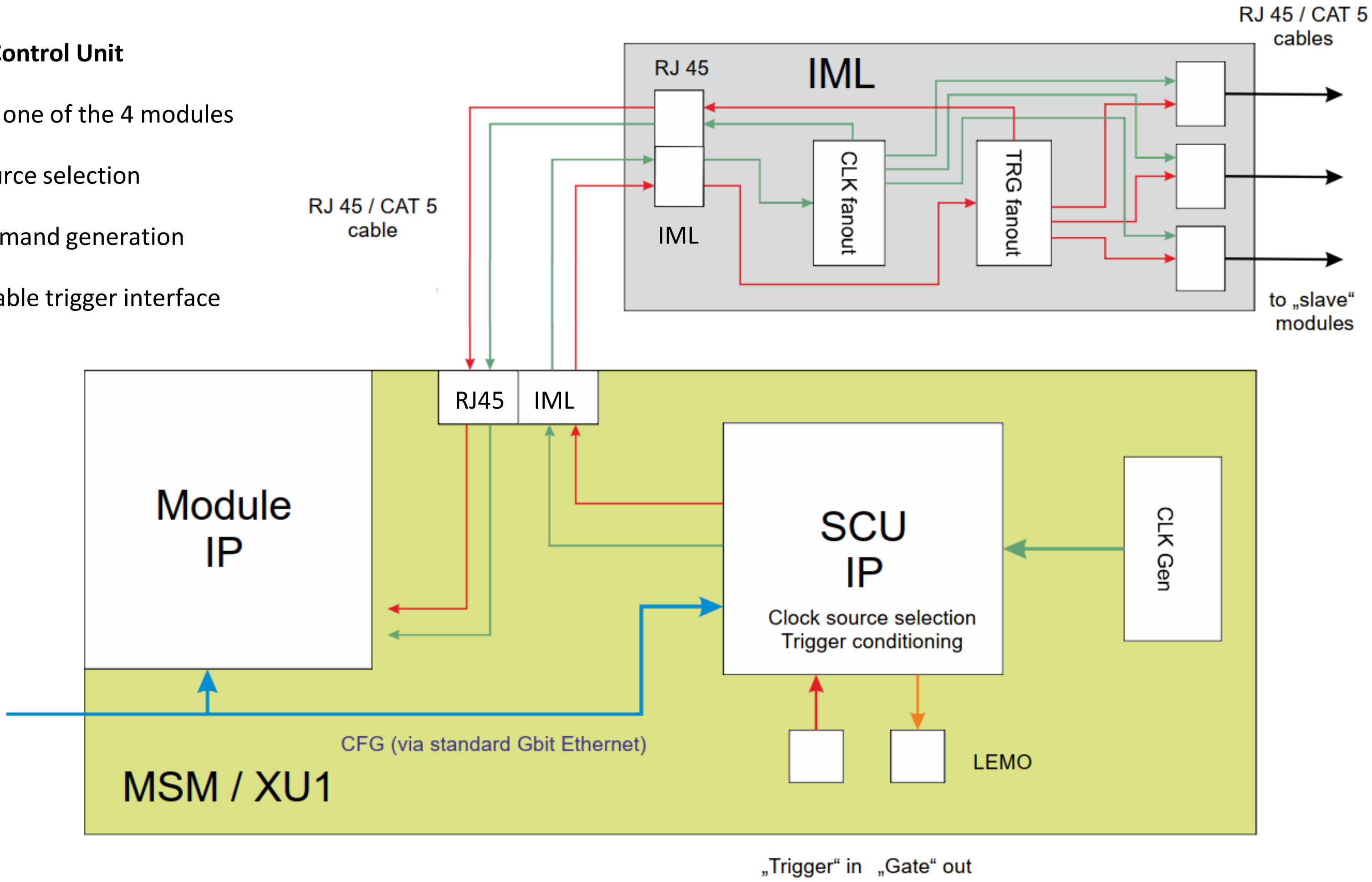
System Control Unit

SCU IP in one of the 4 modules

Clock source selection

TRG command generation

Configurable trigger interface



IML Hardware

Consists essentially of a two-fold LVDS fanout

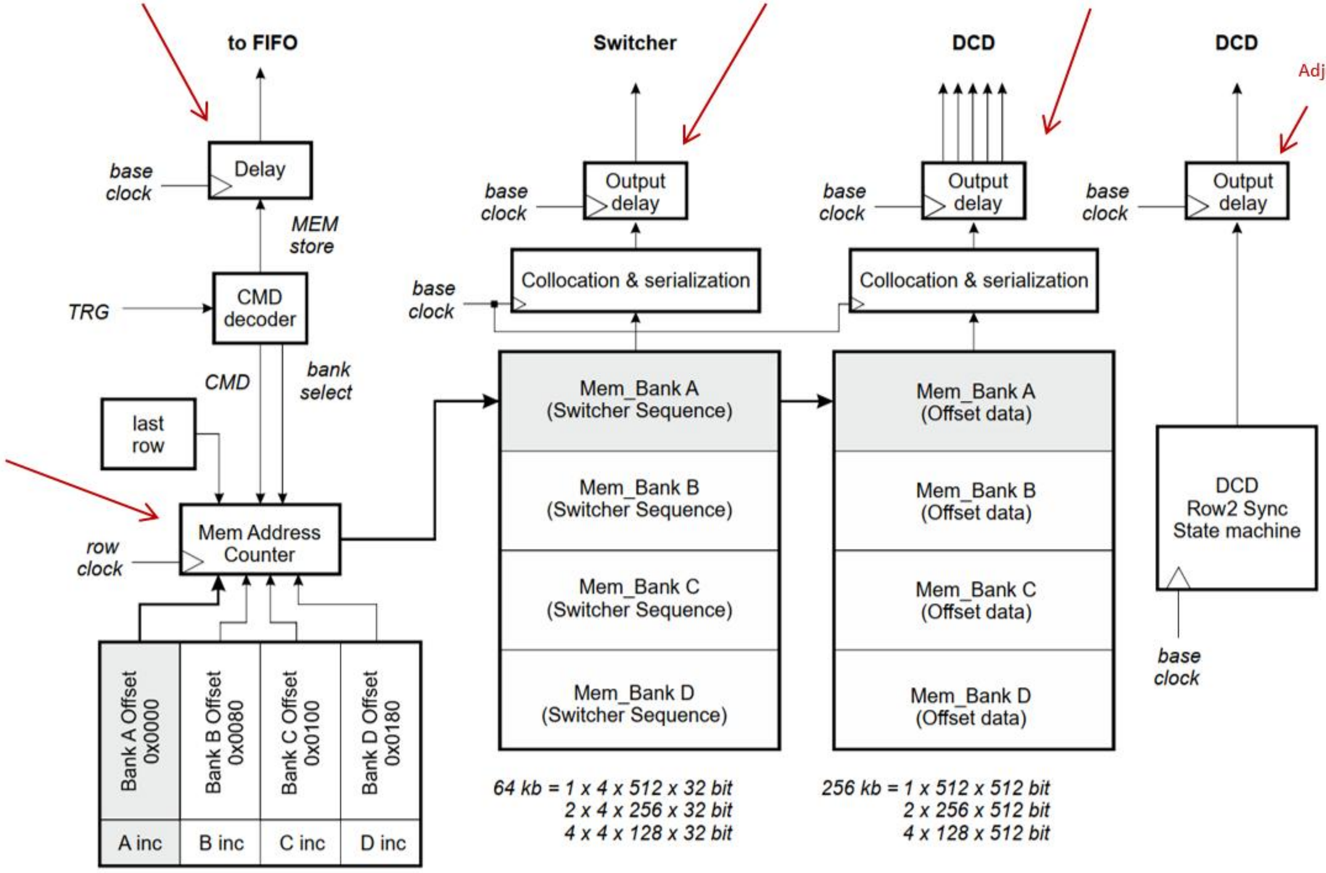
Delay adjustment via identical cables

TRG/CMD distribution to other modules

Coordinated operation of 4 modules on one FPA

Distribution of clock and trigger signals

- Up to four independently selectable pattern memory banks
- Individually commandable for every 100 ns readout cycle
- Stop mode with individual parameter set for each trace:
  - Loop / Freeze mode setting
  - Freeze can be enabled for every track individually for each bank
  - Bank dependent freeze position
- Programmable address offsets after RESET condition
- Programmable return address
- Flexible use of memory banks
- Bank individual memory increment for
  - Fast purging
  - Fast forwarding (e.g. during windowing)



Accounts for DCD data latency (192 clocks)

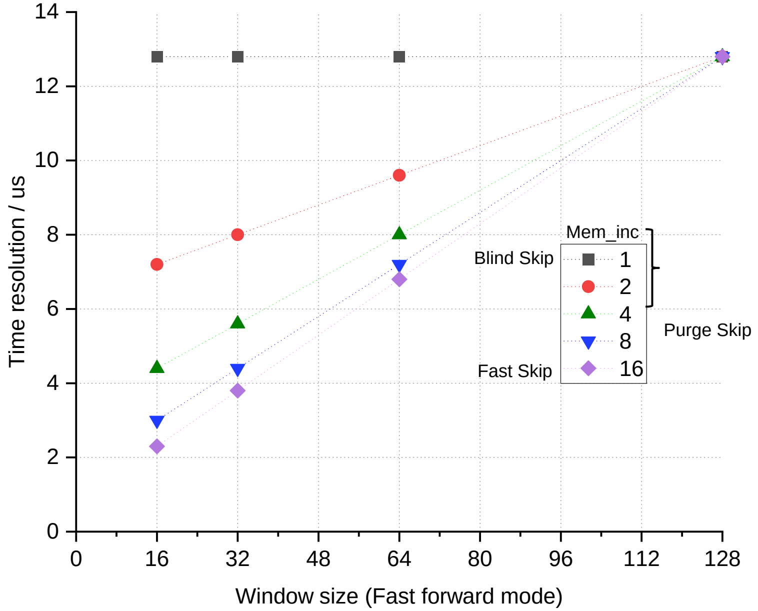
Apply switcher sequence w/ 42 clocks latency

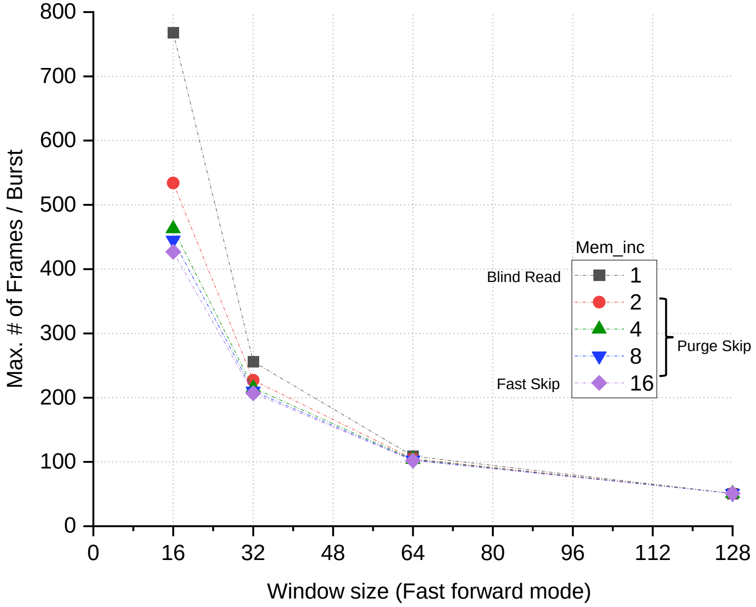
Apply offsets first

Adjust phase

Common for offset and sequencer memory

WINDOW MODE





$$t_{res} = (\# \text{ of } Rows_{read} + \# \text{ of } Rows_{fastforward}/mem\_inc)*100ns$$

Window size (Read)	16	32	64	128
Rows skipped	112	96	64	0

- Tradeoff between time resolution and movie length (max. # of frames/burst)

- Fastest read (2.3us) for the smallest window size for mem_inc = 16 and window size = 1