Development of an FPGA emulator for the RD53B chip

NIHARIKA MITTAL
ACME LAB
UNIVERSITY OF WASHINGTON

GIT: https://gitlab.com/scotthauck/largehadroncollider

UNIVERSITY of WASHINGTON

Figure 1: Aerial depiction of Large Hadron Collider and its experiment sites [CERN]

Figure 1: Aerial depiction of Large Hadron Collider and its experiment sites [CERN]

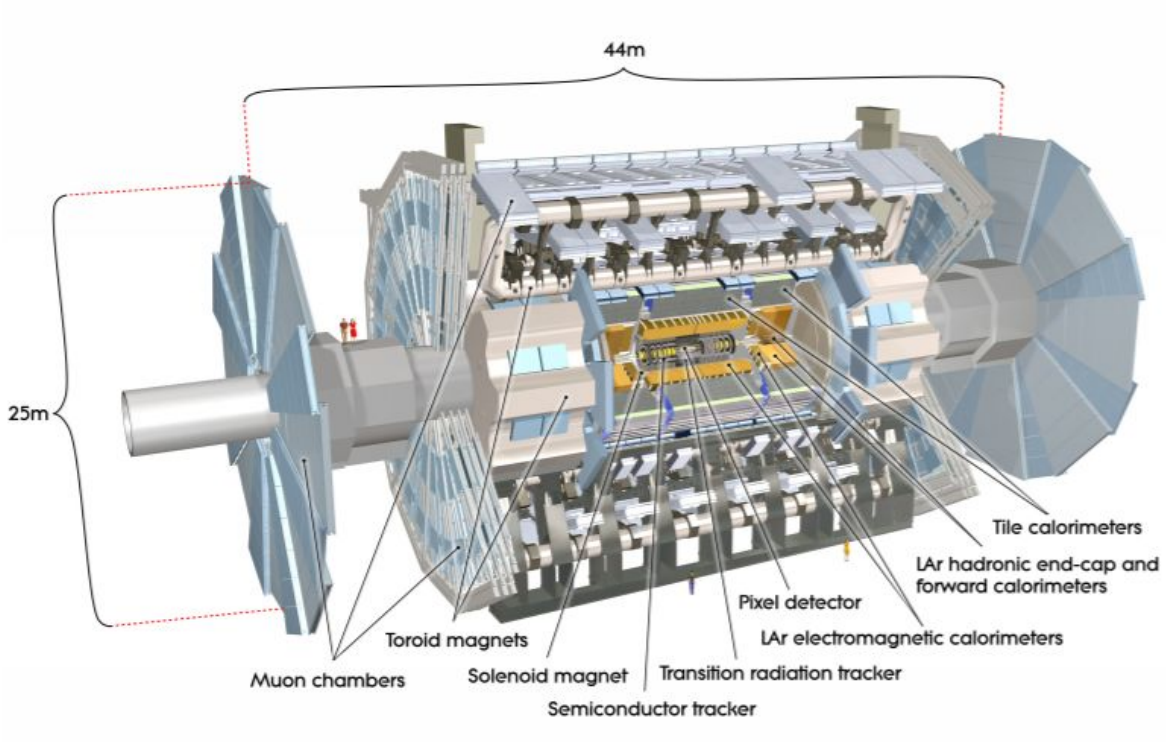


Figure 2: ATLAS detector [CERN]

Image Source

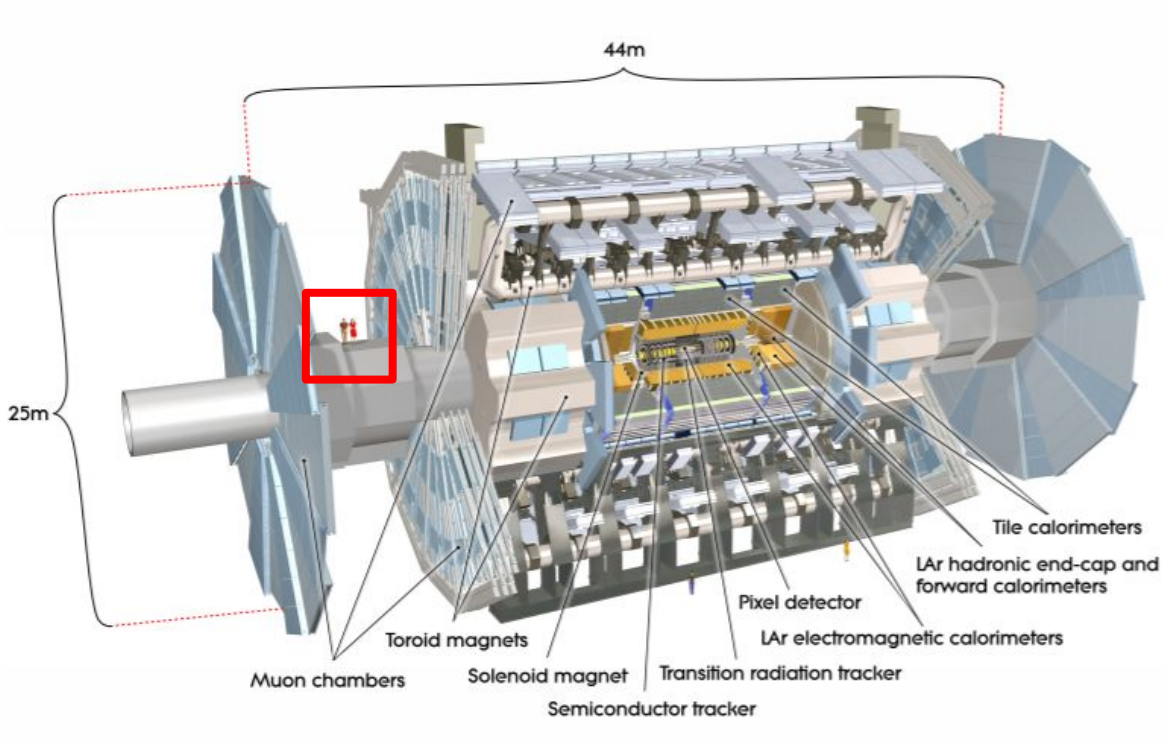


Figure 2: ATLAS detector [CERN]

Image Source

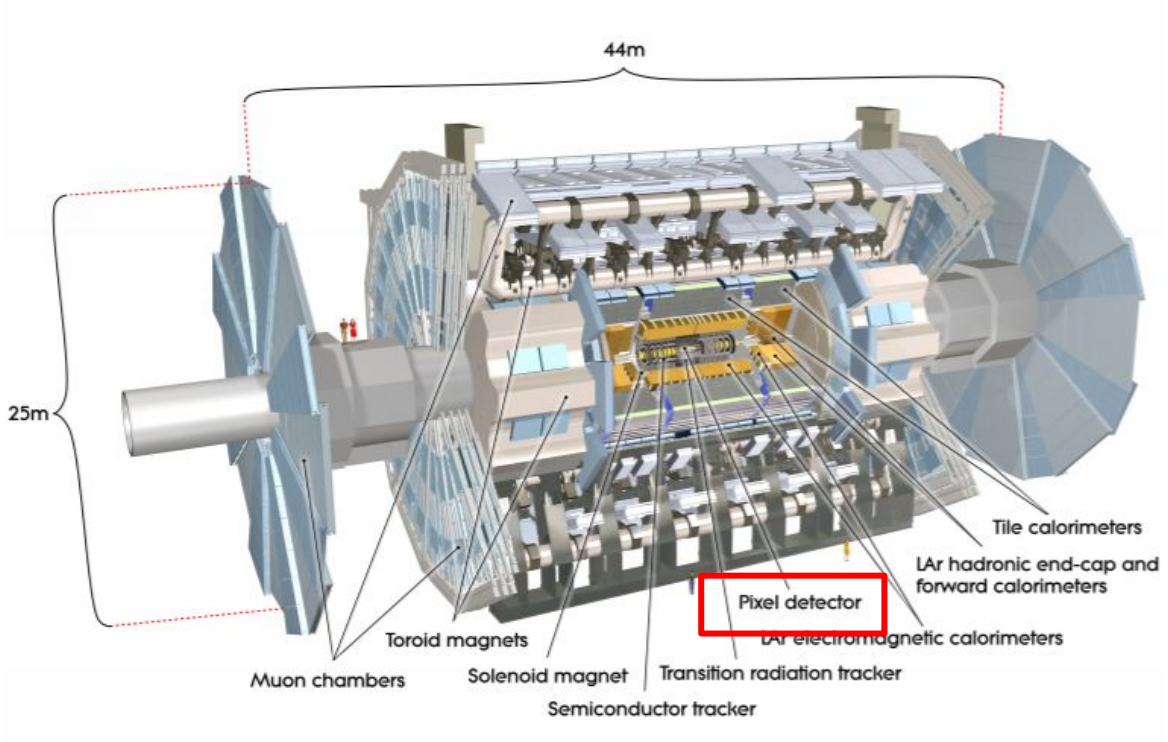
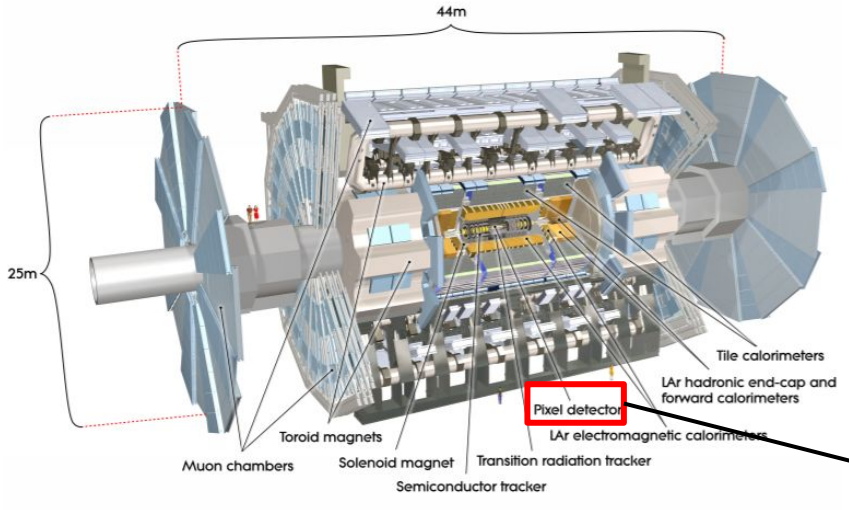


Figure 2: ATLAS detector [CERN]

Image Source



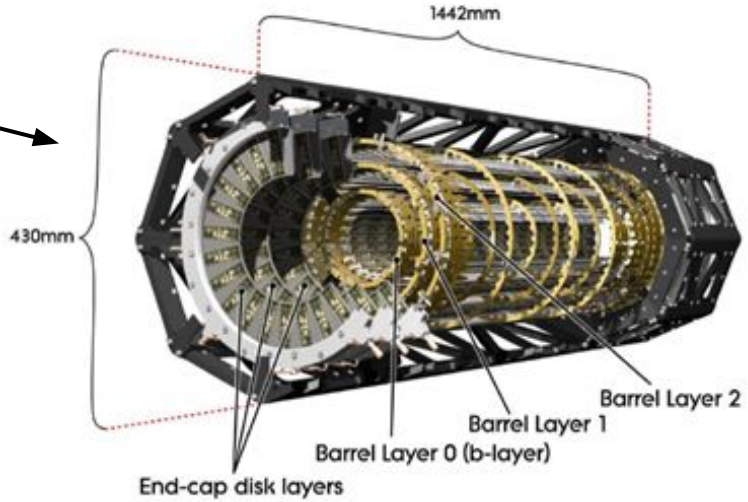


Figure 3: Pixel Detector [CERN]

Image Source

HL-LHC/ ITk PIXEL UPGRADE

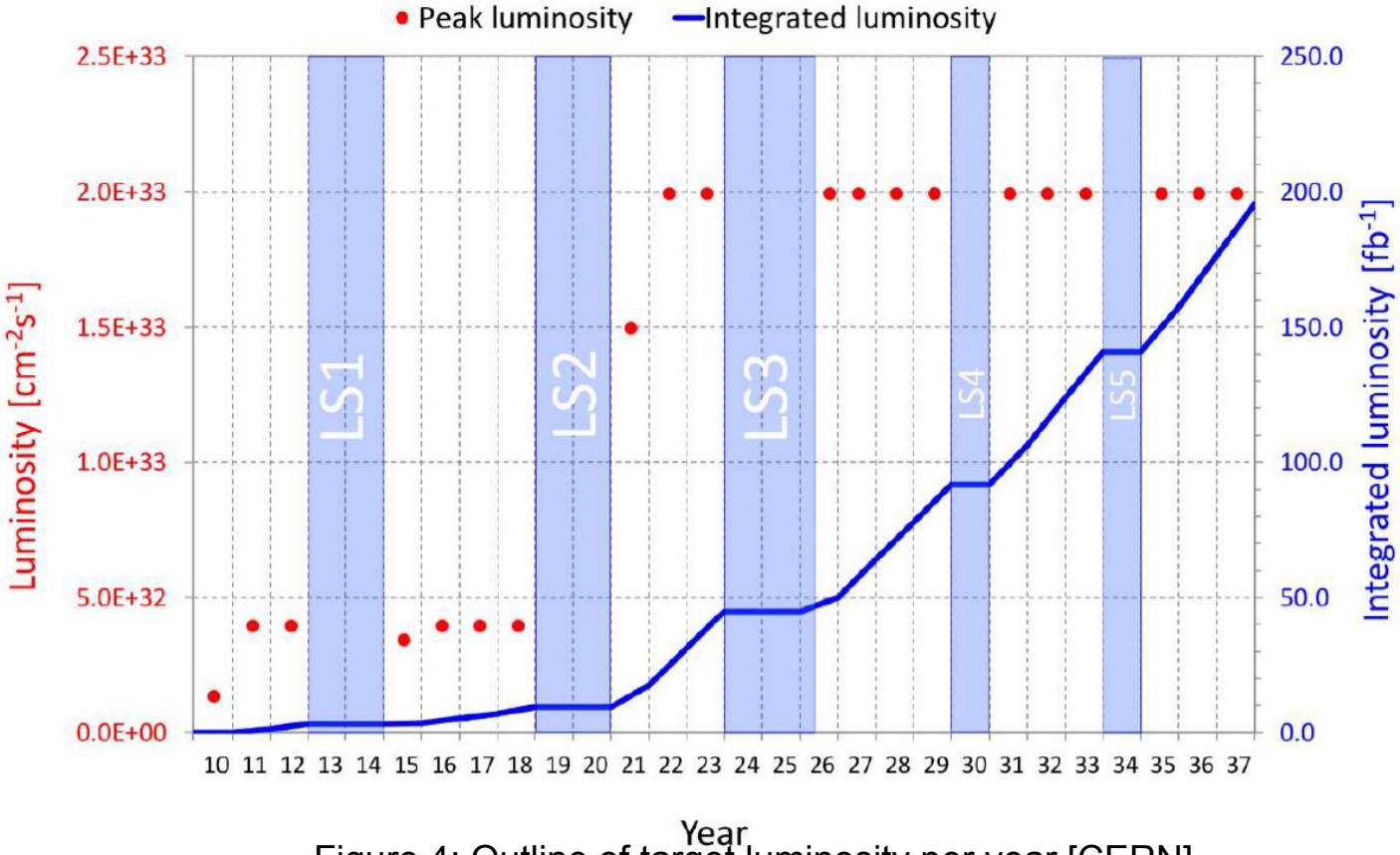


Figure 4: Outline of target luminosity per year [CERN]

Image Source

RD53B CHIP

RD53B is a pixel readout chip framework

The design framework is built upon the RD53A framework

	RD53A	**RD53B**
No of pixels	76800	153600

EMULATOR

Hardware or software that enables one computer system to behave like another computer system

EMULATOR

Hardware or software that enables one computer system to behave like another computer system

RD53B EMULATOR

Mapping of important features of the RD53B ASIC design onto an FPGA using Verilog HDL to allow communications testing

PURPOSE OF RD53B EMULATOR

- A flexible model of the RD53B implemented on an FPGA
- Helps in testing/debugging of DAQ (Data Acquisition)
- Helps to debug the RD53B chip
- Produces more “realistic” hit (collision) data

Before B there was A

RD53A EMULATOR

Formats and decodes input

⇩

Based on the commands received, generates corresponding data

⇩

Generated data get formatted as per RD53A protocol

⇩

Formatted data is sent out through 4 lane system

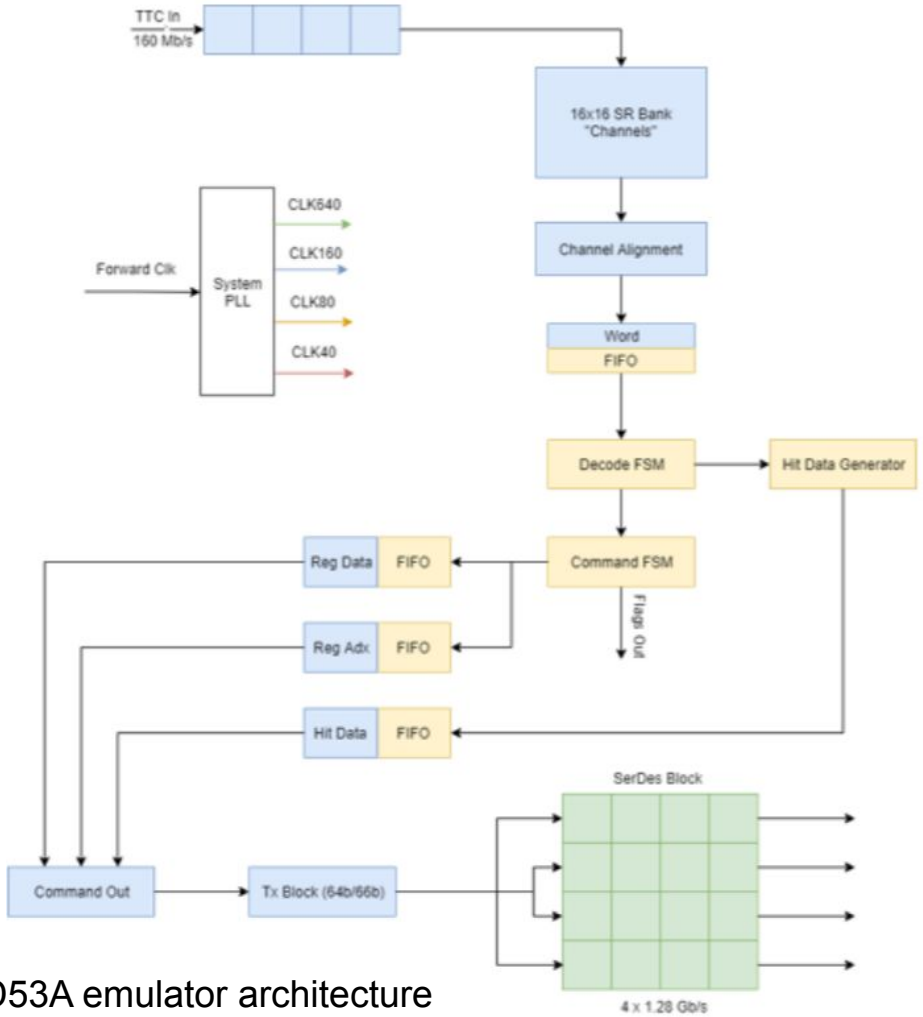


Figure 5: RD53A emulator architecture

RD53B EMULATOR

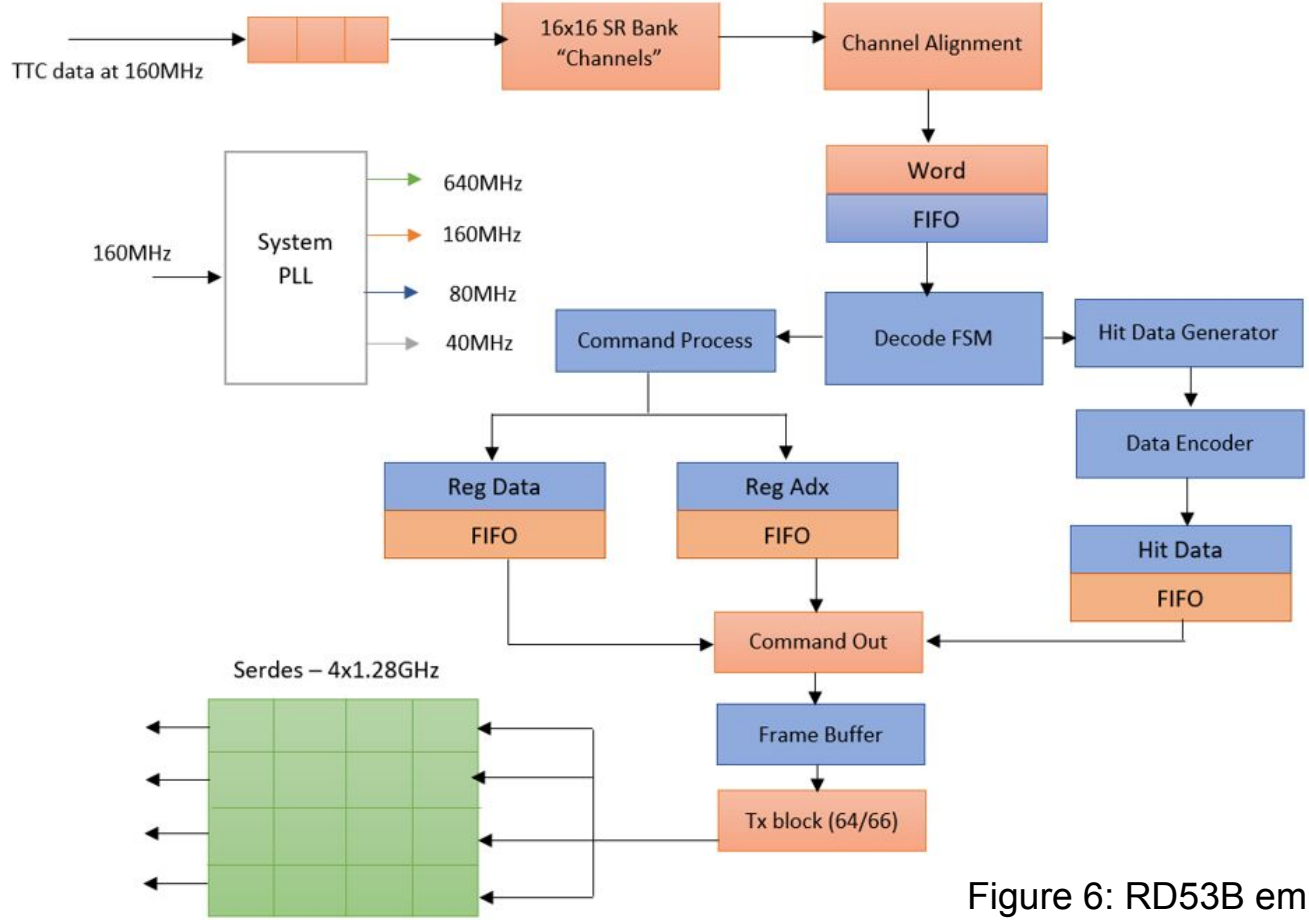


Figure 6: RD53B emulator architecture

RD53B EMULATOR

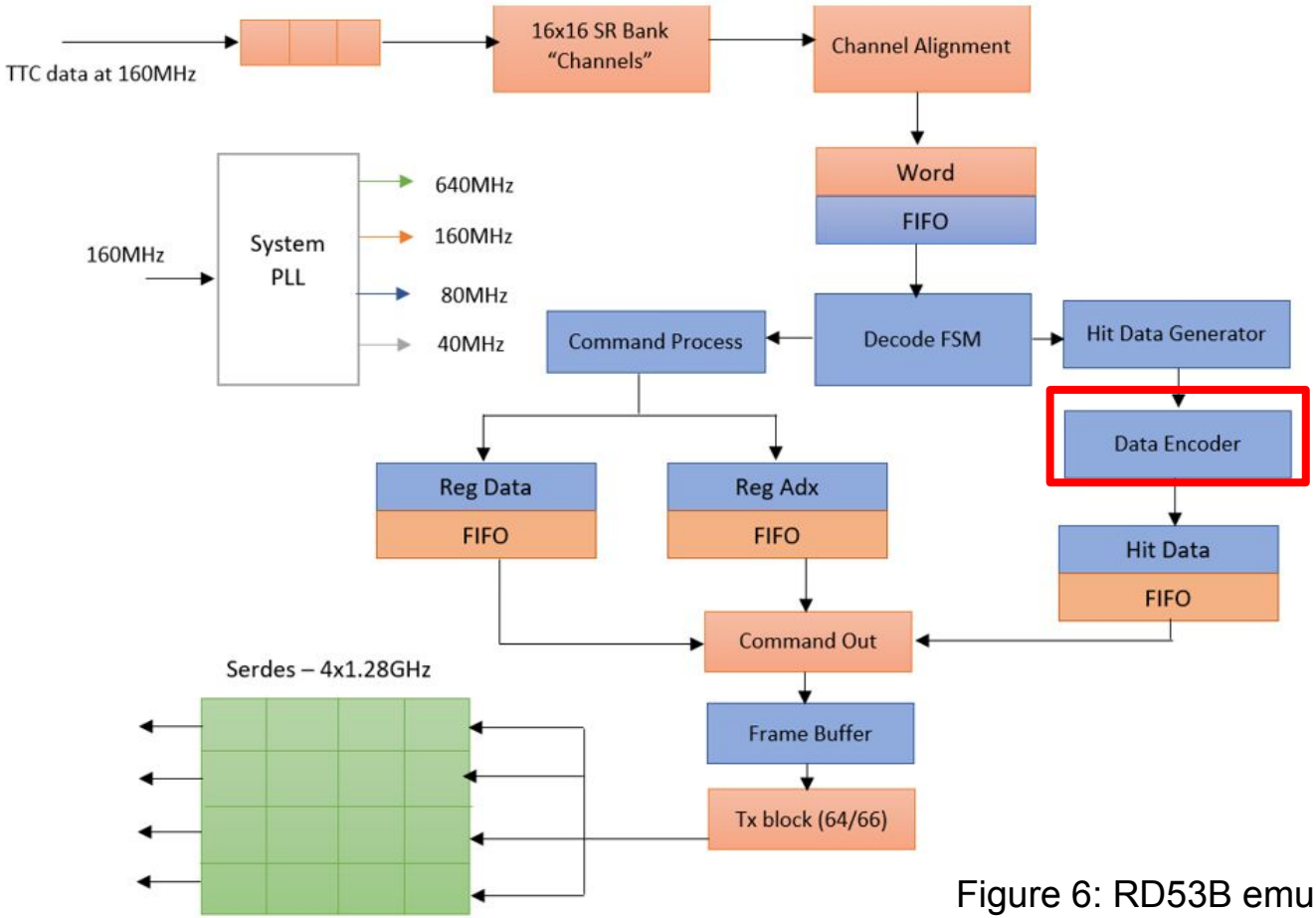
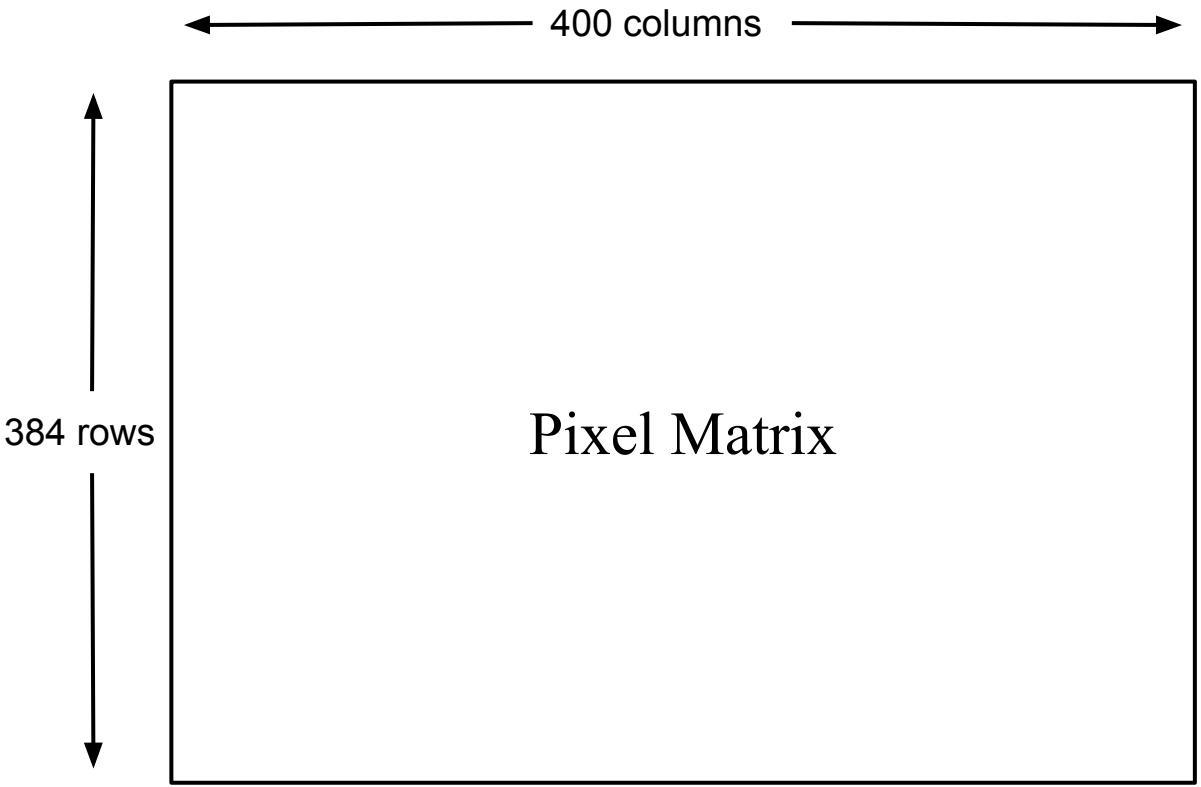
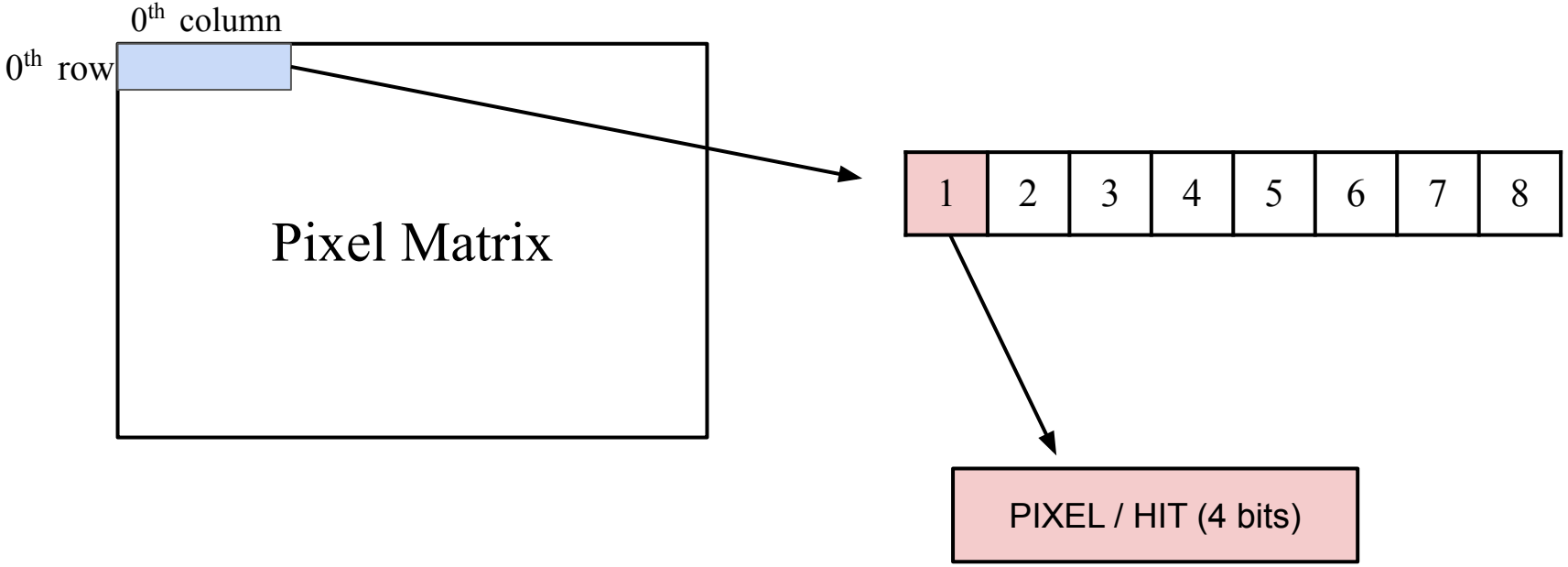


Figure 6: RD53B emulator architecture

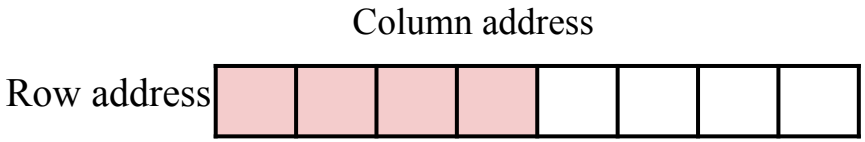
DATA ENCODING

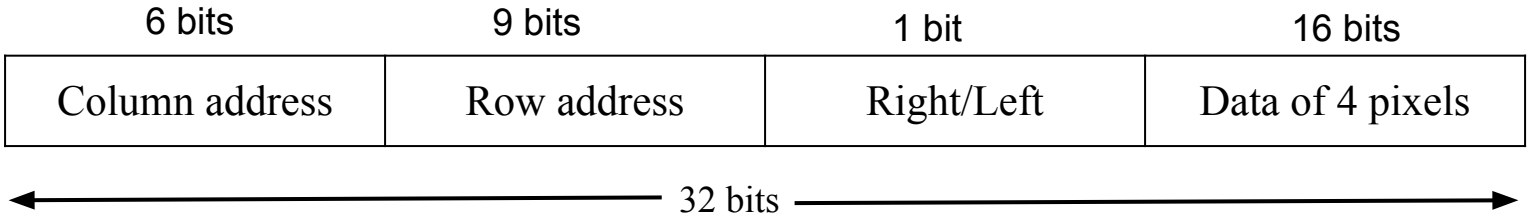


DATA ENCODING

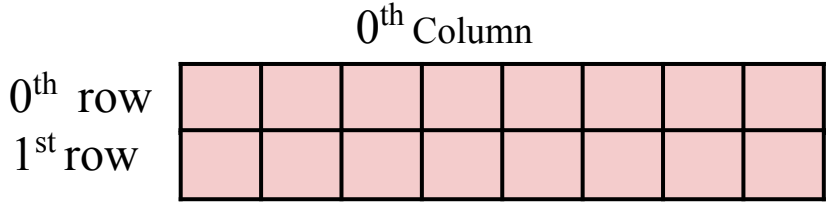


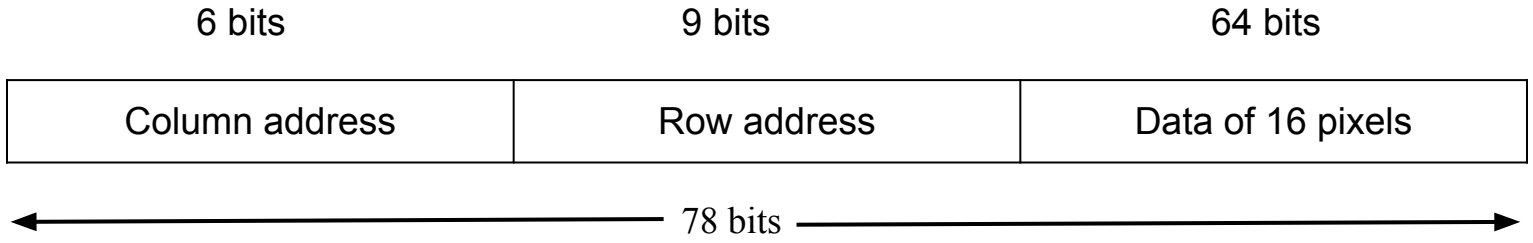
RD53A HIT DATA



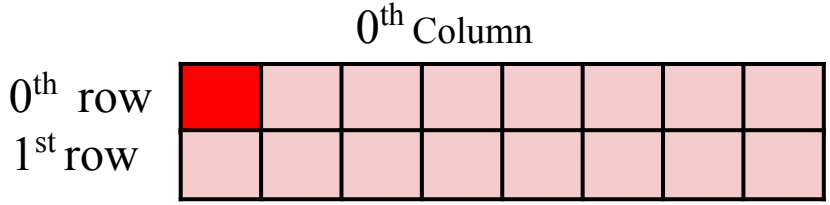


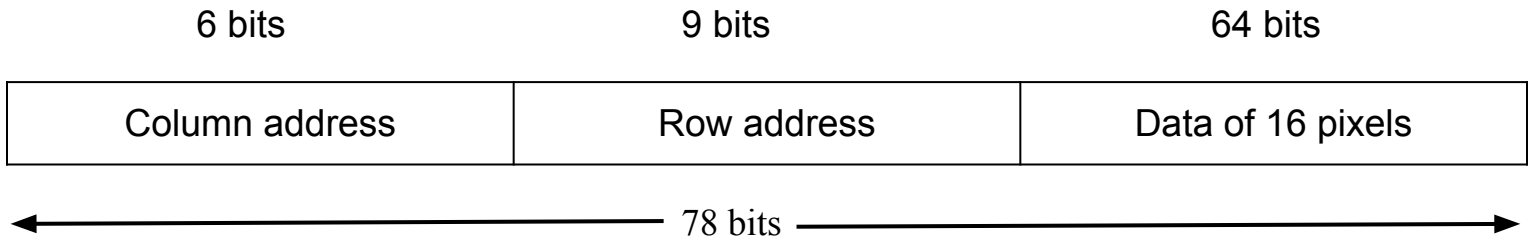
RD53B HIT DATA



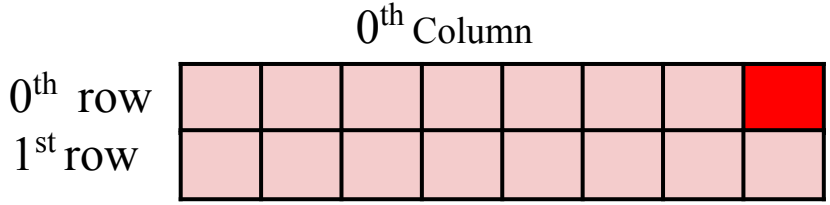


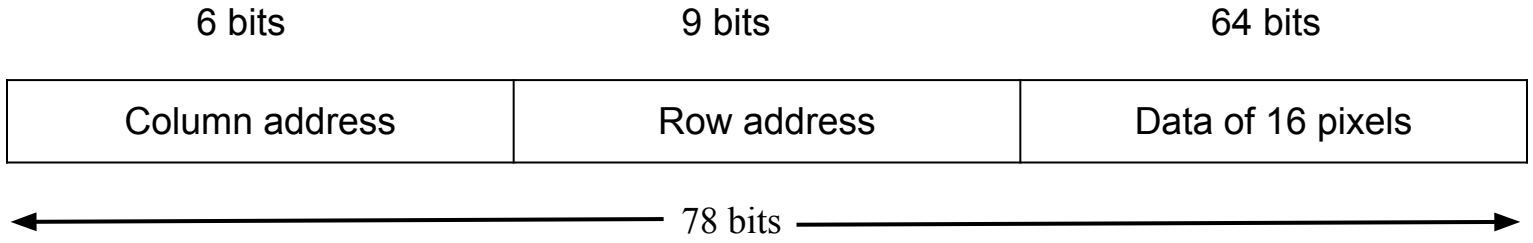
RD53B HIT DATA



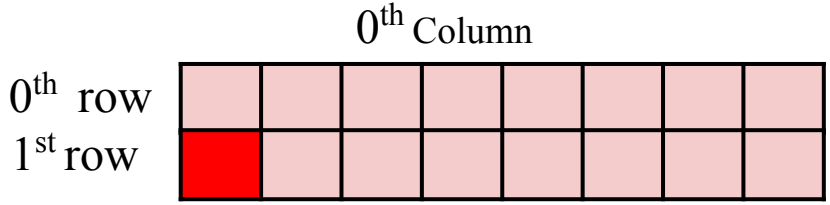


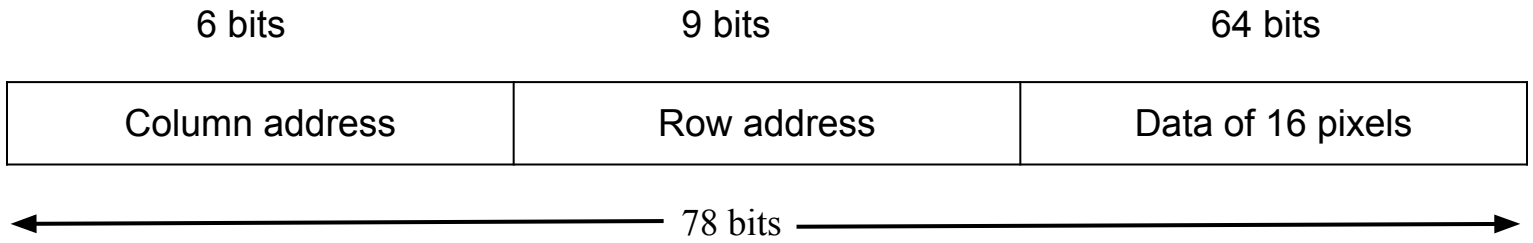
RD53B HIT DATA



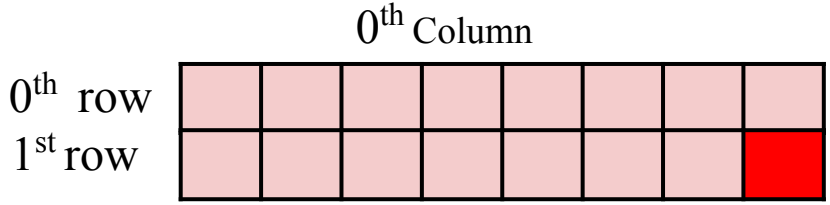


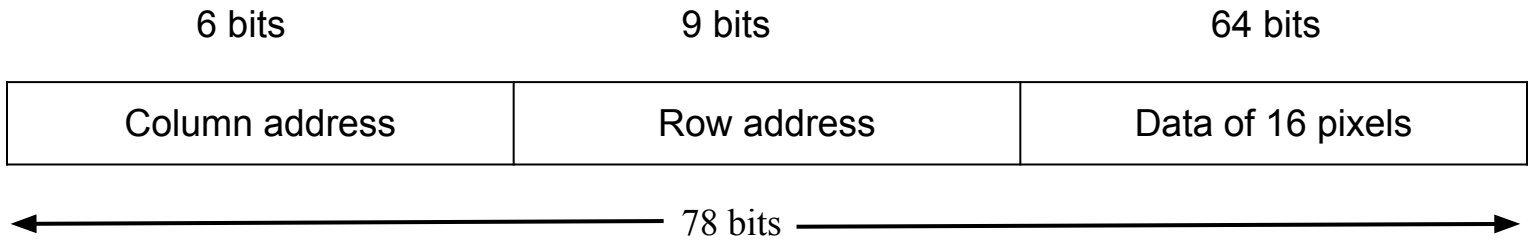
RD53B HIT DATA



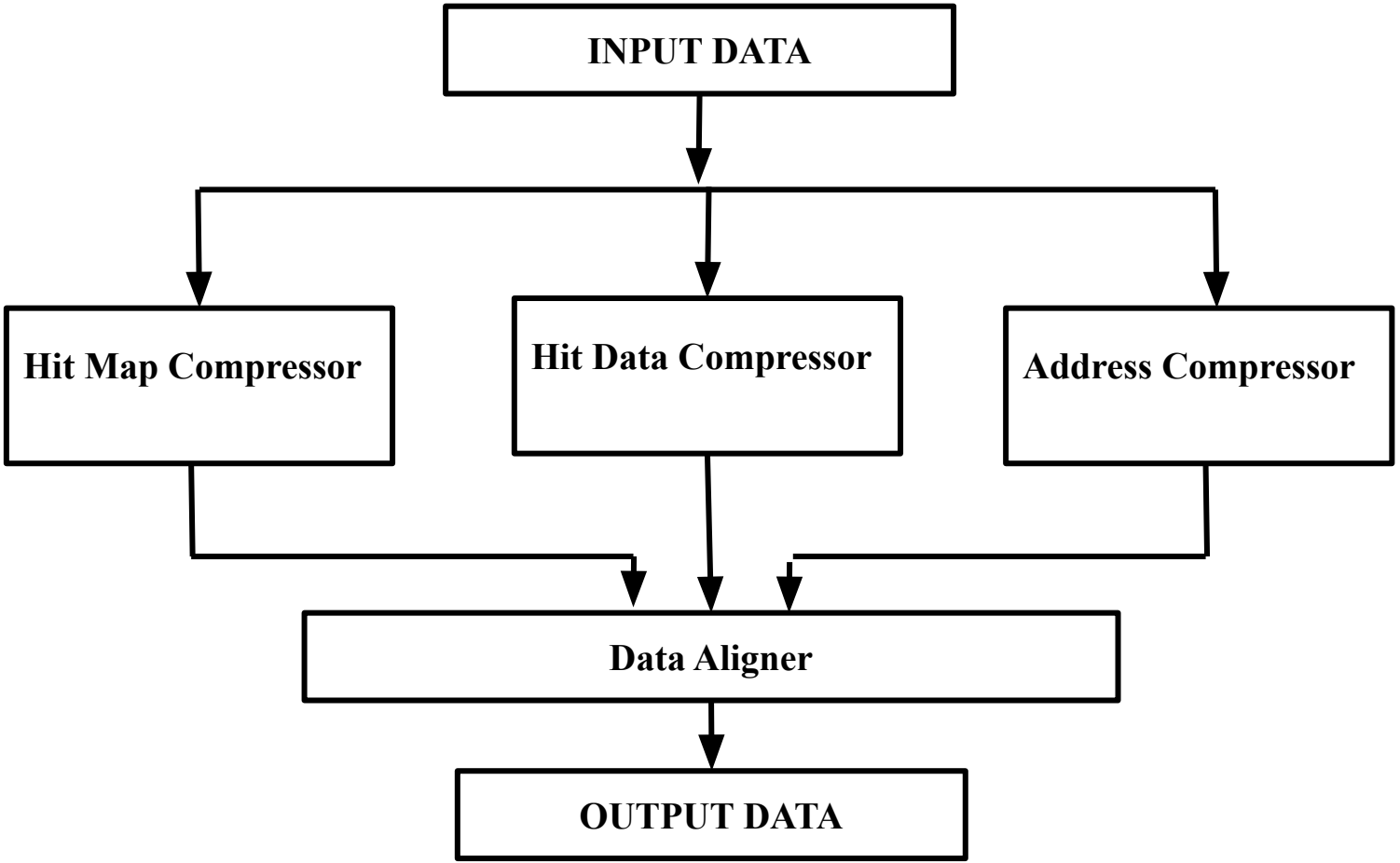


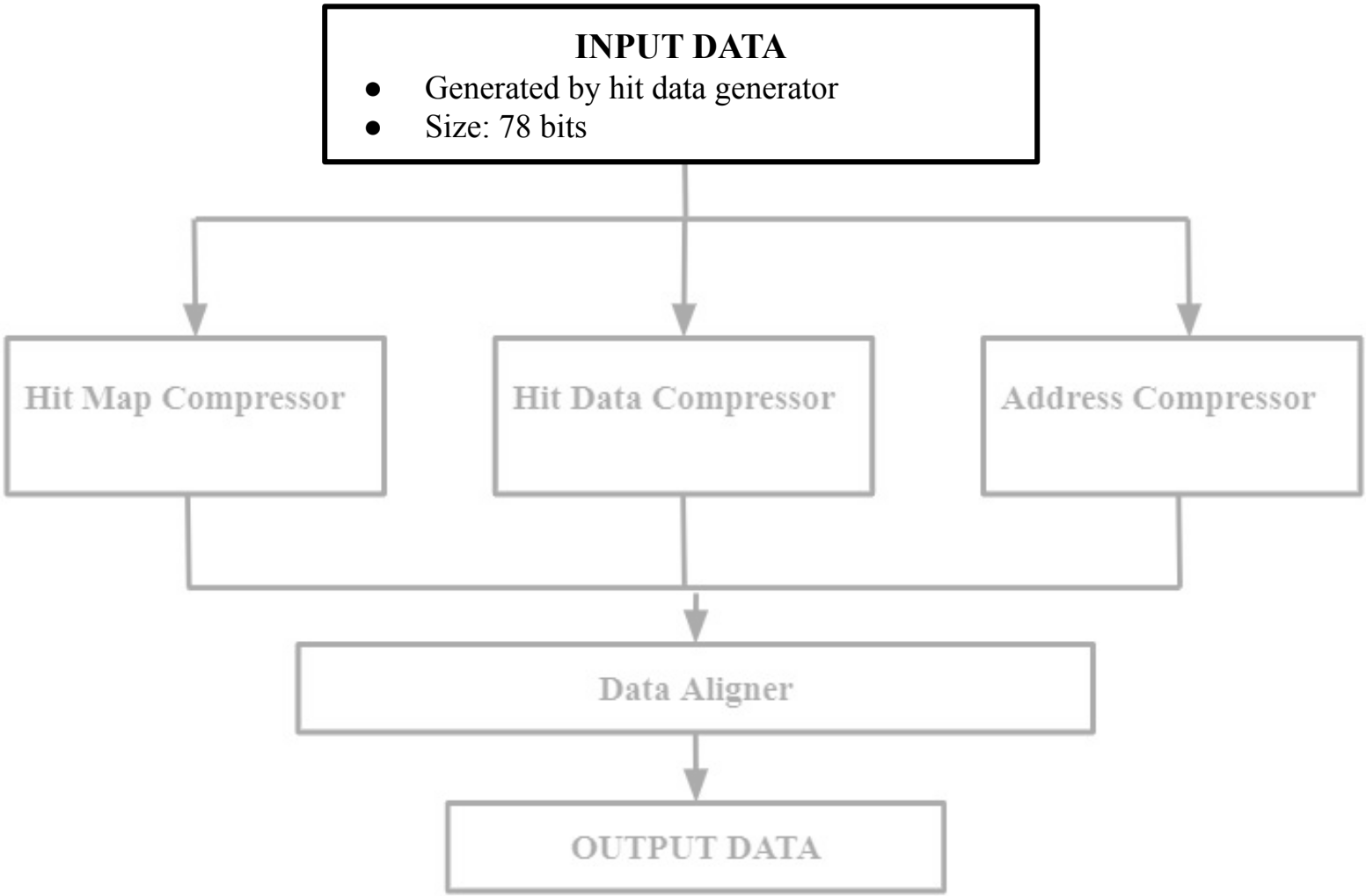
RD53B HIT DATA

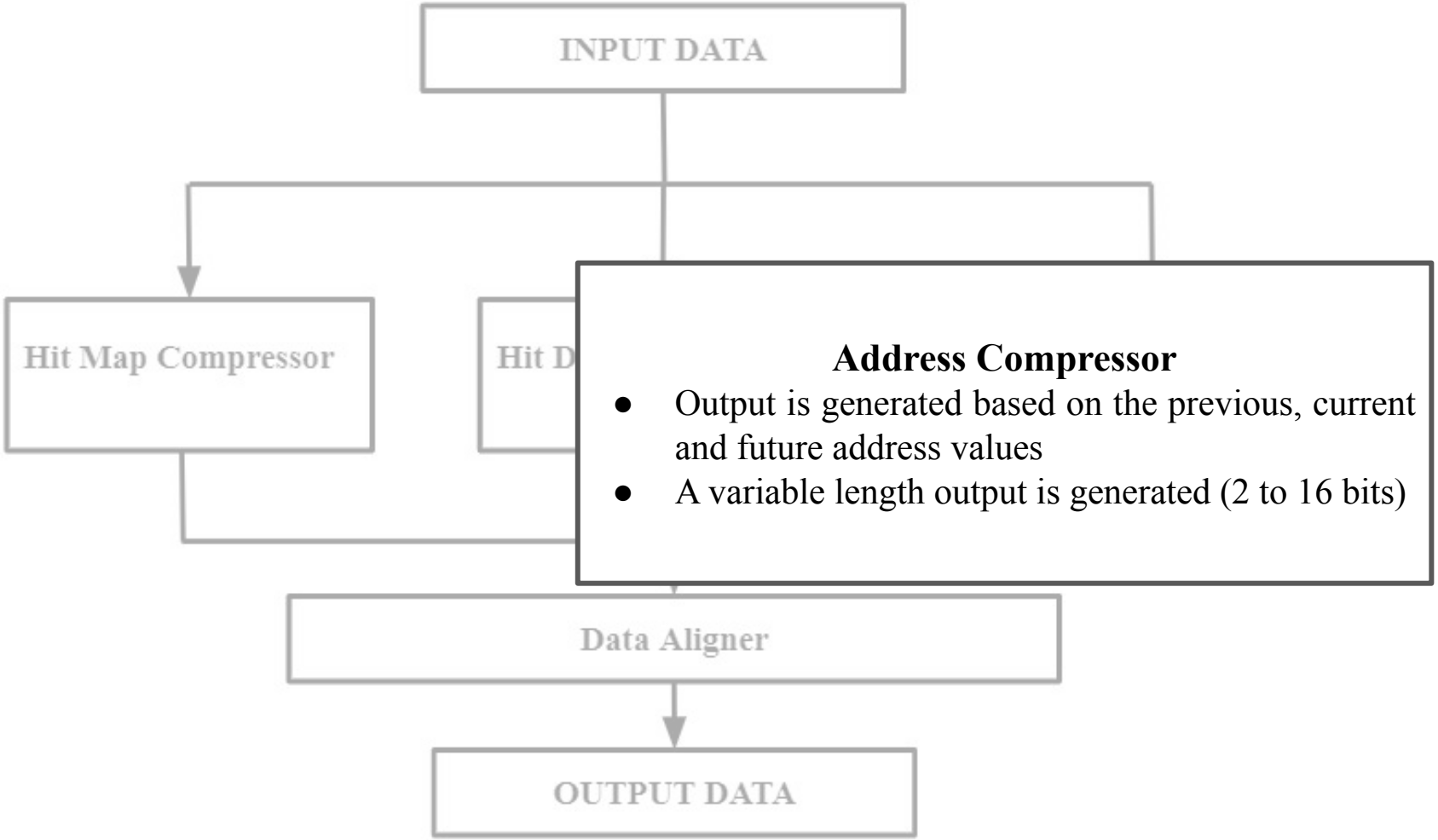


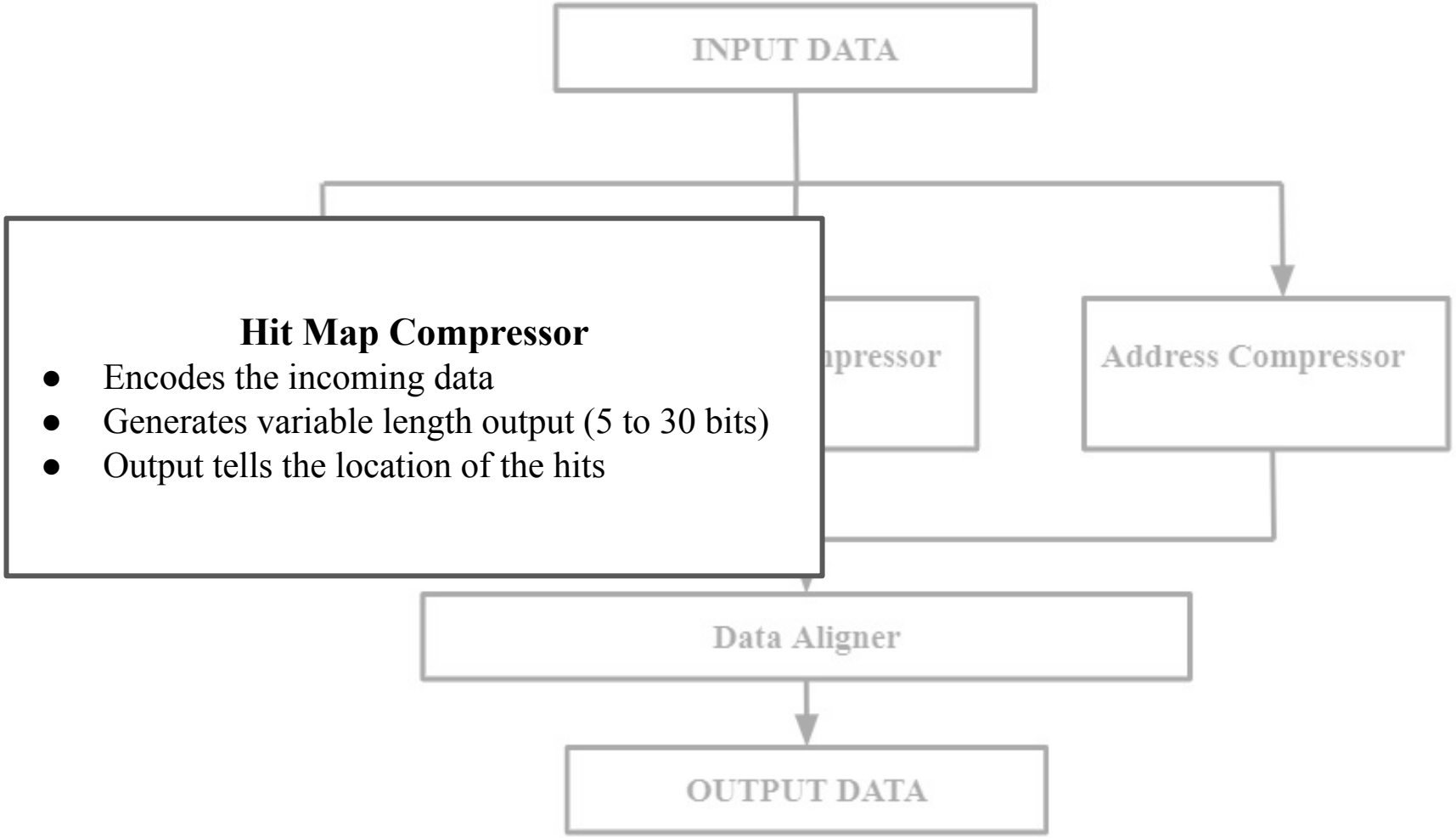


RD53B EMULATOR'S DATA ENCODER




INPUT DATA
Generated by hit data generator
Size: 78 bits
Hit Map Compressor
Hit Data Compressor
Address Compressor
Data Aligner
OUTPUT DATA

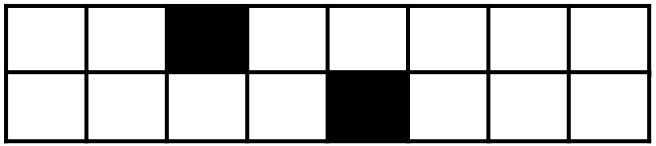

INPUT DATA
Hit Map Compressor
Hit D
Address Compressor
Output is generated based on the previous, current and future address values
A variable length output is generated (2 to 16 bits)
Data Aligner
OUTPUT DATA



Hit Map Compressor

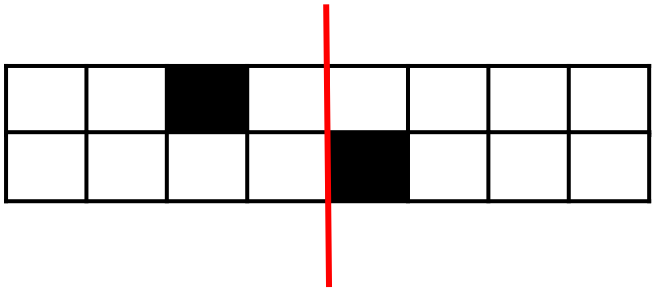
- Encodes the incoming data
- Generates variable length output (5 to 30 bits)
- Output tells the location of the hits

HIT MAP COMPRESSION



HIT MAP COMPRESSION

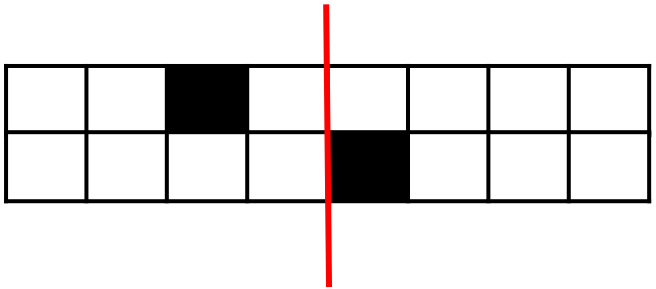
Step1:



1

HIT MAP COMPRESSION

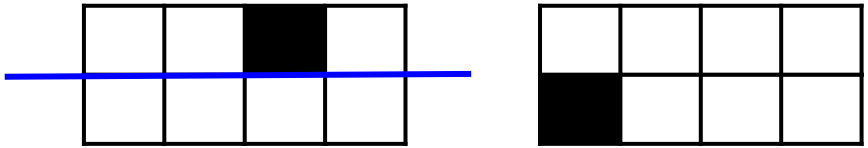
Step1:



11

HIT MAP COMPRESSION

Step2:

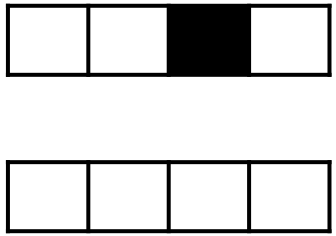


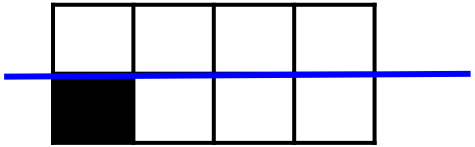
11

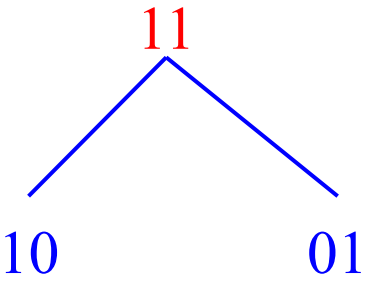
10

HIT MAP COMPRESSION

Step2:

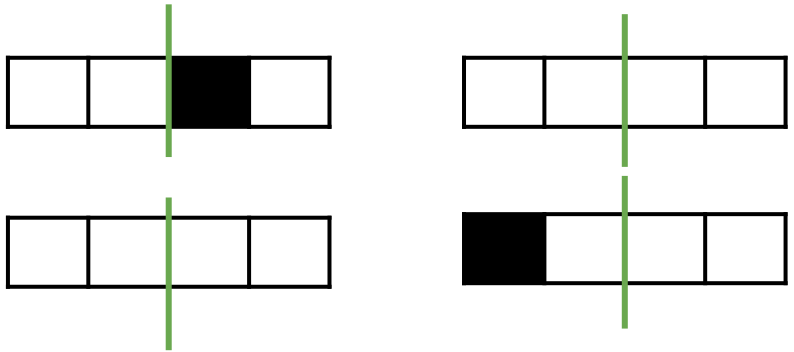


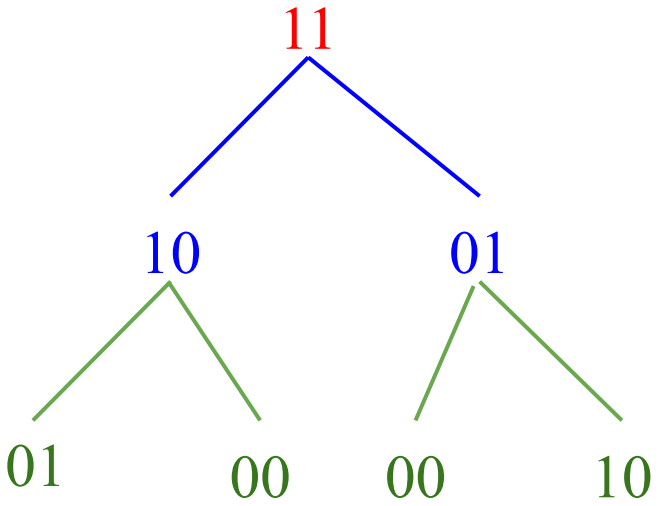




HIT MAP COMPRESSION

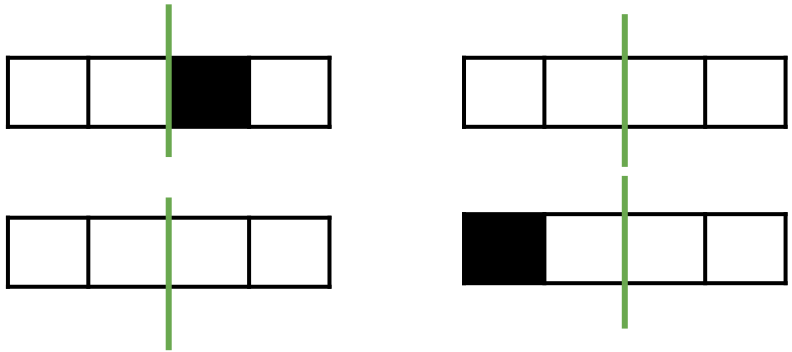
Step3:

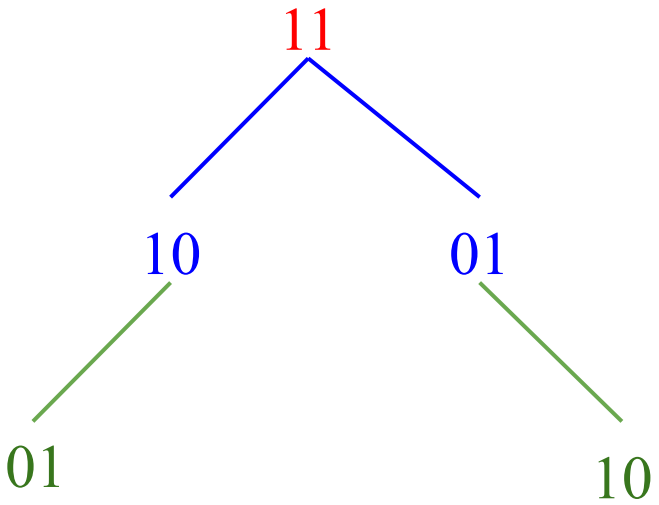




HIT MAP COMPRESSION

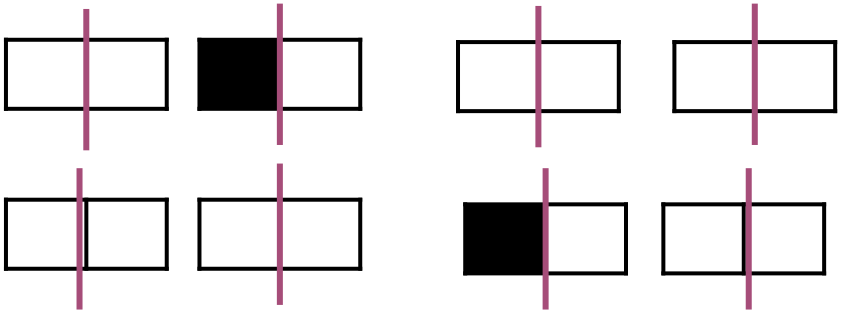
Step3:

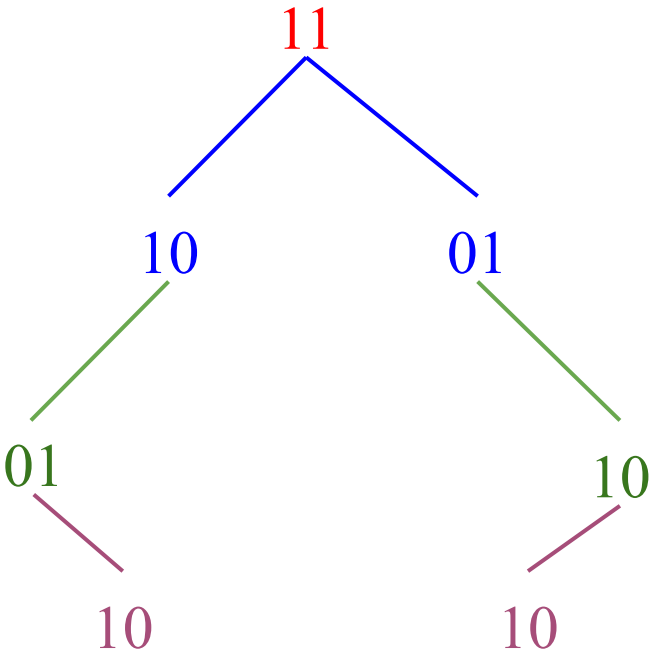




HIT MAP COMPRESSION

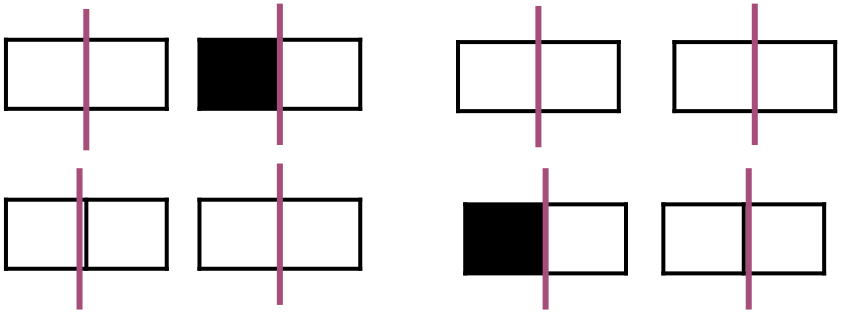
Step4:

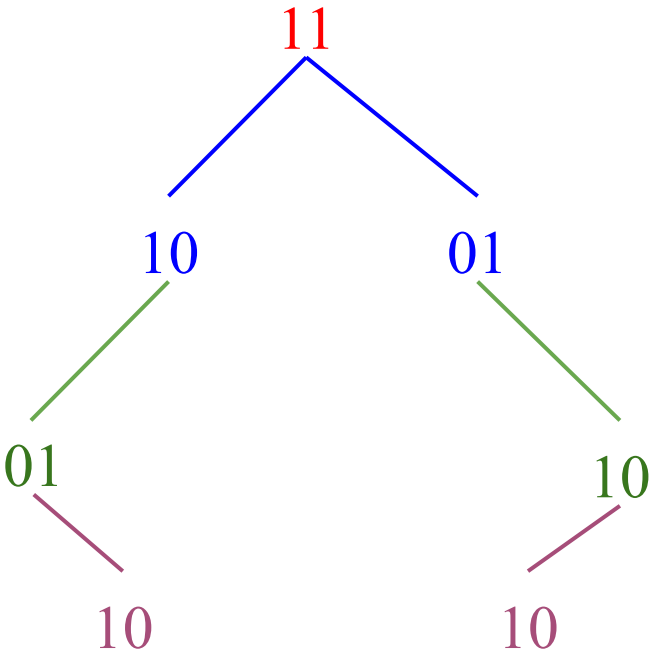




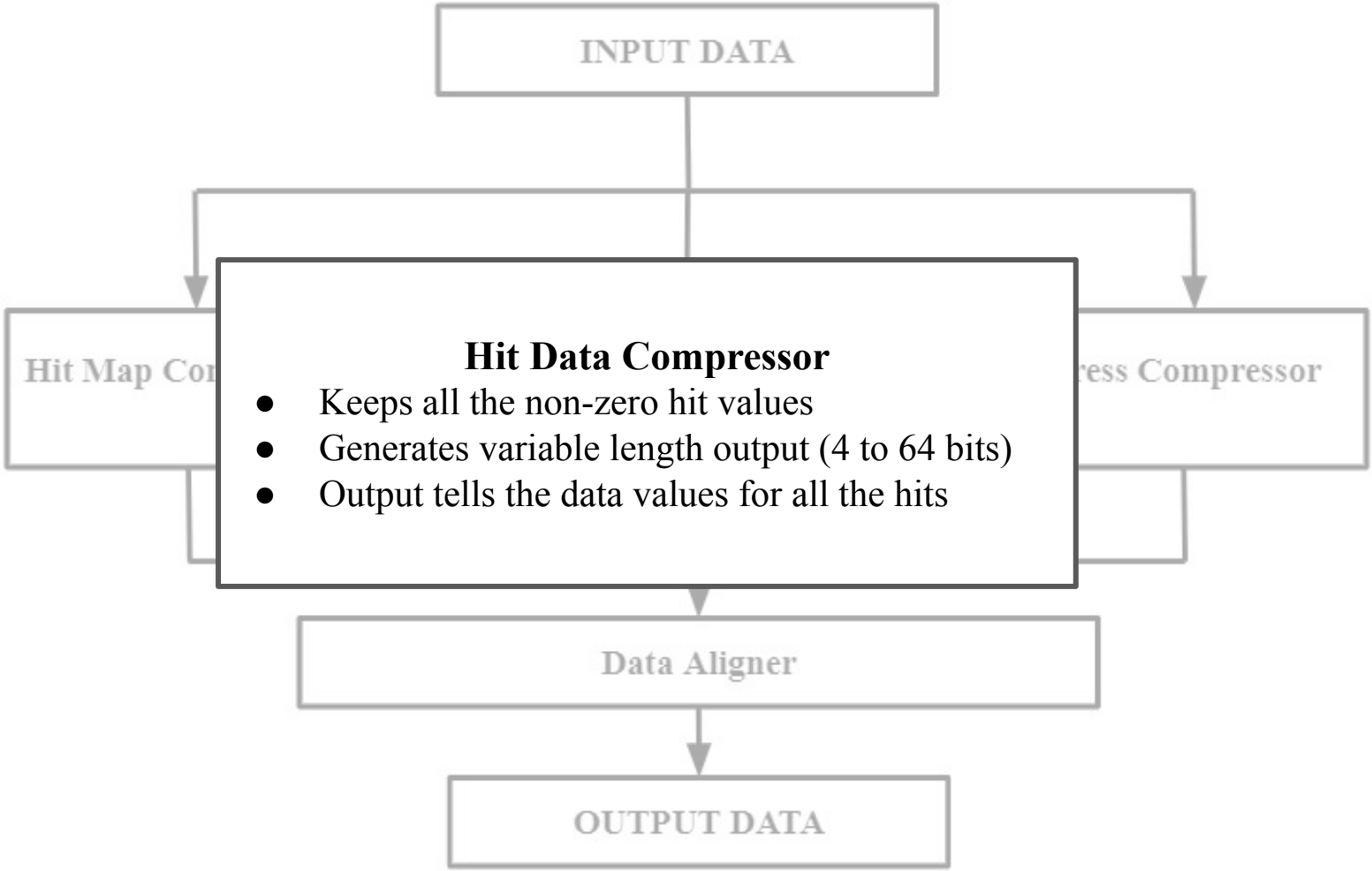
HIT MAP COMPRESSION

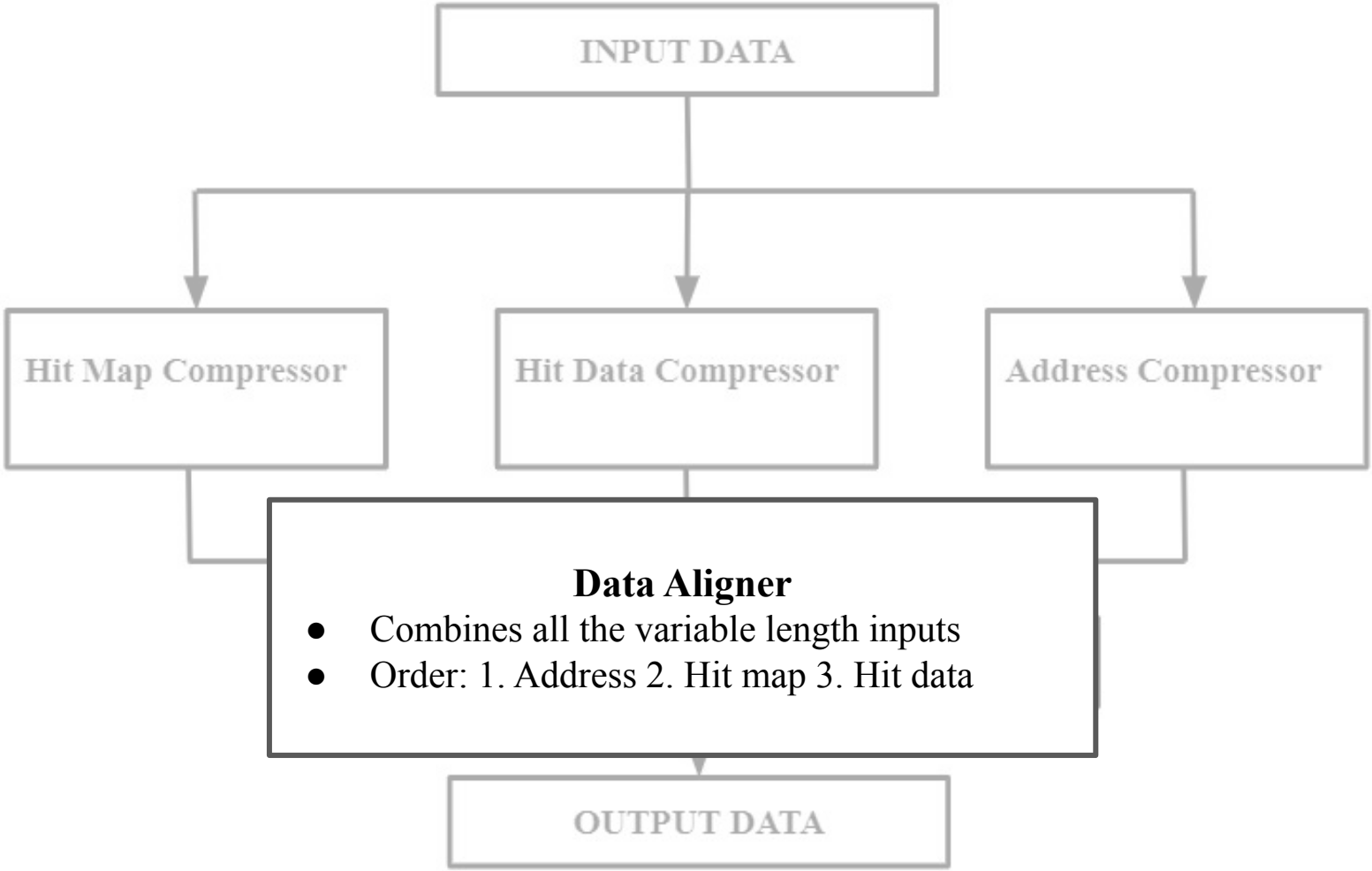
Step4:

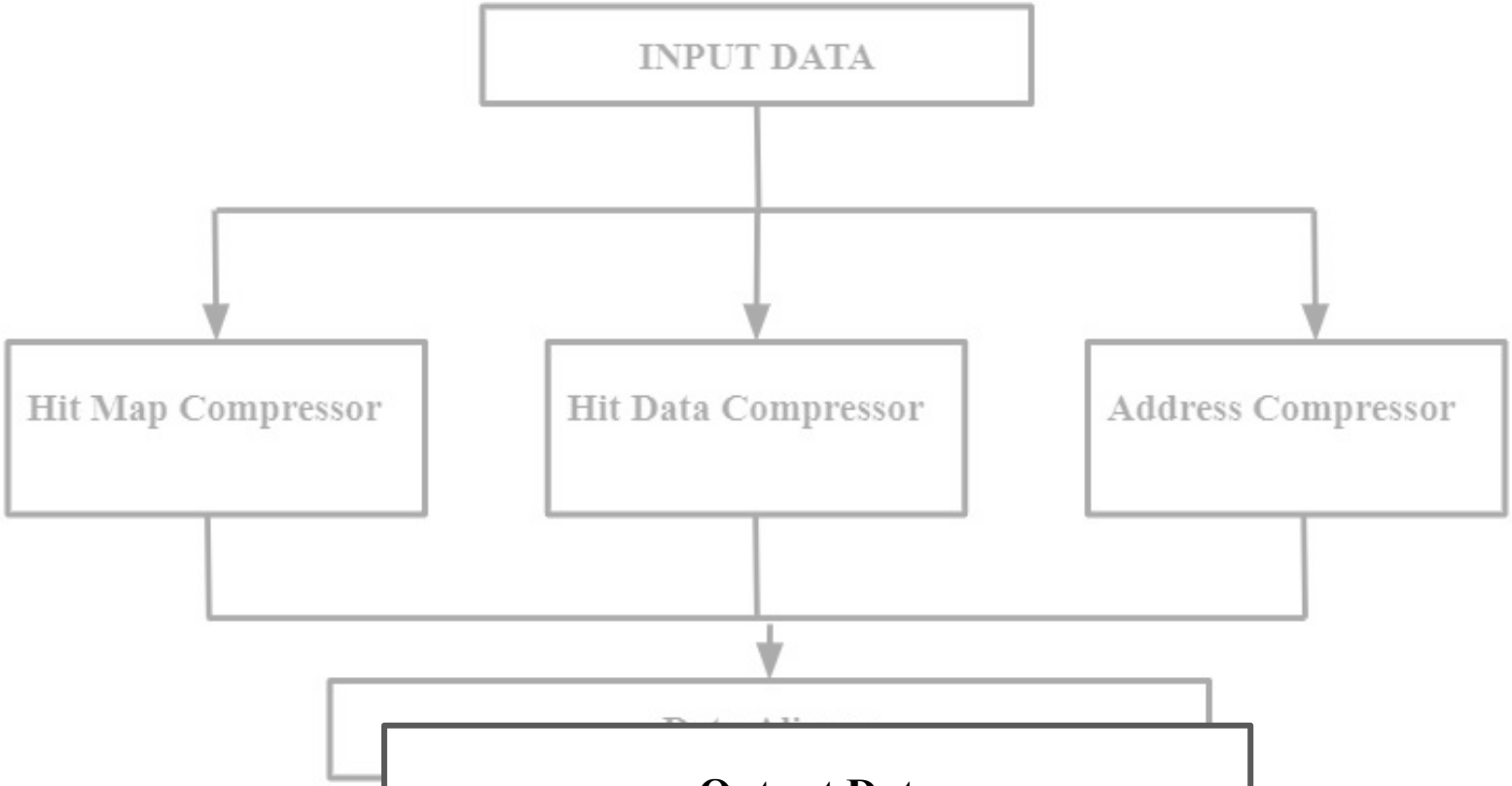




RESULT: 11.1001.0110.1010


INPUT DATA
Hit Map Co
ress Compressor
Hit Data Compressor
Keeps all the non-zero hit values
Generates variable length output (4 to 64 bits)
Output tells the data values for all the hits
Data Aligner
OUTPUT DATA

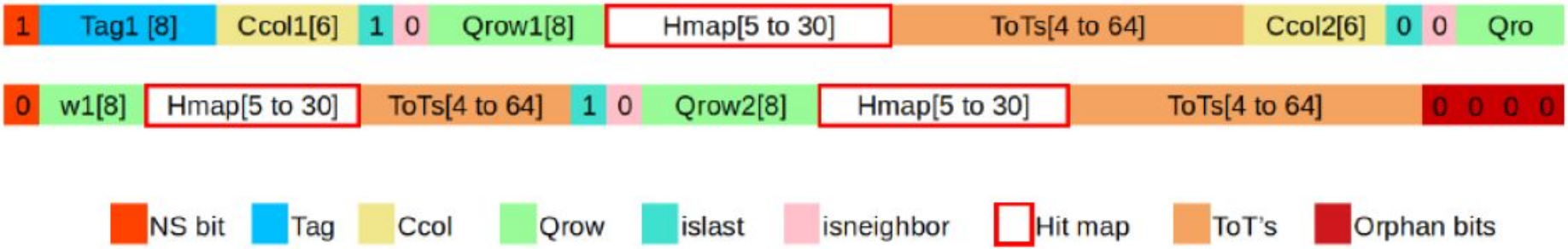

INPUT DATA
Hit Map Compressor
Hit Data Compressor
Address Compressor
Data Aligner
Combines all the variable length inputs
Order: 1. Address 2. Hit map 3. Hit data
OUTPUT DATA



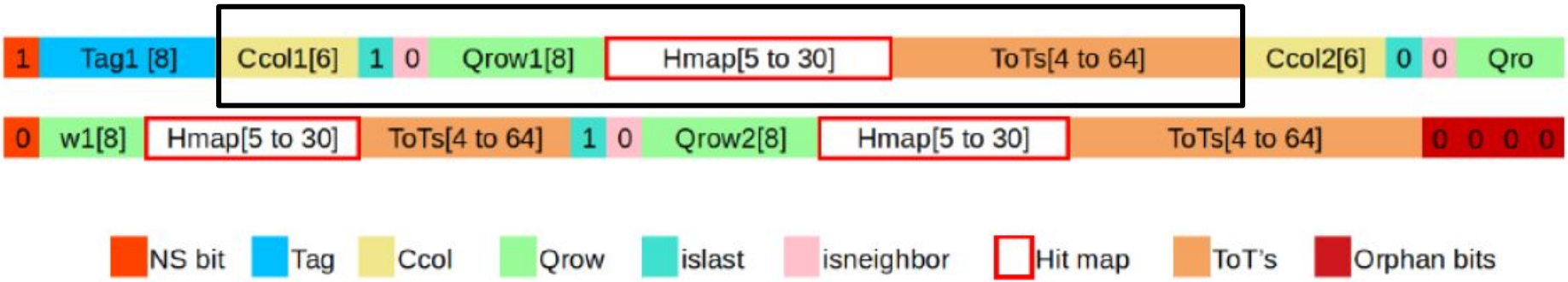
Output Data

- Variable length output (11 to 110 bits)
- Goes to the output data formatter

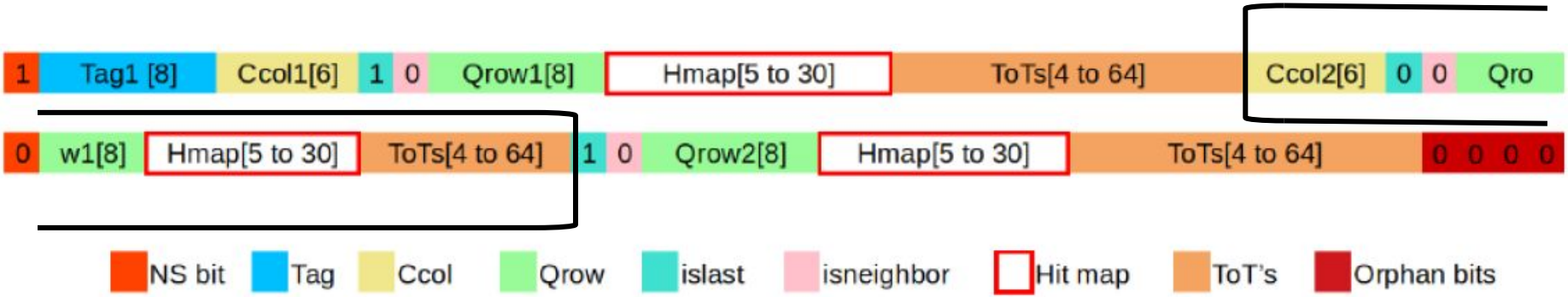
FINAL STREAM OF DATA



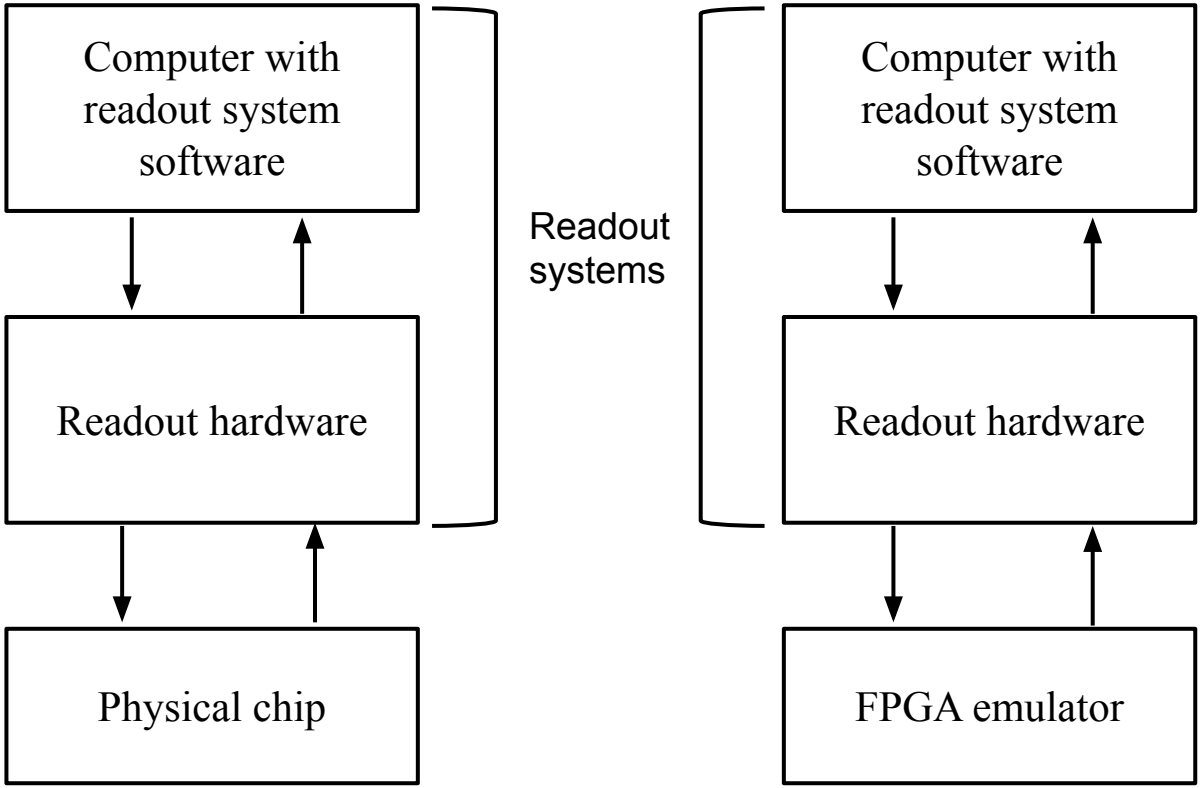
FINAL STREAM OF DATA



FINAL STREAM OF DATA



READOUT SYSTEMS



READOUT SYSTEMS

- YARR (Yet Another Rapid Readout)

Figure 7: YARR setup at UW

READOUT SYSTEMS

- YARR (Yet Another Rapid Readout)
- RCE (Reconfigurable Cluster Element)

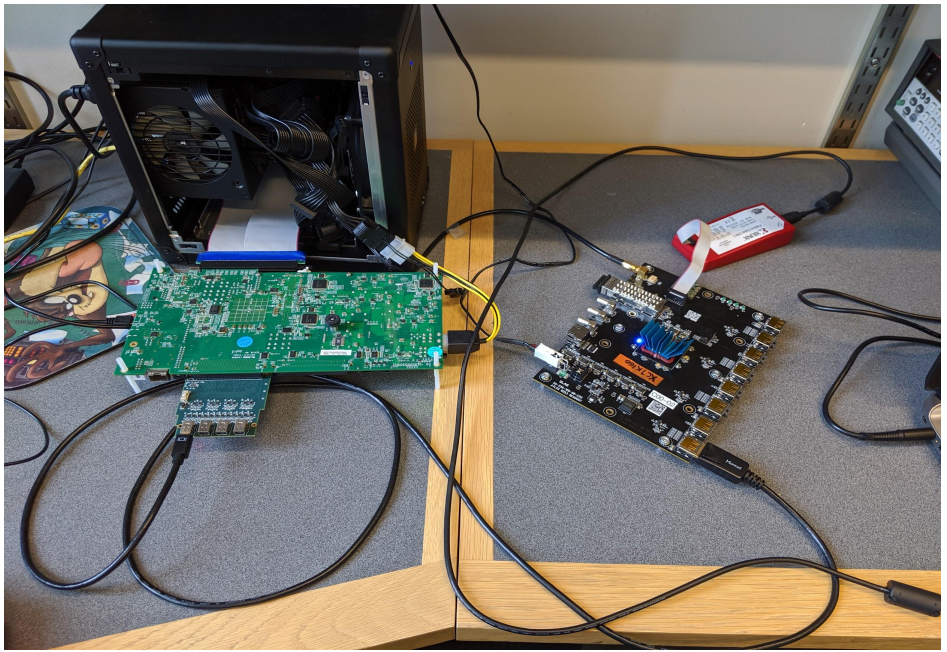


Figure 8: RCE setup at UW

READOUT SYSTEMS

- YARR (Yet Another Rapid Readout)
- RCE (Reconfigurable Cluster Element)
- FELIX (Front End Link eXchange)

Figure 9: FELIX setup at UW

CONCLUSION

- Development of RD53B emulator at UW
- Communication between readout systems and RD53B emulator
- Stayed as close to the RD53B specifications as possible, while developing the FPGA based design

REFERENCES

[1] https://home.cern/ “CERN website”, CERN, [Online]

[2] https://atlas.cern/discover/detector “CERN atlas website”, CERN, [Online]

[3] “Image Source”, [Online]

[4] The ATLAS Collaboration, “The RD53A integrated circuit”, Memo. CERN-RD53-PUB-17-001, January 30, 2017

[5] “The RD53B Pixel Readout Chip Manual”, Version 0.38, April 14, 2020

[6] RD53A Emulator: https://gitlab.com/smithd57/rd53a_hardware_emulator_dev

Thank you Scott and Shih-Chieh
Thank you ACME and LBNL team

IF FURTHER EXPLANATION NEEDED

RD53B EMULATOR FEATURES

- Input data is decoded properly using the custom RD53B protocol
- Output data is properly encoded using the aurora 64/66 protocol at 640MHz
- Reading and writing of global registers.
- Hit data is encoded as per the RD53B specifications
- Trigger commands cause the output of sets of encoded hit data

RD53B EMULATOR NON-FEATURES

- Since this is a digital design project any RD53B features that involve analog circuitry such as the pixels have been greatly simplified or removed entirely

- Commands such as ‘Clear’ and ‘Cal’ are received by the FPGA but have no effect

- The input clock is not derived from the input stream using clock data recovery but needs to be provided to the system as a separate signal

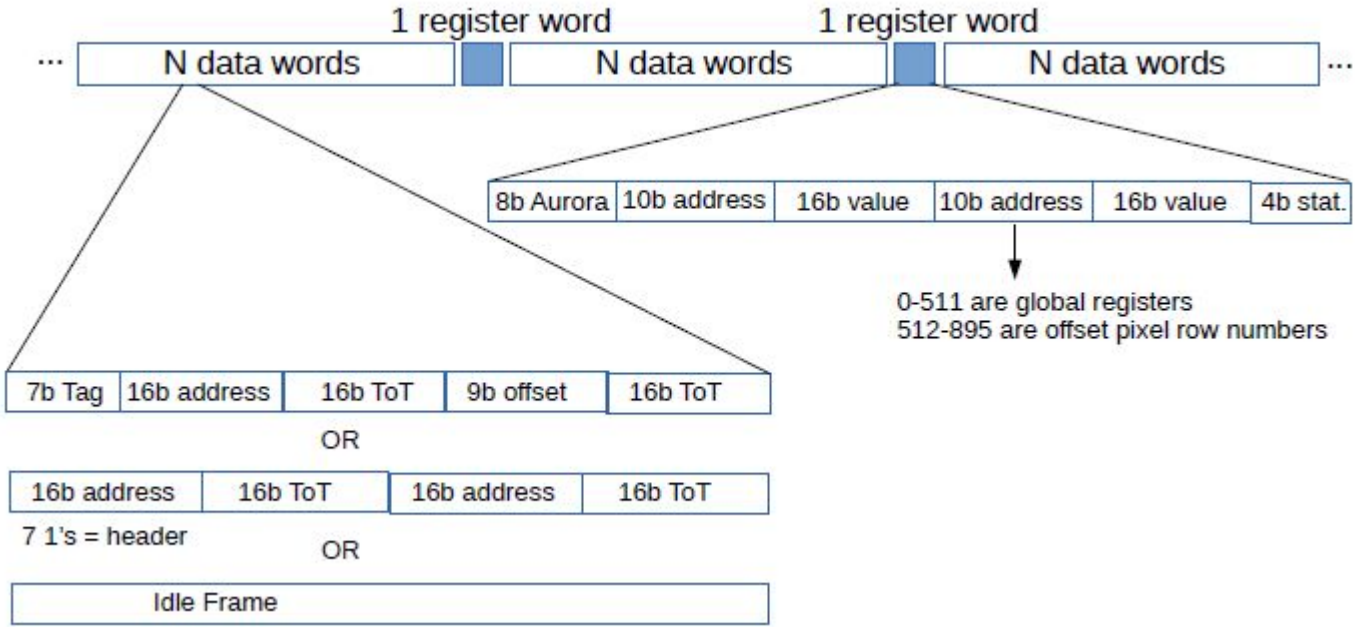
RD53A COMMAND AND THEIR FORMATTING

Command	Encoding	ID/(A)ddress/(D)ata 5-bit Fields					
ECR	2× 0101_1010						
BCR	2× 0101_1001						
Glob. Pulse	2× 0101_1100	ID<3:0>,0	D<4:0>				
Cal	2× 0110_0011	ID<3:0>,D15	D<14:10>	D<9:5>	D<4:0>		
WrReg	2× 0110_0110	ID<3:0>,0	A<8:4>	A<3:0>,D<15>	D<14:10>	D<9:5>	D<4:0>
WrReg	2× 0110_0110	ID<3:0>,1	A<8:4>	A<3:0>,D<15>	D<14:10>	9×(D<9:5>	D<4:0>)
RdReg	2× 0110_0101	ID<3:0>,0	A<8:4>	A<3:0>,0	00000		
Noop	2× 0110_1001						
Sync	1000_0001_0111_1110						

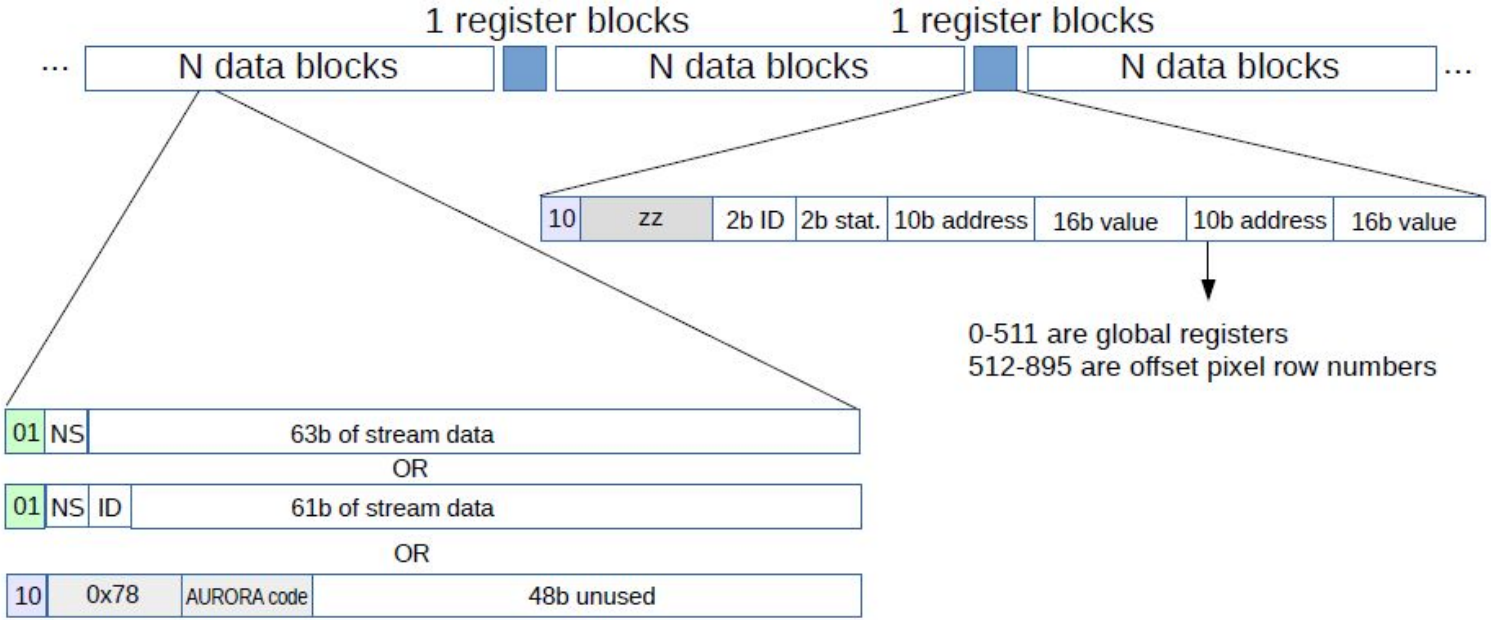
RD53B COMMAND AND THEIR FORMATTING

Command	Encoding		(T)ag, (A)ddress or (D)ata 5-bit content					
Sync	1000_0001	0111_1110						
PLLlock	1010_1010	1010_1010						
Trigger	tttt_tttt	Tag[0..53]						
Read_trigger	0110_1001	ID<4:0>	00,T<7:5>	T<4:0>				
Clear	0101_1010	ID<4:0>						
Global Pulse	0101_1100	ID<4:0>						
Cal	0110_0011	ID<4:0>	D<19:15>	D<14:10>	D<9:5>	D<4:0>		
WrReg(0)	0110_0110	ID<4:0>	0,A<8:5>	A<4:0>	D<15:11>	D<10:6>	D<5:1>	D<0>,0000
WrReg(1)	0110_0110	ID<4:0>	1,xxxx	xxxxx	N×(D<9:5>	D<4:0>)		
RdReg	0110_0101	ID<4:0>	0,A<8:5>	A<4:0>				

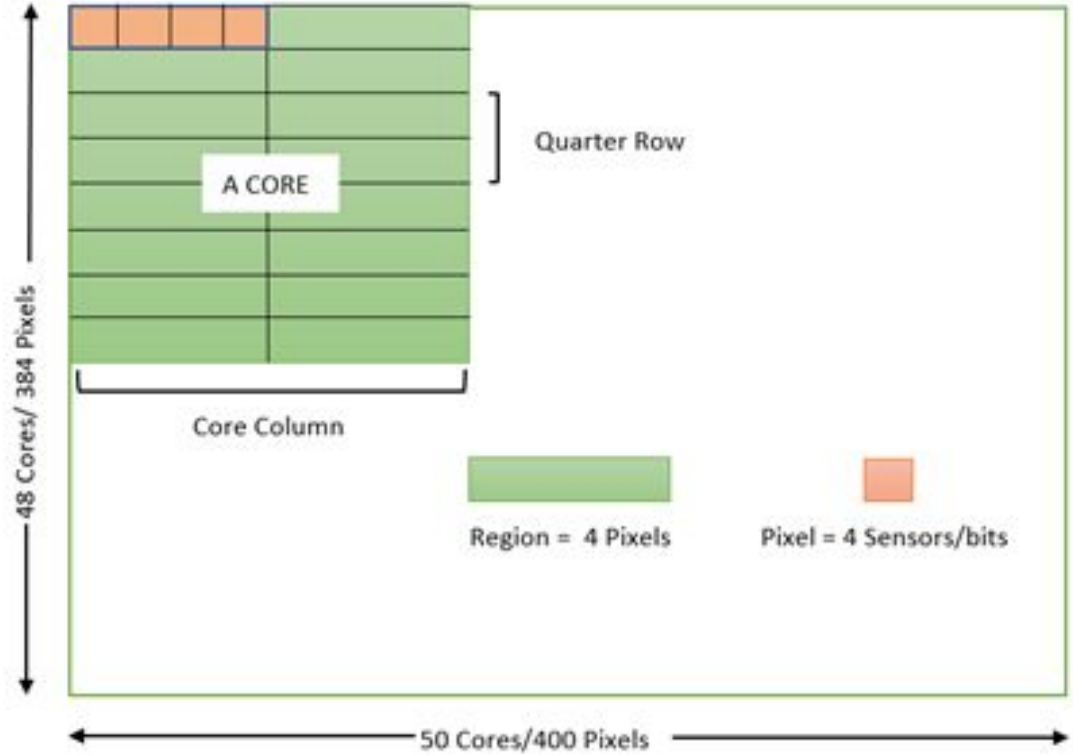
RD53A OUTPUT DATA FORMATTING



RD53B OUTPUT DATA FORMATTING



ADDRESSING - RD53A



ADDRESS COMPRESSOR

$$\text{Islast bit} = \begin{cases} 1 & ; \text{ current column address } != \text{ future column address} \\ 0 & ; \text{ current column address } == \text{ future column address} \end{cases}$$

ADDRESS COMPRESSOR

$$\text{Islast bit} = \begin{cases} 1 & ; \text{ current column address != future column address} \\ 0 & ; \text{ current column address == future column address} \end{cases}$$

$$\text{Isneighbor bit} = \begin{cases} 1 & ; \text{ current row address = previous row address + 1} \\ 0 & ; \text{ otherwise} \end{cases}$$

DATA ALIGNER

Address = **11001**0000000000000 Address_Length = **5**
HitMap = **10011**1000000000000 HitMap_Length = **6**
HitData = **11101**0000000000000 HitData_Length = **5**

Shifted_HitMap = HitMap >> Address_Length
= 00000**10011**10000000

Shifted_HitData = HitData >> (Address_Length + HitMap_Length)
= 0000000000**11101**0

Result = Address + Shifted_HitMap + Shifted_HitData
= **11001.100111.11101.**0