Lossless data compression for the HL-LHC pixel detector readout

Stamatios Poulis 1,2,* Konstantin Androsov 1,2, Roberto Beccherle 1, Massimo Minuti 1, Fabrizio Palla 1

1. INFN Pisa, 2. University of Siena
*Supported by the EU FP7-PEOPLE-2012-ITN project nr 317446, INFIERI, “Intelligent Fast Interconnected and Efficient Devices for Frontier Exploitation in Research and Industry”
CMS pixel detector

- Close to the interaction point
- High flux of particles
- High amount of data
- Tight space
CMS tracker

- 4 Barrel layers
- 10 Forward disks
**HL-LHC pixel detector readout chip**

- Hit rate estimation for inner barrel layer at 140 Pile-Up is about 3 GHz/cm²
- Expected high readout rate (~ 4.8 Gbits/s per chip for 1 MHz trigger rate)
- 2.4 Gbits/s max bandwidth per chip (2 E-links)
Goal

- An efficient transfer protocol should be developed to collect all measurements and ensure good detector performance
- Lossless compression with decreased output rate to reduce the usage of links
- On-chip compression – RD53 collaboration
Readout format

- Using readout format proposed in the draft of Phase 2 pixel system and read-out chip

Event header 32-bit ➔ Event trailer 32-bit ➔ Hit data …

- BX-id, event ID, chip address, status
- Event size, check sum, status/error flag,…

- Default hit data representation:
  - (30 bits/pixel)

Pixel 1
24 bit Address
5 bit ADC
1 bit flag ➔ Pixel 2 ➔ … ➔ Pixel N
Compression scheme

- **Main domains**
  - **Data representation**
    - How the data are accessed
  - **Encoding**
    - Actual data encoding

![Diagram showing the process from Pixel array to Encoding method through Data representation.](Image)
Characteristic Distributions

Pixel X

![Graph of Pixel X distribution]

Pixel Y

![Graph of Pixel Y distribution]

Active ADC

![Graph of Active ADC]

CTD/WIT17, Orsay

6/3/2017
Representations

• Group pixels in 2x2 pixel regions
  ◦ Format: \{PRaddr, ADC0, ADC1, ADC2, ADC3,\}
  ◦ Low compression
  ◦ Very simple

• Delta representation
  ◦ Calculate \( \Delta x \) and \( \Delta y \) for every active pixel
  ◦ \( \Delta x = x_n - x_{n-1}, \Delta y = y_n - y_{n-1} \)
  ◦ Good compression
  ◦ Requires additional logic
Delta representation
(ordering by rows)

(x1, adc1)
(x2, adc2)

(x1, y0, adc1)
(x2, y0, adc2)
(x2, y2, adc4)
(x3, y2, adc3)

Δx encoder
Δy encoder
ADC encoder

Package data

FIFO col 0
FIFO col 1
FIFO col 2
FIFO col 3

FIFO row x1
FIFO row x3
FIFO row x2
Encoder

- Simple compression technique
  - Reduced logic
  - Reduced need for power/area
  - Minimum computation time

- Huffman
  - Simple
  - Fast
Huffman coding

- Prefix coding
- Assigns shorter codes for more probable symbols
Data flow

- **Timing requirements**
  - 160 clk cycles available
- **Huffman processes 1 symbol per clk**
- **Pixel array divided in regions**
- **Encode in parallel**
- **Merge data in packager**
Compressor architecture

- **N Encoding Units for N regions**
  - encoder_adc
  - encoder_dx
  - encoder_dy
- **Packager**
  - Final packet
- **Organizer**
  - Organizes the data into 32-bit words
Compression performance

As a function of number of macro-regions

![Graph showing compression performance as a function of number of macro-regions. The graph includes lines for combined delta (by row ordering), combined delta (by cluster ordering), combined delta (by column ordering), 2x2 regions (ToT are not encoded), and 2x2 regions (ToT are encoded). The x-axis represents the number of macro regions, and the y-axis represents the average data rate reduction. The graph also notes that a 32 letters alphabet is used for coordinates.](image-url)
Compression performance

Number of regions = 4, Dictionary size = 32

- default (no compression)
- combined delta (by row ordering)
- combined delta (by cluster ordering)
- combined delta (by column ordering)
- 2x2 regions (ToT are not encoded)
- 2x2 regions (ToT are encoded)
Synthesis results 65nm

- Compressor unit
  - Power \(\sim 1.93 \text{ mW} \text{ (@160 MHz)}\)
  - Area \(\sim 11000 \text{ um}^2\)

<table>
<thead>
<tr>
<th>Instance</th>
<th>Cells</th>
<th>Leakage Power (nW)</th>
<th>Dynamic Power (nW)</th>
<th>Total Power (nW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top_compressor</td>
<td>4410</td>
<td>3892.184</td>
<td>1933083.865</td>
<td>1936976.049</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Instance</th>
<th>Cells</th>
<th>Cell Area</th>
<th>Net Area</th>
<th>Total Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top compressor</td>
<td>4410</td>
<td>10926</td>
<td>0</td>
<td>10926</td>
</tr>
</tbody>
</table>
Conclusions

- Compression ratio up to ~2.9 with Huffman coding depending on configuration
- Combine pixels in 2x2 regions for a simpler solution/implementation
  - ~1.2 without compressing ToT
  - ~1.5 with compressed ToT
- Optimize compression for each region (Barrel, Forward disks) to increase performance
Thank you!!!