Algorithms for Threshold Dispersion Minimization of the CHIPIX65 Asynchronous Front-End

Mauro Sonzogni[*]
University of Bergamo and INFN-Pavia
E-mail: mauro.sonzogni@unibg.it

F. De Canio, L. Gaioni, M. Manghisoni, V. Re, G. Traversi

University of Bergamo and INFN-Pavia

L. Ratti

University of Pavia and INFN-Pavia

This work discusses four different algorithms for the minimization of threshold dispersion in multichannel readout circuits for pixel detectors. These algorithms, which are based on different methods (e.g. charge scans, threshold scans, etc) and differs in terms of performance and computation time, have been tested on the asynchronous front-end integrated in the CHIPIX65_FE0, a readout ASIC prototype designed in a 65 nm CMOS technology.

Topical Workshop on Electronics for Particle Physics
16 - 21 September 2018
Antwerp, Belgium

[*]Speaker.

1. HL-LHC and RD53 Requirements

Extraordinarily high levels of radiation and particle rates will be attained in the high-luminosity upgrades of the Large Hadron Collider experiments. The instantaneous luminosity will reach $5 \times 10^{34}\,\mathrm{cm}^{-2}\mathrm{s}^{-1}$, while the expected level of ionizing radiation for the innermost layers of the tracking pixel detectors is 1 Grad in 10 years. The CERN RD53 collaboration [1] was founded in 2013 to investigate new technologies and architectures for future readout chips for the innermost detector layers of ATLAS and CMS. The Italian Institute for Nuclear Physics (INFN), through the CHIPIX65 Project [2], has been actively contributing to RD53 by testing new design solutions with a small-scale demonstrator called CHIPIX65_FE0. One of the main requirements of the new HL-LHC readout chip is to guarantee sub-1000 electrons stable threshold operation. In order to satisfy this requirement, minimization of threshold dispersion is fundamental. In this work we present four threshold tuning algorithms and discuss their performance in terms of threshold dispersion and operation time together with some of the characterization results of the CHIPIX65 asynchronous front-end.

2. CHIPIX Asynchronous Analog Front-end

The CHIPIX65 asynchronous front-end [3], shown in figure 1, is designed according to RD53A requirements, such as: capability of operating at a minimum threshold lower than $1000\,e^-$, power consumption not exceeding $5\,\mu W$ in a silicon area close to $1000\,\mu m^2$ and an equivalent noise charge (ENC) lower than $120\,e^-$. It is manly composed by:

- charge sensitive amplifier with a gain stage featuring a folded cascode architecture, with two local feedback networks designed to increase the output small signal resistance;
- Krummenacher feedback, providing a constant current discharge of the capacitors $C_{F1,2}$;
- high speed, low power current comparator;
- in-pixel threshold trimming DAC, based on a 4-bit binary weighted architecture, generating the current I_{DAC} which adds to the current I_{th} (equal for each pixel in the matrix) in order to obtain the same threshold in all matrix pixels (threshold dispersion minimization);

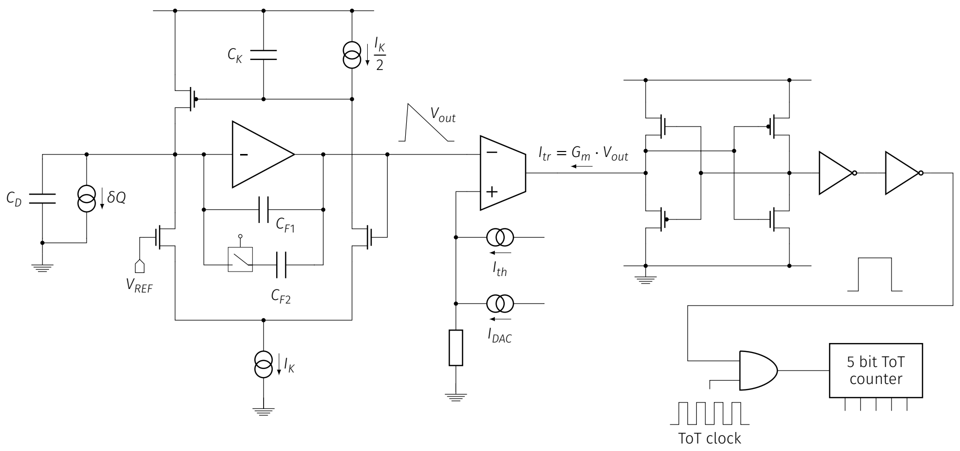
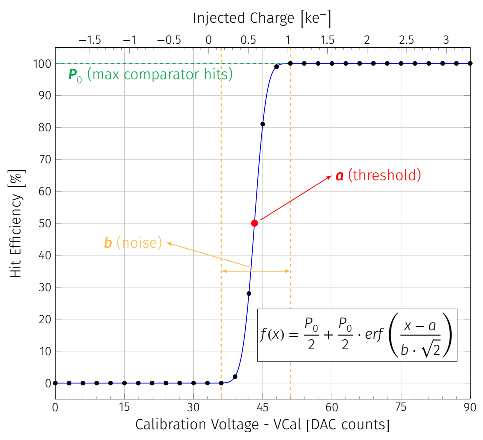
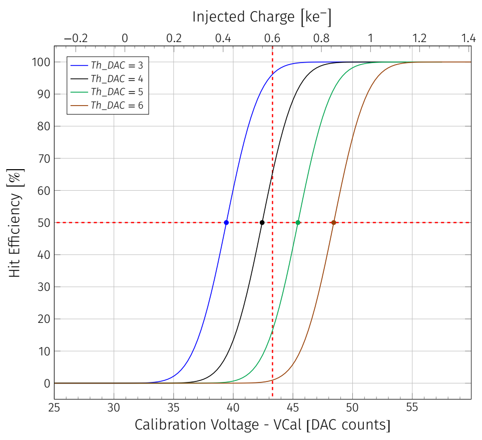


Figure 1: CHIPIX65 Asynchronous Analog Front-end. schematic



(a) Example of an s-curve obtained by a complete charge scan.



(b) Algorithm 1 - Charge scans obtained with four different DAC codes.

Figure 2: Charge Scan, Algorithm 1.

- relatively slow time-over-threshold (ToT) clock (40 MHz) and 5 bit, dual edge ToT counter.

3. Algorithms for threshold dispersion minimization

In this section, four algorithms for threshold minimization are described. All matrix pixels must have a threshold as close as possible to a threshold selected by the user. In figure 2a, the error function curve used to fit hit efficiency data is reported: `a` represents the pixel threshold, P_0 is the maximum number of comparator hits and `b` represents pixel noise. The variables used in the algorithms are described below:

- `nb`: Number of DAC bits
- `np`: Number of pixels
- `sth`: Desired pixels threshold, selected by the user
- `ncomb`: Number of DAC combinations (2^{nb})
- `th_dac`: Array of DAC combinations to obtain `sth`. Each element of the array corresponds to one pixel.
- `nscan`: Number of different charge injections

3.1 Algorithm 1 - Complete Charge Scan

Given one pixel, this algorithm executes a complete charge scan for each DAC code. Injected charges ranges from 0 e^- to 1500 e^-. Data obtained from these scans, must be fitted with the function shown in figure 2a in order to obtain the threshold for all pixels and DAC code combinations. For each pixel, the DAC code which sets the threshold as close as possible to `sth` is selected and saved in the array `th_dac`. An example of this algorithm can be seen in figure 2b: four scans with four different DAC codes are shown, the one with a threshold closer to 600 e^- is selected (in this case DAC code = 4).

An optimal threshold dispersion minimization can be obtained using this algorithm, furthermore faulty pixels can be easily detected. A pixel is considered correctly functioning if the s-curves parameters respect the following conditions: $0.95 \cdot \mathtt{n} \leq P_0 \leq \mathtt{n}$ (where $\mathtt{n}$ is the number of times that each charge is injected) and $\mathtt{b} \leq 120\,\mathrm{e}^-$. The main disadvantage is that it is computationally slow compared to the algorithms described in the following subsections.

The algorithm complexity is: $O(\mathtt{np} \cdot \mathtt{ncomb} \cdot \mathtt{nscan})$, $\Omega(\mathtt{np} \cdot \mathtt{ncomb} \cdot \mathtt{nscan})$.

3.2 Algorithm 2 - Fixed Charge Scan

For each DAC code, a fixed charge equivalent to the threshold `sth`, is injected in each pixel `n` times. The comparator . The DAC code which makes possible to obtain a hit occupancy closer to 50 % is selected for that pixel. An example is shown in figure 3a: in this case DAC code 9 has been selected.

This algorithm is computationally faster than the one described in subsection 3.1, moreover threshold dispersion minimization is optimal and faulty pixels can be easily detected.

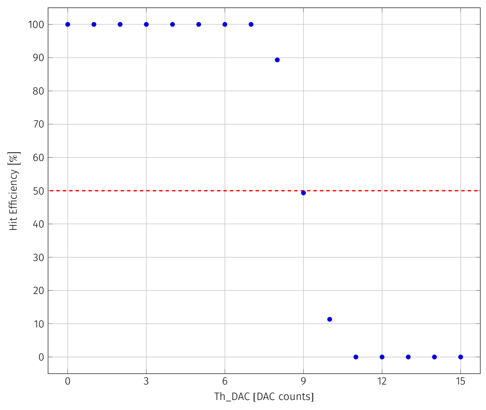
The algorithm complexity is: $O(\mathtt{np} \cdot \mathtt{ncomb})$, $\Omega(\mathtt{np} \cdot \mathtt{ncomb})$

3.3 Algorithm 3 - Root to Leaf Binary Tree Scan

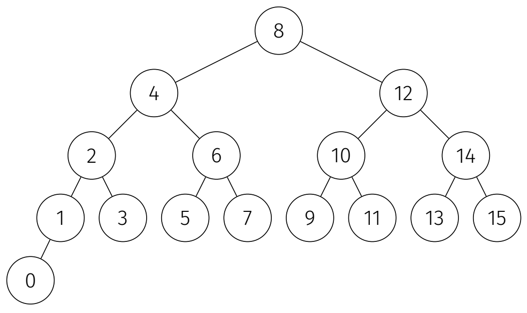
This algorithm is based on the binary search tree shown in figure 3b. Starting from a DAC code equal to $\mathtt{ncomb}/2$, a fixed charge `sth` is injected in each pixel `n` times. At each step, the number of comparator hits is stored as an array element: this number defines the direction of the next branch, either right ($\geq 50\%$) or left ($< 50\%$). Finally, the DAC code associated to the hit occupancy closest to 50 % is selected and saved in the array `th_dac`.

This algorithm is faster than those described in subsections 3.1 and 3.2. Optimal pixel trimming can be obtained only when all pixels work correctly; in fact s-curves are not produced and, consequently, faulty pixels cannot be easily detected.

The algorithm complexity is: $O(\mathtt{np} \cdot (\lfloor \log_2(\mathtt{ncomb}) \rfloor + 1))$, $\Omega(\mathtt{np} \cdot \lfloor \log_2(\mathtt{ncomb}) \rfloor)$



(a) Algorithm 2 - Hit efficiency obtained with all 16 DAC codes for one pixel



(b) Algorithm 3 and 4 - Binary Search Tree.

Figure 3: Algorithm 2 scan, Algorithm 3 and 4 Binary Search Tree.

3.4 Algorithm 4 - Bounded Binary Tree Scan

This algorithm is based on a binary search tree like the one described in subsection 3.3. The user defines two values: `smin` and `smax`. The search starts from a DAC code equivalent to

Threshold dispersion [e^-]	Alg 1	Alg 2	Alg 3	Alg 4
Chip 1	54.1	54.41	54.96	N/A
Chip 2	N/A	N/A	65.79	67.99
Irradiated Chip	81.1	N/A	N/A	73.56[1]

Table 1: Threshold dispersion for 3 different chips

`ncomb`/2, then a fixed charge `sth` is injected in each pixel `n` times. At each step, if the comparator hits are more than `smax` %, the DAC code has to be greater (right in the binary tree), if they are less than `smin` %, the code has to be smaller (left in the binary tree), whereas if they are between `smin` % and `smax` %, that DAC code is selected, saved in the array `th_dac` and the scan of next pixel starts.

With this algorithm, different pairs of `smin` and `smax` have to be tested in order to find the one which minimizes the threshold dispersion. If the range is too wide, tree leafs will never be reached, whereas if it is too narrow the algorithm will stop in the internal nodes. This leads to a non-optimal threshold dispersion minimization.

The algorithm complexity is: $O(\mathtt{np} \cdot \lfloor \log_2(\mathtt{ncomb}) \rfloor)$, $\Omega(\mathtt{np})$

3.5 Results and Conclusions

Different measurements, mainly concerned with threshold dispersion minimization, have been performed on three chips. One of these has been tested after exposure to total ionizing doses (TID) up to 630 Mrad (SiO_2). Threshold dispersion at 600 e^-, obtained with the four tuning algorithms discussed in this paper, is reported in the table 1. In the chip 1 case, an optimal threshold dispersion of 54 e^- has been obtained with a mean threshold of 590 e^-, starting from an initial threshold dispersion of 450 e^-.

Algorithms 1 and 4 are not recommended; the first is computationally too slow, whereas threshold dispersion minimization with algorithm 4 could be not optimal. Algorithm 2 can detect faulty pixels but is slower than 3, which, instead, could give optimal results in a shorter time if all pixels work correctly. Moreover, test results confirm that the threshold dispersion performance of the CHIPIX65 continuous-time analog front-end is compatible with the specifications set by the RD53 consortium, in view of the development of 65 nm CMOS pixel readout chips for the innermost detector layers of ATLAS and CMS.

References

[1] "RD53 Collaboration Web Site."

[2] N. Demaria et al., *CHIPIX65: Developments on a new generation pixel readout ASIC in CMOS 65 nm for HEP experiments*, in *Proceedings - 2015 6th IEEE International Workshop on Advances in Sensors and Interfaces, IWASI*, 2015.

[3] L. Ratti et al., *A Front-End Channel in 65 nm CMOS for Pixel Detectors at the HL-LHC Experiment Upgrades*, *IEEE Transactions on Nuclear Science* **64** (2017) 789.

[1]Algorithm 4 tests have been carried out two months after irradiation with a final dose of 630 Mrad