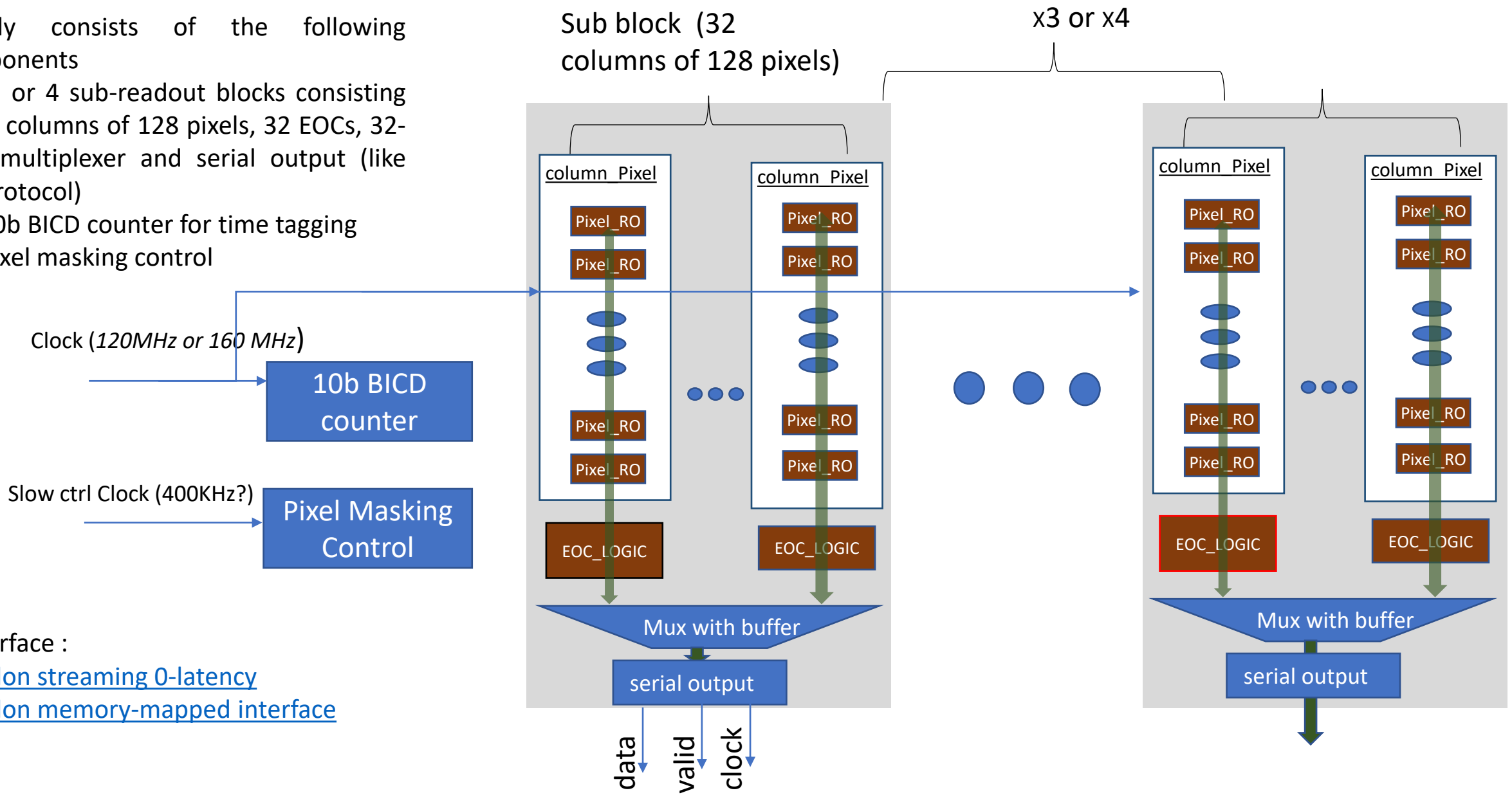
Readout top (Readout_Top.sv)

Mainly consists of the following components

=> 3 or 4 sub-readout blocks consisting of 32 columns of 128 pixels, 32 EOCs, 32-to-1 multiplexer and serial output (like SPI protocol)

⇒ 10b BICD counter for time tagging

⇒ Pixel masking control



Interface :

[Avalon streaming 0-latency]

[Avalon memory-mapped interface]

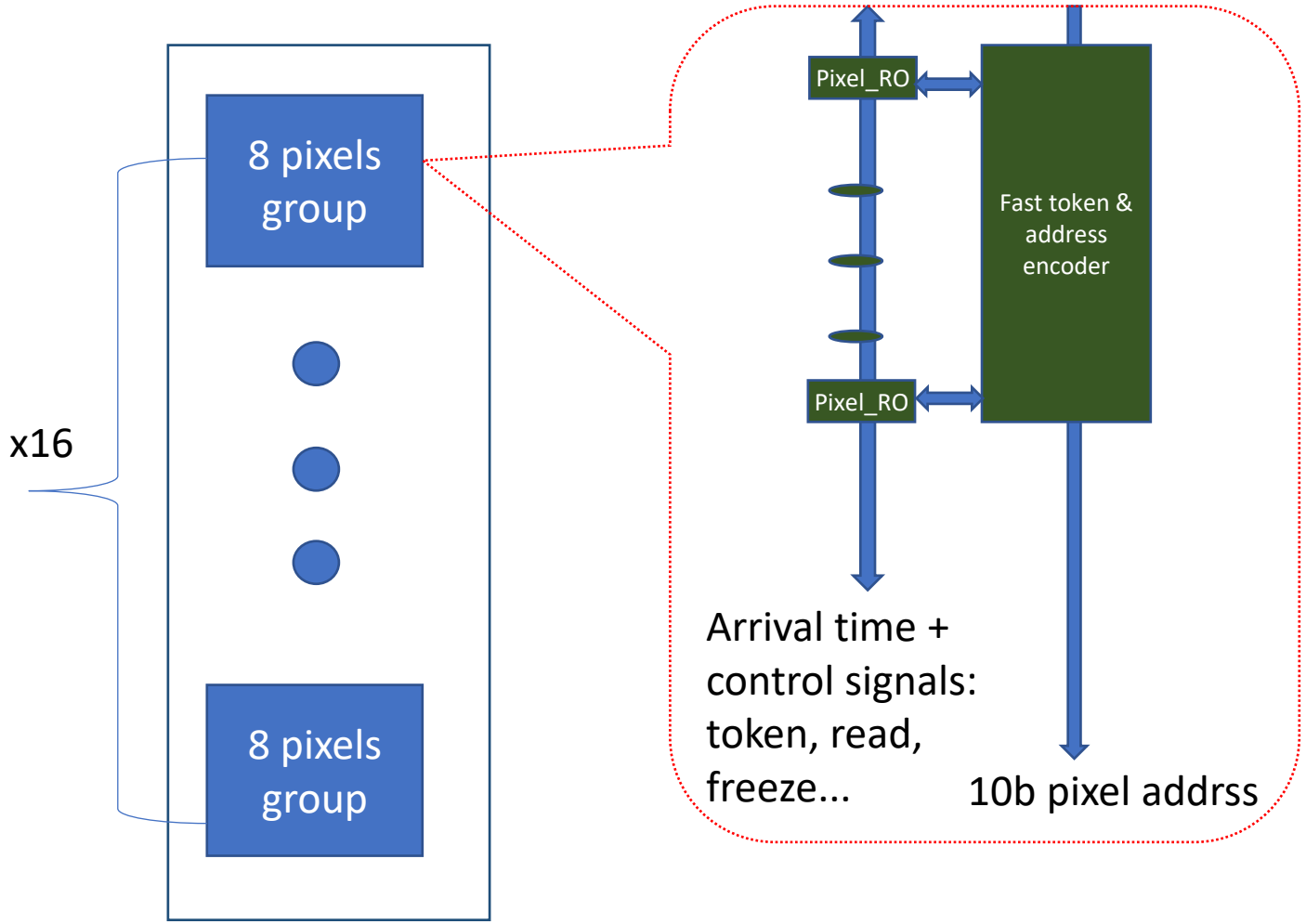
Output data format: column address (5b) + pixel address (10b) + signal edge type (1b) + BCID (10b)

column readout (*pixel_column_readout_vfin.sv*)

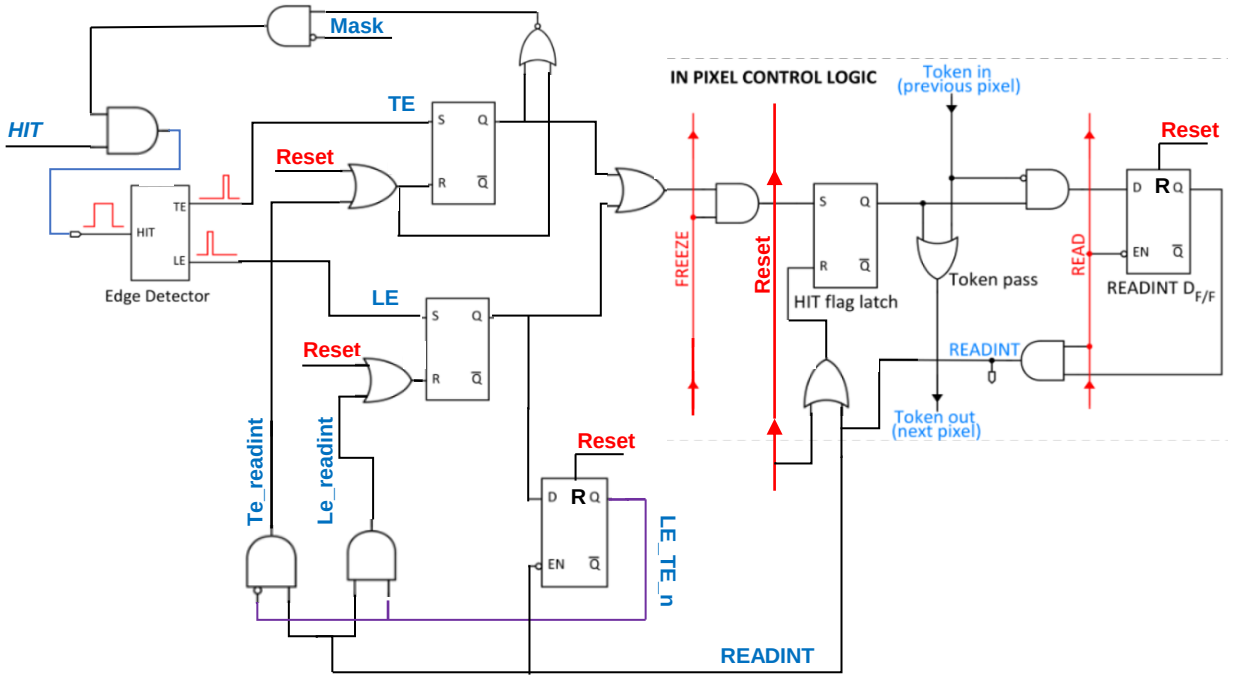
Mainly composed of 16 8-pixel series, each group includes

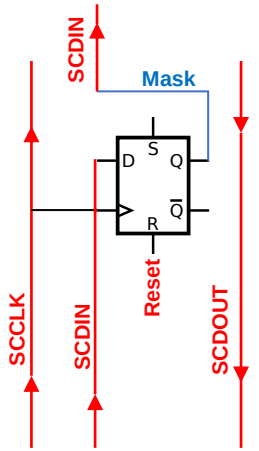
⇒ 8 series pixel readout circuits

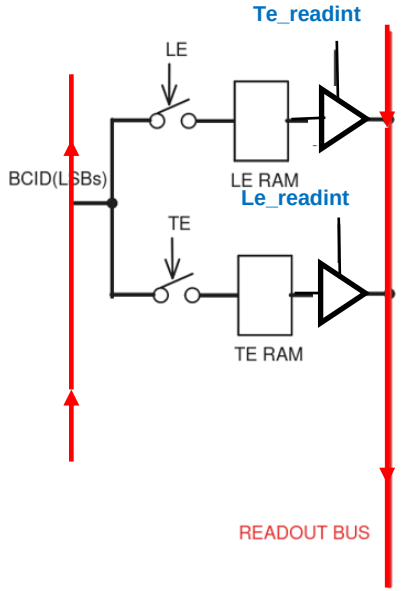
⇒ A circuit for generating fast token and address encoder

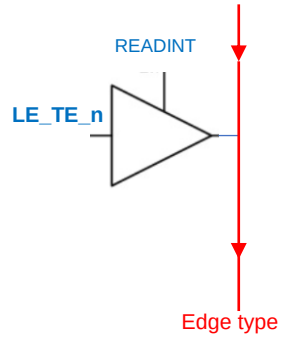


pixel readout (*module_single_Pixel_RO*)



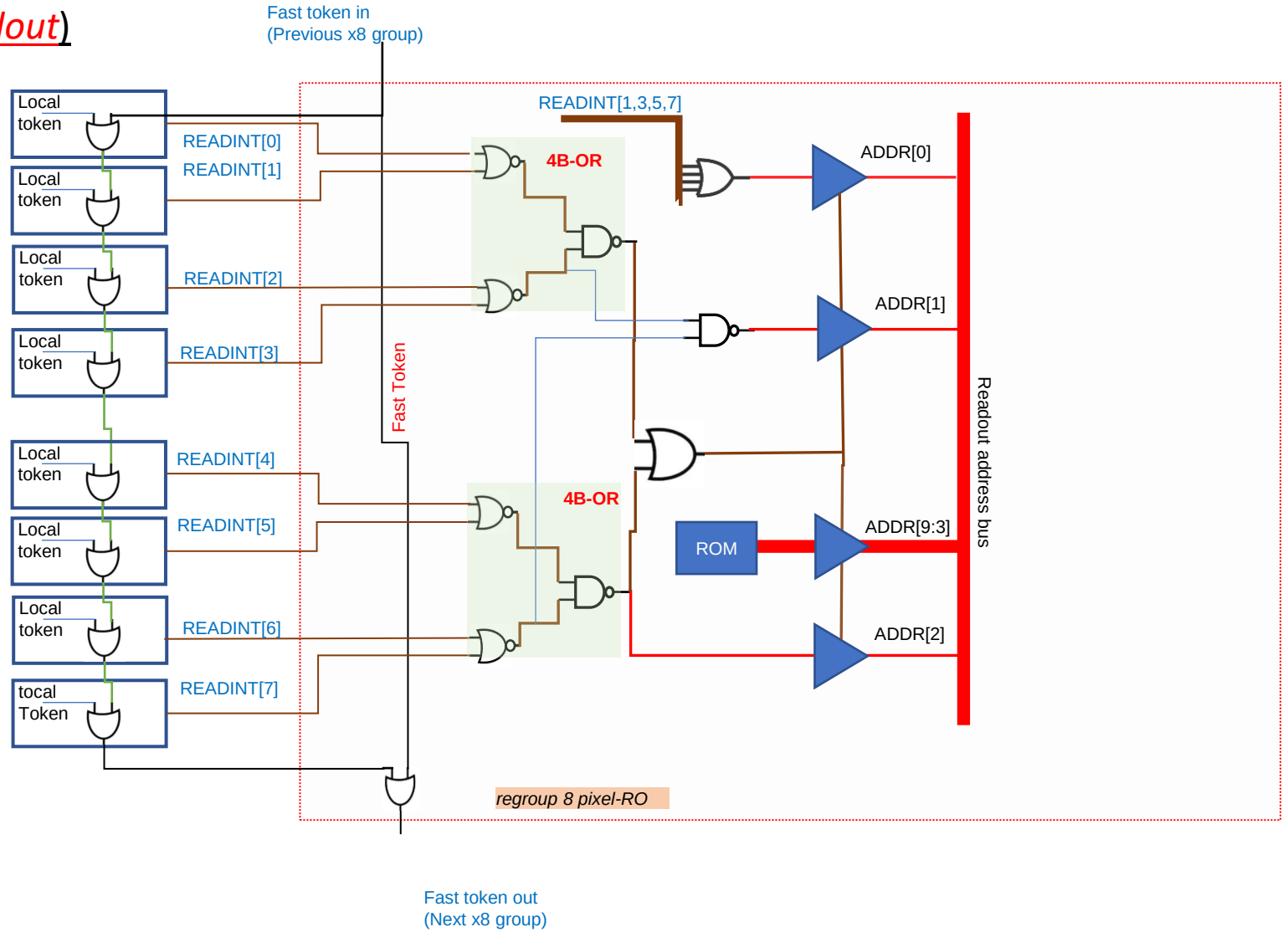






The readout circuit adopts the basic principle of Monopix v1, and the readout of rising edge and falling edge is separated. It works as following: the comparator output (Hit) is first gated and then fed to an edge detector. The gating logic allows pixel masking and prevent a new event from being fed before old one are read. Pixel masking is configurable by slow control. Upon stimuli, the edge detector produces two detected pulses: LE for the rising edge and TE for the following edge. These two pulses trigger their RAM to record the arrival time. They push also their RS logic high, thereby generating a token signal via the combination of OR logic and HIT-flag-latch RS logic. The master (end of column) controls two control signals (freeze and read) for reading out rising or falling edges depending on the "LE_TE_n" value. For each read, through a tri-state circuit, this circuit emits a 4b timestamp (recorded BCID) and 1b edge type (LE or TE). The 10b pixel address is provided by a common circuit for 8 pixels in the same group (see next slide).

8-pixels readout (*module x8_Pixels_readout*)



Two proposes :

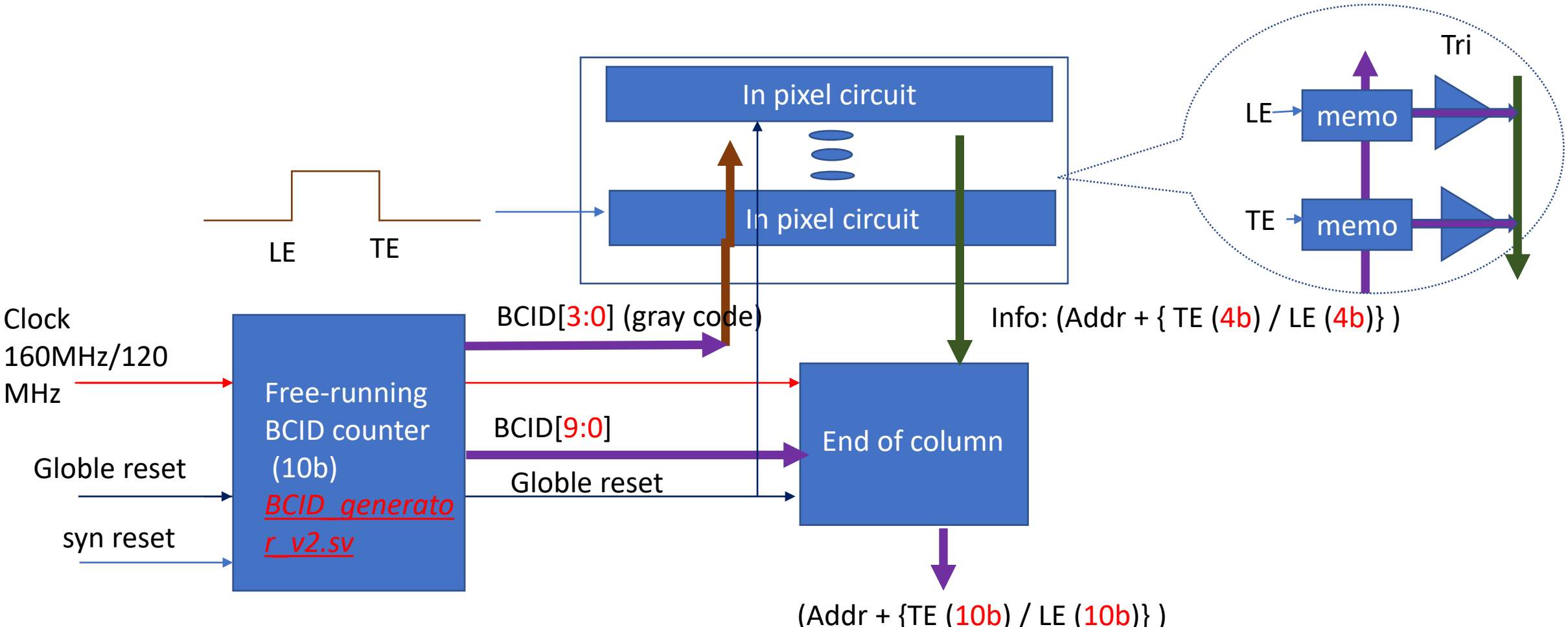
1) Form a fast token that takes less time to propagate to the end-of-column block than going through each pixel's token (monopix V1)
2) Reduce logic consumption of pixel address generator and output tri-state logic : 3b LSB and MSB are provided by 8-bit READINT signal encoder and ROM respectively, , and only one set of tri-state output logic for 8 pixels. Total logic consumption is significantly less than having independent address ROM and tri-state output logic for each pixel (monopix V1).

BCID

Using the system clock (160MHz or 120MHz), the BCID generator generates a 10b BCID for event's time tagging. Dependent slow control configuration, BCID time unit is either 50 ns (default) or 100 ns. BCID counter has a separate reset signal for multiple ASIC synchronization. The counter provides

1) The 4b LSB of BCID is sent to all pixels in Gray code form,
2) The 10b BCID is sent to all end-of-column logic in binary form,

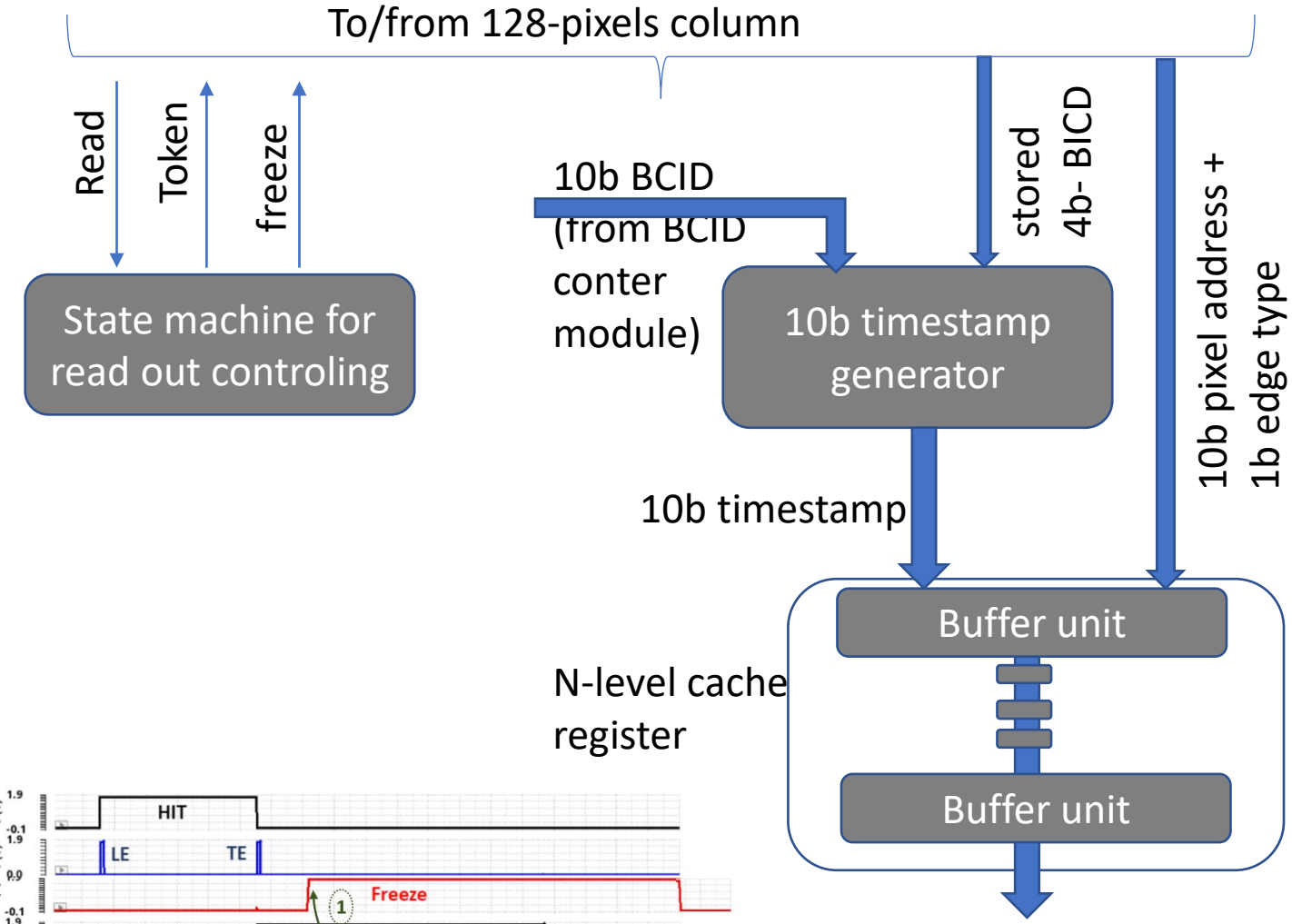
For each stimulus (rising or falling edge), in-pixel circuit sends its pixel address and the stored 4b BCID (in Gray code form). The end-of-column uses the latter with BCID 10b to generate a 10-bit timestamp of the stimulus.

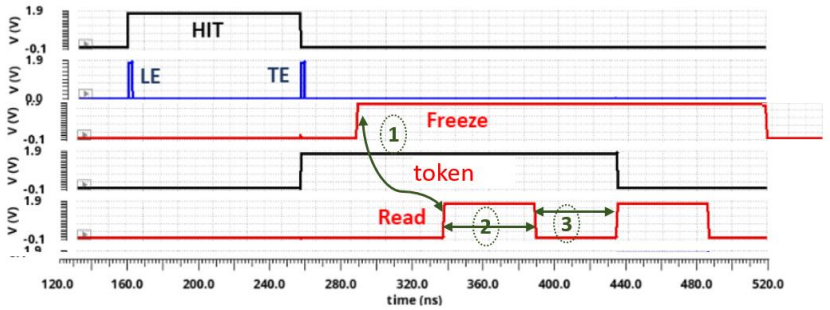


End-of-column (*end_of_column_vfin.sv*)

It mainly allows to control a column to read stimuli (position + 4-bit timestamps) then to extend the timestamps of the stimuli from 4 bits to 10 bits. It consists of the following components:

1) "State machine" for readout controlling via two signals freeze and read. It is triggered by a token signal indicating that one or more events(stimuli) need to be read out. The diagram shows the relationship between these three signals.
2) "10b timestamp generator" allows timestamps to be extended to 10 bits using the free-running 10-bit BCID counter value.
3) Event buffer composed of N cascaded single-event cache registers





32-to-1 multiplexer (*gen_mux.sv*)

Multiplexers are universal modules that can automatically generate multi-layer multiplexers based on the number of input channels. Each layer consists of one or more parallel basic multiplexer units (BMU). Each BMU includes a combinational logic N-to-1 multiplexer, *lower-channel-number-having-higher-priority* selector, and a set of DFFs for event buffering (see figure 1). In addition, the BMU also outputs the selected input channel number.

For our application with 32 input channels (column) (see figure 2) , the instantiated multiplexer has 3 layers: eight 4-to-1 BMUs in the first layer, two 4-to-1 BMUs in the second layer, and one 2-to-1 BMU in the last layer. The output of the previous layer serves as the input of the next layer. Remember that the output of the instantiated multiplexer also gives the address of selected column

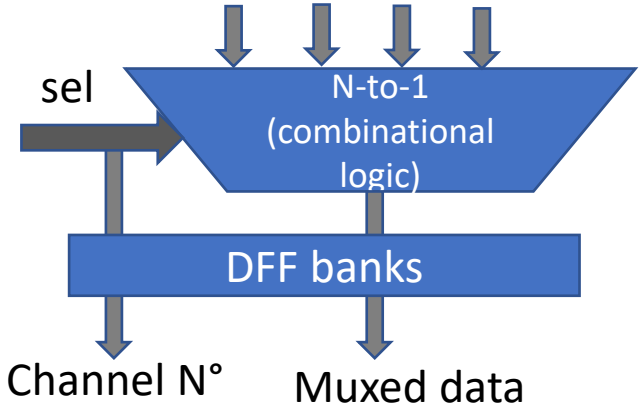


Figure 1

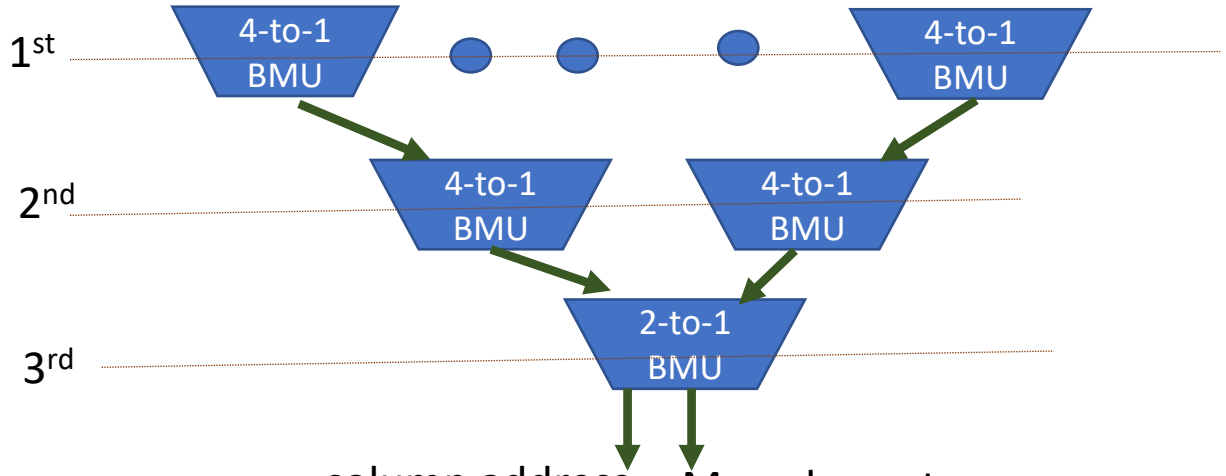
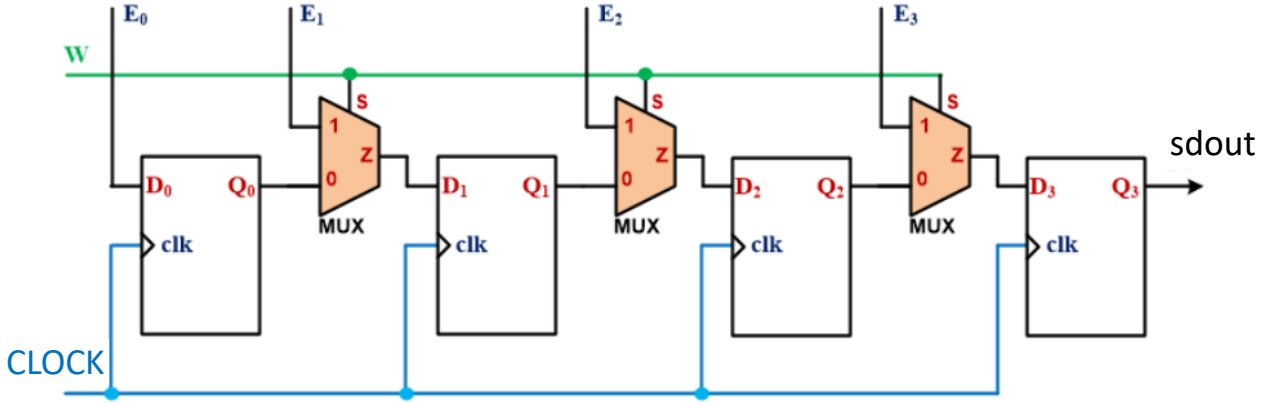


Figure 2

serial output(*tx_spi.sv*)

The principe of serial output is shown in the figure 1. N-bit data (E) to be sent is loaded into N-bit shift registers via N 2-to-1 multiplexers by asserting the “W” signal. The data is then shifted out (sdout) bit by bit at each clock cycle. This module also sends a "valid" signal for SDOUT signal bundling. Figure 2 is the timing diagram. Remember that the LSB bit is sent first for our implementation.



Source https://www.technologuepro.com/

figure 1

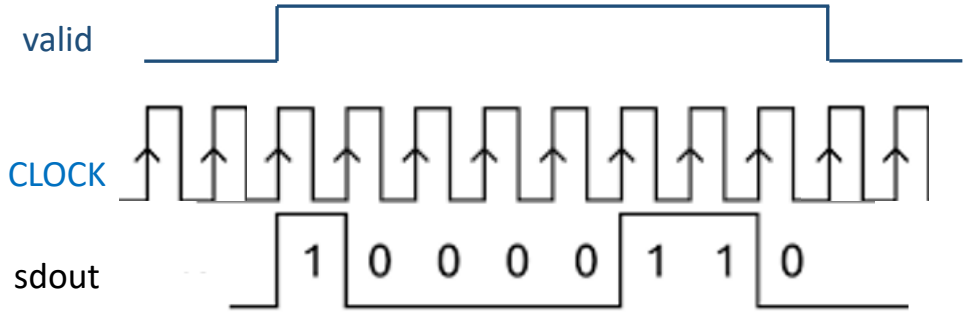
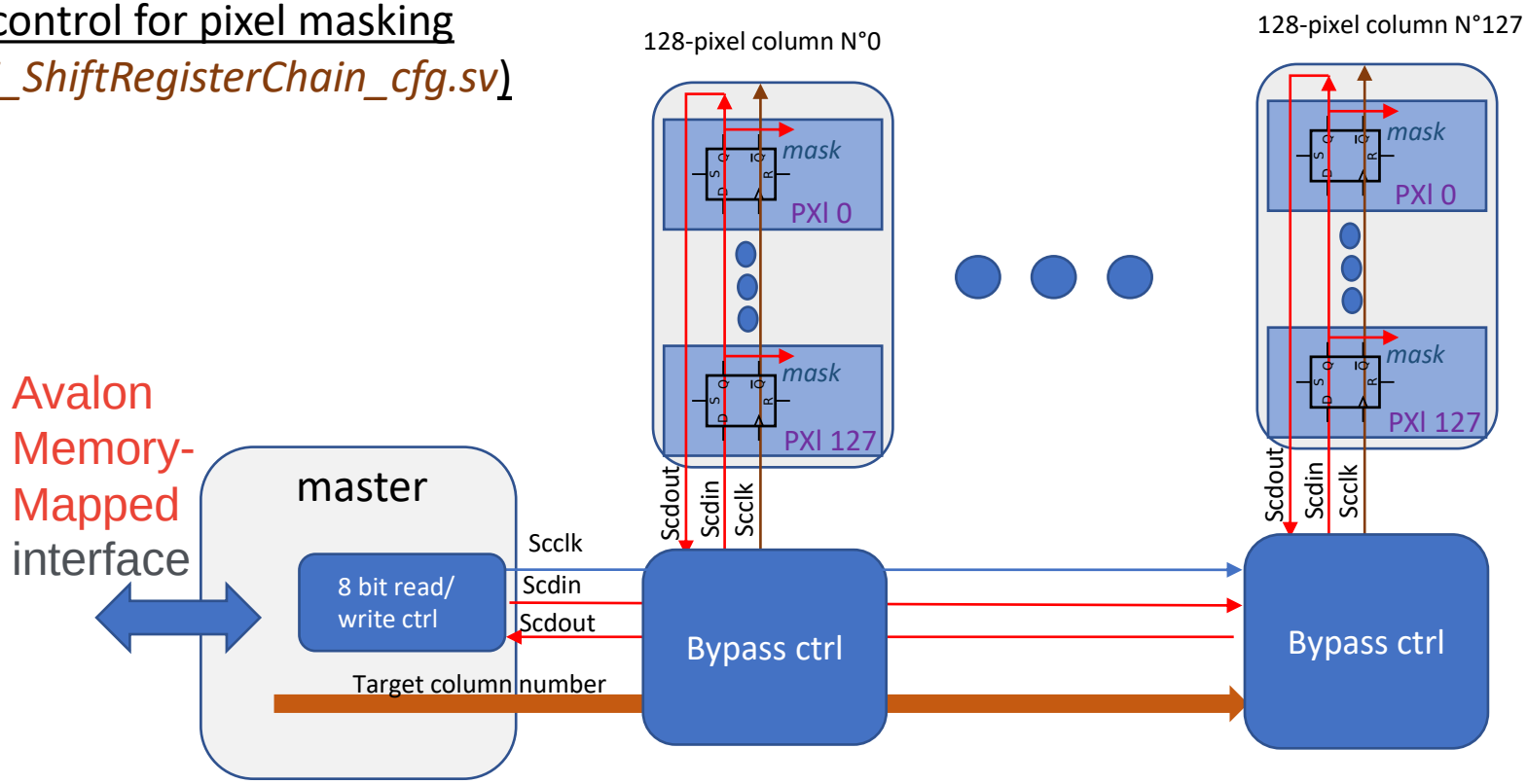


figure 2

Output data format: column address (5b) + pixel address (10b) + signal edge type (1b) + BCID (10b)

Slow control for pixel masking (*multi_ShiftRegisterChain_cfg.sv*)



it consists of several parts:

1) Inside each column, each pixel readout circuit implements one DFF for masking (‘1’:mask) (default: unmasked), and 128 DFFs are cascaded into a 128-bit pixel-masking shift chain (PMSC) .
2) Each PMSC is controlled by a “Bypass ctrl” module, which, depending on whether the column is targeted , connects the PMSC to the master or bypasses the PMSC. In this way, when performing an 8-bit read/write operation, it only involves the operation of the target PMSC, and only requires 128 scclk cycles.
3) The master module mainly includes a controller allowing 8-bit read/write operation on a 128-bit shift chain. The master is triggered via an Avalon memory-mapped interface