

Apsel3D

User Guide

*A digital sparsification and
readout circuit for a 8×32
matrix of MAPS*

Alessandro Gabrielli

**INFN & Physics Department
University of Bologna – Italy**

on behalf of the SLIM5 collaboration

1. Notes for the readers

- All the I/O ports and keywords are shown in **bold blue type**,
- In the block figures of the circuit, the left ports are to be considered as inputs while the right ones are outputs and this rule applies to all the blocks described in this document,
- MPs = Macro Pixels, 16 altogether,
- MCs = Macro Columns, 8 altogether,
- MRs = Macro Rows, 2 altogether,

1.1 Version

- | | | |
|---|-------------------|------------------------|
| - | Datasheet version | 1.0 - 17 June 2007 |
| - | “ “ | 1.1 - 19 June 2007 |
| - | “ “ | 1.2 - 25 June 2007 |
| - | “ “ | 1.3 - 11 July 2007 |
| - | “ “ | 1.4 - 17 December 2007 |

For any question, please email to: alessandro.gabrielli@bo.infn.it

This document can be found in the following web-site:

<http://www.bo.infn.it/slim5/Apsel3D/Readout32x8-Users-Guide.pdf>

2. Introduction

The circuit is a digital architecture for a sparsified readout that interfaces with a matrix of 256 Monolithic Active Pixel Sensor (MAPS). It is the base for a prototype of a mixed-mode ASIC, namely Apsel3D. It reads out and sparsifies the hits of a matrix of 256 pixels. Once read, the hits are switched off. The matrix is divided into regions of 4 x 4 single pixels thus, 256 pixels are clustered into 16 groups of 16 pixels each, here-in-after named macropixels (MPs). In addition, the matrix is arranged in 32 columns by 8 rows of single pixels or, from a different viewpoint, in 8 columns of MPs, by 2 rows of MPs, called MRs.

Basically, let us say that when the matrix has some hits (pixels that detect an over-threshold charge), it is swept from left to right and, at each clock period, all the hits present in a column of pixels, from 1 to 8, can be read out. This operation starts as long as a hardwired readout queue has free locations to temporarily store the information of the hits. In fact, at the hits' coordinates is associated a time mark (time-stamp) and the overall formatted data are either sent to the output port, or temporarily stored in a FIFO-like memory in case the output port is busy. Thus, in principle, the architecture can read out the matrix up to 8 hits at a time in case they belong to the same column and can send the formatted data to the output but, at the same time, the output port can accept only one hit information at a time and this is why a queuing system is necessary.

Moreover, the global architecture might be considered as a circuit that can run in two different operating modes, called **custom-mode** and **digital-mode**. In fact, it can be connected to an actual full-custom matrix of MAPS or to a digital matrix emulator composed of standard cells. In the first case the pixels may only be switched on via striking particles while in the second case the digital matrix must be loaded during an initial slow-control phase. The two different implementations share the same matrix's I/O pins but can be selected and activated only one at a time. For both modes, before running, a slow-control phase is required to load an internal configuration. In particular, 16 mask signals should be provided to select the MPs which are to be read and which are not, for examples in case they are too noisy or broken. Default mask, after a reset phase, is all-at-1, meaning no-mask. Moreover, it must be selected which of the two operating modes is wanted and, consequently, which matrix is to be enabled. The default mode, after a reset phase, is the **digital-mode**. In addition, only for the digitally emulated matrix, 256 registers should be loaded to simulate a given charge injection over the silicon area. Default registers, after a reset phase, are all-at-0, meaning no hits. The readout circuit operates in the same manner for the two modes. Fig. 1 shows a sketch of the operating modes of Apsel3D.

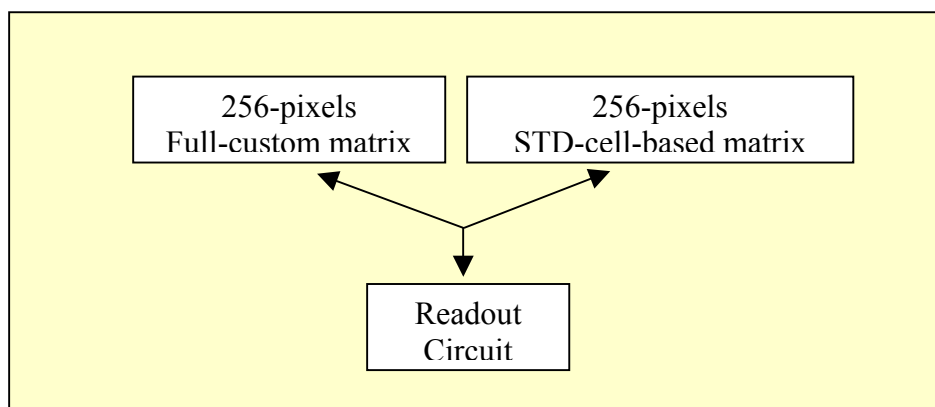


Figure 1: Apsel3D operating modes: custom-mode and digital-mode

3. Readout Block

The figure shows the I/O ports of the entire circuit plus **SEED_VECTOR** and **TEST** ports used only for design and simulation purposes: they will not be present on the final ASIC.

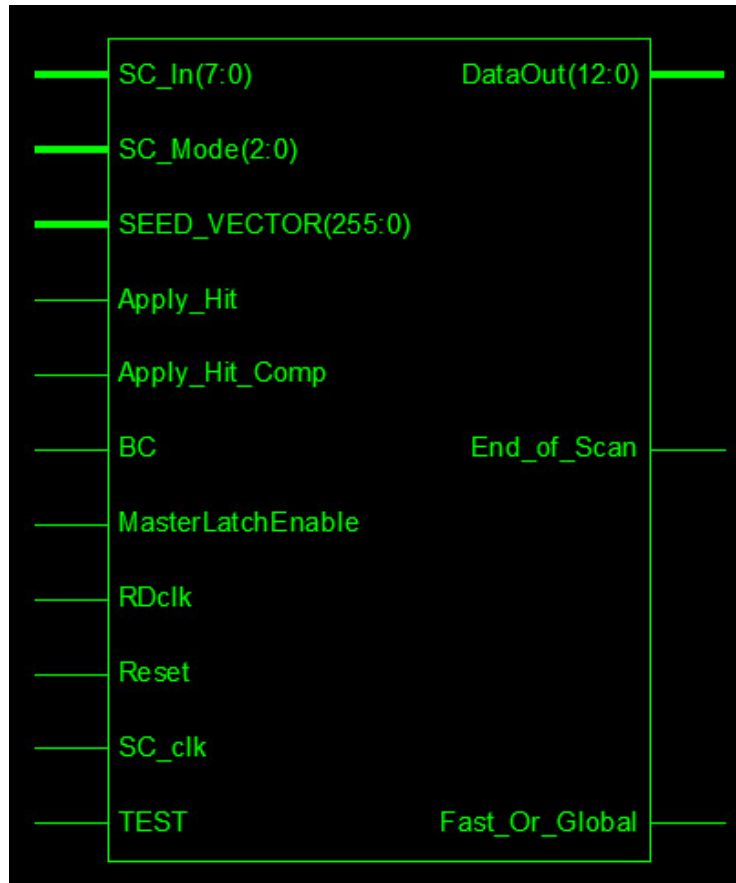


Figure 2: Readout Block

Thus, for this circuit the 256-bit **SEED_VECTOR** and the **TEST** ports are not real, they have been used only to simulate the whole readout architecture. Here is a list of all the I/O ports:

- **SC_In** is an 8-bit port used to load the internal registers, depending on the **SC_Mode** value, at the rising edge of **SC_clk**,
- **SC_Mode** is a 3-bit port which specifies the required slow-control operation. During a slow-control phase several operations may be requested:
 - a dummy-pixels load, in **digital-mode**, to configure the dummy matrix. It requires 32 **SCclk** periods as it loads 8-bits at a time as internal hits,
 - a mask load to select the pixels that must be really seen from others which must be masked. It requires 2 **SCclk** periods as it loads 8-bits at a time,
 - an operating-mode selection to enable either the **custom-mode** or the **digital-mode**. It requires one **SCclk** period as it loads just a 1-bit register,
 - a scramble operation on the dummy-pixels (sum, not, shift). It requires one **SCclk** period as it is a no-load operation,
 - a **Soft_Reset** to reset the time-stamp counter. It requires one **SCclk** period as it is a no-load operation.

All these operations are synchronized on the rising edge of the slow-control clock, **SC_clk**, by following a proper scheme. Here is a summary of the operations that can be selected via the **SC_Mode** slow-control port:

SC_Mode = 000	→	load 256 Dummy-Pixels . First MP address ranges from 0 to 15, second from 16 to 31, and so on up to 16 th MP which ranges from 240 to 255. . At each SC_clk period a half MP is configured, 32 periods for 16 MPs. This is only valid in digital-mode ,
SC_Mode = 001	→	load 16 Mask bits. At first SC_clk period Mask (7:0) are loaded while Mask (15:8) are loaded in a second SC_clk period. During normal running mode, Mask (i) masks MP(i),
SC_Mode = 010	→	load the internal master-latch-enable MLE_Reg (not used) and the Actual_Dummy registers. The latter Actual_Dummy bit enables the custom-mode when is low and the digital-mode when is high,
SC_Mode = 011	→	reserved
SC_Mode = 100	→	Soft Reset request to reset only the time-stamp counter inside the Time-Stamp Block ,
SC_Mode = 101	→	1-bit rolling shift on the Dummy-Pixels ($N^{th} \rightarrow N^{th}+1$, $255^{th} \rightarrow 0^{th}$). This is valid only in digital-mode ,
SC_Mode = 110	→	add 1 to each MP 16-bit configuration. If a given MP has a given configuration of 16 bits, then this configuration is added to 0x"0001" to get a new configuration. This is valid only in digital-mode ,
SC_Mode = 111	→	this is used as the normal running mode once either the dummy-pixels have been loaded in digital-mode or the matrix of MAPS is connected in the custom-mode ,

The **SC_Mode**="010" is crucial because it sets the operating mode. The reserved configurations are not used and the **SC_Mode**="111" is used to the normal run operation after the slow-control phase,

- **SEED_VECTOR** is a 256-bit vector that provides the 16 MPs with a 16 different stimuli composed of 16 bits each. The stimuli are read from a text file, whose single lines correspond to a coded single stimulus, and are applied whenever a **TEST** rising edge is provided. Thus, if n **TEST** pulses are provided, $16 \times n$ lines of the file of stimuli are read, and 16×16 -bit patterns are provided to the 16 MPs n times. These stimuli can be seen or not depending on the MPs' **Latch-Enable** status. In fact, if this is low, it means the the MP is frozen and blind to new hits,
- **Apply_Hit** and **Apply_Hit_Comp** are signals, considered only for **digital-mode**, which force the stored dummy pixels to be copied into mirror registers (called **Latches** even if hardwired with FFs) to be read out as they were output latches of the pixel sensors. The process activates on the rising edge of **SC_clk**. Thus, when the **Apply_Hit/Apply_Hit_Comp**="10" configuration is seen at the **SC_clk** rising edge, the **Dummy-Pixels** (in their 0/1 status) are copied to the 256 **Latches**, while the "01" configuration of **Apply_Hit/Apply_Hit_Comp** indicates that a complemented version ($1 \rightarrow 0$ and $0 \rightarrow 1$) of the **Dummy-Pixels** must be stored on the **Latches**. Once the "10" or "01" **Apply_Hit/Apply_Hit_Comp** configurations are applied, the readout logic starts if it was standing or continues its matrix-sweep if it was on running. Once the whole matrix is swept, the read process stops. Then, a "00" configuration is required to let the column

sweeping process starts again from the left-most column. This allows for avoiding immediate re-readout of the hits,

- **BC** is the Bunch-Xing signal, considered asynchronous, which forces an internal time-stamp register into being copied as a time-mark for any of the MPs that contain at least one hit (MP's FastOr is high). This is carried out after the BC is synchronized to the global readout clock **RDclk**. The **BC** signal is masked by the **SC_Mode(2)**; in other words, during slow-control phases, the **BC** external signal is not seen,
- **MasterLatchEnable** is an asynchronous signal inserted with an *and* logic function along with the internal 16 MPs' **Latch-Enable** signals and with the 16 **Mask** registers. In other words, eventually, the **Latch-Enable** signals that enter the MPs are given by:

$$\text{Latch-Enable}(i) \leq \text{LatchEnableNMatrix}(i) \text{ and } \text{Mask}(i) \text{ and } \text{MasterLatchEnable}$$

where **Mask(i)** are the 16 masks loaded via slow-control and **LatchEnableNMatrix(i)** are the 16 signals provided by the readout control unit for the 16 MPs. These latter signals are synchronized with the rising edge of **SC_clk**. Masked MPs have **Mask(i) = 0**. If **MasterLatchEnable** is low all the **Latch-Enable(i)** are forced to '0' and the MPs to which they belong are frozen and kept blind until **MasterLatchEnable** goes high again,

- **RDclk** is the 40 MHz global clock,
- the three ports with the prefix **SC_**, (**SC_Mode**, **SC_In** and **SC_clk**) stand for slow-control inputs and are used to configure the circuit,
- **Reset** is a hard-reset that resets all the internal registers to a predefined state as soon as it is seen and synchronized via **RDclk**: basically 0 for the registers and 1 for the masks, **digital-mode** for operating mode. It must last high for more that one **SC_clk** and **RDclk** period to take effect.
- **SC_clk** is an up to 40-MHz slow-control clock. It is asynchronous with respect to **RDclk** when it is used to load the internal registers as the **Dummy-Pixels** in **digital-mode** or the masks **Mask(i)** in both modes. When a given configuration of hits has been loaded into the **Dummy-Pixels**, and they have to be copied to the **Latches** via **Apply Hit/Apply Hit Comp** couple, the **SC_clk** clock must be run with the same frequency of the **RDclk** clock. This is why, in **digital-mode**, the dummy matrix is updated on the rising edge of **SC_clk** while it is frozen, read and reset on the rising edge of **RDclk**. Using two different clock frequencies may lead to unpredicted states. Let's say that **SC_clk** frequency must be not lower than **RDclk** during **digital-mode** matrix readout. In **custom-mode** instead, the **SC_clk** may be run at any frequency during configuration.
- **TEST** is an asynchronous non-real port used to specify when the **SEED_VECTOR** must be read and its values must generate the simulated hits, in **digital-mode**,
- **DataOut** is a 13-bit port. It is the output data bus with the following format:

<Data Valid><Pixel Row><Pixel Column within MP><Macro Column Address><Time Stamp>

- Data Valid is a 1-bit signal: 1 when the output word is valid,
 - Pixel Row is a 3-bit bus: 0 to 7 as the row moves from bottom to top,
 - Pixel Column inside a MP is a 2-bit bus: 0 to 3 as the MP column moves from right to left,
 - Macro Column Address is a 3-bit bus: 0 to 7 as MC moves from right to left,
 - Time Stamp is a 4-bit time mark: 0 to 15 incremented upon BC rising edge.
- **End_of_Scan** is a single bit that indicates the end of a bunch readout. In other words, a pulse on this port confirms that a total sweep over the columns of pixels, from left to right, has finished. This occurs some clock periods later the sweeping phase has concluded, accounting for the internal latencies,

- **Fast_Or_Global** is a single bit that indicates the status of the MPs' **FastOr** output pins. In particular it is a global *nor* logic function of the 16 **FastOr** signals after the masking operation. By following the above description for MasterLatchEnable,

$$\text{Fast_Or_Global} \leq \text{nor}_{(0-15)} [\text{FastOr}(i) \text{ and } \text{Mask}(i)].$$

3.1 Configuration Steps

For this operation **Apse13D** requires, after being powered, a couple of clock periods, both on **SC_clk** and **RDclk**, with **Reset**='1' and **SC_Mode** ≤ "010". This is the way all the internal registers are initialized to their default values and the chip is ready to run in **digital-mode**. Then, depending on the required operating mode, other slow-control operations might be provided. It is important that, during this phase, **SC_Mode** does not move forth and back from the highest to the lowest values. In other words, if the **Dummy-Pixels** or the **Masks** have to be loaded, this step must be done once after reset phase. Then, when all the desired internal registers have been loaded, along with the operating mode register **Dummy Actual**, the **SC_Mode**="111", meaning running mode, can be provided once and together with at least one **SC_clk** rising edge. Figures 3 and 4 show this configuration steps in **custom-mode** and **digital-mode**, respectively. The **SC_Mode**=0, **SC_Mode**=1 and **SC_Mode**=2 may be swapped.

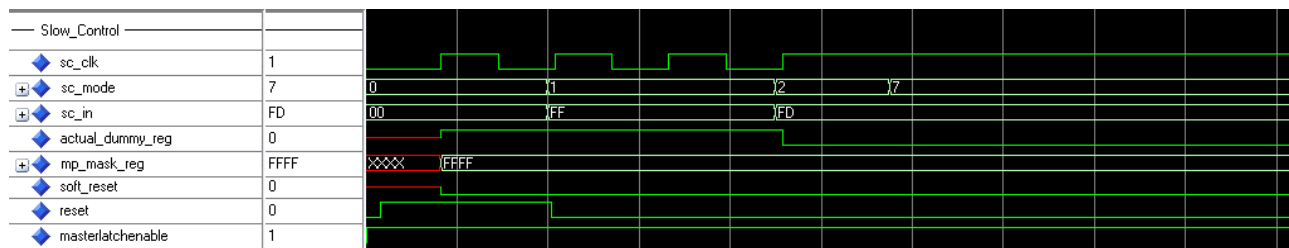


Figure 3: Simulation of the APSEL3D configuration in **custom-mode**

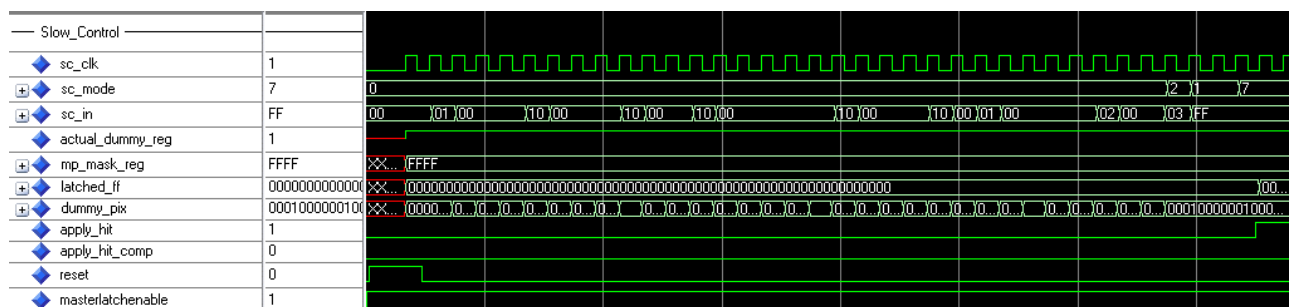


Figure 4: Simulation of the APSEL3D configuration in **digital-mode**

For both modes there is a **Reset**='1' initializing phase. The **Reset** port is to be seen by both **SC_clk** and **RDclk** clocks. As it should be clear from the figures, in **custom-mode**, after the configuration the **SC_clk** can be frozen while, in **digital-mode**, cannot. In digital-mode, besides two periods of **SC_clk** with **SC_Mode**=1 and one period with **SC_Mode**=2 (that could be omitted as it overwrites the default operating **digital-mode**), 32 periods with **SC_Mode**=0 are required to load the **Dummy-Pixels**.

In both modes after the initial **Reset**= '1' is applied, 8 **RDclk** periods are once required to provide the reset codes to the MCs. This assures that all the pixels are switched off.

3.2 Internal Registers

For its operation modes the circuit has some internal registers that must be loaded via the slow-control.

REGISTER	# OF BITS	OPERATING MODE	NOTES
Dummy-Pixels	256	digital	- Dummy MAPS, default at '0' at Reset = '1', - Loaded with SC_Mode = "000", 8-bits at a time - Scrambled with SC_Mode = "101" or "110"
Latches	256	digital	- Dummy MAPS' output latches, default at '0' at Reset - Updated on rising edge of SC_clk when Apply_Hit/Apply_Hit_Comp = "10", "01"
Mask	16	both	- no-mask = '1', default at '1' at Reset = '1', - Loaded with SC_Mode = "001", 8-bits at a time
MLE	1	not-used	- internal MasterLatchEnable , non-active = '1' - Loaded with SC_Mode = "010", SC_In (0)
Dummy_Actual	1	both	- '1' = digital-mode , default at Reset = '1', - '0' = custom-mode , - Loaded with SC_Mode = "010", SC_In (1)

3.3 Stand_By

The entire circuit, whenever is not able to store the data provided by the **Sparsifier** circuit, stops the sweeping of the pixel columns. This happens when the number of free location in the **BarrelOut** circuit is smaller than the number of data that are going to be stored. In such a situation the readout stops for 16 (**BarrelOut** depth) **RDclk** periods until the **BarrelOut** circuit is empty for sure.

3.4 Latency

After a **BC** signal is detected by a **RDclk** rising edge, it is synchronized at the following **RDclk** period and, if some hits are present, the entire readout architecture takes 7 **RDclk** periods to out the first valid data. In other words, a latency of 7 clock periods can be retained as the minimum time before a valid output is provided, after the internally synchronized **BC** signal. Thus, the whole latency of the **DataOut** port, with respect to the **BC** signal, is 8 clock periods (after the **RDclk** rising edge that detects the **BC** rising edge).

3.5 Simulations

Below follow two plots corresponding to the two operating modes. The plots have just a graphical purpose and, of course, are not to be clearly understood.

Fig. 5 shows the circuit operating in the **custom-mode**. By following the picture, from left to right, it can be seen a slow-control stage where the internal registers are loaded. After a given time, some **TEST** pulses are provided, then **BC** signal starts and, consequently, the **DataOut** bus begins to out valid data. Three **End_of_Scan** pulses are also visible.

Fig. 6 shows the circuit operating in the **digital-mode**. By following the picture, from left to right, it can be seen a slow-control stage that here lasts a longer time as all the internal registers correspondent to the **Dummy-Pixels** that have to be loaded. After a give time, some **BC** signal are provided (here does not make sense to provide the **TEST** pulses) and, consequently, the **DataOut** bus begins to out valid data.

Three **End_of_Scan** pulses are also visible. On the right hand side of the picture there is a **soft-reset** request, by means of the **SC_Mode** value 4. In Fig. 6 the two clocks, **RDclk** and **SC_clk** have the same frequency.

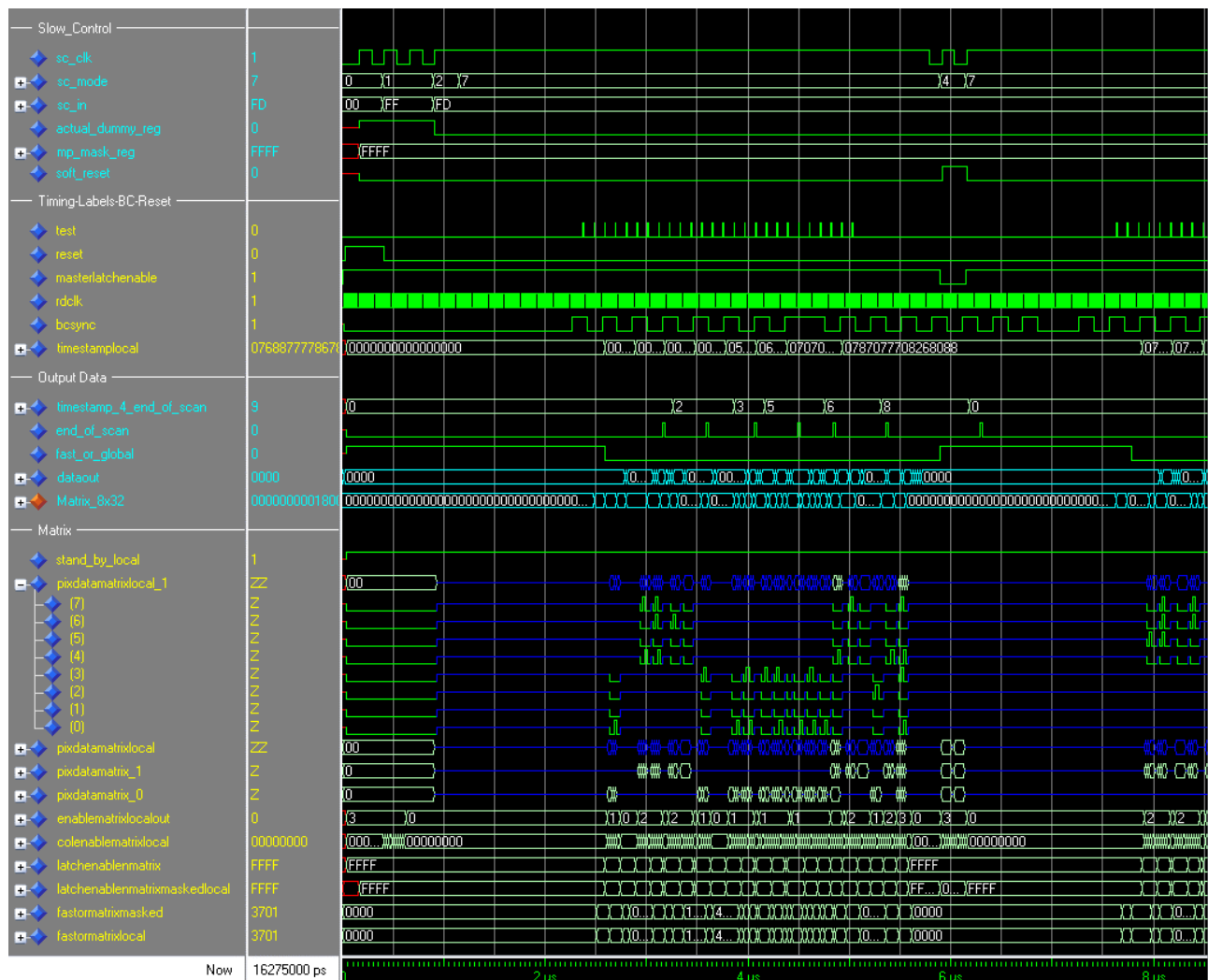


Figure 5: Simulation of the entire circuit in the **custom-mode**

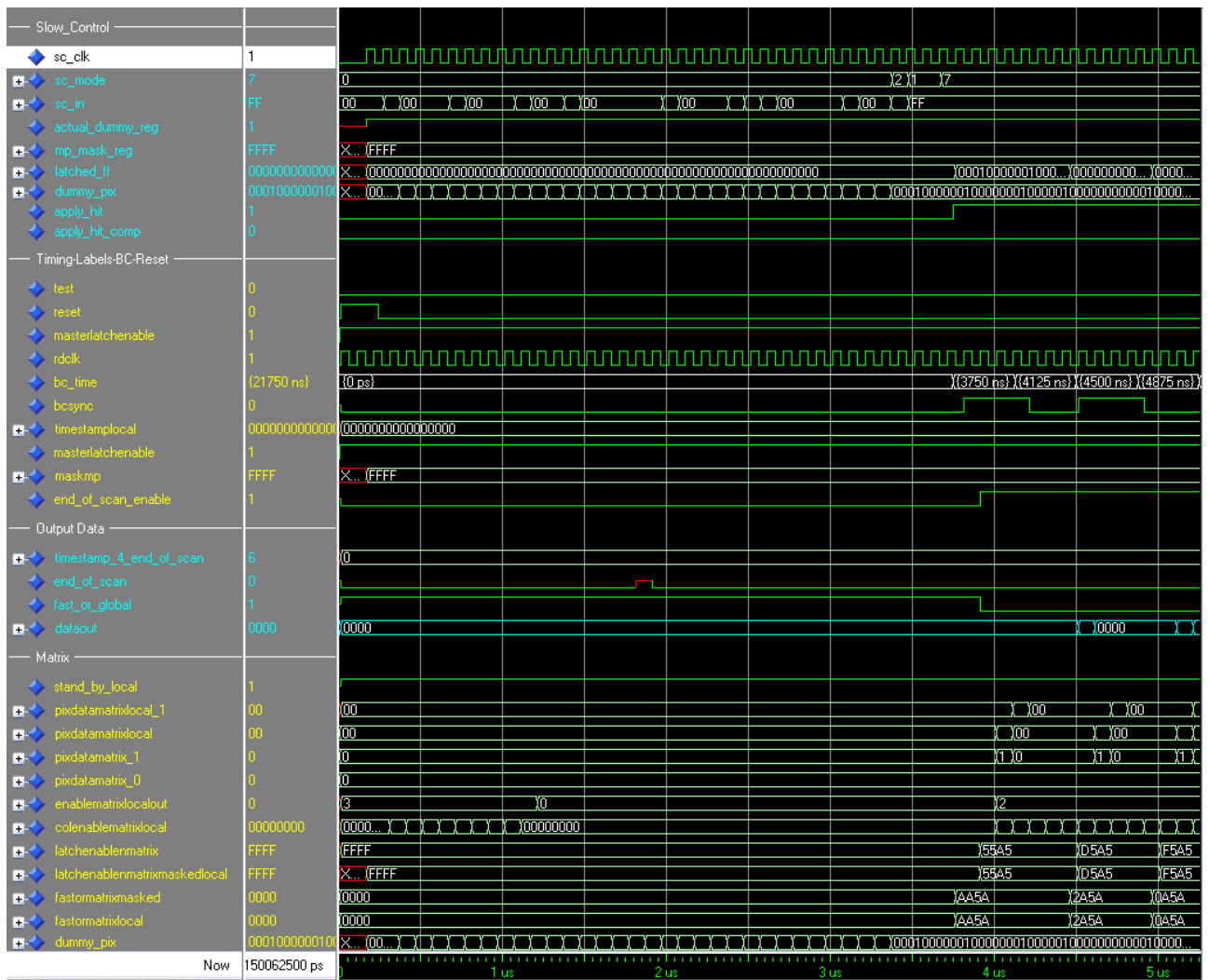


Figure 6: Simulation of the entire circuit in the digital-mode

4. Time Stamp Block

The figure shows the I/O ports of the TimeStamp circuits.

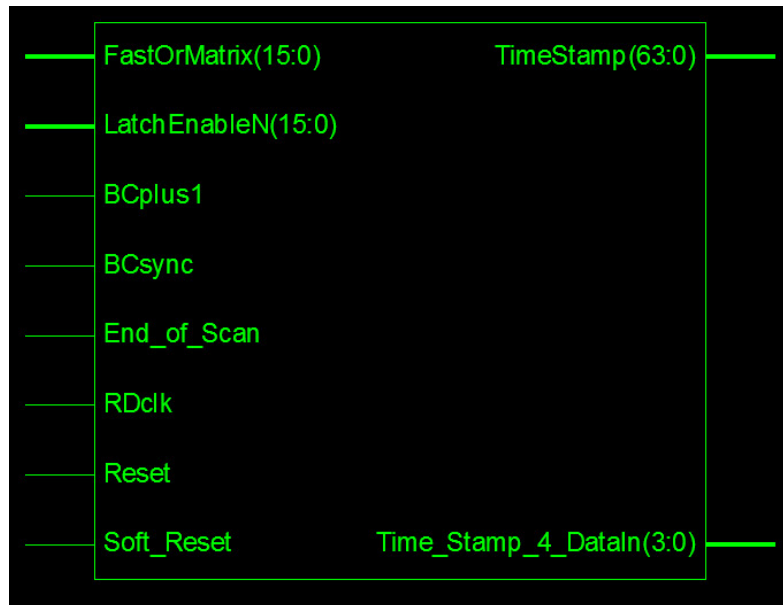


Figure 7: TimeStamp Block

Here follows a description of the I/O ports:

- **FastOrMatrix** is a 16-bit port that reads the relative **FastOrMatrix Matrix** output, after being masked depending on the internal 16 Mask register,
- **LatchEnableN** is a 16-bit port that reads the relative **LatchEnableNMatrix** Latch-Enable-Block output, after being masked depending on the internal 16 Mask register and the external **MasterLatchEnable** signal,
- **BCsyn** is the bunch-Xing signal described in the **Latch Enable Block**,
- **Bcplus1** is the bunch-Xing signal described in the **Latch Enable Block**,
- **End_of_Scan** is a single bit that indicates the end of a bunch readout. In other words, a pulse on this port states that a total sweep over the columns of pixels, from left to right, has finished,
- **RDclk** is the external clock described in the **Readout Block**,
- **Reset** is the external reset described in the **Readout Block**,
- **Soft_Reset** is an external requested reset to let the **time-stamp** register start from 0,
- **TimeStamp** is a 64-bit port that provides the status of the time mark of all the 16 MPs (16 by 4-bits of the time mark),
- **Time_Stamp_4_dataIn** is a 4-bit port that provides the proper time mark related to the **End_of_Scan** signal.

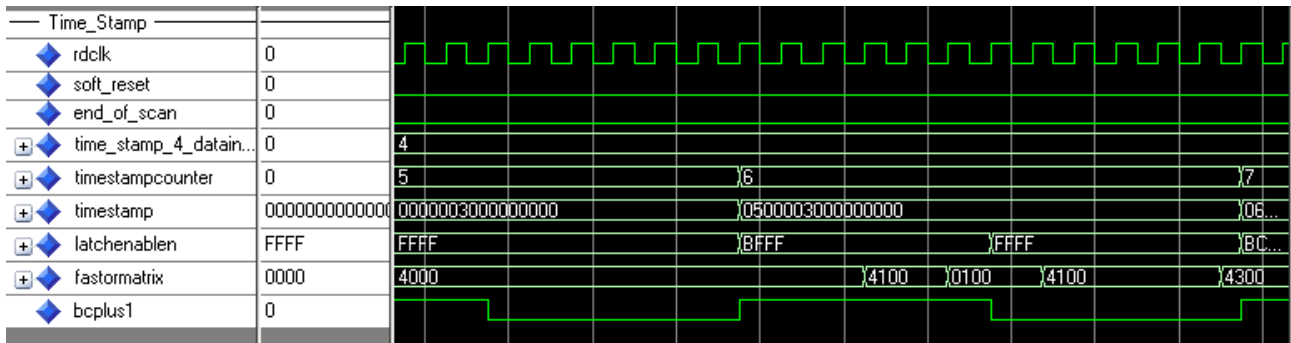


Figure 8: Simulation of the TimeStamp Block

Fig. 8 shows the circuit simulation. The **TimeStamp**, **Time_Stamp_4_dataIn** and **End_of_Scan** signals are valid immediately after **Bcplus1**. Also the internal **TimeStampCounter** is shown and it can be seen that it updates on the **Bcplus1** rising edge.

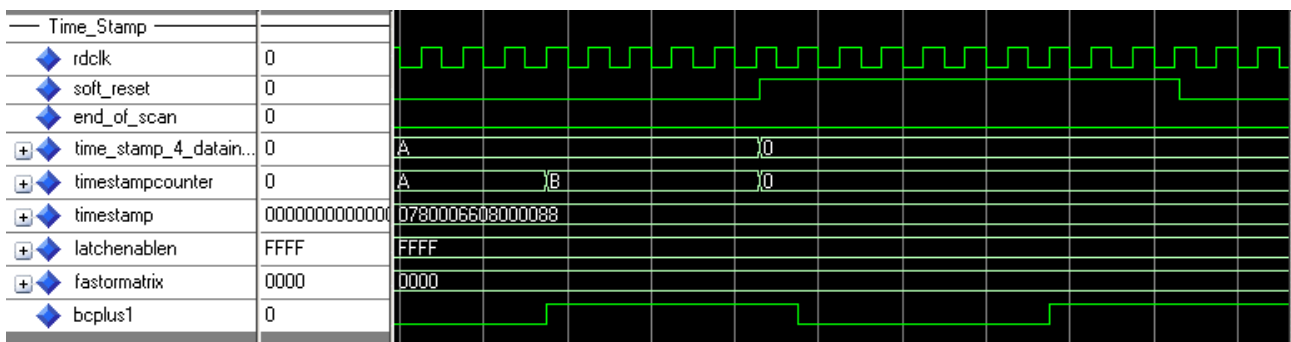


Figure 9: Simulation of the TimeStamp Block

Fig. 9 shows the simulation when a **Soft_Reset** occurs: the internal **TimeStampCounter** is reset to 0.

4.1 Latency

After a **BCsyn** / **Bcplus1** couple “10” is detected by a **RDclk** rising edge, the **Time-Stamp** counter, the **TimeStamp** and the **Time_Stamp_4_DataIn** ports are updated. Total latency from the asynchronous **BC** is 2 **RDclk** periods after **BC** is detected. The latency is 2 periods.

5. Barrel Out Block

The figure shows the I/O ports of the BarrelOut circuits. This circuit provides a queue for the output data. As the entire architecture reads out at most one valid 13-bit word at a time, i.e. at a **RDclk** cycle, in case more than one hit is read in parallel from the matrix, the exceeding hits must be temporarily held into a FIFO-like memory. This memory is a barrel that can be written with 1 to 8 24-bit words, and can be read one location at a time. The barrel depth is 16: all in all it has 16 locations of 24-bit words even though just a subset of the overall bits are used.

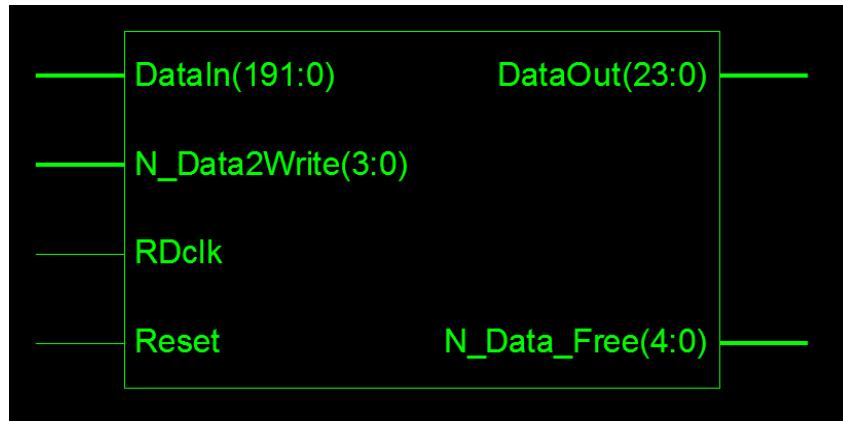


Figure 10: BarrelOut Block

Here follows a description of the I/O ports:

- **DataIn** is a 192-bit port that reads the relative **DataIn** output of the **Sparsifier Block**. 192 bits are to be considered the maximum width of the valid data. In fact, it is 24-bit times 8 hits. Here only 13-bits out of 24 are really used: the synthesizer removed unused registers,
- **N_Data2Write** is a 4-bit port that reads the number of valid data to pack together into the barrel. The **N_Data2Write** is provided by the readout control unit,
- **RDclk** is the external clock described in the **Readout Block**,
- **Reset** is the external reset described in the **Readout Block**,
- **N_Data_Free** is a 5-bit port that send to output the number of free locations of the barrel,
- **DataOut** is a 24-bit port. It is the output data bus with the following format:

<10-bit-X><Pixel-Row><2-bit-X><Pixel-Column-within-MP><Macro-Column-Address><Time Stamp>

- Pixel Row is a 3-bit bus: 0 to 7 as the row moves from bottom to top,
- Pixel Column within MP is a 2-bit bus: 0 to 3 as the MP column moves from right to left,
- Macro Column Address is a 3-bit bus: 0 to 7 as MC moves from right to left,
- Time Stamp is a 4-bit time mark: 0 to 15 incremented upon BC rising edge,
- XX are unused bits.

Only 12 out of 24 bits are used to which one extra DataValid bit will be added via the Readout control unit.

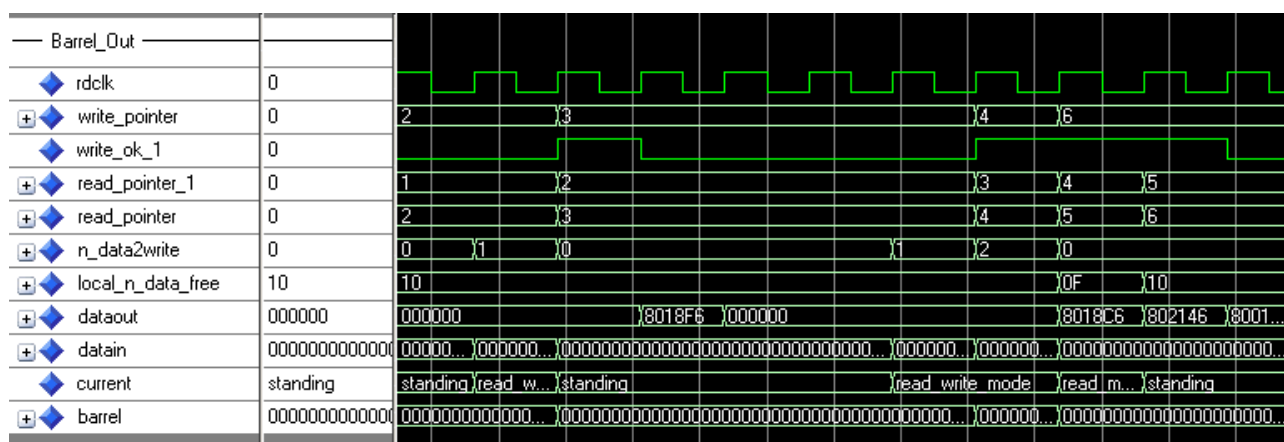


Fig. 11 shows the circuit simulation. The **write_pointer** increases by the **N_Data2Write** -1. This is why one data is always sent to output thus, if for example 3 data are written, two are stored and one goes right to the output port. The **read_pointer** is decreases by 1 at any **RDclk** cycle till it reaches 0. When the readout logic is in a stand_by situation, the **BarrelOut** is gradually emptied by 16 **RDclk** periods.

After the **N_Data2Write** is detected as different to 0 at the **RDclk** rising edge, the **N_Data_Free** and **DataOut** ports are updated 2 periods later.

6. Latch Enable Block

The figure shows the I/O ports of the LatchEnable circuits.

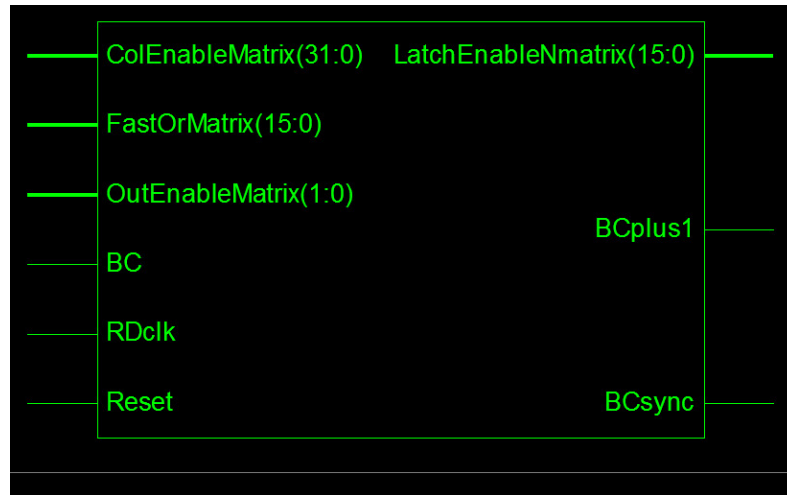


Figure 12: LatchEnable Block

Here follows a description of the I/O ports:

- **ColEnableMatrix** is a 32-bit port that reads the relative **ColEnable** port of the **MC_Address_Decoder** block,
- **FastOrMatrix** is a 16-bit port that reads the relative **FastOrMatrix** output of the **Matrix**, after being masked depending on the internal 16 Mask registers,
- **BC** is the external Bunch-Xing signal described in the **Readout Block**,
- **RDclk** is the external clock described in the **Readout Block**,
- **Reset** is the external reset described in the **Readout Block**,
- **LatchEnableNmatrix** is a 16-bit output port that freezes those MPs which have **FastOr** high, once the **BC** rising edge is detected. The port is also masked depending on the **Mask** register,
- **BCsync** is the Bunch-Xing signal **BC** after being synchronized to the global read clock **RDclk**,
- **Bcplus1** is the same as **BCsync** after being synchronized and delayed one clock cycle **RDclk**. Together with **BCsync** is used to detect the rising edge of **BC**.

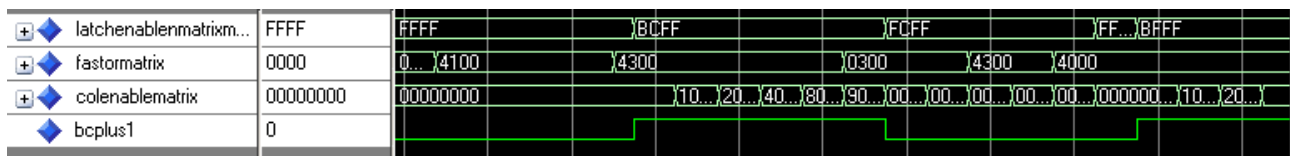


Figure 13: Simulation of the LatchEnable Block

6.1 Latency

After the **Bcplus1** rising edge occurs, the **BCsync** is updated at the first **RDclk** rising edge. **BCplus1** and **LatchEnableNmatrix** ports are rather updated 2 periods later. The latency is 2 periods.

7. MC Address Decoder Block

The figure shows the I/O ports of the **MC_Address_Decoder** circuits. This provides the address of the column of pixels while the matrix readout is ongoing. It stops only over the MCs that have at least one hit. The readout of a MC lasts 5 **RDclk** periods, 4 to read out the columns inside the MC and 1 to reset the MPs just read. It can last more periods if the readout enters a **Stand_By** condition.

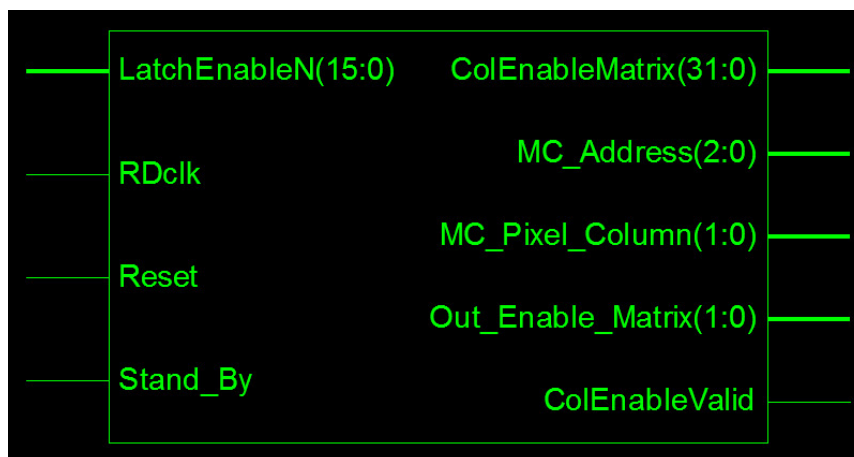


Figure 14: MC_Address_Decoder Block

Here follows a description of the I/O ports:

- **LatchEnableN** is a 16-bit port that reports the freeze status of those MPs which have **FastOr** high, once the **BC** rising edge is detected. The port is also masked depending by the internal 16 Mask register,
- **RDclk** is the external clock described in the **Readout Block**,
- **Reset** is the external reset described in the **Readout Block**,
- **Stand_By** is signal to freeze the scan of the matrix,
- **ColEnableMatrix** is a 32-bit port that enables the scan of the matrix. For each MC, it assumes the following states in the following order: “0001”-“0010”-“0100”-“1000”-“1001”. The first 4 states enable the reading of the single pixel columns of the MC, from right to left. The last state (“1001”) is the reset of the MPs, which belong to that MC, that have been previously frozen,
- **MC_Address** is a 3-bit port that holds the address code of the MC, ranging from 7 to 0 as the MC address moves from left to right,
- **MC_Pixel_Column** is a 2-bit port that holds the address code of the column of pixels inside the active MC, ranging from 3 to 0 as the pixel column address moves from left to right,
- **Out_Enable_Matrix** is a 2-bit port that enables either the readout of a given MP or its reset phase. It is high when the **ColEnableMatrix** has one subsection of 4 bits at one of the following values: “0001”-“0010”-“0100”-“1000”-“1001”,
- **ColEnable_Valid** is a signal that indicates if the readout of a given MP is ongoing or not. It is high when **ColEnableMatrix** is “0001”-“0010”-“0100”; it is low wherever else,

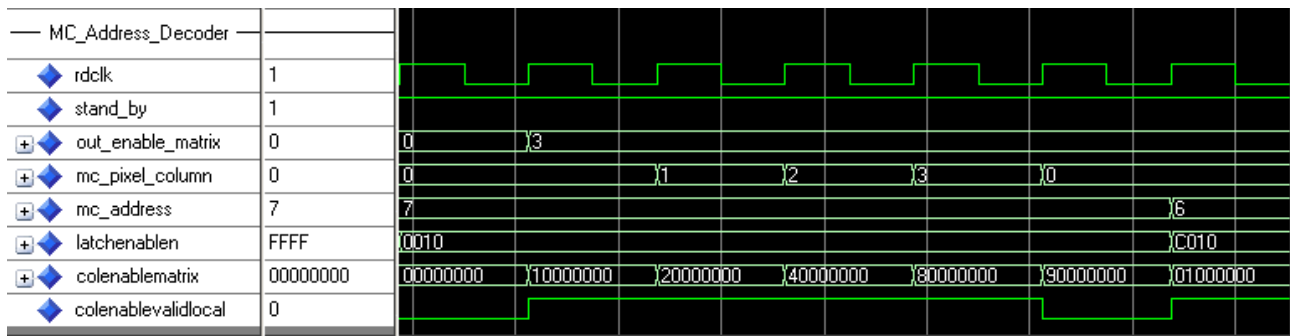


Figure 15: Simulation of the MC_Addres_Decoder Block

7.1 Latency

After the **LatchEnable** is updated at the next **RDclk** rising edge the address ports is also updated. The latency is just 1 period.

8. Dummy Matrix Block

The figure shows the I/O ports of the **Dummy Matrix** circuits.

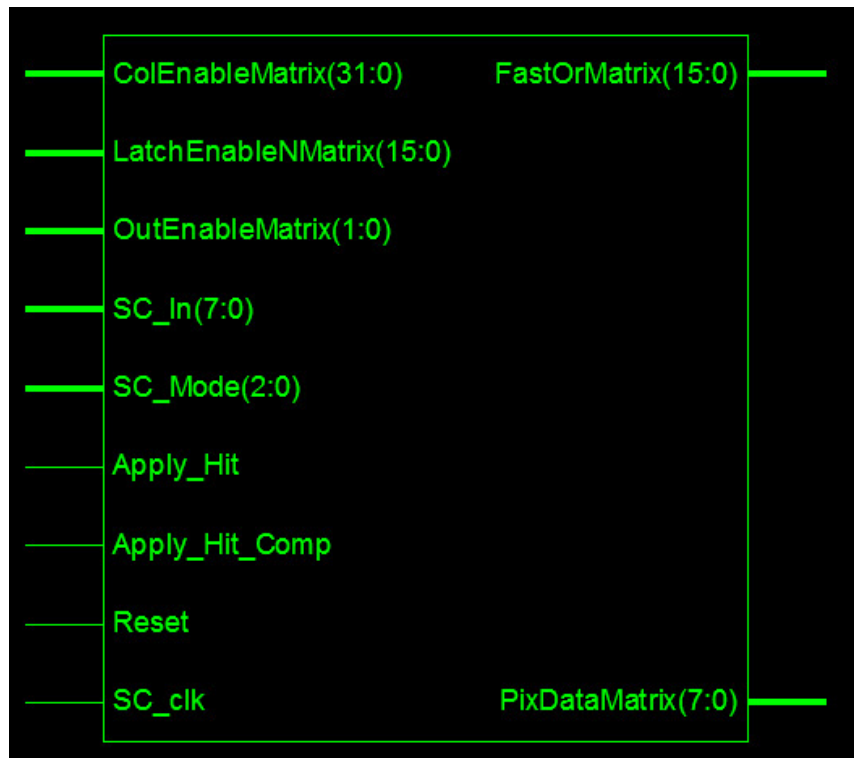


Figure 16: Dummy Matrix Block

Here follows a description of the I/O ports:

- **ColEnableMatrix** is a 32-bit port that reads the relative **ColEnable** port of the **MC_Address_Decoder** block,
- **LatchEnableNMatrix** is a 16-bit port that reads the relative **LatchEnableN** port of the **Latch Enable** block,
- **Out_Enable_Matrix** is the 2-bit port that reads the relative **Out_Enable_Matrix** port of the **MC_Address_Decoder** block,
- **SC_In** is the 8-bit external slow-control port described in the **Readout Block**,
- **SC_Mode** is the 3-bit external slow-control port described in the **Readout Block**,
- **SC_clk** is the external slow-control port described in the **Readout Block**,
- **Apply_Hit/Apply_Hit_Comp** are the external ports described in the **Readout Block**,
- **SC_clk** is the external clock described in the **Readout Block**,
- **Reset** is the external reset described in the **Readout Block**,
- **FastOrMatrix** is a 16-bit port that provides the *or* signals of the 16 MPs. If one of these is '1' it means that at least one pixel of the MP has a hit. The port is reset back to '0', MP by MP, depending on the **LatchEnableNMatrix** bits,
- **PixDataMatrix** is a 8-bit port that provides the hit configuration of a single column of the whole matrix. If both MPs of the selected MC are to be read out, this port can send up to 8 hits at a time. If one of the two MPs is not to be read out, the corresponding pins of the port are masked to '0' as there were no hits.

The custom-designed matrix has the same behavior of the **Dummy-Matrix**, except that it is not synchronized with **SC_clk**, is not reset with **Reset** port and does not see **Apply_Hit/Apply_Hit_Comp** couple nor the slow-control ports. It only shares the

LatchEnableNMatrix, the **ColEnableMatrix**, the **Out_Enable_Matrix** and provides the corresponding **PixDataMatrix** and **FastOrMatrix** output ports.

The **Dummy-Matrix** implementation on the silicon shares the same area as the readout circuit while the custom-designed matrix of MAPS occupies its own silicon area as it is the real sensor of the whole device.

8.1 Latency

After the input ports are updated on the rising edge of **SC_clk**, the **PixDataMatrix** and **FastOrMatrix** output ports are updated asynchronously. There is no extra latency.

9. Sparsifier Block

The figure shows the I/O ports of the **Sparsifier** circuits.

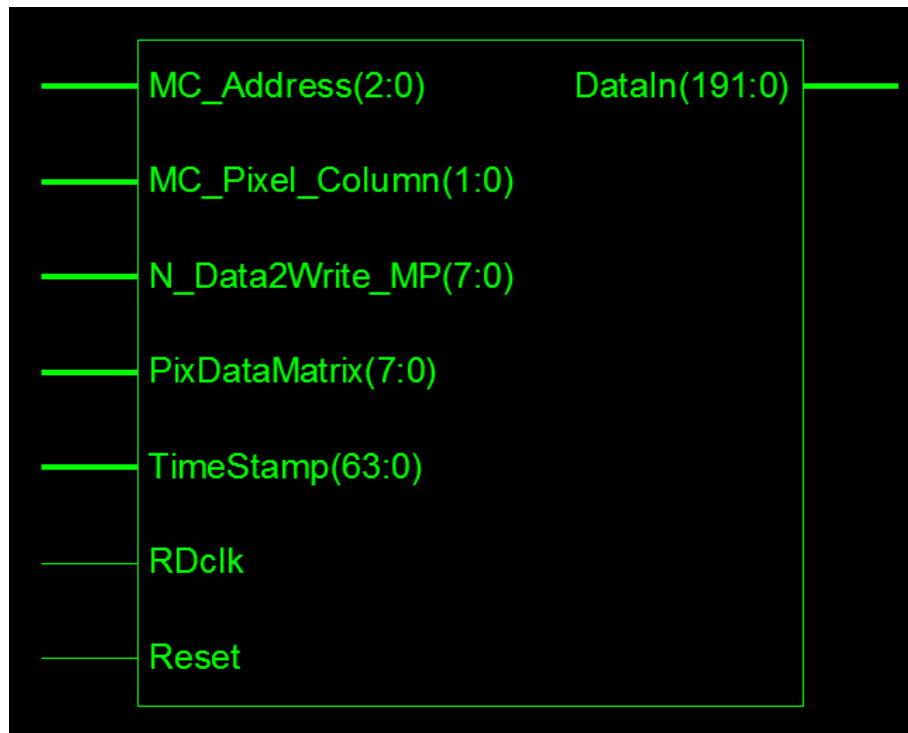


Figure 17: Sparsifier Block

Here follows a description of the I/O ports:

- **MC_Address** is a 3-bit port that reads the relative port of the **MC_Address_Decoder** block,
- **MC_Pixel_Columnn** is a 2-bit port that reads the relative port of the **MC_Address_Decoder** block,
- **N_DATA2Write_MP** is an 8-bit port that indicates how many hits have to be considered at the same time as they all belong to the same column of pixels. The information of these hits are then sent to the **BarrelOut** block through the **DataIn** output
- **PixDataMatrix** is the 8-bit port that specifies the hit configuration of the column of pixels.
- **TimeStamp** is a 64-bit port that reads the relative port of the **TimeStamp** block,
- **RDclk** is the external clock described in the **Readout Block**,
- **Reset** is the external reset described in the **Readout Block**,
- **DataIn** is the output 192-bit port that combines the information of the hits with the **time-stamp** into an up to 192 bits. In the majority of the cases the bits are redundant but, in case 8 hits are read within the same column of pixels, after being added with the 4-bit **time-stamp** each, they all are used. It should be said that both the **DataIn** port is dimensioned for 24-bit words while, at the moment, only 16 bits are used per words. This is in order to face future bigger matrixes.

9.1 Latency

After the input ports are updated, two **RDclk** cycles later the **DataIn** port is also updated. The latency is 2 periods.

10. Slow Control Block

The figure shows the I/O ports of the **Slow-Control** circuits.

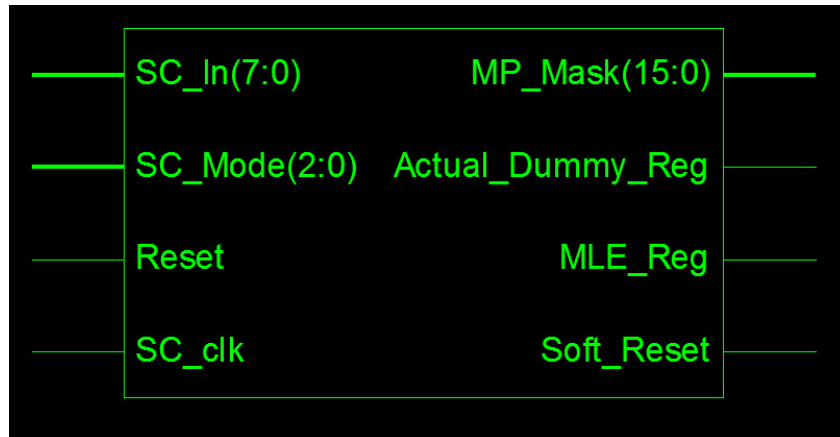


Figure 18: Slow-Control Block

Here follows a description of the I/O ports:

- **SC_In** is the 8-bit external slow-control port described in the **Readout Block**,
- **SC_Mode** is the 3-bit external slow-control port described in the **Readout Block**,
- **Reset** is the external reset described in the **Readout Block**,
- **SC_clk** is the external slow-control port described in the **Readout Block**,
- **MP_Mask** is a 16-bit port used to mask the 16 MPs. This port masks the **LatchEnableNMatrix** and the **FastOrMatrix** of the **Readout Block**. The masks are loaded with **SC_Mode**="001" in two **SC_clk** periods. The default masks at **Reset**='1' are all-at-'1',
- **Actual_Dummy_reg** is 1-bit register that indicated the operating mode: 0 for **custom-mode** and 1 for **digital-mode**. It is loaded with **SC_Mode**="010", by copying **SC_In(1)**, in one **SC_clk** period. The default is 1 for **digital-mode** at **Reset**='1',
- **Actual_Dummy_reg** is not used,
- **Soft_Reset** is a signal used to reset only the **time-stamp** in the **TimeStamp Block**. It is requested with **SC_Mode**="100" in one **SC_clk** period. After that, the **SC_Mode** has to be set again to "111" to normal running operation. It is suggested to not force a **Soft_Reset** during the scan of valid hits in order to avoid unpredictable output data.

10.1 Latency

After the input ports are updated, one **SC_clk** cycle later the output ports are also updated. The latency is 1 period.

11. DEBUG

Test, BC and Output Data saved as formatted text files for debugging purposes.

Time TEST_actual.txt only for custom operating mode
.....
Test Time = 1437.5ns
Test Time = 1543.75ns
Test Time = 1650ns
Test Time = 1756.25ns
Test Time = 1862.5ns
Test Time = 1968.75ns
Test Time = 2075ns
.....

Time BC_dummy.txt & Time BC_actual.txt for digital and custom operating modes
.....
BC Time = 8550ns
BC Time = 9300ns
BC Time = 10050ns
BC Time = 10950ns
BC Time = 11700ns
BC Time = 12450ns
BC Time = 13350ns
.....

Time_DataOut_dummy.txt & Time_DataOut_actual.txt for digital and custom operating modes			
Clock Time	Binary Format Out	Hexadecimal Format Out	Decimal Format Out
.....			
RDclk Time = 7637.5ns	Bin = 010110001001	Hex = 1589	Dec = 5513
RDclk Time = 7662.5ns	Bin = 001110001001	Hex = 1389	Dec = 5001
RDclk Time = 7687.5ns	Bin = 000110001001	Hex = 1189	Dec = 4489
RDclk Time = 7787.5ns	Bin = 101011111010	Hex = 1AFA	Dec = 6906
RDclk Time = 7812.5ns	Bin = 111101111010	Hex = 1F7A	Dec = 8058
RDclk Time = 7887.5ns	Bin = 000001100000	Hex = 1060	Dec = 4192
RDclk Time = 8012.5ns	Bin = 110001011110	Hex = 1C5E	Dec = 7262
RDclk Time = 8037.5ns	Bin = 010011011100	Hex = 14DC	Dec = 5340
RDclk Time = 137.5ns	Bin = 111000110001	Hex = 1E31	Dec = 7729
RDclk Time = 8162.5ns	Bin = 000000110001	Hex = 1031	Dec = 4145
RDclk Time = 8187.5ns	Bin = 000010111111	Hex = 10BF	Dec = 4287
RDclk Time = 8287.5ns	Bin = 001010010001	Hex = 1291	Dec = 4753
RDclk Time = 8312.5ns	Bin = 011100010001	Hex = 1711	Dec = 5905
RDclk Time = 8387.5ns	Bin = 111000000010	Hex = 1E02	Dec = 7682
RDclk Time = 8412.5ns	Bin = 110000000010	Hex = 1C02	Dec = 7170
RDclk Time = 8437.5ns	Bin = 101000000010	Hex = 1A02	Dec = 6658
RDclk Time = 8462.5ns	Bin = 100000000010	Hex = 1802	Dec = 6146
.....			

12. APSEL3D Layout

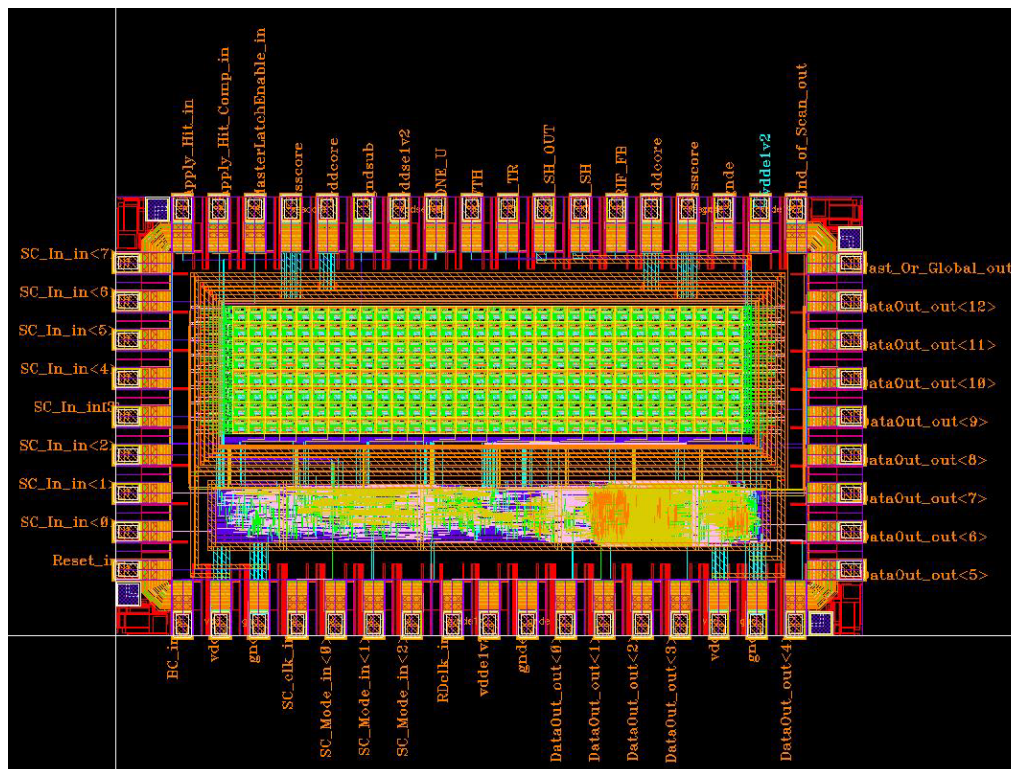


Figure 19: ApSel3D Layout

This is the picture of APSEL3D ASIC, designed with STM 130nm 6M Technology

The whole layout dimension is: **2343.56 μ m $\times$ 1379.24 μ m**

The pitch of the pads is: **120.54 μ m** on the Left, Bottom and Right sides

The pitch of the pads is: **114.16 μ m** on the Top side

13. APSEL3D Pinout

APSEL3D			
9 Left Pads (top-2-bottom)	9 Right Pads (top-2-bottom)	18 Top Pads (left-2-right)	17 Bottom Pads (left-2-right)
SC_IN(7)	Fast Or Global	Apply_Hit	BC
SC_IN(6)	Data_Out(12)	Apply_Hit_Comp	VDD_CORE_0
SC_IN(5)	Data_Out(11)	MasterLatchEnable	VSS_CORE_0
SC_IN(4)	Data_Out(10)	VSS_Core	SC_clk
SC_IN(3)	Data_Out(9)	VDD_Core	SC_Mode(2)
SC_IN(2)	Data_Out(8)	GNDSUB	SC_Mode(1)
SC_IN(1)	Data_Out(7)	VDDselv2	SC_Mode(0)
SC_IN(0)	Data_Out (6)	ONE_U	RDclk
Reset	Data_Out(5)	VTH	VDD_Pery_1
		TR	DataOut(3)
		SH_out	DataOut(2)
		SH	DataOut(1)
		RTF_FB	DataOut(0)
		VDD_Core	VDD_CORE_1
		VSS_Core	VSS_CORE_1
		GNDe	DataOut(4)
		VDDelv2	
		End_of_Scan	

14. Test Results

The APSEL3D prototype chips were tested in the laboratory of INFN Pisa on November-December 2007 (F. Morsani, G. Rizzo and S. Bettarini that are with INFN and Physics Department of Pisa University). It can be stated that the chip works basically as it was designed even though some design-bugs have been found.

Most important things are:

The chip works properly if the stand-by mode is not in action. In other words, if the average hit input-rate is under the 40Mhit/s throughput (output) rate the readout can follow the hit creation and the correct information is sent to output.

Basically, the chip works fine even if the input-rate is slightly greater than 40Mhit/s throughput. Some tests that were performed with 5 hits per MP, that is an over 30% occupancy of the matrix, show that the stand-by mode is in action and the readout logic does not lose any hit.

Conversely, if the occupancy is very high, say over 50%, some hits are lost and never come out. This particularly applies for the left-hand front of the hits. For example, if the matrix is completely “lit”, the two left-most columns of pixels are lost (16 pixels). This bug is due to the queuing system and not due to the sparsification logic.

Another bug that was found occurs when a given MP is frozen and, the other MP that belongs to the same MC is to be frozen before the first one is read. In this case the hits that belong to both MPs are read with the same time-stamp, independently of their actual time-stamp values. Thus, only the hits that belong to one MP have the right time-stamp, while those that belong the other MP do have not. It can be said that this situation occurs when a high occupancy is present over the matrix so that the BC signal results fast with respect to the readout rate. In the “normal” rate situation described above the readout logic is sufficiently fast to do not let this case occur.

As a summary, it can be said that the chip works properly whenever the input rate is in the range of the expected values, say some tens of MHz/cm². In this region the readout logic can follow the hits and all the information is properly read. If the logic is over-stressed, from the occupancy viewpoint, some hits are lost due to design-bugs into the queuing system.

15. INDEX

Index of Chapters:

1. Notes for the readers	2
1.1 Version	2
2. Introduction	3
2.1 The matrix organization	4
2.2 Hierarchy	4
3. Readout Block	5
3.1 Configuration Steps	8
3.2 Internal Registers	9
3.3 Stand_By	9
3.4 Latency	9
3.5 Simulations	9
4. Time Stamp Block	12
4.1 Latency	13
5. Barrel Out Block	14
5.1 Latency	15
6. Latch Enable Block	16
6.1 Latency	16
7. MC Address Decoder Block	17
7.1 Latency	18
8. Dummy Matrix Block	19
8.1 Latency	20
9. Sparsifier Block	21
9.1 Latency	21
10. Slow Control Block	22
10.1 Latency	22
11. DEBUG	23
12. APSEL3D Layout	24
13. APSEL3D Pinout	25
14. Test Results	26
Index of Chapters:	27
Index of Figures:	28

Index of Figures:

Figure 1: Apsel3D operating modes: custom-mode and digital-mode	3
Figure 2: Readout Block	5
Figure 3: Simulation of the APSEL3D configuration in custom-mode.....	8
Figure 4: Simulation of the APSEL3D configuration in digital-mode	8
Figure 5: Simulation of the entire circuit in the custom-mode	10
Figure 6: Simulation of the entire circuit in the digital-mode.....	11
Figure 7: TimeStamp Block.....	12
Figure 8: Simulation of the TimeStamp Block	13
Figure 9: Simulation of the TimeStamp Block	13
Figure 10: BarrelOut Block.....	14
Figure 11: Simulation of the BarrelOut Block.....	15
Figure 12: LatchEnable Block	16
Figure 13: Simulation of the LatchEnable Block.....	16
Figure 14: MC_Address_Decoder Block.....	17
Figure 15: Simulation of the MC_Addres_Decoder Block	18
Figure 16: Dummy Matrix Block.....	19
Figure 17: Sparsifier Block	21
Figure 18: Slow-Control Block.....	22
Figure 19: Apsel3D Layout	24