MOST2 digital peripheral logic

Xiaomin WEI

2019-4-30

Content

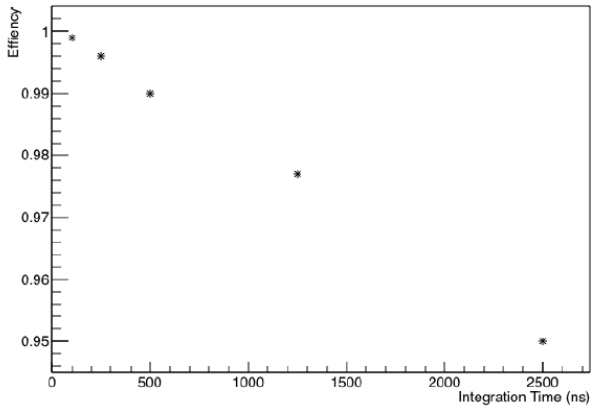
- Design requirements
- Readout architecture
- Interface & function
- Verification
- Layout
- Summary

Design requirements

Parameter	Unit	Higgs	W	Z
Bunch spacing	**ns**	**680**	**210**	**25**
Hit density	**hits/bunch/cm²**	**2.5**	**2.5**	**0.2**
	hits/bunch	8.2	8.2	0.66
	pixels/bunch	25	25	2
Designed data rate (trigger)	**MHz/32bit**	**5**	**5**	**5**
Designed data rate (triggerless)	**MHz/32bit**	**160**	**160**	**160**

Efficiency

- How long is the acceptable dead time for the pixel readout?
  - Assuming worse occupancy case (pixel) = 0.05%
  - 100ns : 99.9% ; 200ns : 99.6% ; 500ns: 99%



From Zhijun Liang

输入： 平均120MHz

输出： triggerless 160MHz

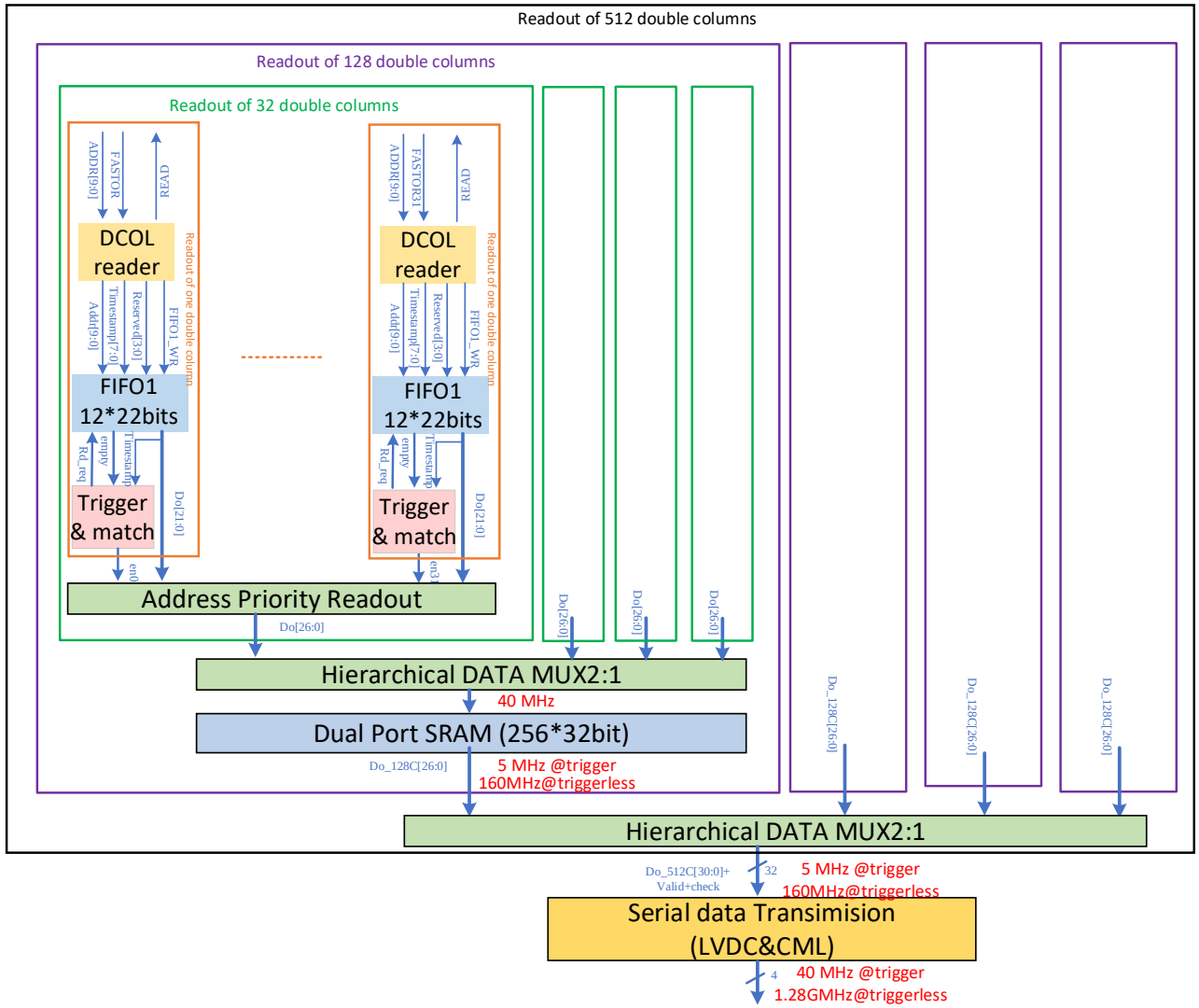
Trigger 5 MHz

Dead time: 双列 500ns

增加trigger的窗口（7LSB,175ns），Higgs和W的数据率是否会增加？

Parameter	Unit	Higgs	W	Z
Bunch spacing	ns	680	210	25
Hit density	hits/bunch/cm^2	2.5	2.5	0.2
	hits/bunch	8.2	8.2	0.66
	pixels/bunch	25	25	2
Hit pixel rate	MHz/cm^2	11	36	24
	MHz/chip	36	120	80
Chip data rate (triggerless)	Gbps	1.15	3.84	2.56
	MHz/32bit	36	120	80
Chip data rate (trigger, no error)	Mbps（Average）	40	40	3.2
	MHz/32bit	1.25	1.25	0.1
Chip data rate (trigger, 7LSB error)	Mbps （Average）	40	40	25.6
	MHz/32bit	1.25	1.25	0.8
Designed data rate (trigger)	MHz/32bit	5	5	5
Designed data rate (triggerless)	MHz/32bit	160	160	160

Readout architecture



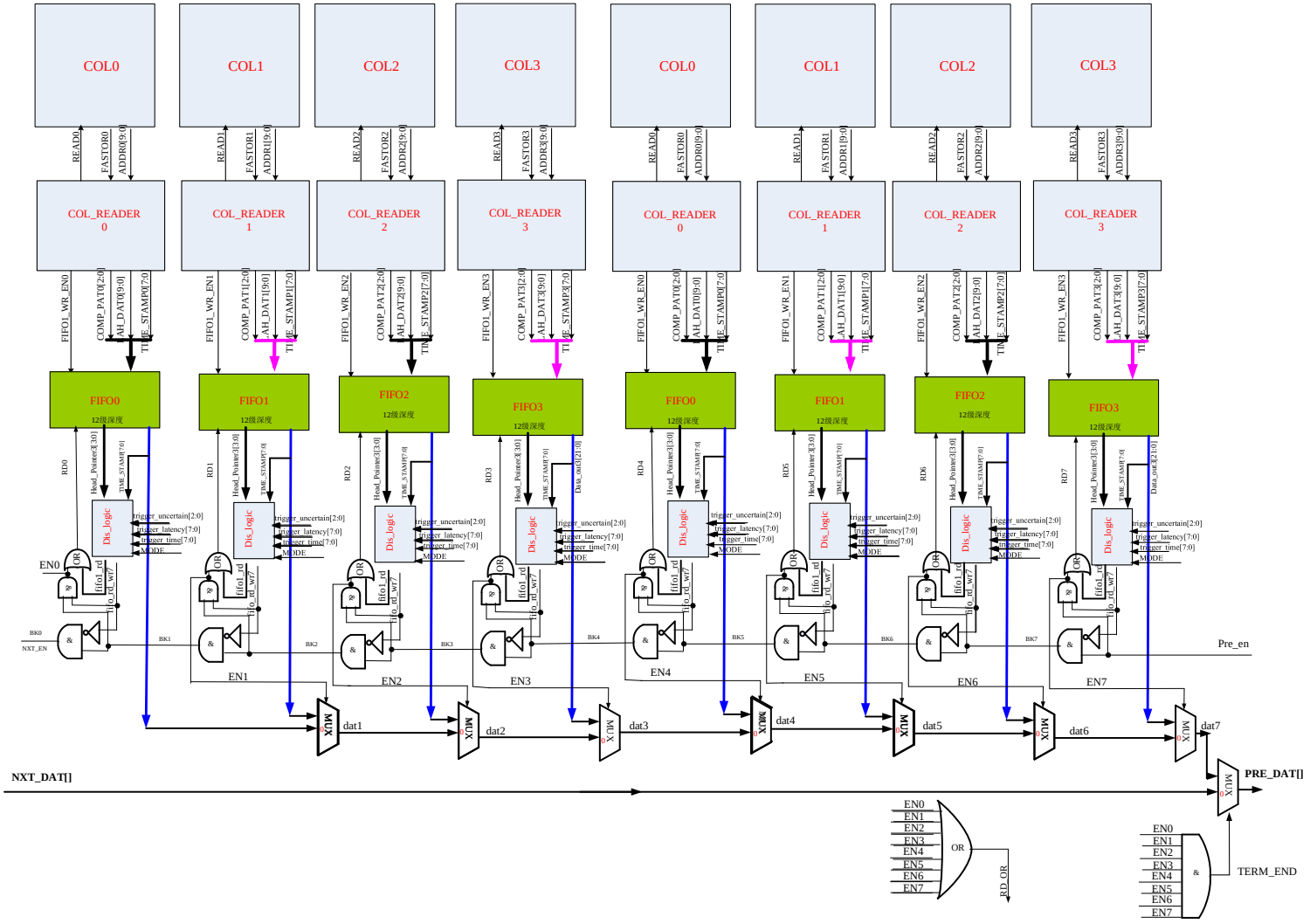
列电路：

- 列端记录timestamp
- Trigger模式，在FIFO1输出时丢弃不匹配数据，FIFO2只存储时间戳匹配的数据

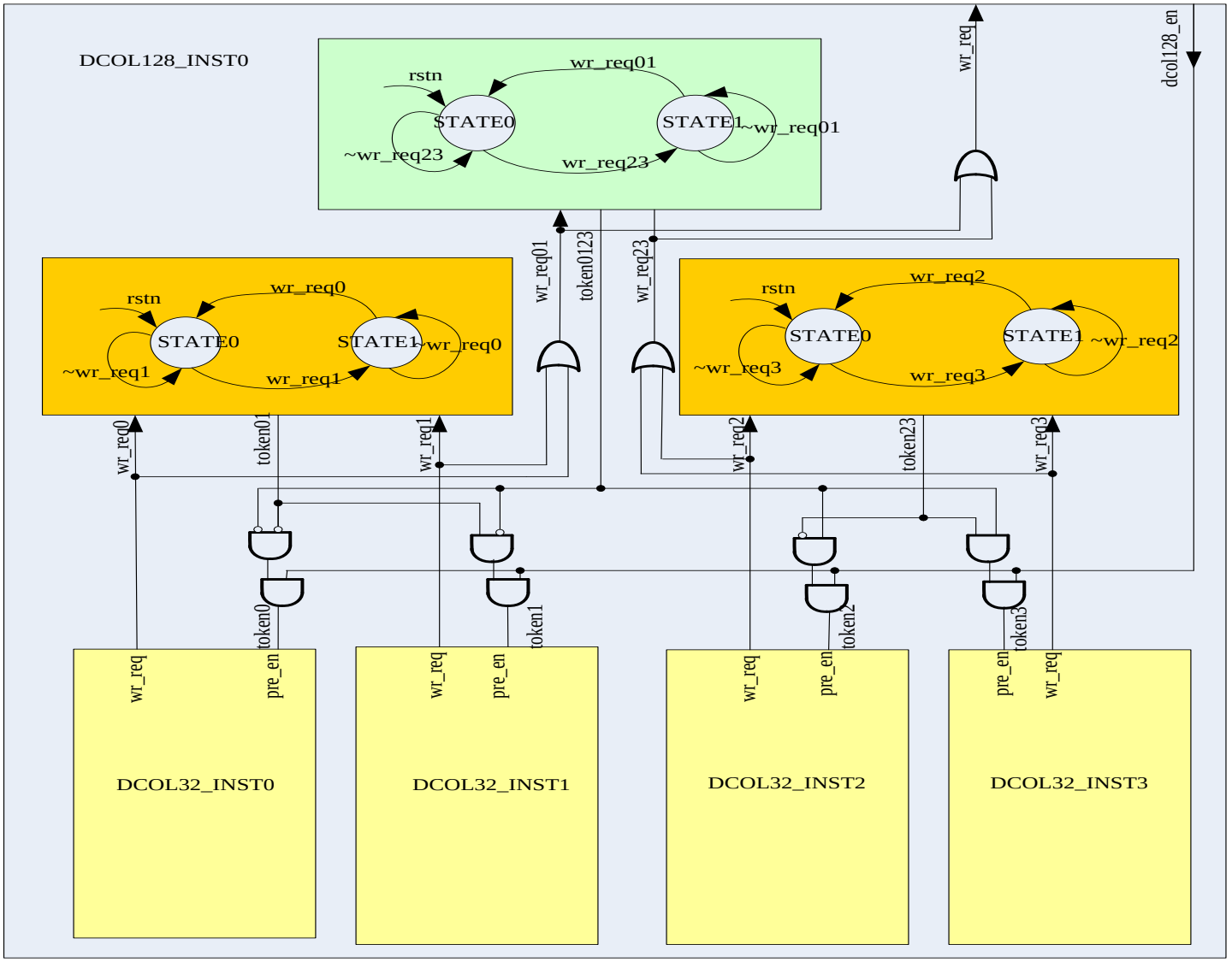
列并行读出方式：

- 512个双列并行读出以满足dead time 的要求，每列对应一个FIFO1
- 512分为4个128双列，每128对应一个FIFO2
- 128分为4个32双列，32双列内部采用数据驱动（分两级，8个一组），32双列之间轮询读出。轮询策略是每块读一个数就转读下一个块以避免数据拥塞。

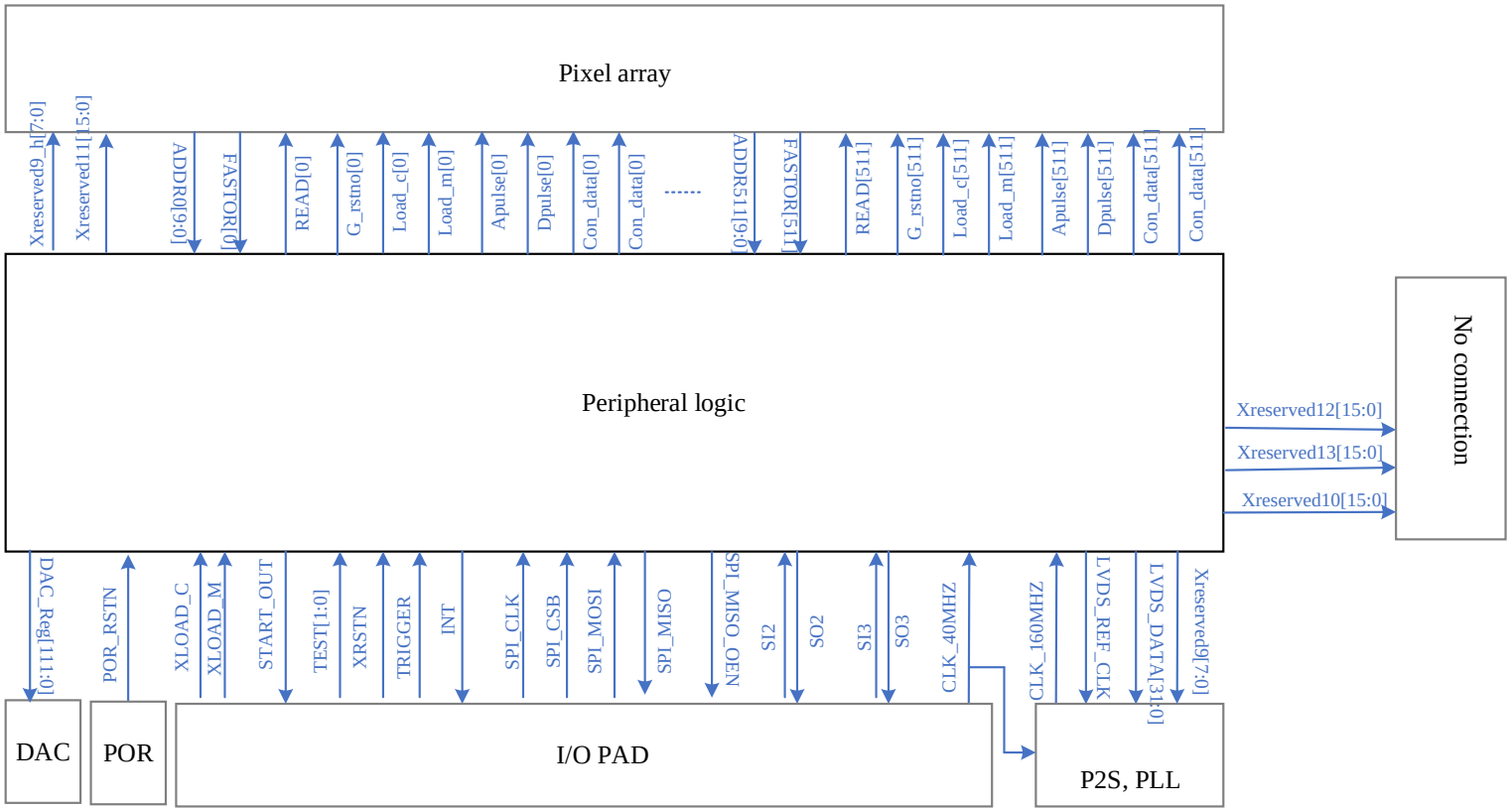
8列级优先级读出


COL0
COL1
COL2
COL3
COL_READER 0
COL_READER 1
COL_READER 2
COL_READER 3
FIFO0
FIFO1
FIFO2
FIFO3
12级深度
Dis_logic
MUX
dat1
dat2
dat3
dat4
dat5
dat6
dat7
Pre_en
NXT_DAT[]
PRE_DAT[]
TERM_END
EN0
EN1
EN2
EN3
EN4
EN5
EN6
EN7
OR

32列的轮循读出


DCOL128_INST0
rstn
wr_req01
STATE0
STATE1
~wr_req01
~wr_req23
wr_req23
wr_req
dcol128_en
wr_req01
token0123
wr_req23
rstn
wr_req0
STATE0
STATE1
~wr_req0
~wr_req1
wr_req1
rstn
wr_req2
STATE0
STATE1
~wr_req2
~wr_req3
wr_req3
wr_req0
token01
wr_req1
wr_req2
token23
wr_req3
token0
token1
token2
token3
wr_req
pre_en
DCOL32_INST0
DCOL32_INST1
DCOL32_INST2
DCOL32_INST3

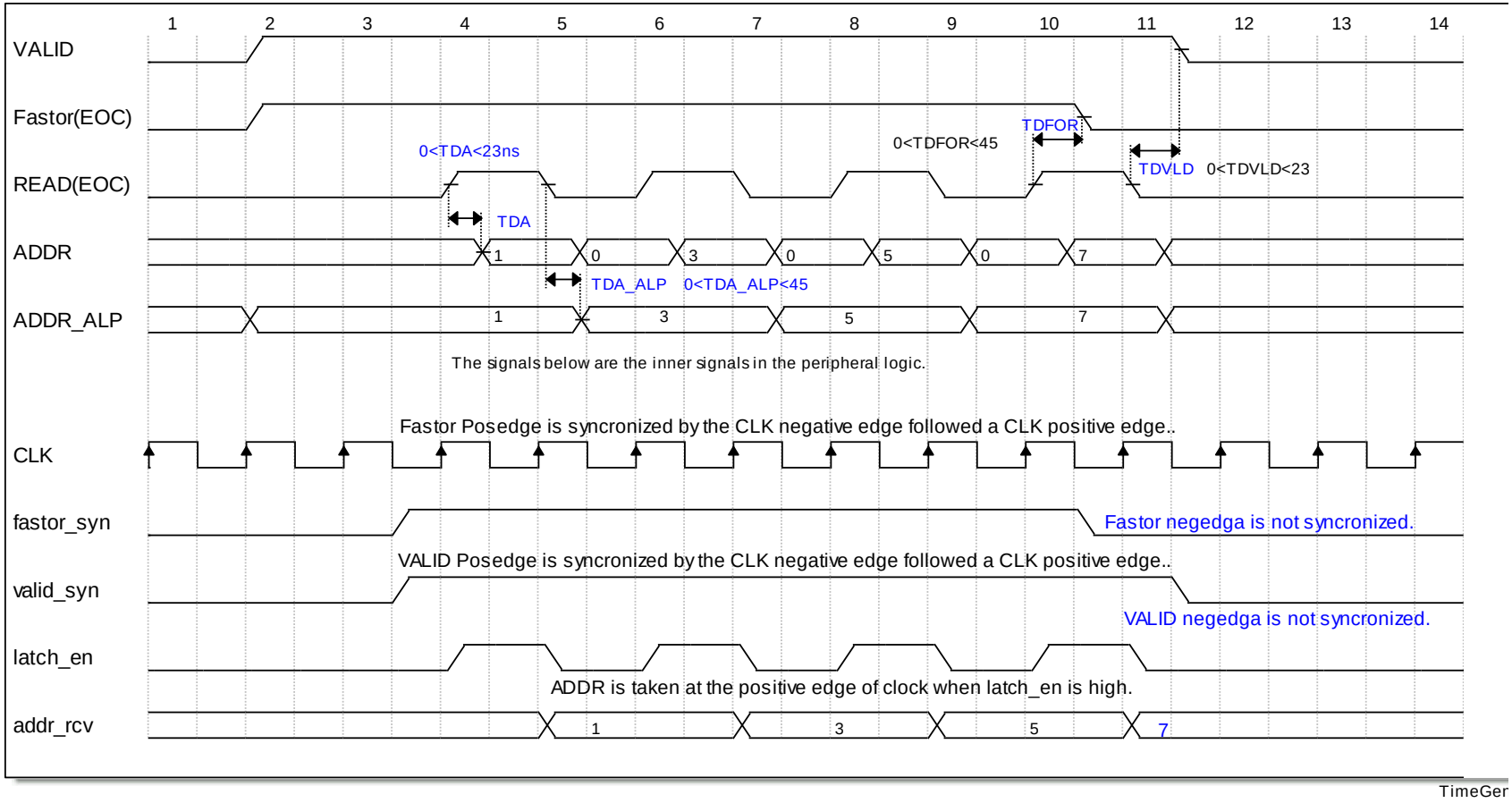
Interface & function



3个主要数据接口：像素阵列、SPI访问控制寄存器、数据输出至LVDS

3个功能设计：Trigger数据匹配、地址压缩、像素掩码

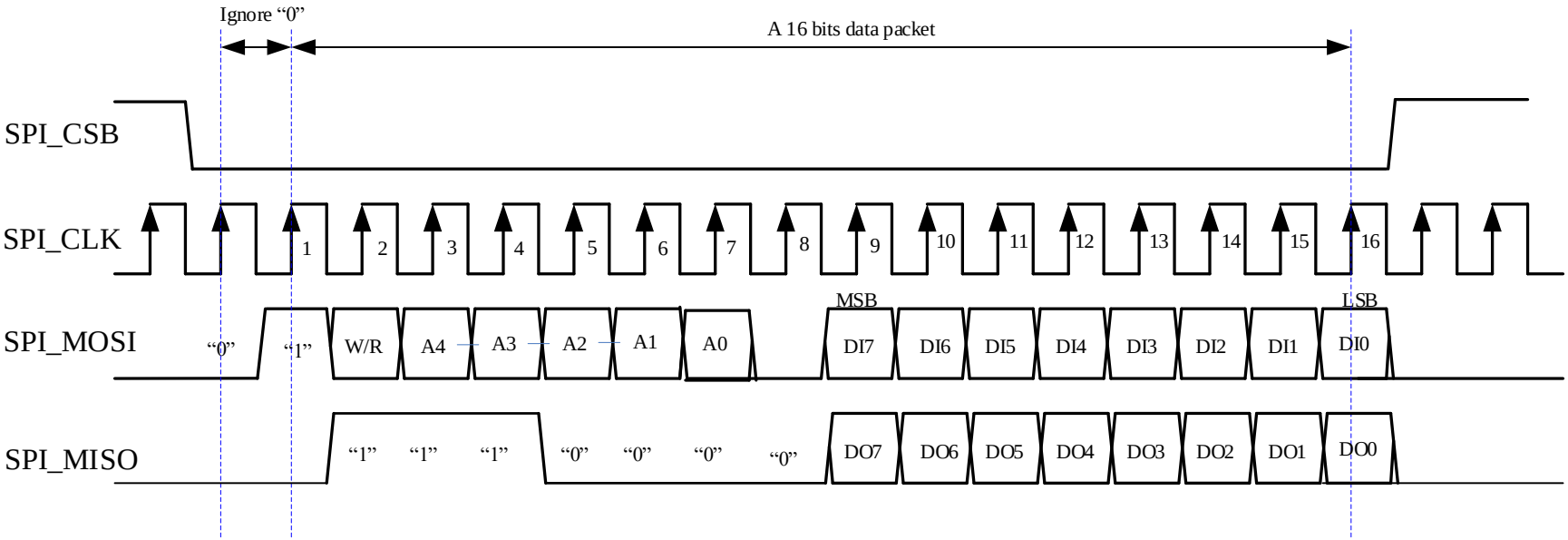
Interface 1 - Pixel array



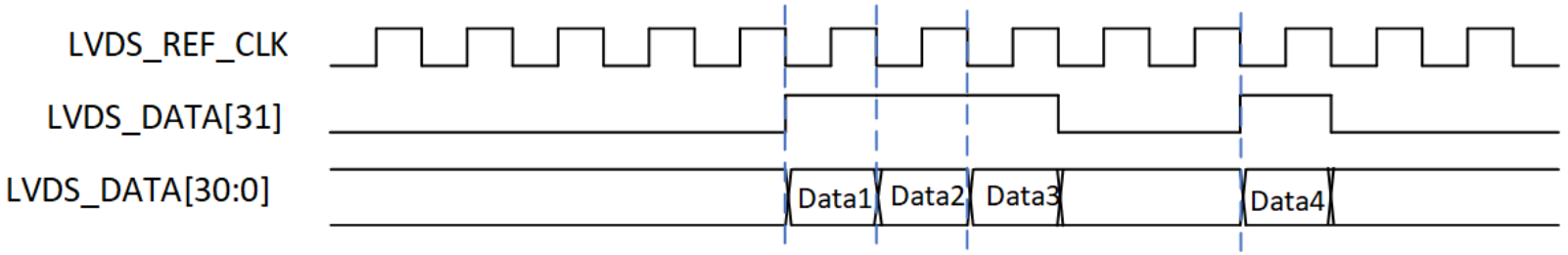
CLK cycle: 25ns READ cycle： 50ns 外围数字在READ下降沿读数

- ✓ FEI3版本： READ高电平产生有效地址 => TDA <23ns （大阵列这是极限值）
- ✓ ALPID版本：READ（SYNC）下降沿地址复位=> TDA_ALP < 45ns

Interface 2 - SPI



Interface 3 - LVDS

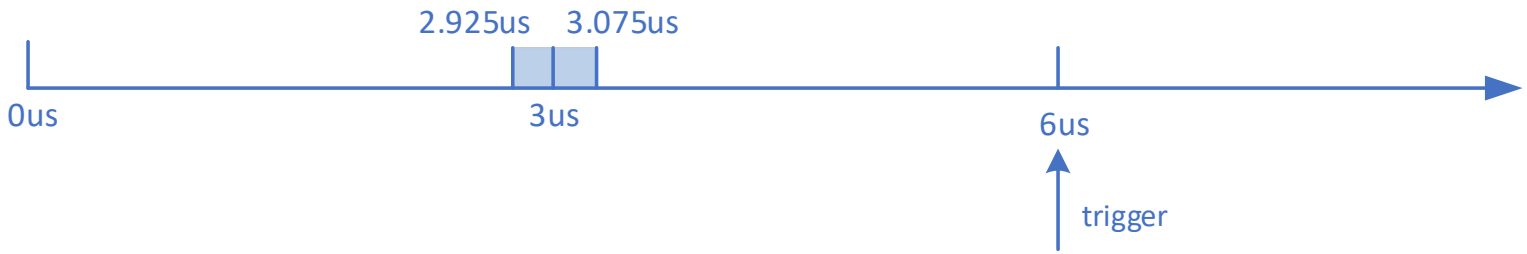


Function1- trigger mode data match

TRIGGER_LATENCY 8位寄存器：延时0-6us

=> FIFO1只存储6us的数据，超过6us的数据会丢弃。

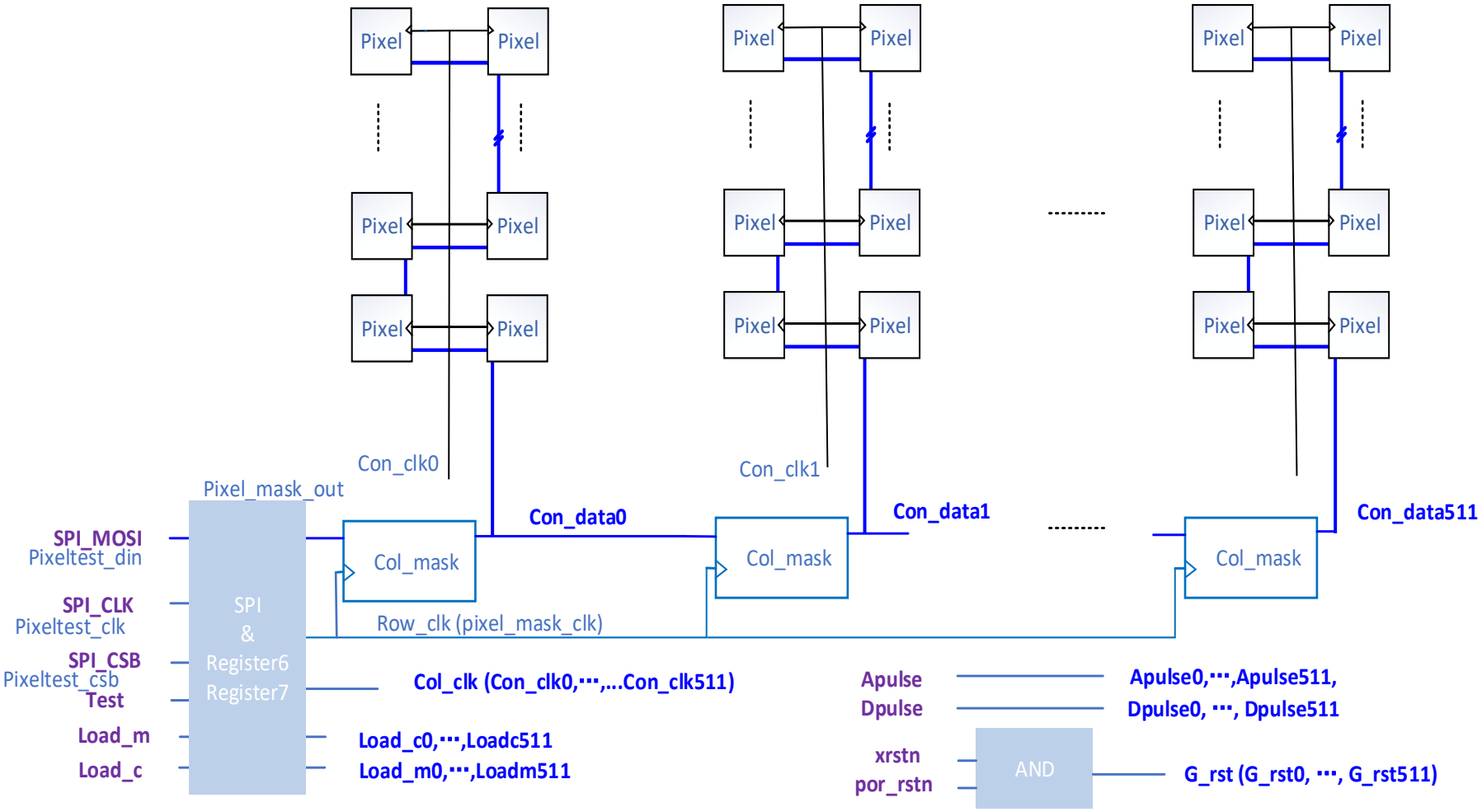
TRIGGER_UNCERTAIN 3位寄存器：窗口范围0-175ns



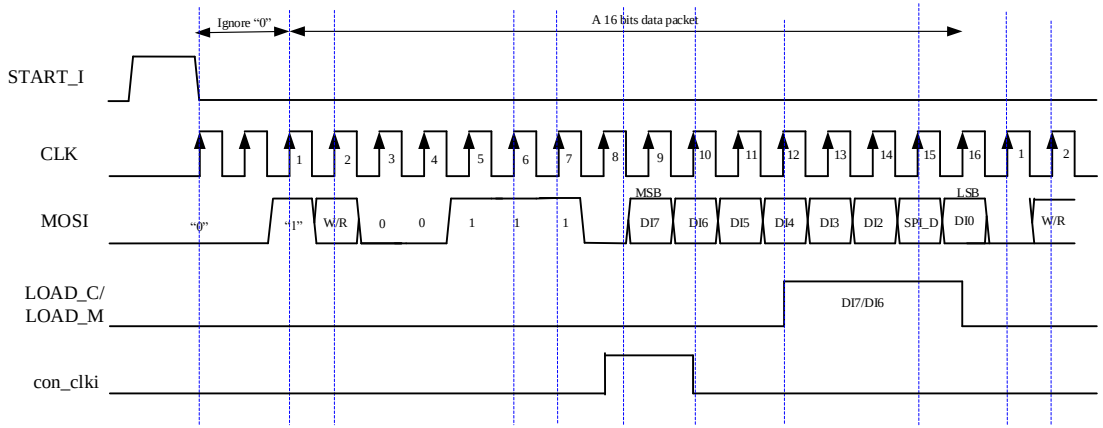
Example of setting trigger latency and trigger uncertain.

Suppose the trigger signal comes at 6us, and users wants to acquire the hits from 2.925us to 3.075us, then we should set TRIGGER_LATENCY as 8'd 123 (123*25ns=3.075us), and TRIGGER_UNCERTAIN as 3'b110. (0.15us).

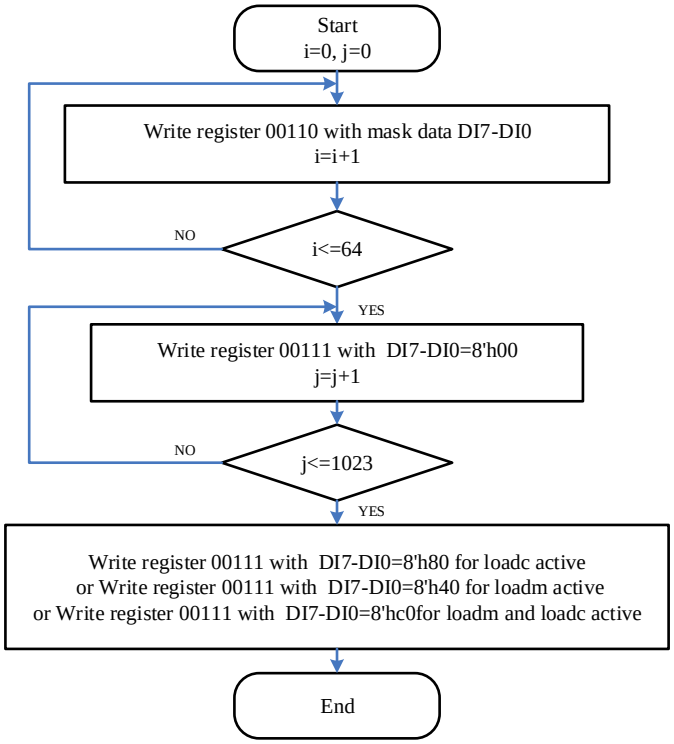
Function2-Pixel mask function (1)



Function2-Pixel mask function (2)



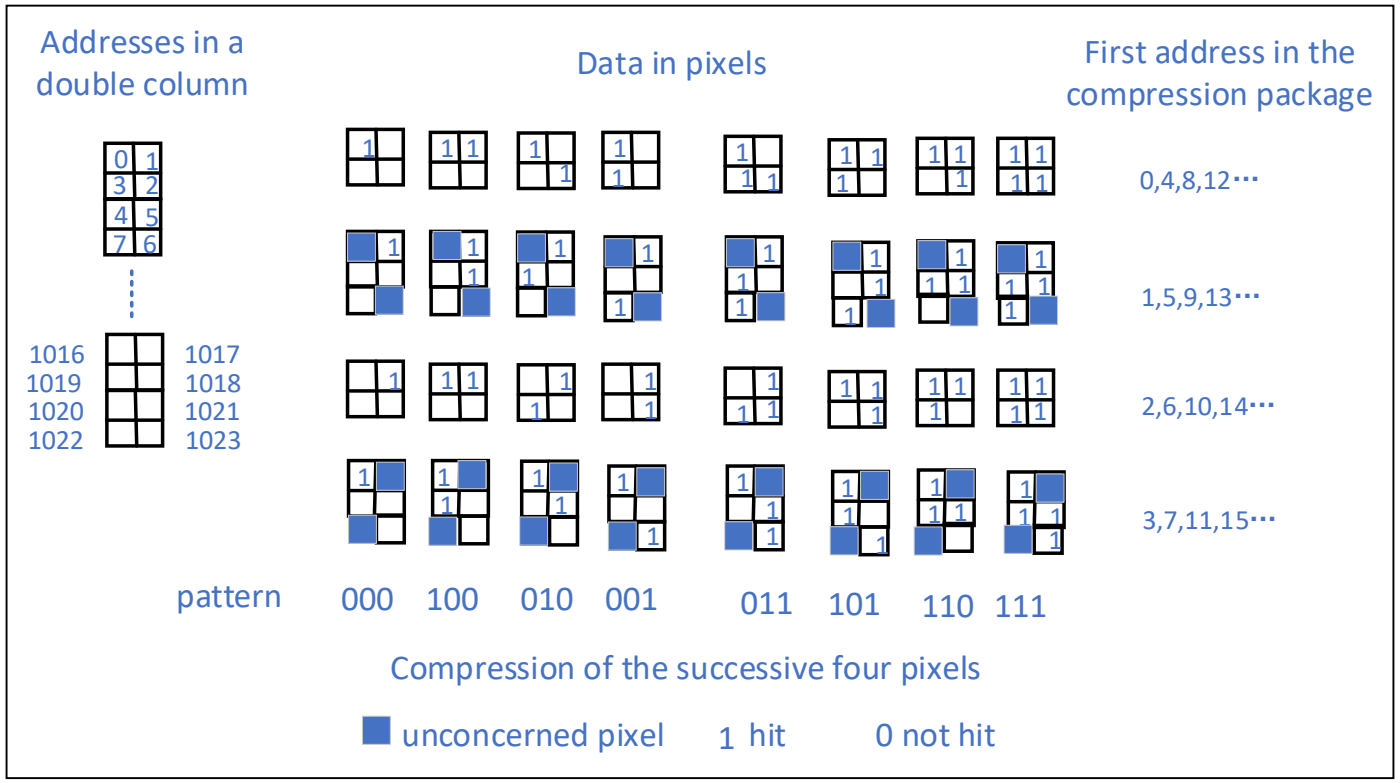
Internal timing sequence generation of
con_clk (2 spi_clk cycles), load_c/load_m (4 spi_clk cycles)



Setting flow for pixel mask function

- Setting the pixel mask by software
  - Easy to use
  - Test [1:0] can be fixed to 2'b00 when bonded on a ladder
  - External Loadc,loadm can be removed in the next version.

Founction3- data compression



Example of Data compression in peripheral logic

在FIF01之前将数据实时压缩

- 提高电路处理高数据率数据的能力
- 减小所需FIF01、FIF02的存储深度
- 减少所需输出的数据量

目前版本连续5个地址可压缩（32位数据有4位空闲），具体几个地址压缩比较合适需要统计实际粒子入射数据。

Verification

- Testbench

- Hit generation

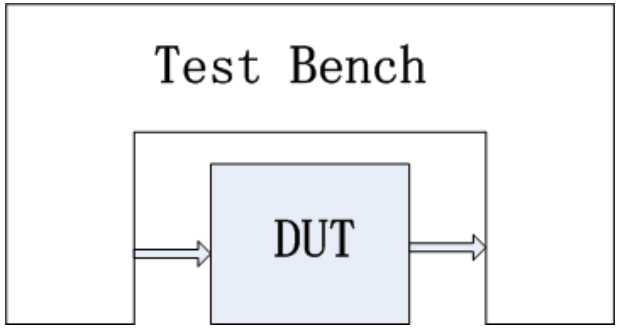
按照泊松分布随机产生某一中心频率的粒子数，均匀分散在每列。

- Pixel col signal

根据随机分配在每列的粒子数目产生fastor/valid。地址定义为可增序、降序、随机。

- Data analysis

对比发送和接受到的数据（含时间戳）。统计数据率120MHz及以下都可以正确读出。



Verification

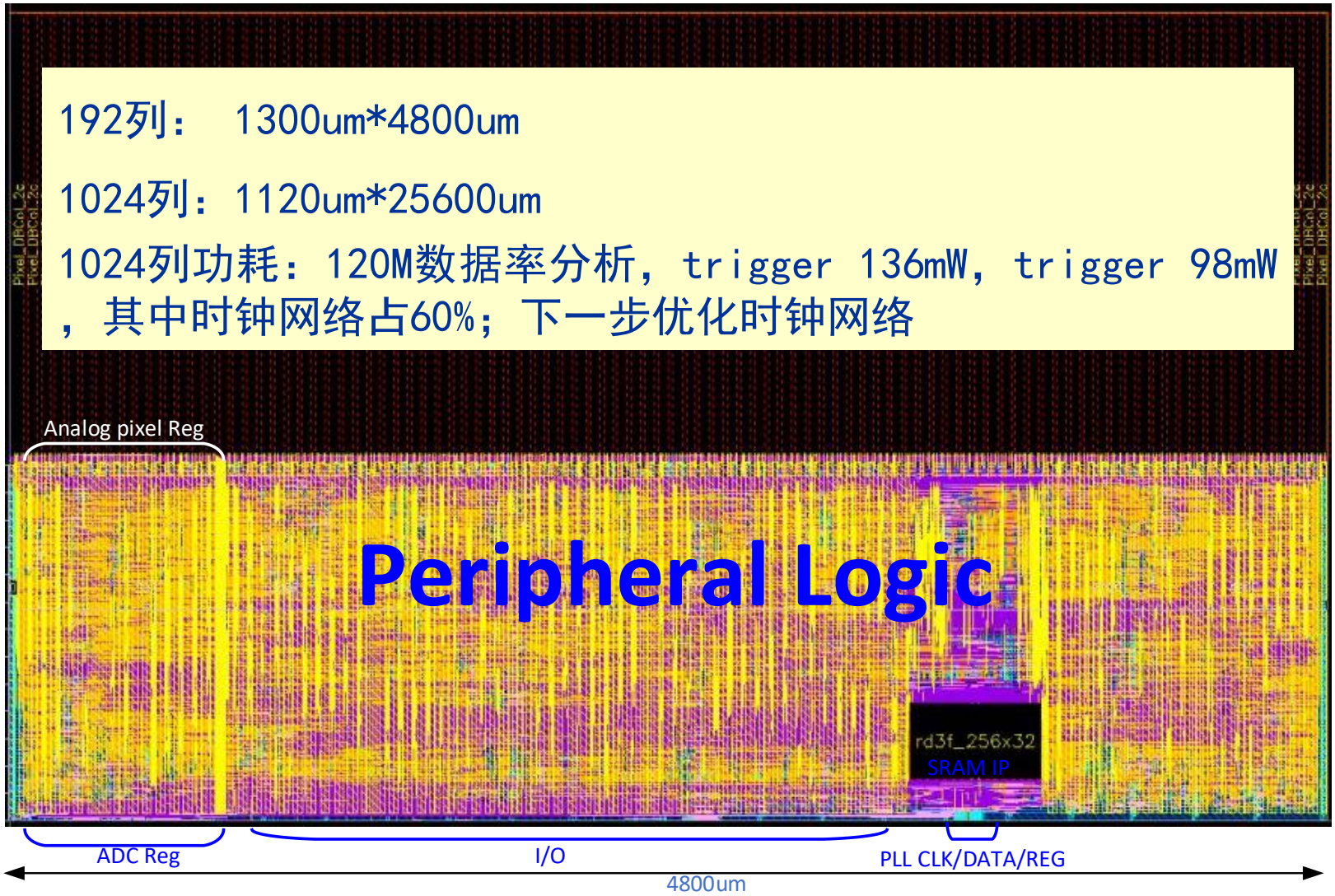
- 验证内容
  - SPI接口读写数据：复位后读寄存器检查复位状态、对寄存器逐个访问设置可能的值、pixel mask功能测试（写不同的mask pattern并发送内部loadc、loadm）
  - Apulse Dpulse 外部loadc loadm测试
  - 设置各种寄存器装态组合

    是否trigger（不同的trigger latency和trigger window设置）、是否数据压缩、地址（连续、不连续、增序、降序、随机增降）

Verification

- 仿真结果：
  - 基本功能全部正确。
  - Triggerless模式且不做数据压缩，仿真300ms，数据率120MHz-130MHz完全正确，数据率超过或接近160MHz有时会丢数（受接口160MHz限制）。
  - Trigger模式因数据量少可以支持更高的数据率。
  - 压缩模式可以支持更高数据率（提高程度依赖于地址可压缩的程度）。

Layout

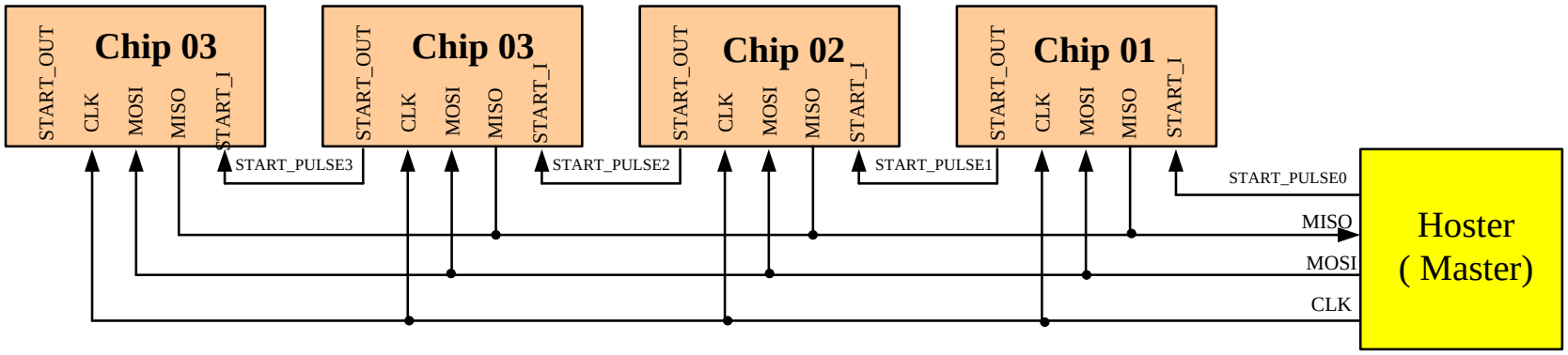

192列： 1300um*4800um
1024列：1120um*25600um
1024列功耗：120M数据率分析，trigger 136mW，trigger 98mW，其中时钟网络占60%；下一步优化时钟网络
Analog pixel Reg
Peripheral Logic
rd3f_256x32
SRAM IP
1300um
ADC Reg
I/O
PLL CLK/DATA/REG
4800um

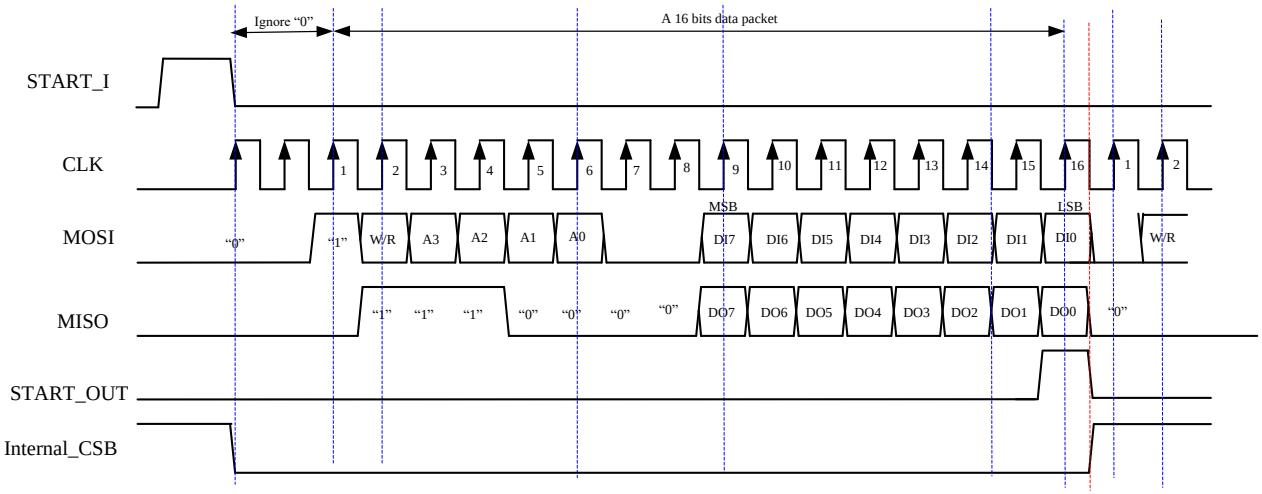
Summary

- 支持trigger、triggerless模式
- 支持临近地址实时压缩
- 可正确读出120MHZ的数据率
- 还需集成内部debug模块
- 版图还需优化，并提供时序和功耗报告

Thank you!

Modified SPI interface





Question:
The driven strength of PAD for START_OUT ?

- Use STAR_I generate a internal CSB (SPI_CSB).
- Write register control "SPI_D" to disable the SPI interface and to generate START_OUT, which is connected to STAR_I of the next chip.

Next work

- Integrate the on-chip debug module
- Finish the final layout
- Provide timing report
- Provide power analysis report
- Extract delay information & Post simulation

On-chip test

- **Register 00100~00101:**

On chip channel test control. The function is not included in the present version.

- OCTE: On chip channel test enable. 1: enable , 0: disable (default).
- F/V: Fastor or Valid waveform select. 0: fastor waveform, 1: Valid waveform.
- FVL[2:0]: Fastor or Valid waveform length. Unit in read counts
- API: Line-by-line or interlaced address generating selecting. 0: Line-by-line, 1: interlaced
- ADR[9:0]: The on chip channel test controller start address.