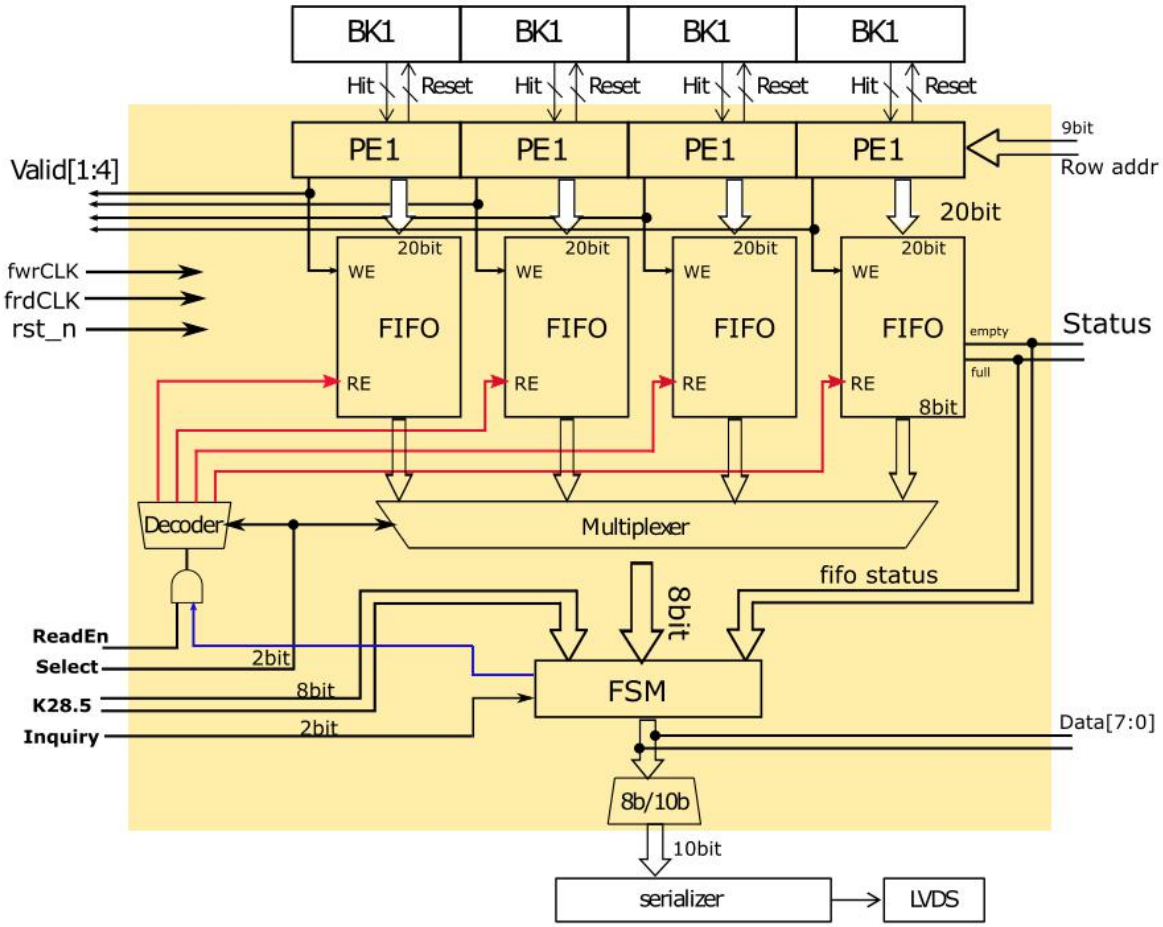
ReadOut 后端实现

2019-1-25

肖乐

ReadOut设计



Readout后端实现方式：

1.每个模块分别制作版图，然后做整体版图

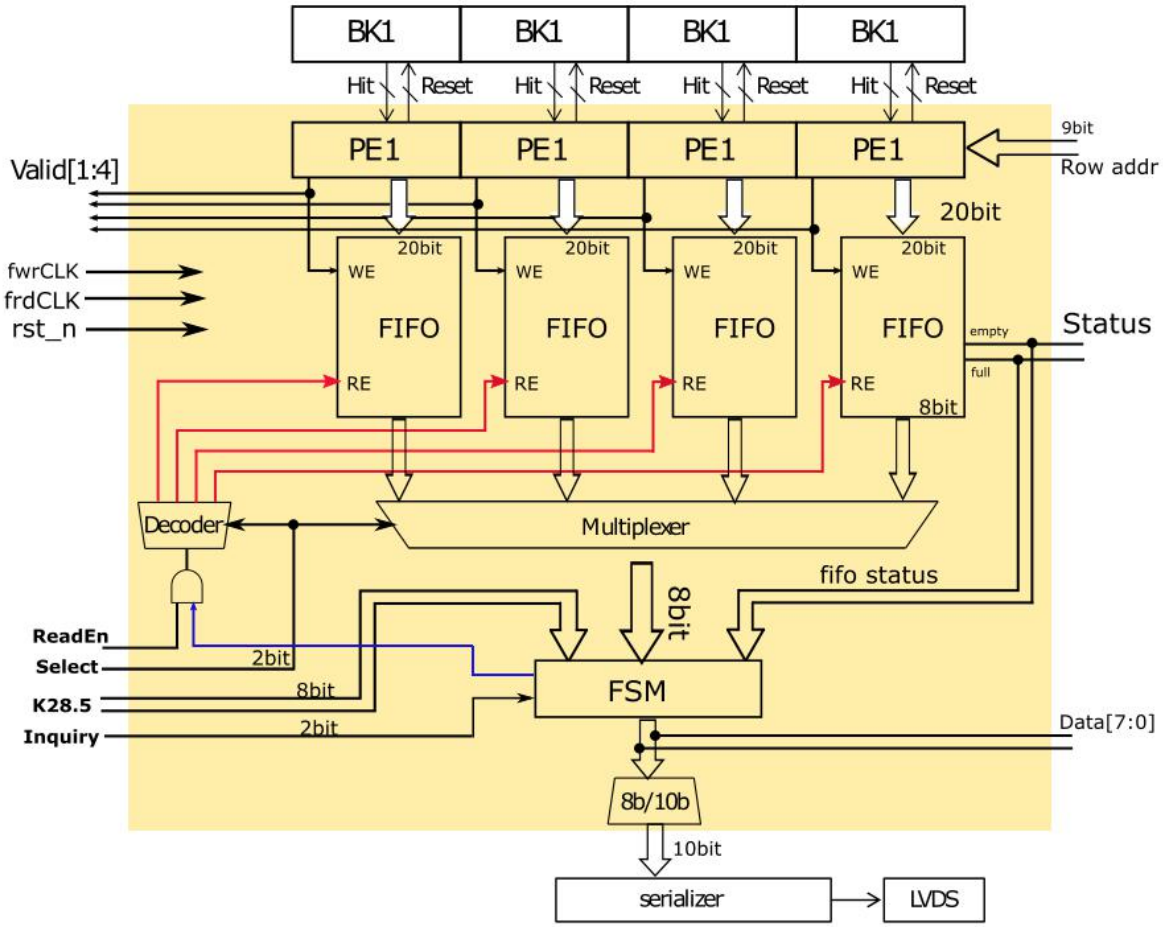
2. 所有模块一起制作版图

将所有模块一起制作版图：

- 时序约束简单
- 时序更加可靠
- 节省时间

注：数字后端实现左图背景为黄色的部分

ReadOut设计



Readout接口：

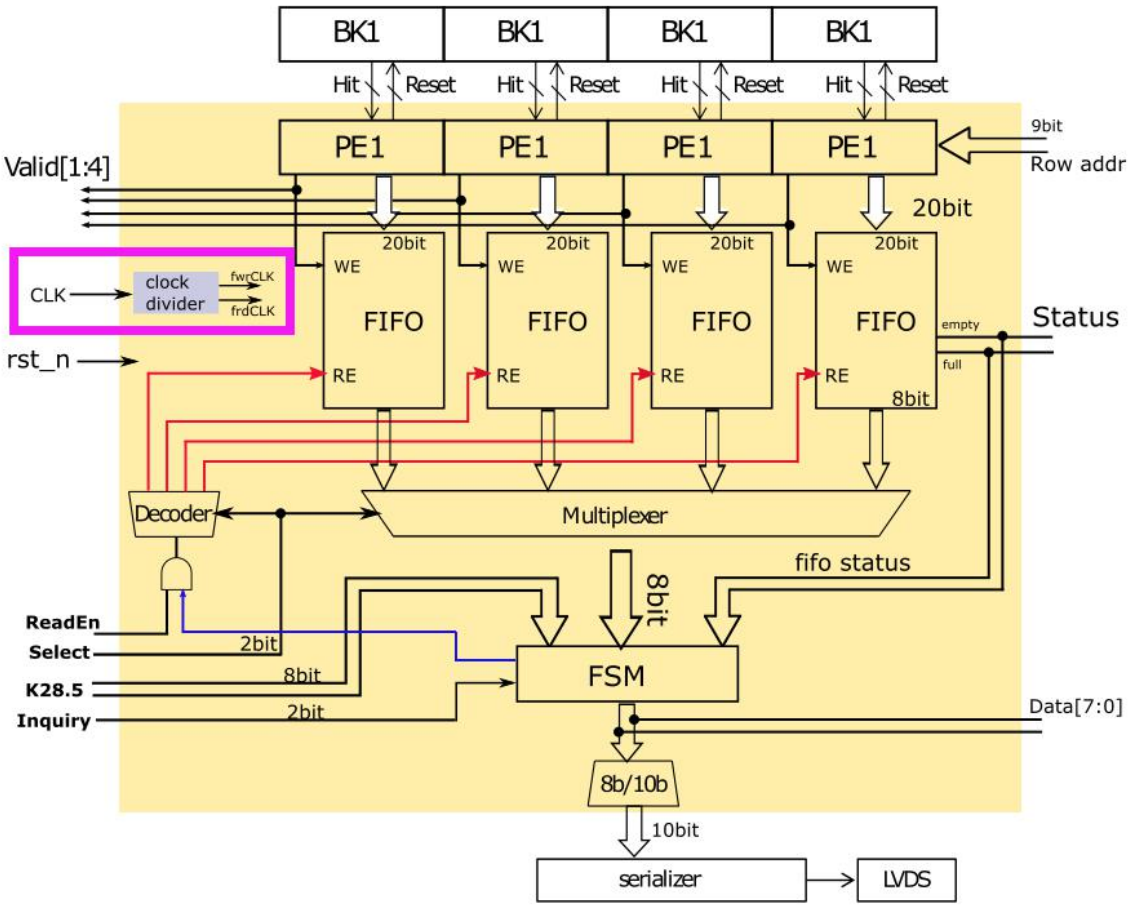
- fwrCLK:40Mhz, fifo写入时钟
- frdCLK:62.5Mhz, fifo读出时钟

而这两个时钟是同源时钟:

- 为了设计约束更加准确
- 预估面积和功耗

加入 **时钟分频** 模块

ReadOut设计



Readout接口：

如左图所示，加入 **时钟分频** 模块，将外部输入CLK分频生成

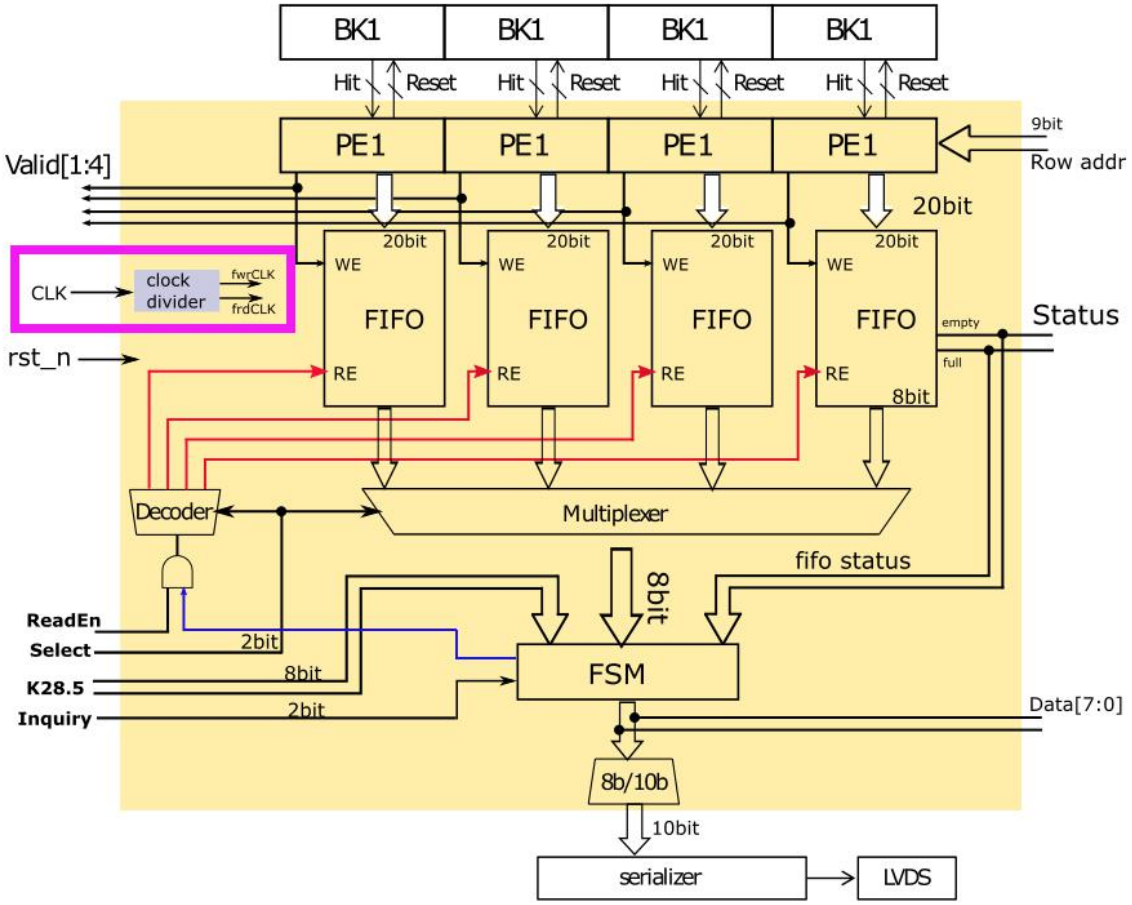
- fwrCLK:40Mhz(25ns), fifo写入时钟
- frdCLK:62.5Mhz(16ns), fifo读出时钟

因此，要求外部输入CLK时钟为 1000Mhz(1ns),

为了降低输入CLK频率，以及物理实现简单、节省面积和功耗；

将fwrCLK周期改为24ns, 此时CLK为250Mhz

ReadOut设计



Readout接口：

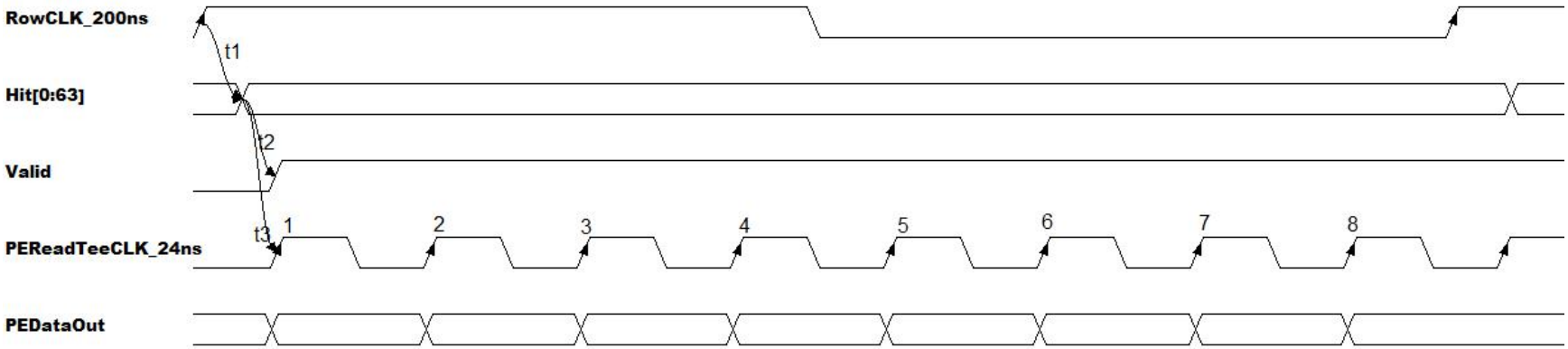
- 系统接口
  - CLK(1bit,4ns)
  - rst_n(1bit)
- **PE接口**，参考时钟fwrCLK,
  - ValidOut[3:0]
  - Reset[255:0]
  - Hit[255:0]
  - RowAddr[8:0]
- 控制接口,参考时钟frdCLK,
  - Status(8)
  - Data(8)
  - BitStream(10)
  - Inquiry(2)
  - Select(2)
  - K28.5(8)?
  - ReadEn(1)

ReadOut Clock Manage

由于ReadOut源时钟是250Mhz(4ns)，分频产生24ns的fwrCLK(PEReadTreeCLK)和16ns的frdCLK。

而Row Scan CLK是采用200ns的时钟，当每段都有击中时(每个BK 64-bit拆分成8段，每一段8个像素)，为了避免在200ns期间将**最后一段数据丢失**：

- 要求Hit[0:63]有效与Valid有效之后第一个fwrCLK(PEReadTreeCLk)的时间差 < (24+8)ns
- 而Hit[0:63]在RowCLK上升沿之后t1时刻有效，因此为了避免数据丢失
  - RowCLK和fwrCLK(PEReadTreeCLk)也采用**同源时钟**
  - **需要t1值**



ReadOut时钟约束

Clock Constraint

源时钟约束：

- create_clock -name CLK -period 4 -waveform {0 2} [get_ports CLK];
- set_clock_uncertainty -setup 0.2 [get_clocks CLK]; #To model clock network skew
- set_clock_transition -max 0.12 [get_clocks CLK];

为占空比50%, 建立时间0.2ns, 最大转换时间是0.12，周期为4ns的ReadOut源时钟

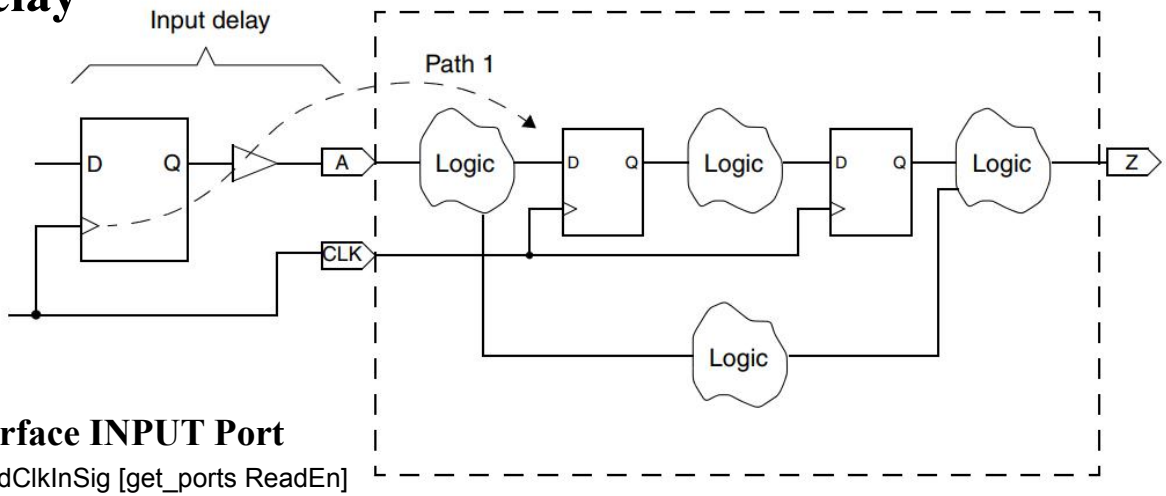
生成时钟约束：

- create_generated_clock -name fwrCLK -source [get_clocks CLK] -divide_by 6 [get_pins INS_CLKDIV/fwrCLK];
- create_generated_clock -name frdCLK -source [get_clocks CLK] -divide_by 4 [get_pins INS_CLKDIV/frdCLK];

生成时钟周期为24ns的fwrCLK， 以及时钟周期为16ns的 frdCLK ;

ReadOut 输入接口约束

Input Port Delay



FPGA Control Interface INPUT Port

- append_to_collection rdClkInSig [get_ports ReadEn]
- append_to_collection rdClkInSig [get_ports CtrlSym*]; # K28.5
- append_to_collection rdClkInSig [get_ports Inquiry*]
- append_to_collection rdClkInSig [get_ports Select*]

set_input_delay 7 -clock [get_clocks frdCLK] $rdClkInSig; # 7ns

PE Interface INPUT Port

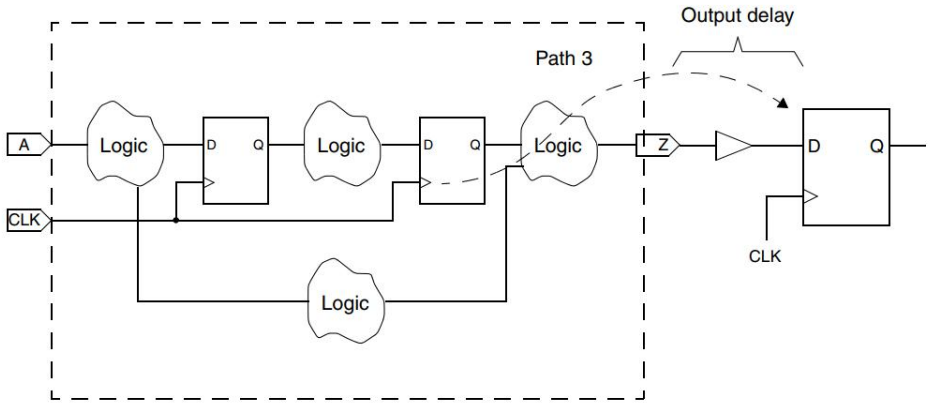
- append_to_collection wrClkInSig [get_ports **DataIn1***]; # blank1 hit [0:63]
- append_to_collection wrClkInSig [get_ports DataIn2*]; # blank2 hit [0:63]
- append_to_collection wrClkInSig [get_ports DataIn3*]; # blank3 hit [0:63]
- append_to_collection wrClkInSig [get_ports DataIn4*]; # blank4 hit [0:63]
- append_to_collection wrClkInSig [get_ports RowAddr*]; # row address [0:8]

set_input_delay 20 -clock fwrCLK $wrClkInSig; # 20ns

set_driving_cell -lib_cell inv0d1 [all_inputs];

ReadOut 输出接口约束

Output Port Delay



PE Interface OUTPUT Port

- append_to_collection wrClkOutSig [get_ports ReadTree1Clk*]; # **BK1 Reset**[0:63]
- append_to_collection wrClkOutSig [get_ports ReadTree2Clk*]; # **BK2 Reset**[0:63]
- append_to_collection wrClkOutSig [get_ports ReadTree3Clk*]; # **BK3 Reset**[0:63]
- append_to_collection wrClkOutSig [get_ports ReadTree4Clk*]; # **BK4 Reset**[0:63]
- append_to_collection wrClkOutSig [get_ports ValidOut*]; # **Valid[1:4]**

set_output_delay 20 -clock fwrCLK $wrClkOutSig; # 20ns

DataOut & Status OUTPUT Port

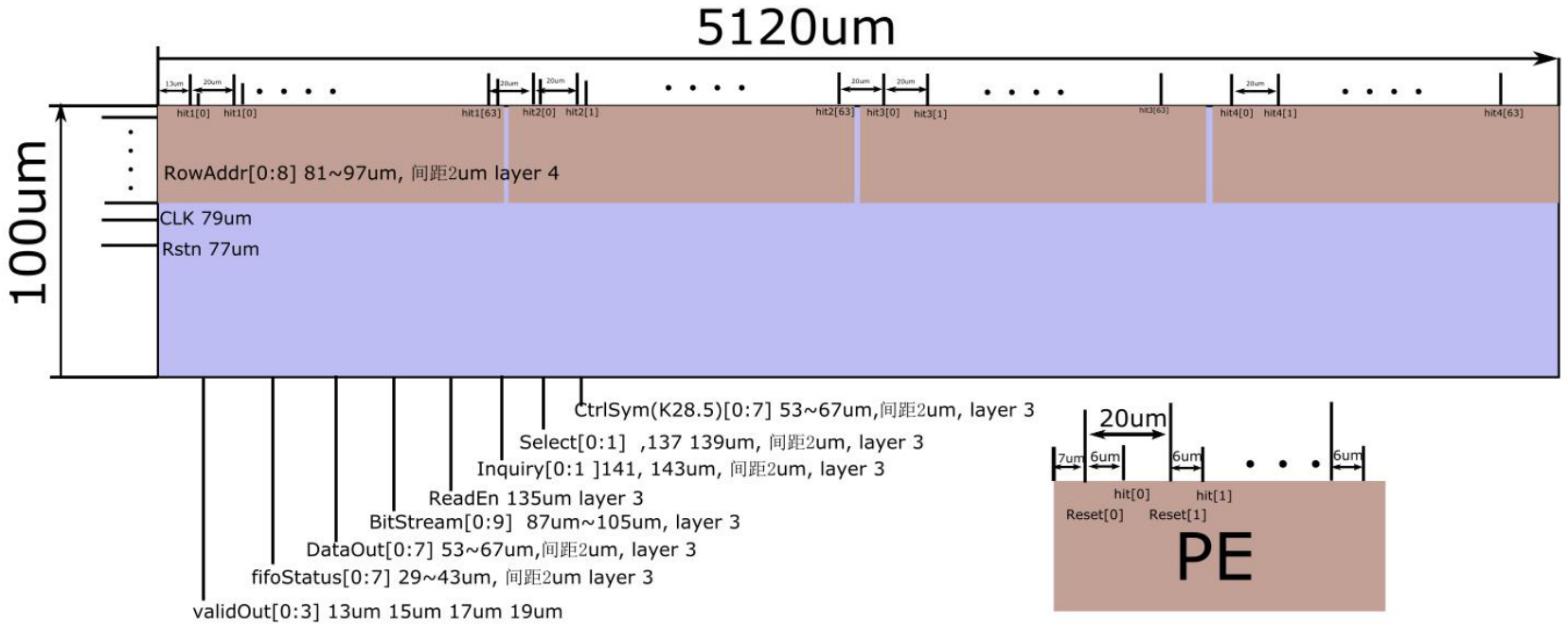
- append_to_collection rdClkOutSig [get_ports **BitStream***]; # 8b10b Output data
- append_to_collection rdClkOutSig [get_ports **DataOut***]; # 8B10B input data
- append_to_collection rdClkOutSig [get_ports **FifoSta***]; # fifo status

set_output_delay 7 -clock frdCLK $rdClkOutSig; # 7ns

输出端口带负载能力：

set_load -pin_load [expr [load_of tsl18fs120_ss_1p62v_125c/inv0d0/I] *10] [all_outputs];

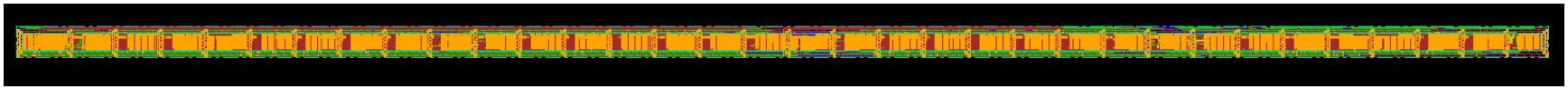
ReadOut设计_物理实现



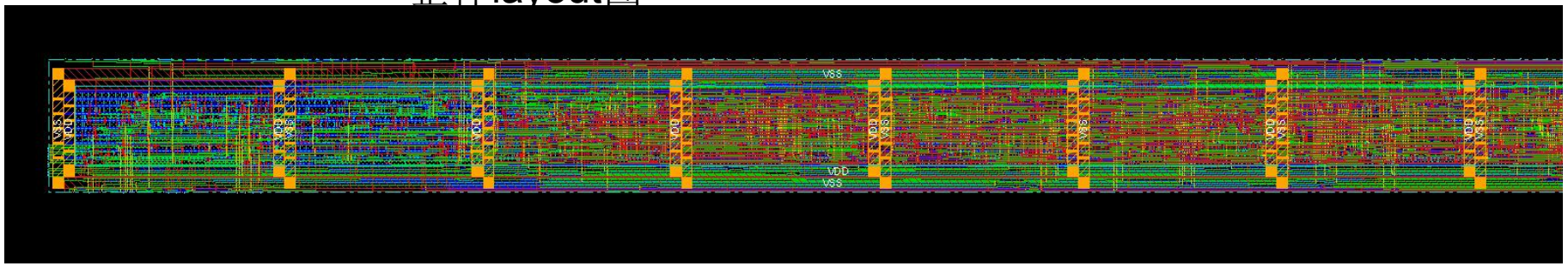
实现参数：

- 面积　　：　5120um*100um
- 金属布线层：　6
- 功耗　　：15.35mW
- 电源环宽度：8um
- postrouter_hold core utilization: 68%

ReadOut设计_物理实现

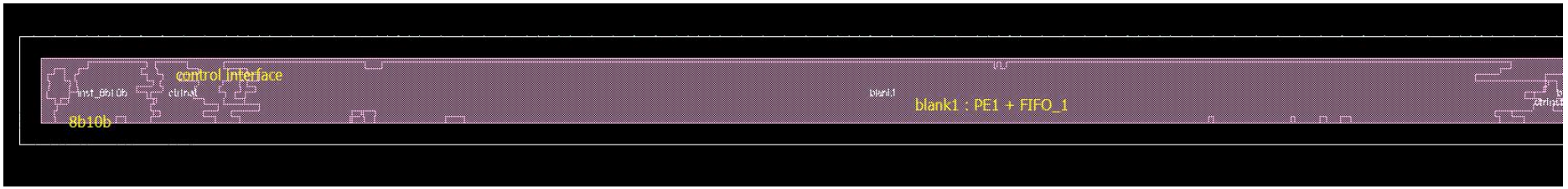


整体layout图



整体layout左侧图，每150um添加 Power Stripe

layout Amoeba view图



layout Amoeba view左侧图

图中显示包含8b10B，Control_Interface, Blabk1(PE+FIFO)