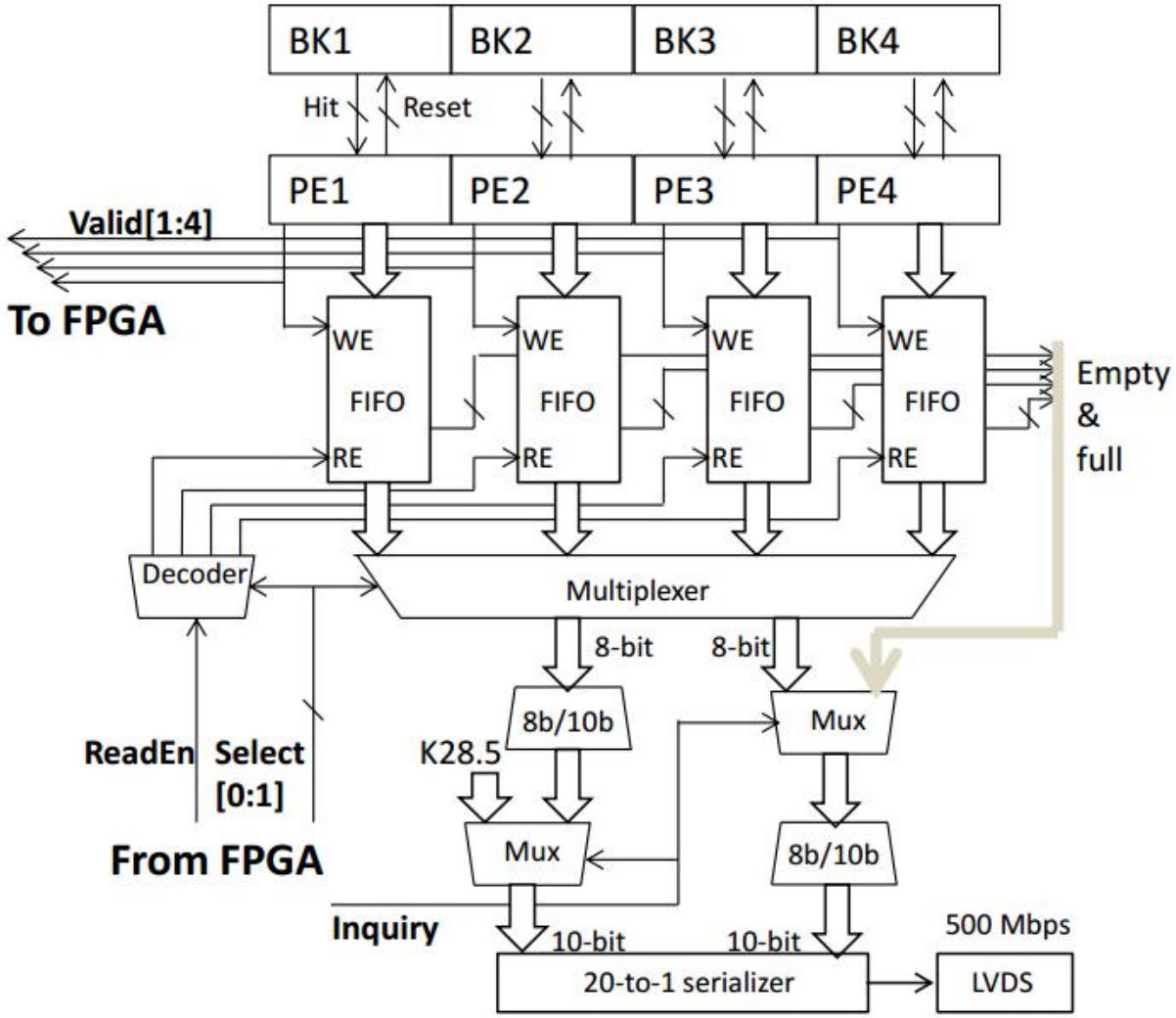
zero suppression and data readout

肖乐

2019年1月11日

读出方案

- 采用热编码，优先级读出
  - 将每个BK 64-bit拆分成8段，每一段8个像素，8个像素的击中情况采用8位热编码。
  - 每个BK中8段的读出则采用优先级编码读出
- 将每个BK 64-bit输入拆分成8个8-Byte
  - 压缩数据
  - 减小硬件开销
  - 不存在hit丢失情况(原方案每个BK超过10个hit会丢失),完成一个Byte读出时间只要25ns，读出一个BK最长时间需要200ns
  - 读出位宽9bit(Row)+3bit(Prority)+8bit(hot coding)
- 将两个8B/10B编码合并
  - Inquiry: 1bit---> 2bit



读出方案

新方案引入问题：

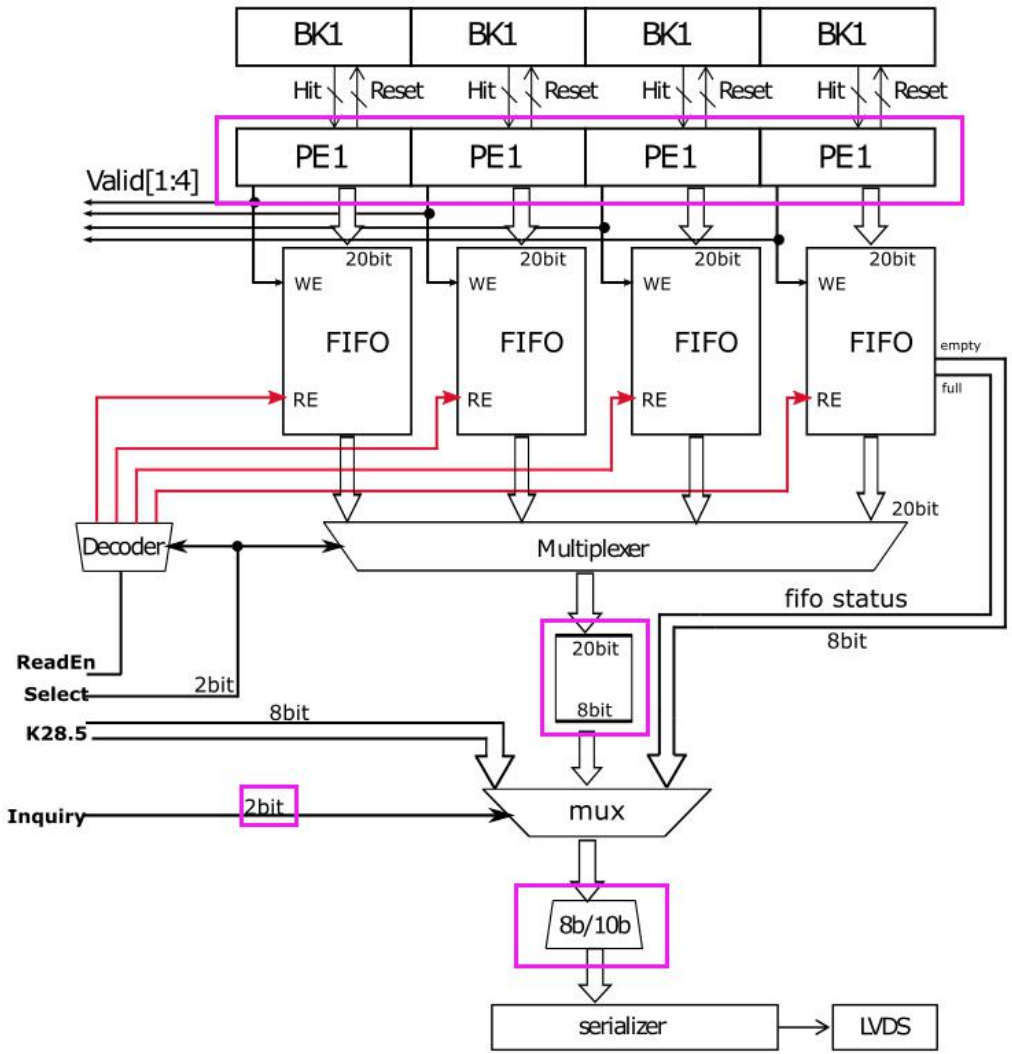
PE输出输出位宽20bit，而8b/10b编码输入数据是8bit，位宽不一致

解决方案：

- 加入20bit转8bit模块
- 为减小硬件开销，20b-to-8b模块放在两个MUX之间

缺陷：

- ✓ 4个FIFO中数据被混淆
- ✓ Empty和Full不包含20b-to-8b状态，容易导致20b-to-8b中数据丢失

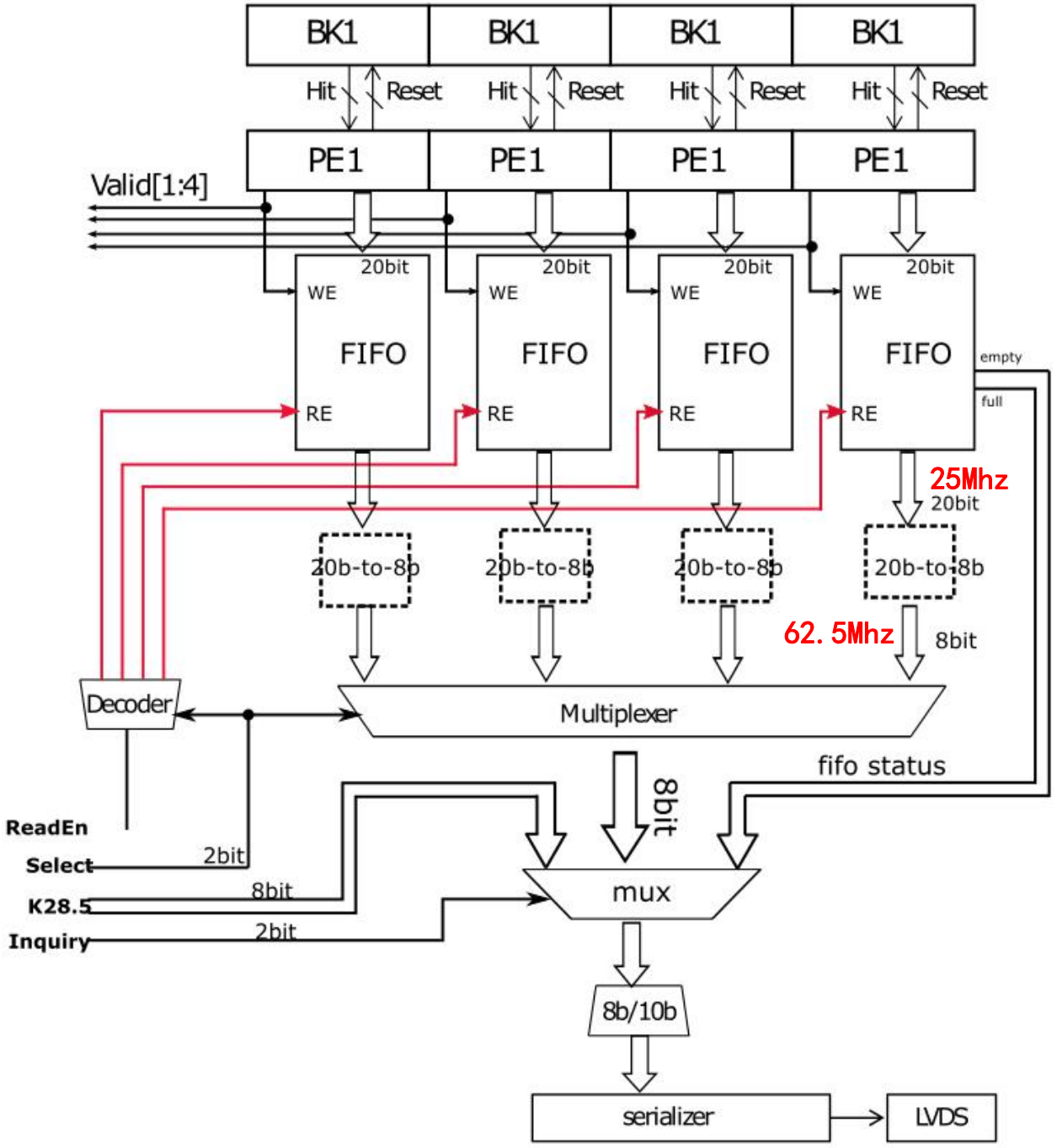


读出方案

解决FIFO中读出数据混淆的问题？
每个FIFO后接20b-to-8b

缺陷：

- ✓ select 选择FIFO与multiplexer的时序波形不一致； FIFO读扫描处于25Mhz信号控制,而multiplexer需要以62.5Mhz的信号控制
- ✓ FIFO和20b-to-8b都存储数据，empty和full不包含20b-to-8b状态，容易导致20b-to-8b中数据丢失
- ✓ FPGA外部控制读出操作复杂(ReadEn关闭之后，需等待20b-to-8b模块读空，才能改变Inquiry)



读出方案

为解决上一方案中存在缺陷：

将20b-to-8b与FIFO合并，变成输入(20)输出(8)位宽不一致的FIFO

- 控制接口

ReadEn：有效时，可以读出FIFO数据

Select ：选择其中1个FIFO读出

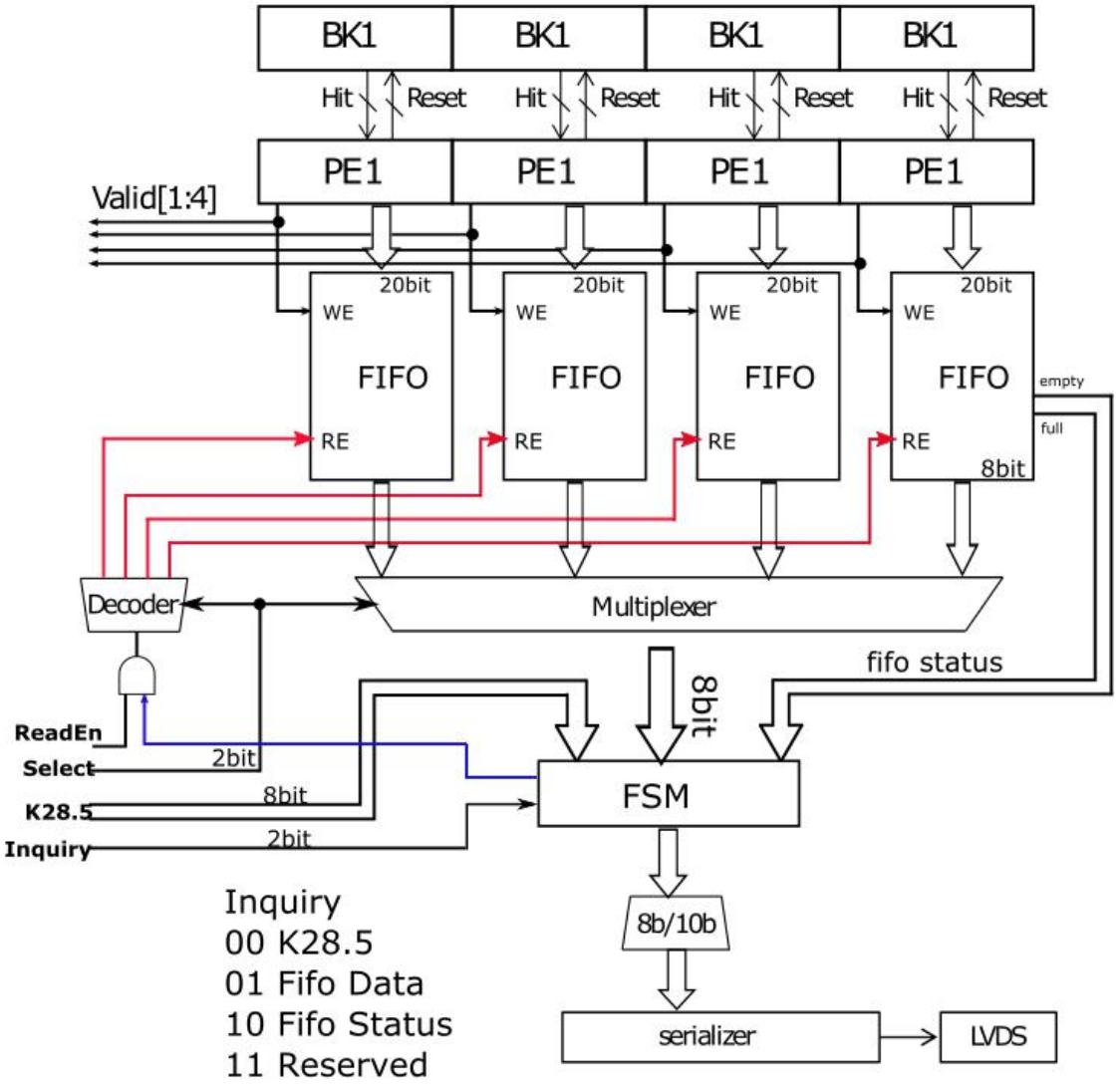
Inquiry：选择送入8b/10b数据来源

K28.5 ：外部输入接口

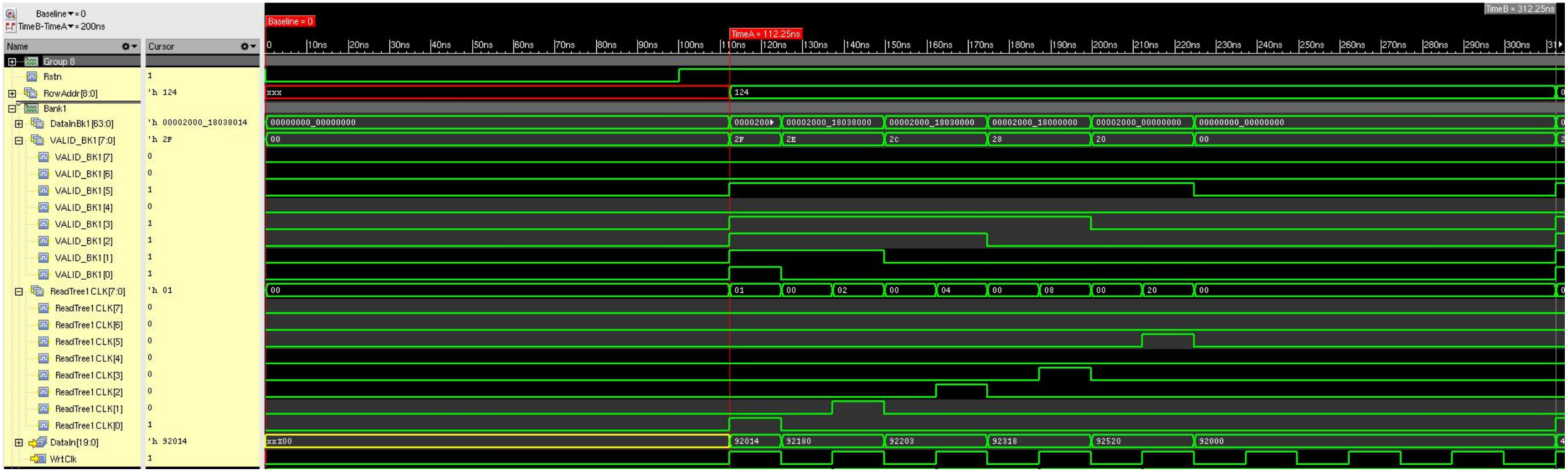
- 控制方法

电路刚上电时，可以通过Inquiry 选择输出格式：n byte(K28.5) + n byte（FIFO Status）；n 代表Inquiry持续n个FIFO输出时钟周期；

一段时间(或检测到FIFO非空)，使能ReadEn，Select选择需要读出的FIFO，Inquiry外部输入2'b10，开始将FIFO数据读出



优先级读出仿真



仿真添加激励方法：

1. 复位100ns，在复位之后的第一个优先级读出时钟上升沿时刻，添加激励
2. 激励的生成方式，生成一个1到10之间的随机数rand1，模拟64列中击中的总列数;生成一个0到63 之间的随机数rand2，模拟64列中第几列被击中

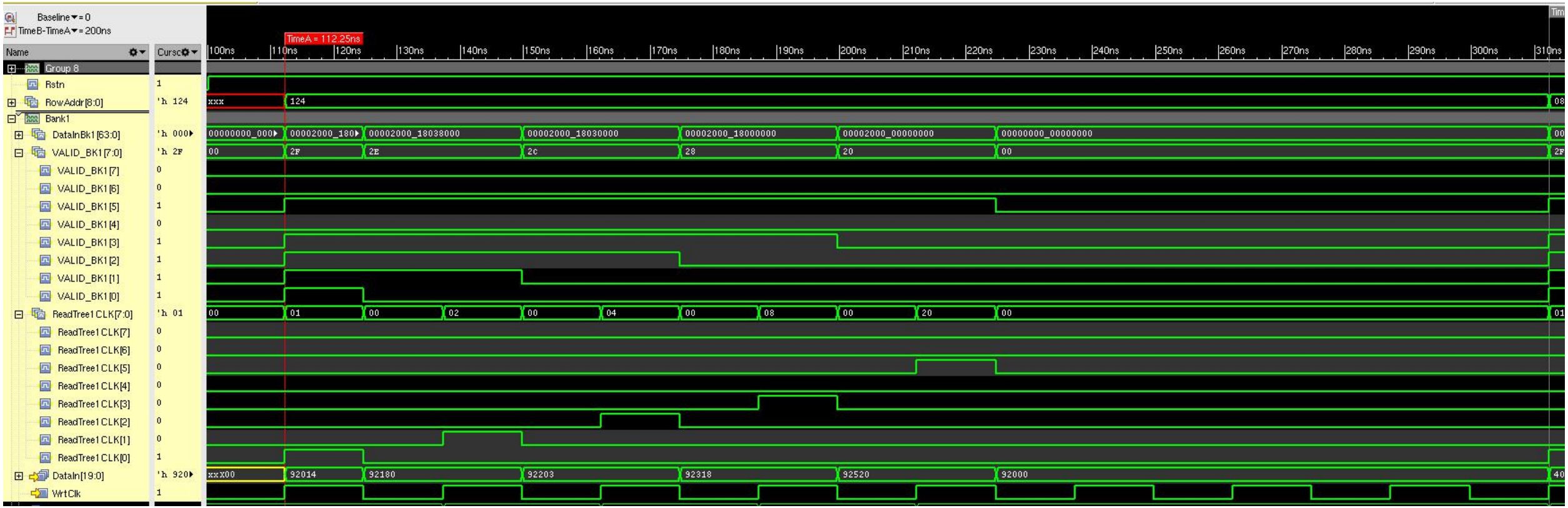
仿真结果：

DataBk1[7:0]中bit4和bit2击中，Valid_Bk1[0]置高，表示该Byte有击中；

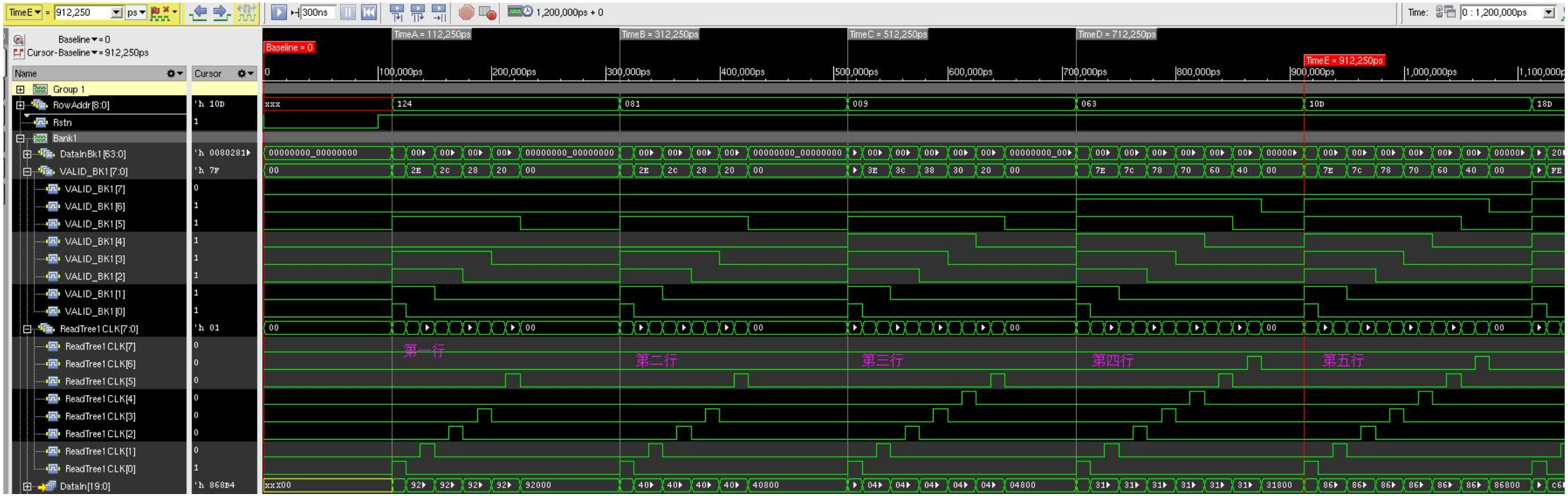
该读出时钟的上升沿时刻完成DataBk1[7:0]数据读出

该读出时钟的下降沿时刻完成DataBk1[7:0]复位

优先级读出仿真


Baseline = 0
TimeB-TimeA = 200ns
TimeA = 112.25ns
Name
Cursc
Group 8
Rstn
RowAddr[8:0]
Bank1
DataInBk1[63:0]
VALID_BK1[7:0]
VALID_BK1[7]
VALID_BK1[6]
VALID_BK1[5]
VALID_BK1[4]
VALID_BK1[3]
VALID_BK1[2]
VALID_BK1[1]
VALID_BK1[0]
ReadTree1CLK[7:0]
ReadTree1CLK[7]
ReadTree1CLK[6]
ReadTree1CLK[5]
ReadTree1CLK[4]
ReadTree1CLK[3]
ReadTree1CLK[2]
ReadTree1CLK[1]
ReadTree1CLK[0]
DataIn[19:0]
WrtClk
'h 124
'h 000
'h 2F
'h 01
'h 920
100ns
110ns
120ns
130ns
140ns
150ns
160ns
170ns
180ns
190ns
200ns
210ns
220ns
230ns
240ns
250ns
260ns
270ns
280ns
290ns
300ns
310ns
xxx
124
00000000_000
00002000_180
00002000_18038000
00002000_18030000
00002000_18000000
00002000_00000000
00000000_00000000
00
2F
2E
2C
28
20
01
02
04
08
xxX00
92014
92180
92203
92318
92520
92000

优先级读出仿真



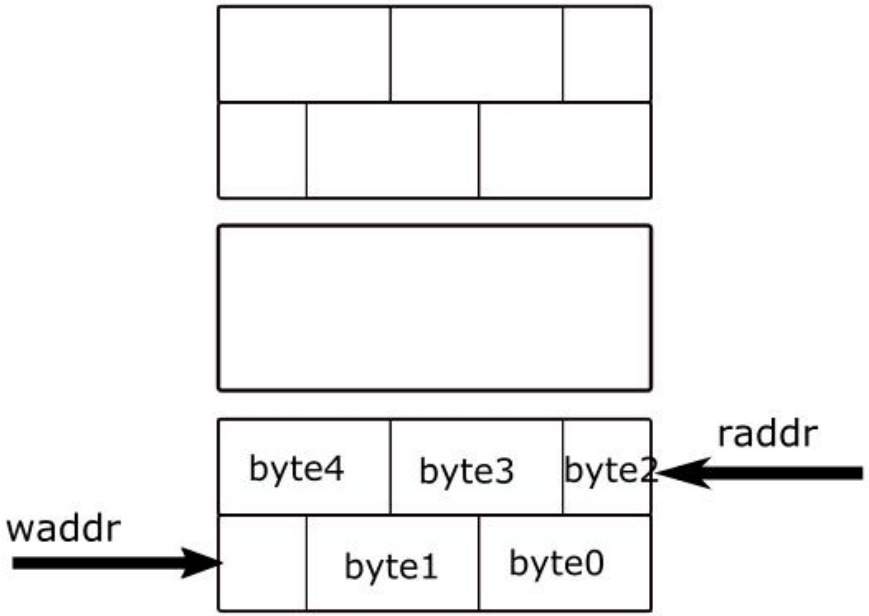
每200ns一行， 多行仿真测试

FIFO设计

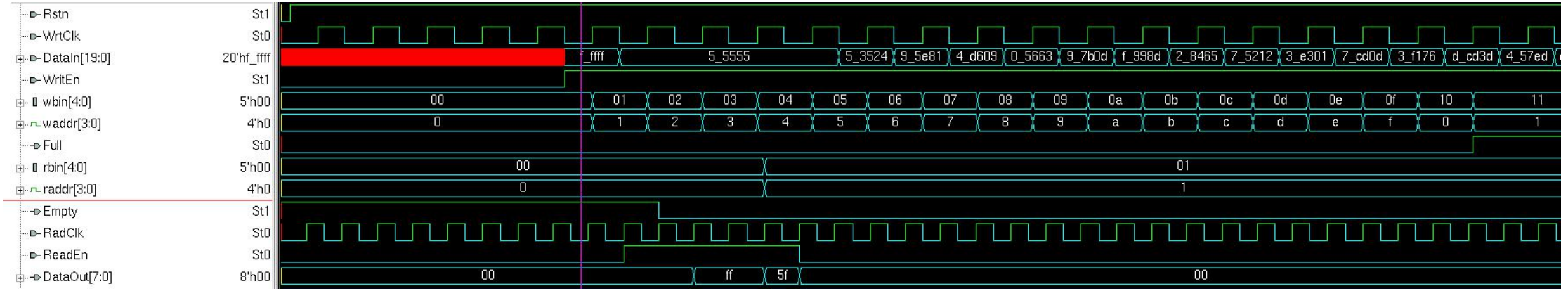
由于FIFO输入位宽20bit，写入深度为16。输出位宽为8bit，深度为40;
为了实现数据对齐，读出端将FIFO看做8个40bit，byte指针表示40bit中地址，读到byte2或byte4时， 读指针加1。
为了易于产生full和empty信号，读指针与写指针位宽为5bit。

FIFO Full产生：
将读指针和byte指针同步到写时钟域，通过和写指针比较，读写指针最高位不同，其他位相同且byte指针指向Byte4或Byte0时,将Full置高

FIFO Empty产生：
将写指针同步到读时钟域，通过和写指针比较，读写指针相同且byte指针指向Byte4或Byte0时,将Empty置高



FIFO仿真



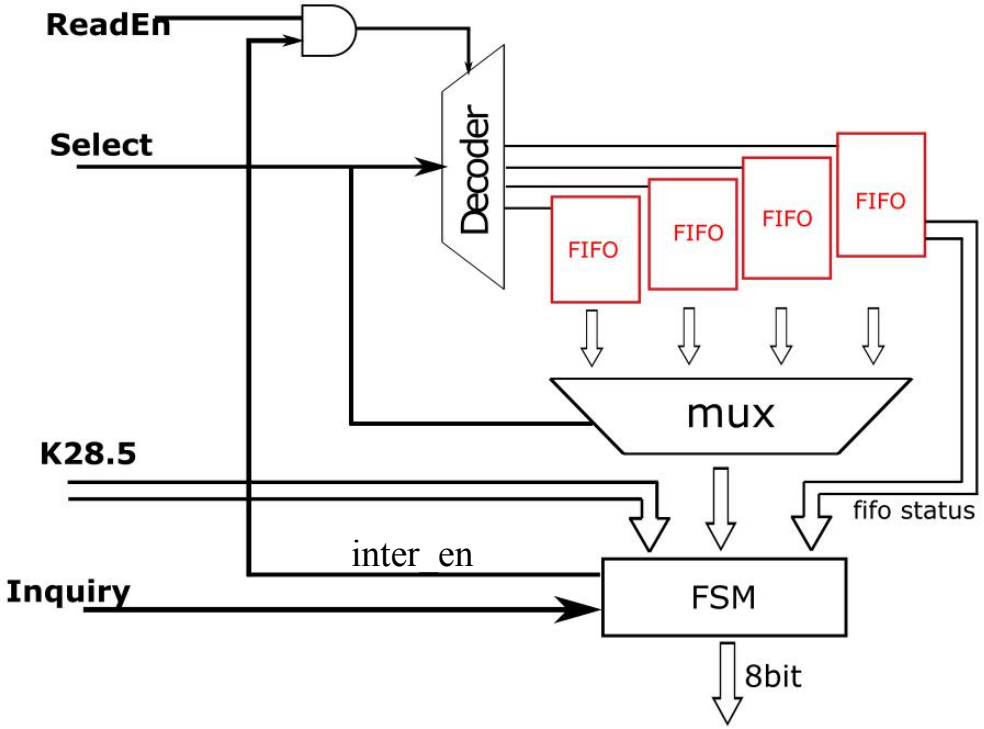
FIFO仿真：

1. 电路复位104ns,读写使能置低，复位结束之后，先写入2 * 20bit数据，读出5 * 8byte数据，再写入18 * 20bit数据；
2. 通过仿真结果可见，Empty信号会在写入数据之后第一个读时钟上升沿时刻拉低，Full信号会FIFO最后可以地址写入的下一写时钟上升沿时刻置高

仿真测试方法：

2. 写入20bit@25Mhz；读出8bit@62.5Mhz
3. 每次写入两个写周期， 每次读出五个读周期
4. FIFO仿真情景： 写数据， 读数据， 读空， 写满， 先读后写， 先写后读， 边读边写

读出控制接口设计



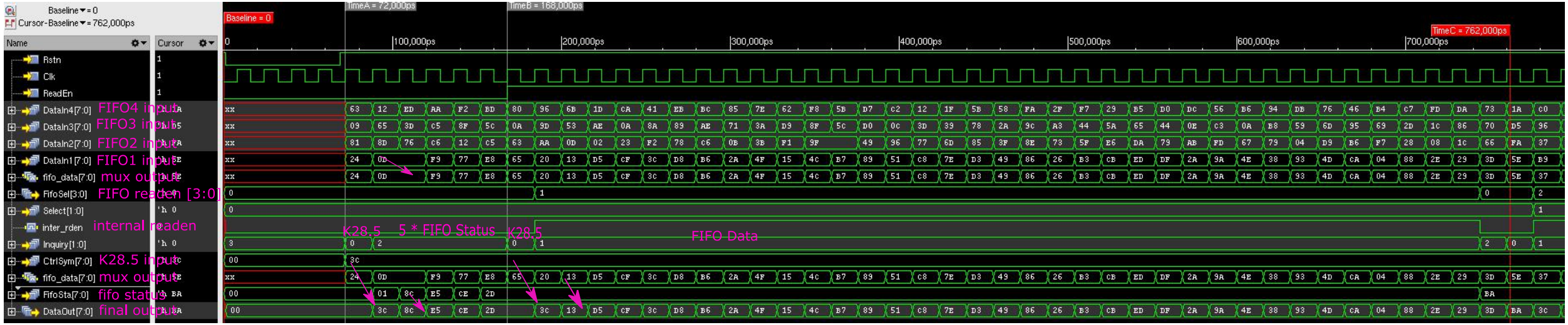
该接口由一个2输入译码器，一个4输入多路复器，一个状态机组成；

Inquiry控制状态机跳转：

- Inquiry为2'b00，输出由外部输入的K28.5；此时inter_en置低，使所有的FIFO读出都失效
- Inquiry为2'b01，输出Select选中FIFO的读出数据；此时inter_en置高，且要求外部ReadEn也置高
- Inquiry为2'b10，输出fifo状态信息；此时inter_en置低，使所有的FIFO读出都失效
- Inquiry为2'b11，保留

注：为避免数据丢失或误操作，每次Inquiry为2'b10时，持续5Period(以FIFO读时钟为参考)，才能切换至其他状态

读出控制接口仿真



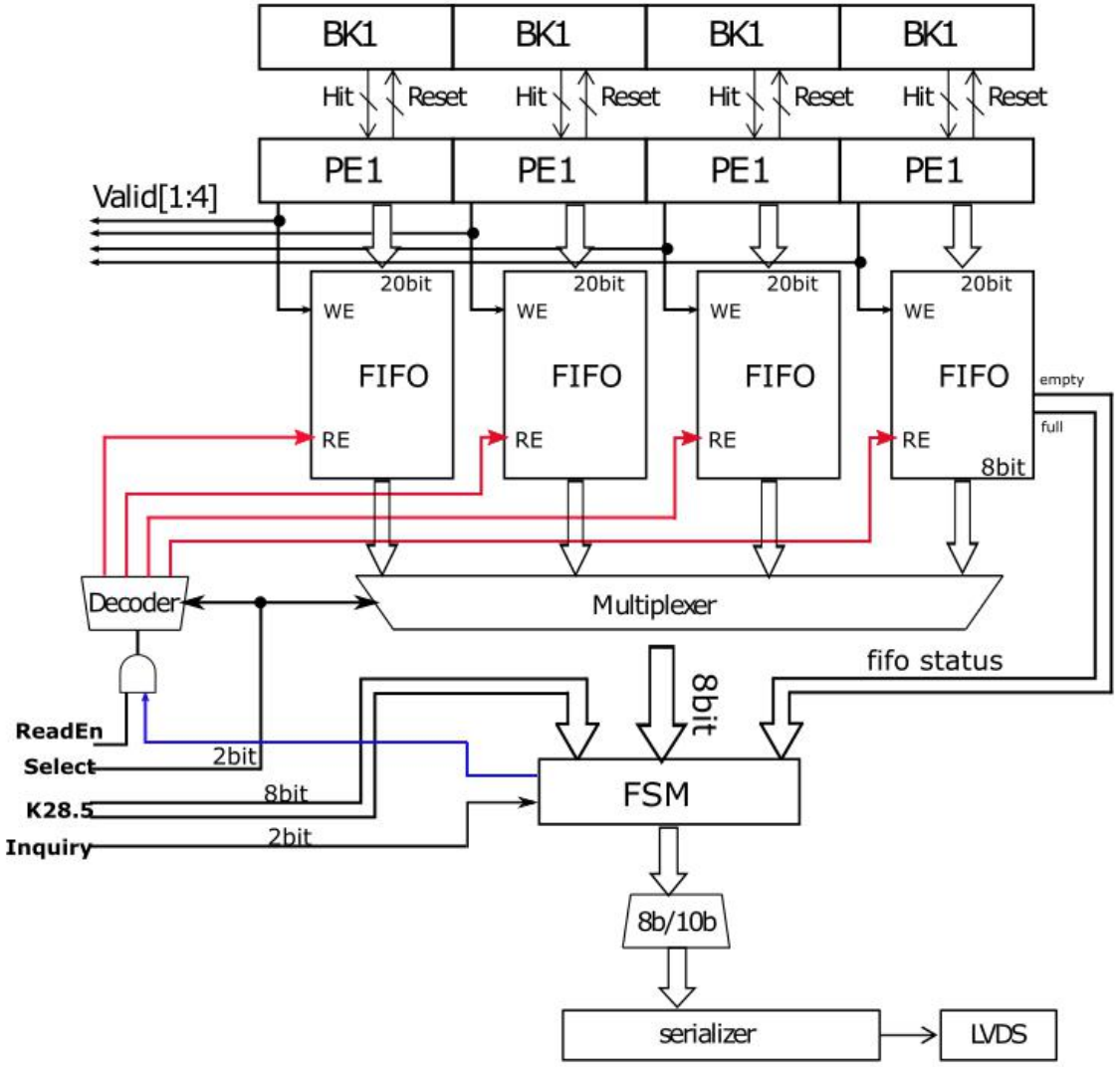
仿真激励添加：

1. 电路复位100ns，Select输入2'b00；复位结束后，读出一个控制码+5次 FIFO 状态信息
2. 读出ReadEn置高，实现读出控制码+35byte FIFO数据+ 1 FIFO Status，此时由FSM产生的inter_en信号会为低,从而让FIFO读出使能失效

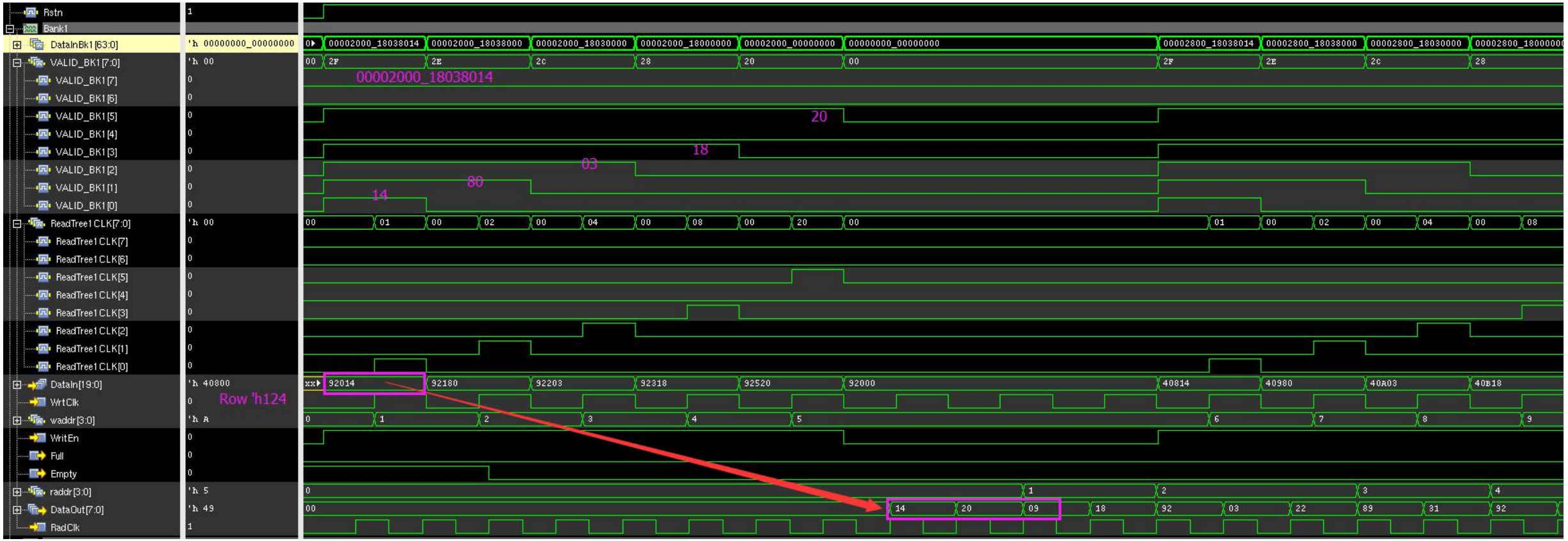
Inquiry 从“FIFO数据读出”切换至“K28.5输出”或“FIFO状态信息输出”状态时，会拉低inter_en，让4个FIFO都不能读出

Inquiry 从“K28.5输出”或“FIFO状态信息输出”状态切换至“FIFO数据读出”时，会置高inter_en，此时可以读出Select选中FIFO

TopLevel设计框图



TopLevel仿真结果



仿真添加激励方法：

1. 复位104ns，Select输入2'b00;复位结束之后，采用随机数的方法模拟像素击中，读出一个控制码+5次FIFO 状态信息
2. ReadEn置高，读出 1 控制码 + 35byte FIFO数据+ 1 FIFO Status

TopLevel仿真结果