|  |  |
| --- | --- |
| Gen\_waddr模块 | Wbank\_update信号，由计数器cnt\_hsync\_eq\_2line 生成  写此设计文档时可以从SRAM写入的位置排序方式考虑  用多个bank来存储全连接层并不好，因为读起来也会很难受 |
| 计数器 | R\_cnt\_hsync计数包括padding，写满4行拉高，之后写2个，考虑不用计数器，纯粹比较计数器大小不加与信号。  考虑是否会出现当写的慢但读的很快时，使得r\_cnt\_hsync一直为4联系触发读状态机，答案是不会，因此此状态机跳出IDLE的条件是外界输入的  计算r2bank\_done个数，用来产生s\_rbank2idle，使得SRAM从读状态变回IDLE状态，因为SRAM处于读状态时，只会进入一次done状态后结束，所以可以把 读取最后一个状态当成转换条件  Cnt\_matrix:  Mode0 : 2'd2\*(pic\_size-2'd2 + padding + padding)  Mode1: (pic\_size-2'd2 + padding + padding) or 2'd2\*(pic\_size-2'd2 + padding + padding)  Mode3 : 2'd2\*(pic\_size-2'd2 + padding + padding) – 1’b1  Mode4 : r\_cnt\_matrix \* 4'd9 > pic\_size  Mux\_ctrl\_6\_1中cnt\_ctrl\_update,  Mode0: ,改用read\_done与empty相与  计数sram2reg\_vld与rdy握手信号个数，当达到一定个数后SRAM从读写状态变成读状态  计数r\_reg2opu\_ctrl\_bit，一方面作为bit选择信号 |
| SRAM TO REG拉高条件 | 与SRAM状态有关，、  SRAM只写状态时，写满2个bank拉高；同时读写状态时，写完一个bank且SRAM读完2个bank且寄存器堆为空才拉高；SRAM为只读状态时，读完两个bank拉高 |
| 注意 | Gen\_raddr中最终回到idle的状态条件没给 |
| 虚假握手信号 | s\_opu\_1152\_rdy\_fake |
| S\_w\_all\_done | 写完一张图片的所有数据之后，会一直处于done状态，需要考虑它在下一张照片到来时的状态转换 |
| 全卷积层模式 | 当读取的matrix\*9大于pic\_size则表示读取结束  Gen\_raddr的状态机进入done状态同时也是sram\_interface回到idle状态的条件 |
|  | SRAM的状态变化，即写SRAM时，写满一个 bank后移位、读写完一个SRAM后读写信号移位、读完一个SRAM后读写信号移位。现在改为写状态时，写满一个bank移位、其它状态时，sram\_2\_reg状态移位 |
| tb | data |
| S\_wdata\_done | 包括mode3，所有模式均写入完成后一直拉高，直到新图片传来，data\_sop使进入sram写模式，s\_wdata\_done才会拉低 |
| r\_gen\_raddr\_done | 当读取一列数据完成后，会暂时进入done状态，此时会拉高此信号，直到sram2reg信号传来才会拉低 |
| S\_rsram2idle | 从读SRAM转为idle，等于rsram\_done，外部等于r\_gen\_raddr\_done，因为sram只要进入只读状态，读完立刻变idle |
| s\_w1bank\_done | 写完一个bank后拉高，用于与r\_gen\_raddr\_done相与拉高sram2reg\_rdy,但是当写的非常慢时，可能下一个hsync未传入，s\_w1bank\_done一直拉高，导致sram2reg\_rdy立刻被拉高。所以最好是在写完一个bank后此信号拉高，但下一次读写开始后立刻拉低 |
| 状态机纠错 | 原来是写的快，读的慢，所以sram从只读状态到IDLE状态的条件是读出完成，但当写的慢，读的快时，SRAM还需要额外读一列才算读完，才能回到idle状态 |
| 全连接层 | 注意：padding一定要为0 |