# 整体结构



# 数据生成之后就是计算模块，它主要是由计算卷积和量化两个部分组成。卷积部分承接3变九的输出，然后和加载的权重和偏置做乘加运算，由于做乘加时，数据时int8类型的，

# 根据 公式

# 因此后面我还需要将卷积结果经过量化处理，这样得到最终的结果。

# 计算模块

该模块是主要是指3\*3的卷积模块，是承接数据生成模块的。数据生成之后就会作为该模块的输入。该模块中有9个FIFO用来以此接收数据生成模块中三变九的输出，然后加载Bram中的的权重和9个FIFO的feature做3\*3的卷积运算，这个模块中，主要包含了计算的控制模块，加载权重模块，通道累加以及conv2d的计算模块。其中conv2d模块主要用于输入feature和weight的乘加，因为每次计算都是第一次乘加9对数据，为了方便，我们将前两个数据做延迟处理，一共延迟两个时钟，之后的加法就会变为8，4，2，1。



如上图所示，一张图片为4个通道，他的存储方式是以像素点的形式存储的，R0，G0，B0，00。



但是在计算的过程中我们是逐通道进行的，因此我们需要将数据逐通道提取出来重新组合，以方便后面的卷积运算。



这个过程在硬件是通过一个RAM实现的，通过配置RAM我们将像素点的数据重新按照通道进行整合，同时权重参数也是以通道的方式存储，这样输入到conv2d的模块时，就可以实现逐通道进行乘加运算。



上图表示卷积的整个过程，需要进行运算的9个点，分别进入到9个中FIFO然后分别与加载的权重参数做乘加运算。在进行乘加运算的过程中，会有2个时钟周期的延迟，即完成前两个像素点的乘加，之后这样后面的加法运算就依次只需要进行4次，2次，1次，最后的输出就是一个通道中，一个3\*3卷积核与相应3\*3个点的运算结果。对于4通道输入数据，我们还需要在4个通道结束计算的时候，将这4个通道的数值相加。这就是一个4通道输入经过卷积之后的结果。

**伪代码：**

软件形式的伪代码：

for i =0 ; i<C\_in ; i++ （输入通道）

for j=0 ; j<C\_out ; j++ （输出通道）

for k=0 ; k<F\_r ; k++ （特征层行数）

for l=0 ; l<F\_c ; l++ （特征层列数）

for m=0 ; m<k\_r ; m++ （卷积核行数）

for n=0 ; n<k\_c ; n++ （卷积核列数）

在硬件中，我们可以将卷积核的部分直接以乘加树的形式展开，这样整个循环就变成了4个。

for i =0 ; i<C\_in ; i++ （输入通道）

for j=0 ; j<C\_out ; j++ （输出通道）

for k=0 ; k<F\_r ; k++ （特征层行数）

for l=0 ; l<F\_c ; l++ （特征层列数）

乘加树

## Conv\_Norm

|  |  |
| --- | --- |
| **外部参数Parameter** | **数值** |
| KERNEL\_NUM | 9 |
| CONV\_TYPE | CONV\_3\_3 |
| COMPUTE\_CHANNEL\_IN\_NUM | 4 |
| COMPUTE\_CHANNEL\_OUT\_NUM | 4 |
| WIDTH\_RAM\_ADDR\_SIZE | 11 |
| WIDTH\_TEMP\_RAM\_ADDR\_SIZE | 7 |
| WIDTH\_FEATURE\_SIZE | 12 |
| WIDTH\_CHANNEL\_NUM\_REG | 10 |
| WIDTH\_DATA\_ADD | 32 |
| WIDTH\_RAM\_ADDR\_TURE | 11 |
| WIDTH\_WEIGHT\_NUM | 15 |
| WIDTH\_BIAS\_RAM\_ADDRA | 7 |

| **输信号** | **方向** | **位宽** | **组** | **信号描述** |
| --- | --- | --- | --- | --- |
| clk | I | 1 | 时钟和复位 | 时钟 |
| rst | I | 1 | 复位 |
| Start | I | 1 | 控制  信号 | 开始信号 |
| Bram\_Data\_In | I | 128 | 输入的Bram数据 |
| Bram\_Addrb | O | WIDTH\_RAM\_ADDR\_TURE | 两个时钟延迟 |
| Compute\_Complete | O | 1 | 计算完成信号 |
| S\_Data | I | 2\*COMPUTE\_CHANNEL\_IN\_NUM\*KERNEL\_NUM\*8 | Slave | 三变九的输出结果 |
| S\_Valid | I | KERNEL\_NUM | 上一层三变九完成 |
| S\_Ready | I | 1 | 本层的计算完成 |
| M\_Data\_out | O | 2\*COMPUTE\_CHANNEL\_OUT\_NUM\*WIDTH\_DATA\_ADD | Master | 完成本层计算的数据输出 |
| M\_ready | I | 1 | 下一层的量化准备完毕 |
| M\_Valid | O | 1 | 完成一行的计算可以输出给量化模块 |
| Ram\_Read\_Addrb\_Base | I | WIDTH\_RAM\_ADDR\_TURE | 控制  信号 | Ram 的基地址 |
| Row\_Num\_Out\_REG | I | WIDTH\_FEATURE\_SIZE | 计算之后的输出行数 |
| RowNum\_After\_Padding | I | WIDTH\_FEATURE\_SIZE | Padding 之后的行数 |
| Channel\_In\_Num\_REG | I | WIDTH\_CHANNEL\_NUM\_REG | 输入的通道数 |
| Channel\_Out\_Num\_REG | I | WIDTH\_CHANNEL\_NUM\_REG | 输出的通道数 |
| Weight\_Single\_Num\_REG | I | WIDTH\_WEIGHT\_NUM |  |
| Bias\_Num\_REG | I | WIDTH\_BIAS\_RAM\_ADDRA | 偏置的数量 |
| Bias\_Addrb | I | 13 | 偏置的地址 |
| Data\_Out\_Bias | O | 32\*4 | 量化中的Bias |
| Data\_Out\_Scale | O | 32\*4 | 量化中的Scale |
| Data\_Out\_Shift | O | 32\*4 | 量化中的Shift |

| **内部信号** | **信号类型** | **位宽** | **信号描述** |
| --- | --- | --- | --- |
| COMPUTE\_TIMES\_CHANNEL\_IN\_REG | Wire | WIDTH\_FEATURE\_SIZE | 要计算的输入通道次数 |
| COMPUTE\_TIMES\_CHANNEL\_OUT\_REG | wire | WIDTH\_FEATURE\_SIZE | 要计算的输出通道次数 |
| Load\_Start | wire | 1 | 加载权重的信号 |
| Load\_Weight\_Complete | wire | 1 | 权重加载完成 |
| Weight\_Addrb | wire | 13 | 取权重的地址 |
| Data\_Out\_Weight | wire | 9\*4\*4\*8 | 权重的输出数据，4入4出9个点 |
| rd\_en\_fifo | wire | 1 | 计算完成一行，可以读出 |
| fifo\_compute\_ready | wire | 1 | Compute\_control 的输入，第一个FIFO以及写满一行 |
| ram\_temp\_read\_address | wire | WIDTH\_TEMP\_RAM\_ADDR\_SIZE |  |
| ram\_temp\_write\_address | wire | WIDTH\_TEMP\_RAM\_ADDR\_SIZE |  |
| First\_Compute\_Complete\_Temp | Wire | 1 | 承接compute\_control的输出，第一个点的计算完成 |
| data\_result\_temp | Wire | 2\*10\*2 |  |
| M\_Valid\_Temp | Wire | 1 |  |

### compute\_control

compute\_control：用于控制整个计算模块，包括控制权重的加载，数据的计算，读写FIFO等内容。

整个的过程主要是，首先启动权重加载模块，加载完毕之后，会启动往9个FIFO中写入数据。

|  |  |
| --- | --- |
| **外部参数Parameter** | **数值** |
| WIDTH\_RAM\_ADDR\_SIZE | 12 |
| WIDTH\_FEATURE\_SIZE | 12 |
| WIDTH\_TEMP\_RAM\_ADDR\_SIZE | 7 |
| WIDTH\_CHANNEL\_NUM\_REG | 10 |
| DELAY\_TIMES | 11+4+1+1+1 |

| **传输信号** | **方向** | **位宽** | **组** | **信号描述** |
| --- | --- | --- | --- | --- |
| clk | I | 1 | 时钟和复位 | 时钟 |
| rst | I | 1 | 复位 |
| Start | I | 1 | 控制  信号  控制  信号 | 开始信号 |
| Load\_Start | O | 1 | 开始加载参数信号 |
| Load\_Weight\_Complete | I | 1 | 加载权重完成 |
| Compute\_Complete | O | 1 | 计算完成信号 |
| First\_Compute\_Complete | O | 1 | 第一次计算完成的信号 |
| compute\_fifo\_ready | I | 1 | 计算Fifo准备完成 |
| rd\_en\_fifo | O | 1 | 计算完成，可以读出 |
| weight\_addrb | O | 13 | 读权重，延迟两个时钟 |
| ram\_temp\_read\_address | I | WIDTH\_TEMP\_RAM\_ADDR\_SIZE | Ram的读地址，延迟两个时钟 |
| ram\_temp\_write\_address | I | WIDTH\_TEMP\_RAM\_ADDR\_SIZE | Ram的写地址 |
| COMPUTE\_TIMES\_CHANNEL\_IN\_REG | I | WIDTH\_FEATURE\_SIZE | 输入通道的次数 |
| COMPUTE\_TIMES\_CHANNEL\_OUT\_REG | I | WIDTH\_FEATURE\_SIZE | 输出通道的次数 |
| ROW\_NUM\_CHANNEL\_OUT\_REG | I | WIDTH\_FEATURE\_SIZE | Padding之后的行数 |
| M\_ready | I | 1 | Master |  |
| M\_Valid | O | 1 |  |

| **内部信号** | **信号类型** | **位宽** | **信号描述** |
| --- | --- | --- | --- |
| Cnt\_Column | Reg | WIDTH\_FEATURE\_SIZE | 列计数 |
| Cnt\_Row | Reg | WIDTH\_FEATURE\_SIZE | 行计数 |
| Cnt\_Channel\_Out\_Num | Reg | WIDTH\_CHANNEL\_NUM\_REG | 输出通道计数 |
| Cnt\_Channel\_In\_Num | Reg | WIDTH\_CHANNEL\_NUM\_REG | 输入通道计数 |
| En\_Compute\_Column | Wire | 1 | 完成一行最后一点的卷积 |
| En\_Compute\_Row | Wire | 1 | 完成最后一行的计算 |
| First\_Complete | Reg | 19 | 开始计算的信号 |
| M\_Fifo\_Valid | Reg | 20 | 上一层的FIFO |

状态转移图：



### Load\_Weight\_Bias

|  |  |
| --- | --- |
| **外部参数Parameter** | **数值** |
| WIDTH\_RAM\_ADDR\_TURE | 15 |
| WIDTH\_WEIGHT\_NUM | 15 |
| WIDTH\_RAM\_ADDR\_SIZE | 10 |
| WIDTH\_BIAS\_RAM\_ADDRA | 7 |
| KERNEL\_NUM | 9 |
| COMPUTE\_CHANNEL\_IN\_NUM | 4 |
| COMPUTE\_CHANNEL\_OUT\_NUM | 4 |

| **传输信号** | **方向** | **位宽** | **组** | **信号描述** |
| --- | --- | --- | --- | --- |
| clk | I | 1 | 时钟和复位 | 时钟 |
| rst | I | 1 | 复位 |
| S\_Data | I | 128 | Slave | 输入数据 |
| Start | I | 1 | 控制  信号 |  |
| Weight\_Single\_Num\_REG | I | WIDTH\_WEIGHT\_NUM | 单通道的权重个数 |
| Bias\_Num\_REG | I | WIDTH\_BIAS\_RAM\_ADDRA | 偏置个数 |
| Load\_Weight\_Complete | O | 1 | 加载权重完成 |
| Bram\_Addrb | I | WIDTH\_RAM\_ADDR\_TURE | Bram 的读地址 |
| Ram\_Read\_Addrb\_Base | I | WIDTH\_RAM\_ADDR\_TURE | 读ram 的基地址 |
| Weight\_Addrb | I | 13 | 权重地址 |
| Bias\_Addrb | I | 13 | 偏置地址 |
| Data\_Out\_Weight | O | 9\*4\*4\*8 | Master | 权重数据输出 |
| Data\_Out\_Bias | O | 32\*4 | bias |
| Data\_Out\_Scale | O | 32\*4 | Scale |
| Data\_Out\_Shift | O | 32\*4 | Shift |

| **内部信号** | **信号类型** | **位宽** | **信号描述** |
| --- | --- | --- | --- |
| Cnt\_Bias | Reg | WIDTH\_BIAS\_RAM\_ADDRA | Bias计数 |
| Cnt\_Single\_Weight | Reg | WIDTH\_WEIGHT\_NUM | 单个权重的计数 |
| Cnt\_Ram\_Weight\_Num | Reg | 4 | 权重Ram计数，8个 |
| En\_Single\_Ram\_Temp | Reg | 1 |  |
| En\_Weight | wire | 1 | 所有通道权重加载完毕 |
| En\_Bias | wire | 1 | 单通道偏置加载完毕 |
| En\_Single\_Ram | wire | 1 | 单通道权重加载完毕 |

状态转移图：



## Conv2d\_state

| **传输信号** | **方向** | **位宽** | **组** | **信号描述** |
| --- | --- | --- | --- | --- |
| clk | I | 1 | 时钟和复位 | 时钟 |
| rst | I | 1 | 复位 |
| Control | I | 4 | Slave | 上位机控制的模块 |
| Pare\_Complete | I | 1 | 参数加载完成 |
| Cp\_Complete | I | 1 | 计算完成 |
| State | O | 4 | master |  |
| Start\_Cp | O | 1 | 给计算模块的开始计算信号 |
| Start\_Pa | O | 1 | 给参数模块的开始加载参数 |
| dest | O | 1 | 作为axi\_stream 的输入信号 |

状态转移图：



通过上位机控制是开启计算模块还是加载参数模块，该过程的输出信号会作为计算模块或者加载参数模块的状态机开始信号。当计算模块或者加载参数模块完成的时候又会回传一个完成信号。

## Compute\_3\_3\_Para

状态表：

|  |  |  |
| --- | --- | --- |
| **状态变量** | **变量描述** | **状态码** |
| Idle\_State | 空闲状态 | 6'b00\_0000 |
| Block\_State | 计算状态 | 6'b00\_0010 |

状态转移图：



模块设计图和信号：

|  |  |
| --- | --- |
| **外部参数Parameter** | **数值** |
| WIDTH\_RAM\_ADDR\_TURE | 11 |

| **传输信号** | **方向** | **位宽** | **组** | **信号描述** |
| --- | --- | --- | --- | --- |
| clk | I | 1 | 时钟和复位 | 时钟 |
| rst | I | 1 | 复位 |
| Start | I | 1 | 状态机开始信号 |
| Write\_Block\_Complete | O | 1 | 控制信号 | 计算信号 |
| Ram\_Write\_Num | I | WIDTH\_RAM\_ADDR\_TURE | RAM写数据信号 |
| Ram\_Write\_Addra\_Base | I | WIDTH\_RAM\_ADDR\_TURE | RAM写地址信号 |
| Ram\_Addrb | I | WIDTH\_RAM\_ADDR\_TURE | RAM读数据 |
| Ram\_Data\_Out | O | 128 | RAM数据输出 |
| S\_Data | I | 64 | Slave | 输入数据 |
| S\_Valid | I | 1 | 接收上层数据请求 |
| S\_Ready | O | 1 | 接收上层响应请求 |

|  |  |  |  |
| --- | --- | --- | --- |
| **内部信号** | **信号类型** | **位宽** | **信号描述** |
| Inter\_Write\_Complete | reg | 1 | 计算信号 |

## Quan\_模块

## 基本公式：Out = feature \* weight +bias

对应

（其中weight对应的zero\_point等于0；记bias对应的scale：，bias对应的zero\_point：，所以

于是有：



其中为计算主体，为B

其中scale 是 M, shift 为 2 ，而为B。

也就是说，经过int8类型的feature和weight数据在conv2d 部分计算完成之后，还需要与scale，bias以及zero\_point 这些参数运算，最后才能得到量化后的结果。

模块设计图和信号：

|  |  |
| --- | --- |
| **外部参数Parameter** | **数值** |
| CHANNEL\_OUT\_NUM | 4 |
| WIDTH\_FEATURE\_SIZE | 12 |
| WIDTH\_CHANNEL\_NUM\_REG | 10 |
| WIDTH\_DATA\_ADD | 32 |
| WIDTH\_BIAS\_RAM\_ADDRA | 7 |

| **传输信号** | **方向** | **位宽** | **组** | **信号描述** |
| --- | --- | --- | --- | --- |
| clk | I | 1 | 时钟和复位 | 时钟 |
| rst | I | 1 | 复位 |
| bias\_data\_in | I | WIDTH\_DATA\_ADD \* CHANNEL\_OUT\_NUM | 控制信号 | Bias输入数据信号 |
| scale\_data\_in | I | WIDTH\_DATA\_ADD \* CHANNEL\_OUT\_NUM | Scale输入数据信号 |
| shift\_data\_in | I | WIDTH\_DATA\_ADD \* CHANNEL\_OUT\_NUM | Shift输入数据信号 |
| Zero\_Point\_REG3 | I | 8 | 零点输入数据 |
| bias\_addrb | O | WIDTH\_BIAS\_RAM\_ADDRA | Bias读地址信号 |
| Row\_Num\_Out\_REG | I | WIDTH\_FEATURE\_SIZE | 图片大小 |
| Channel\_Out\_Num\_REG | I | WIDTH\_CHANNEL\_NUM\_REG | 通道数 |
| S\_Data | I | 2\*WIDTH\_DATA\_ADD\*CHANNEL\_OUT\_NUM | Slave | bias\_fifo |
| S\_Valid | I | 1 | 接收上层数据请求 |
| S\_Ready | O | 1 | 接收上层响应请求 |
| M\_Data | O | 2\*CHANNEL\_OUT\_NUM\*8 | Master | 输出数据 |
| M\_Ready | I | 1 | 接收下层数据响应 |
| M\_Valid | O | 1 | 接收下层数据请求 |

|  |  |  |  |
| --- | --- | --- | --- |
| **内部信号** | **信号类型** | **位宽** | **信号描述** |
| scale\_data\_in\_temp | reg | WIDTH\_DATA\_ADD  \*CHANNEL\_OUT\_NUM | Scale输入数据 |

### Conv\_quan\_control

主要是从权重中提取数据，包括Bias，Scale和Zero。

状态表：

|  |  |  |
| --- | --- | --- |
| **状态变量** | **变量描述** | **状态码** |
| Idle\_State | 空闲状态 | 6'b00\_0000 |
| Judge\_Before\_FIFO\_State | 判断是否是fifo之前 | 6'b00\_1000 |
| Judge\_After\_FIFO\_State | 判断是否是fifo之后 | 6'b00\_0001 |
| Compute\_State | 计算状态 | 6'b00\_0010 |

状态转移图：



模块设计图和信号：

|  |  |
| --- | --- |
| **外部参数Parameter** | **数值** |
| CHANNEL\_OUT\_NUM | 8 |
| WIDTH\_FEATURE\_SIZE | 12 |
| WIDTH\_CHANNEL\_NUM\_REG | 10 |
| WIDTH\_DATA\_ADD | 32 |
| WIDTH\_BIAS\_RAM\_ADDRA | 7 |

| **模块内部信号** | **信号类型** | **位宽** | **信号描述** |
| --- | --- | --- | --- |
| Cnt\_Column | reg | WIDTH\_FEATURE\_SIZE | 列计数 |
| Cnt\_Cout | reg | WIDTH\_CHANNEL\_NUM\_REG | 输出通道计数 |
| EN\_Row | wire | 1 | 最后一行 |
| EN\_Last\_Cout | wire | 1 | 最后一个通道 |

| **传输信号** | **方向** | **位宽** | **组** | **信号描述** |
| --- | --- | --- | --- | --- |
| clk | I | 1 | 时钟和复位 | 时钟 |
| rst | I | 1 | 复位 |
| bias\_addrb | O | WIDTH\_BIAS\_RAM\_ADDRA | 控制信号 | Bias读地址信号 |
| EN\_Rd\_Fifo | O | 1 | 读fifo使能 |
| Fifo\_Ready | I | 1 | 往Fifo里存放 |
| Row\_Num\_Out\_REG | I | WIDTH\_FEATURE\_SIZE | 输出第几行 |
| Channel\_Out\_Num\_REG | I | WIDTH\_CHANNEL\_NUM\_REG | 输出第几个通道 |
| M\_Ready | I | 1 | Master | 接收下层数据响应 |
| M\_Valid | O | 1 | 接收下层数据请求 |

### Conv\_Bias

偏置是加在每组通道的卷积计算结果之后，因此每个像素点需要偏置的数量为：Channel\_Times =输出通道数/每组通道的数量。这个小模块是通过一个FIFO来控制的，每次写入或者读出一行数据所需要的偏置数量，与卷积之后的结果做加法运算。每次读写的偏置数量是：Cnt\_Column \*Channel\_Times;

模块设计图和信号：

|  |  |
| --- | --- |
| **外部参数Parameter** | **数值** |
| CHANNEL\_OUT\_NUM | 8 |
| WIDTH\_FEATURE\_SIZE | 12 |
| WIDTH\_CHANNEL\_NUM\_REG | 10 |
| WIDTH\_DATA\_ADD | 32 |

| **模块内部信号** | **信号类型** | **位宽** | **信号描述** |
| --- | --- | --- | --- |
| Channel\_Times | Wire | WIDTH\_CHANNEL\_NUM\_REG | 通道次序 |
| data\_fifo\_out | Wire | 2\*WIDTH\_DATA\_ADD\*CHANNEL\_OUT\_NUM | Fifo输出数据 |

| **传输信号** | **方向** | **位宽** | **组** | **信号描述** |
| --- | --- | --- | --- | --- |
| clk | I | 1 | 时钟和复位 | 时钟 |
| rst | I | 1 | 复位 |
| fifo\_ready | O | 1 | 控制信号 | 往fifo里写数据 |
| rd\_en\_fifo | I | 1 | 读fifo使能 |
| bias\_data\_in | I | WIDTH\_DATA\_ADD \*CHANNEL\_OUT\_NUM | Bias输入数据信号 |
| Channel\_Out\_Num\_REG | I | WIDTH\_CHANNEL\_NUM\_REG | 输出第几个通道 |
| Row\_Num\_Out\_REG | I | WIDTH\_FEATURE\_SIZE | 输出第几行 |
| S\_Data | I | 2\*WIDTH\_DATA\_ADD\*CHANNEL\_OUT\_NUM | Slave |  |
| S\_Valid | I | 1 | 接收上层数据请求 |
| S\_Ready | O | 1 | 接收上层响应请求 |
| M\_Data | O | 1 | Master | 输出数据 |

### Conv\_Scale

模块设计图和信号：

|  |  |
| --- | --- |
| **外部参数Parameter** | **数值** |
| CHANNEL\_OUT\_NUM | 8 |
| WIDTH\_DATA\_ADD | 32 |

| **传输信号** | **方向** | **位宽** | **组** | **信号描述** |
| --- | --- | --- | --- | --- |
| clk | I | 1 | 时钟和复位 | 时钟 |
| rst | I | 1 | 复位 |
| Scale\_Data\_In | I | WIDTH\_DATA\_ADD \*CHANNEL\_OUT\_NUM | 控制信号 | Scale输入数据 |
| Scale\_Data\_Out | O | 2\*CHANNEL\_OUT\_NUM  \*WIDTH\_DATA\_ADD | Scale输出数据 |
| S\_Data | I | 2\*WIDTH\_DATA\_ADD\*CHANNEL\_OUT\_NUM | Slave | 完成conv2d计算的数据 |

### Conv\_Zero

模块设计图和信号：

|  |  |
| --- | --- |
| **外部参数Parameter** | **数值** |
| CHANNEL\_OUT\_NUM | 8 |

| **模块内部信号** | **信号类型** | **位宽** | **信号描述** |
| --- | --- | --- | --- |
| data\_out\_temp | Wire | 2\*CHANNEL\_OUT\_NUM |  |
| data\_out\_relu | Reg | CHANNEL\_OUT\_NUM |  |
| data\_out\_judge | Wire | CHANNEL\_OUT\_NUM |  |

| **传输信号** | **方向** | **位宽** | **组** | **信号描述** |
| --- | --- | --- | --- | --- |
| clk | I | 1 | 时钟和复位 | 时钟 |
| shift\_data\_in | I | 2\*CHANNEL\_OUT\_NUM\*8\*2 | 控制信号 | Shift输入数据信号 |
| zero\_data\_in | I | 7 | Zero输入数据信号 |
| data\_out | O | 2\*CHANNEL\_OUT\_NUM\*8 | 输出数据 |