**CevaXM4 Data DMA Behavior**

# DTCM 结构



# Direction

1. DTCM to DTCM
2. DTCM to External Memory
3. External Memory to DTCM

# Transfer Type

1. Message
2. Linear

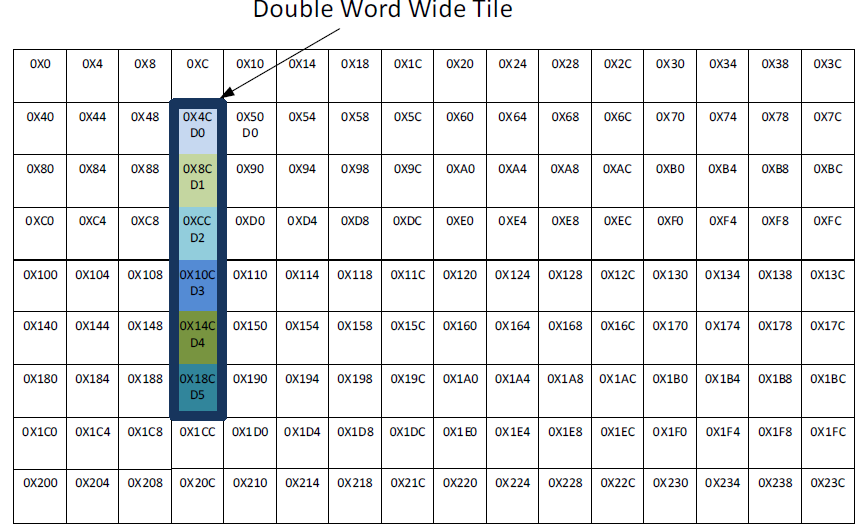
参数：起始地址，搬移的BYTE数目

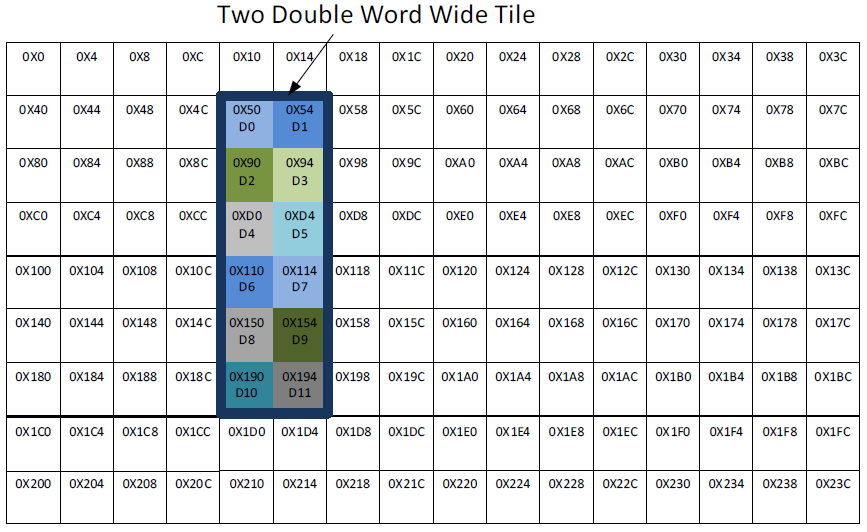
1. 2-dimensional

任意二维的块，确定具体块的参数如下

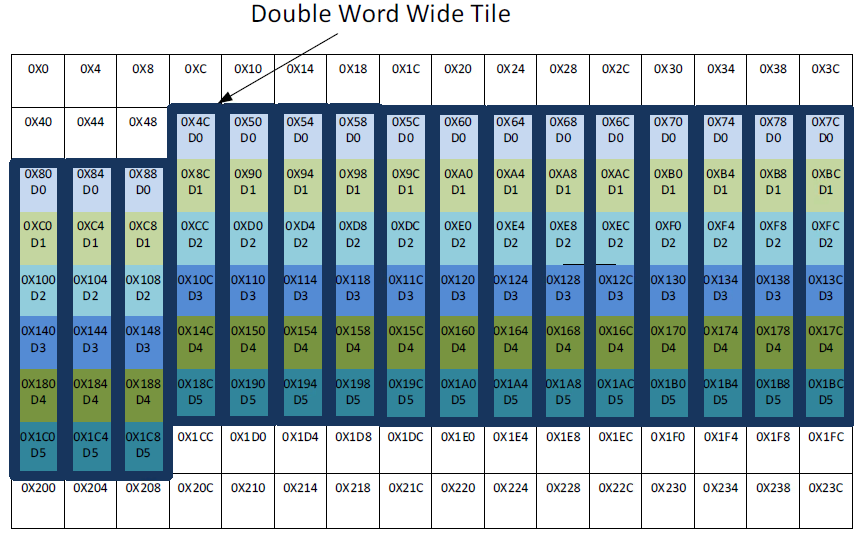


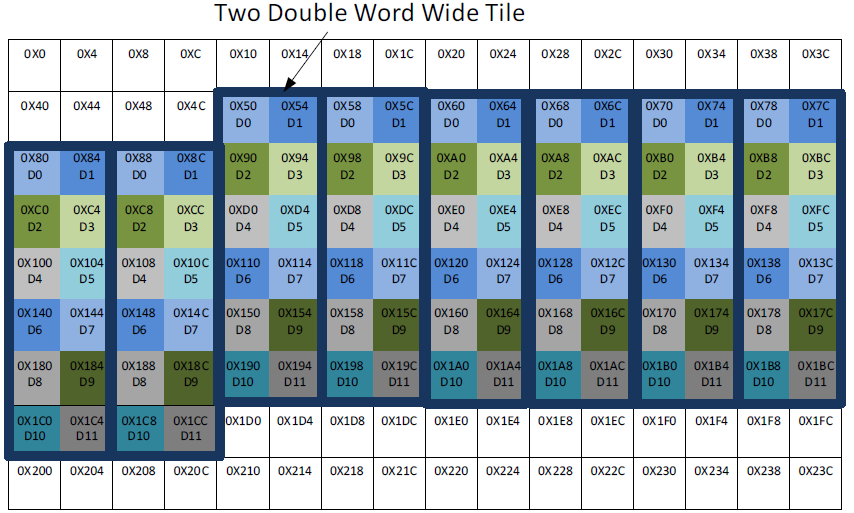
1. One-bank/Two-bank Tile
2. Source或者Destination必须至少有一个是DTCM
3. Tile的宽度必须是32bit(One-bank)/64bit(Two-bank)，且必须落在DTCM的一个bank内或两个bank内





1. Duplicate Bank Transfer
2. Destination 必须是DTCM
3. Tile的宽度必须是32bit(One-bank)/64bit(Two-bank)，且必须落在DTCM的一个bank内或两个bank内





# DDMA Task Descriptor

1. DDMA task descriptor主要是以下寄存器

DDEA, DDIA, DDTC, 2DCFG1, 2DCFG2

1. Descriptor有两种方法配置：CPM（高优先级），QMAN（低优先级）
2. DDTC最后配置，配置完DDMA启动
3. 有专门的queue可以缓存三个task
4. Exact Read：通过DACU的寄存器ADDx\_ATT1配置，以Region为单位
5. Max Burst：既可以以Region为单位，通过DACU的寄存器配置（高优先级），也可以以task为单位，通过DDTC来配置（低优先级）。
6. Outstanding burst：通过DACU的寄存器ADDx\_ATT1，MSS\_DACC配置，以Region为单位

# DDMA External Memory Access Behavior

1. Read:三种burst完成dma task
   1. Normal Size Bursts：根据配置的最大burst length传输
   2. Last Burst：用一个burst（length大于1，小于最大burst length）来传输
   3. Partial DMAW Single

Exact\_RD=0: 用一个length=1的burst，允许的最小beat size来完成DMA的传输。（所用cycle数最少）

Exact\_RD=0: 用很多个length=1的burst，最接近剩余需传输bit数的允许的beat size来完成DMA的传输。（避免读取冗余的数据）

1. Write：三种burst完成dma task
   1. Mximum Size Bursts：根据配置的最大burst length传输
   2. Full DMAW Burst：用一个burst（length大于1，小于最大burst length）来传输
   3. Partial DMAW Single

用一个length=1的burst，允许的最小beat size来完成

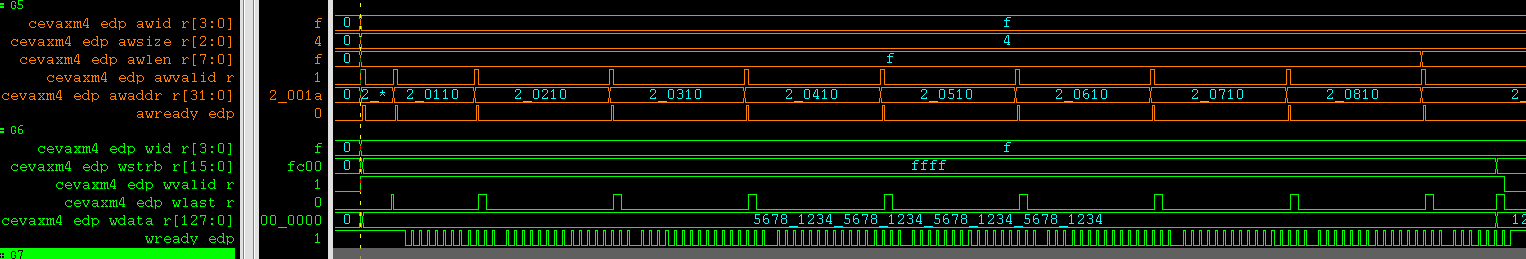
## Example

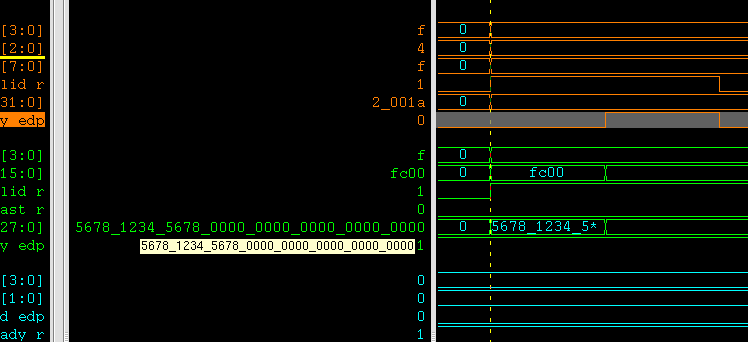
EDP的data width=128bit，little endian

### Linear

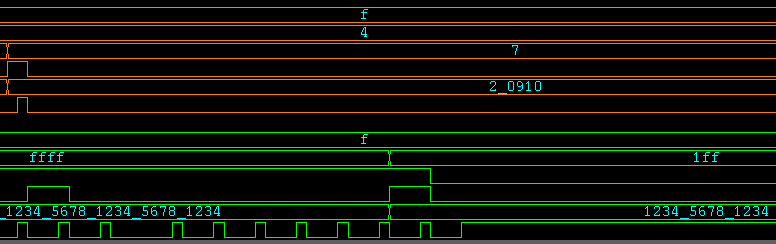
#### （1） External Memory Write

|  |  |  |  |
| --- | --- | --- | --- |
| Direction | External start address | Total Bytes | Max burst length |
| DTCM to external | 0x2001a | 2415 | 0xf |





* 第一笔burst的第一个beat：如上图，burst\_length=f，可以看出因为起始地址为0x2001a，没有16Byte对齐，所以这笔数据的wstrb=0xfc00，只写入0x2001a-0x2001f 六个Bytes，wdata中另外10个Byte都为0。
* 第一笔burst的其他beats：正常，burst\_length=f，wstrb=0xffff，每个beat写入16个Bytes。
* 中间每笔burst的beats：正常，burst\_length=f，wstrb=0xffff，每个beat写入16个Bytes。
* 最后一笔burst：如下图，burst length=7，其中前7个beats正常，wstrb=0xffff，每个beat写入16个Byte；最后一个beat，wstrb=0x1ff，写入9个Bytes，wdata中另外7个Byte为0。



（有burst的size均为4，即16Bytes）

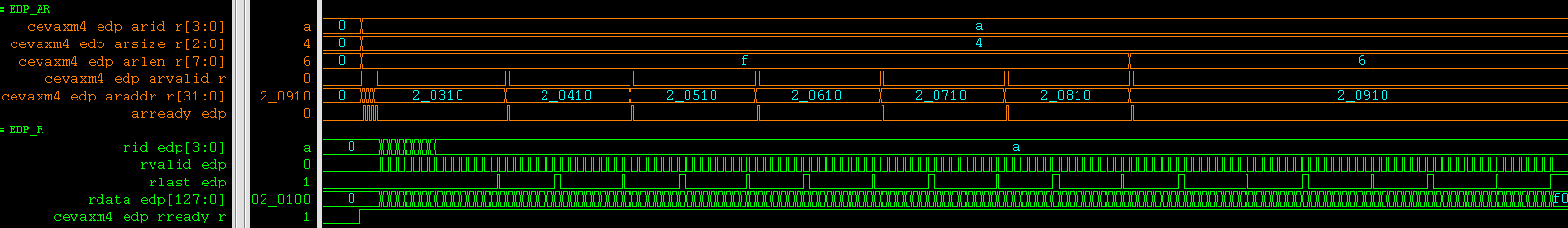
所以，总共传输Byte为

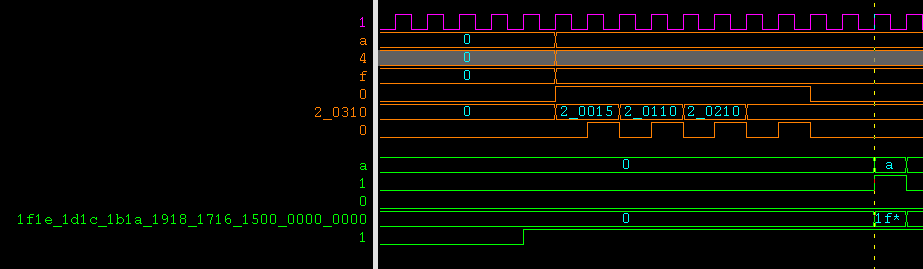
（6+15x16）(第一个burst)+（8x16x16）（中间八笔burst）+（7x16+9）（最后一笔burst）=2415

#### （2） External Memory Read

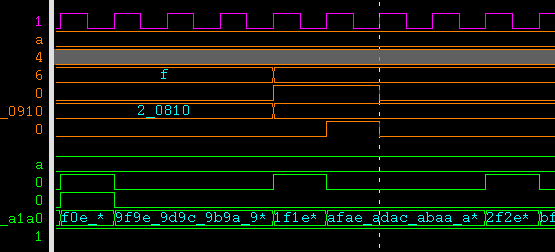
|  |  |  |  |
| --- | --- | --- | --- |
| Direction | External start address | Total Bytes | Max burst length |
| External to DTCM | 0x20015 | 2397 | 0xf |

##### ①EXACT\_RD=0



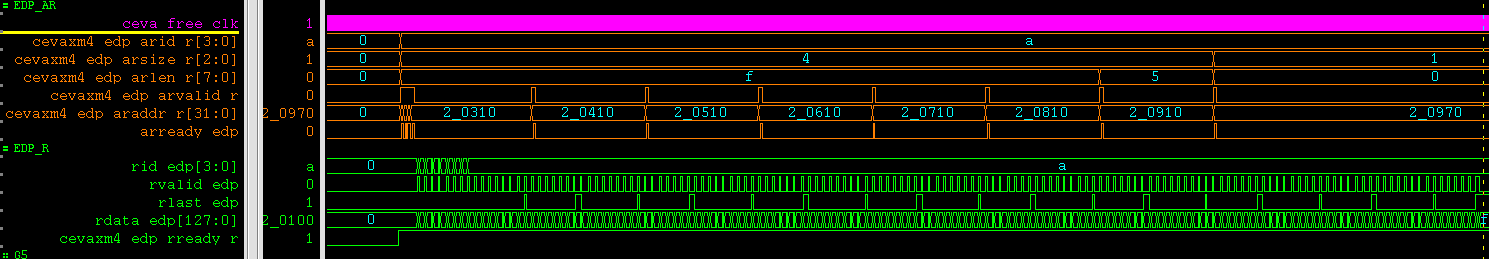


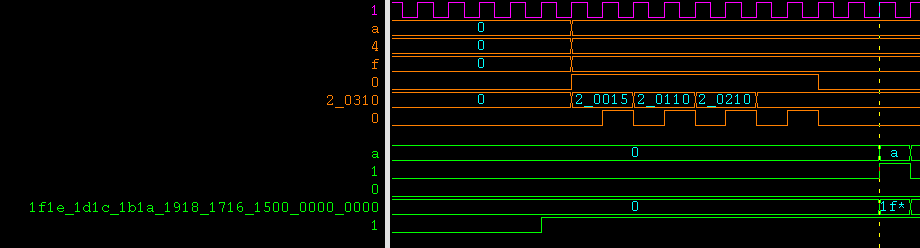
* 第一笔burst的第一个beat：如上图，burst\_length=f，可以看出起始地址为0x20015，没有16Byte对齐，读进来的rdata只有高11个Bytes有值，其他为0，即读进了0x20015-0x2001f地址的数据。
* 第一笔burst的其他beats：正常，burst\_length=f，每个beat读进16个Bytes。
* 中间每笔burst的beats：正常，burst\_length=f，每个beat读16个Bytes。
* 最后一笔burst之前一共读进Byte数为：(11+15x16)+8x16x16=2299，需要读进Byte数：2397-2299=98，介于7x16=112和6x16=96之间，所以最后一笔burst如下图，burst length=6，即传7个beat，最后一个beat只有2个Bytes有效，但是波形上看rdata把16个Bytes都读出来，应该是在内部选了2个Bytes。



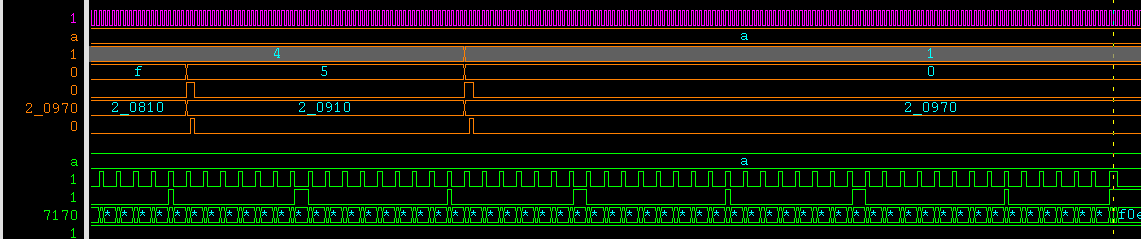
（有burst的size均为4，即16Bytes）

##### ②EXACT\_RD=1





* 第一笔burst：上图如①
* 中间每笔burst：如①
* 最后一笔burst之前一共读进Byte数为：(11+15x16)+8x16x16=2299，还需要读进Byte数：2397-2299=98，介于7x16=112和6x16=96之间，因为采用了exact read，所以最后如下图，有一笔burst length=5，burst size=4的burst，还有一笔burst length=0，burst size=1的burst，共读进6x16+1x2=98个Bytes，没有读进任何冗余的数据。

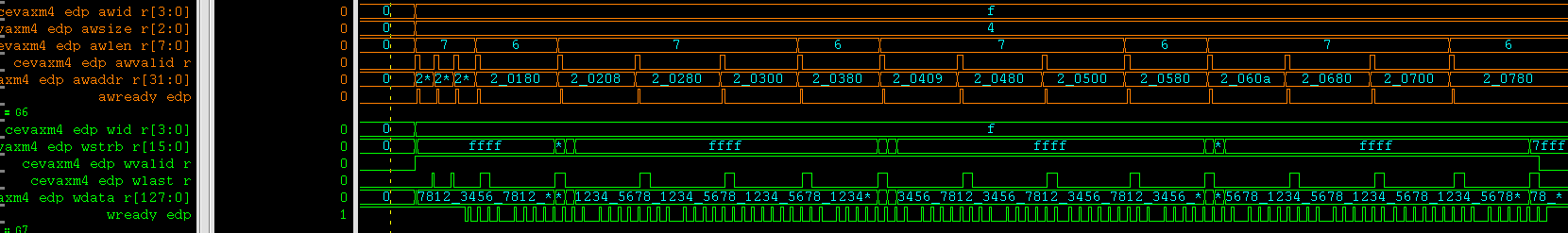


### 2-dimentional

将块的每一行按照上述的linear处理

#### （1）External Memory Write

|  |  |
| --- | --- |
| Direction | DTCM to External memory |
| External Start Address | 0x20007 |
| Max burst length | 0x7 |
| Hight | 0x4 |
| Width | 0x1e5 |
| Int\_Stride | 0x1f0 |
| Ext\_Stride | 0x201 |



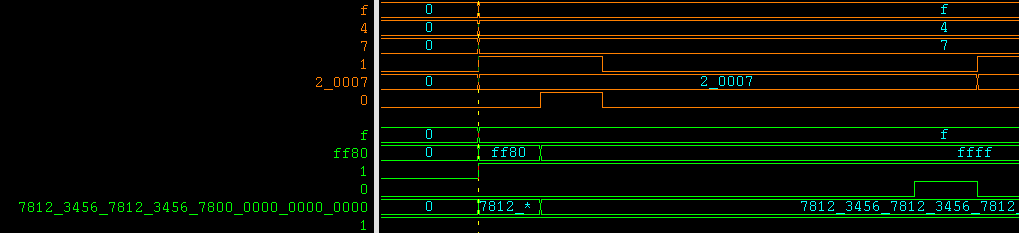
整体上看:

* 四行，其中每行都用了三笔burst\_len=7，一笔burst\_len=6的传输来完成
* 四行的首地址分别是0x20007，0x20208，0x20409，0x2060a。

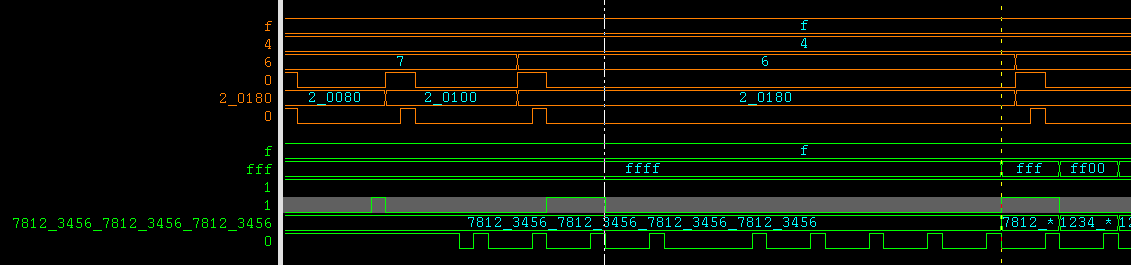
每行的传输：

第一行：（有burst的size均为4，即16Bytes）

* 第一笔burst的第一个beat：如下图，可以看出因为起始地址为0x20007，没有16Byte对齐，所以这笔数据的wstrb=0xff80，只写入0x20007-0x2000f 九个Bytes，wdata中另外7个Byte都为0。



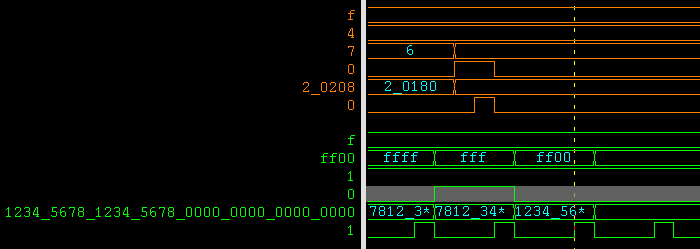
* 第一笔burst的其他beats：正常，burst\_len=7，wstrb=0xffff，每个beat写入16个Bytes。
* 中间两笔burst的beats：正常，burst\_len=7，wstrb=0xffff，每个beat写入16个Bytes。
* 最后一笔burst： 如下图，burst\_len=6，前6个beat正常，wstrb=0xffff，每个beat写入16个Bytes；最后一个beat，wstrb=0x0fff，写入12个Bytes，wdata中另外4个Byte为0。



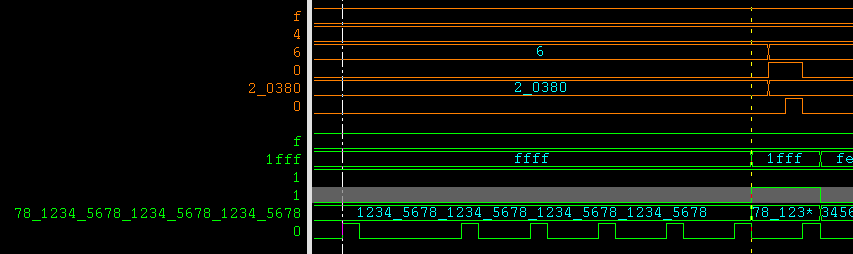
第一行共写入Byte：9+7x16+2x8x16+6x16+12=485=0x1e5

第二行：（有burst的size均为4，即16Bytes）

* 第一笔burst的第一个beat：如下图，可以看出因为起始地址为0x20208，没有16Byte对齐，所以这笔数据的wstrb=0xff00，只写入0x20208-0x2020f 八个Bytes，wdata中另外8个Byte都为0。



* 第一笔burst的其他beats：正常，burst\_len=7，wstrb=0xffff，每个beat写入16个Bytes。
* 中间两笔burst的beats：正常，burst\_len=7，wstrb=0xffff，每个beat写入16个Bytes。
* 最后一笔burst： 如下图，burst\_len=6，前6个beat正常，wstrb=0xffff，每个beat写入16个Bytes；最后一个beat，wstrb=0x1fff，写入13个Bytes，wdata中另外3个Byte为0。



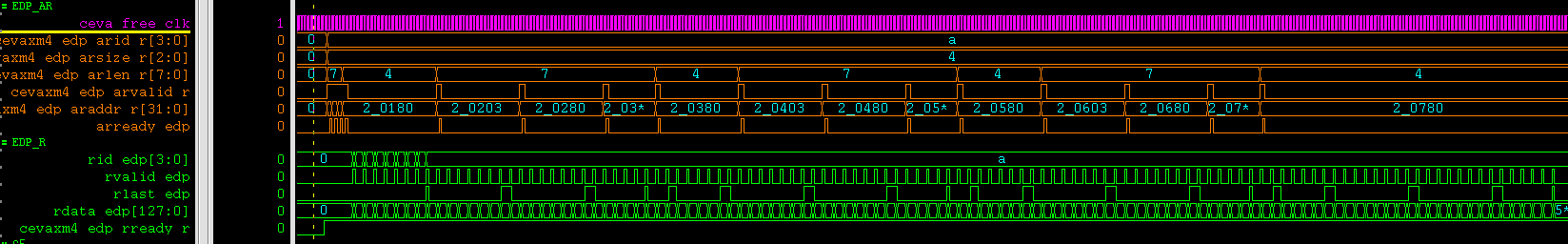
第一行共写入Byte：8+7x16+2x8x16+6x16+13=485=0x1e5

（第三行第四行以此类推）

#### （2）External Memory Read

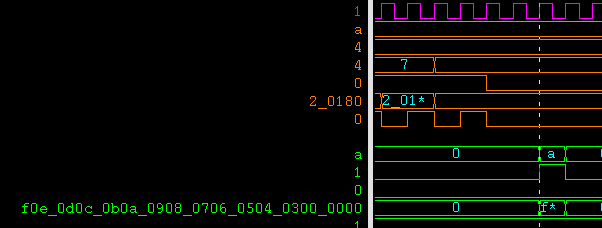
|  |  |
| --- | --- |
| Direction | External memory to DTCM |
| External Start Address | 0x20003 |
| Max burst length | 0x7 |
| Hight | 0x4 |
| Width | 0x1c7 |
| Int\_Stride | 0x1f0 |
| Ext\_Stride | 0x200 |

##### Exact\_RD=0



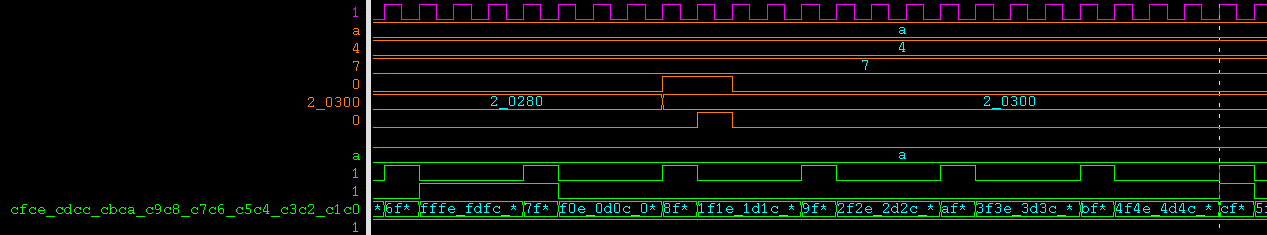
整体上看:

* 四行，其中每行都用了三笔burst\_len=7，一笔burst\_len=4的传输来完成
* 四行的首地址分别是0x20003，0x20203，0x20403，0x20603。



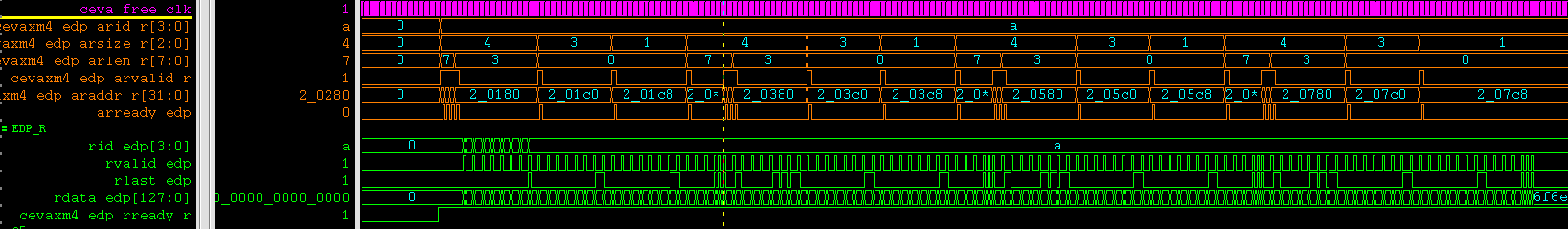
每行的传输：

* 第一笔burst的第一个beat：如上图，burst\_length=7，可以看出起始地址为0x20003，没有16Byte对齐，读进来的rdata只有高13个Bytes有值，其他为0，即读进了0x20003-0x2000f地址的数据。
* 第一笔burst的其他beats：正常，burst\_length=7，每个beat读进16个Bytes。
* 中间两笔burst的beats：正常，burst\_length=7，每个beat读16个Bytes。
* 最后一笔burst之前一共读进Byte数为：(13+7x16)+2x8x16=381，需要读进Byte数：455-381=74，介于4x16=64和5x16=80之间，所以最后一笔burst如下图，burst length=4，即传5个beats，最后一个beat只有10个Bytes有效，但是波形上看rdata把16个Bytes都读出来，应该是在内部选了10个Bytes。



（有burst的size均为4，即16Bytes）

##### Exact\_RD=1



整体上看:

* 四行，其中每行都用了三笔burst\_len=7，一笔burst\_len=3，两笔burst\_len=0的传输来完成
* 四行的首地址分别是0x20003，0x20203，0x20403，0x20603。

每行的传输：

* 第一笔burst的第一个beat：如①。
* 第一笔burst的其他beats：如①。
* 中间两笔burst的beats：如①
* 前三笔burst之前一共读进Byte数为：(13+7x16)+2x8x16=381，需要读进Byte数：455-381=74，介于4x16=64和5x16=80之间，因为采用了exact read， 所以最后有一笔burst\_len=3，burst\_size=4的burst，一笔burst\_len=0，burst\_size=3的burst，和一笔burst\_len=0，burst\_size=1的burst，共读进4x16+1x8+1x2=74个Bytes，没有读进任何冗余的数据。

