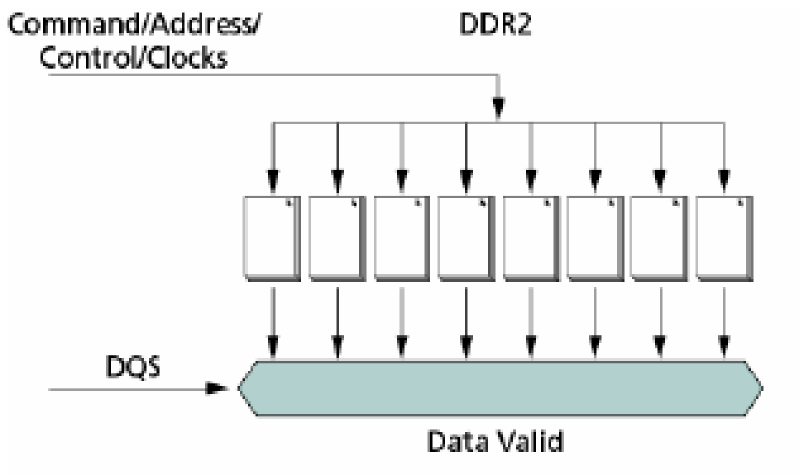
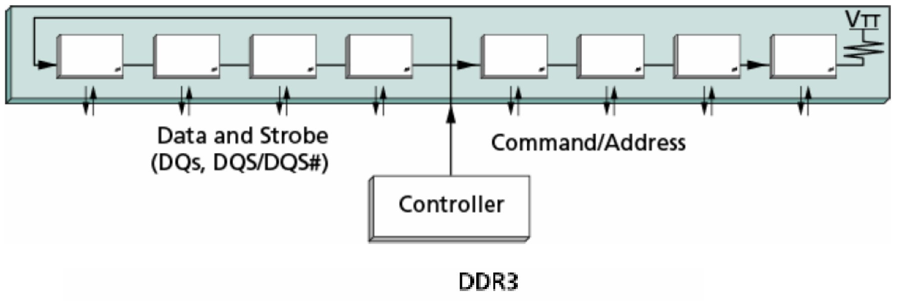
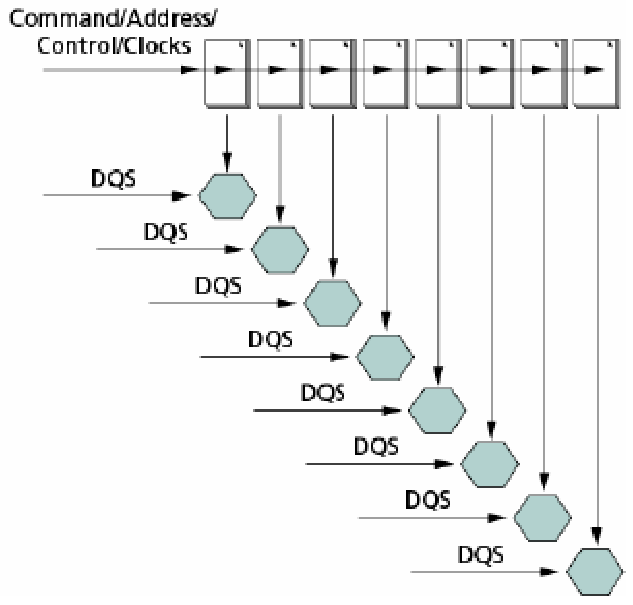
# 提纲

- 内存相关概念介绍
- 内存控制器结构及参数介绍
- PMON下内存调试
- 内存训练程序
- 内存信号测量

## 内存走线拓扑结构

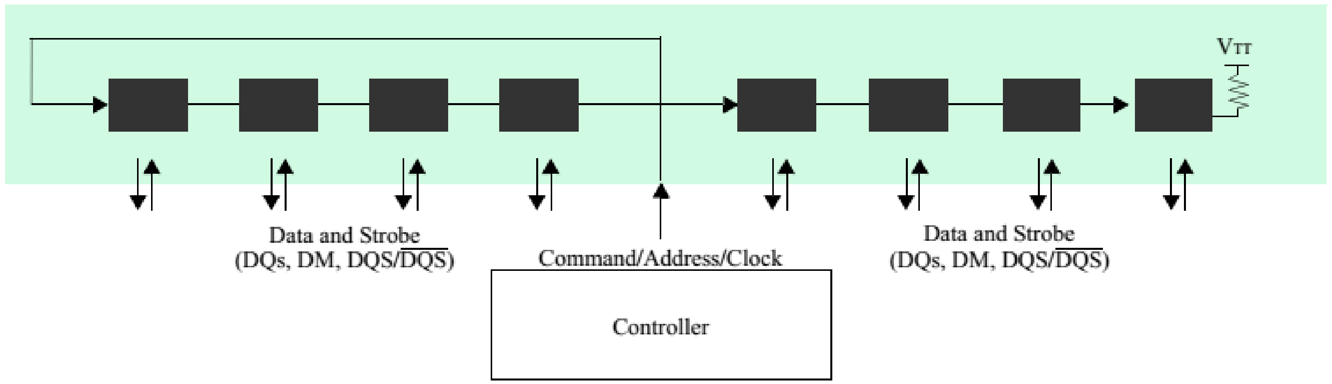

Command/Address/
Control/Clocks
DDR2
DQS
Data Valid


VTT
Data and Strobe
(DQs, DQS/DQS#)
Command/Address
Controller
DDR3


Command/Address/
Control/Clocks
DQS
DQS
DQS
DQS
DQS
DQS
DQS
DQS

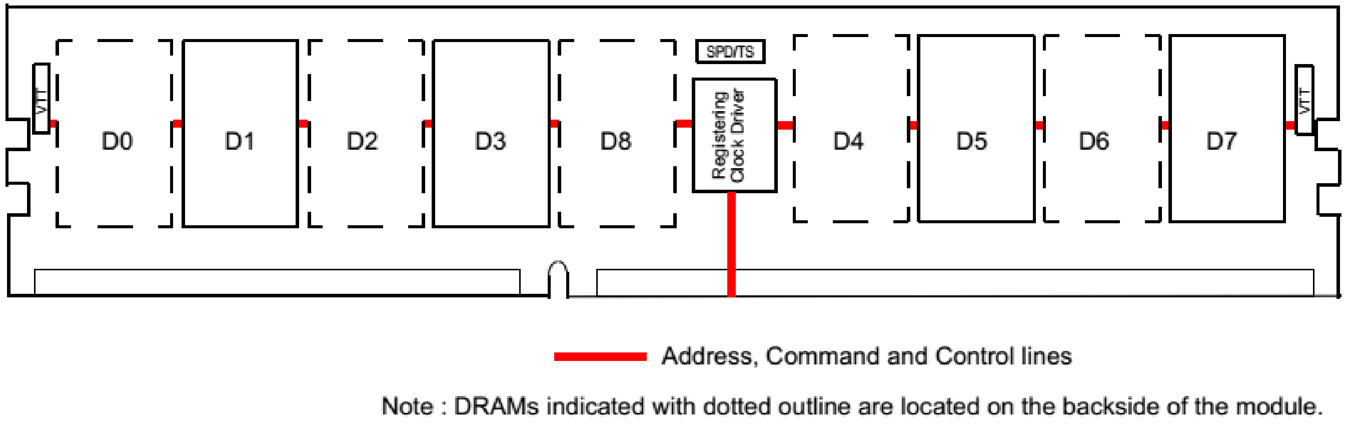
## UDIMM和RDIMM走线区别

UDIMM



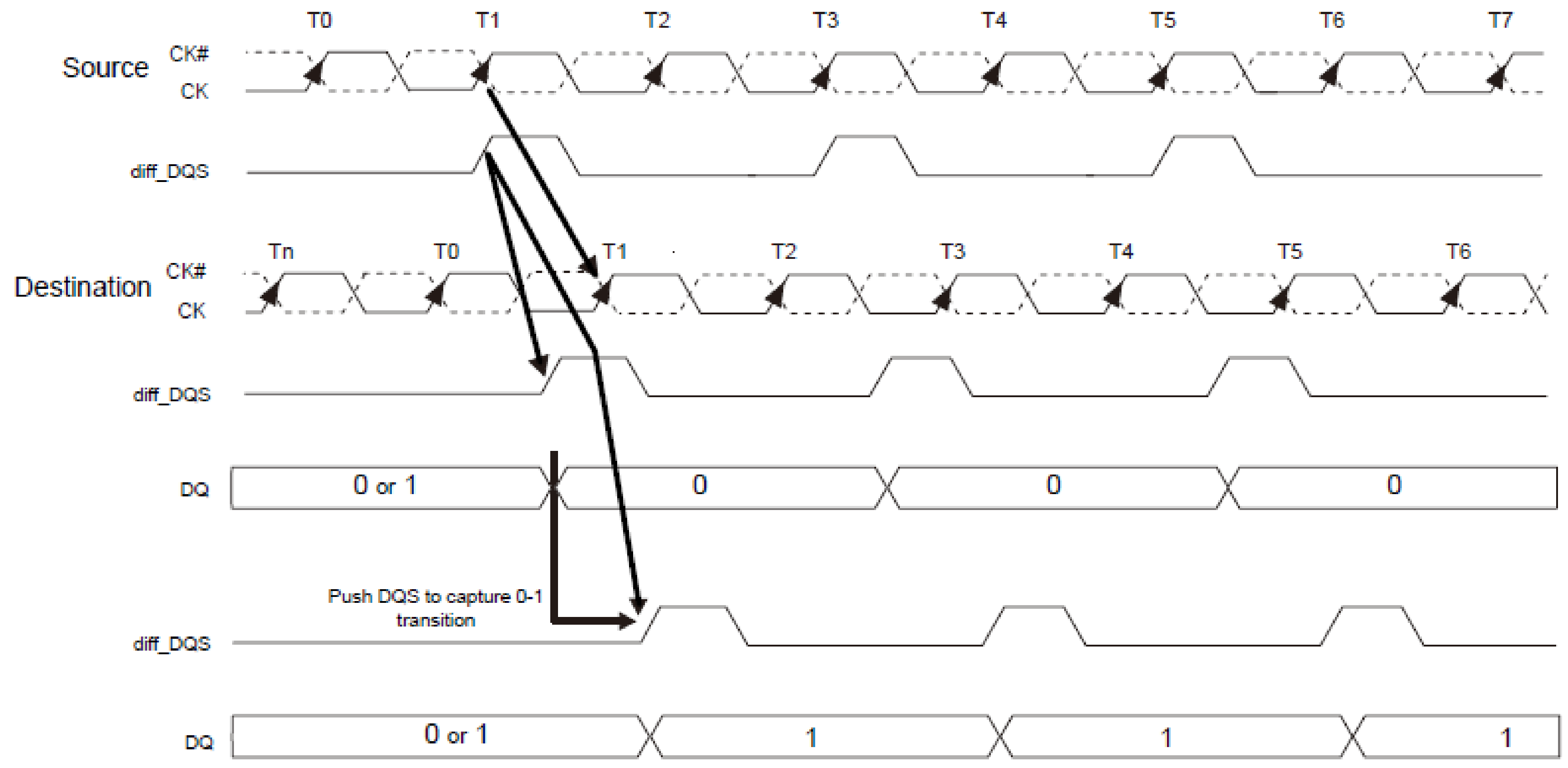
VTT
Data and Strobe
(DQs, DM, DQS/$\overline{\text{DQS}}$)
Command/Address/Clock
Data and Strobe
(DQs, DM, DQS/$\overline{\text{DQS}}$)
Controller

RDIMM

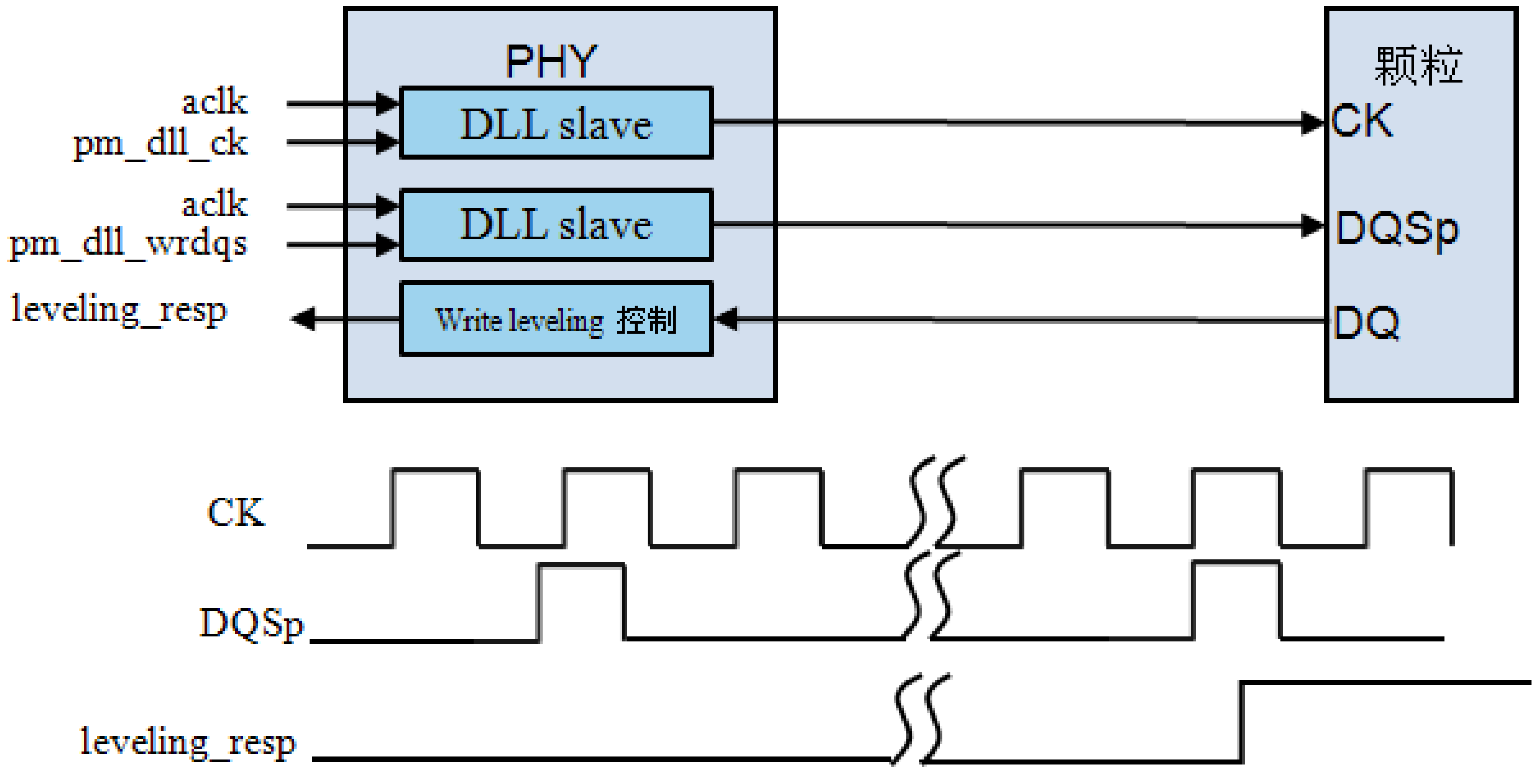


Note : DRAMs indicated with dotted outline are located on the backside of the module.

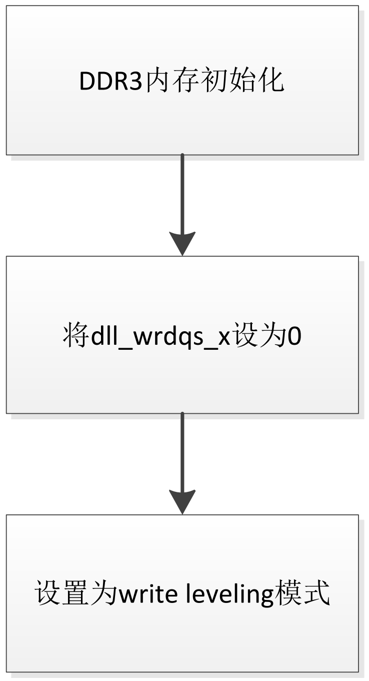
## Write leveing示意图



## Write leveling实现框图



## Write leveling软件流程



1. 完成控制器初始化；

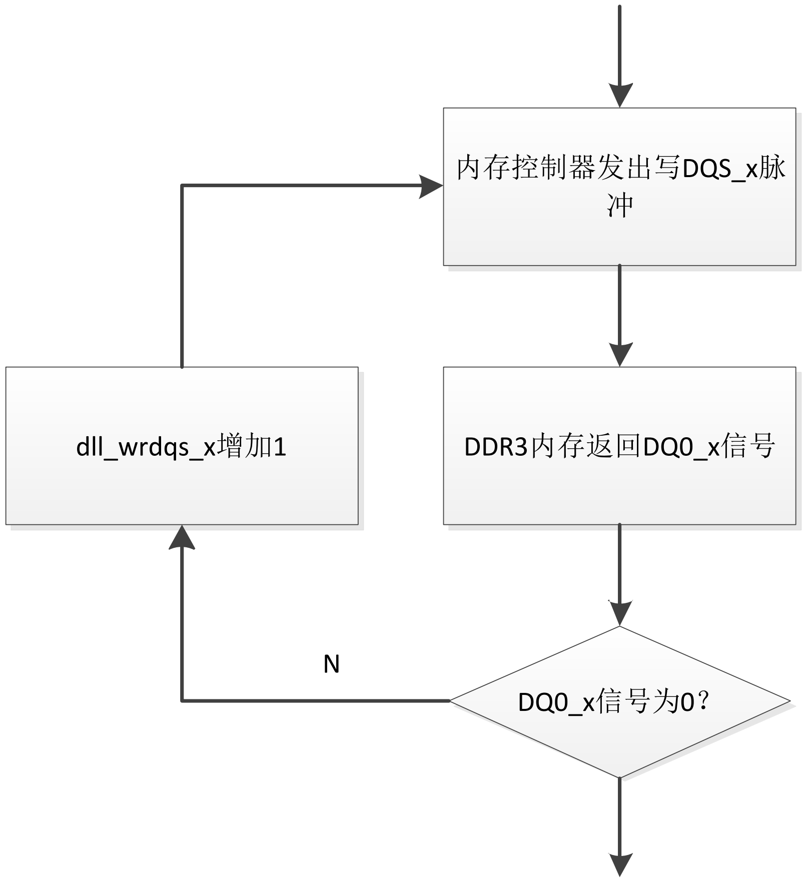
2. 将~~hardware_pd3/2/1/0(0x1f8)设置为4’b0，cs_resync、cs_zq(0x168)设置为4’b0，ref_sch_en(0x340)设置为1’b0；~~

3.将Dll_wrdqs_x（x = 0…8）设置为0；

4. 设置Lvl_mode（0x180）为2’b01;

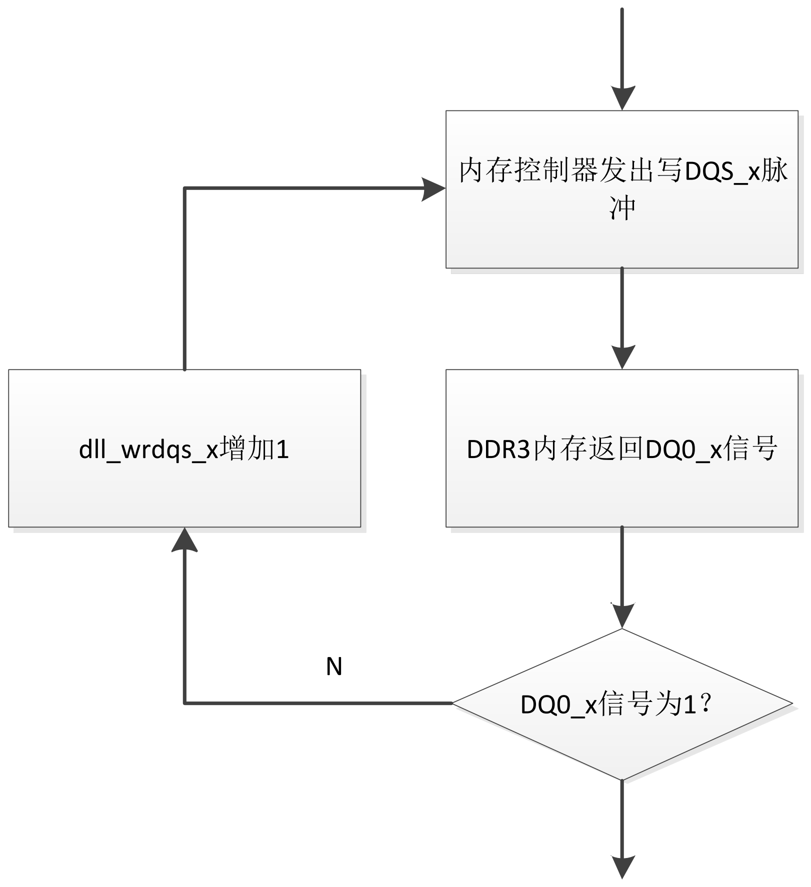
5. 采样Lvl_ready（0x185）寄存器，如果为1，表示可以开始Write Leveling请求；

## Write leveling软件流程



6.开始训练一个Byte，设置Lvl_req（0x181）为1；
7.采样Lvl_done（0x186）寄存器，如果为1，表示一次Write Leveling请求完成；
8.采样对应Byte的Lvl_resp_x（0x187-0x18f）寄存器，如果为0则跳至步骤9。否则（采样Lvl_resp_x结果为1），将对应的Dll_wrdqs_x[6:0]增加1，并重复执行步骤6-8直至采样Lvl_resp_x结果为0；

# Write leveling软件流程



6.开始训练一个Byte，设置Lvl_req（0x181）为1；

7.采样Lvl_done（0x186）寄存器，如果为1，表示一次Write Leveling请求完成；

8.采样对应Byte的Lvl_resp_x（0x187-0x18f）寄存器，如果为0则跳至步骤9。否则（采样Lvl_resp_x结果为1），将对应的Dll_wrdqs_x[6:0]增加1，并重复执行步骤6-8直至采样Lvl_resp_x结果为0；

9.采样对应Byte的Lvl_resp_x（0x187-0x18f）寄存器，如果为0则将对应的Dll_wrdqs_x[6:0]增加1，并重复执行步骤6，7，9直至采样Lvl_resp_x结果为1；

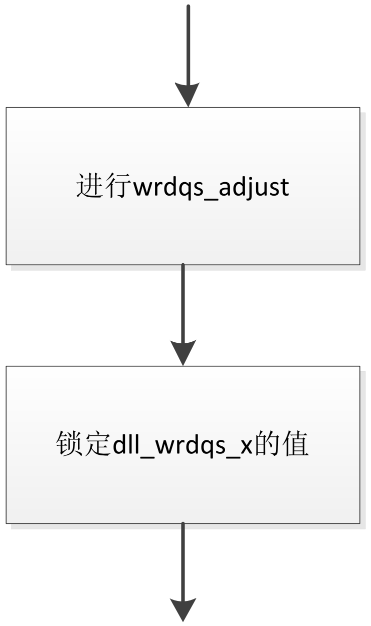
10.至此可能已经找到CLK的边沿，为了保证步骤9的采样为可靠结果，需要重复步骤6，7，9数次（该次数由WR_FILTER_LENGTH宏定义确定）。

11.将Dll_wrdqs_x[6:0]减去WR_FILTER_LENGTH

12.重复执行步骤6-11，训练下一个Byte。

13.此时所有Dll_wrdqs_x的值设置正确；

## Write leveling软件流程



14.对每一个处在特定的区间的Dll_wrdqs_x[6:0]进行微调。调整方式为：

Dll_wrdqs_x[6:0]在[0x00,0x08)范围内时设置为0x08；

Dll_wrdqs_x[6:0]在[0x20,0x28)范围内时设置为0x28；

Dll_wrdqs_x[6:0]在[0x40,0x48)范围内时设置为0x48；

Dll_wrdqs_x[6:0]在[0x60,0x68)范围内时设置为0x68；

Dll_wrdqs_x[6:0]在(0x18,0x1F]范围内时设置为0x18；

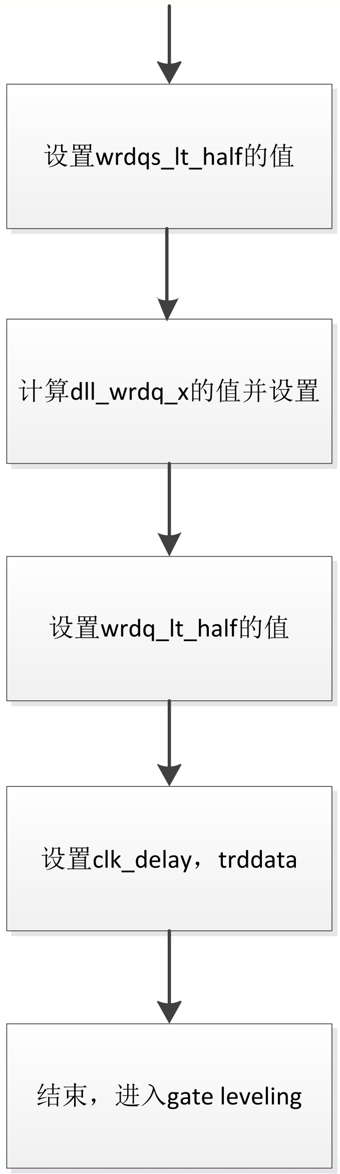
Dll_wrdqs_x[6:0]在(0x38,0x3F]范围内时设置为0x38；

Dll_wrdqs_x[6:0]在(0x58,0x5F]范围内时设置为0x58；

Dll_wrdqs_x[6:0]在(0x78,0x7F]范围内时设置为0x78；

由宏定义WRDQS_ADJUST_LOOP决定是否执行该步骤

## Write leveling软件流程



15.设置Wrdqs_lt_halt_x，如果Dll_wrdqs_x的值小于WRDQS_LTHF_STD，则将对应的Wrdqs_lt_half_x置为1，否则置为0；

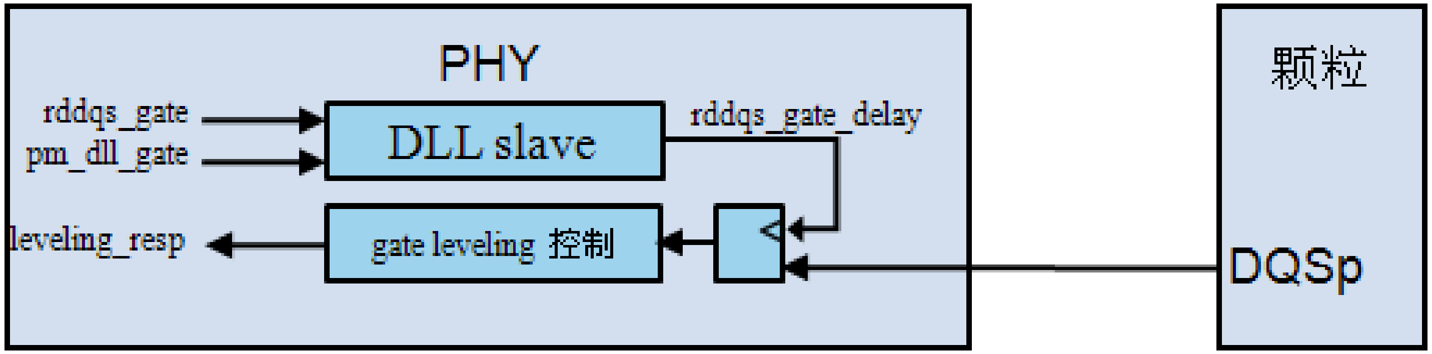
16.设置Dll_wrdq_x，用Dll_wrdqs_x[6:0]减去宏定义DLL_WRDQ_SUB

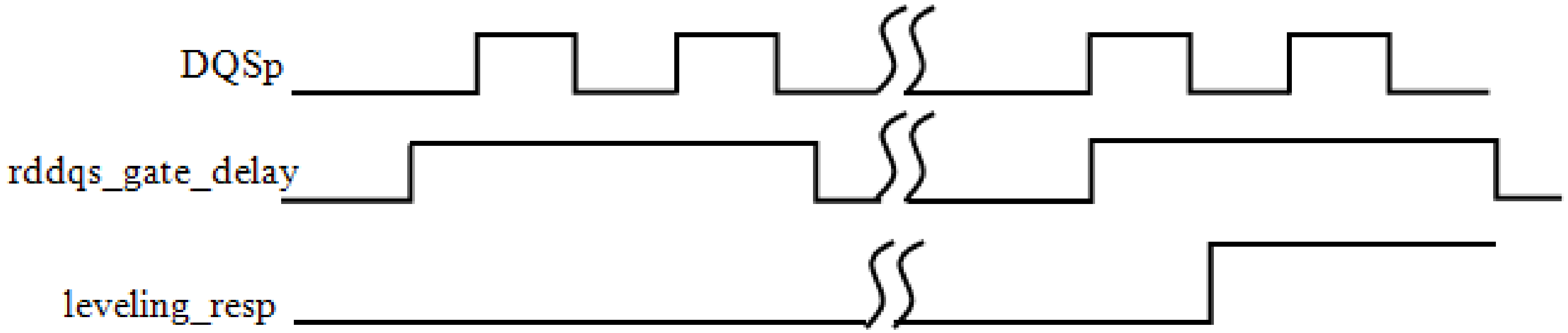
17.设置Wrdq_lt_half_x，如果Dll_wrdq_x小于WRDQ_LTHF_STD，则将对应的Wrdq_lt_half_x置为1，否则置为0

18.根据颗粒顺序(由ORDER_OF_XDIMM宏定义确定)逐个检测Wrdq_lt_half_x的值。如果所有的Wrdq_lt_half_x都为1，则将tPHY_WRLAT(0x1d4)和tPHY_RDDATA(0x1c0)减1；如果检测过程中出现了Wrdq_lt_half_x从1变为0的情况，则从第一个变为0的颗粒开始，所有后面颗粒对应的Wrdq_clkdelay_x设置为1，然后将tPHY_WRLAT(0x1d4)和tPHY_RDDATA(0x1c0)减1；如果所有Wrdq_lt_half_x都为0，不做任何处理。对于RDIMM类型的内存条，需要对寄存器两端的内存颗粒分别进行处理。

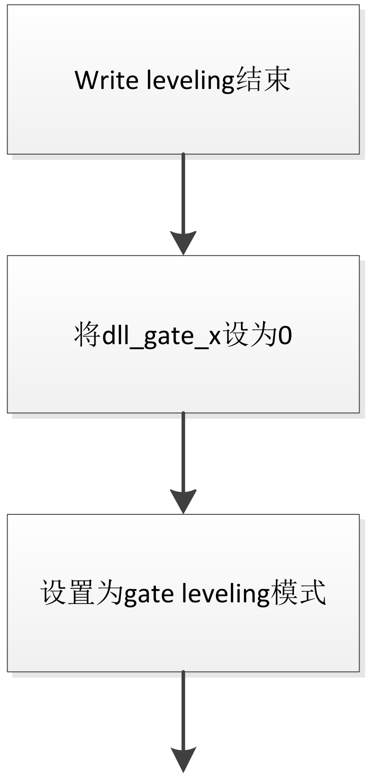
19.将Lvl_mode（0x180）设置为2’b00，退出Write Leveling模式；

## gate leveling实现框图





## gate leveling软件流程



1.完成控制器初始化；

2.完成Write Leveling;

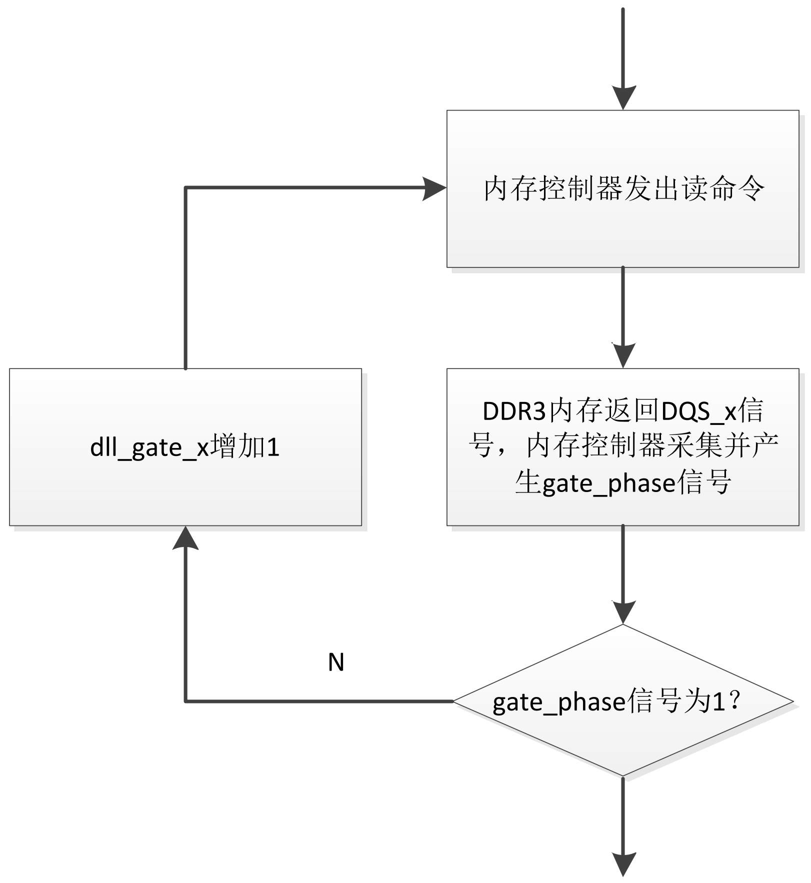
~~3.将Cs_zq(0x168)设置为4’b0；~~

4.将Dll_gate_x（x = 0…8）设置为0；

5.设置Lvl_mode（0x180）为2’b10；

6.采样Lvl_ready（0x180）寄存器，如果为1，表示可以开始Gate Leveling请求；

## gate leveling软件流程

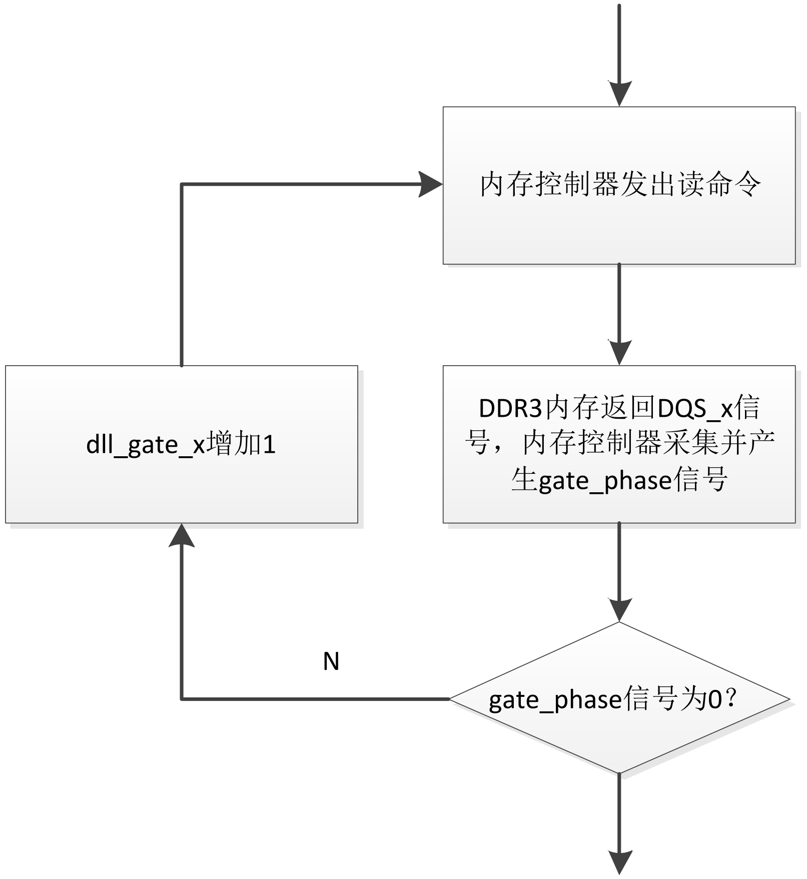


7.开始训练一个Byte，设置Lvl_req（0x180）为1；

8.采样Lvl_done（0x180）寄存器，如果为1，表示一次Gate Leveling请求完成；

9.采样对应Byte的Lvl_resp_x[0]（0x187-0x18f）寄存器，如果为0则跳至步骤10。否则（采样Lvl_resp_x[0]结果为1），将对应的Dll_gate_x[6:0]增加1，并重复执行步骤7-9直至采样结果为0；

# gate leveling软件流程



7.开始训练一个Byte，设置Lvl_req（0x180）为1；

8.采样Lvl_done（0x180）寄存器，如果为1，表示一次Gate Leveling请求完成；

9.采样对应Byte的Lvl_resp_x[0]（0x187-0x18f）寄存器，如果为0则跳至步骤10。否则（采样Lvl_resp_x[0]结果为1），将对应的Dll_gate_x[6:0]增加1，并重复执行步骤7-9直至采样结果为0；

10.采样对应Byte的Lvl_resp_x[0]（0x187-0x18f）寄存器，如果为0则将对应的Dll_gate_x[6:0]增加1，并重复执行步骤7，8，10直至采样结果为1；

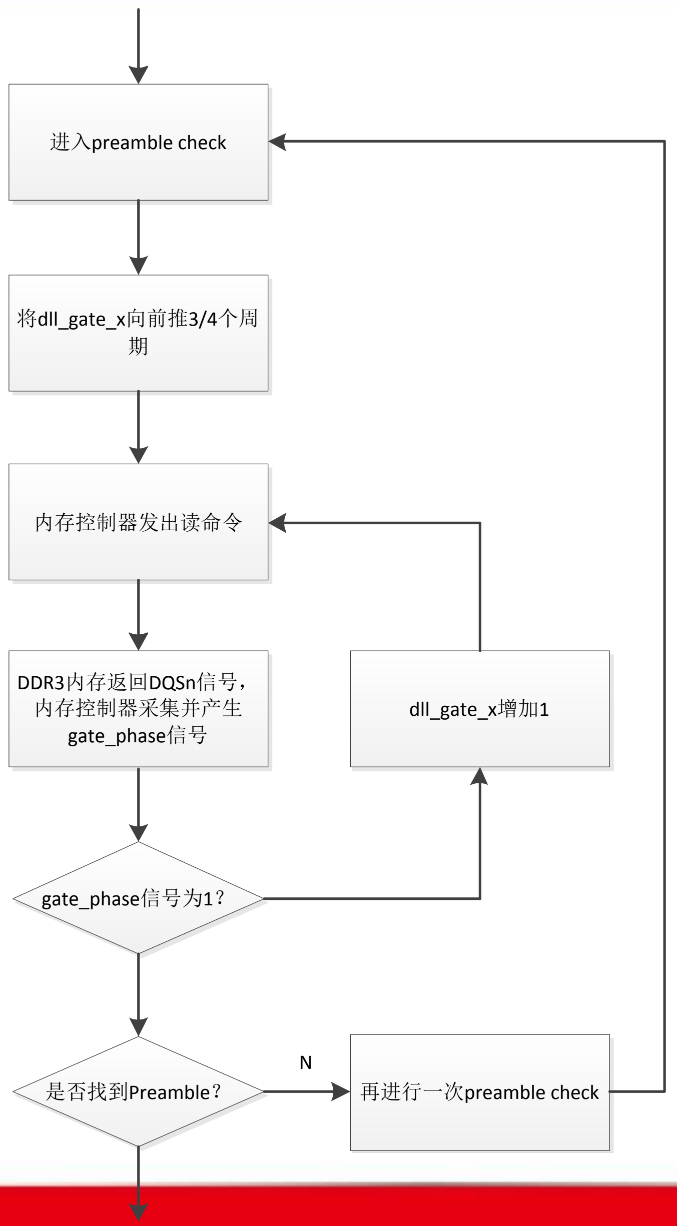
11.至此可能已经找到读DQS的边沿，为了保证步骤10的采样为可靠结果，需要重复步骤7，8，10数次（该次数由GATE_FILTER_LENGTH宏定义确定）；

12.将Dll_gate_x[6:0]减去GATE_FILTER_LENGTH；

13.重复执行步骤7-12，训练下一个Byte；

14.此时所有Dll_gate_x的值设置正确；

## gate leveling软件流程



15.Preamble check。该步骤用于找到读DQS的起始位置，配置Rd_oe_begin/end_x, tRDDATA(0x1c0)寄存器的值。具体过程说明如下：

i.把当前颗粒的Dll_gate减去PREAMBLE_LENGTH_3AX

ii.流程与gate leveling相似，但是不找返回值为0的位置，直接找到返回值为1的位置

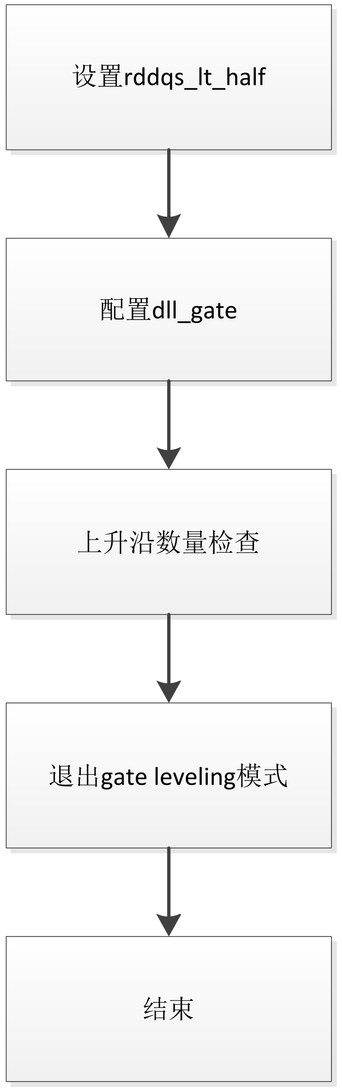
iii.若此时Dll_gate没有增加PREAMBLE_LENGTH_3AX-5次，则说明没有找到Preamble，将该颗粒的Rd_oe_begin/end减1，即向前推一个周期

iv.重新进行一次gate_leveling，直到找到上升沿

v.重复步骤i-iv再次进行Preamble Check。若Dll_gate增加了PREAMBLE_LENGTH_3AX-5次，则说明找到了Preamble，继续做下一个Byte的Preamble Check。全做完后进行下一步

此时Dll_gate_x[6:0]与Dll_wrdata_x[6:0]的和实际上就是读DQS相对于PHY内部时钟的相位关系。

## gate leveling软件流程



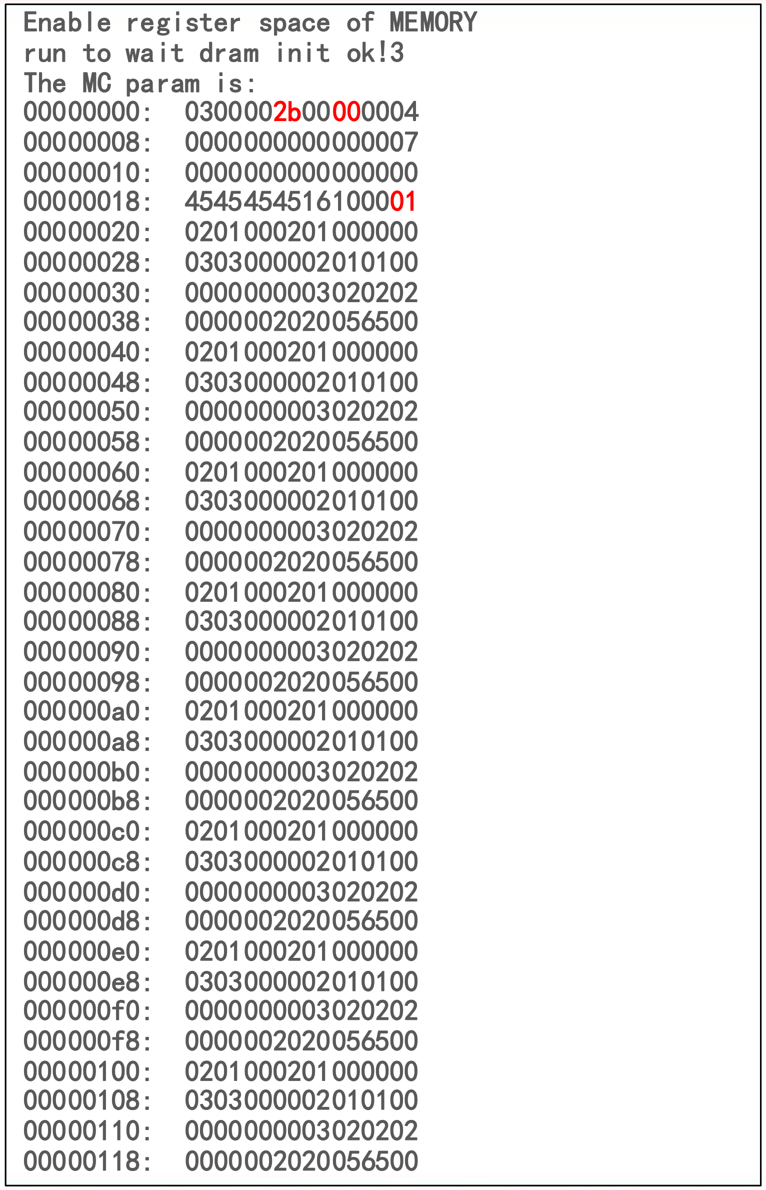
16.如果Dll_gate_x[6:0]与Dll_wrdata_x[6:0]的和大于RDDQS_LTHF_STD1或小于RDDQS_LTHF_STD2则设置Rddqs_lt_half_x为1,否则设置为0

17.把每个颗粒的Dll_gate_x减DLL_GATE_SUB

18.调整完毕后，再分别进行两次Lvl_req操作，观察Lvl_resp_x[7:5]与Lvl_resp_x[4:2]的值变化，如果各增加为Burst_length/2，则继续进行第19步操作；如果不为4，可能需要对Rd_oe_begin_x进行加一或减一操作，如果大于Burst_length/2，很可能需要对Dll_gate_x的值进行一些微调

19.将Lvl_mode（0x180）设置为2’b00，退出Gate Leveling模式；

# Leveling过程中参数打印



```
Enable register space of MEMORY
run to wait dram init ok!3
The MC param is:
00000000:   0300002b00000004
00000008:   0000000000000007
00000010:   0000000000000000
00000018:   4545454516100001
00000020:   0201000201000000
00000028:   0303000002010100
00000030:   0000000003020202
00000038:   0000002020056500
00000040:   0201000201000000
00000048:   0303000002010100
00000050:   0000000003020202
00000058:   0000002020056500
00000060:   0201000201000000
00000068:   0303000002010100
00000070:   0000000003020202
00000078:   0000002020056500
00000080:   0201000201000000
00000088:   0303000002010100
00000090:   0000000003020202
00000098:   0000002020056500
000000a0:   0201000201000000
000000a8:   0303000002010100
000000b0:   0000000003020202
000000b8:   0000002020056500
000000c0:   0201000201000000
000000c8:   0303000002010100
000000d0:   0000000003020202
000000d8:   0000002020056500
000000e0:   0201000201000000
000000e8:   0303000002010100
000000f0:   0000000003020202
000000f8:   0000002020056500
00000100:   0201000201000000
00000108:   0303000002010100
00000110:   0000000003020202
00000118:   0000002020056500
```

关注参数:

1. dll_value
2. dll_init_done
3. Init_start

错误情况分析:

1. Enable register space of MEMORY后卡死:考虑时钟问题
2. run to wait dram init ok!3 循环打印:频率过低

## Leveling过程中参数打印

```
Start Hard Leveling...

Enable register space of MEMORY

write leveling begin

all dll_wrdqs set 0

set leveling mode to be WRITE LEVELING

write leveling ready
```

错误情况分析：

1. write leveling ready后卡死:接触问题，硬件问题

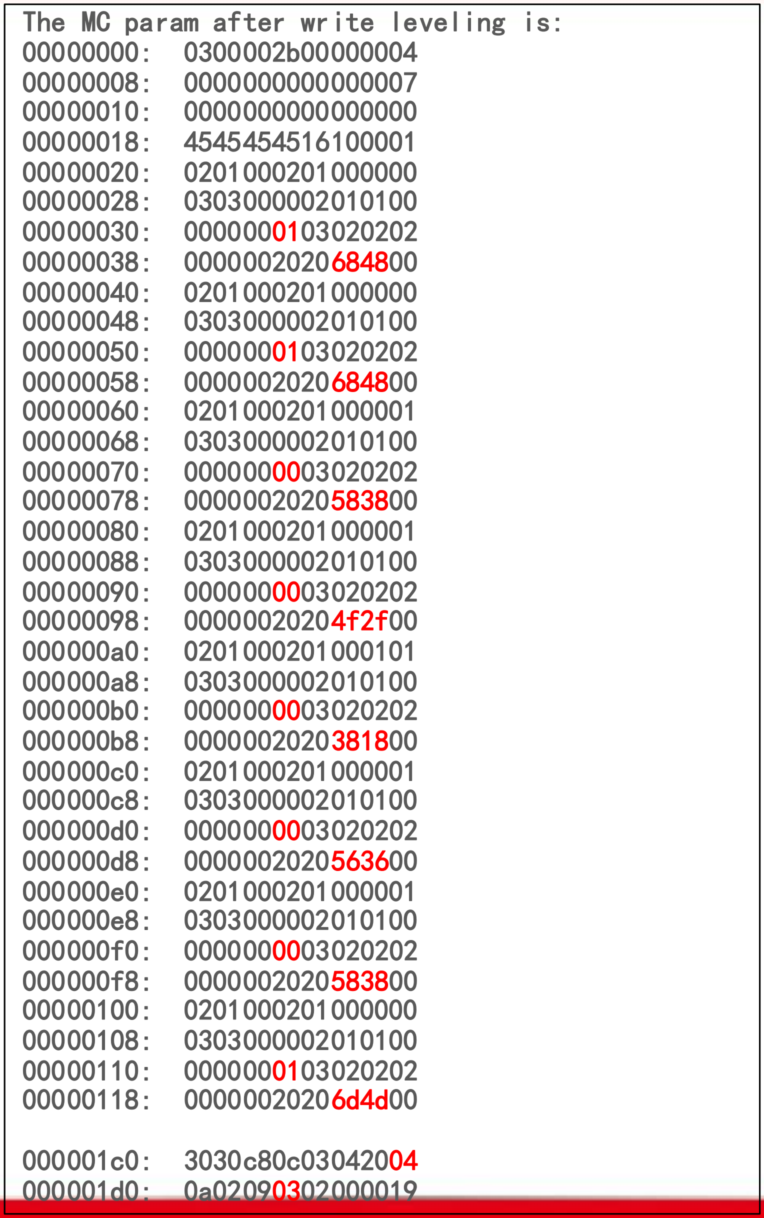
# Leveling过程中参数打印

```
The MC param after write leveling 0 to 1 is:
00000000:   0300002b00000004
00000008:   0000000000000007
00000010:   0000000000000000
00000018:   4545454516100001
00000020:   0201000201000000
00000028:   0303000002010100
00000030:   0000000003020202
00000038:   0000002020674700
00000040:   0201000201000000
00000048:   0303000002010100
00000050:   0000000003020202
00000058:   0000002020614100
00000060:   0201000201000001
00000068:   0303000002010100
00000070:   0000000003020202
00000078:   00000020205b3b00
00000080:   0201000201000001
00000088:   0303000002010100
00000090:   0000000003020202
00000098:   00000020204f2f00
000000a0:   0201000201000101
000000a8:   0303000002010100
000000b0:   0000000003020202
000000b8:   00000020203e1e00
000000c0:   0201000201000001
000000c8:   0303000002010100
000000d0:   0000000003020202
000000d8:   0000002020563600
000000e0:   0201000201000001
000000e8:   0303000002010100
000000f0:   0000000003020202
000000f8:   00000020205e3e00
00000100:   0201000201000000
00000108:   0303000002010100
00000110:   0000000003020202
00000118:   00000020206d4d00

000001c0:   3030c80c03042005
000001d0:   0a02090402000019
```

关注参数：

1. dll_wrdqs
2. dll_wrdq
3. wrdqs_lt_half
4. wrdq_lt_half

## Leveling过程中参数打印

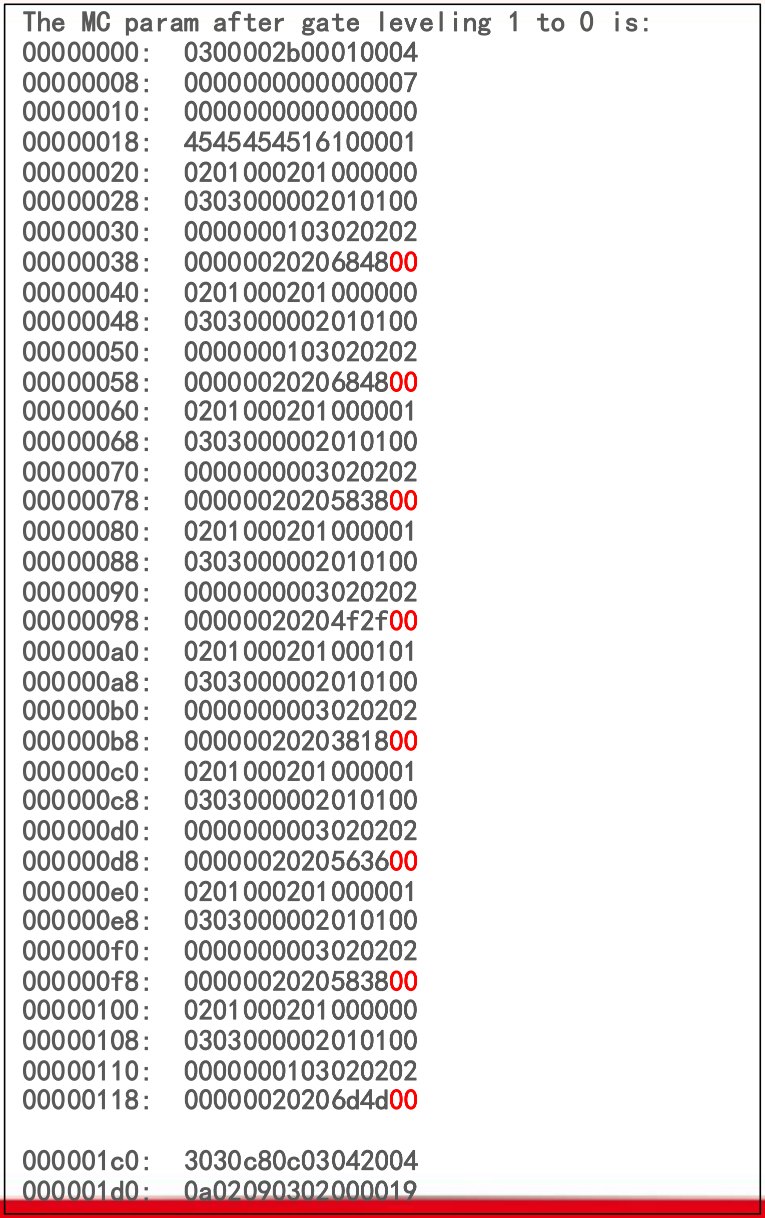


```
The MC param after write leveling is:
00000000:  0300002b00000004
00000008:  0000000000000007
00000010:  0000000000000000
00000018:  4545454516100001
00000020:  0201000201000000
00000028:  0303000002010100
00000030:  0000000103020202
00000038:  0000002020684800
00000040:  0201000201000000
00000048:  0303000002010100
00000050:  0000000103020202
00000058:  0000002020684800
00000060:  0201000201000001
00000068:  0303000002010100
00000070:  0000000003020202
00000078:  0000002020583800
00000080:  0201000201000001
00000088:  0303000002010100
00000090:  0000000003020202
00000098:  00000020204f2f00
000000a0:  0201000201000101
000000a8:  0303000002010100
000000b0:  0000000003020202
000000b8:  0000002020381800
000000c0:  0201000201000001
000000c8:  0303000002010100
000000d0:  0000000003020202
000000d8:  0000002020563600
000000e0:  0201000201000001
000000e8:  0303000002010100
000000f0:  0000000003020202
000000f8:  0000002020583800
00000100:  0201000201000000
00000108:  0303000002010100
00000110:  0000000103020202
00000118:  00000020206d4d00

000001c0:  3030c80c03042004
000001d0:  0a02090302000019
```

关注参数：

1. dll_wrdqs(微调)
2. dll_wrdq(微调)
3. wrdq_clkdelay
4. tRDDATA
5. tPHY_WRLAT

## Leveling过程中参数打印



```
The MC param after gate leveling 1 to 0 is:
00000000:   0300002b00010004
00000008:   0000000000000007
00000010:   0000000000000000
00000018:   4545454516100001
00000020:   0201000201000000
00000028:   0303000002010100
00000030:   0000000103020202
00000038:   0000002020684800
00000040:   0201000201000000
00000048:   0303000002010100
00000050:   0000000103020202
00000058:   0000002020684800
00000060:   0201000201000001
00000068:   0303000002010100
00000070:   0000000003020202
00000078:   0000002020583800
00000080:   0201000201000001
00000088:   0303000002010100
00000090:   0000000003020202
00000098:   00000020204f2f00
000000a0:   0201000201000101
000000a8:   0303000002010100
000000b0:   0000000003020202
000000b8:   0000002020381800
000000c0:   0201000201000001
000000c8:   0303000002010100
000000d0:   0000000003020202
000000d8:   0000002020563600
000000e0:   0201000201000001
000000e8:   0303000002010100
000000f0:   0000000003020202
000000f8:   0000002020583800
00000100:   0201000201000000
00000108:   0303000002010100
00000110:   0000000103020202
00000118:   00000020206d4d00

000001c0:   3030c80c03042004
000001d0:   0a02090302000019
```

关注参数：

1. dll_gate

## Leveling过程中参数打印

```
The MC param after gate leveling 0 to 1 is:
00000000:   0300002b00000004
00000008:   0000000000000007
00000010:   0000000000000000
00000018:   4545454516100001
00000020:   0201000201000000
00000028:   0303000002010100
00000030:   0000000103020202
00000038:   0000002020684807
00000040:   0201000201000000
00000048:   0303000002010100
00000050:   0000000103020202
00000058:   0000002020684800
00000060:   0201000201000001
00000068:   0202000002010100
00000070:   0000000002010202
00000078:   000000202058387f
00000080:   0201000201000001
00000088:   0202000002010100
00000090:   0000000002010202
00000098:   00000020204f2f7f
000000a0:   0201000201000101
000000a8:   0202000002010100
000000b0:   0000000002010202
000000b8:   000000202038187f
000000c0:   0201000201000001
000000c8:   0202000002010100
000000d0:   0000000002010202
000000d8:   0000002020563667
000000e0:   0201000201000001
000000e8:   0202000002010100
000000f0:   0000000002010202
000000f8:   000000202058387f
00000100:   0201000201000000
00000108:   0202000002010100
00000110:   0000000102010202
00000118:   00000020206d4d73

000001c0:   3030c80c03042005
000001d0:   0a02090302000019
```

关注参数：

1. rd_oe_begin/end
2. odt_oe_begin/end
3. dll_gate
4. tRDDATA

## Leveling过程中参数打印

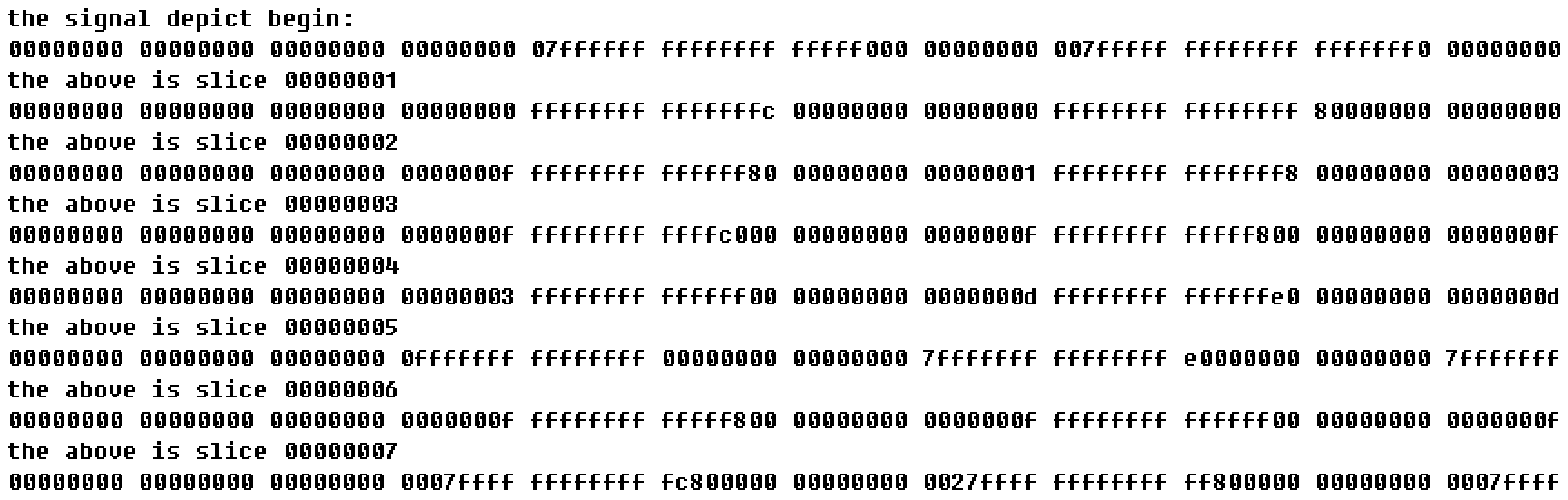
```
The MC param after leveling is:
00000000:   0000002b00000004
00000008:   0000000000000007
00000010:   0000000000000000
00000018:   4545454516100101
00000020:   0201000201010000
00000028:   0202000002010100
00000030:   0000000102010202
00000038:   0000002020684868
00000040:   0201000201010000
00000048:   0202000002010100
00000050:   0000000102010202
00000058:   0000002020684860
00000060:   0201000201000001
00000068:   0202000002010100
00000070:   0000000002010202
00000078:   0000002020583860
00000080:   0201000201000001
00000088:   0202000002010100
00000090:   0000000002010202
00000098:   00000020204f2f5e
000000a0:   0201000201000101
000000a8:   0202000002010100
000000b0:   0000000002010202
000000b8:   0000002020381860
000000c0:   0201000201000001
000000c8:   0202000002010100
000000d0:   0000000002010202
000000d8:   000000202056364a
000000e0:   0201000201000001
000000e8:   0202000002010100
000000f0:   0000000002010202
000000f8:   000000202058385e
00000100:   0201000201000000
00000108:   0202000002010100
00000110:   0000000102010202
00000118:   00000020206d4d50

000001c0:   3030c80c03042005
000001d0:   0a02090302000019
```

关注参数：

1. rddqs_lt_half
2. rd_oe_begin/end
3. odt_oe_begin/end
4. dll_gate
5. tRDDATA

## Gate leveling时信号描绘

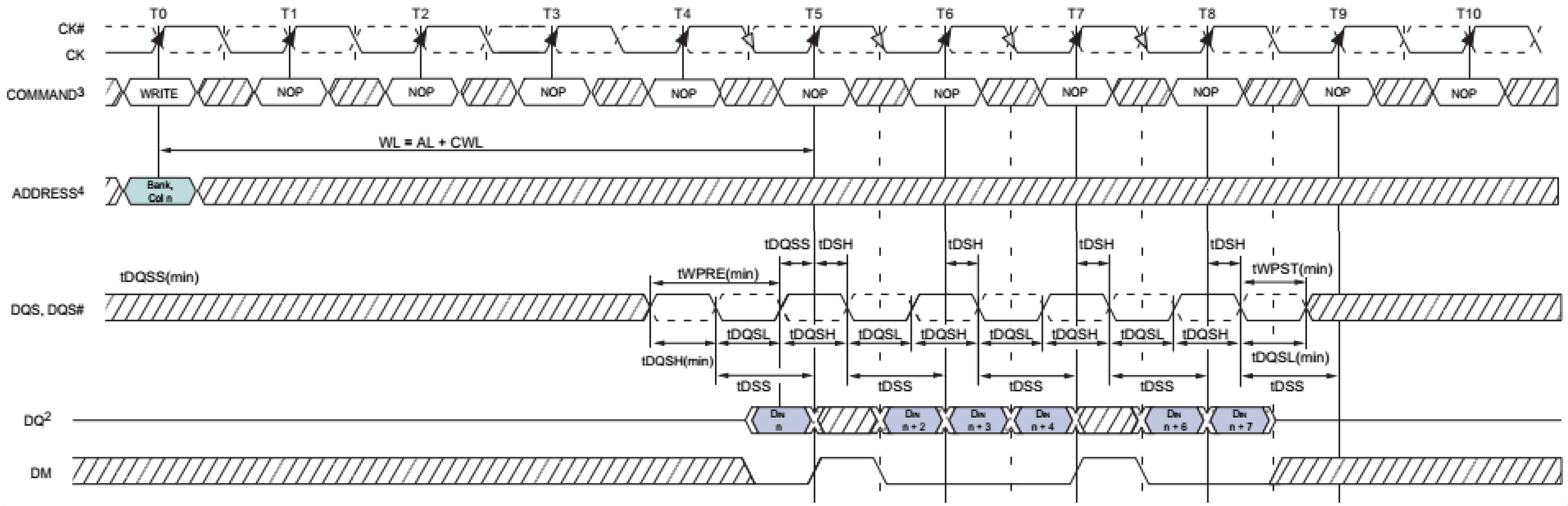
定义SIGNAL_DEPICT_DEBUG会打印以下信息



```
the signal depict begin:
00000000 00000000 00000000 00000000 07ffffff ffffffff fffff000 00000000 007fffff ffffffff fffffff0 00000000
the above is slice 00000001
00000000 00000000 00000000 00000000 ffffffff fffffffc 00000000 00000000 ffffffff ffffffff 80000000 00000000
the above is slice 00000002
00000000 00000000 00000000 0000000f ffffffff ffffff80 00000000 00000001 ffffffff fffffff8 00000000 00000003
the above is slice 00000003
00000000 00000000 00000000 0000000f ffffffff ffffc000 00000000 0000000f ffffffff fffff800 00000000 0000000f
the above is slice 00000004
00000000 00000000 00000000 00000003 ffffffff ffffff00 00000000 0000000d ffffffff ffffffe0 00000000 0000000d
the above is slice 00000005
00000000 00000000 00000000 0fffffff ffffffff 00000000 00000000 7fffffff ffffffff e0000000 00000000 7fffffff
the above is slice 00000006
00000000 00000000 00000000 0000000f ffffffff fffff800 00000000 0000000f ffffffff ffffff00 00000000 0000000f
the above is slice 00000007
00000000 00000000 00000000 0007ffff ffffffff fc800000 00000000 0027ffff ffffffff ff800000 00000000 0007ffff
```

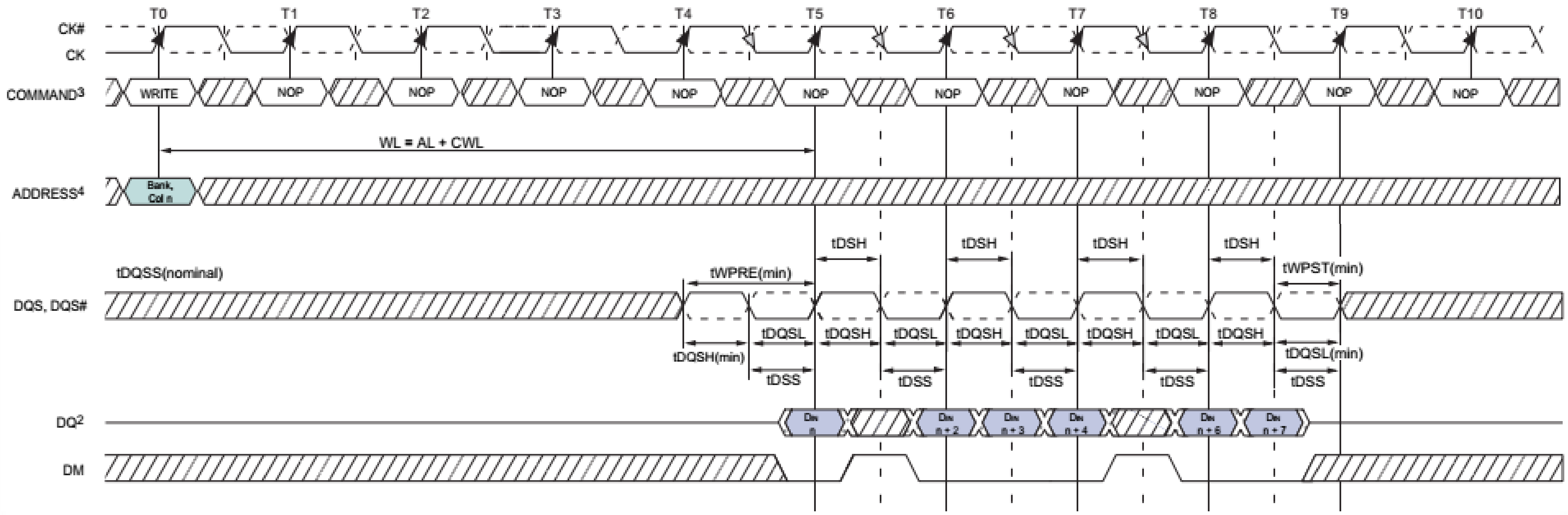
## DQS/CK相位关系范围

DQS可以相对CK提前



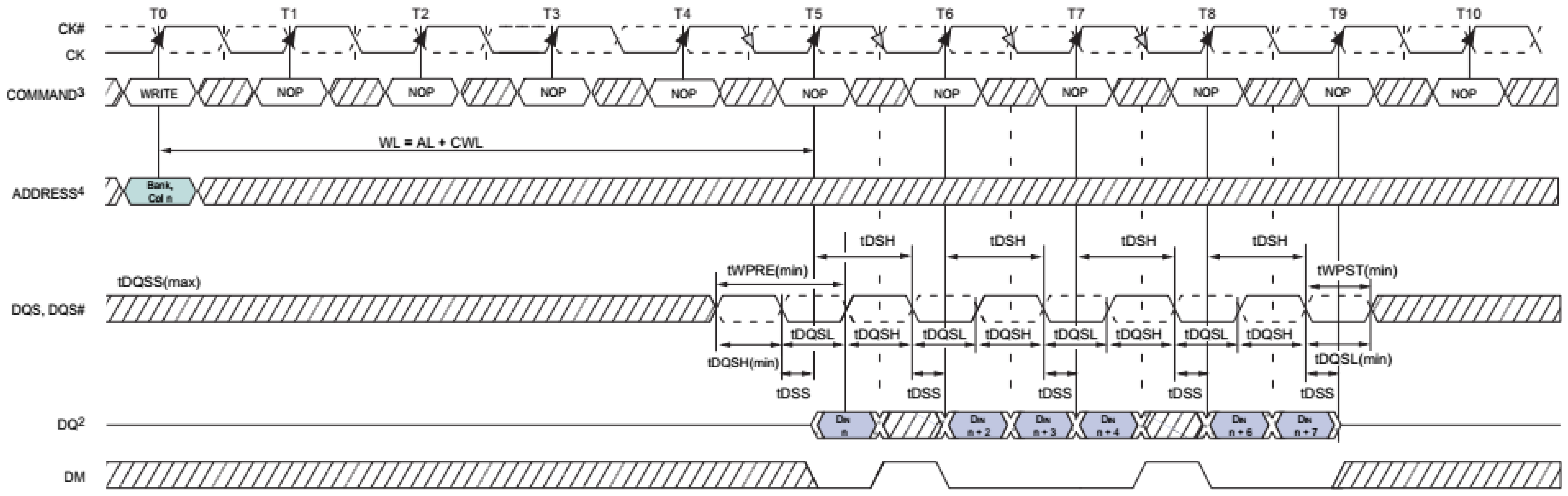
## DQS/CK相位关系范围

DQS与CK正常相位关系



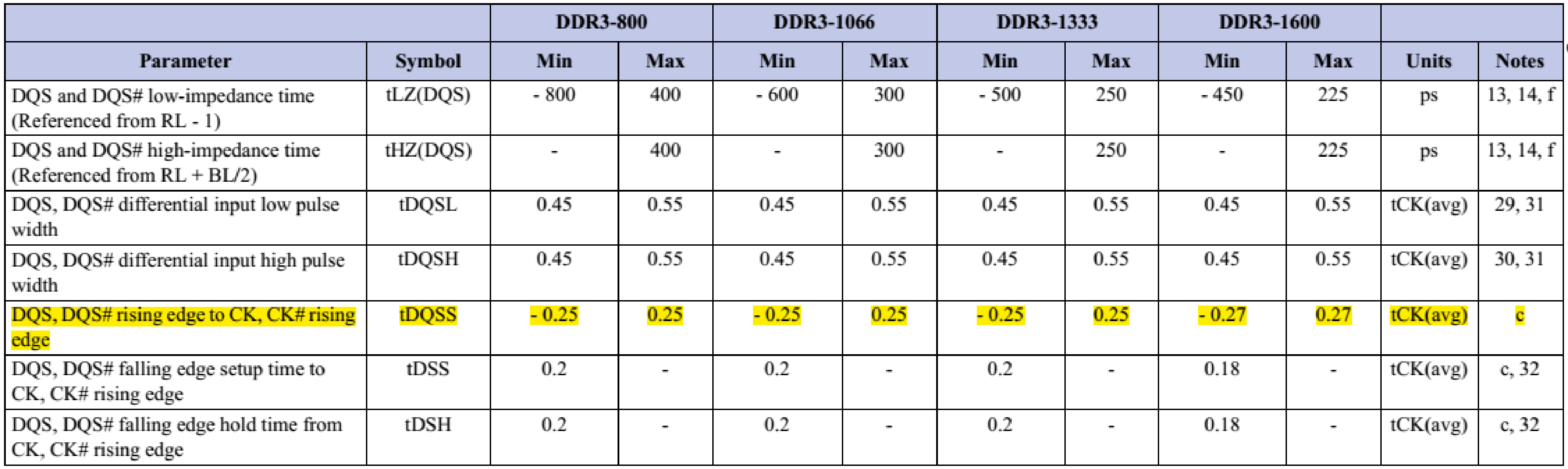
## DQS/CK相位关系范围

DQS可以相对CK延后



## DQS/CK相位关系范围

tDQSS



| | | DDR3-800 | | DDR3-1066 | | DDR3-1333 | | DDR3-1600 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Parameter | Symbol | Min | Max | Min | Max | Min | Max | Min | Max | Units | Notes |
| DQS and DQS# low-impedance time (Referenced from RL - 1) | tLZ(DQS) | - 800 | 400 | - 600 | 300 | - 500 | 250 | - 450 | 225 | ps | 13, 14, f |
| DQS and DQS# high-impedance time (Referenced from RL + BL/2) | tHZ(DQS) | - | 400 | - | 300 | - | 250 | - | 225 | ps | 13, 14, f |
| DQS, DQS# differential input low pulse width | tDQSL | 0.45 | 0.55 | 0.45 | 0.55 | 0.45 | 0.55 | 0.45 | 0.55 | tCK(avg) | 29, 31 |
| DQS, DQS# differential input high pulse width | tDQSH | 0.45 | 0.55 | 0.45 | 0.55 | 0.45 | 0.55 | 0.45 | 0.55 | tCK(avg) | 30, 31 |
| DQS, DQS# rising edge to CK, CK# rising edge | tDQSS | - 0.25 | 0.25 | - 0.25 | 0.25 | - 0.25 | 0.25 | - 0.27 | 0.27 | tCK(avg) | c |
| DQS, DQS# falling edge setup time to CK, CK# rising edge | tDSS | 0.2 | - | 0.2 | - | 0.2 | - | 0.18 | - | tCK(avg) | c, 32 |
| DQS, DQS# falling edge hold time from CK, CK# rising edge | tDSH | 0.2 | - | 0.2 | - | 0.2 | - | 0.18 | - | tCK(avg) | c, 32 |

## 训练后参数分析

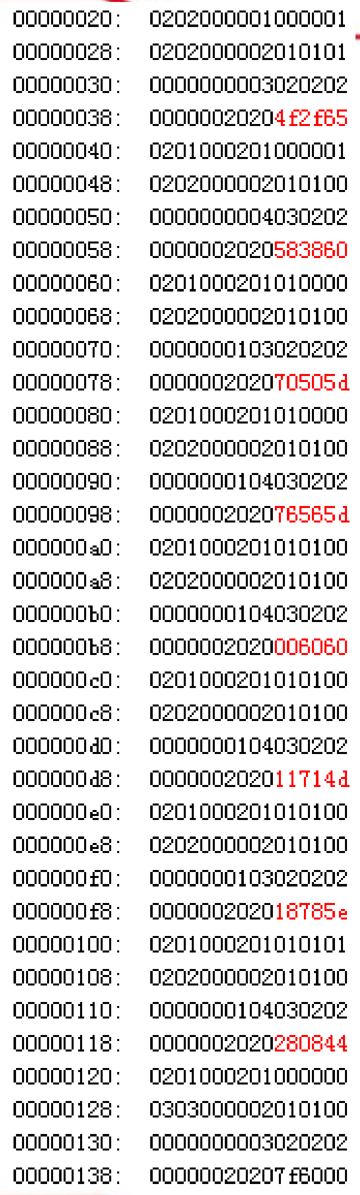
**UDIMM:**

- dll_wrdqs/dll_wrdata的值变化趋势：

  dll_wrdqs_0至dll_wrdqs_8(如果未使能ECC则为dll_wrdqs_7)的值依次增大dll_wrdata_0至dll_wrdata_8的值也依次增大。体现DDR3的FLY-BY走线结构

- dll_gate的值：

  反映了DQS走线长度，其值越大则对应的Byte的DQS/DQ线越长。一般来说不同BYTE之间的DQS/DQ线等长，所以dll_gate_*的值应该大致相等，差别不会太大。



```
00000020:  0202000001000001
00000028:  0202000002010101
00000030:  0000000003020202
00000038:  00000020204f2f65
00000040:  0201000201000001
00000048:  0202000002010100
00000050:  0000000004030202
00000058:  0000002020583860
00000060:  0201000201010000
00000068:  0202000002010100
00000070:  0000000103020202
00000078:  000000202070505d
00000080:  0201000201010000
00000088:  0202000002010100
00000090:  0000000104030202
00000098:  000000202076565d
000000a0:  0201000201010100
000000a8:  0202000002010100
000000b0:  0000000104030202
000000b8:  0000002020006060
000000c0:  0201000201010100
000000c8:  0202000002010100
000000d0:  0000000104030202
000000d8:  000000202011714d
000000e0:  0201000201010100
000000e8:  0202000002010100
000000f0:  0000000103020202
000000f8:  000000202018785e
00000100:  0201000201010101
00000108:  0202000002010100
00000110:  0000000104030202
00000118:  0000002020280844
00000120:  0201000201000000
00000128:  0303000002010100
00000130:  0000000003020202
00000138:  00000020207f6000
```

## 训练后参数分析

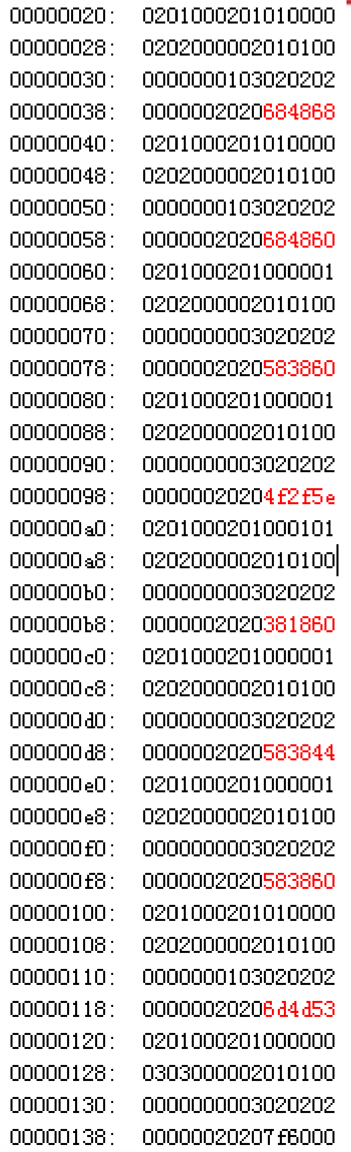
**RDIMM:**

- dll_wrdqs/dll_wrdata的值变化趋势：

  参数分为两组，其中一组dll_wrdqs_8、dll_wrdqs_3、dll_wrdqs_2、dll_wrdqs_1、dll_wrdqs_0的值依次增加，另外一组dll_wrdqs_4、dll_wrdqs_5、dll_wrdqs_6、dll_wrdqs_7的值依次增加。dll_wrdata_*同理。

  - dll_gate的值：

  反映了DQS走线长度，其值越大则对应的Byte的DQS/DQ线越长。一般来说不同BYTE之间的DQS/DQ线等长，所以dll_gate_*的值应该大致相等，差别不会太大。



```
00000020:   0201000201010000
00000028:   0202000002010100
00000030:   0000000103020202
00000038:   0000002020684868
00000040:   0201000201010000
00000048:   0202000002010100
00000050:   0000000103020202
00000058:   0000002020684860
00000060:   0201000201000001
00000068:   0202000002010100
00000070:   0000000003020202
00000078:   0000002020583860
00000080:   0201000201000001
00000088:   0202000002010100
00000090:   0000000003020202
00000098:   00000020204f2f5e
000000a0:   0201000201000101
000000a8:   0202000002010100
000000b0:   0000000003020202
000000b8:   0000002020381860
000000c0:   0201000201000001
000000c8:   0202000002010100
000000d0:   0000000003020202
000000d8:   0000002020583844
000000e0:   0201000201000001
000000e8:   0202000002010100
000000f0:   0000000003020202
000000f8:   0000002020583860
00000100:   0201000201010000
00000108:   0202000002010100
00000110:   0000000103020202
00000118:   00000020206d4d53
00000120:   0201000201000000
00000128:   0303000002010100
00000130:   0000000003020202
00000138:   00000020207f6000
```

## 训练文件中宏定义说明

- **ORDER_OF_UDIMM：**
  内存条为UDIMM时，内存颗粒的排布顺序
- **ORDER_OF_RDIMM：**
  内存条为RDIMM时，内存颗粒的排布顺序
- **WRDQS_LTHF_STD：**
  wrdqs_lt_half设置的参考基准值，dll_wrdqs高于此值时lt_half设为1，低于此值时设为0
- **WRDQ_LTHF_STD：**
  wrdq_lt_half设置的参考基准值，dll_wrdq高于此值时lt_half设为1，低于此值时设为0
- **RDDQS_LTHF_STD1/2：**
  rddqs_lt_half设置的参考基准值，算出的rddqs偏移高于STD1或低于STD2时设置为1，否则设置为0
- **DLL_WRDQ_SUB：**
  dll_wrdq相对于dll_wrdqs减小的值
- **DLL_GATE_SUB：**
  dll_gate相对于rddqs减小的值

## 训练文件中宏定义说明

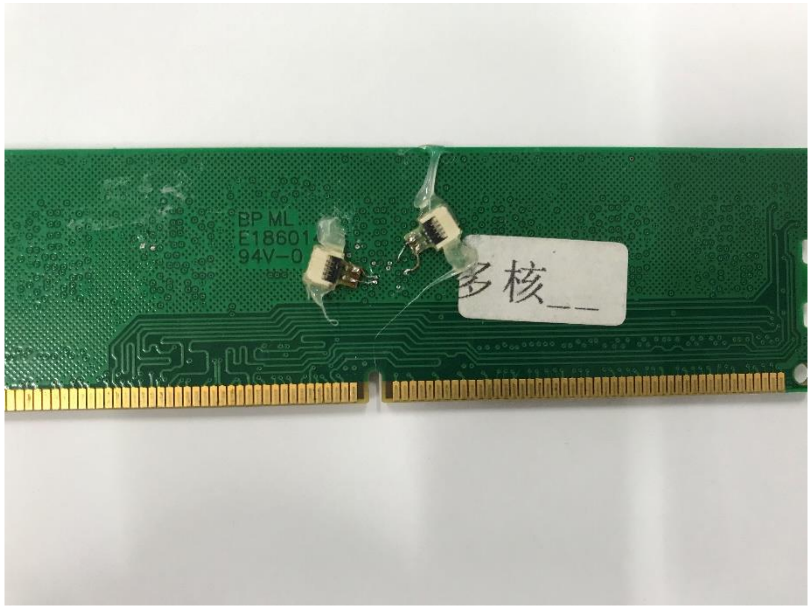
- **WR_FILTER_LENGTH：**
  在write leveling找到0到1的跳变时，多校验的次数
- **GATE_FILTER_LENGTH：**
  在gate leveling找到0到1的跳变时，多校验的次数
- **PREAMBLE_LENGTH_3AX：**
  进行preamble check时，检验的preamble长度
- **GET_NUMBER_OF_SLICES：**
  获取颗粒数量（是否有ECC）保存至t0
- **PRINT_THE_MC_PARAM：**
  打印所有内存参数
- **SIGNAL_DEPICT_DEBUG**
  打印Gate leveling时对内部读DQS的描绘
- **RDOE_SUB_TRDDATA_ADD：**
  rd_oe参数增加超过阈值（3）时进行所有颗粒rd_oe减少并增加trddata的操作
- **RDOE_ADD_TRDDATA_SUB：**
  rd_oe参数减少超过阈值（0）时进行所有颗粒rd_oe增加并减少trddata的操作

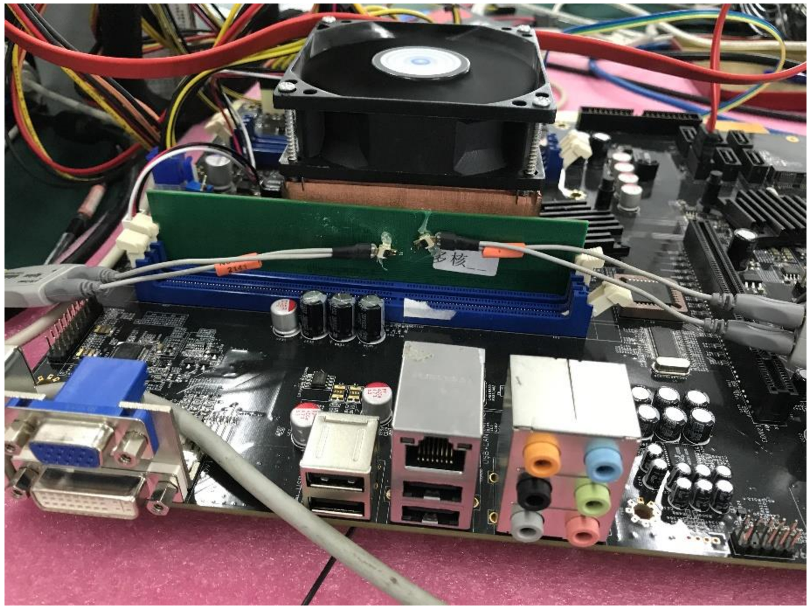
## 提纲

## 测量点选择

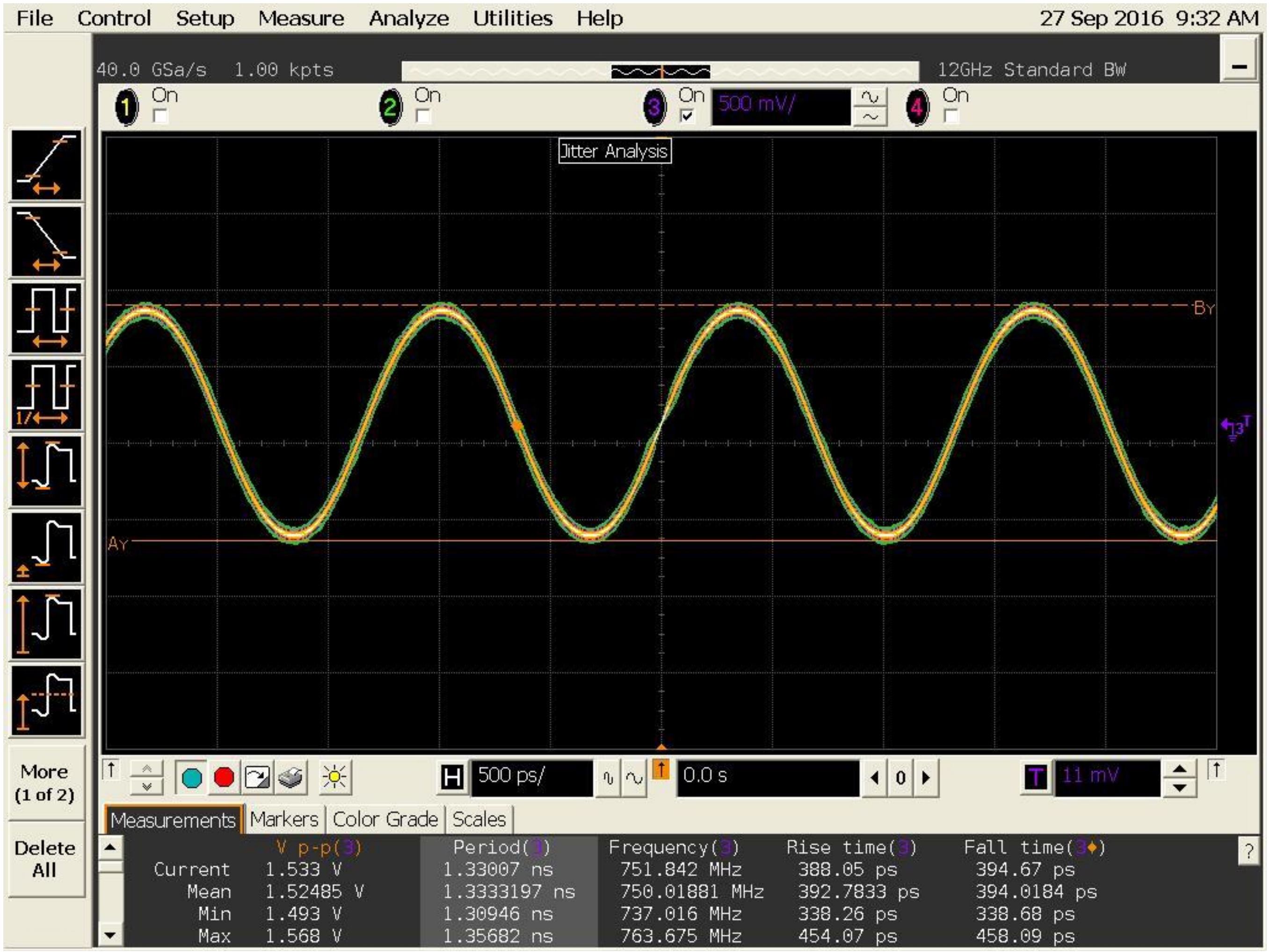
- 为了保证测量结果的可靠性，测量点需要选择在信号的终端。一般测量时选择单面内存条，在所测颗粒的背面的过孔选择所测信号。
- 时钟/地址/命令/控制信号都是单向传输的，测量时在内存条背面选测量点；
- 写DQS-DQ由内存控制器发给内存条，测量点也选在内存条背面过孔；
- 读DQS-DQ由内存条发给内存控制器，测量点需要在CPU端选择对应信号。

## 测量点示例



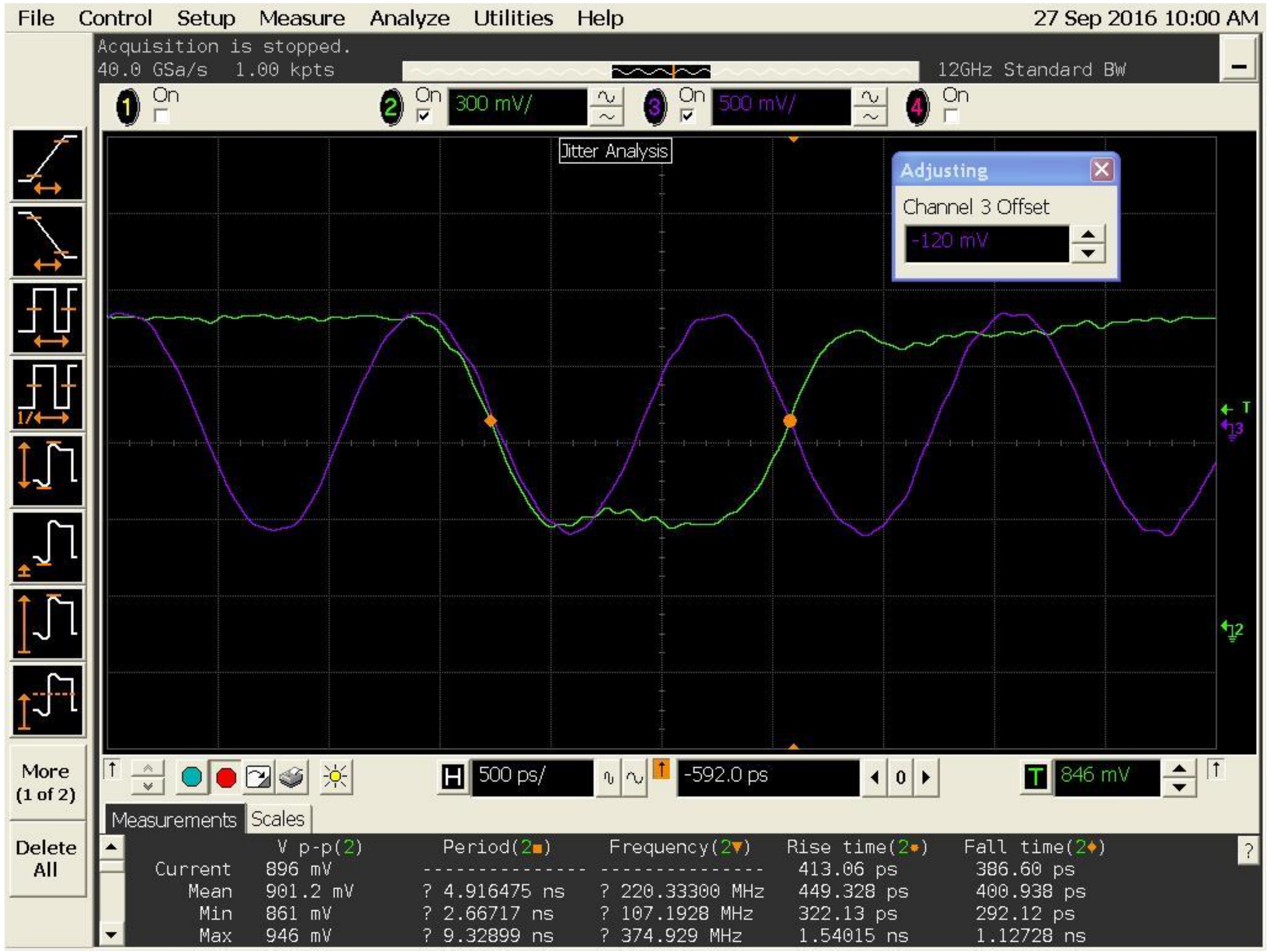


## 时钟信号CLK



```
          V p-p(3)      Period(3)       Frequency(3)    Rise time(3)    Fall time(3)
Current   1.533 V       1.33007 ns      751.842 MHz     388.05 ps       394.67 ps
   Mean   1.52485 V     1.3333197 ns    750.01881 MHz   392.7833 ps     394.0184 ps
    Min   1.493 V       1.30946 ns      737.016 MHz     338.26 ps       338.68 ps
    Max   1.568 V       1.35682 ns      763.675 MHz     454.07 ps       458.09 ps
```

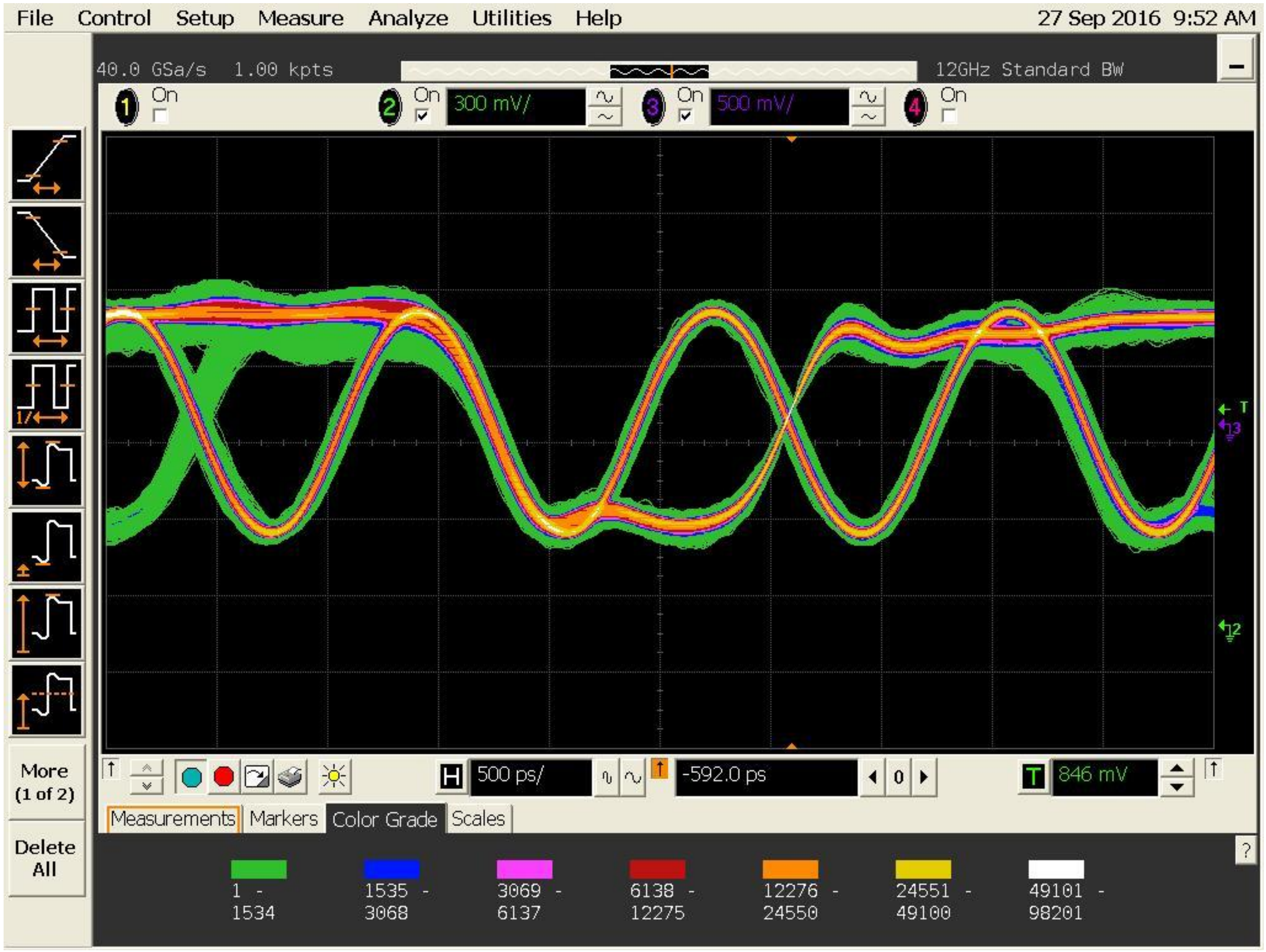
## 地址/命令信号



```
27 Sep 2016 10:00 AM
Acquisition is stopped.
40.0 GSa/s   1.00 kpts                12GHz Standard BW
Channel 2: 300 mV/    Channel 3: 500 mV/
Adjusting
Channel 3 Offset
-120 mV
H 500 ps/     -592.0 ps     0     T 846 mV
```

```
          V p-p(2)    Period(2)       Frequency(2)     Rise time(2)   Fall time(2)
Current   896 mV      --------------- ---------------  413.06 ps      386.60 ps
   Mean   901.2 mV    ? 4.916475 ns   ? 220.33300 MHz  449.328 ps     400.938 ps
    Min   861 mV      ? 2.66717 ns    ? 107.1928 MHz   322.13 ps      292.12 ps
    Max   946 mV      ? 9.32899 ns    ? 374.929 MHz    1.54015 ns     1.12728 ns
```

## 地址/命令信号



27 Sep 2016 9:52 AM

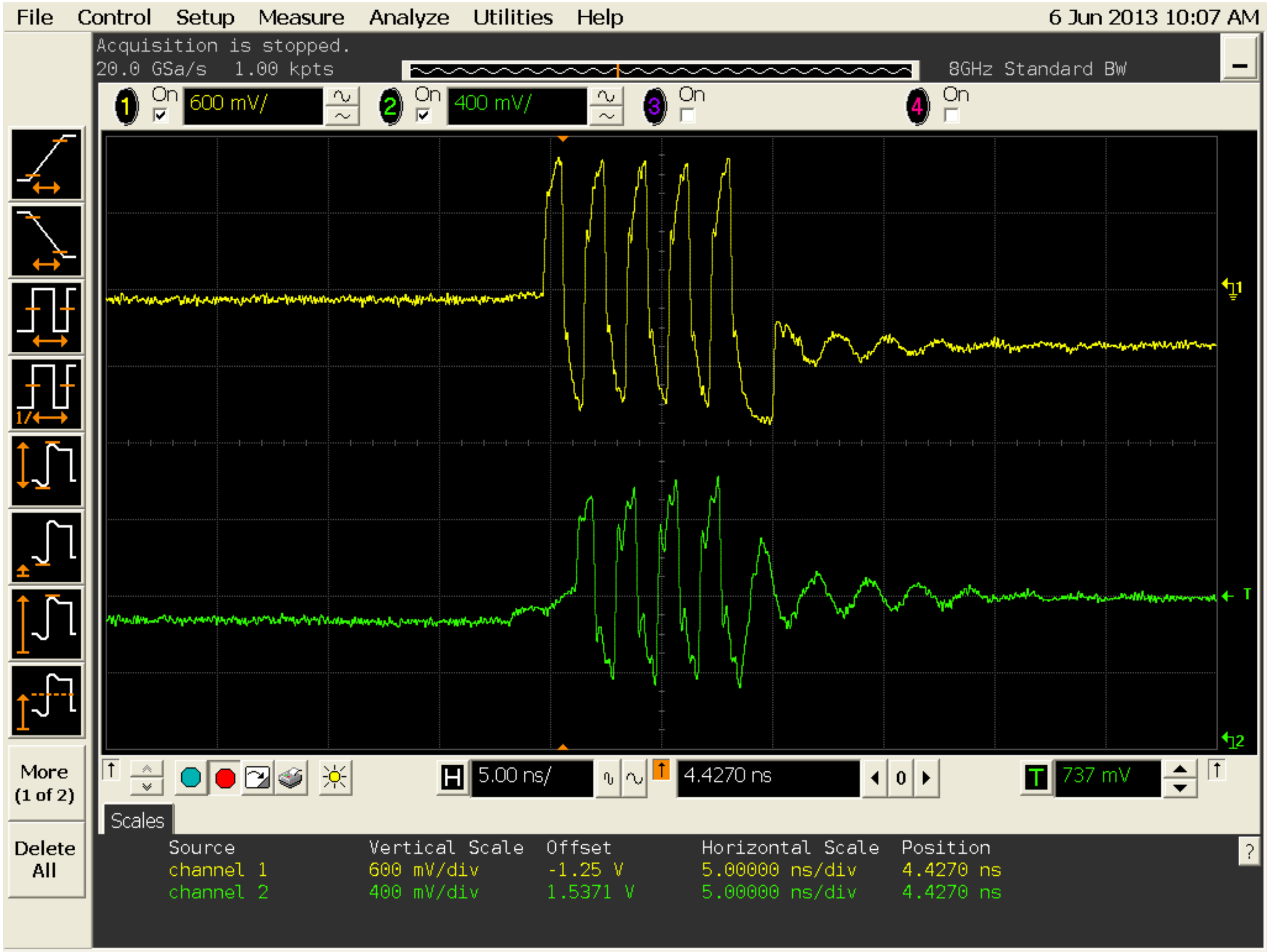
40.0 GSa/s 1.00 kpts 12GHz Standard BW

1 On ☐ 2 On ☑ 300 mV/ 3 On ☑ 500 mV/ 4 On ☐

H 500 ps/ -592.0 ps 0 T 846 mV

| | | | | | | |
|---|---|---|---|---|---|---|
| 🟩 | 🟦 | 🟪 | 🟥 | 🟧 | 🟨 | ⬜ |
| 1 - 1534 | 1535 - 3068 | 3069 - 6137 | 6138 - 12275 | 12276 - 24550 | 24551 - 49100 | 49101 - 98201 |

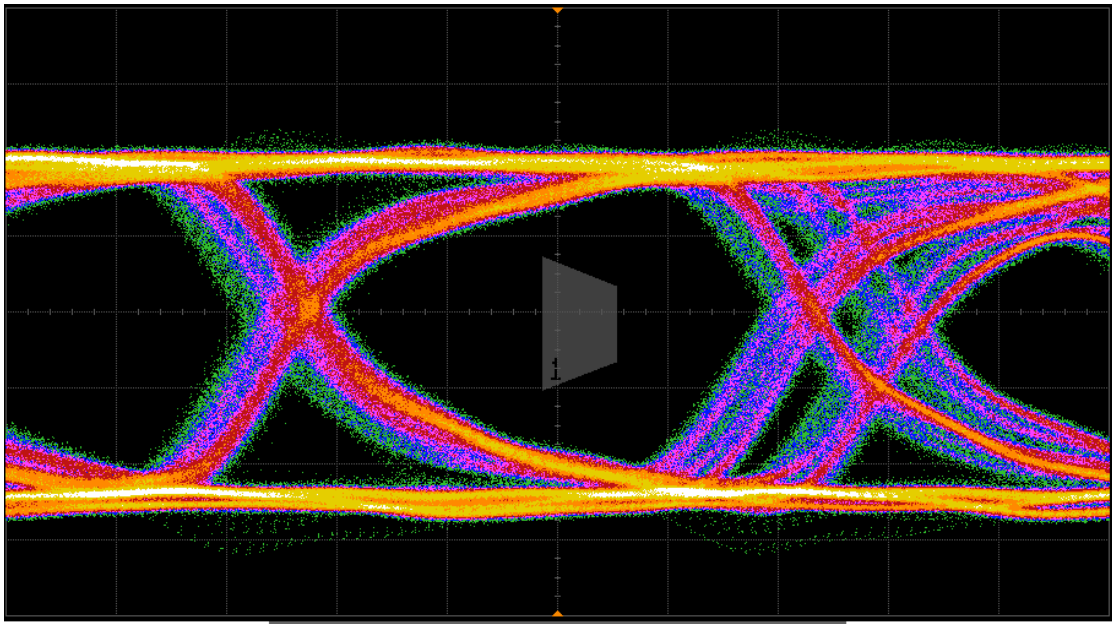
## 写信号



```
Acquisition is stopped.
20.0 GSa/s   1.00 kpts
```

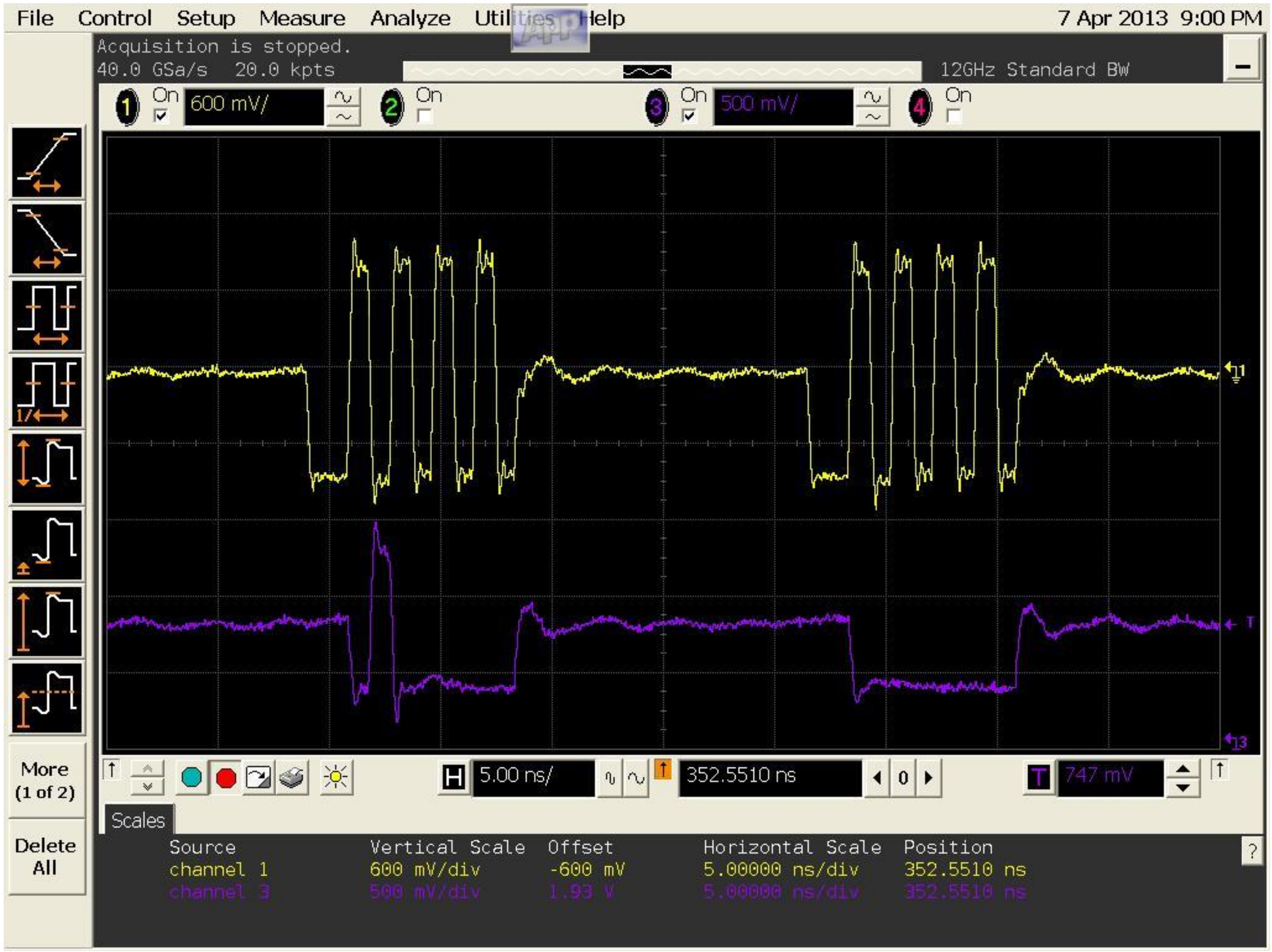
```
Source       Vertical Scale   Offset     Horizontal Scale   Position
channel 1    600 mV/div       -1.25 V    5.00000 ns/div     4.4270 ns
channel 2    400 mV/div       1.5371 V   5.00000 ns/div     4.4270 ns
```

## 写眼图



```
Scales ------------------------------------------------------------
                   Horiz Scale   Position      Vertical Scale   Offset
Channel 2          200.0ps/div   0.000s        200.0mV/div      781.0mV
Function 1         200.0ps/div   0.000s        1.000V/div       -84.12mV
```

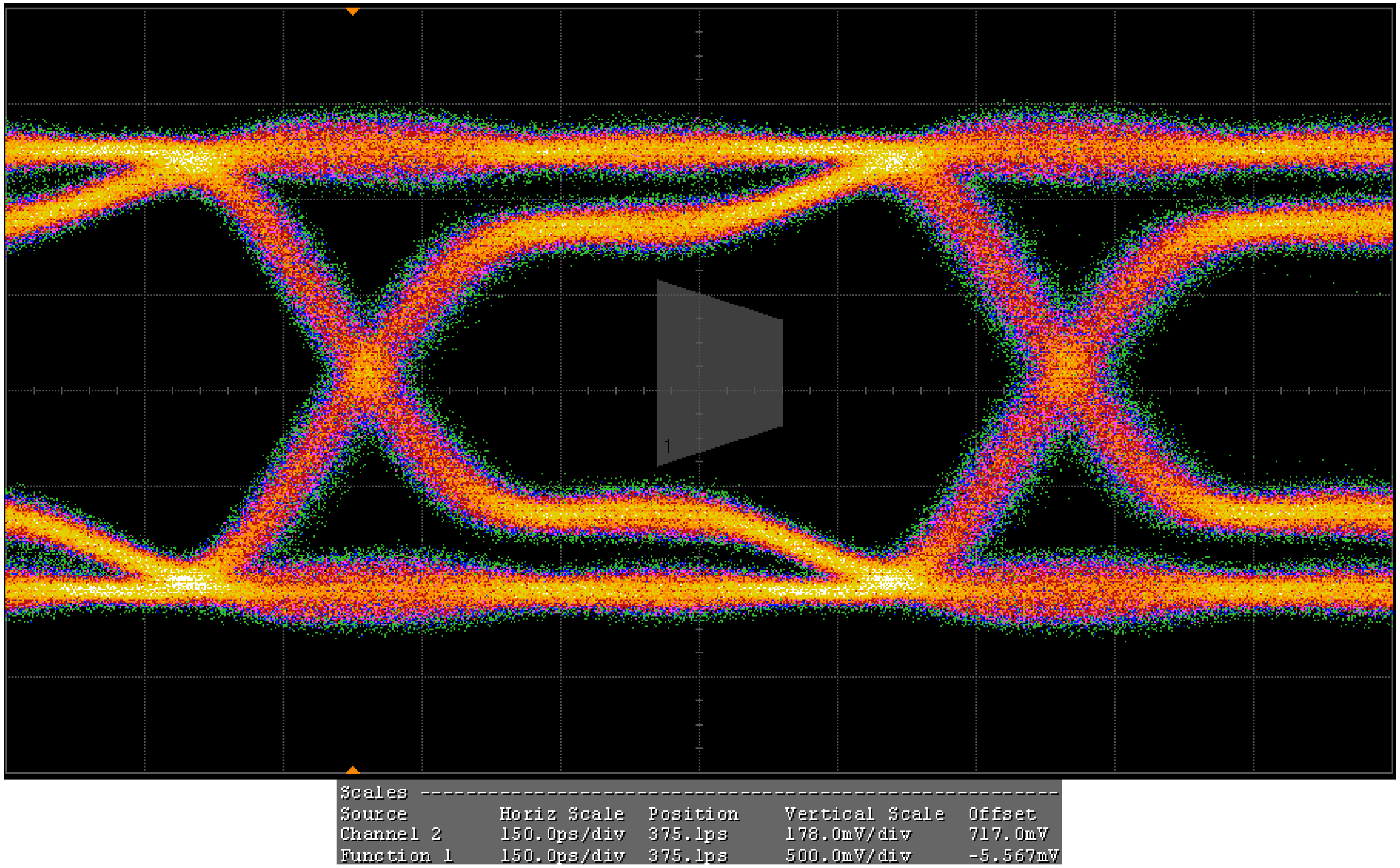
## 读信号



Acquisition is stopped.
40.0 GSa/s 20.0 kpts

```
Source       Vertical Scale  Offset    Horizontal Scale  Position
channel 1    600 mV/div      -600 mV   5.00000 ns/div    352.5510 ns
channel 3    500 mV/div      1.93 V    5.00000 ns/div    352.5510 ns
```

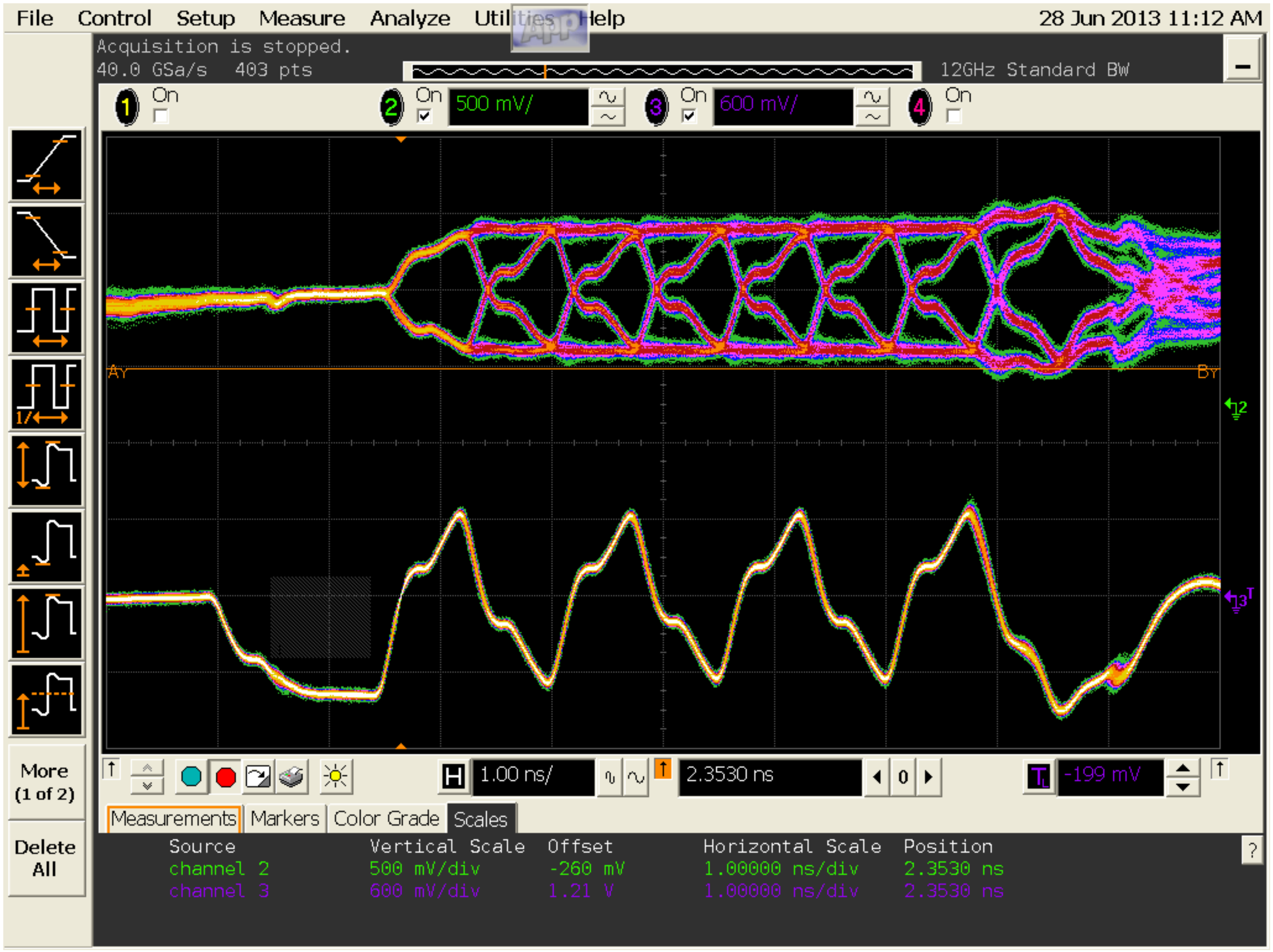
## 读眼图



```
Scales -----------------------------------------------------------
Source          Horiz Scale   Position     Vertical Scale   Offset
Channel 2       150.0ps/div   375.1ps      178.0mV/div      717.0mV
Function 1      150.0ps/div   375.1ps      500.0mV/div      -5.567mV
```

## ODT影响实例

CPU端ODT 120欧，单条双面内存rd_odt_map配置为0x4812

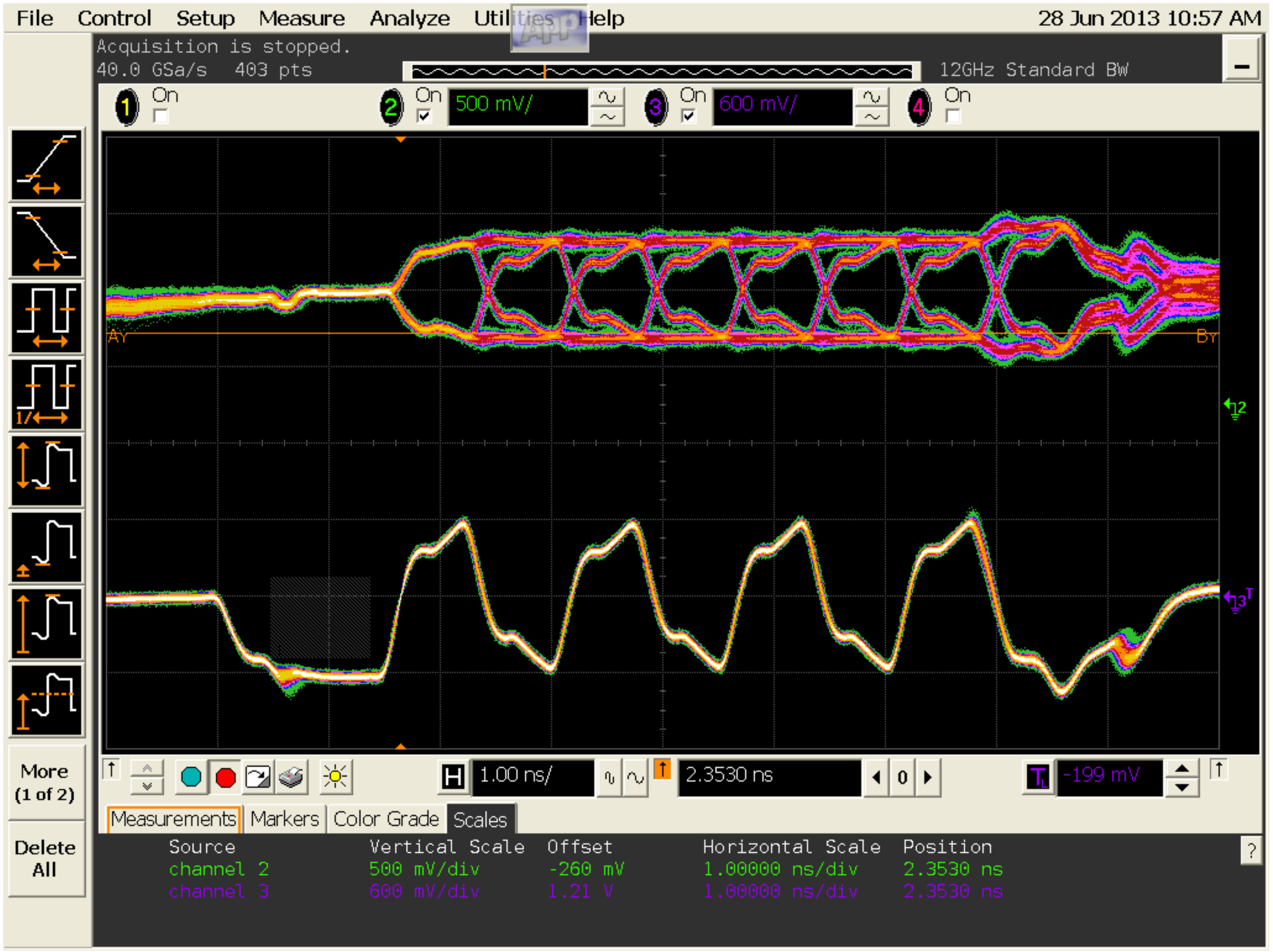


```
Acquisition is stopped.
40.0 GSa/s   403 pts
```

```
Source       Vertical Scale   Offset     Horizontal Scale   Position
channel 2    500 mV/div       -260 mV    1.00000 ns/div     2.3530 ns
channel 3    600 mV/div       1.21 V     1.00000 ns/div     2.3530 ns
```

## ODT影响实例

修改CPU端ODT为60欧，其他配置不变，台阶消失，数据摆幅减小

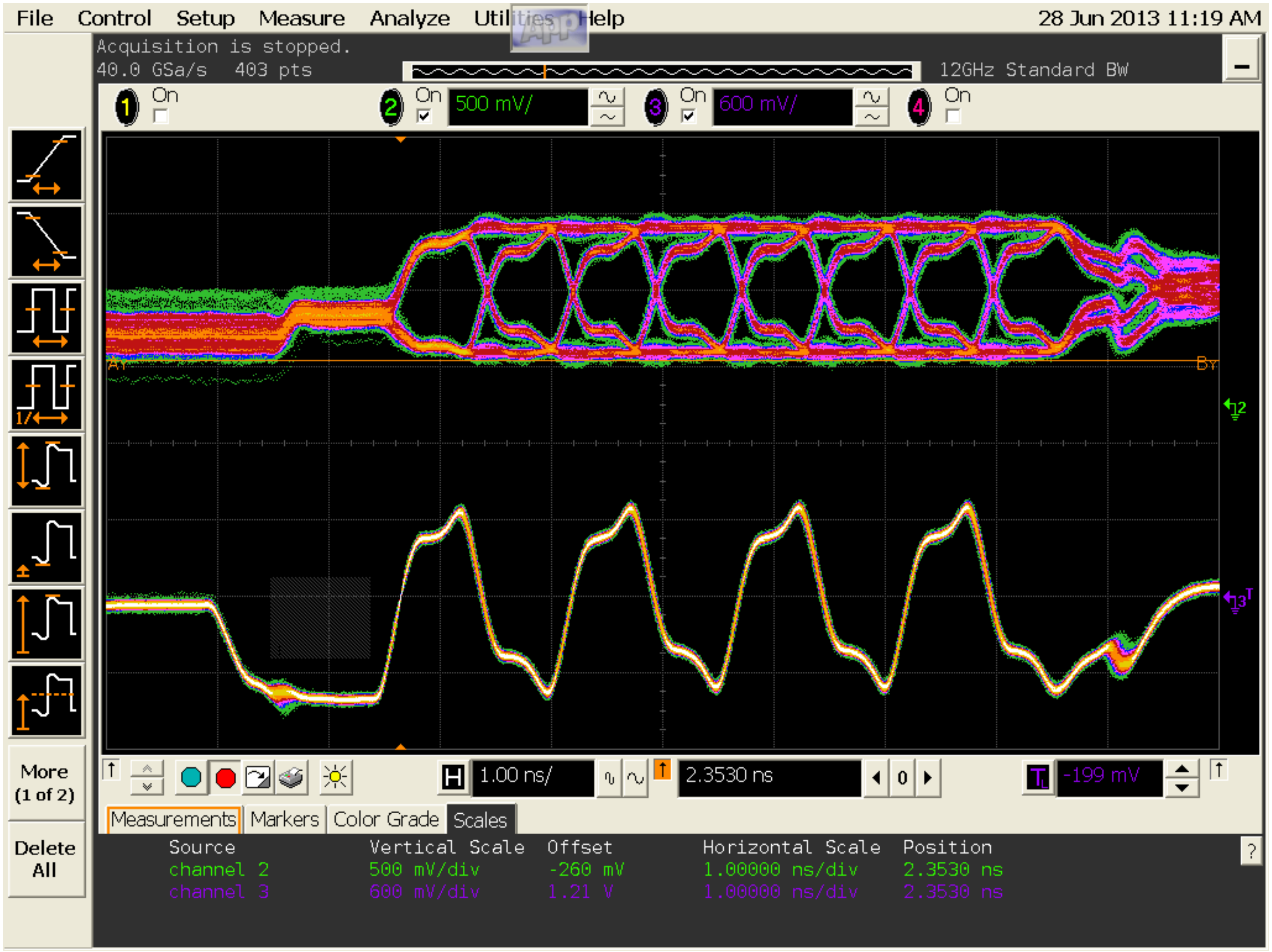


```
Acquisition is stopped.
40.0 GSa/s   403 pts
```

```
Source       Vertical Scale  Offset    Horizontal Scale  Position
channel 2    500 mV/div      -260 mV   1.00000 ns/div    2.3530 ns
channel 3    600 mV/div      1.21 V    1.00000 ns/div    2.3530 ns
```

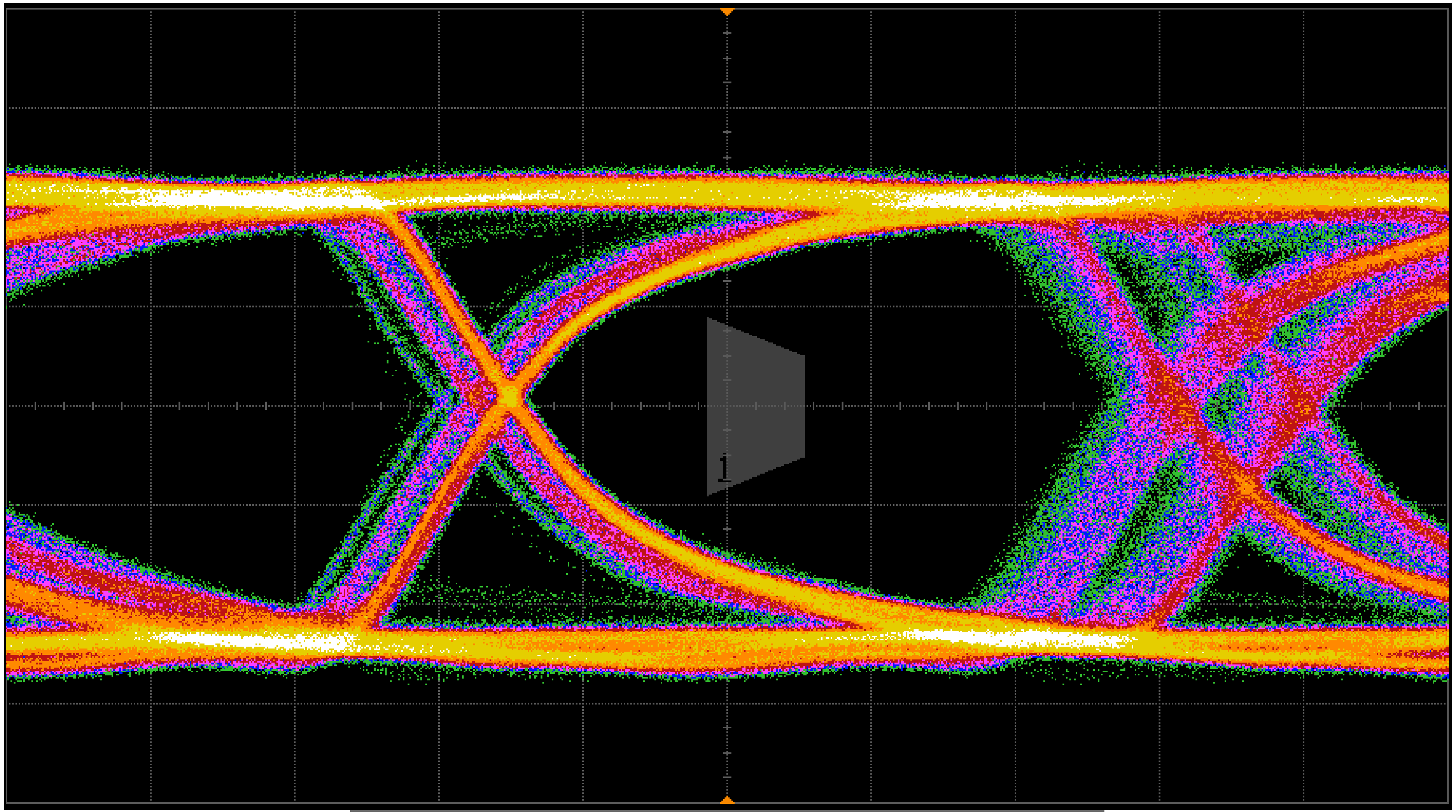
## ODT影响实例

修改CPU端ODT为60欧，并修改rd_odt_map为0x0000，信号质量达到最好



```
Source        Vertical Scale   Offset    Horizontal Scale   Position
channel 2     500 mV/div       -260 mV   1.00000 ns/div     2.3530 ns
channel 3     600 mV/div       1.21 V    1.00000 ns/div     2.3530 ns
```

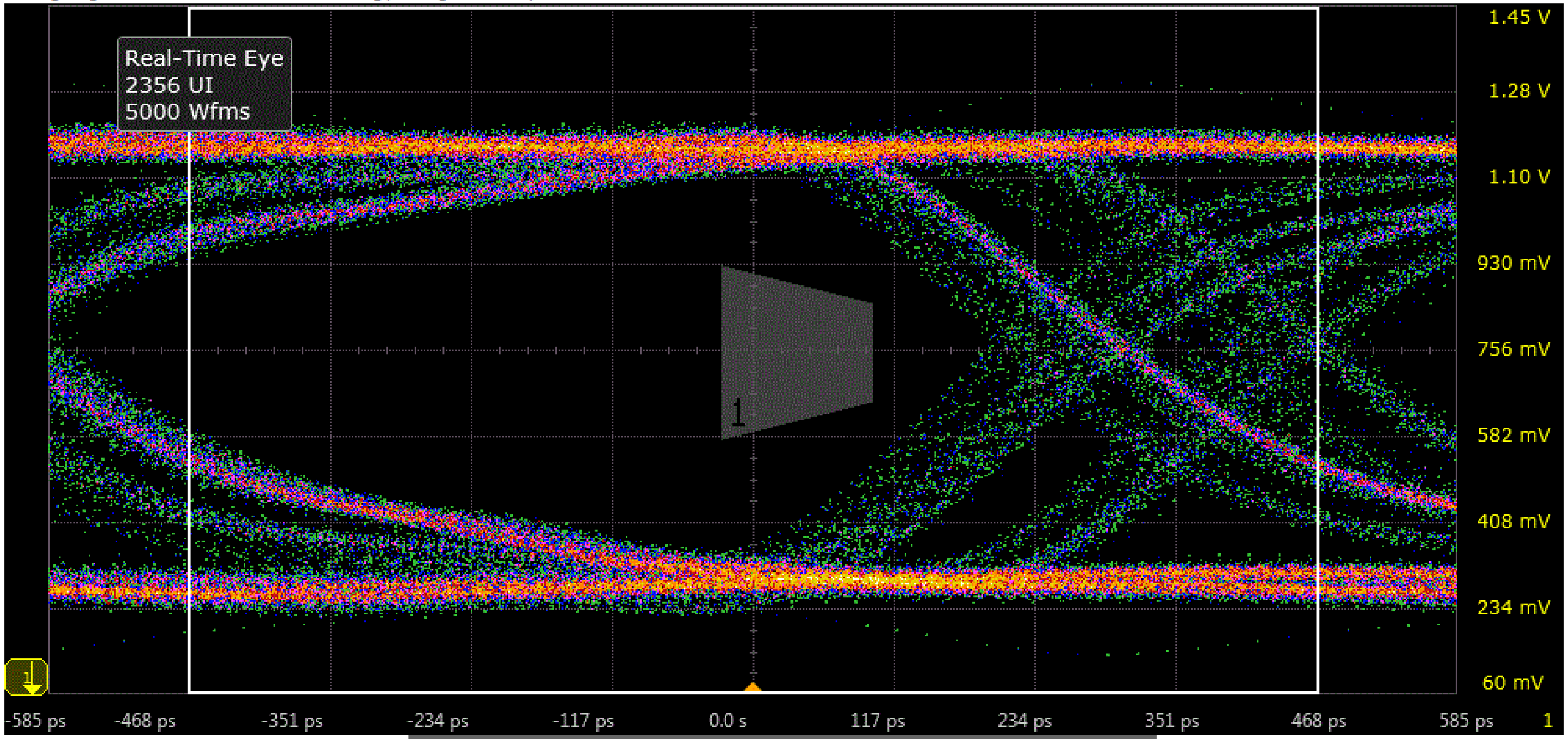
## DQS-DQ相位错误



```
Scales -------------------------------------------------------
                 Horiz Scale  Position     Vertical Scale  Offset
Channel 1        200.0ps/div  0.000s       196.0mV/div     750.9mV
Function 1       200.0ps/div  0.000s       1.000V/div      -81.75mV
```

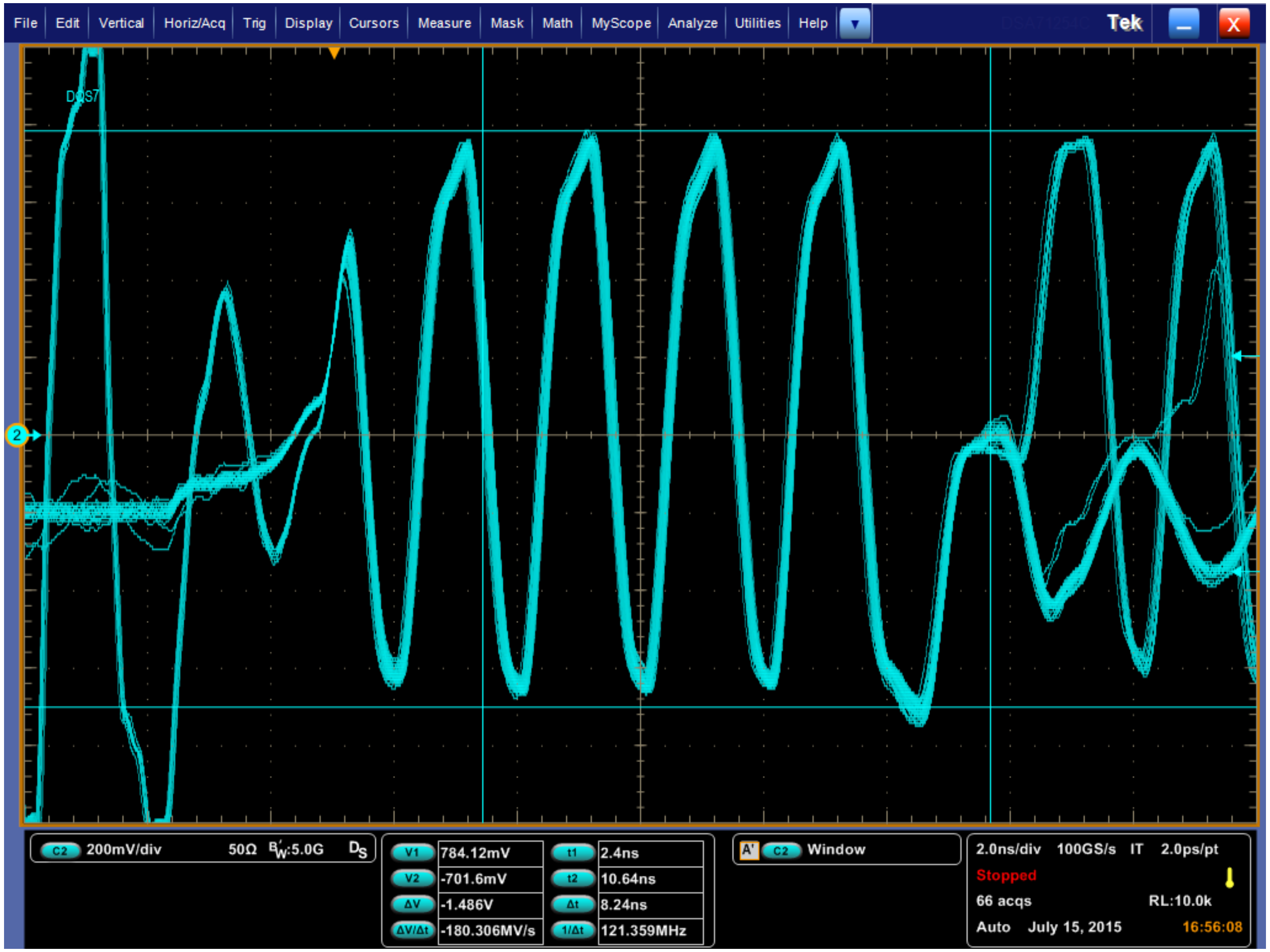
## DQS-DQ相位错误

Keysight Infiniium : Tuesday, August 11, 2015 10:11:11 PM



```
Scales -----------------------------------------------------------
             Horiz Scale  Position    Vertical Scale  Offset
Channel 1    188.0 ps/div 0.000s      174.0 mV/div    756.0 mV
Function 1   188.0 ps/div 0.000s      500.0 mV/div    -93.77 mV
```

## DQS-DQ相位错误



C2 200mV/div 50Ω BW:5.0G

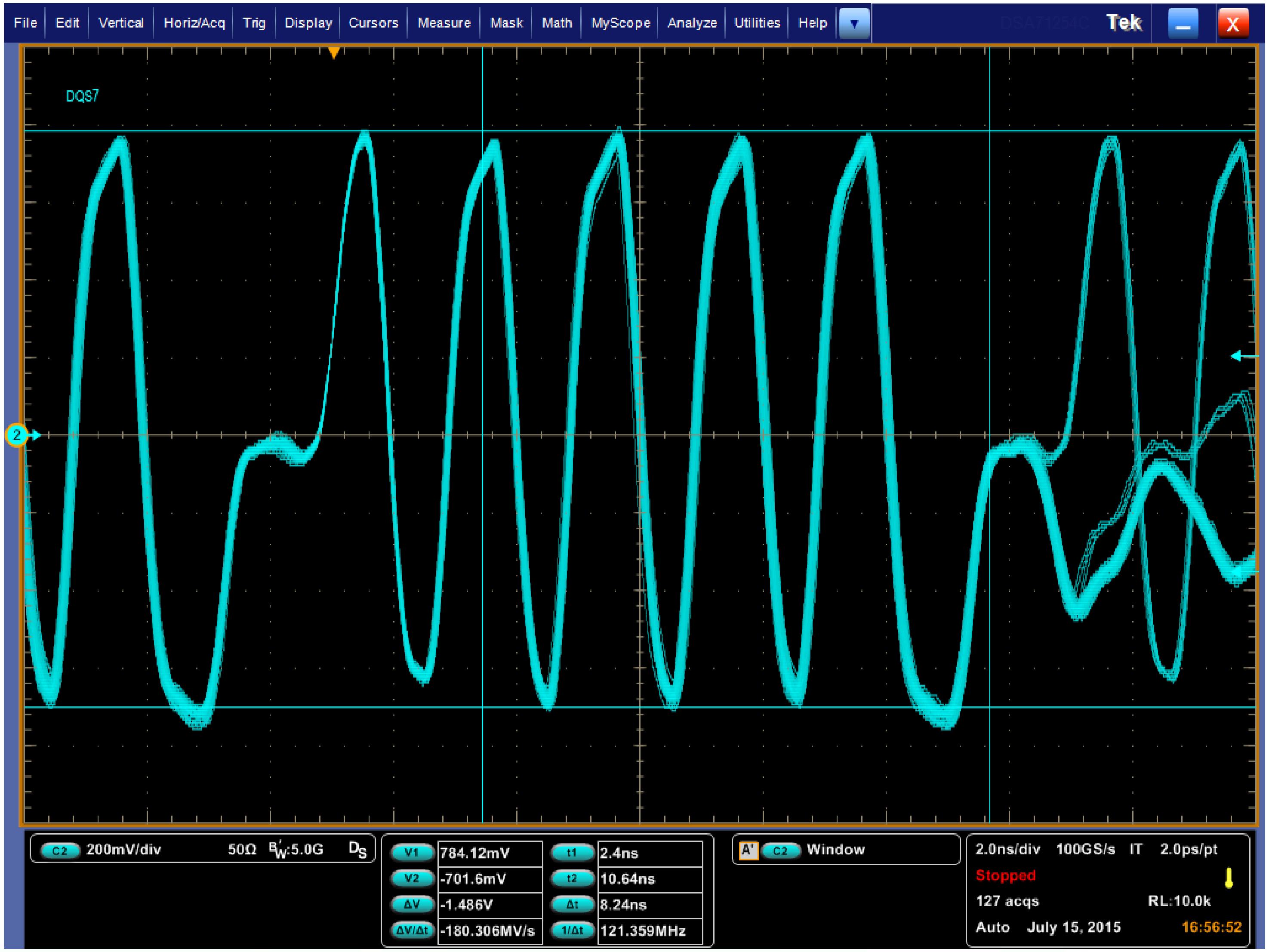
V1 784.12mV
V2 -701.6mV
ΔV -1.486V
ΔV/Δt -180.306MV/s

t1 2.4ns
t2 10.64ns
Δt 8.24ns
1/Δt 121.359MHz

A' C2 Window

2.0ns/div 100GS/s IT 2.0ps/pt
Stopped
66 acqs RL:10.0k
Auto July 15, 2015 16:56:08

## DQS-DQ相位正确



C2 200mV/div 50Ω BW:5.0G

V1 784.12mV t1 2.4ns
V2 -701.6mV t2 10.64ns
ΔV -1.486V Δt 8.24ns
ΔV/Δt -180.306MV/s 1/Δt 121.359MHz

A' C2 Window

2.0ns/div 100GS/s IT 2.0ps/pt
Stopped
127 acqs RL:10.0k
Auto July 15, 2015 16:56:52

谢谢！

Thanks!