VT3514 Vendor define command flow

V0.1 draft 20150108

# 背景

VT3514兼容SATA/AHCI/NVME三种mode，为了尽量减少后续客户support和host debug tool的effort，我们需要尽量统一host上各种tool的接口以及在device端实现的接口。

本文旨在定义vendor define command的flow和接口数据结构，这个flow和数据接口兼容SATA/AHCI/NVME三种mode。目标是host上一套tool在三种mode下都能使用，并且host端和Device需要adapter的代码尽量少。

# 接口及命令分类

为了统一vendor define命令传递的参数，以SATA能传递的最大参数为参考。Host->Device使用Type(1Byte) + param(9Bytes), Device->host使用1DW。其他数据/参数通过DMA传输。这个两个DW在AHCI/NVME下不包括host memory address。表2-1描述了基本命令类型，更详细的命令及包含的参数接口定义见vend\_cmd\_interface.xls。

table 2‑ Vendor command table

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| HostCmd | inband | 1byte |  | 9 bytes | | | | | |
| Type | SubType | P1 | | P2 | P3 | P4 | P5 | P6 |
| Device Control | 0xFF | >=8 | TBD | | TBD | TBD | TBD | TBD | TBD |
| MEM\_Read | 0xFF | 1 | McuID | | Device Addr | DataLen |  |  |  |
| MEM\_Write | 0xFF | 2 | McuID | | Device Addr | DataLen |  |  |  |
| Flash\_Read | 0xFF | 3 | McuID | | Block | Page | Plnmode | Pln | PuMask |
| Flash\_Write | 0xFF | 4 | McuID | | Block | Page | Plnmode | Pln | PuMask |
| Flash\_Erase | 0xFF | 5 | McuID | | Block | Page | Plnmode | Pln | PuMask |

# 命令处理流程

MCU0从host收到一条vendor define命令(0xff)，通过SubType看判断是什么命令，如果是Subtype >= 8则表示是一条Device Control命令，如果<8则是6条通用命令之一。下面详细定义每类命令的处理流程。

## Device Control

Device Control命令会根据SubType的不同再有不同的流程，这个Subtype会有非常多，不在这里一一列举，后面会附上一个subtype的详细列表。但是整体就分为两类，一类(A)是MCU0自己处理就完事了，MCU12不知道有任何事情发生，另一类(B)就是需要MCU12也进行处理。

(A). MCU0处理完，把处理结果配置WBQ或者D2H fis，直接回给host，命令结束。

(B). MCU0通过一个VIA\_DEV\_CTRL的subcmd，经过必要的预处理之后，给对应的subsystem发SCmd，这条SubCmd的参数包括subtype/9Byteparam/cmdtag/1DWRetValue。Subsystem处理之后把结果返回到1DWRetValue域中，MCU0根据命令类型判断是否把这个结果return给host。



3‑ VIA Device Control command flow

## MEM\_Read

MEM read包括对DRAM\_Read/SRAM\_Read的操作，因为部分SRAM跟MCU相关，其他MCU不能访问，因此host在读取的时候需要指定对应的McuID。而对于DRAM的访问，必须从MCU0走。

对于MCU0处理的MEM\_Read，MCU0把数据准备好放在tempbuffer中，给MCU12发一条RAW\_DATA\_REQ的Subcmd，MCU12按照RAW\_DATA\_REQ正常处理即可。

对于需要MCU12处理的MEM\_Read，使用subcmd RAW\_DATA\_REQ，MCU12根据DeviceAddr把数据copy到subcmd指定的teampbuffer中，然后HostDataIO。



3‑ VIA MEM Read command flow

## MEM Write

MEM Write包括对DRAM Write/SRAM Write的操作，因为部分SRAM跟MCU相关，其他MCU不能访问，因此host在读取的时候需要指定对应的McuID。而对于DRAM的访问，必须从MCU0走。

对于MCU0处理的MEM Write，MCU0准备好放数据的tempbuffer，给MCU12发一条RAW\_DATA\_REQ的Subcmd，MCU12按照RAW\_DATA\_REQ正常处理。当cachestatus被HW清了之后，MCU12结束这条subcmd，MCU0收到之后，把tempbuffer中的数据搬移到DeviceAddr中。

对于需要MCU12处理的MEM Write，使用subcmd RAW\_DATA\_REQ，MCU12把数据接收到tempbuffer，然后copy到DeviceAddr中，然后通知MCU0命令结束。



3‑ VIA MEM Write flow

## Flash\_Read

Flash的read/write其实是一个组合操作，需要与前面一笔MEM Read/Write配合，这里只关注其中一部，组合的事情由host完成。

对于一笔Flash read来说，device需要做的事情：

1. MCU0准备一笔subcmd，type是RAW\_DATA\_REQ，参数见[vendor cmd table](#vendor_cmd_table)，发给对应的subsys。
2. MCU12把data 从flash中读出来，data放在之前(Var Table)约定的DRAM buffer中。读flash结果的status放到tempbuffer中。
3. MCU12通过一笔HostDataIO把tempbuffer中的内容回给host。
4. MCU12通知MCU0这笔Subcmd结束，总的status放在RetValue中。
5. MCU0把RetValue回给host，这笔host command结束。

然后，host会再发MEM read来取具体的数据。



3‑ VIA Flash Read flow

## Flash\_Write

host会先发MEM Write，把数据先写入到VarTable约定的DRAM buffer中。然后再发flash write命令。对于一笔Flash Write，device需要做的事情。

1. MCU0准备一笔subcmd，type是RAW\_DATA\_REQ，参数见[vendor cmd table](#vendor_cmd_table)，发给对应的subsys。
2. MCU12把DRAM中的data写入到Flash中。
3. MCU12把写Flash的status放到tempbuffer中。
4. MCU12把tempbuffer通过HostDataIO回给host。
5. MCU12通知MCU0这笔SubCmd结束。
6. MCU0回给host，这笔host command结束。



3‑ VIA Flash write flow

## Flash\_Erase

对于一笔Flash Erase，device需要做的事情。

1. MCU0准备一笔subcmd，type是RAW\_DATA\_REQ，参数见[vendor cmd table](#vendor_cmd_table)，发给对应的subsys。
2. MCU12把对应flash block擦除掉。
3. MCU12把erase Flash的status放到tempbuffer中。
4. MCU12把tempbuffer通过HostDataIO回给host。
5. MCU12更新RetValue，通知MCU0这笔SubCmd结束。
6. MCU0把RetValue会给host，这笔host command结束。



3‑ VIA Flash Erase flow

# 代码结构划分

## Common部分：

|  |  |  |  |
| --- | --- | --- | --- |
| MCU0 | L0 | L0\_ViaCmd.h L0\_ViaCmd.c | 完成此文档定义的公共部分及DevCtrl命令 |
| MCU12 | L1 | FW\_ViaCmd.h FW\_ViaCmd.c | 主要实现DevCtrl命令，及为vendor define新增的SCMD |

## Adapter部分

|  |  |  |
| --- | --- | --- |
| SATA | L0\_SataVIACmd() | Sata mode下vendor define命令的入口，接收sata host传来的参数，调用L0\_ViaCmd.c中的接口（L0\_ViaHostCmd），然后回FIS给host完成命令。 |
| AHCI | L0\_AhciVIACmd() | AHCI mode下vendor define命令的入口，接收AHCI host传来的参数，调用L0\_ViaCmd.c中的接口（L0\_ViaHostCmd），然后配WBQ完成命令。 |
| NVME | L0\_NVMeVIACmd() | NVMe mode下vendor define命令的入口，接收NVMe host传来的参数，调用L0\_ViaCmd.c中的接口（L0\_ViaHostCmd），然后配WBQ完成命令。 |

注：Adapter部分只存在MCU0，命令到达到MCU12之后完全统一，不区分什么host。

## Key data structure

# 附录

## Vendor command table

见[table2-1 vendor command table](#vendor_cmd_table)

## Device Control command table

TBD

## Var Table

TBD

## Scmd Table

TBD

## Figure list

3‑1 VIA Device Control command flow

3‑2 VIA MEM Read command flow

3‑3 VIA MEM Write flow

3‑4 VIA Flash Read flow

3‑5 VIA Flash write flow

3‑6 VIA Flash Erase flow