**目录**

[第1章 PS/2 3](#_Toc47453628)

[1.1 物理连接 3](#_Toc47453629)

[1.2 电气属性 3](#_Toc47453630)

[1.3 数据示意图 3](#_Toc47453631)

[1.4 总线上数据的收发 3](#_Toc47453632)

[1.4.1 设备向主机发送数据 3](#_Toc47453633)

[1.4.2 PS/2设备向PC机读取一个字节 4](#_Toc47453634)

[1.4.3 主机发送数据到设备 5](#_Toc47453635)

[第2章 LPC 6](#_Toc47453636)

[2.1 物理连接 6](#_Toc47453637)

[2.2 电气属性 6](#_Toc47453638)

[2.3 Memory, I/O and DMA Cycle 7](#_Toc47453639)

[2.3.1 Start 7](#_Toc47453640)

[2.3.2 Cycle Type / Direction (CyCTYPE + DIR) 7](#_Toc47453641)

[2.3.3 Size 8](#_Toc47453642)

[2.3.4 Turn-Around (TAR) 8](#_Toc47453643)

[2.3.5 ADDR 8](#_Toc47453644)

[2.3.6 Channel 8](#_Toc47453645)

[2.3.7 DATA 8](#_Toc47453646)

[2.3.8 SYNC 9](#_Toc47453647)

[2.3.9 SYNC TimeOut 9](#_Toc47453648)

[2.3.10 SYNC Error Indication 9](#_Toc47453649)

[2.3.11 LFRAME# 10](#_Toc47453650)

[2.3.12 Start of Cycle 10](#_Toc47453651)

[2.3.13 Abort机制 11](#_Toc47453652)

[2.4 内存读写位域定义 11](#_Toc47453653)

[2.4.1 启动内存读 12](#_Toc47453654)

[2.4.2 启动内存写 12](#_Toc47453655)

[2.5 IO读写域定义 13](#_Toc47453656)

[2.5.1 主机启动IO读操作 14](#_Toc47453657)

[2.5.2 主机启动IO写操作 14](#_Toc47453658)

[2.6 Firmware Memory Cycle 15](#_Toc47453659)

[2.6.1 start 15](#_Toc47453660)

[2.6.2 IDSEL (Device Select) 15](#_Toc47453661)

[2.6.3 MADDR (Memory Address) 15](#_Toc47453662)

[2.6.4 MSIZE (Memory Size) 15](#_Toc47453663)

[2.6.5 TAR 16](#_Toc47453664)

[2.6.6 SYNC 16](#_Toc47453665)

[2.6.7 DATA 16](#_Toc47453666)

[2.6.8 Protocol 16](#_Toc47453667)

[2.6.9 Preamble 16](#_Toc47453668)

[2.6.10 Firmware Memory Read Cycle 17](#_Toc47453669)

[2.6.11 Firmware Memory Write Cycles 17](#_Toc47453670)

[2.6.12 Error Reporting 18](#_Toc47453671)

[2.6.13 固件存储器字段定义 18](#_Toc47453672)

[2.6.14 主机启动固件存储器读操作 19](#_Toc47453673)

[2.6.15 主机启动固件存储器写操作 19](#_Toc47453674)

[第3章 eSPI 20](#_Toc47453675)

[第4章 SMBus 21](#_Toc47453676)

[第5章 PECI 22](#_Toc47453677)

[第6章 ACPI 23](#_Toc47453678)

# PS/2

## 物理连接

一般之后4个引脚有意义；分别是Clock(时钟脚)、DATA(数据脚)、+5V(电源脚)和Ground(电源地)，其中时钟线和数据线必须接上拉电阻。

## 电气属性

一般两设备间传输数据的最大时钟频率是33kHz，大多数ps/2设备工作在10~20kHz。推荐值在 15kHz左右，也就是说，Clock(时钟脚)高、低电平的持续时间都为40μs。

## 数据示意图

|  |  |
| --- | --- |
| 1 start bit. This is always 0 | 1 个起始位 总是为 0 |
| 8 data bits, least significant bit first. | 8 个数据位 低位在前 |
| 1 parity bit (odd parity) | 1 个校验位 奇校验 |
| 1 stop bit. This is always 1 | 1 个停止位 总是为 1 |
| 1 acknowledge bit | 1 个应答位 仅在主机对设备的通讯中 |

## 总线上数据的收发

### 设备向主机发送数据

设备到主机的通讯 当时钟为高 数据线改变状态 在时钟信号的下降沿，数据被锁存。

#### PS/2设备向PC机发送一个字节

(1)等待时钟为高电平，并延时50us；

(2)检测判断时钟信号是否为高，为高，则向下执行，为低，则转到(1)；

(3)检测数据线是否为高，如果为高则继续执行，如果为低，则放弃发送(此时PC机在向PS/2设备发送数据，所以PS/2设备要转移到接收程序处接收数据)；

(4)延时20μs(如果此时正在发送起始位，则应延时４0μs)；

(5)输出起始位(0)到数据线上。这里要注意的是：在送出每一位后都要检测时钟线，以确保PC机没有抑制PS/2设备，如果有则中止发送；

(6)输出8个数据位到数据线上；

(7)输出校验位；

(8)输出停止位(1)；

(9)延时30μs(如果在发送停止位时释放时钟信号则应延时50μs)；

#### 发送单个位

(1)准备数据位(将需要发送的数据位放到数据线上)；

(2)延时20μs；

(3)把时钟线拉低；

(4)延时40μs；

(5)释放时钟线；

(6)延时20μs。

### PS/2设备向PC机读取一个字节

这时为PC主机发送数据，所以PC机通过下拉时钟线大于 100μs来抑制通讯，然后再释放总线来发送数据，主机此时在时钟线变为低时准备数据到数据线，并在时钟上升沿锁存数据。而PS/2设备则要配合PC机才能读到准确的数据。

#### PS/2设备向PC机读取一个字节

(1)等待时钟线为高电平。

(2)判断数据线是否为低，为高则错误退出，否则继续执行。

(3)读地址线上的数据内容，共8个bit，每读完一个位，都应检测时钟线是否被PC机拉低，如果被拉低则要中止接收。

(4)读地址线上的校验位内容，1个bit。

(5)读停止位。

(6)如果数据线上为0(即还是低电平)，PS/2设备继续产生时钟，直到接收到1且产生出错号为止(因为停止位是1，如果PS/2设备没有读到停止位，则表明此次传输出错)。

(7) 输出应答位。

(8) 检测奇偶校验位，如果校验失败，则产生错误信号以表明此次传输出现错误。

(9)延时４5 μs，以便PC机进行下一次传输。

#### 读取一个位的步骤如下：

(1)延时20μs；

(2)把时钟线拉低

(3)延时４0μs

(4)释放时钟线

(5)延时20μs

(6)读数据线。

#### 发出应答位

(1)延时15μs；

(2)把数据线拉低；

(3)延时5μs；

(4)把时钟线拉低；

(5)延时４0μs；

(6)释放时钟线；

(7)延时5μs；

(8)释放数据线。

### 主机发送数据到设备

时钟都是由PS/2设备发出。主机要向设备发送数据，需要把时钟和数据线设置为“请求发送”状态。主机拉低信号线表示抑制通讯。

#### 主机发送数据到从机

(1) 主机拉低时钟线至少100us

(2) 吧数据线拉低，相当于请求发送信号，也就是起始位。

(3) 主机释放数据线。

(4) 主机释放数据线，并等待设备把时钟线拉低，主机拉低时钟到设备产生时钟时间不超过15ms

(5) 主机发送8位数据，将数据放置在数据线上，每设置一个数据都得等待设备吧时钟线拉高，在时钟上升沿时，对数据进行锁存，等待设备在把时钟拉低，如此反复8次发送数据。

(6) 主机释放数据线。如果如果主机在第11个时钟脉冲后不释放数据线，设备将继续产生时钟脉冲直到数据线被释放然后设备将产生一个错误。

(7) 等待设备把数据线拉低。

(8) 等待设备吧时钟线线拉低。

(9) 等待设备释放时钟线和数据线。

# LPC

## 物理连接

LPC同时又是一个灵活的总线，它包括7个必选信号及6个可选信号。在实际应用中，并不是所有的信号都是必需的，可以根据情况对可选信号进行增减。

LPC 总线必选信号列表

|  |  |  |  |
| --- | --- | --- | --- |
| 信号 | 外设 | Host 设备 | 信号描述 |
| LAD[3:0] | I/O | I/O | 命令/数据/地址复用信号 |
| LFRAME# | I | O | 指示一个操作循环的开始 |
| LRESET# | I | I | 复位信号 |
| LCLK | I | I | 33MHz时钟 |

LPC总线可选信号列表

|  |  |  |  |
| --- | --- | --- | --- |
| 信号 | 外设 | Host 设备 | 信号描述 |
| LDRQ# | O | I | DMA master 总线请求 |
| SERIRQ | I/O | I/O | 串行化中断信号 |
| CLKRUN# | OD | I/OD | Clock Run 信号 |
| LPME# | OD | I/OD | 电源管理唤醒 |
| LPCPD# | I | O | Power Down |
| LSMI | OD | I | SMI 信号 |

**注: (1) OD 表示漏极开路; (2) 带#的信号线表示低电平有效；**

## 电气属性

上拉电阻的硬件要求。

|  |  |
| --- | --- |
| **Signal Name** | **Pull-Up** |
| LAD[3:0] | 15k - 100k ohm |
| LDRQ[1:0]# | 15k - 100k ohm |

## Memory, I/O and DMA Cycle

LPC总线支持多种总线Cycle，比如内存读、内存写、IO读、IO写、DMA读、DMA写等.一个Cycle通常按照如下的流程进行:

---主机Host有效FRAME#信号指示一个Cycle开始，并将相应的信息放在LAD[3:0]上.

---主机Host根据Cycle类型驱动相应的信息到LAD[3:0]上，比如DMA通道，访问地址等.同时Host也将Cycle的类型，读写的方向，传输数据大小等信息驱动到总线上。

---主机然后根据Cycle类型不同选择进行驱动数据或者是移交总线控制权.

---.外设获取总线后，将相应的数据驱动到LAD[3:0]上.表示该Cycle完成.

---外设释放总线控制权。至此该Cycle结束。

根据LPC信号线的不同组合及在整个Cycle中出现的位置，一个LPC总线Cycle可以划分为多个场/域(Field)，下面就分别对I0读写Cycle所涉及的Field进行简单说明。

### Start

这个场用于指示一个传输的开始或者结束。当FEAME#信号有效时所有的外设都要监视LAD[3:0]信号，并在FRAME#信号有效的最后一个时钟后进入START场。LAD[3:0]的值编码如下表。

|  |  |
| --- | --- |
| Bits[3:0] | 定义 |
| 0000 | 开始一个Cycle，用于外设Memory，IO，DMA操作 |
| 0001 | 保留 |
| 0010 | 相应 bus master0 |
| 0011 | 响应 Bus master1 |
| 0100-1100 | 保留 |
| 1101 | 开始Firmware Memory Read |
| 1110 | 开始Firmware Memory Write |
| 1111 | Stop/Abort 结束一个Cycle |

### Cycle Type / Direction (CyCTYPE + DIR)

该场由Host 驱动，对Cycle的传输类型（memory, IO, DMA）以及传输方向进行说明。LAD[0]在该场中被保留。作为外设应该忽略。可以得值见表。

|  |  |  |
| --- | --- | --- |
| Bit[3:2] | Bit[1] | 定义 |
| 00 | 0 | IO读 |
| 00 | 1 | IO写 |
| 01 | 0 | Memory 读 |
| 01 | 1 | Memory 写 |
| 10 | 0 | DMA 读 |
| 10 | 1 | DMA写 |
| 11 | X | 保留 |

### Size

该字段是一个时钟。 它在DMA传输中由主机驱动，在总线主存储器传输中由外设驱动，以确定要传输多少字节。 位[3：2]被保留，必须由驱动程序将其驱动为“ 00b”，并且必须被目标忽略。 其余位编码如下：

|  |  |
| --- | --- |
| Bit[1:0] | Size |
| 00 | 8 bit（1byte） |
| 01 | 16bit(2 bytes) |
| 10 | 保留 |
| 11 | 32bit(4 bytes) |

### Turn-Around (TAR)

TAR 场占用两个时钟宽度。Host驱动该场将总线控制权利移交给外部设备（进行数据读）；而外设驱动该场将总线控制权移交回来。

在第一个时钟相内，外设或Host驱动LAD[3:0]为1111b,在第二个时钟相内，外设或Host将LAD[3:0]置为3态。

### ADDR

这个场在进行IO操作时占用4个时钟，进行Memory操作时占用8个时钟，target 存取时由于Host 驱动，bus master 存取时由外设驱动。DMA操作不需要该场。当该场被驱动时，高位最先被驱动到总线上。比如，Memory操作时，第一个时钟包含了Address[31:28],最后一个时钟包含了Address[3:0]。

### Channel

这是一个时钟宽度的字段，由主机在DMA周期上驱动，以向外设指示已授予哪个DMA通道。 [2：0]位包含DMA通道号，[3]位包含基于ISA编码的TC（终端计数）线。 在目标或总线主控传输时，不会驱动该字段。

### DATA

该场占用两个时钟周期，用于传送一个字节数据。当数据流向外设时，该场由Host驱动;反之，当数据流向Host时，则该场由外设驱动。在传输过程的时候，低4位最先被驱动到总线上.具体的说，在第一个时钟，Data[3:0]被驰动，第二个时钟，Data[7:4]被驱动。

### SYNC

SYNC场用来加入等待状态。它占用时间为时钟周期的整数倍.在target或者DMA传输操作时，该场由外设驱动:在bus master操作时，由Host驱动。可能的组合见表

|  |  |
| --- | --- |
| Bits[3:0] | 定义 |
| 0000 | 准备 |
| 0001-0100 | 保留 |
| 0101 | 短等待 |
| 0110 | 长等待 |
| 0111-1000 | 保留 |
| 1001 | Ready More(DMA only) |
| 1010 | 错误 |
| 1011-1111 | 保留 |

当设备需要插入等待周期时，应先驱动为“0101"(短时等待)或者“0110"(长等待)，等到Ready状态来到时，设备或选择驱动为“0000"(准备好)，" 1010"(错误)或者“1001”(Ready More)。

### SYNC TimeOut

总线上通常可能发生一下几种潜在的错误:

1.当Host发起一个Cycle (Memory,IO,DMA)后，但是总线上没有设备驱动SYNC场，当Host检测到有3个连续的时钟内都没有响应时，就认为总线上没有外设响应此次Cycle操作.

Host驱动一个Cycle(Memory,IO,DMA )，一个设备驱动一个有效的SYNC场来插入等待(LAD[3:0]='0101b’或者’0110b')，但是却不完成该Cycle，这种情况在外设锁定的时候就发生了。在这种情况下，Host应采取以下措施以保护总线死锁：

·假如SYNC是，0101b'，那么最大SYNC时钟周期被假设为8。当Host检测到有多于8个时钟周期的SYNC场，那么Host将取消这个Cycle。

·假如SYNC是，0110b'，那么这里将没有最大SYNC长度的限制。外设必须设计有保护机制来完成这个Cycle。

当由于host来驱动SYNC时，因为延迟的原因，它可能不得不插入大量的等待周期，但外设不应该认为有time-out 发生。

### SYNC Error Indication

通过将LAD[3:0驱动为‘10l0b', SYNC协议允许外设报告一个错误。

假如Host从外设读取数据，数据仍将在下两个时钟内传输，这个数据可能是无效的，但外设仍将传输它。假如Host是写数据到外设中，那么该数据其实已经被传输了。

在多字节DMA传输中，一个错误SYNC将会终止这个Cycle。因此，如果Host正企图进行一个4字节的传输，但当在传输第一个字节时就出错了，那么剩下的3个字节将不会被传输。

在 Host接收到一个错误SYNC后，它可以有多种处理方式.对于ISA总线，它可以有效IOCHK#信号。

### LFRAME#

Host采用LFRAME#信号来开始一个Cycle，当一个abort或time-out出现时来终止一个Cycle，该信号也被外设用来决定什么时候监视总线。

这个信号通常是提醒设备LAD[3:0]线包含了Cycle开始或结束的有关信息，此时所有外设需要监视总线并决定该Cycle是否与自己有关。这样做的好外在于FRAME#信号可以使设备内部进入低功耗状态。假如一个外设没有被访问时，它就不需要监视总线，这样一来，设备就可以关闭内部时钟或者将状态机停下来。

当外设监测到FRAME#有效后，他们在下一个时钟立即停止驱动L.AD[3:0)信号线，并继续监视下一个总线信息。

### Start of Cycle

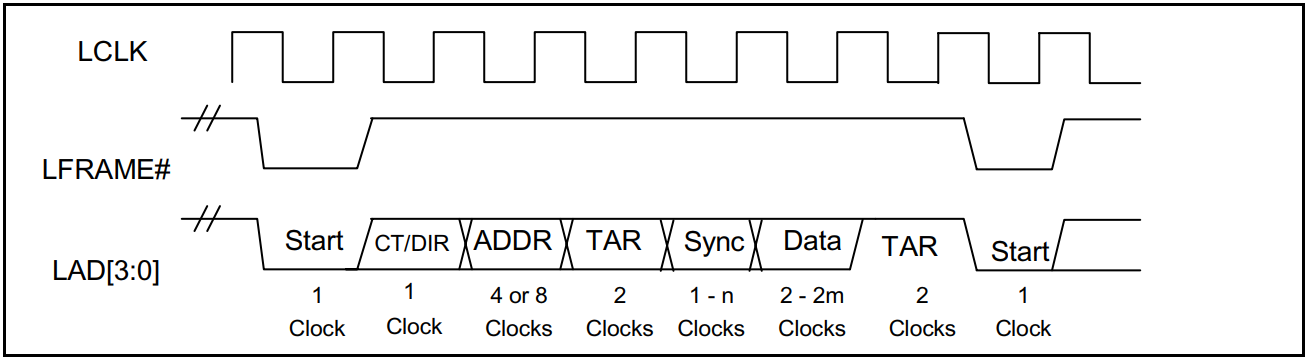
不同的Cycle都以相同的方式开始。

Host有效LFRAME#一个或多个时钟，并且驱动一个START值到LAD[3:0]信号线上。一旦发现LFRAME#信号有效后，所有的外设将停止驱动LAD[3:0],即使此时数据正在传输中。

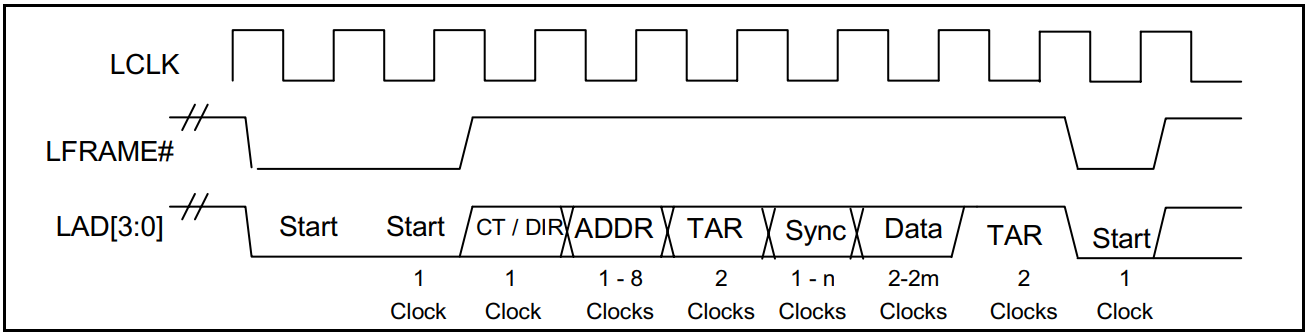
在Cycle的开始阶段，Host有可能保持FRAME#信号有效超一个时钟周期，START值也有可能发生变化。因此，外设必须总是使用最后一个值。比如，假如FRAME#有效两个时钟周期，那么外设应该忽略第一个时钟时的值而使用第二个时钟周期的值。

外设监视LAD[3:0]值并在LFRAME#有效时得到相应的START值，假如外设支持些种类型的Cycle，它应该试图解析Cycle余下的部分。假如不支持这个特定的类型，它将忽略Cycle余下的部分，直到FRAME#再次有效。

当最后一个START值被驱动后，Host就准备开始一个Cycle，并无效FRAME#信号。外设则根据START值开始进入工作状态。



典型的FRAME#时序

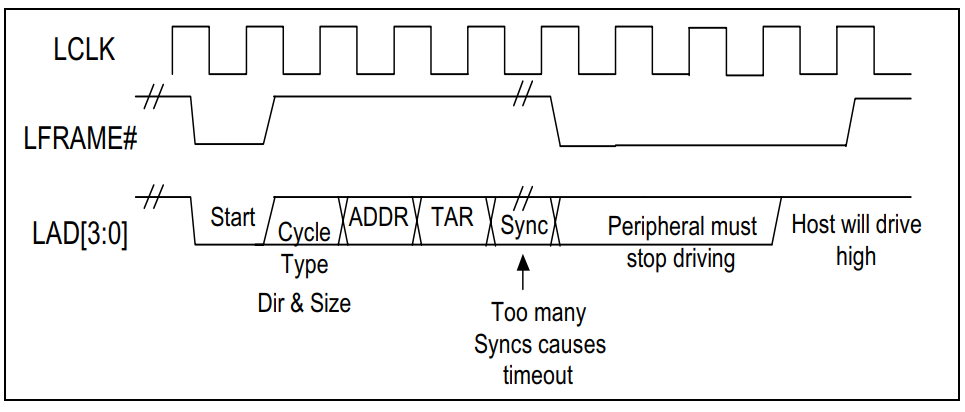


拓展的FRAME#时序

### Abort机制

FRAME#信号用来将一个外设kick出总线接口.Host驱动LFRAME#信号并驱动一个START值，’1111b’来产生一个Abort。因为有效LFRAME#通常表示一个Cycle的开始，所以即使当一个外设正在进行一个Cycle时，它必须认为Host正在取消当前的Cycle并且立刻停止驱动LAD[3:0]信号。为了确保一个Abort被正确识别，Host需要保持LFRAME#有效至少4个连续的时钟周期并且在第4个时钟内将LAD[3:0]驱动为’1111b'. Abort完成后，LFRAME#应被驱动为无效至少一个时钟周期。

Abort 通常发生在SYNC超时，Cycle 无响应或者一个设备驱动一个保留SYNC字的情况下。Abort被用来结束当前的Cycle以保证所有的设备进入下一个Cycle的准备状态。Abort也可以用来当作软件复位来使用。在使用长等待SYNC并不建议使用Abort机制。



Abort 机制

## 内存读写位域定义

在这些情况下，内存读取或写入周期用于内存映射的设备。 同步时间将取决于设备的速度。

|  |  |  |
| --- | --- | --- |
| **Field** | **# Clocks** | **Comment** |
| START | 1 | **Start of Cycle:** ‘0000b’ to indicate a start of a cycle. |
| CYCTYPE + DIR | 1 | **Cycle Type:** Indicates the type of cycle.Bits 3:2 must be ‘01b’ for memory cycle.Bit 1 indicates the direction of the transfer: ‘0b’ for read, ‘1b’ for write.Bit 0 is reserved. |
| ADDR | 8 | **Address Phase for Memory Cycle:** This is the 32-bit memory address.It is transferred with the most significant nibble first. |
| TAR | 2 | **Turn-Around Time:** The last component driving LAD[3:0] will drive it to “1111b” during the first clock, and tri-state it during the second clock. |
| SYNC | N | **Sync:** Allows peripheral or host to synchronize (add wait-states). Generally, the peripheral or host drives 0101 or 0110 until no more wait states are needed.At that point it will drive 0000.All other combinations are reserved.If the host sees a reserved combination, it is allowed to abortthe transfer. |
|  |  | |  |  | | --- | --- | | **Bits** | **Indication** | | 0000 | Sync Achieved with no error. | | 0101 | Indicates that Sync not Achieved yet, but the part is driving the bus. | | 0110 | Indicates that Sync not Achieved yet, but the part is driving the bus, and expects a long Sync. | | 1010 | Special Case: peripheral indicating errors.See Section 4.2.1.10 for details. | |
| Data | 2(1 byte) | **Data Phase:** The data byte is transferred least significant nibble first (D[3:0] on LAD[3:0], then D[7:4] on LAD[3:0]). |

### 启动内存读

|  |  |  |
| --- | --- | --- |
| **Memory Read** | **Driven By** | **Clocks** |
| START | Host | 1 |
| CYCTYPE + DIR | Host | 1 |
| ADDR | Host | 8 |
| TAR | Host | 2 |
| SYNC | Peripheral | 5 |
| DATA | Peripheral | 2 |
| TAR | Peripheral | 2 |
| **Total Clocks** |  | **21** |

在以上示例中，选择了5个时钟的SYNC值来读取第一个字节。 该值基于对EPROM等存储组件的已知访问时间，典型访问时间为120ns。

### 启动内存写

|  |  |  |
| --- | --- | --- |
| **Memory Read** | **Driven By** | **Clocks** |
| START | Host | 1 |
| CYCTYPE + DIR | Host | 1 |
| ADDR | Host | 8 |
| DATA | Host | 2 |
| TAR | Host | 5 |
| SYNC | Peripheral | 2 |
| TAR | Peripheral | 2 |
| **Total Clocks** |  | **17** |

在上面的示例中，假定外围设备不需要等待状态，因为存储设备将数据发布。 因此，只需准备就绪同步指示器

## IO读写域定义

在这些情况下，I / O读或写周期用于外设。 这些通常用于寄存器或FIFO访问，并且通常具有最小的同步时间。 字节之间的最小等待状态数为1。增强的并行端口周期将取决于连接器的速度。

|  |  |  |
| --- | --- | --- |
| START | 1 | **Start of Cycle:** ‘0000b’ to indicate a start of a cycle. |
| CYCTYPE +DIR | 1 | **Cycle Type:** Indicates the type of cycle.Bits 3:2 must be ‘00b’ for I/O cycle.Bit 1 indicates the direction of the transfer: ‘0b’ for read, ‘1b’ for write.Bit 0 is reserved and must be ignored by the peripheral. |
| ADDR | 4 | **Address Phase for I/O Cycle:** 16 bit I/O address transferred with the most significant nibble first. |
| TAR | 2 | **Turn-Around Time:** The last component driving LAD[3:0] will drive it to ‘1111b’ during the first clock, and tri-state it during the second clock. |
| Sync | N | **Synchronize:** Allows peripheral add wait-states.The peripheral drives ‘0101b’ or ‘0110b’ until no more wait-states are needed.At that point it will drive ‘0000b’.All other combinations are reserved. |
|  |  | |  |  | | --- | --- | | **Bits** | **Indication** | | 0000 | Sync Achieved with no error. | | 0101 | Indicates that Sync not Achieved yet, but the part is driving the bus. | | 0110 | Indicates that Sync not achieved yet, but the part is driving the bus, and expects a long Sync (probably due to EPP cycle). | | 1010 | Special Case: peripheral indicating errors.See Section 4.2.1.10for details. | |
| Data | 2 (1 byte) | **Data Phase:** The data byte is transferred with the least significant nibble first: (Data[3:0] on LAD[3:0], then Data[7:4] on LAD[3:0] ). |

### 主机启动IO读操作

|  |  |  |
| --- | --- | --- |
| **Memory Read** | **Driven By** | **Clocks** |
| START | Host | 1 |
| CYCTYPE + DIR | Host | 1 |
| ADDR | Host | 4 |
| TAR | Host | 2 |
| SYNC | Peripheral | 1 |
| DATA | Peripheral | 2 |
| TAR | Peripheral | 2 |
| **Total Clocks** |  | **13** |

在以上示例中，假定外设没有传递读取数据所需的等待状态，因为它很可能是从内部FIFO中拉出的。

### 主机启动IO写操作

|  |  |  |
| --- | --- | --- |
| **Memory Read** | **Driven By** | **Clocks** |
| START | Host | 1 |
| CYCTYPE + DIR | Host | 1 |
| ADDR | Host | 4 |
| DATA | Host | 2 |
| TAR | Host | 2 |
| SYNC | Peripheral | 1 |
| TAR | Peripheral | 2 |
| **Total Clocks** |  | **13** |

在以上示例中，假定外围设备上不需要等待状态。因此，所必要的是就绪同步指示器。

## Firmware Memory Cycle

本节介绍了用于系统BIOS固件的内存周期类型。

### start

该一个时钟字段指示一个周期的开始。 在最后一个时钟上LFRAME＃被采样为低电平有效。 下表中显示了用于循环的两个开始字段。

|  |  |
| --- | --- |
| AD[3:0] | 描述 |
| 1101 | Fireware memory Read |
| 1110 | Fireware memory Write |

### IDSEL (Device Select)

该一个时钟字段用于指示正在选择多个固件组件中的哪个。 将在此时钟期间通过AD [3：0]传输的四个位与绑定到固件组件引脚上的值进行比较。 如果存在匹配项，则固件组件将继续对cycle进行解码，以确定读取时请求了哪些字节或写入时要更新哪些字节。 如果不匹配，则固件组件可能会丢弃剩余的周期并进入待机电源状态。

### MADDR (Memory Address)

这是一个7时钟字段，提供28位存储器地址。 每个存储设备最多可容纳256MB，总共4GB的可寻址空间,先传高位。

### MSIZE (Memory Size)

下表中显示了此字段的编码。所有支持固件存储周期的设备都需要单字节传输。 多字节传输是可选的，主机和设备都需要一种机制来通知软件可用的支持（即功能寄存器位）。 注意，此机制的定义超出了本规范的范围。对于多字节写入，FLASH存储设备可能需要增加的编程电压（Vpp）才能执行4字节存储写入。如果没有更高的编程电压，但外设需要支持更高的编程电压以支持4字节存储器写，则应使用1或2字节存储器写。

|  |  |  |
| --- | --- | --- |
| Bits | Direction | Size of Transfer |
| 0000 | R/W | 1 byte |
| 0001 | R/W | 2bytes(可选)，访问必须与WORD边界对齐 |
| 0010 | R/W | 4bytes(可选)，访问必须与WORD边界对齐 |
| 0011 |  | Reserved |
| 0100 | R | 16 bytes,访问必须与16 bytes边界对齐 |
| 0101-0110 |  | Reserved |
| 0111 | R | 128 bytes,访问必须与128 bytes边界对齐 |
| 1000-1111 |  | reserved |

### TAR

TAR 场占用两个时钟宽度。Host驱动该场将总线控制权利移交给外部设备（进行数据读）；而外设驱动该场将总线控制权移交回来。

在第一个时钟相内，外设或Host驱动LAD[3:0]为1111b,在第二个时钟相内，外设或Host将LAD[3:0]置为3态。

与2.3.4描述一致。

### SYNC

SYNC场用来加入等待状态。它占用时间为时钟周期的整数倍.在target或者DMA传输操作时，该场由外设驱动:在bus master操作时，由Host驱动。可能的组合见表

|  |  |
| --- | --- |
| Bits[3:0] | 定义 |
| 0000 | 准备 |
| 0001-0100 | 保留 |
| 0101 | 短等待 |
| 0110 | 长等待 |
| 0111-1000 | 保留 |
| 1001 | Ready More(DMA only) |
| 1010 | 错误 |
| 1011-1111 | 保留 |

当设备需要插入等待周期时，应先驱动为“0101"(短时等待)或者“0110"(长等待)，等到Ready状态来到时，设备或选择驱动为“0000"(准备好)，" 1010"(错误)或者“1001”(Ready More)。

与2.3.8描述一致。

### DATA

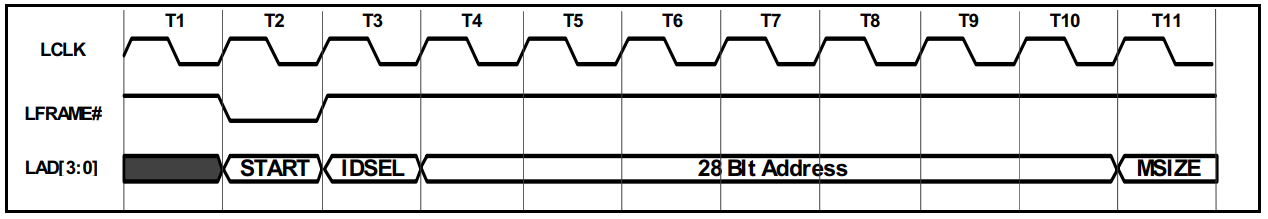
该字段为（2 \* N）个时钟宽度，表示由MSIZE字段确定的“ N”个数据字节的传输。 当数据流向外设时，主机在固件存储周期中驱动它（写周期），当数据流向主机时，则由外设驱动它（读周期）。每个字节的数据低位在前。这意味着对于每个字节，在第一个时钟上驱动Data [3：0]，在第二个时钟上驱动Data [7：4]。 这也意味着每个后续数据字节的地址通过传输顺序递增。

### Protocol

固件存储周期使用一系列事件，这些事件以START字段（LFRAME＃处于活动状态，并带有适当的AD [3：0]组合）开始，并以数据传输结束。

### Preamble

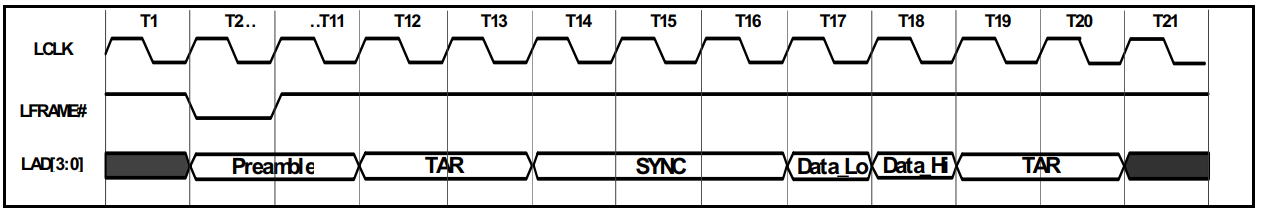
固件存储器周期的初始化如下图所示。固件存储器事务以LFRAME＃变低并在AD [3：0]上驱动START字段开始。 对于固件内存读取周期，START字段必须为“1101b”；对于固件内存写周期，START字段必须为“1110b”。在START字段之后是IDSEL字段。该字段就像片选一样，它指示哪个设备应响应当前事务。接下来的七个时钟是28位地址，从该地址开始在所选器件中进行读取。接下来，MSIZE值指示要传输的字节数。



Firmware Memory Cycle Preamble

### Firmware Memory Read Cycle

对于读取周期在Preamble之后，主机驱动TAR字段以将总线所有权授予固件组件。 在TAR阶段的第二个时钟之后，目标设备采用总线并开始驱动SYNC值。当目标设备准备就绪时，它将驱动数据（首先是最低有效字节），直到传输完所有数据，然后再进行TAR循环，以将控制权交还给主机。

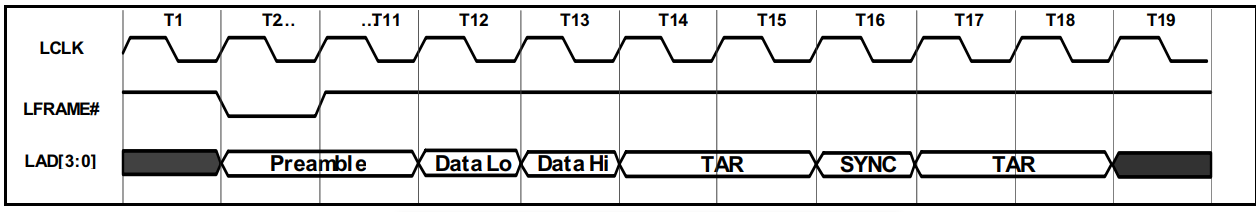


Firmware Memory Cycle Single Byte Read

上图显示了一个设备传输1字节数据并需要3个SYNC时钟来访问数据的过程，因为访问时间可以在地址阶段完成后开始，因此TAR阶段的两个时钟可以被视为设备访问时间的一部分。例如，访问时间为120ns的设备可以为SYNC阶段的时钟1和2设置“0101b”，为SYNC阶段的最后一个时钟设置“ 0000b”。 如果设备在Preamble阶段结束时开始访问，则相当于5个时钟的访问时间。一旦实现了SYNC，设备便会在随后的两个时钟中返回数据，并通过TAR阶段将总线所有权返还给主机。对于多字节读取，附加数据字节以紧接第一个字节之后的顺序传输（即在上图中的时间T18和T19之间），然后是TAR阶段。

### Firmware Memory Write Cycles

对于读取周期在Preamble之后，所有支持固件内存写周期的设备都必须支持单字节写操作。 进一步支持多字节固件存储器写周期是可选的。



Firmware Memory Cycle Single Byte Write

上图显示了一个固件存储器写周期，其中传输了一个字节。 主机将MSIZE值声明为0。在地址传输之后，2时钟数据阶段开始。在数据阶段之后，总线所有权通过TAR周期转移到固件组件。在TAR阶段之后，设备必须声明SYNC值“ 0000b”（就绪）或“ 1010b”（错误），以指示是否已接收到数据。有关可能的错误的说明，下部分有说明错误码。 然后，在另一个TAR阶段将总线所有权交还给主机。 对于多字节写入，附加数据字节紧接在第一个字节之后（即在上图中的时间T13和T14之间）以顺序方式传输，然后是第一个TAR阶段。

固件存储器写入仅允许一个时钟进入SYNC阶段。 固件存储器写周期结束时的TAR + SYNC + TAR阶段必须恰好是5个时钟。

### Error Reporting

固件组件不得通过LPC接口报告错误情况。 它只能报告等待状态和“就绪”状态。 它可以选择在内部记录要调试的错误，但一定不能通过LPC接口本身发出错误信号

### 固件存储器字段定义

固件存储器的读或写周期旨在用于PC系统引导固件，尽管它们可用于任何存储器周期。 同步时间将取决于设备的速度。

ADDR字段为28位，并以最高有效半字节优先发送。 尽管支持完整的28位寻址，但是存储设备可能支持的位数远少于此。

|  |  |  |
| --- | --- | --- |
| **Field** | **# Clocks** | **Comment** |
| START | 1 | **Start of Cycle:** ‘1101b’ to indicate a start of a firmware memory read cycle or ‘1110b’ to indicate start of a firmware memory write cycle. |
| IDSEL | 1 | **ID Select:** Selects the targeted firmware component based on device pin straps. |
| ADDR | 7 | **Address Phase for Firmware Memory Cycle:** This is the 28-bit memory address.It is transferred with the most significant nibble first.The device selected by IDSEL field uses this to address its internal memory array. |
| MSIZE | 1 | **Memory Size:** Indicates the number of bytes (N) to be transferred.See Table 4 above. |
| TAR | 2 | **Turn-Around Time:** The last component driving LAD[3:0] will drive it to “1111b” during the first clock, and tri-state it during the second clock. |
| SYNC | N | **Sync:** Allows peripheral or host to synchronize (add wait-states). Generally, the peripheral or host drives 0101 or 0110 until no more wait states are needed.At that point it will drive 0000.All other combinations reserved.If the host sees a reserved combination, it is allowed to abort the transfer. |
|  |  | |  |  | | --- | --- | | **Bits** | **Indication** | | 0000 | Sync Achieved with no error. | | 0101 | Indicates that Sync not Achieved yet, but the part is drivingthe bus | | 0110 | Indicates that Sync not Achieved yet, but the part is drivingthe bus, and expects a long Sync. | |
| Data | 2 \* N (N bytes) | **Data Phase:** The data byte is transferred least significant nibble first (D[3:0] on LAD[3:0], then D[7:4] on LAD[3:0]), and so on. |

### 主机启动固件存储器读操作

|  |  |  |
| --- | --- | --- |
| **Single Byte Memory Read** | **Driven By** | **Clocks** |
| START | Host | 1 |
| IDSEL | Host | 1 |
| ADDR | Host | 7 |
| MSIZE | Host | 1 |
| TAR | Host | 2 |
| SYNC | Peripheral | 3 |
| DATA | Peripheral | 2 |
| TAR | Peripheral | 2 |
| **Total Clocks** |  | **19** |

|  |  |  |
| --- | --- | --- |
| **128 Byte Memory Read** | **Driven By** | **Clocks** |
| START | Host | 1 |
| IDSEL | Host | 1 |
| ADDR | Host | 7 |
| MSIZE | Host | 1 |
| TAR | Host | 2 |
| SYNC | Peripheral | 3 |
| DATA | Peripheral | 256 |
| TAR | Peripheral | 2 |
| **Total Clocks** |  | **273** |

在上面的示例中，选择了3个时钟的SYNC值来读取第一个字节。

|  |  |  |
| --- | --- | --- |
| **4 Byte Memory Write** | **Driven By** | **Clocks** |
| START | Host | 1 |
| IDSEL | Host | 1 |
| ADDR | Host | 7 |
| MSIZE | Host | 1 |
| DATA | Host | 8 |
| TAR | Host | 2 |
| SYNC | Peripheral | 1 |
| TAR | Peripheral | 2 |
| **Total Clocks** |  | **23** |

### 主机启动固件存储器写操作

|  |  |  |
| --- | --- | --- |
| **Single Byte Memory Write** | **Driven By** | **Clocks** |
| START | Host | 1 |
| IDSEL | Host | 1 |
| ADDR | Host | 7 |
| MSIZE | Host | 1 |
| DATA | Host | 2 |
| TAR | Host | 2 |
| SYNC | Peripheral | 1 |
| TAR | Peripheral | 2 |
| **Total Clocks** |  | **17** |

在以上示例中，假定外围设备上不需要等待状态，因为存储设备将数据发布。 因此，所有必要的是就绪同步指示器。

## DMA

在LPC总线上，允许有8个DMA通道。

### DMA数据请求

LPC上的DMA通过使用来自外设的LDRQ＃线和来自主机的LAD [3：0]上的特殊编码来处理。 LPC接口支持Single, Demand, Verify, and Increment模式。 不支持Block, decrement, and cascade modes模式。 通道的0 -3是8位通道。 频道的5 -7是16位频道。 通道4被保留为通用总线主控器请求.

LPC接口上支持一种新的32位传输模式，该模式可用于根据每个请求从8位或16位通道传输多个字节，以提高吞吐量。

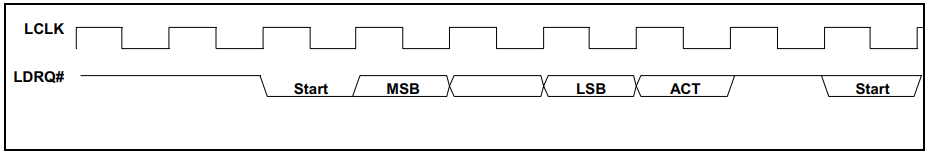
LDRQ＃与LCLK同步。其中LDRQ 接的上拉电阻，所以空闲状态为高电平，外设使用以下串行编码序列：

•通过将LDRQ＃置为低电平（起始位）来启动序列。 在空闲条件下，LDRQ＃为高。

•接下来的3位包含编码的DMA通道号（MSB在前）。

•下一位（ACT）指示对指示的DMA通道的请求是有效还是无效。 ACT位将为1（高）以指示它是否处于活动状态，而为0（低）以指示其是否处于活动状态。 ACT低的情况很少见，仅用于指示该信道的先前请求已被放弃。 有关放弃DMA请求的原因，请参见第6.3节。

•在启用/禁用指示之后，LDRQ＃信号必须至少在1个时钟周期内变为高电平。 在一个时钟之后，可以将LDRQ＃信号拉低到下一个编码序列。



DMA Request Assertion through LDRQ#

### 取消DMA请求

可以通过两种方式取消激活DMA请求：在错误情况下，通过将“ ACT”位设置为“ 0”来发送LDRQ＃消息，或者通常在DMA传输期间通过SYNC字段。

### DMA 传输

基本DMA传输的一般流程如下：

1.主机通过声明LFRAME＃在LAD [3：0]上声明“ 0000b”来开始传输。

2.Host根据DMA传输方向声明DMA的“周期类型”。

3.主机声明频道号，如果适用，还声明终端数。

4.主机指示传输的大小：8位，16位或32位。

5，如果DMA读取：

•主机驱动数据的前8位并调转总线。

•外设使用有效的SYNC确认数据。

•如果进行16位传输，则对接下来的8位重复该过程。

•如果是32位传输，则每8位数据重复该过程两次。

6，如果是DMA写：

•主机扭转总线并等待数据。

•外设通过SYNC指示数据准备就绪，并传输第一个字节。

•如果进行16位传输，则外设指示数据就绪并传输下一个字节。

•如果是32位传输，则外设再重复SYNC +数据传输两次。

7.外围设备绕过总线。

上述DMA传输中使用的字段显示为：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| START | 1 | | **Start of Cycle:** ‘0000b’ for DMA.The start condition is the last clock ofLFRAME# active. | | | | |
| CYCTYPE +DIR | 1 | | **Cycle Type:** Bits [3:2] are ‘10b’ to indicate DMA.Bit 2 indicates the transfer direction: ‘0’ for DMA read (host to peripheral), and ‘1’ for DMA write (peripheral to host).Bit 0 is reserved and must be ignored by the peripheral. | | | | |
| SIZE | 1 | | **Size of Transfer:** Bits [3:2] are reserved and must be ignored by the peripheral. | | | | |
|  | | | LAD[1:0] | | Transfer size | | |
| **00** | | **8 bit** | | |
| **01** | | **16 bit** | | |
| **10** | | **Reserved** | | |
| **11** | | **32 bit** | | |
| 32 bit transfers are optional.If the host drives a size field of ‘10b’, the peripheral must ignore the transfer. | | | | |
| TAR | | 1 | Channel #: Used only for DMA cycles to indicate channel # being granted.The LAD[2:0] bits indicate the channel number being granted, and LAD[3] indicates the TC bit. See Section 6.4.1 for a description of terminal count.The encoding on LAD[2:0] for channel number is as follows: | | | | |
|  | | | Bits | Indication | Bits | | Indication |
| 000 | Channel 0 | 100 | | Channel 4 |
| 001 | Channel 1 | 101 | | Channel 5 |
| 010 | Channel 2 | 110 | | Channel 6 |
| 011 | Channel 3 | 111 | | Channel 7 |
| If a peripheral sees an encoding for a DMA channel that it did not request, or an encoding for channel 4, it must ignore the cycle | | | | |
| SYNC | | 1 - N | Sync: Allows peripheral or host to synchronize (add wait-states).Generally,the peripheral or host drives ‘0101b’ or ‘0110b’ until no more wait-states areneeded.At that point it will drive ‘0000b’.All other combinations arereserved | | | | |
|  | | | Bits | | | Indication | |
| 0000 | | | Sync achieved with no error.Also indicates no more transfers desired for that channel, and DMA request is de-asserted. | |
| 0101 | | | Part indicating wait states. | |
| 0110 | | | Part indicating wait states, and many wait states will be added. | |
| 1010 | | | Sync achieved with error.Also indicates no more transfers desired for that channel, and DMA request is de-asserted. | |
| 1001 | | | Sync achieved with no error and more DMA transfers desired to continue after this transfer. | |
| If the host sees a SYNC value that is not one of the above combinations, it is allowed to abort the transfer. | | | | |
| DATA | | 1, 2, 4 bytes | Data Phase: Data bytes transferred with the least significant nibble first (D[3:0] on LAD[3:0], then D[7:4] on LAD[3:0]).For transfers greater than 1 byte, the bytes are sent with least significant byte first. | | | | |

### Terminal Count

终端计数通过LAD [3]在与DMA通道在LAD [2：0]上进行通信的同一时钟上进行通信。 此字段是“频道”字段。 终端计数根据传输的大小指示传输的最后一个字节。

### 取消传输

传输结束消息通过外设发送的特殊SYNC字段传送到主机。 LPC设备禁止通过取消置位LDRQ＃来尝试表示传输结束。

DMA代理在要传输的每个数据字节上使用SYNC编码，它向主机指示这是传输的最后一个字节还是请求更多的字节。 为了指示传输的最后一个字节，外设使用SYNC值“ 0000b”（就绪）或“ 1010b”（错误）。 这些编码告诉主机这是在DMA读取（主机到外围设备）上传输的最后一段数据，或者后面的字节是在DMA写入（外围设备到主机）上传输的最后一段数据。

当主机看到这两种编码中的一种时，它将在该字节之后结束DMA传输，并将DMA请求置为无效。 因此，如果主机指示16位传输，则外设可以通过将SYNC值指示为“ 0000b”或“ 1010b”来结束一个字节的传输。 主机将不会尝试传输第二个字节，并将在内部取消声明DMA请求。 对于32位传输中的任何字节也是如此。 因此，这使外设可以终止DMA传输。

如果外设希望保持DMA请求处于活动状态，则它使用SYNC值“ 1001b”（就绪以及更多数据）。这告诉8237控制器，在传输了当前字节之后，将请求更多数据字节，因此主机会将DMA请求保持对8237控制器的激活状态。 因此，在8位传输大小上，如果外设向主机指示SYNC值为“ 1001b”，则数据将被传输，并且DMA请求将对8237控制器保持有效。 稍后，主机将返回另一个START –>CYCTYPE-> CHANNEL-> SIZE等组合，以发起另一次到外围设备的传输。

### SYNC field / LDRQ# Rules

由于LPC上的DMA传输是通过LDRQ＃断言消息请求的，并且在DMA传输过程中通过SYNC字段结束，因此外设在从DMA通道启动背对背传输时必须遵循以下规则。

# eSPI

# SMBus

# PECI

# ACPI

。