# 电脑键盘工作原理

随着IBM PC机的发展，键盘也分为XT, AT, PS/2键盘以至于后来的USB键盘. PC系列机使用的键盘有83键、84键、101键、102键和104键等多种。XT和AT机的标准键盘分别为83键和84键，而286机以上微机的键盘则 普遍使用101键、102键或104键。83键键盘是最早使用的一种PC机键盘，其键号与扫描码是一致的。这个扫描码被直接发送到主机箱并转换为 ASCII码；随着高档PC机的出现，键盘功能和按键数目得到了扩充，键盘排列也发生了变化，产生的扫描码与83键键盘的扫描码不同。为了保持PC系列微机的向上兼容性，需将84/101/102/104键键盘的扫描码转换为83键键盘的扫描码，一般将前者叫作行列位置扫描码，而将后者称为系统扫描码。显然，对于83键键盘，这两种扫描码是相同的。

键盘是由一组排列成矩阵方式的按键开关组成，通常有编码键盘和非编码键盘两种类型，IBM系列个人微型计算机的键盘属于非编码类型。微机键盘主要由单片机、译码器和键开关矩阵三大部分组成。其中单片机采用了INTEL8048单片微处理器控制，这是一个40引脚的芯片，内部集成了8位 CPU、1024×8位的ROM、64×8位的RAM、8位的定时器／计数器等器件。由于键盘排列成矩阵格式，被按键的识别和行列位置扫描码的产生，是由键盘内部的单片机通过译码器来实现的。单片机在周期性扫描行、列的同时，读回扫描信号线结果，判断是否有键按下，并计算按键的位置以获得扫描码。当有键按下时，键盘分两次将位置扫描码发送到键盘接口；按下一次，叫接通扫描码；释放时再发一次，叫断开扫描码。因此可以用硬件或软件的方法对键盘的行、列分别进行扫视，去查找按下的键，输出扫描位置码，通过查表转换为ASCII码返回。

键盘是与主机箱分开的一个独立装置，通过一根5芯电缆与主机箱连接，系统主板上的键盘接口按照键盘代码串行传送的应答约定，接受键盘发送来的扫描码；键盘在扫描过程中，7位计数器循环计数。当高5位(D6一D2)状态为全“0”时，经译码器在0列线上输出一个“0”,其余均为“1”；而计数器的低二位 (D1D0)通过4选1多路选择器控制0—3行的扫描。计数器计一个数则扫描一行，计4个数全部行线扫描一遍，同时由计数器内部向D2进位，使另一列线1 变低，行线再扫描一遍。只要没有键按下，多路选择器就一直输出高电平，则时钟一直使计数器循环计数，对键盘轮番扫描。当有一个键被按下时，若扫描到该键所 在的行和列时，多路选择器就会输出一个低电平，去封锁时钟门，使计数器停止计数。这时计数器输出的数据就是被按键的位置码(即扫描码)。8048利用程序读取这个键码后，在最高位添上一个“0”，组成一个字节的数据，然后从P22引脚以串行方式输出。在8048检测到键按下后，还要继续对键盘扫描检测，以发现该键是否释放。当检测到释放时，8048在刚才读出的7位位置码的前面(最高位)加上一个“1”，作为“释放扫描码”，也从 P22引脚串行送出去，以便和“按下扫描码”相区别。送出“释放扫描码”的目的是为识别组合键和上、下档键提供条件。

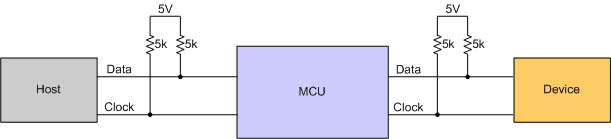
同时，主机还向键盘发送控制信号，主机CPU响应键盘中断请求时，通过外围接口芯片8255A一5的PA口读取键盘扫描码并进行相应转换处理和暂存；通过PB口的PB6和PB7来控制键盘接口工作。

从用途上看，键盘可分为台式机键盘、笔记本电脑键盘和工控机键盘三大类；其中台式机键盘从按键结构上又可分为两类，即机械键盘和电容键盘（又称有触点键盘和无触点键盘）。机械键盘存在着开关容易损坏、易污染、易老化的缺点，现已基本淘汰。电容键盘在可靠性上比前者有质的飞跃，使用寿命较长，目前大多为电容键盘。

早期的键盘是由美国IBM公司推出的，当时采用的计算机键盘为83键键盘。不久IBM又推出了84键的键盘设计标准，将键盘分为三个区，即功能区、打字键区及负责光标控制和编辑的副键盘区，这种键盘主要区域的划分标准一直沿用至今。随着微软Windows视窗操作系统的广泛应用，IBM公司于1986年首次推出了101键的标准键盘，除添加了F11、F12两个功能键之外，还在键盘的中部多加了一组专用的光标控制和编辑的键，使键盘功能得到了进一步扩充， 成为当时业界的标准键盘。后来，为与微软的Win95操作系统相配合，IBM又推出了104键键盘，新增了3个功能键（亦称Windows快捷键或热键 <HotKey>），使以前需要打开好几个窗口才能完成的某些功能，通过快捷键的设定直接启动菜单完成，而不必再点击鼠标，这样就使计算机的 操作更加便易。目前，这种104键的键盘（亦称Win95键盘）的设计已成为业界和市场上最为普遍、最为流行的一种标准，业界通称为标准键盘。实际上，所谓的标准键盘并没有标准，只因其应用较为广泛而被业界通称为标准键盘。由于软件的不断升级更新，为配合软件的需要，所谓的标准键盘的键数也在随之扩增，并因其又被业界所通常采纳而又被业界通称为标准键盘。

但是，无论是机械键盘或者电容键盘，还是从早期的83键键盘发展到目前通称的标准键盘104键键盘，其应用都是主要依赖于硬件CPU和软件操作系统。以下就键盘的构造及工作原理等问题进行介绍。

## (一)键盘构造及工作原理



PS/2设备履行一种双向同步串行协议。换句话说，每次数据线上发送一位数据并且每在时钟线上发一个脉冲就被读入。设备可以发送数据到主机，而主机也可以发送数据到设备，但主机总是在总线上有优先权，它可以在任何时候抑制来自设备的通信，只需把时钟线电平拉低即可。

　　键盘的内部结构主要包括控制电路板、按键、底板和面板等。电路板是整个键盘的控制核心，位于键盘的内部，主要担任按键扫描识别、编码和传输接口工作；它将各个键所表示的数字或字母转换成计算机可以识别的信号，是用户和计算机之间主要的沟通者之一。

键盘主要由键开关矩阵、单片机和译码器三大部分组成。键开关矩阵即键盘按键由一组排列成矩阵方式的按键开关组成，所输入的信号由按键所在的位置决定。单片机即键盘内部采用的Intel8048单片机微处理器，这是一个40引脚的芯片，内部集成了8位CPU、1024×8位的ROM、64×8位的RAM以 及8位的定时器／计数器等。译码器即信号编码转译装置，把键盘的字符信号通过编码翻译转换成相应的二进制码。由于键盘排列成矩阵格式，被按键的识别和行列位置扫描码的产生，是由键盘内部的单片机通过译码器来实现的。根据键盘向主机送入的二进制代码类型，可把键盘分为编码键盘和非编码键盘两种。IBMPC机的键盘属于非编码键盘，其特点是不直接提供所按键的编码信息，而是用较为简单的硬件和一套专用程序来识别所按键的位置，并提供与所按键相对应的中间代码，然后再把中间代码转换成要对应的编码。这样，非编码键盘就为系统软件在定义键盘的某些操作功能上提供了更大的灵活性。

计算机键盘通常采用行列扫描法来确定按下键所在的行列位置。所谓行列扫描法是指，把键盘按键排列成n行×m列的n\*m行列点阵，把行、列线分别连接到两个并行接口双向传送的连接线上，点阵上的键一旦被按动，该键所在的行列点阵信号就被认为已接通。按键所排列成的矩阵，需要用硬件或软件的方法轮转顺序地对其行、列分别进行扫描，以查询和确认是否有键按动。如有键按动，键盘就会向主机发送被按键所在的行列点阵的位置编码，称为键扫描码。单片机通过周期性扫描行、列线，读回扫描信号结果，判断是否有键按下，并计算按键的位置以获得扫描码。键被按下时，单片机分两次将位置扫描码发送到键盘接口：按下一次，叫接通扫描码；按完释放一次，叫断开扫描码。这样，通过硬件或软件的方法对键盘分别进行行、列扫视，就可以确定按下键所在位置，获得并输出扫描位置码，然后转换 为ASCII码，经过键盘I/O电路送入主机，并由显示器显示出来。

键盘要增加键数是很容易的，任何矩阵键盘通过增加键盘的行或列便可实现增加按键数。如64 键的键盘排列成8行×8列的行列点阵，128键的键盘排列成8行×16列的行列点阵，256键的键盘排列成16行×16列的行列点阵，这在1992年7月出版的《微处理机为基础之设计》出版物上已作介绍(此对比文献在2001年12月13日已递交)。因此，键盘结构采用矩阵式早已是公有技术。

## (二)键盘的按键增加及功能改进

随着计算机CPU（硬件）和Windows操作系统（软件）的不断升级换代，计算机的发展进入到了包括多媒体在内的高速多功能时代。与此相适应，计算机键盘也出现了一些新的变化，主要表现在键盘键数的增加及功能的整合。美国苹果(Apple)公司早在1992年就曾推出过设有调整音量按键的键盘。时下流行的多媒体键盘，也叫多功能键盘，其特征是增加了一些快捷键（或功能键），通过设定主板BIOS或软件设定，使这些新增的键可以实现开关电脑、休眠启动、 CD播放、音量调整、一键上网等特殊功能。此外还出现了一些多功能集成的键盘，如将扫描仪、录音设备或手写板、鼠标等整合为一体的键盘，以及带有文字编辑 或指纹识别功能的智能键盘等。

如前所述，IBM定义的101键键盘与微软(Microsoft)所定义的三键共同形成的104 键Win95键盘一般被称作标准键盘，那么标准键盘的104键之外如何增加按键扩充功能呢？从目前键盘发展情况看，可以有多种方法。单就增加键盘按键的设定技术看，一般来讲，按键的增加取决于三个方面的因素：一是CPU能识别，二是操作系统能响应，三是键盘内部电路结构能安排。从前面的介绍可知，CPU和 操作系统这两个因素是计算机必备的现成技术，不成问题。这样，增加键盘按键的关键就在于键盘的内部结构。以下两种方法可以直接简便地增加键盘按键。

目前PC机键盘按键一直沿用IBM的行列矩阵方式布局排列，称为扫描矩阵，其内部电路结构的核心是8048（包括其后续升级系列）单片机微处理器。根据单 片机微处理器特定的I/O（即Input/Output输入/输出）电路，IBM将键盘按键的行列矩阵定义为8行×16列＝128键，而现行的标准键盘只使用了其中的104键，故尚有128-104＝24键的阵点闲余，尚未定义。因此，增加按键的方法之一是，直接在这尚未定义的24个闲余按键阵点中继续定义新键以扩增新的功能键。方法之二是，通过改变单片机微处理器I/O电路的配线方式，可以把IBM的原8行×16列扫描矩阵变为（8+n）行×16列，如（8+1）行×16列＝144键等，这样就会在128键之外获得新增加的一行16个键位，再根据需要为其定义功能。以上两种方法都可以达到增加104键的标准键盘的按键数的目的。简而言之，第一种方法是在原扫描矩阵8行×16列的行列点阵中尚未利用的24个点上，根据需要可任意增加1至24个新键；第二种方法是在单片机微处理器原扫描矩阵8行×16列基础上再增加一行，变为9行×16列扫描矩阵，再根据需要任意增加新键，并不会带来单片机微处理器本身原有功能的更新。如IBM 的83键盘发展到原来的标准键盘101键，其要增加的键数只要通过增加键盘的行或列，并在增加的行列阵点上进行定义便可实现；而原来的101键的标准键盘 后发展到目前的104键的标准键盘，其增加的3个键数可以不增加键盘的行和列，只要在原扫描矩阵中尚未定义的点上直接定义就可以。无论是第一种方法还是第二种方法，都已经属于公有技术领域的技术。

## (三)什么是键盘的键位冲突？

简单的说，也就是当你同时按下键盘上的几个键的时候，这几个键不能同时反映出来，这就叫做键盘的键位冲突。例如说，你能想象当你按下Ctrl－Alt－Del时，系统只能接收到前两个键，而死活不承认你按了Del键吗？

当然，现实中是不会有这样的键盘的，按不下Ctrl－Alt－Del的键盘根本就没法出厂，但其他一些常见的键位冲突就不稀奇了。例如，经常有一些键盘不能同时对例如A-S-空格这样的按键组合作出反应，这样在FPS游戏中，使用者就会大为吃亏。

对于键位冲突问题，电脑用户中有两种完全不同的态度，绝大多数人根本意识不到键位冲突问题的存在，直到遇到冲突的时候才叫苦不迭，并且大呼“为什么他就能做这个动作？”；而另一部分人，特别是一些游戏高手，又在孜孜不倦的寻找“没有键位冲突的键盘”。这两种人都是对键位冲突的原因认识不清楚的。

键位冲突的直接起因，是键盘的非编码结构。在2月号的专题中，我们提到过现在的键盘几乎都是非编码的薄膜接触式键盘，那么什么是非编码键盘呢？在专题里我们没有详细的解释，下面我们就来详细说明一下非编码键盘的原理与结构。

传统的键盘，是编码式键盘，它的每个键按下时都会产生唯一的按键编码，并且通过专有的一组导线传输到键盘接口电路，由于其线路和编码的唯一性，这种键盘是不存在键位冲突的问题的，但是编码键盘结构复杂，现在已经很少使用了。

而现代的薄膜接触式键盘，任何一个按键都有上下两层薄膜的触点，我们将它拆开来仔细看一看，就会发现在任何一层薄膜上，导线数都远少于按键数，而 且每一条导线都同时连通多个按键的触点，而且，上层和下层的任何两条导线都最多只在一个按键上重合。也就是说，上层的1号导线可能会同时经过1、2、3、4、5……等按键，而下层的1号导线可能同时经过1、Q、A、Z……等按键，且两条导线只在1键上重合。

这样，根据上层薄膜和下层薄膜所经过的按键，就可以排出一个类似下面的表格：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 上层导线1 | 上层导线2 | 上层导线3 | 上层导线4 | 上层导线5 | … |
| 下层导线1 | 1 | 2 | 3 | 4 | 5 | … |
| 下层导线2 | Q | W | E | R | T | … |
| 下层导线3 | A | S | D | F | G | … |
| 下层导线4 | Z | X | C | V | B | … |
| … | … | … | … | … | … | … |

在非编码键盘的接口控制电路中，就存储着这样一张表格，当按下某个按键，例如“Q”时，那么在这一点上，上下两个触点就会连通，反映到接口电路中，就会检测到上层导线1与下层导线2被连通了。相对应在表中一查，就会知道，被按下的是字母“Q”，然后通过接口输出其ASCII码。

与老式的编码式键盘相比，非编码键盘的结构要简单许多，省略了复杂的编码电路和蜘蛛网似的走线，而且更重要的是，由于非编码式键盘将按键结构和输出键码分离，所以当需要制造不同键位排列的键盘时，不需要重新设计键盘线路，而只要将控制电路中的键位排列表格重新刷新就可以了。例如在德文键盘上，子母“Y”和 “Z”的位置是互调的，那么如果要把一条英文键盘生产线改造成德文键盘生产线，什么都不用改变，只须在控制芯片中刷进一个新的排列表，将“上层导线1－下层导线4”的位置写作“Y”，“Z”字也作同样处理就行了，显然比重新布线要简单了很多。这也就是为什么非编码式键盘大行其道的原因。

但是非编码式键盘带来的就是“键位冲突”的问题。以上面的按键排列表为例，当按下一个按键时，键盘肯定会正常识别的；当按下两个按键时，例如同时按下 “Q”与“D”，此时上层导线1与下层导线2连通，而上层导线3与下层导线3连通，系统完全可以正常识别；或者同时按下“Q”与“E”，此时，上层导线 1、下层导线2、上层导线3同时连通，系统同样可以正常识别出是按下了这三个按键。

但在同时按下3个或3个以上按键时，情况就复杂了。

在一种情况下，例如同时按下“Q”、“D”、“B”时，此时，从表中我们可以看出，上层导线1－下层导线2、上层导线3－下层导线3、上层导线5－下层导线4都分别连通，此时系统完全可以正常识别出按下了这三个按键。

但当按下的第三个按键不是“B”，而是“E”呢？

此时，我们可以发现，由于第三组连通的导线变成了上层导线3－下层导线2，所以，此时反映到接口控制芯片的信号是：上层导线1－下层导线2－上层导线3－下层导线3统统被连通到了一起。

但是，问题就在于，除了按下的第三个按键是“E”以外，如果按下的第三个键是用来连通上层导线1和下层导线3的“A”，也会有同样的信号组合？事实上， “Q”、“E”、“A”、“B”四个按键中按下任何三个都会有同样的信号组合，那么此时接口电路怎么判断呢？究竟是按下了哪三个按键或者是同时把四个按键都按下了呢？

这个时候，大多数键盘选择的方式，就是按照一个固定的规则输出固定的按键组合代码，对于其他可能的组合方式统统予以忽略，换句话说，在上面的键盘中，可能无论你怎么按这四个按键，它可能永远只会输出“Q”－“E”－“D”的组合，尽管可能你按下的实际是“Q”－“A”－“D”（这里只是一个假设，实际上为了避免误码，大多数键盘根本只输出两个按键而已）。于是，“键位冲突”就出现了。

由于非编码键盘的固有结构，“键位冲突”是不可避免的。于是就需要键盘的设计者绞尽脑汁修改键盘的线路排列和待查表格，尽量使可能会有的常用组合按键避开冲突键位——尽管要做到所有的按键都不发生冲突是不可能的，但你总不能让Ctrl－Alt－Del相互冲突吧？那这样的键盘还怎么用？还有如果四个方向键 相互冲突，那赛车类的游戏还怎么玩？

不过，由于市场上的游戏实在太多，每个游戏都可能会有自己常用的按键组合，甚至每个人都有自己的习惯按键，键盘的设计者当然不可能面面俱到，有经验的设计师可能会考虑得多一些，至少对最常见的游戏和最常见的按键设置能够保证不出现问题，但要考虑到所有情况当然是不可能的。

所以，当你购买键盘的时候，绝对不要忽略“键位冲突”的问题，一定要把键盘装到电脑上，试一试自己常玩的游戏是否能够正常使用（只有实际试用才能试出键位冲突问题，由于它与键位表相关，所以看外表是看不出来的）。同时，一些游戏高手朋友也不要白费心机去寻找什么“没有键位冲突的键盘”了，只要它依然是非编码键盘，那么键位冲突就是不可避免的，只要它的键位冲突不对你所玩的游戏构成影响，这就足够了。

## (四)PS/2协议分析

PS/2鼠标和键盘执行一个双向同步串行协议。当两条线都是高电平时(集电极开路)，总线空闲。只有在这种状态下，才允许键盘/鼠标开始传输数据。主机对总线有最高的控制权，在任何时候通过将时钟线拉低就可以禁止通信。时钟信号总是由设备端（鼠标或者键盘）生成的。如果主机想发送数据，它必须先将时钟拉低来禁止来自设备端的通信。然后主机再拉低数据线，释放时钟。 注释:释放时钟，就是再恢复时钟为高。这就是"请求发送(Request-to-Send)"状态，提示设备端开始生成时钟信号（并接收主机发来的数据·）。

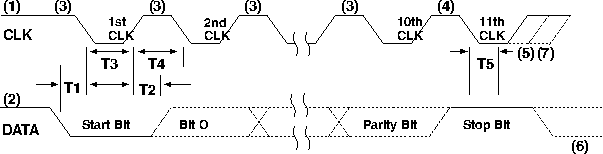
总线状态

|  |  |  |
| --- | --- | --- |
|  | 数据线0 | 数据线1 |
| 时钟线0 | 禁止通信 | 禁止通信 |
| 时钟线1 | 主机要求发送数据 | 总线空闲 |

时钟和数据引脚是双向集电极开路的信号，可以被键盘内部的上拉电阻拉高到5V，从设备发送给主机的数据时在时钟信号的下降沿读取的，从主机发给设备的数据是在上升沿读取的。时钟频率必须在10-16.7KHz之间。这意味着时钟必须是 高电平持续30～50毫秒，低电平持续30～50毫秒。如果你设计一个键盘鼠标或者主机模拟器，你必须在每个单元的中间时刻 （也就是,在时钟跳变之后的15～25毫秒后）修改/取样数据线.重复一遍，键盘/鼠标总是生成时钟信号，而主机控制着整个通信过程。时序是非常重要的。在本文中给出的时间数必须严格遵循。

## (五)设备发送数据到主机

数据和时钟线都是集电极开路的。在+5V和每根线之间连接着一个电阻，所以总线的空闲状态是高电平。当键盘或者鼠标想发送数据时，它首先必须检查时钟线 ，确认它处于高电平。如果不是，主机禁止通信，设备必须缓冲任何要发送的数据，直到主机释放时钟。在设备开始传输数据之前，时钟线必须持续为高电平的时间必须在50ms以上。当时钟为高电平时，键盘/鼠标写一个bit到数据线上；当时钟为低电平时，主机从数据线上读取这个bit 。当时钟位高时，数据线改变状态；当时钟位低时，（数据线上的)数据是有效的。时钟频率是10-16.7KHz，从时钟脉冲的上升沿到数据跳变的时间必须至少5ms，从数据跳变到时钟脉冲的下降沿必须至少5ms,且不超过25ms，主机可在任何时间禁止通信，只需要将时钟线下拉位低电平超过100ms即可，如果在第11个脉冲时禁止传输，设备必须中止当前的传输，准备在主机释放时钟后重新传输当前的数据"块"。一个数据块可能是通码，断码，设备ID，鼠标移动包等等。举个例子，如果当发送一个两字节断码的第2个字节时，键盘被中断，它将需要重新发送此断开的两个字节，而不仅仅时被中断掉的那个字节。如果在第一个高->低时钟跳变时，（或者在最后一个时钟脉冲的下降沿之后）主机将时钟拉低，键盘/鼠标不必重新传输任何数据。但是，如果新产生的数据需要传输，它必须将数据缓冲，直到主机释放时钟。键盘有一个16字节的缓冲区。如果有超过16个字节的按键数据需要缓存，16字节之后的数据将被忽略，直到缓冲区有空余位置。鼠标只能缓冲最近的一个要传输的移动数据包。



|  |  |
| --- | --- |
| 时间参数 | 最大值/最小值 |
| T1数据跳变到时钟的下降沿 | 5/25 us |
| T2时钟的上升沿 到 数据跳变 | 5/T4 - 5 us |
| T3时钟inactive | 30-50 us |
| T4时钟active | 30-50 us |
| T5 | >0/50 us |

时间到外设（键盘等）禁止时钟11后，以确保外设不启动另一个传输。

外设(指键盘) 检查时钟线 ，如果时低电平，禁止发送数据。

主机检查数据线，如果是低电平，那么控制器从系统接收数据

外设在传输过程中检查时钟线间隔不超100us。如果外设发现主机将电平拉低，就终止传输。在传输过程的前10个时钟周期的任何时候，主机都可以中止传输。

终止传输的最后一个检查 要持续至少5us。

主机拉低时钟线，将禁止下一次传输

主机如果有一个字节要传输给设备，可以拉低数据线。当开始位（总是0）放置到数据线时，数据线被拉低。

主机拉高时钟将允许下一次传输。

## (六)主机发送数据到设备

首先，PS/2设备总是产生时钟信号

如果主机想发送数据，它必须先将时钟线和数据线设置成"Request-to-send"状态:

1)拉低时钟线至少100us来禁止通信

2) 通过拉低数据线来请求 "Request-to-send", 然后释放时钟 .

设备应该在不超过10ms(注意，是毫秒)的间隔内就要检查一次这个状态。

当设备检测到这个状态，它将开始产生时钟信号，时钟在8个数据位和一个停止位。

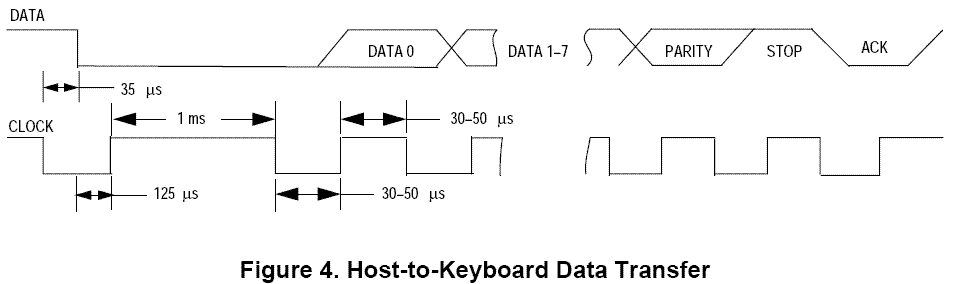
只有当时钟线为低的时候，主机才可以改变数据线（也就是将数据写入到数据线）。数据将在时钟为高电平的时候被设备读取。

在收到停止位之后，设备将通过拉低数据线，生成最后一个时钟脉冲来应答收到的字节

在第11个时钟脉冲之后，如果主机并没有释放数据线，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）。

在第11个脉冲之前（回应位），主机可以随时中止传输，只要拉低时钟持续100us即可

|  |  |
| --- | --- |
| 时间参数 | 最大值/最小值 |
| T7 时钟低电平 | 30-50 us |
| T8 时钟高电平 | 30-50 us |
| T9 | 30-50 us |



FreeScale(Moto)提供的时序图

## (七)主机与键盘的数据传输总结

主机启动通信：

1.主机通过拉低时钟线来启动一个主机到键盘的数据传输。大约35us后，主机拉低数据线。这个事件序列就是通知键盘，主机要传输一个命令了。在数据信号的下降沿大约 125us 后释放时钟线，就被键盘内部的上拉电阻拉为高电平 。

2. 在时钟信号的上升沿约1ms ，数据传输开始（这个期间数据线保持低电平）。键盘拉低时钟线. 这个就做传输的起始位。

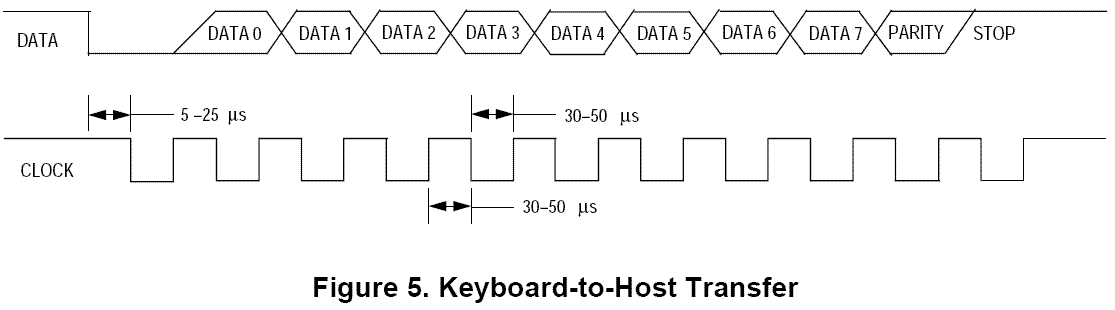
3. 键盘产生时钟，让主机在时钟的驱动下将8个数据位传输到键盘。 这个时钟有50％的占空比，高电平和低电平的时间都在30~50us之间。主机在每个周期的低电平的时候改变数据线上的数据。在每个时钟的上升沿5～25us之后，键盘开始从数据线上取样，取出主机发来的数据。

4.8个数据位之后是一个奇偶校验位（采用奇校验）。

5.最后是停止位。

6.如果键盘读到一个高电平的停止位，键盘就在取样停止位的那个时钟的下降沿之后的低电平期间拉低数据线。这个行为叫做键盘给主机的应答信号。键盘拉高时钟之后，也拉高数据线。

7.在收到一个字节后，键盘对接收的数据执行一个奇偶校验。如果奇偶校验错误，或者接收的数据无法识别为有效的命令，那么键盘请求重新发送（发送0xFE给主机)。



键盘发送数据：

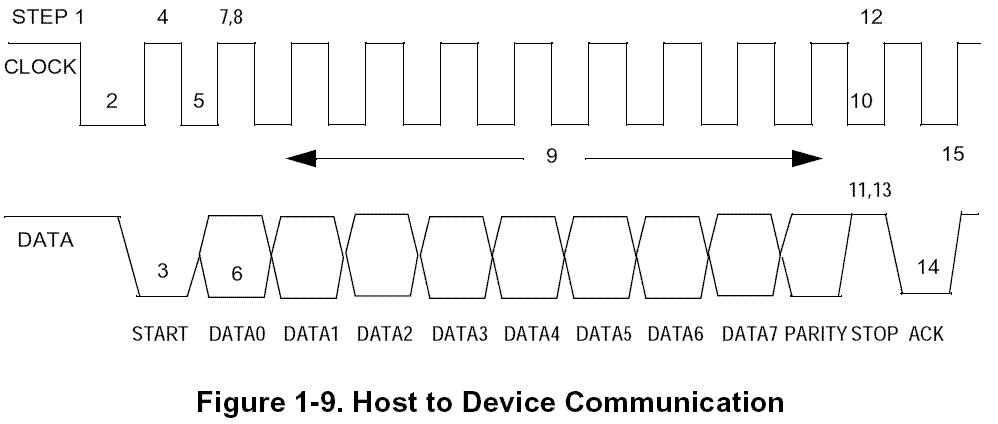
1. 键盘发送数据前，首先必须让时钟和数据线都在键盘内部的上拉电阻作用下拉成高电平。然后键盘拉低数据线，5－25us之后，键盘拉低时钟线。在时钟的下降沿，开始传输起始位。

2. 键盘产生时钟，在时钟的驱动下键盘将 8个数据位传输给主机。这个时钟有50％的占空比。高电平和低电平的时间都在30~50us之间。键盘在每个周期的高电平的时候改变数据线上的数据。这个改变出现在时钟的上升沿的5us后和下降沿的5us前。键盘的数据在时钟的下降沿被锁存进主机。

3.数据位之后是一个奇校验位。

4.最后是停止位。在时钟的下降沿0～50us之间主机拉低时钟来锁存停止位。这就告诉键盘主机现在忙，不能在接收其他的键盘传输数据。在处理完接收的数据，准备好接收下一个数据之前，主机释放时钟。

6.在键盘传输数据到主机的过程中，主机任何时候都可以中断这个传输，发送一个命令到键盘。任何时间，主机拉低时钟和数据线即可中止传输。所以，键盘在时钟为低电平的时候必须对数据线进行采样。在产生时钟的上升沿时，必须读取时钟。在任何时候发现这两个条件不满足，就拉高时钟和数据线，放弃这次传输，并准备从主机接收命令。



Moto 香港公司的 Derek Lau

主机读取完数据并发送：

当时钟高电平时，键盘发送给主机的数据被主机读取。在空闲状态，时钟和数据线都被拉为高电平。主机拉低时钟最少100us开始发送数据给键盘：

1.主机等待，处理完所有外部时钟，直到进入空闲状态

2.主机拉低时钟

3.主机拉低数据作为开始位

4 .主机释放时钟

5.设备拉低时钟

6.主机发送数据

7.设备释放时钟（即时钟线被拉高），读取数据

8.设备检查时钟线，如果时钟为低（主机将时钟拉低了），中止传输

9.重复5－7步，发送DATA1－DATA7，以及奇校验位

10.设备拉低时钟

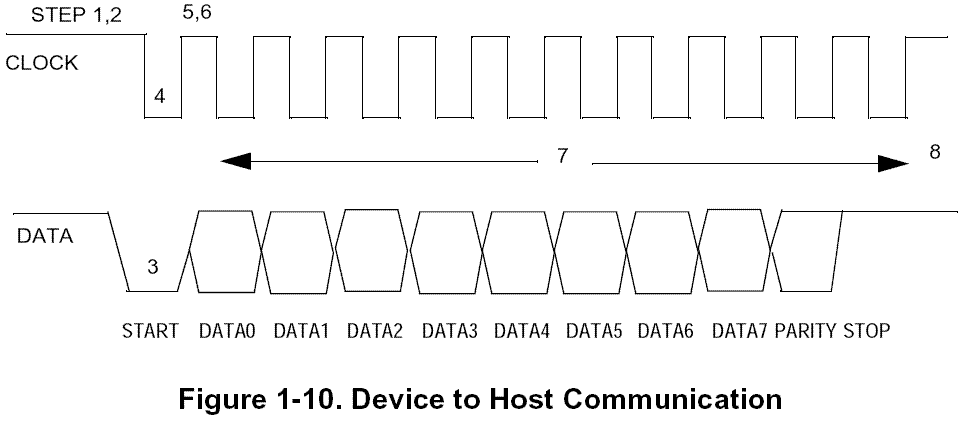
11.主机释放数据线

12.设备时钟

13.设备读取数据线，想得到停止位，如果数据线为低（不是停止位），产生错误

14.设备拉低时钟和数据线

15.设备释放时钟和数据线



设备等待主机读完数据继续发送：

从设备发送到主机的数据，在时钟的下降沿被读取。在发送数据前，设备检查通过检测时钟信号为高来得知主机是否贮备好：

1.键盘（设备）等待时钟为高（约50us）

2.如果数据为低，终止

3.键盘发送数据

4.键盘拉低时钟为低

5.键盘释放时钟为高

6.键盘读取时钟线，如果时钟为低，中止传输

7.重复 4－6步，传输DATE0－DATE7，奇校验位和停止位

8.键盘释放时钟为高，结束传输