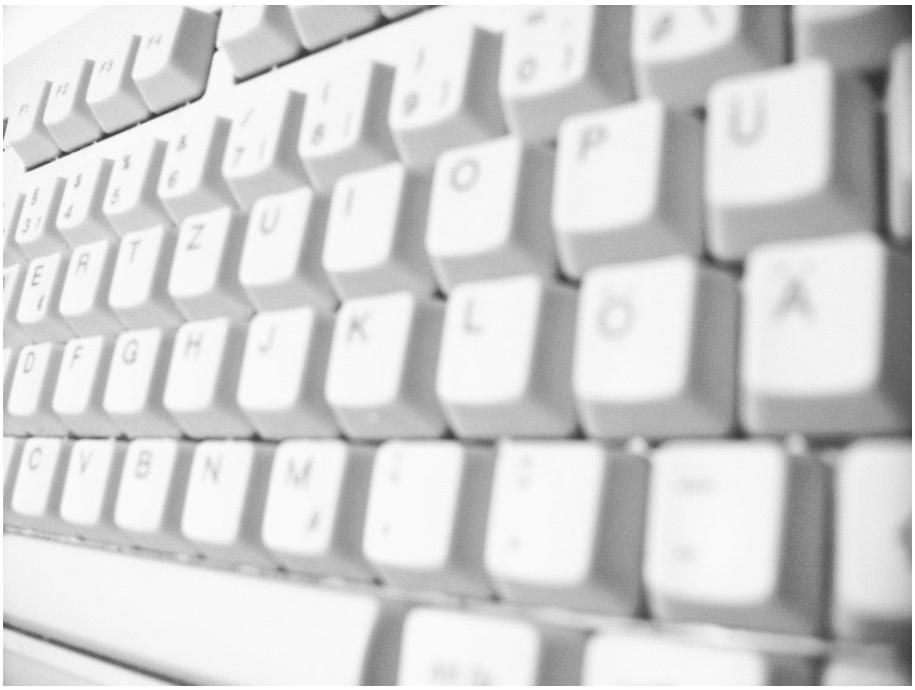
Die PS/2 Tastaturschnittstelle

Da es im deutschsprachigen Internet wenig bis keine Informationen über die Funktionsweise und Ansteuerung einer PC-Tastatur gibt, wurde mit diesem Dokument der hervoragende Artikel von Adam Chapweske ins Deutsche übersetzt.



Deutsche Übersetzung: Bernward Mock
Satz in LaTeX: Stefan Schöttke

Die deutsche Version stammt von der Seite: `http://www.marjorie.de/`
Dieser Artikel ist im Original zu finden auf: `http://www.computer-engineering.org/`

Inhaltsverzeichnis

1 Die PS/2 Tastatur-Schnittstelle

1.1 Einleitung

Dieser Artikel versucht alle Fragen rund um AT und PS/2 Tastaturen zu beleuchten. Er enthält Informationen über die Low-Level-Signale, das Protokoll, die Scancodes, den Befehlssatz, die Initialisierung, Kompatibilitätsfragen und weitere Punkte. Da es vom Thema her eng verwandt ist, habe ich auch Informationen über den PC Tastaturcontroller beigefügt. Alle Programmbeispiele zum Tastaturencoder sind in Microchip PIC Assembler. Alle Programmbeispiele zum Tastaturcontroller sind in x86 Assembler.

1.2 Ein Rückblick in die Geschichte

Die gebräuchlichsten Tastaturen sind heute:

- die USB-Tastatur - die aktuellste Tastatur, die von nahezu allen neuen Computern (Macintosh und IBM/kompatibel) unterstützt werden. Diese Schnittstelle ist relativ kompliziert und wird hier nicht behandelt.
- die IBM/kompatible Tastatur - auch bekannt als „AT-Tastatur“ oder „PS/2-Tastatur“, alle modernen PCs unterstützen diese Tastatur. Sie haben die einfachste Schnittstelle und sind Thema dieses Artikels.
- die ADB Tastatur - für den Apple Desktop Bus älterer Macintosh-Systeme. Sie wird hier nicht behandelt.

IBM brachte mit jeder neuen Computergeneration auch eine neue Tastatur heraus. Der Original PC und später der IBM XT benutzten die sogenannte „XT-Tastatur“. Diese sind nicht mehr gebräuchlich und unterscheiden sich gravierend von modernen Tastaturen; sie werden hier nicht behandelt. Danach kam das IBM AT System und später der IBM PS/2. Mit ihnen kamen auch die heute gebräuchlichen Tastaturen, um die es in diesem Artikel geht. AT Tastaturen und PS/2 Tastaturen sind sehr ähnlich; die PS/2 Tastatur hat jedoch einen kleineren Stecker und unterstützt eine Reihe von Zusatzfunktionen. Trotzdem blieb sie abwärtskompatibel mit dem AT-System, und nur wenige der zusätzlichen Funktionen wurde jemals genutzt (da auch die Software immer abwärtskompatibel bleiben wollte). Hier ist eine Zusammenstellung der drei wichtigsten IBM-Tastaturmodelle:

IBM PC/XT Keyboard (1981):

- 83 Tasten
- 5-poliger DIN Stecker
- einfaches serielles, unidirektionales Protokoll
- benutzt den als Scancode-Set 1 bekannten Satz
- keine Host-Tastaturbefehle

IBM AT Keyboard (1984) - Nicht abwärtskompatibel mit XT Systemen [1].

[1] XT Tastauren haben ein völlig anderes Protokoll als bei AT und PS/2 Systemen üblich, was sie für neuere Computer unbrauchbar macht. Es gab eine Übergangszeit, in der einige Keyboardcontroller mit beiden Tastaturen arbeiten konnten (umschaltbar durch Dipschalter, Jumper oder automatische Erkennung). Ebenso gab es Tastaturen, die mit beiden Systemen arbeiten konnten (ebenso manuell oder automatisch umschaltbar). Wenn Sie so einen PC oder Tastatur besitzen, lassen Sie sich nicht verwirren; XT-Tastaturen sind NICHT kompatibel mit modernen PCs.

- 84 -101 Tasten
- 5-poliger DIN Stecker
- bi-direktionales serielles Protokoll
- benutzt den als Scancode-Set 2 bekannten Satz
- acht Host-Befehle

IBM PS/2 Keyboard (1987) - Kompatibel mit AT Systemen, nicht kompatibel mit XT systemen.

- 84 - 101 Tasten
- 6-poliger Mini-DIN Stecker
- bi-direktionales serielles Protokoll
- bietet zusätzlich den Scancode-Set 3 an
- 17 Host-Befehle

Die PS/2-Tastatur war ursprünglich eine Erweiterung der AT-Tastatur. Sie besaß einige zusätzliche Host-Befehle und einen kleineren Stecker. Das waren die einzigen Unterschiede zwischen diesen beiden Modellen. Die Computerindustrie kümmert sich jedoch lieber um Kompatibilität als um die Einhaltung von Standards; daher wird jede heute gekaufte Tastatur kompatibel zu PS/2- und AT-Systemen sein, wenn sie auch nicht alle Merkmale der Originalmodelle vollständig unterstützen.

Heute meint man mit „AT-Tastatur“ und „PS/2-Tastatur“ nur die Steckergröße. Welche Einstellungen und Befehle jede einzelne Tastatur unterstützt oder nicht unterstützt, kann man nur vermuten. Die Tastatur, die ich z.B. gerade jetzt benutze, hat einen PS/2-Stecker, unterstützt aber nur sieben Befehle vollständig, zwei teilweise und quittiert mehr oder weniger die restlichen. Zum Vergleich, meine "Testtastatur"hat einen AT-Stecker, aber unterstützt jede Funktion der Original PS/2 Tastatur (und einige zusätzliche). Moderne Tastaturen müssen als "kompatibel", nicht als SStandard"betrachtet werden. Wenn Sie ein Gerät entwickeln, das sich auf exotische Funktionen verlässt, wird es in der einen Konfiguration funktionieren, in der anderen jedoch nicht...

Moderne PS/2 (AT) kompatible Tastaturen

- beliebige Anzahl von Tasten (üblich sind 101 oder 104)
- 5-poliger oder 6-poliger Stecker; Adapter wird gewöhnlich mitgeliefert
- bi-direktionales serielles Protokoll
- Nur Scancode-Set 2 garantiert
- quittiert alle Kommandos, reagiert aber nicht auf alle

1.3 Allgemeine Beschreibung

Tastaturen bestehen aus einer Matrix von Tasten, die durch den eingbauten Mikroprozessor, den sogenannten Keyboard-Encoder überwacht werden. Die verwendeten Prozessoren [2] unterscheiden sich je nach Tastatur, aber alle tun in etwa das gleiche. Sie registrieren, welche Taste(n) gedrückt oder losgelassen wurden und senden die entsprechenden Daten zum Host. Sie sorgen für die Tastenentprellung und speichern gegebenenfalls die Daten in ihrem 16-Byte Buffer. Ihr

[2] Ursprünglich benutzte IBM den Intel 8048 als Tastatur-Encoder. Inzwischen gibt es zahlreiche Tastatur-Encoder von verschiedenen Herstellern.

Computer-Mainboard hat einen Tastatur-Controller [3] der alle Daten von der Tastatur dekodiert und an die Software weitergibt. Der Datenaustausch zwischen Tastatur und Host verwendet ein IBM-Protokoll.

1.4 Elektrisches Interface und Protokoll

AT- und PS/2-Tastaturen benutzen das selbe Protokoll wie PS/2-Mäuse. Weitere Informationen über dieses Protokoll erhalten sie hier.

1.5 Scancodes

Der Prozessor in Ihrer Tastatur verwendet die meiste Zeit damit, die Tasten zu beobachten, zu „scannen“. Wenn er feststellt, dass eine Taste gedrückt, losgelassen oder festgehalten wird, sendet er ein Datenpaket, das als „Scancode“ bezeichnet wird, an Ihren Computer. Es gibt zwei verschiedene Scancode-Typen, die „Makecodes“ und die „Breakcodes“. Ein Makecode wird gesendet, wenn eine Taste heruntergedrückt oder festgehalten wird; ein Breakcode wird gesendet, wenn eine Taste losgelassen wird. Jede Taste besitzt einen eigenen unverwechselbaren Makecode und Breakcode, so dass der Host exakt bestimmen kann, was mit jeder einzelnen Taste passiert ist. Die Zuordnungen von Breakcodes und Makecodes zu jeder einzelnen Taste bilden ein sogenanntes SScancode-Set”; es gibt 3 Standard-Scancode-Sets, die als Set 1, 2 und 3 bezeichnet werden. Alle modernen Tastaturen verwenden defaultmäßig das Scancode-Set 2. [4]

Wie findet man nun den Scancode für jede Taste? Leider gibt es keine einheitliche Formel dafür. Die Makecodes und Breakcodes jeder Taste müssen in einer Tabelle nachgeschlagen werden. Ich habe Tabellen für alle 3 Scancode-Sets zusammengestellt:

- Scancode-Set 1 - Original XT Scancode-Set; wird durch einige moderne Tastaturen unterstützt
- Scancode-Set 2 - Standard Scancode-Set für alle modernen Tastaturen
- Scancode-Set 3 - Optionales PS/2 Scancode-Set, selten benutzt

1.6 Makecodes, Breakcodes, und Tastenwiederholfunktion Typematic Repeat

Bei jedem Tastendruck wird der Makecode dieser Taste zum Computer gesendet. Beachten Sie, dass der MakeC code nur eine Taste auf der Tastatur repräsentiert, und nicht das Zeichen, das auf der Taste aufgedruckt ist. Das heißt, es gibt keinen Zusammenhang zwischen einem Scancode und einem ASCII-Code. Es ist Aufgabe des Computers, die Scancodes in Buchstaben oder Befehle umzusetzen.

Obwohl die die meisten Makecodes aus einem Byte bestehen, gibt es auch einige „erweiterte“ Tasten, deren Makecode zwei oder vier Byte lang sind. Diese Makecodes erkennt man daran, dass ihr erstes Byte 0xe0 ist.

[3] Ursprünglich benutzte IBM den Intel 8042 als Tastatur-Controller. Inzwischen sind diese Controller in die Chipsätze moderner Mainboards integriert.

[4] Ursprünglich verwendete die AT-Tastatur nur Set 1; die PS/2-Tastatur verwendet standardmäßig Set 2, unterstützt aber alle 3 Sets. Die meisten modernen Tastaturen verhalten sich wie die PS/2-Tastatur, es gibt aber einige, die Set 1 oder Set 3 oder beide nicht unterstützen. Wenn Sie sich einmal mit Low-Level PC-Programmierung beschäftigt haben, werden Sie sicher festgestellt haben, dass der Tastatur-Controller standardmäßig die Scancodes aus Set 1 liefert. Das liegt daran, dass der Tastatur-Controller alle ankommenden Scancodes zu Set 1 konvertiert (aus Gründen der Kompatibilität zu Software, die für das XT-System geschrieben wurde). Nach wie vor werden aber Scancodes aus Set 2 über das Tastaturkabel übertragen.

So wie die Tatatur beim Herunterdrücken einer Taste der Makecode sendet, wird beim Loslassen der Taste der Breakcode gesendet [5]. Zum Glück müssen Sie die Breakcodes jedoch nicht in der Tabelle nachschlagen, da es gewisse Beziehengen zwischen Makecodes und Breakcodes gibt. Die meisten Breakcodes im Set 2 sind zwei Byte lang, wobei das erste Byte 0xf0 ist, und das zweite Byte der Makecode der Taste. Die Breakcodes der erweiterten Tasten sind gewöhnlich drei Bytes lang, wobei die ersten beiden Bytes 0xe0, 0xf0 sind. Das dritte Byte ist das letzte Byte des Makecodes. Als Beispiel habe ich hier die Makecodes und Breakcodes einiger Tasten für das Scancode-Set 2 aufgeführt:

Taste	Makecode (Set 2)	Breakcode (Set 2)
„A“	0x1c	0xf0, 0x1c
„5“	0x2e	0xf0, 0x2e
„F10“	0x09	0xf0, 0x09
rechte Pfeiltaste	0xe0, 0x74	0xe0, 0xf0, 0x74
rechte „Ctrl“-Taste	0xe0, 0x14	0xe0, 0xf0, 0x14

Beispiel: Welche Folge von Makecodes und Breakcodes müssen an Ihren Computer gesendet werden, damit der Buchstabe „G“ in einem Textverarbeitungsprogramm erscheint? Da dies ein Großbuchstabe ist, müssen folgende Vorgänge ablaufen: Drücken der Shift-Taste, drücken der G-Taste, loslassen der G-Taste, loslassen der Shift-Taste. Die Scancodes dieser Vorgänge sind: Makecode für die Shift-Taste (0x12), Makecode für die G-Taste (0x34), Breakcode für die G-Taste (0xf0, 0x34), Breakcode für die Shift-Taste (0xf0, 0x12). Deshalb werden folgende Daten an Ihren Computer gesendet: 0x12, 0x34, 0xf0, 0x34, 0xf0, 0x12.

Wenn Sie eine Taste drücken, wird der Makecode zum Computer gesendet. Wenn Sie die Taste festhalten, setzt die Wiederholfunktion ein („Typematic“), d.h. es wird fortlaufend der Makecode der Taste gesendet, solange bis die Taste losgelassen oder eine andere Taste gedrückt wird. Um das zu prüfen, starten Sie den Texteditor und halten Sie die Taste „A“ gedrückt. Es erscheint sofort der Buchstabe „a“ auf dem Bildschirm. Nach einer kurzen Verzögerung folgen weitere „a“s bis Sie die Taste loslassen. Es gibt zwei wichtige Parameter: Die Verzögerung (typematic delay) zwischen dem ersten und dem zweiten „a“, und die Wiederholrate (typematic rate), die angibt, wie viele Zeichen pro Sekunde gesendet werden. Die Verzögerung kann zwischen 0,25 und 1,00 Sekunden liegen, und die Wiederholrate zwischen 2,0 cps (Zeichen pro Sekunde) und 30,0 cps. Diese beiden Parameter können mit dem Befehl 0xf3 („Set Typematic Rate/Delay“) eingestellt werden.

Wiederholte Daten werden nicht in der Tastatur zwischengespeichert. Wenn mehrere Tasten festgehalten werden, so setzt die Wiederholfunkton nur für die Taste ein, die als letztes heruntergedrückt wurde. Die Wiederholfunktion endet, wenn diese Taste losgelassen wird, selbst wenn noch weitere Tasten gedrückt sein sollten.

1.7 Reset

Nach einem Einschalt-Reset oder einem Software-Reset (siehe „Reset“-Befehl) führt die Tastatur einen Selbsttest, genannt BAT (Basic Assurance Test) durch und lädt folgende Grundeinstellung:

Verzögerung 500 ms,

Wiederholrate 10,9 cps,

Scancode Set 2

[5] Tatsächlich hat die „Pause/Untbr“-Taste keinen Breakcode im Scancode-Set 1 und 2. Wird diese Taste gedrückt, wird der Makecode gesendet; wenn sie losgelassen wird, passiert nichts. Es ist also nicht möglich zu bestimmen, wann diese Taste losgelassen wird.

Alle Tasten Typematic/Make/Break.

Zu Beginn des BAT schaltet die Tastatur ihre drei LEDs ein, und am Ende wieder aus. Dann wird BAT-Ende-Code gesendet, entweder 0xaa (BAT erfolgreich) oder 0xfc (Fehler). Der BAT-Ende-Code muss 500 bis 750 ms nach dem Einschalten der Versorgungsspannung gesendet werden.

Viele der Tastaturen, die ich getestet habe, ignorieren die Clock- und Data-Leitung bis nach dem Senden des BAT-Ende-Codes. Damit kann ein Inhibit-Signal (Clock Low) die Tastatur nicht am Senden des BAT-Ende-Codes hindern.

1.8 Befehlssatz

Einige Anmerkungen zu den Befehlen, die der Host zur Tastatur senden kann:

- Die Tastatur löscht ihren Ausgabepuffer bei jedem empfangenen Befehl.
- Wenn die Tastatur einen ungültigen Befehl oder Parameter empfängt, muss sie mit "resend"(0xfe) antworten.
- Die Tastatur darf keine Scancodes senden, während sie einen Befehl verarbeitet.
- Wenn die Tastatur auf ein Parameter-Byte wartet und statt dessen einen Befehl empfängt, sollte sie den letzten Befehl verwerfen und den neuen Auswerten.

Es folgen alle Befehle, die der Host zur Tastatur senden kann:

- 0xff (Reset) - Die Tastatur antwortet mit „Acknowledge“ (0xfa) und führt dann den Reset aus (siehe Abschnitt Reset)
- 0xfe (Resend) - Die Tastatur wiederholt das letzte gesendete Byte. Ausnahme: Wenn das letze gesendete Byte „resend“ (0xfe) war, wiederholt die Tastatur das letzte Byte, das nicht 0xfe war. Mit diesem Befehl zeigt der Host an, dass ein Empfangsfehler aufgetreten ist.

Die nächsten sechs Befehle können in jedem Modus an die Tastatur gesendet werden. Sie wirken sich jedoch nur im Modus 3 aus, d.h. wenn die Tastatur auf Scancode-Set 3 gestellt ist.

- * 0xfd (Set Key Type Make) - Unterbindet Breakcodes und Wiederholfunktion für spezifizierte Tasten. Die Tastatur antwortet mit „Acknowledge“ (0xfa) und stoppt die Tastenabfrage (sofern aktiv) und liest eine Liste von Tasten vom Host ein. Diese Tasten werden durch ihren Makecode im Scancode-Set 3 spezifiziert. Die Tastatur antwortet auf jeden Makecode mit „Acknowledge“. Der Host beendet die Übertragung der Liste durch einen im Scancode-Set 3 nicht vorhandenen Makecode (z.B. einen gültigen Befehl). Die Tastatur setzt dann mit der Tastenabfrage fort (sofern vorher deaktiviert).
- * 0xfc (Set Key Type Make/Break) - Wie vorheriger Befehl, es wird jedoch nur die Wiederholfunktion unterbunden.
- * 0xfb (Set Key Type Typematic) - Wie die beiden vorherigen Befehle, es werden jedoch nur die Breakcodes unterbunden.
- * 0xfa (Set All Keys Typematic/Make/Break) - Die Tastatur antwortet mit „Acknowledge“ (0xfa). Stellt die Normaleinstellung für alle Tasten her (Makecodes, Breakcodes und Wiederholfunktion ein)
- * 0xf9 (Set All Keys Make) - Die Tastatur antwortet mit „Acknowledge“ (0xfa). Ähnlich wie 0xfd gilt jedoch für alle Tasten.

- * 0xf8 (Set All Keys Make/Break) - Die Tastatur antwortet mit „Acknowledge“ (0xfa). Ähnlich wie 0xfc, gilt jedoch für alle Tasten.
- * 0xf7 (Set All Keys Typematic) - Die Tastatur antwortet mit „Acknowledge“ (0xfa). Ähnlich wie 0xfb, gilt jedoch für alle Tasten.
- 0xf6 (Set Default) - Lädt Grundeinstellungen: Wiederholfunktion 10.9cps / 500ms; Aktiviert Makecode, Breakcode und Wiederholfunktion für alle Tasten; stellt Scancode-Set 2 ein.
- 0xf5 (Disable) - Stoppt die Tastenabfrage, lädt die Grundeinstellung (siehe „Set Default“-Befehl) und wartet auf weitere Befehle.
- 0xf4 (Enable) - Aktiviert die Tastenabfrage, nachdem sie mit dem vorherigen Befehl deaktiviert wurde.
- 0xf3 (Set Typematic Rate/Delay) - Der Host sendet nach diesem Befehl ein Parameterbyte, welches die Wiederholrate und -verzögerung entsprechend folgender Tabelle einstellt:

Wiederholrate

Bits 0-4	Rate (cps)	Bits 0-4	Rate (cps)	Bits 0-4	Rate (cps)	Bits 0-4	Rate (cps)
0x00	30.0	0x08	15.0	0x10	7.5	0x18	3.7
0x01	26.7	0x09	13.3	0x11	6.7	0x19	3.3
0x02	24.0	0x0a	12.0	0x12	6.0	0x1a	3.0
0x03	21.8	0x0b	10.9	0x13	5.5	0x1b	2.7
0x04	20.7	0x0c	10.0	0x14	5.0	0x1c	2.5
0x05	18.5	0x0d	9.2	0x15	4.6	0x1d	2.3
0x06	17.1	0x0e	8.6	0x16	4.3	0x1e	2.1
0x07	16.0	0x0f	8.0	0x17	4.0	0x1f	2.0

Verzögerung

Bits 5-6	Verzögerung (sek)
0b00	0.25
0b01	0.50
0b10	0.75
0b11	1.00

- * 0xf2 (Read ID) - Die Tastatur antwortet mit einem Zwei-Byte ID-Code 0xab, 0x83. (0xab wird zuerst gesendet, danach 0x83)
- * 0xf0 (Set Scancode-Set) - Die Tastatur antwortet mit „Acknowledge“ (0xfa) und liest dann ein Parameterbyte vom Host. Der Parameter kann 0x01, 0x02, oder 0x03 sein, womit Scancode-Set 1, 2 oder 3 ausgewählt wird. Die Tastatur antwortet auf den Parameter mit „Acknowledge“. Wenn der Parameter 0x00 ist, antwortet die Tastatur mit „Acknowledge“, gefolgt vom aktuellen Scancode-Set.
- 0xee (Echo) - Die Tastatur antwortet mit „Echo“ (0xee).
- 0xed (Set/Reset LEDs) - Der Host sendet danach ein einen Parameter, mit dem die Num Lock, Caps Lock, und Scroll Lock LEDs eingestellt werden. Das Parameterbyte ist wie folgt definiert:

MSB				LSB			
immer 0	immer 0	immer 0	immer 0	immer 0	Caps-Lock	Num-Lock	Scroll-Lock

  - „Scroll Lock“ - Scroll Lock LED aus (0) / an (1)

– „Num Lock“ - Num Lock LED aus (0) / an (1)
– „Caps Lock“ - Caps Lock LED aus (0) / an (1)

Anm. d. Ü.: Bereits nach dem Befehlsbyte 0xed sendet die Tastatur ein „Acknowledge“. Befehl und Parameter dürfen nicht unmittelbar nacheinander gesendet werden. Nach dem Parameterbyte sendet die Tastatur noch ein „Acknowledge“ zurück.

* Ursprünglich nur auf PS/2 Tastaturen verfügbar

Anm. d. Ü.: Im Originaltext von Adam Chapweske folgt an dieser Stelle noch eine Beschreibung zur Programmierung des Keyboardcontrollers Intel 8042. Dies ist aber zum Verständnis der Tastaturansteuerung und zum Aufbau eigener Microcontroller-Schaltungen mit Tastatureingang nicht erforderlich. Diese Beschreibung kann im englischen Originaltext nachgelesen werden (siehe oben).

Es folgt noch eine kurze Darstellung der Vorgänge beim Einschalten des Computers:

1.9 Initialisierung

Nachfolgend die Kommunikation zwischen meinem Computer und der Tastatur beim Bootvorgang. Ich vermute, dass die ersten 3 Befehle durch den Tastaturcontroller initiiert werden, der nächste Befehl, der die LEDs setzt, vom BIOS gesendet wird, und die restlichen Befehle vom Betriebssystem (Win98SE). Beachten Sie, dass diese Angaben speziell nur für meinen Computer zutreffen; sie sollen nur eine allgemeine Vorstellung von der Tastaturinitialisierung geben.

```
Tastatur: 0xaa  Selbsttest bestanden         ;Keyboard controller init
Host:     0xed  Setzen der Status-LEDs
Tastatur: 0xfa  Acknowledge
Host:     0x00  Alle LEDs aus
Tastatur: 0xfa  Acknowledge
Host:     0xf2  Lese ID
Tastatur: 0xfa  Acknowledge
Tastatur: 0xab  Erstes Byte der ID
Host:     0xed  Setzen der Status-LEDs       ;BIOS init
Tastatur: 0xfa  Acknowledge
Host:     0x02  Num Lock LED an
Tastatur: 0xfa  Acknowledge
Host:     0xf3  Wiederholrate einstellen     ;Windows init
Tastatur: 0xfA  Acknowledge
Host:     0x20  500 ms / 30.0 Zeichen/sec
Tastatur: 0xfa  Acknowledge
Host:     0xf4  Enable
Tastatur: 0xfA  Acknowledge
Host:     0xf3  Wiederholrate einstellen
Tastatur: 0xfa  Acknowledge
Host:     0x00  250 ms / 30.0 Zeichen/sec
Tastatur: 0xfa  Acknowledge
```

2 Das PS/2 Maus / Tastatur-Protokoll

2.1 Einleitung

Dieser Artikel beschreibt den Anschluss der PS/2 Maus, der PS/2 Tastatur und der AT-Tastatur. Es werden die physikalischen und elektrischen Eigenschaften beschrieben, sowie das Low-Level-Kommunikationsprotokoll.

2.2 Die physikalische Schnittstelle

Den PS/2-Port gibt es mit zwei verschiedenen Steckverbindern: Als 5-poliger DIN-Stecker oder als 6-poliger Mini-DIN-Stecker. Beide sind, abgesehen von der Pinbelegung, elektrisch völlig gleichwertig. Beide Steckertypen können durch einen handelsüblichen oder selbst gebauten Adapter aufeinander umgesetzt werden, einfach indem die entsprechenden Pins miteinander verbunden werden. Die Steckverbindung ist genormt durcht das Deutsche Institut für Norm. Weitere Informationen dazu gibt es unter `www.din.de`

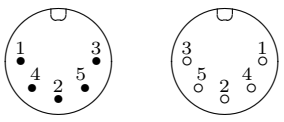
PC-Tastaturen haben entweder den 6-poligen Mini-DIN-Stecker oder den 5-poligen DIN-Stecker. Wenn Ihre Tastatur einen 6-poligen Stecker und Ihr Computer eine 5-polige Buchse hat (oder umgekehrt), können sie mit dem erwähnten Adapter miteinander verbunden werden. Tastaturen mit 6-poligem Mini-DIN-Stecker werden üblicherweise als „PS/2“-Tastatur bezeichnet, Tastaturen mit dem 5-poligen DIN-Stecker dagegen als „AT“-Tastatur („XT“-Tastaturen besaßen übrigens auch den 5-poligen DIN-Stecker. Diese werden allerdings schon seit Jahren nicht mehr hergestellt). Alle neueren Tastaturen für PC's sind entweder PS/2-, AT- oder USB-Tastaturen. USB-Tastaturen werden in diesem Artikel nicht beschrieben, da sie ein völlig anderes Protokoll verwenden.

Mäuse gibt es in den verschiedensten Formen, Ausführungen, und Schnittstellen. Die bekannteste Ausführung dürfte vermutlich die PS/2-Maus sein, die jedoch mittlerweile von der USB-Maus verdrängt wird. Noch vor einigen Jahren war die serielle Maus Standard. In diesem Artikel werden nur Mäuse mit PS/2-Anschluss beschrieben.

Das Anschlusskabel der Tastaturen und Mäuse ist vier bis sechspolig abgeschirmt Typ 26 AWG (ca. 0,13 mm2) und gewöhnlich 2 Meter lang. Wenn Sie ein längeres Kabel brauchen, können Sie eine entsprechende Verlängerung im Computer- oder Elektronikfachhandel kaufen. Mehrere Verlängerungen sollten nicht hintereinandergeschaltet werden. Wenn Sie 10 m Kabel brauchen, kaufen Sie eine 10 m Verlängerung. Schließen Sie nicht fünf 2 m Verlängerungen hintereinander. Die Kommunikation zwischen Tastatur/Maus und Computer könnte sonst unzuverlässig werden.

Nebenbei bemerkt, es gibt einen weiteren Steckertyp, auf den sie bei einer Tastatur stossen könnten. Während bei den meisten Tastaturen das Kabel fest verbunden ist, gibt es einige mit abnehmbaren Kabeln. Diese Kabel haben an der einen Seite den gewohnten DIN-Stecker (zum PC), und an der anderen Seite einen sogenannten SDL (Sheilded Data Link) Steckverbinder. SDL wurde von der Firma AMP eingeführt. Der Stecker besitzt eine gewisse Ähnlichkeit mit einem Western-Telefonstecker, da er Federzungen statt Stiften besitzt, und mit einer Federzunge einrastet. Mehr Informationen dazu gibt es auf der Webseite von AMP, `www.connect.amp.com/`. Verwechseln Sie den SDL-Stecker nicht mit dem USB-Stecker. Sie sehen auf der Zeichnung unten ähnlich aus; tatsächlich sind sie aber sehr verschieden. Der SDL-Stecker hat Federn und bewegliche Teile, der USB-Stecker hat keine beweglichen Teile.

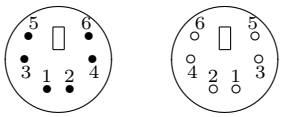
Die folgenden Abbildungen zeigen die Pinbelegungen für jeden Stecker:



Stecker Kupplung

5-pin DIN (AT/XT):
1 - Clock
2 - Data
3 - nicht belgt
4 - Ground
5 - V_{CC} (+5 V)

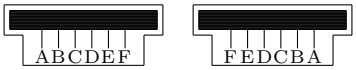
Abbildung 1: Pinbelegung 5-pin DIN (AT/XT)



Stecker Kupplung

6-pin Mini-DIN (PS/2):
1 - Data
2 - nicht belegt
3 - Ground
4 - V_{CC} (+5 V)
5 - Clock
6 - nicht belegt

Abbildung 2: Pinbelegung 6-pin Mini-DIN (PS/2)



Stecker Kupplung

6-pin SDL:
A - nicht belegt
B - Data
C - Ground
D - Clock
E - V_{CC} (+5 V)
F - nicht belegt

Abbildung 3: Pinbelegung 6-pin SDL

2.3 Die elektrische Schnittstelle

Hinweis: In diesem Text werde ich den Begriff „Host“ für den Computer, oder woran auch immer die Tastatur/Maus angeschlossen sein mag, verwenden. Die Tastatur oder Maus werde ich allgemein als „Eingabegerät“ bezeichnen.

Über V_{CC}/Ground wird die Tastatur/Maus mit Strom versorgt. Die Tastatur/Maus sollte nicht mehr als 275 mA verbrauchen. Besondere Vorsicht ist gegenüber Spannungsspitzen angebracht. Diese entstehen, wenn die Tastatur /Maus ein- oder ausgestöpselt wird, während der Computer eingeschaltet ist. Ältere Mainboards hatten SMD-Sicherungen als Schutz für die Maus- und Tastaturanschlüsse. Wenn diese durchbrannten, war das Board in der Regel unbrauchbar und durch einen durchschnittlichen Techniker nicht zu reparieren. Die meisten neueren Mainboards haben selbstheilende (Polyswitch-) Sicherungen, um diesem Problem abzuhelfen. Es ist jedoch nicht Standard, und es sind noch viele alte Mainboards im Einsatz. Deshalb rate ich davon ab, die Tastatur/Maus unter Spannung zu stecken.

Zusammenfassung Technische Daten der Stromversorgung:
$V_{CC} = +4.5\,\text{V}$ bis $+5.5\,\text{V}$
Maximale Stromaufnahme = 275 mA.

Die Data- und Clock-Leitungen sind vom Typ Open Collector mit Pullup-Widerständen. Eine Open-Collector-Schnittstelle hat zwei mögliche Zustände: Low oder hochohmig. Im Zustand Low schaltet ein Transistor gegen Ground durch. Im hochohmigen Zustand wird die Leitung weder High noch Low getrieben. Zusätzlich ist ein Pullup-Widerstand zwischen der Leitung und V_{CC} geschaltet. Dieser sorgt dafür, dass die Leitung High-Potential hat, wenn sie von keinem der Busteilnehmer auf Low gezogen wird. Der Wert dieses Widerstandes ist unkritisch und sollte zwischen 1 und 10 kΩ liegen. Höhere Werte reduzieren den Stromverbrauch, kleinere Werte reduzieren die Anstiegszeiten. Ein universelles Open-Collector-Interface ist auf der folgenden Abbildung gezeigt:

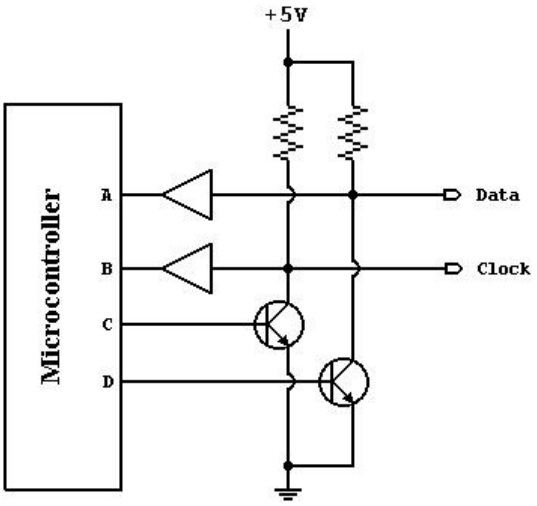


Abbildung 4: Allgemeines Open-Collector-Interface. Data und Clock werden an Pin A bzw. B des Microcontrollers eingelesen. Beide Leitungen haben im Ruhezustand +5 V und können auf Ground gezogen werden, indem logisch „1“ an C oder D angelegt wird. Folglich hat Data den invertierten Zustand von D, und Clock den invertierten Zustand von C.

> Hinweis: In meinen Beispielen habe ich ein paar Tricks benutzt, um ein Open-Collector-Interface mit PIC-Microcontrollern zu implementieren. Ich benutze den selben Port als Eingang und Ausgang und verwende die internen Pull-Up-Widerstände des PICs anstelle von externen Widerständen. Eine Leitung wird auf Low gezogen, indem man den entsprechenden Port als Ausgang setzt und logisch „0“ ausgibt. Den hochohmigen Zustand erreicht man, indem der Port als Eingang geschaltet wird. Mit den eingebauten Schutzdioden und dem hohen Stromtreibervermögen der PICs, denke ich, kann man das durchaus so machen.

2.4 Kommunikation: Allgemeine Beschreibung

PS/2 Mäuse und Tastaturen benutzen ein bidirektionales, synchrones serielles Protokoll. Im Ruhezustand sind beide Busleitungen High. Nur in diesem Zustand darf die Tastatur/Maus mit der Datenübertragung beginnen. Der Host hat die absolute Kontrolle über den Bus und darf die Kommunikation jederzeit unterbrechen, indem er die Clockleitung auf Low zieht.

Das Clocksignal wird immer vom Eingabegerät erzeugt. Wenn der Host Daten senden möchte, muss er zunächst die Kommunikation unterbrechen, indem er die Clockleitung auf Low zieht. Anschließend zieht er die Datenleitung ebenfalls auf Low und setzt die Clockleitung wieder auf High, indem er in den Clock-Port hochohmig werden lässt. Dieser Zustand ist der „Request-to-Send“-Zustand und weist das Eingabegerät an, jetzt mit der Ausgabe der Clockimpulse zu beginnen.

Zusammenfassung: Bus-Zustände
Data = high, Clock = high: *Ruhezustand (Idle)*
Data = high, Clock = low: *Kommunikation gestoppt (Inhibit)*
Data = low, Clock = high: *Sendeanforderung (Request-to-Send) durch den Host*

Alle Daten werden byteweise gesendet. Jedes Byte besteht aus einem Frame zu 11 oder 12 Bits:

- 1 Startbit. Dieses ist immer 0
- 8 Datenbits mit LSB voran
- 1 Parity-Bit (ungerade Parität)

- 1 Stopbit. Diese ist immer 1
- 1 Acknowledge-Bit (nur bei Datenübertragung vom Host zum Eingabegerät)

Das Paritybit ist 1, wenn das Datenbyte eine gerade Anzahl von Einsen enthält, und 0 bei einer ungeraden Anzahl. Die Anzahl der Einsen im Datenbyte plus das Paritybyte ist immer ungerade (ungerade Parität). Dieses Verfahren dient zur Fehlererkennung. Die Tastatur/Maus muss das Paritybit prüfen und im Fehlerfall wie auf ein ungültiges Kommando reagieren.

Daten, die vom Eingabegerät zum Host gesendet werden, werden mit der fallenden Flanke des Clock-Signals gelesen; Daten vom Host zum Eingabegerät mit der steigenden Flanke. Die Clockfrequenz muss im Bereich von 10-16,7 kHz liegen. Das heißt, das Clock-Signal muss für 30-50 μs high und für 30-50 μs low sein. Wenn Sie eine Tastatur, Maus oder einen Host Emulator entwickeln, sollten Sie die Datenleitung in der Mitte jedes Clockimpulses ändern bzw. abtasten, d.h. 15-25 μs nach der entsprechenden Clock-Flanke. Noch einmal: Das Clock-Signal wird immer von der Tastatur/Maus erzeugt, aber der Host hat die Kontrolle über die Kommunikation.

Das Timing ist absolut entscheidend. Alle Zeitangaben müssen genau eingehalten werden.

2.5 Kommunikation: Vom Eingabegerät zum Host

Daten- und Clockleitung sind Open-Collector-Leitungen. Ein Widerstand ist von jeder Leitung nach +5V verbunden, somit ist der Ruhezustand high. Wenn die Tastatur/Maus Daten senden will, prüft sie zunächst, ob die Clockleitung High ist. Falls nicht, blockiert der Host gerade die Übertragung, und alle zu sendenden Zeichen müssen zwischengespeichert werden, bis der Host die Clockleitung wieder freigibt. Die Clockleitung muss für mindstens 50 μs durchgehend High sein, bevor das Eingabegerät mit der Übertragung beginnen kann.

Wie bereits im letzten Abschnitt erwähnt, benutzen Tastatur und Maus ein serielles Protokoll mit 11 Bit langen Frames:

- 1 Startbit. Dieses ist immer 0.
- 8 Datenbits mit dem LSB voran
- 1 Paritätsbit (ungerade Parität)
- 1 Stopbit. Dieses ist immer 1

Das Eingabegerät gibt ein Datenbit auf den Bus, wenn Clock high ist; der Host liest es ein, wenn Clock low ist (siehe Abbildung 5 und 6).

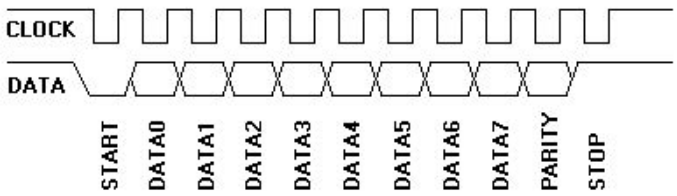


Abbildung 5: Übertragung vom Eingabegerät zum Host. Die Datenleitung ändert ihren Zustand wenn die Clockleitung high ist. Daten sind gültig, wenn Clock low ist.

Abbildung 6: Scancode für die Taste „Q“ (0x15), von der Tastatur zum Computer gesendet. Kanal A ist der Clock, Kanal B sind die Daten.

Die Clockfrequenz beträgt 10 - 16,7 kHz. Zwischen der steigenden Flanke von Clock und einem Wechsel von Data müssen mindestens 5 μs liegen. Zwischen einem Wechsel von Data und der fallenden Flanke von Clock müssen mindestens 5 μs und maximal 25 μs liegen.

Der Host kann die Übertragung jederzeit unterbrechen, indem er die Clockleitung für mindestens 100 μs auf low zieht. Wenn eine Übertragung vor dem 11. Clockimpuls unterbrochen wird, muss das Eingabegerät den Rest der Übertragung verwerfen und die Neuübertragung des aktuellen „Chunks“ vorbereiten, wenn der Host die Clockleitung wieder freigibt. Ein „Chunk“ ist z.B. ein Makecode, ein Breakcode, ein Device-ID, ein Mouse-Move-Paket usw. Wird z.B. eine Tastatur bei der Übertragung des zweiten Byte eines Zwei-Byte Breakcodes unterbrochen, so muss der komplette Breakcode noch einmal gesendet werden, und nicht nur das zweite Byte.

Wenn der Host die Clockleitung vor dem ersten Clock-Impuls (fallende Flanke) oder nach der fallenden Flanke des letzten Clock-Impulses auf low zieht, ist keine Wiederholung erforderlich. Fallen aber neue Daten an, die gesendet werden müssen, so müssen diese solange gespeichert werden, bis der Host die Clockleitung freigibt. Tastaturen haben zu diesem Zweck einen 16 Byte Buffer. Fallen mehr als 16 Byte durch Tastaturanschläge an, werden diese ignoriert, bis im Buffer wieder freier Platz ist. Mäuse speichern nur das jeweils letze Datenpaket.

2.6 Kommunikation: Vom Host zum Eingabegerät

In dieser Richtung sehen die Datenpakete etwas anders aus. Zunächst einmal werden die Clockimpulse immer vom Eingabegerät erzeugt. Will der Host Daten zum Eingabegerät senden, muss er zuerst Daten- und Clockleitung wie folgt in den Request-To-Send-Zustand bringen:

- Unterbrechen der Kommunikation, indem die Clockleitung für mindestens 100 μs auf Low gezogen wird
- Ausgeben des „Request-to-send“-Signals, indem Data auf Low gezogen wird, und anschließend Clock wieder freigegeben wird.

Das Eingabegerät sollte den Bus in Abständen von maximal 100 ms auf diesen Zustand prüfen. Wenn das Eingabegerät diesen Zustand erkennt, beginnt es Clockimpulse für 8 Datenbits, Paritätsbit und ein Stopbit auszugeben. Der Host ändert die Datenleitung entsprechend wenn Clock low ist. Die Daten werden vom Eingabegerät übernommen, wenn Clock high ist. Dies ist der umgekehrte Vorgang der Kommunikation vom Eingabegerät zum Host.

Nachdem das Stopbit empfangen wurde, bestätigt das Eingabegerät, indem Data Low gesetzt wird (Acknowledge) und ein letzter Clockimpuls ausgegeben wird. Sollte der Host die Datenleitung nach dem 11. Clockimpuls nicht freigeben, gibt das Eingabegerät solange Clockimpulse aus, bis die Datenleitung freigegeben wird (das Eingabegerät erzeugt dabei einen Fehler).

Der Host kann die Übertragung jederzeit vor dem 11. Clockimpuls abbrechen, indem er Clock für mindestens 100 μs low hält.

Zur Verdeutlichung hier noch einmal die Schritte, die der Host bei der Datenübertragung zum PS/2-Eingabegerät einhalten muss:

1. Die Clock-Leitung für mindestens 100 μs auf low bringen
2. Die Datenleitung auf low bringen
3. Die Clockleitung wieder high werden lassen
4. Warten, bis das Eingabegerät die Clockleitung auf low bringt
5. Die Datenleitung entsprechend dem ersten Datenbit auf 0 oder 1 setzen
6. Warten, bis das Eingabegerät die Clockleitung auf high bringt
7. Warten, bis das Eingabegerät die Clockleitung auf low bringt
8. Wiederholen der Schritte 5 bis 7 für die anderen 7 Datenbits und das Paritätsbit
9. Die Datenleitung freigeben
10. Warten, bis das Eingabegerät die Datenleitung auf low bringt
11. Warten, bis das Eingabegerät die Clockleitung auf low bringt
12. Warten, bis das Eingabegerät Clockleitung und Datenleitung wieder freigibt

Abbildung 7 zeigt diesen Vorgang und Abbildung 8 zeigt noch einmal getrennt, welche Signale vom Host und welche vom Eingabegerät (Device) erzeugt werden. Beachten Sie, dass sich beim Acknowledge-Bit die Datenleitung - anders als üblich - ändert, während Clock high ist.

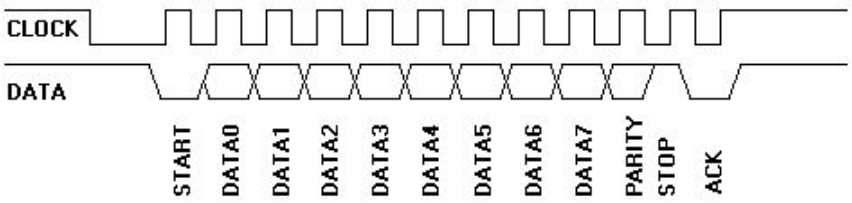


Abbildung 7: Übertragung vom Host zum Eingabegerät

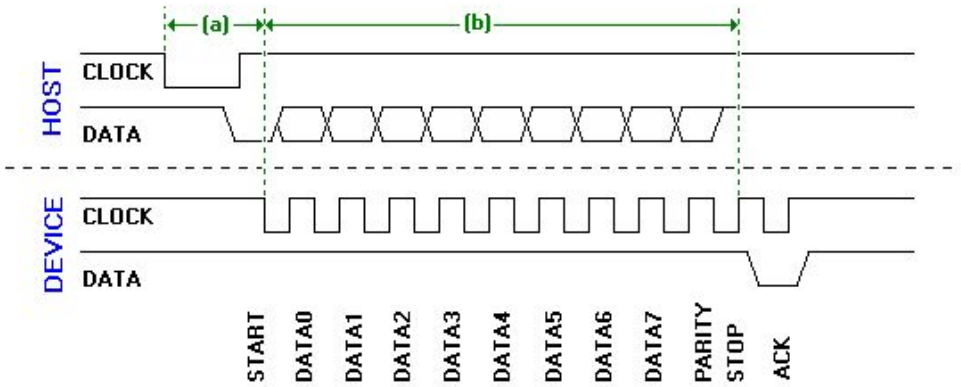


Abbildung 8: Übertragung vom Host zum Eingabegerät, getrennt für beide Geräte

Bezugnehmend auf Abbildung 8 muss der Host 2 Zeitabschnitte beachten.

(a) ist die Zeit innerhalb der das Eingabegerät beginnt, Clockimpulse auszugeben, nachdem der Host erstmals die Clockleitung auf low gesetzt hat. Sie darf nicht länger als 15 ms sein.

(b) ist die Zeit für die Übertragung des gesamten Datenpakets, die nicht größer als 2 ms sein darf. Wird eine dieser Vorgaben nicht eingehalten, sollte der Host einen Error generieren. Unmittelbar nach dem Acknowledge-Bit darf der Host die Clockleitung auf low bringen um die Kommunikation zu unterbrechen, während er die Daten verarbeitet. Wenn das vom Host gesendete Kommando eine Quittierung verlangt, muss diese innerhalb von 20 ms gesendet werden, nachdem die Clockleitung freigegeben wurde. Fehlt diese Quittierung, generiert der Host einen Fehler.

A Keyboard Scancodes: Set 1

A.1 101-, 102- and 104-key keyboards:

KEY	MAKE	BREAK	KEY	MAKE	BREAK
A	0x1e	0x9e	APPS	0xe0, 0x5d	0xe0, 0xdd
B	0x30	0xb0	ENTER	0x1c	0x9c
C	0x2e	0xae	ESC	0x01	0x81
D	0x20	0xa0	F1	0x3b	0xbb
E	0x12	0x92	F2	0x3c	0xbc
F	0x21	0xa1	F3	0x3d	0xbd
G	0x22	0xa2	F4	0x3e	0xbe
H	0x23	0xa3	F5	0x3f	0xbf
I	0x17	0x97	F6	0x40	0xc0
J	0x24	0xa4	F7	0x41	0xc1
K	0x25	0xa5	F8	0x42	0xc2
L	0x26	0xa6	F9	0x43	0xc3
M	0x32	0xb2	F10	0x44	0xc4
N	0x31	0xb1	F11	0x57	0xd7
O	0x18	0x98	F12	0x58	0xd8
P	0x19	0x99	PRNT SCRN	0xe0, 0x2a, 0xe0, 0x37,	0xe0, 0xb7 0xe0, 0xaa
Q	0x10	0x19	SCROLL	0x46	0xc6
R	0x13	0x93	PAUSE	0xe1, 0x1d, 0x45, 0xe1, 0x9d, 0xc5	-none-
S	0x1f	0x9f	[	0x1a	0x9a
T	0x14	0x94	INSERT	0xe1, 0x52	0xe1, 0xd2
U	0x16	0x96	HOME	0xe1, 0x47	0xe1, 0xc7
V	0x2f	0xaf	PG UP	0xe1, 0x49	0xe1, 0xc9
W	0x11	0x91	DELETE	0xe1, 0x53	0xe1, 0xd3
X	0x2d	0xad	END	0xe1, 0x4f	0xe1, 0xcf
Y	0x15	0x95	PG DN	0xe1, 0x51	0xe1, 0xd1
Z	0x2c	0xac	U ARROW	0xe1, 0x48	0xe1, 0xc8
0	0x0b	0x8b	L ARROW	0xe1, 0x4b	0xe1, 0xcb
1	0x02	0x82	D ARROW	0xe1, 0x50	0xe1, 0xd0
2	0x03	0x83	R ARROW	0xe1, 0x4d	0xe1, 0xcd
3	0x04	0x84	NUM	0x45	0xc5
4	0x05	0x85	KP /	0xe1, 0x35	0xe1, 0xb5
5	0x06	0x86	KP *	0x37	b7
6	0x07	0x87	KP -	0x4a	ca
7	0x08	0x88	KP +	0x4e	ce
8	0x09	0x89	KP EN	0xe0, 0x1c	0xe0, 0x9c
9	0x0a	0x8a	KP .	0x53	0xd3
‘	0x29	0x89	KP 0	0x52	0xd2
-	0x0c	0x8c	KP 1	0x4f	0xcf
=	0x0d	0x8d	KP 2	0x50	0xd0
\	0x2b	0xab	KP 3	0x51	0xd1
BKSP	0x0e	0x8e	KP 4	0x4b	0xcb
SPACE	0x39	0xb9	KP 5	0x4c	0xcc
TAB	0x0f	0x8f	KP 6	0x4d	0xcd
CAPS	0x3a	0xba	KP 7	0x47	0xc7
L SHFT	0x2a	0xaa	KP 8	0x48	0xc8
L CTRL	0x1d	0x9d	KP 9	0x49	0xc9
L GUI	0xe0, 0x5b	0xe0, 0xdb	]	0x1b	0x9b
L ALT	0x38	0xb8	;	0x27	0xa7
R SHFT	0x36	0xb6	’	0x28	0xa8
R CTRL	0xe0, 0x1d	0xe0, 0x9d	,	0x33	0xb3
R GUI	0xe0, 0x5c	0xe0, 0xdc	.	0x34	0xb4
R ALT	0xe0, 0x38	0xe0, 0xb8	/	0x35	0xb5

A.2 ACPI Scancodes:

KEY	MAKE	BREAK
Power	0xe0, 0x5e	0xe0, 0xde
Sleep	0xe0, 0x5f	0xe0, 0xdf
Wake	0xe0, 0x63	0xe0, 0xe3

A.3 Windows Multimedia Scancodes:

KEY	MAKE	BREAK
Next Track	0xe0, 0x19	0xe0, 0x99
Previous Track	0xe0, 0x10	0xe0, 0x90
Stop	0xe0, 0x24	0xe0, 0xa4
Play/Pause	0xe0, 0x22	0xe0, 0xa2
Mute	0xe0, 0x20	0xe0, 0xa0
Volume Up	0xe0, 0x30	0xe0, 0xb0
Volume Down	0xe0, 0x2e	0xe0, 0xae
Media Select	0xe0, 0x6d	0xe0, 0xed
E-Mail	0xe0, 0x6c	0xe0, 0xec
Calculator	0xe0, 0x21	0xe0, 0xa1
My Computer	0xe0, 0x6b	0xe0, 0xeb
WWW Search	0xe0, 0x65	0xe0, 0xe5
WWW Home	0xe0, 0x32	0xe0, 0xb2
WWW Back	0xe0, 0x6a	0xe0, 0xea
WWW Forward	0xe0, 0x69	0xe0, 0xe9
WWW Stop	0xe0, 0x68	0xe0, 0xe8
WWW Refresh	0xe0, 0x67	0xe0, 0xe7
WWW Favorites	0xe0, 0x66	0xe0, 0xe6

B Keyboard Scancodes: Set 2

B.1 101-, 102- and 104-key keyboards:

KEY	MAKE	BREAK	KEY	MAKE	BREAK
A	0x1c	0xf0, 0x1c	APPS	0xe0, 0x2f	0xe0, 0xf0, 0x2f
B	0x32	0xf0, 0x32	ENTER	0x5a	0xf0, 0x5a
C	0x21	0xf0, 0x21	ESC	0x76	0xf0, 0x76
D	0x23	0xf0, 0x23	F1	0x05	0xf0, 0x05
E	0x24	0xf0, 0x24	F2	0x06	0xf0, 0x06
F	0x2b	0xf0, 0x2b	F3	0x04	0xf0, 0x04
G	0x34	0xf0, 0x34	F4	0x0c	0xf0, 0x0c
H	0x33	0xf0, 0x33	F5	0x03	0xf0, 0x03
I	0x43	0xf0, 0x43	F6	0x0b	0xf0, 0x0b
J	0x3b	0xf0, 0x3b	F7	0x83	0xf0, 0x83
K	0x42	0xf0, 0x42	F8	0x0a	0xf0, 0x0a
L	0x4b	0xf0, 0x4b	F9	0x01	0xf0, 0x01
M	0x3a	0xf0, 0x3a	F10	0x09	0xf0, 0x09
N	0x31	0xf0, 0x31	F11	0x78	0xf0, 0x78
O	0x44	0xf0, 0x44	F12	0x07	0xf0, 0x07
P	0x4d	0xf0, 0x4d	PRNT SCRN	0xe0, 0x12, 0xe0, 0x7c	0xe0, 0xf0, 0x7c, 0xe0, 0xf0, 0x12
Q	0x15	0xf0, 0x15	SCROLL	0x7e	0xf0, 0x7e
R	0x2d	0xf0, 0x2d	PAUSE	0xe1, 0x14, 0x77, 0xe1, 0xf0, 0x14, 0xf0, 0x77	-none-
S	0x1b	0xf0, 0x1b	[	0x54	0xf0, 0x54
T	0x2c	0xf0, 0x2c	INSERT	0xe0, 0x70	e0,0xf0, 0x70
U	0x3c	0xf0, 0x3c	HOME	0xe0, 0x6c	e0,0xf0, 0x6c
V	0x2a	0xf0, 0x2a	PG UP	0xe0, 0x7d	e0,0xf0, 0x7d
W	0x1d	0xf0, 0x1d	DELETE	0xe0, 0x71	e0,0xf0, 0x71
X	0x22	0xf0, 0x22	END	0xe0, 0x69	e0,0xf0, 0x69
Y	0x35	0xf0, 0x35	PG DN	0xe0, 0x7a	e0,0xf0, 0x7a
Z	0x1a	0xf0, 0x1a	U ARROW	0xe0, 0x75	e0,0xf0, 0x75
0	0x45	0xf0, 0x45	L ARROW	0xe0, 0x6b	e0,0xf0, 0x6b
1	0x16	0xf0, 0x16	D ARROW	0xe0, 0x72	e0,0xf0, 0x72
2	0x1e	0xf0, 0x1e	R ARROW	0xe0, 0x74	e0,0xf0, 0x74
3	0x26	0xf0, 0x26	NUM	0x77	0xf0, 0x77
4	0x25	0xf0, 0x25	KP /	0xe0, 0x4a	e0,0xf0, 0x4a
5	0x2e	0xf0, 0x2e	KP *	0x7c	0xf0, 0x7c
6	0x36	0xf0, 0x36	KP -	0x7b	0xf0, 0x7b
7	0x3d	0xf0, 0x3d	KP +	0x79	0xf0, 0x79
8	0x3e	0xf0, 0x3e	KP EN	0xe0, 0x5a	e0,0xf0, 0x5a
9	0x46	0xf0, 0x46	KP .	0x71	0xf0, 0x71
‘	0x0e	0xf0, 0x0e	KP 0	0x70	0xf0, 0x70
-	0x4e	0xf0, 0x4e	KP 1	0x69	0xf0, 0x69
=	0x55	0xf0, 0x55	KP 2	0x72	0xf0, 0x72
\	0x5d	0xf0, 0x5d	KP 3	0x7a	0xf0, 0x7a
BKSP	0x66	0xf0, 0x66	KP 4	0x6b	0xf0, 0x6b
SPACE	0x29	0xf0, 0x29	KP 5	0x73	0xf0, 0x73
TAB	0x0d	0xf0, 0x0d	KP 6	0x74	0xf0, 0x74
CAPS	0x58	0xf0, 0x58	KP 7	0x6c	0xf0, 0x6c
L SHFT	0x12	0xf0, 0x12	KP 8	0x75	0xf0, 0x75
L CTRL	0x14	0xf0, 0x14	KP 9	0x7d	0xf0, 0x7d
L GUI	0xe0, 0x1f	0xe0, 0xf0, 0x1f	]	0x5b	0xf0, 0x5b
L ALT	0x11	0xf0, 0x11	;	0x4c	0xf0, 0x4c
R SHFT	0x59	0xf0, 0x59	’	0x52	0xf0, 0x52
R CTRL	0xe0, 0x14	0xe0, 0xf0, 0x14	,	0x41	0xf0, 0x41
R GUI	0xe0, 0x27	0xe0, 0xf0, 0x27	.	0x49	0xf0, 0x49
R ALT	0xe0, 0x11	0xe0, 0xf0, 0x11	/	0x4a	0xf0, 0x4a

B.2 ACPI Scancodes:

KEY	MAKE	BREAK
Power	0xe0, 0x37	0xe0, 0xf0, 0x37
Sleep	0xe0, 0x3f	0xe0, 0xf0, 0x3f
Wake	0xe0, 0x5e	0xe0, 0xf0, 0x5e

B.3 Windows Multimedia Scancodes:

KEY	MAKE	BREAK
Next Track	0xe0, 0x4d	0xe0, 0xf0, 0x4d
Previous Track	0xe0, 0x15	0xe0, 0xf0, 0x15
Stop	0xe0, 0x3b	0xe0, 0xf0, 0x3b
Play/Pause	0xe0, 0x34	0xe0, 0xf0, 0x34
Mute	0xe0, 0x23	0xe0, 0xf0, 0x23
Volume Up	0xe0, 0x32	0xe0, 0xf0, 0x32
Volume Down	0xe0, 0x21	0xe0, 0xf0, 0x21
Media Select	0xe0, 0x50	0xe0, 0xf0, 0x50
E-Mail	0xe0, 0x48	0xe0, 0xf0, 0x48
Calculator	0xe0, 0x2b	0xe0, 0xf0, 0x2b
My Computer	0xe0, 0x40	0xe0, 0xf0, 0x40
WWW Search	0xe0, 0x10	0xe0, 0xf0, 0x10
WWW Home	0xe0, 0x3a	0xe0, 0xf0, 0x3a
WWW Back	0xe0, 0x38	0xe0, 0xf0, 0x38
WWW Forward	0xe0, 0x30	0xe0, 0xf0, 0x30
WWW Stop	0xe0, 0x28	0xe0, 0xf0, 0x28
WWW Refresh	0xe0, 0x20	0xe0, 0xf0, 0x20
WWW Favorites	0xe0, 0x18	0xe0, 0xf0, 0x18

C Keyboard Scancodes: Set 3

KEY	MAKE	BREAK	KEY	MAKE	BREAK
A	0x1c	0xf0, 0x1c	APPS	0x8d	0xf0, 0x8d
B	0x32	0xf0, 0x32	ENTER	0x5a	0xf0, 0x5a
C	0x21	0xf0, 0x21	ESC	0x08	0xf0, 0x08
D	0x23	0xf0, 0x23	F1	0x07	0xf0, 0x07
E	0x24	0xf0, 0x24	F2	0x0f	0xf0, 0x0f
F	0x2b	0xf0, 0x2b	F3	0x17	0xf0, 0x17
G	0x34	0xf0, 0x34	F4	0x1f	0xf0, 0x1f
H	0x33	0xf0, 0x33	F5	0x27	0xf0, 0x27
I	0x43	0xf0, 0x48	F6	0x2f	0xf0, 0x2f
J	0x3b	0xf0, 0x3b	F7	0x37	0xf0, 0x37
K	0x42	0xf0, 0x42	F8	0x3f	0xf0, 0x3f
L	0x4b	0xf0, 0x4b	F9	0x47	0xf0, 0x47
M	0x3a	0xf0, 0x3a	F10	0x4f	0xf0, 0x4f
N	0x31	0xf0, 0x31	F11	0x56	0xf0, 0x56
O	0x44	0xf0, 0x44	F12	0x5e	0xf0, 0x5e
P	0x4d	0xf0, 0x4d	PRNT SCRN	0x57	0xf0, 0x57
Q	0x15	0xf0, 0x15	SCROLL	0x5f	0xf0, 0x5f
R	0x2d	0xf0, 0x2d	PAUSE	0x62	0xf0, 0x62
S	0x1b	0xf0, 0x1b	[	0x54	0xf0, 0x54
T	0x2c	0xf0, 0x2c	INSERT	0x67	0xf0, 0x67
U	0x3c	0xf0, 0x3c	HOME	0x6e	0xf0, 0x6e
V	0x2a	0xf0, 0x2a	PG UP	0x6f	0xf0, 0x6f
W	0x1d	0xf0, 0x1d	DELETE	0x64	0xf0, 0x64
X	0x22	0xf0, 0x22	END	0x65	0xf0, 0x65
Y	0x35	0xf0, 0x35	PG DN	0x6d	0xf0, 0x6d
Z	0x1a	0xf0, 0x1a	U ARROW	0x63	0xf0, 0x63
0	0x45	0xf0, 0x45	L ARROW	0x61	0xf0, 0x61
1	0x16	0xf0, 0x16	D ARROW	0x60	0xf0, 0x60
2	0x1e	0xf0, 0x1e	R ARROW	0x6a	0xf0, 0x6a
3	0x26	0xf0, 0x26	NUM	0x76	0xf0, 0x76
4	0x25	0xf0, 0x25	KP /	0x4a	0xf0, 0x4a
5	0x2e	0xf0, 0x2e	KP *	0x7e	0xf0, 0x7e
6	0x36	0xf0, 0x36	KP -	0x4e	0xf0, 0x4e
7	0x3d	0xf0, 0x3d	KP +	0x7c	0xf0, 0x7c
8	0x3e	0xf0, 0x3e	KP EN	0x79	0xf0, 0x79
9	0x46	0xf0, 0x46	KP .	0x71	0xf0, 0x71
‘	0x0e	0xf0, 0x0e	KP 0	0x70	0xf0, 0x70
-	0x4e	0xf0, 0x4e	KP 1	0x69	0xf0, 0x69
=	0x55	0xf0, 0x55	KP 2	0x72	0xf0, 0x72
\	0x5c	0xf0, 0x5c	KP 3	0x7a	0xf0, 0x7a
BKSP	0x66	0xf0, 0x66	KP 4	0x6b	0xf0, 0x6b
SPACE	0x29	0xf0, 0x29	KP 5	0x73	0xf0, 0x73
TAB	0x0d	0xf0, 0x0d	KP 6	0x74	0xf0, 0x74
CAPS	0x14	0xf0, 0x14	KP 7	0x6c	0xf0, 0x6c
L SHFT	0x12	0xf0, 0x12	KP 8	0x75	0xf0, 0x75
L CTRL	0x11	0xf0, 0x11	KP 9	0x7d	0xf0, 0x7d
L GUI	0x8b	0xf0, 0x8b	]	0x5b	0xf0, 0x5b
L ALT	0x19	0xf0, 0x19	;	0x4c	0xf0, 0x4c
R SHFT	0x59	0xf0, 0x59	’	0x52	0xf0, 0x52
R CTRL	0x58	0xf0, 0x58	,	0x41	0xf0, 0x41
R GUI	0x8c	0xf0, 0x8c	.	0x49	0xf0, 0x49
R ALT	0x39	0xf0, 0x39	/	0x4a	0xf0, 0x4a