Heinrich Heine
Universität Düsseldorf

UNIVERSITÄT DÜSSELDORF
Mathematisch-Naturwissenschaftliche Fakultät
Institut für Informatik

Prof. Dr. Michael Schöttner

Zeichen- und Tastaturcodes

ASCII-Code

Der "American Standard Code for Information Interchange" ist eine in der Computerwelt sehr weit verbreitete Zuordnungstabelle für die Darstellung von Buchstaben, Ziffern und Sonderzeichen. Ursprünglich waren pro Zeichen 7 Bits vorgesehen, mittlerweile haben sich 8 Bits, also ein Byte durchgesetzt. Die folgende Tabelle zeigt einen kleinen Ausschnitt aus der ASCII-Tabelle:

Zeichen	ASCII-Code
(	40
0	48
1	49
2	50
A	65
B	66
a	97

Zeichen und Zeichenketten werden üblicherweise im ASCII-Code abgespeichert.

Scancode

Mit dem Scancode werden den Tasten der PC-Tastatur eindeutige Nummern zugeordnet. Damit ist es auch möglich, Tasten, wie die Cursortasten, denen kein druckbares Zeichen entspricht, zu identifizieren. Bei Scancodes wird nicht zwischen Groß- und Kleinbuchstaben unterschieden, da beide mit derselben Taste erreicht werden.

Taste	Scancode
A	30
S	31
D	32
Cursor hoch	72
Cursor runter	80

Im Laufe der PC-Entwicklungsgeschichte wurden unterschiedliche Tastaturen mit einer unterschiedlichen Anzahl und Bedeutung von Tasten herausgebracht. Gerade bei den Funktions- und Spezialtasten gibt es daher auch unterschiedliche Scancodes. Da PC-Tastaturen nur wenig mehr als 100 Tasten besitzen, genügen 7 Bits, um den Scancode darzustellen.

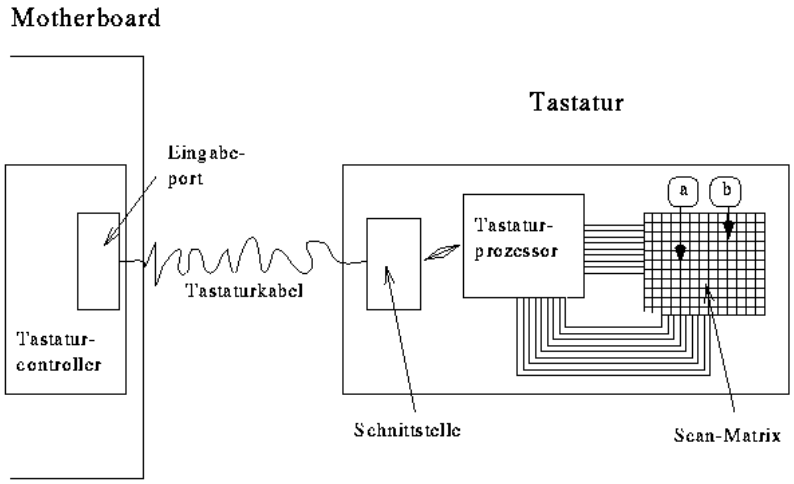
Make- und Breakcodes

Programme müssen nicht nur feststellen können, welche der "normalen" Tasten gedrückt wurden, sondern auch, ob gleichzeitig die Shift-(Umschalt-)Taste, die Control-(Steuerungs-)-Taste oder die ALT-Taste festgehalten wurden. Daher sendet die Tastatur statt eines einfachen Scancodes einen oder mehrere sogenannte Makecodes für jedes Drücken und einen oder mehrere Breakcodes für jedes Loslassen einer Taste. Wenn eine Taste länger als eine bestimmte Zeitspanne festgehalten wird, werden darüber hinaus zusätzliche Makecodes gesendet, durch die die Wiederholungsfunktion realisiert wird. Bei den meisten Tasten entspricht der Makecode dem Scancode und der Breakcode dem Scancode mit gesetztem 7. Bit. Einige Tasten erzeugen jedoch aus historischen Gründen schon beim einmaligen Drücken und Loslassen mehrere Make- und Breakcodes.

Anmerkung: Da die Interpretation der Make- und Breakcodes sehr mühsam, langweilig und wenig lehrreich ist, ist die Dekodierung in der Vorgabe. Es gibt hier jedoch Unterschiede zwischen verschiedenen Tastaturen. Es kann daher sein, dass die Vorgabe nicht alle Zeichen korrekt erkennt, insbesondere die deutschen Umlaute. Dies kann an die eigene Hardware angepasst werden. Dies aber im Rahmen der Lehrveranstaltung nicht zwingend notwendig.

Ablauf beim Drücken einer Taste

Wenn bei einer PC-Tastatur eine Taste gedrückt wird, werden zwei sich kreuzende Leitungen der *Scan-Matrix* innerhalb der Tastatur verbunden. Der Tastaturprozessor (8042 für PC/XT-, 8048 für AT- und MF II-Tastaturen) ermittelt die Position der gedrückten Taste und daraus den Scancode. Über eine serielle Schnittstelle wird der Code zum Motherboard des PCs gesendet.



Auf dem Motherboard des PCs befindet sich ein Tastaturcontroller, der auf der einen Seite mit der Tastatur über einen Eingabeport und einen Ausgabeport (Kommandos an die Tastatur) kommuniziert. Auf der anderen Seite hat der Controller Register, die mit Hilfe von `in`- und `out`-Befehlen über den Systembus gelesen und beschrieben werden können.

Port	lesen(R), schreiben(W)	Register	Bedeutung
0x60	R	Ausgabepuffer	Make-/Breakcode von der Tastatur
0x60	W	Eingabepuffer	Befehle für die Tastatur (z.B. LEDs setzen)
0x64	W	Steuerregister	Befehle für den Tastaturcontroller
0x64	R	Statusregister	Zustand des Tastaturcontrollers (z.B. Ausgabepuffer voll?)

Immer wenn ein Byte vom Tastaturcontroller in seinen Ausgabepuffer geschrieben wird, signalisiert er das durch Setzen einer Interruptanforderung. Der Prozessor muss darauf reagieren, indem er das angekommene Byte aus dem Ausgabepuffer ausliest. Im Statusregister wird dann vermerkt, dass der Ausgabepuffer wieder leer ist. Erst jetzt können neue Zeichen von der Tastatur entgegengenommen werden. Wenn die Tastatur im Pollingbetrieb benutzt wird, kann durch Bit 0 überprüft werden, ob sich tatsächlich ein Zeichen im Ausgabepuffer des Tastaturcontrollers befindet. In die Gegenrichtung muss immer gewartet werden, bis der Eingabepuffer des Tastaturcontrollers leer ist (Bit 1 gelöscht), bevor ein neues Zeichen geschrieben wird.

Bei PS/2-PCs ist die Maus ebenfalls an den Tastaturcontroller angeschlossen. Dadurch landen im Ausgabepuffer sowohl Codes von der Tastatur als auch von der Maus. Damit die Quelle des Bytes unterschieden werden, steht im Statusregister das Bit 5 (AUXB) zur Verfügung (1 = Maus, 0 = Tastatur).

Bit	Maske	Name	Bedeutung
0	0x01	outb	Gesetzt, wenn ein Zeichen im Ausgabepuffer des Tastaturcontrollers zum Lesen bereitsteht
1	0x02	inpb	Gesetzt, solange der Tastaturcontroller ein von der CPU geschriebenes Zeichen noch nicht abgeholt hat
5	0x20	auxb	Gesetzt, wenn der Wert im Ausgabepuffer von der Maus und nicht von der Tastatur stammt

Tastaturcontroller programmieren
Der Tastaturcontroller kann durch das Schicken von Befehlscodes an den Eingabepuffer konfiguriert werden. Zuvor sollte man sicherstellen, das der Eingabepuffer des Tastaturcontrollers leer ist (inpb). Danach wird der Befehlscode (siehe Tabelle) an den Datenport geschrieben. Danach sollte man warten, bis der Tastaturcontroller geantwortet hat und der Ausgabepuffer das Bestätigungsbyte 0xfa (ACK) enthält (wieder vor dem Lesen outb prüfen). Wir werden von den etwa 20 Befehlen, die der Tastaturcontroller versteht, nur zwei verwenden:

Befehlscode	Name	Beschreibung
0xed	set_led	Ein-/Ausschalten der Tastatur-LEDs. Nach dem Befehlscode muss ein weiteres Byte an den Tastaturcontroller geschrieben werden, das den Zustand der LEDs steuert. Der Aufbau ist in einer eigenen Tabelle beschrieben.
0xf3	set_speed	Wiederholungsrate und Verzögerung setzen durch ein zweites Byte, dass nach dem Befehlsbyte folgt. Der Aufbau ist in einer eigenen Tabelle beschrieben.

Die folgende Tabelle zeigt den Aufbau des Steuerbytes von set_led zum Setzen der Tastatur-LEDs. MSB bedeutet hierbei *most significant bit* (entspricht also 0x80 in Hexadezimal-Darstellung), LSB *least significant bit* (also 0x01).

MSB							LSB
Always 0	Always 0	Always 0	Always 0	Always 0	Caps Lock	Num Lock	Scroll Lock

Der Aufbau des Konfigurationsbyte von set_speed ist in diesen zwei Tabellen beschrieben. Die Wiederholungsrate wird durch die Bits 0-4 spezifiziert, die Verzögerung durch Bit 5 und 6.

Bits 0-4 (hex)	Wiederholungsrate (Zeichen pro Sekunde)
0x00	30
0x02	25
0x04	20
0x08	15
0x0c	10
0x10	7
0x14	5

Bits 5 und 6 (hex)	Verzögerung (in Sekunden)
0x00	0.25
0x01	0.5
0x02	0.75
0x03	1.0

Weiterführende Informationen

Unter anderem hier: http://homepages.cwi.nl/~aeb/linux/kbd/scancodes-8.html