Fachbereich Computertechnik

I B M - P S / 2

T a s t a t u r T r e i b e r

Pascal Näf, Fabian Heusser
Dozent: Alex Klapproth

HTA LUZERN
Technikumstrasse 21
CH—6048 Horw

www.w3p.ch/ps2/; www.planetnaef.ch
Projekt-Ref #: 11

History

Nr	Datum	Name	Beschreibung
1	2002-06-18	hef	initial Version
2	2002-06-24	hef	referenzen hinzugefügt
3	2002-06-24	nap	Dokument redigiert
4	2002-06-25	hef	Abschnitt 4 x86 Architektur

Status

approved final

Inhalt

1 Distribution

Zu unserem Projekt gehören die unter 1.1 aufgelisteten Produkte. Sollten Sie eines dieser Dokumente oder Sourcecode nicht erhalten haben, können sie diese von `www.w3p.ch/ps2/` beziehen.

1.1 Dokumentation

main.pdf	Dies ist die Haupdokumentation, welche auf die andere Dokumente verweist und eine Zusammenfassung enthält.
srs.pdf	Enthält die Spezifikation des Tastaturtreibers, erklärt Funktionen und gibt Auskunft über die Schnittstellen.
usergide.pdf	Eine ausführliche Benutzeranleitung, die den Benutzer unterstütz den Tastatur-Treiber an den Medusa-Trainer anzuschliessen.

1.2 Source Code

ps2drv.h	Modul, welches die Funktionen exportiert.
ps2drv.c	Implementation des Tatstaturtreibers.
ps2demo.c	Test- und Demonstrationsmodul.
ps2.zip	Einsatzbereites Metrowerksprojekt.

2 Zusammenfassung

Bei vielen 68HC11 Programmen bzw. Projekten möchte man, dass das Programm einfach vom Benutzer gesteuert werden kann. Diese Aufgabe haben wir in Angriff genommen und das serielle Protokoll der Tastatur für den 68HC11 implementiert.
So ermöglicht unser Treiber eine handelsübliche Computertastatur ohne grossen Aufwand an den 68HC11 anzuschliessen.
Dabei brilliert die Lösung nicht nur durch eine einfache und pinsparende Anbindung, sondern auch durch die Verwendung von Standardkomponenten. Es braucht keine spezielle Hardware für die Anbindung lediglich eine Buchse und zwei Pull-Up Widerstände. Auch können AT (5 Pin DIN Stecker) und PS/2 Tastaturen (6 Pin Mini-DIN Stecker) gleichermassen angeschlossen werden.
Das Projekt verlief am Anfang etwas stockend, da wir auf den Baustein 8042 der x86 Architektur setzten, welcher uns die Kommunikation abgenommen und eine komfortable, interruptgesteuerte und gepufferte Implementation erlaubt hätte. Dieser wird jedoch nicht mehr hergestellt und wir mussten unser Projektziel neu auf eine hardwarenähere Implementation umformulieren. Dies hatte als grösste Folge, dass wir nur noch ein Bruchteil der Funktionen realisieren konnten, die wir geplant hatten
Der nun erstellte PS/2 Tastaturtreiber kommuniziert erfolgreich mit der Tastatur in beide Richtungen. Einerseits stellt er Low-Level-Funktionen zur Verfügung, welche Zeichen und Nachrichten von der Tastatur lesen und an die Tastatur senden. Anderseits stellt der Treiber Funktionen zur Verfügung,

die Tasten einlesen und deren Zeichen als ASCII-Code zurückliefern oder die Status-LEDs der Tastatur setzten.
Der Treiber wird durch ein Demo ergänzt, das mit wenigen Zeilen Code, Zeichen von der Tastatur liest und sie auf dem Display ausgibt. Dabei funktionieren die Tasten a-z, 0-9, Backspace, Space, Enter, Capslock, Scrolllock, Numlock erwartungsgemäss. Zusätzlich kann mit F12 ein Reset durchgeführt werden. Alle anderen Tasten werden ignoriert.
Die Anwendungsgebiete dieses Tastaturtreibers sind vielfältig. Vom einfachen Projekt, um die in der Einleitung erwähnte Problematik elegant zu lösen, bis zum Erleichtern der Titeleingabe eines MP3-Players, es sind keine Grenzen gesetzt. Viele Barcode-Leser werden auch an der Tastatur-schnittstelle angeschlossen und können somit auch mit unserem Treiber betrieben werden. Würde man ein 68CH11 gepaart mit Ethernetanbindung, einem Display und Tastatur verwenden, wäre die Implementation eines Mini- Telnet-Terminal möglich.
In die andere Richtung erleichtert und ermutigt unser Projekt, Geräte für die PS/2 Schnittstelle zu entwerfen, wie zum Beispiel ein Stift der gedruckter Text erkennt und über die Tastaturschnittstelle dem Computer sendet. Dabei ist ein Vorteil der Tastaturschnittstelle, dass diese bereits eine Speisespannung von 5V und mindestens 300mA zur Verfügung stellt.
Für Projekte, welche einen Tastatur-Treiber ähnlich dem Funktionsumfang und Ressourcenverbrauch von PC-Betriebsystemen benötigen, müsste man jedoch mit dem 68HC11 und unserem Tastaturtreiber einen 8042- Baustein emulieren. So würde dann die Kommunikation mit der Tastatur vom Hauptprozessor getrennt und die Übertragung zum Hauptprozessor würde Parallel und Interrupt gesteuert vor sich gehen.

3 Übersicht Projekt Dokumentation

3.1 Software Requirements Specification

Diese Dokument macht Aussagen über die Funktionalität unseres Treibers. Im Abschnitt 2 werden diese aus der Sicht des Benutzers dargestellt und im Abschnitt 3 aus der Sicht des Entwicklers. Hier ein Auszug aus dem Inhaltsverzeichnis:

1. Einführung
   - 1.1 Absicht
   - 1.2 Umfang
   - 1.3 Definitionen, Begriffe und Abkürzungen
   - 1.4 Verweise
   - 1.5 Überblick
2. Allgemeine Beschreibung (C-Requirements)
   - 2.1 Produkt Perspektive
   - 2.2 Schnittstellen
   - 2.3 Benutzer Charakteristiken
   - 2.4 Vorraussetzungen
   - 2.5 Verteilung von Anforderungen
3. Konkrete Anforderungen (D-Requirements)
   - 3.1 Äussere Schnittstellen
   - 3.2 Interne Schnittstellen
   - 3.3 Performance Anforderungen
   - 3.4 Vorraussetzungen Design
   - 3.5 Software System Eigenschaften

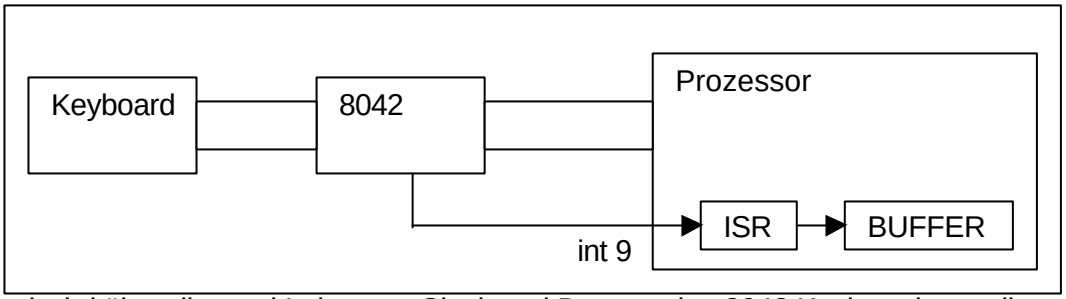
3.2 Benutzeranleitung

Diese Anleitung unterstützt den Benutzer beim Anschlissen der Tastatur, beim Einbinden des Treibers und beim Gebrauch des Treiberdemos.
Hier ein Auszug aus dem Inhaltsverzeichnis:

4 Intel x86 Keyboard Architektur

In der Zusammenfassung wurde die Intel x86 Architektur in Bezug auf die Keyboardanbindung mit dem 8042 Baustein erwähnt. Dieser ist leider nicht mehr als einzelner Baustein lieferbar, jedoch in jedem Chipset vorhanden und Rückwärtskompatibel. Hier möchten wir einen Überblick über diese Architektur geben und die Vor- bzw. Nachteile betrachten. Weiterführende Informationen betreffend dieser Architektur und dessen Ansteuerung findet man in „The AT-PS/2 Keyboard Interface“ auf http://govschl.ndsu.nodak.edu/~achapwes/PICmicro/keyboard/atkeyboard.html

4.1 Blockdiagramm



Das Keyboard wird über die zwei Leitungen Clock und Data an den 8042 Keyboardcontroller angeschlossen. Dieser übernimmt die Kommunikation über das Serielle Protokoll der Tastatur und ist über die Ports 0x60 und 0x64 an den Prozessor angeschlossen. Wird ein Zeichen von der Tastatur gelesen, löst dieser über das IRQ- System einen Interrupt 0x09 aus.
Die Interrupt Service Routine liest nun der Scancode der gedrückten Taste, decodiert ihn und Speichert diesen in einem Ringbuffer. In DOS wird typischerweise ein etwa 30 Zeichen grosser Ringbuffer verwendet, welcher sich im Speicherbereich des BIOS befindet. Dieser kann von der Anwendung bei Bedarf gelesen werden.

4.2 Vorteile

Der grösste Vorteil dieser Architektur ist sicher, dass der Prozessor vom seriellen Protokoll entlastet wird. Dies spart viel Zeit, wenn man bedenkt, dass das Keyboard mit einer Geschwindigkeit von 10 – 20 kHz zwischen 0,5 und 1 ms Übertragungszeit für ein Zeichen benötigt. Auch Timeouts, wie sie bei einer fehlerhaften Kommunikation auftreten können, kann der Keyboardcontroller elegant verarbeiten ohne, dass dabei den Prozessor zu belasten.
Der 2. Vorteil gewinnt man durch die Abstrahierung mit dem Interrupt. So wird der Prozessor unterbrochen, sobald ein Zeichen eingelesen wurde. Dieses kann er nun in einer übersichtlichen Art decodieren. So hält die ISR beispielsweise die Statusbits der Control, Alt und weiteren Funktionstasten, da uns die Tastatur nur mitteilt wann diese gedrückt und wann diese losgelassen werden.

4.3 Nachteile

Wie jede Architektur, hat auch diese Nachteile. Im falle der Keyboardanbindung an den 68HC11 sogar entscheidende. So ist diese Architektur nur mit einem erheblichen Mehraufwand, Hard- wie auch Software mässig, zu realisieren.
Diese x86 Architektur benötig am Prozessor einiges mehr Pins als die direkte Lösung. Diese fallen jedoch weniger ins Gewicht, wenn eine Busankopplung mit mehreren Bausteinen realisiert werden kann.
Weiter benötigt diese Architektur eine Interrupt- Leitung. Da diese nicht unbegrenzt vorhanden sind kann es auch hier zu Engpässen führen.

4.4 Schlussfolgerung

Soll eine Tastatur eingesetzt werden und dabei der volle Funktionsumfang mit minimalen Zeitressourcen genutzt werden, kommt man um eine, der x86er ähnliche Architektur, nicht herum. Diese bedeutet jedoch einen erheblichen Mehraufwand, welcher mit Komfort und geringerem Ressourcenverbrauch gerechtfertigt wird.
Die Entscheidung ob die Tastatur direkt oder über einen Keyboardcontroller an den Prozessor angeschlossen werden soll, muss so für jedes Projekt individuell getroffen und die Lösung entsprechend den Anforderungen angepasst werden.

5 Referenzen

Folgend unserer Referenzen nach derer Relevanz sortiert. Weiterer Links zum Thema Keyboard finden Sie auch auf unserer Homepage www.w3p.ch/ps2

- PS/2 Mouse/Keyboard Protocol
  Adam Chapweske
  http://panda.cs.ndsu.nodak.edu/~achapwes/PICmicro/PS2/ps2.htm

- Interfacing the AT keyboard.
  Craig Peacock
  http://www.beyondlogic.org/keyboard/keybrd.htm
  Achtung: Das Diagram, welches die Host to Keyboard Kommunikation darstellt ist falsch. Das Keyboard liest im Gegensatz zur Darstellung bei der steigenden Flanke.

- PC Schnittstellen
  Micheael Thieser
  ISBN – 3-7723-8092-1