

Aktiver PS/2 Tastatur Umschalter

von Klaus2m5

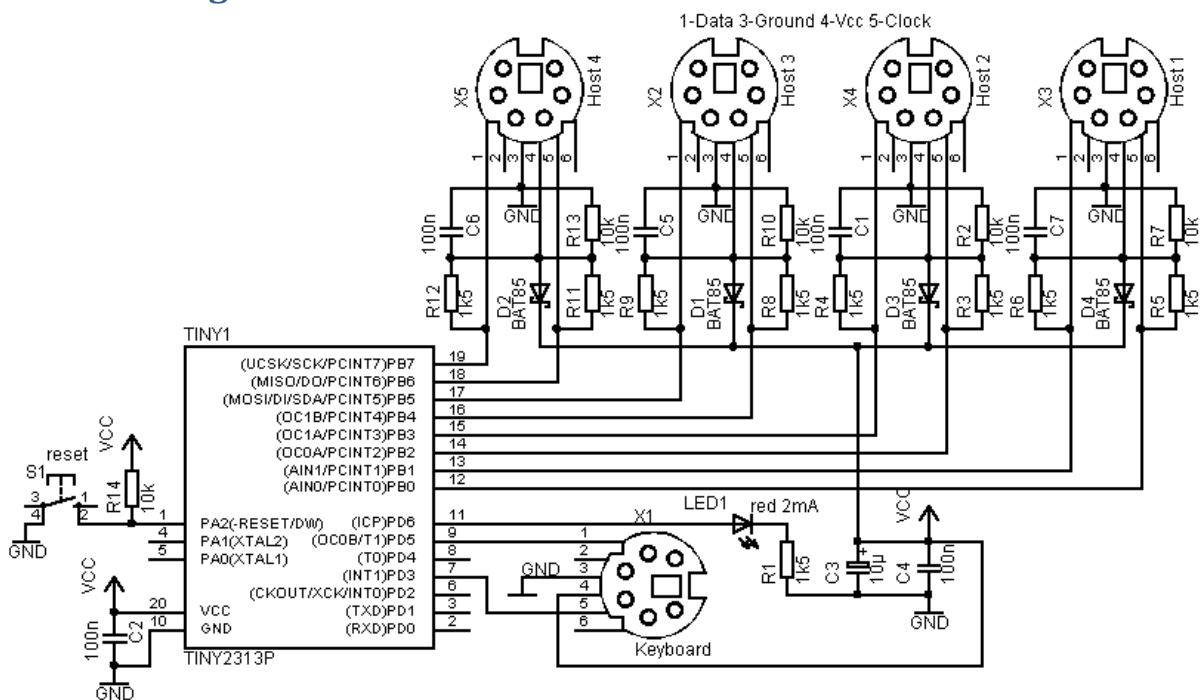
Ein in Assembler programmierter AVR ATTINY2313 erlaubt den Anschluss einer PS/2 Tastatur an bis zu 4 PCs.

An inaktiven PC Ports wird die Tastatur emuliert und die Parameter der Tastatur (typematic, Lock LEDs) gespeichert. Das Umschalten der Tastatur erfolgt über Hotkey (Shift Lock). LEDs und typematic rate werden dabei entsprechend des ausgewählten PCs an die Tastatur übertragen. Die Anzeige des aktiven PCs erfolgt durch eine gepulste LED (PC = Anzahl Pulse).

Einschränkungen

Es werden nur Tastaturen mit Scancode Set 2 unterstützt (XT Tastatur). Soweit bekannt, verwenden alle aktuellen PCs, Betriebssysteme und PS/2 Tastaturen diesen Standard.

Die Schaltung



Stromversorgung

Die Schaltung wird durch die angeschlossenen PCs über Schottky Dioden versorgt. Schaltung und Tastatur sind versorgt, falls einer der angeschlossenen PCs eingeschaltet ist oder die Tastatur Schnittstelle durch die Standby Spannung des PC Netzteils versorgt wird.

PC Ports

Jeweils 2 Pins des Port B bilden die Clock und Data Anschlüsse für einen PC. Das benutzte PS/2 Protokoll erfordert eine Open Collector Ansteuerung mit Pullup Widerständen (1,5k) ähnlich I²C. Ursprünglich hatte ich 10k als Pullup vorgesehen, mit 1,5m Kabel traten aber Übertragungsfehler auf und das Signal war deutlich sichtbar beeinträchtigt (Scope).

Um erkennen zu können, ob ein PC auch tatsächlich eingeschaltet ist, erfolgt die Versorgung der Pullups direkt aus der PC Spannung. Ein Pulldown (10k) nebst Block Kondensator (100n) pro PC sorgt für definierten low Pegel, falls kein PC angeschlossen ist.

Tastatur Port

Die Tastatur ist an Port D3 und D5 angeschlossen. Die Tastatur Clock ist dabei mit INT1 verbunden, damit die Data Bits mit der fallenden Flanke des Clock Signals ausgewertet werden können.

Status LED

Die Status LED wird vom Port D6 angesteuert und signalisiert den im Augenblick aktiven PC und andere Betriebszustände.

Reset

Der Reset Taster ist noch ein Überbleibsel aus der Debug Zeit und nicht zwingend erforderlich. Damals war es äußerst lästig, erst die Stromversorgung von allen PCs unterbrechen zu müssen, um einen Reset zu erreichen.

Freie Ports

Einige Ports sind noch frei und wurden während der Entwicklung benutzt oder sind für zusätzliche Funktionen vorgesehen.

- RS232 (Port D0:1) – zum Überwachen der Tastatur Scancodes und der Emulation. Gleichzeitig ist ein 7,372 MHz Quarz an Port A0:1 erforderlich. Der Schalter „debug“ muss beim Assemblieren der Firmware gesetzt sein
- ISP (MISO, MOSI, SCK, Reset) – falls eigene Experimente geplant sind. Während des Programmiervorgangs dürfen PC3 & PC4 nicht verbunden sein.
- BCD codierte Ausgabe des aktiven PC (Port A0:1) – zur Erweiterung durch einen externen Maus oder VGA Umschalter. Der Schalter „hst_sel_out“ muss beim Assemblieren der Firmware gesetzt sein.
- Maus (Port D2,4) – Erweiterung um einen zusätzlichen Maus Eingang. Zurzeit ohne Funktion. Die Erweiterung auf eine Maus Umschaltung ist von mir nicht geplant und ist wahrscheinlich nur auf einem größeren AVR sinnvoll.

Aufbau

Ein Layout für eine gedruckte Schaltung existiert nicht. Die Schaltung lässt sich jedoch leicht auf einer Lochrasterplatine aufbauen.

Die 6-poligen Mini-DIN Buchsen zu den PC Tastatur Ports können durch in der Mitte auseinander geschnittene, und direkt mit der Platine verlötete PS/2 Stecker/Stecker Kabel ersetzt werden.

Wenn der Schalter für weniger PCs aufgebaut wird, kann die Hardware für diese PCs entfallen. Es wird jedoch empfohlen, die freien Pins des Port B mit 10k Pulldowns zu bestücken.

Für eigene Experimente sollte man noch eine externe Stromversorgung, Probe Pins fürs Scope, einen RS232 Anschluss nebst Quarz und einen ISP Stecker verbauen (nicht Teil des Schaltplans).

Die Firmware

Das zum Download angebotene ZIP File enthält sowohl den Sourcecode (Assembler) als auch ein HEX File zum flashen des AVR ohne Debug-Funktion und BCD-Ausgabe an Port A. Benötigte Fuses:

- 8MHz internal RC Osc mit Startup 14 CK + 65ms (voreingestellt)
- BOD enabled – 2,7V
- CKDIV8 disabled !!!!!!!
- Finger weg vom Rest der Fuses!

Status

Der momentane Betriebszustand wird durch eine LED dargestellt. Wenn kein besonderer Betriebszustand vorliegt, Zeigt die Anzahl der Lichtimpulse den aktiven PC an. Wenn nach dem Drücken einer Taste die Kommunikation zum aktiven PC nicht möglich ist, z.B. keine Stromversorgung von diesem PC, dann wird dies durch 2 Sekunden 8Hz Blinken dargestellt.

Steuerung

Die Umschaltung der Tastatur erfolgt durch Klammergriff. Die Shift-Lock/Caps-Lock Taste dient dabei als Hotkey. Wird der Hotkey betätigt, dann leuchtet die LED dauernd bis diese Taste wieder losgelassen wird. Die Zifferntasten 1-4 im Alpha Block (nicht der Nummern Block) schalten auf den entsprechenden PC. Selbstverständlich kann man auch noch Caps-Lock erreichen, indem man die [^] Taste drückt ([~] bei englischer Tastatur). Die Tastenkombination kann im Sourcecode frei gewählt werden. Ein Beispiel ist im Source enthalten. Dabei werden die Tasten F1-F4 zum Umschalter, Caps-Lock über [Esc] erreicht.

Das Umschalten der Tastatur auf einen PC ohne Versorgungsspannung oder mit dauerhaft auf low liegendem Data- oder Clock-Signal wird unterdrückt und mit 8Hz Blinken angezeigt.

Initialisierung

Bei einem Reset durch Einschalten (auch Brown Out) wird der erste PC eingestellt, der eine Versorgungsspannung zur Verfügung stellt. Die Initialisierung der Tastatur wird dann dem PC an diesem Port überlassen.

Wird der Reset durch den Reset Pin ausgelöst (durch drücken der Reset Taste), dann wird die Tastatur durch den AVR initialisiert und die bisherigen Einstellungen wieder hergestellt.

Timer und Interrupts

Alle internen Funktionen des Tastatur Umschalters benutzen die Timer des AVR. Beide Timer werden mit einem Prescaler von 8 initialisiert, so dass diese mit einer Auflösung von 1µs arbeiten ($8\text{MHz}/8 = 1\text{MHz} = 1\mu\text{s}$). Natürlich eignet sich der 16 Bit Timer in diesem Falle für längere Zeiträume (65ms) und der 8 Bit Timer für eher kurze Zeiträume (256 µs).

So wird die LED mit Hilfe des Timer 1 Overflow Interrupts gesteuert. Dieser Interrupt wird enabled ausgeführt, da er nicht zu den höchsten Prioritäten des Umschalters gehört.

Die Kommunikation mit der Tastatur wird durch das Clock-Signal der Tastatur angestoßen. Diese erzeugt über Portpin D3 mit der fallenden Flanke einen externen Interrupt über den INT1-Vektor. Laut Spezifikation generiert die Tastatur eine Clock mit maximal 16,6kHz. Die Daten sind während der low Phase der Clock gültig, entsprechend 30µs des Clock-Zyklus von 60µs. Daten müssen also nach

240 AVR Clocks erfasst sein (inklusive aller vorausgehender Interrupts). Ein 11 Bit Datenframe von der Tastatur erreicht den AVR in minimal 660µs, maximal in 1,1ms (minimum Clock 10kHz). Um verlorene Clocks sicher erkennen zu können, wird am Anfang eines Frames der Timer 1 Compare A auf Timer 1 + 2ms gesetzt. Läuft der Timer Compare vor Ende des Frames ab, so wird durch den resultierenden Interrupt ein Fehler Bit im Status Register gesetzt (0x40). Dies geschieht auch bei Start- und Stop-Bits mit falschem Pegel und bei falschem Parity (0x80).

Die Kommunikation mit den PC Tastatureingängen erfolgt mit Hilfe des Timer 0 Compare A. Für jede Clock-Phase wird der Compare auf Timer 0 + 30µs gesetzt (240 AVR Clocks + die eventuell von INT1 benötigten AVR Clocks). Der Clock Zyklus kann durch verzögerte Auslösung des Compare Interrupts bis auf 50µs verlängert werden, er wird aber immer mindestens 30 µs lang sein (10kHz-16,6kHz). Mit dem Versenden eines Frames an einen PC wird für diesen PC ein Software Timer gestartet, so dass das nächste Frame für diesen PC mindestens 3,5ms warten muss. Dadurch bleibt genügend Zeit für die Kommunikation mit den anderen PCs, und der PC kann mit Hilfe seines 2ms Timeouts mit Sicherheit verlorene Clocks erkennen.

Hauptprogramm

Im Hauptprogramm werden permanent die durch die Tastatur- und PC-Kommunikation gesetzten Status Register überprüft („looking for work“). Wenn die Statusregister die maximale Anzahl von Interrupts erreicht haben, war die Übertragung erfolgreich. Wenn zum Beispiel der Tastatur Interrupt 11 erreicht hat, ist ein Frame komplett empfangen, und kann an den PC weitergegeben werden. Wenn ein Frame nicht vollständig übertragen werden konnte, wird dies durch setzen von Bit 7 oder Bit 6 indiziert. In diesem Fall führt das Hauptprogramm die Fehlerbehandlung durch (Resend, Resend Request).

Ein Kommando vom aktiven PC wird an die Tastatur weitergegeben. Bei den inaktiven PCs sorgt eine Emulation für die Antworten. In jedem Falle werden alle wichtigen Parameter in einem Array gespeichert. Aus diesem Array können beim Wechsel des aktiven PCs die entsprechenden Parameter an die Tastatur gesendet werden.

Die Kommunikation mit einem PC erfordert oft das Senden mehrerer Frames hintereinander und ist deshalb durch einen Sequencer realisiert. Das Hauptprogramm kann dadurch in den Wartezeiten zwischen den Frames auch mit anderen PCs kommunizieren, obwohl nur eine Task existiert. Auch für die Tastatur gibt es einen Sequencer, der nach dem Umschalten des aktiven PCs die Tastatur enabled und mit den Parametern für diesen PC lädt.

Quellen

Zu dieser Dokumentation gehören auch:

- keyboard_switch.asm
- keyboard_switch.hex

Informationen im Web

[PS/2 Keyboard Protocol](#)
[Interfacing the PCs keyboard](#)