3. Übung aus Treiberentwicklung SS2015, Abgabetermin: 20.5.2015

Ausgearbeitet von: Gruppe:

Aufgabe 1: I²C-Interface über die serielle Schnittstelle

Über die serielle Schnittstelle sollen I²C-Devices angeschlossen werden. Der I²C-Bus ist eine Zweidraht Busverbindung mit folgenden Leitungen:

- **SCL** - die Taktleitung
- **SDA** - die Datenleitung

Im PC wird ein UART-Baustein des Typs 8250 eingesetzt. Informationen über verfügbare Register finden Sie im Internet. Diese Register sollen direkt vom Treiber aus programmiert werden. Dazu bietet die HAL unter anderem folgende Funktionen an:

- `WRITE_PORT_UCHAR`
- `READ_PORT_UCHAR`

Die Basisadresse für den COM1 ist 0x3F8.

Über folgende Steuerleitungen der RS232-Schnittstelle sollen SCL bzw. SDA beschrieben bzw. gelesen werden (siehe auch den Schaltplan für die Testplatine):

- Schreiben (Modem Control Register - MCR)
  DTR cWriteDataBit (Pin 4, 9 pin connector)
  RTS cWriteClockBit (Pin 7, 9 pin connector)
- Lesen (Modem Status Register - MSR)
  CTS cReadDataBit (Pin 8, 9 pin connector)
  DSR cReadClockBit (Pin 6, 9 pin connector)

Hilfe zu den verfügbaren Kernel Funktionen kann z.B. in der Datei `kmarch.chm` (im WINDDK) gefunden werden (`WRITE_PORT_UCHAR` findet man unter Reference ⇒ Driver Support Routines ⇒ Hardware Abstraction Layer Routines)

Zum Testen der neuen Treiber-Funktionalität soll eine Python-Testbench entwickelt werden, die während der Entwicklung zum Testen verwendet werden kann.

Achten Sie beim Erstellen der Testbench auf einen guten Programmierstil. Insbesondere sollen Code-Duplizierungen vermieden werden!

Vorgehensweise:

- Zu Beginn soll das Lesen und das Schreiben von SCL/SDA implementiert werden. Um das zu Verifizieren messen Sie einfach den Spannungspegel an diesen beiden Pins.
- Danach soll aufbauend auf diese Funktionalität das I²C-Protokoll realisiert werden.
- Gerade am Anfang der Entwicklung macht es Sinn, den Großteil der Funktionalität im User Mode zu haben, dadurch verkürzt sich der Debug-Zyklus. Sobald die Routinen stabil laufen, können diese in den Kernel Mode verlagert werden, falls es sich um Teile handelt, die Laufzeit optimiert werden sollen.
- **Achtung:** Das Powermanagement des Rechners verlangt, dass die serielle Schnittstelle geöffnet ist, damit ein `WRITE_PORT_UCHAR` eine Auswirkung für die Hardware hat. Das kann durch das Öffnen der seriellen Schnittstelle von Python aus (Modul pyserial) oder durch Öffnen eines Terminalprogrammes (wie z.B. Hyper Terminal) geschehen.

Ziel dieser Aufgabe ist es, die I²C-Echtzeituhr auf der Versuchsplatine ansprechen zu können.

Folgende Hardwareelemente werden für die Übung benötigt:

- I²C-Echtzeituhr-Platine (erhältlich bei Armin Brandl)
- passendes Netzgerät (erhältlich bei Armin Brandl)
- Ein 1:1-verbundenes RS232-Kabel