Versuch 2, LPC11U24 - GPIO

Die LED auf dem Piggyback (weiße Zusatzplatine) soll mit Hilfe der BOOT-Taste gesteuert werden. Solange die Taste gedrückt ist, soll die LED leuchten. Ist die Taste nicht gedrückt, dann bleibt die LED aus.

1 Vorbereitung

Legen Sie einen neuen, leeren Workspace an. Importieren Sie dann aus dem Archiv v2.zip die beiden Projekte lpc_chip_11uxx_lib und v2.
Falls Sie die Vorbereitung ohne das Programm MCUXpresso machen, dann extrahieren Sie die beiden Projekte aus dem Archiv und sehen sich die im Folgenden erwähnten Dateien in einem Editor an.
Das Projekt *lpc_chip_11ux_lib* ist eine stark reduzierte Version der Bibliothek, die der Hersteller für alle µC der Serie LPC11Uxx mitliefert. Das Projekt *v2* enthält ein C-Skelett, das diese Bibliothek benutzt.

Im ersten Versuch wurde die LED auf dem Board (PIO0_7) benutzt. Dazu haben Sie mit Hilfe von Zeigern direkt auf die beteiligten Peripherieregister zugegriffen.
Diese Methode ist die universellste – man kann damit alles an Funktionen ausnutzen, was der µC überhaupt bietet. Sie ist aber zugleich die mühsamste und am wenigsten portable. Daher bieten heute alle µC-Hersteller mindestens einen C-Header an, in dem schon einmal die Registeradressen als auch typische Funktionen mit symbolischen Namen definiert sind.

Für den LPC11U24 ist in der Bibliothek *lpc_chip_11uxx_lib* der Header *chip.h* vorhanden. Dort sind für die beiden Module (IOCON, GPIO), die benötigt werden, die Basisadressen schon mit je einem Symbol vordefiniert.

Wie heißt das Symbol, das der Basisadresse für die GPIO-Register entspricht und welche Adresse stellt es dar?

Symbolname	Adresse (hexadezimal)

Sehen Sie im User Manual (UM) nach, ob Sie eine Tabelle mit dieser Adresse als Basis finden können. Im UM sind die Adressen auch hexadezimal angegeben, aber 8 Stellen werden in zwei Blöcken á 4 Bit geschrieben. Wenn Sie also die Adresse 0x12345678 finden wollen, dann suchen Sie nach 0x1234 5678.
Um welche Tabelle handelt es sich?

Tabelle für die GPIO-Register	

Die Register in einem Block werden in C mit Hilfe einer Struktur nachgebildet. Sehen Sie sich den Header *iocon.h* an. Wie heißt der Datentyp, der hier für die Registerstruktur definiert ist?

Typ für die Struktur des IOCON-Blocks	

Gehen Sie jetzt zurück zu *chip.h*. Dort finden Sie ein Symbol, das einen **Zeiger** mit der richtigen Basisadresse für das IOCON-Modul darstellt. Wie heißt das Symbol?

Name des Zeigers auf den IOCON-Block	

2 Einstellen der Pins für die LED

2.1 Zusammenstellen der benötigten Information

Für jeden Pin hat der Mikroprozessor ein eigenes Register, in dem die Funktionen des Pins als auch diejenigen elektrischen Eigenschaften, die eingestellt werden können, aufgeführt sind. Sämtliche Registerbeschreibungen finden sich im User Manual.
Welche Tabelle gilt für den Pin PIO0_6 (LED auf dem Piggyback?)

Tabelle für PIO0_6	

Bei diesem Mikrocontroller sind pro Pin 8 verschiedene Funktionen mit den Funktionsnummern 0-7 einstellbar. Zu einem bestimmten Zeitpunkt kann nur eine dieser Funktionen aktiv sein. Es kann sein, dass nicht alle der acht möglichen Funktionen tatsächlich eine Bedeutung haben. Bedeutungslose Funktionsnummern sind meist als *reserved* bezeichnet. Die Funktion „GPIO“ wird im UM mit „PIO“ leicht abgekürzt.
Welche Funktionsnummer müssen Sie für den PIO0_6 wählen und wie heißt das zugehörige Symbol für diese Nummer?

Funktionsnummer	Symbol aus *iocon.h*

Da der Anschluss eine LED ansteuern soll, muss er selber die passende Spannung erzeugen – entweder 0V (L-Pegel) oder 3.3V (H-Pegel). Daher wäre es sinnlos, einen Pullup- oder Pulldown-Widerstand einzuschalten.
Welche Bitkombination (2 Bits) wählen Sie für *MODE* und wie heißt das zugehörige Symbol für diesen Modus?

MODE	Symbol aus *iocon.h*

Zusatzinformation:
Die Eigenschaften *Hysterese* und *Inverter* haben nur für Eingangssignale eine Bedeutung. Man kann für einen Ausgang also prinzipiell einstellen, was man will. Empfehlenswert ist es, nicht benötigte Funktionen (hier die Hysterese) abzuschalten. Das kann (muss aber nicht) ein wenig Energie sparen.
Die Funktion *Open Drain* bewirkt, dass der Anschluss als Ausgang selbst keinen H-Pegel erzeugen kann. Wird das gefordert, dann bleibt er einfach inaktiv, d.h. er stellt weder Spannung noch Strom bereit. Den L-Pegel kann er dagegen wie üblich aktiv ausgeben.
Diese spezielle Eigenschaft wird bei keinem der vorgesehenen Versuche benötigt.

2.2 Umsetzung in einem Programm

Wenn alle benötigten Informationen vorhanden sind, dann kann man nach einer geeigneten C-Funktion in der Bibliothek suchen. Derartige Funktionen sind im Header deklariert und ggf. in einer weiteren Datei ausgeführt. Für den IOCON-Block ist das die Datei *iocon.c* im Verzeichnis *src* der Bibliothek.
Hier sind im Header zwei Funktionen deklariert, die aber beide dasselbe leisten – man kann mit ihnen die Eigenschaften eines Pins vollständig einstellen. Eine der beiden Funktionen ist also überflüssig. So etwas entsteht meist aus der Historie, z.B. der Zusammenlegung zweier Bibliotheken für verschiedene µC-Familien.

Welcher Unterschied besteht in der Anwendung der beiden Funktionen?
Tipp: Sehen Sie sich die Parameterliste **genau** an.

Unterschied der Funktionen *Chip_IOCON_PinMuxSet* und *Chip_IOCON_PinMux*

Bei dem Kommentar über der Funktionsdeklaration sehen Sie bei dem Parameter *mode* die folgende Beschreibung: *OR'ed values or type IOCON_**.
Damit ist gemeint, dass man mehrere Werte des Typs *IOCON_* (siehe die Symbolnamen) mit Hilfe des bitweisen Oder-Operators zu einem gemeinsamen Wert zusammenfassen soll. Möchte man die Eigenschaften *IOCON_FILT_DIS*, *IOCON_SFI2C_EN* und *IOCON_HYS_EN* gemeinsam wählen, dann schreibt man also: *IOCON_FILT_DIS* | *IOCON_SFI2C_EN* | *IOCON_HYS_EN*.
Jetzt können Sie die Funktion *Chip_IOCON_PinMux* sinnvoll in einem C-Programm anwenden. Tragen Sie für die der fehlenden Parameter die tatsächlichen Werte für Anwendung ein:

```
Chip_IOCON_PinMux(          , 0, 6,          ,          );
```

3 Einstellen des der Pins für die BOOT-Taste

Die Einstellung dieses Pins läuft genauso ab wie die für die LED. Abgesehen davon, dass es sich um einen anderen Anschluss handelt, handelt es sich hier um einen Eingang. Die Taste kann selber keinen H-Pegel erzeugen. Sammeln Sie zunächst wieder die benötigten Informationen.

Welche Tabelle gilt für den Pin (BOOT-Taste auf dem Piggyback?)

Tabelle für PIO0_1	

Welche Funktionsnummer müssen Sie wählen und wie heißt das zugehörige Symbol für diese Nummer?

Funktionsnummer	Symbol aus iocon.h

Tragen Sie alle weiteren elektrischen Eigenschaften für den Pin ein (Symbole IOCON_).

Tragen Sie für die fehlenden Parameter die tatsächlichen Werte für Anwendung ein:

```
Chip_IOCON_PinMux(          , , ,          ,          );
```

4 Nutzung der GPIO-Funktionen

Bisher haben Sie nur die elektrischen Eigenschaften der beiden Pins eingestellt sowie jeweils die Funktion GPIO ausgewählt. Jetzt müssen Sie zunächst (einmalig) die Richtung der GPIO einstellen und dann im Fall der Taste Werte einlesen bzw. im Fall der LED Werte ausgeben können. Auch dafür stehen passenden Symbole und Funktionen bereit.
Wie heißt das Symbol (in *chip.h*), das einen **Zeiger** auf die Basisadresse der GPIO-Register darstellt?

Name des Zeigers auf den GPIO-Block	

Wie heißt die Funktion, die den GPIO-Block einmalig initialisiert (in *gpio.h*)?

Funktion zum Initialisieren der GPIOs	

Welche Richtung müssen Sie für die GPIOs einstellen?

Richtung PIO0_6 (LED)	Richtung PIO _ (BOOT-Taste)

In *gpio.h* gibt es mehrere Funktionen, mit denen die Richtung eines GPIO eingestellt werden kann. Es ist Ihnen überlassen, mit welchen Funktionen Sie das Problem lösen. Dazu folgende Hinweise:

- Manche Funktionen sind spezialisierter als andere.
- Einige Funktionen beziehen sich auf einen einzigen GPIO, andere auf mehrere

Wählen Sie jetzt nach Ihrem Ermessen Funktionen aus und schreiben Sie je einen passenden Aufruf hin (wie bei *Chip_IOCON_PinMux*).

Zum Setzen eines Wertes bei einem Ausgang gibt es ebenfalls mehrere Möglichkeiten. Hier soll die Funktion *Chip_GPIO_SetPinState* verwendet werden. Schreiben Sie passenden Aufruf, wenn Sie an PIO0_6 einen H-Pegel ausgeben wollen.

Mit welcher Funktion können Sie den Wert an einem einzelnen GPIO abfragen?

5 Bedienung der LED über die Taste

Jetzt können Sie das Programmskelett in *v2.c* vervollständigen. Tragen Sie nach jedem Kommentar eine C-Funktion mit den passenden Aufrufparametern ein.

```
// Eigenschaften des Pins PIO0_1 einstellen


// Eigenschaften des Pins PIO0_6 einstellen


// GPIO Modul für die Benutzung einmalig initialisieren


// GPIO PIO0_1 auf Eingang stellen


// GPIO PIO0_6 auf Ausgang stellen


// Jetzt endlos Taste abfragen und LED ansteuern
while(1)
{
      // Variable, um den eingelesenen Wert zwischenzuspeichern
      bool v;

      // Einlesen des Wertes am GPIO PIO0_1 in die Variable v
      v=

      // und nun v am GPIO PIO0_6 ausgeben

}
```

Lernziele

- Umgang mit der Entwicklungsumgebung
- Einstellen eines Anschlusses des µC, Nutzung eines GPIO

Material zur Vorbereitung

- User Manual UM10462, Revision 5.3 für den µC
- Pinliste lpc11u_pins.pdf
- Projektarchiv v2.zip für MCUXpresso

Material zur Durchführung

- Basisplatine, Piggyback weiß
- Entwicklungswerkzeug (hier MCUXpresso Version 10.0.0)