Aufgabenblatt 5 GPIO

Sie können bei diesem Aufgabenblatt auf **freiwilliger** Basis Struktogramme erstellen und diese zu Bewertung vorlegen.

1 Fortschrittsanzeige

Implementieren Sie eine Fortschrittsanzeige, das einem Fortschrittsbalken auf den LEDs 0 bis 7 einmalig „hochzieht“ (Abbildung 1). Der Fortschritt soll dabei gut wahrnehmbar sein, also insgesamt einige wenige Sekunden andauern.Für die Verzögerung zwischen den LEDs verwenden Sie bitte ihre delay() Funktion.

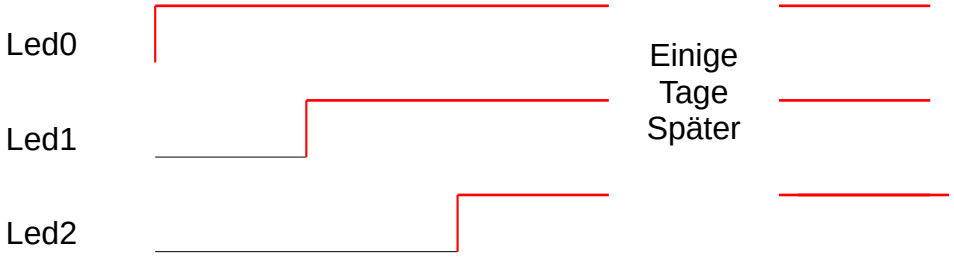

Abbildung 1: Fortschrittsanzeige

Um diese Aufgabe zu lösen benötigen Sie zwei I/O Register:

- IODIR1 hat die Adresse 0xE0028018
- IOSET1 hat die Adresse 0xE0028014

Die LEDs belegen Pins 16 bis 23 auf dem Port 1. Zuerst müssen die entsprechenden Pins mit dem ***IODIR*** Register in dem Port als Ausgang gestellt werden, da diese nach einem Reset des Mikrocontrollers immer als Eingänge geschaltet sind. Schreibt man dann eine 1 auf dem entsprechenden Pin in dem Port (über dem ***IOSET*** Register) so wird die entsprechende LED eingeschaltet (Abbildung 2).

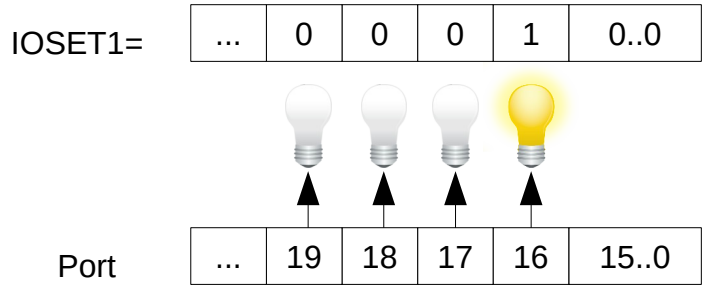

Abbildung 2: Ausgänge auf „1“ setzen

Bitte Beachten Sie, dass das das schreiben von „0“ auf dem ***IOSET*** Register keine Auswirkung auf den Status der Pins auf dem Port hat (Abbildung 3), da der ***IOSET*** mit dem eigentlichen Ausgang des Mikrocontrollers mit einer Binären Oder Funktion verknüpft ist.

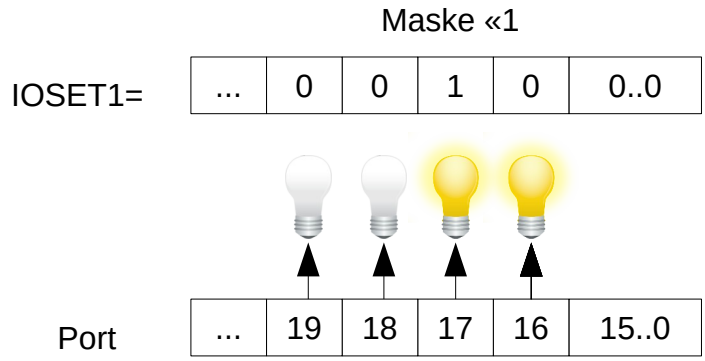

Abbildung 3: Weitere Ausgänge auf „1“ setzen

Tipps:

- Da Sie mit Funktionen arbeiten müssen, vergessen Sie bitte nicht den Stack zu Initialisieren, die korrekte Adresse des Stackpointers auf dem Board ist 0x40001000 .
- Innerhalb der LED-Bitmaske, können Sie die Umschaltung auf die folgende LED mit einem Schiebebefehl erreichen.

Skript:

3.2 Zugriff auf einem der Hardwareregister.
3.3 GPIO
3.3.1 IODIR – Richtung einstellen (Eingang / Ausgang)
3.3.2 IOSET – Ausgänge setzen
3.2.2 Definition der Registeradressen
3.2.5 I/O Einheit verwenden
3.4.5 Anwendungsfall – LEDs ein und ausschalten

2 Permanentes Lauflicht

Erweitern Sie bitte die Aufgabe 1 zu einem permanenten Lauflicht, indem diesmal zu einem gegebenen Zeitpunkt immer nur eine der LEDs in der aktiven Phase ist (Abbildung 5). Eine aktive Phase, für diese Aufgabe, wird als 2/3 ein und 1/3 aus definiert (Abbildung 4). S Das Programm soll in einer Endlosschleife ablaufen, so das nach der letzten LED wieder auf die erste LED umgeschaltet wird.

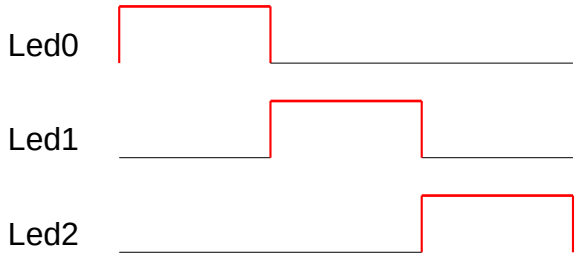

Abbildung 5: Leds bei 100% Phase

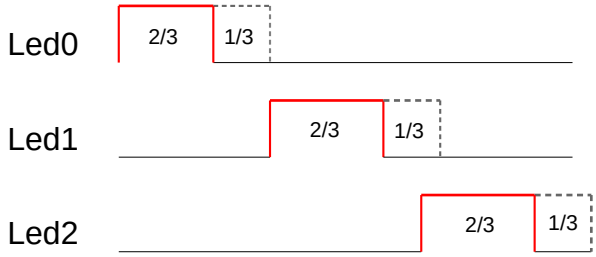

Abbildung 4: Leds bei 2/3 und 1/3 Phase

Zum ausschalten der LEDs benötigen sie einem zusätzliche Register, den IOCLR:

- IOCLR1 hat die Adresse 0xE002801C

Sollte eine LED auf dem Port „gelöscht“ werden, so ist es an der entsprechenden Stelle eine 1 auf dem ***IOCLR*** Register zu schreiben (Abbildung 6). Wie auch bei dem ***IOSET*** Register haben die 0 Werte in der Löschmaske keine Wirkung auf das Ergebnis der Operation da hier eine binäre ***und nicht*** Operation auf dem dem Port durchgeführt wird.

Skript:
3.3.3 IOCLR – Ausgänge löschen

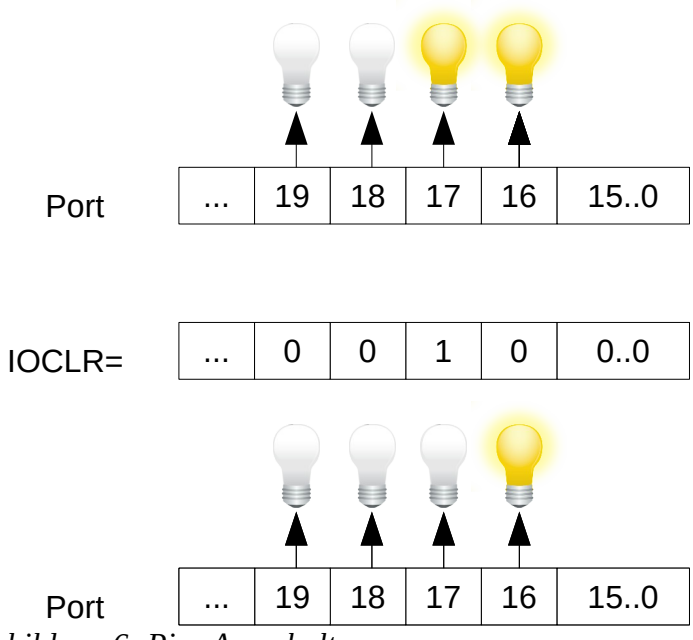

Abbildung 6: Pins Ausschalten

3 Ein- und Ausgabe über Taster und LEDs

3.1 Voraufgabe

Der folgende Programmausschnitt schaltet die beiden ersten LEDs (0 sowie 1) durch Betätigen des Tasters 0 um.

```
      mov r6, #1«16       // Load mask for the LED 0 in r6
      mov r5, #1«10       // Load mask for the button 0 in r5
      ldr r0, [r4]        // Load input values from IOPIN to register r0
      ands r0, r5, r0     // check if button 0 is pressed
      bne noled1          // branch if button is not pressed

      // button is pressed,
      str r6, [r2]        // switch pins defined in r9 on (IOSET1) (first LED on)
      mov r6, r6, lsl #1 // shift mask to second LED
      str r6, [r3]        // switch pins defined in r9 off (IOCLR1) (second LED off)
      b led_done          // brunch to end

      // button is not pressed
noled1:
      str r6, [r3]        // switch pins defined in r9 off (IOSET1) (first LED off)
      mov r6, r6, lsl #1 // shift mask to second LED
      str r6, [r2]        // switch pins defined in r9 on (IOSET1) (second LED on)
 led_done:                // End subrutine
```

Implementieren Sie ein Programm, das die vorgegebene Funktion verwendet um zwischen der LED 0 und 1 umzuschalten. Zum auslesen der Pins brauchen sie dem IOPIN Register:

- IOPIN0 hat die Adresse 0xE0028000

Schreiben Sie diese Funktion so um, das Sie für jeden Port einem Basisregister verwenden (IOPINx) und die anderen Register relativ über Indexe adressieren (Pseudocode : *„Relative Definition der Registeradressen“* sowie *„Konfiguration der I/O Einheit (relativ)“*). Somit sollten Sie für die Hardwareansteuerung nur noch zwei Basisregister (IOPIN0 sowie IOPIN1) sowie die relative Indexe zu den restlichen Registern eines Ports, definieren.

Beispiel zum kopieren und ergänzen:
.equ IOPIN0, 0xE0028000
.equ IOPIN1, ….

.equ IOSET, 0x04 // IOSET has an offset of 4
.equ IODIR, …. // IODIR has an offset of ...
.equ IOCLR, ….

3.1.1 Taster

Äquivalent zu den LEDs belegt jeder Taster auch einen I/O Pin allerdings diesmal auf dem Port 0. Der Status eines Tasters kann dann über den IOPIN Register ausgelesen werden. Damit ein Taster ausgelesen werden kann muss der entsprechende Pin in dem I/O Register als Eingang geschaltet werden (über IODIR umschaltbar). Da der vom IOPIN gelesene Wert Informationen über alle Pins des Registers enthält, müssen diese vor der Auswertung mit einer geeigneten Binären verknüpfung ausfiltern.

Die vier Drucktaster sind an die Pins P0.10 bis P0.13 angeschlossen. Nach einem Reset ist P0 automatisch als Eingang konfiguriert. Es ist also keine weitere Richtungsumschaltung notwendig. Zur Vereinfachung definieren wir Bitmasken für die den Tastern entsprechenden GPIO-Bits:

```
.equ BUTTON_0_bm, (1«10)          // Button 1 mask
.equ BUTTON_1_bm, (…)     // Button 2 mask
.equ BUTTON_2,_bm (…)     // Button 3 mask
.equ BUTTON_3_bm,  (…)    // Button 4 mask
```

3.2 Mehrfache Ein- und Ausgabe über Taster und LEDs

Generalisieren Sie den gegebenen Quelltext der Funktion so, dass jetzt die erste LED sowie der Button als Parameter an die Funktion übergeben werden. Erweitern Sie Ihr Programm so, das Sie alle 4 Buttons sowie alle LEDs auf folgende weise verwenden:

- Button 0 schaltet zwischen den LEDs 0 und 4
- Button 1 schaltet zwischen den LEDs 1 und 5
- Button 2 schaltet zwischen den LEDs 2 und 6
- Button 3 schaltet zwischen den LEDs 3 und 7

Hinweise & Tipps:

- Die Maske für alle LEDs können (***lese sollen***) Sie genauso zusammenbauen wie Sie es in dem ersten Aufgabenblatt gelernt haben.

Skript:

3.3.4 IOPIN – Eingänge lesen
3.4.6 Anwendungsfall – Taster auslesen