SPiC-Aufgabe #3: button

(14 Punkte, in Zweier-Gruppen)

In dieser Übungsaufgabe implementieren Sie einen Teil des `button` Moduls, welches innerhalb der `libspicboard` für das Einlesen der Taster zuständig ist.

Halten Sie sich dabei genau an die in `button.h` vorgegebene Schnittstelle und implementieren Sie diese in der Datei `button.c`. Zusätzlich sollen Sie in der Datei `test-button.c` eine Anwendung schreiben, die die Funktionalität des von Ihnen implementierten `button` Moduls testen soll.

Teilaufgabe a: Initialisierung (3 Punkte)

Implementieren Sie die modulinterne Initialisierungsfunktion `sb_button_init()`, die sich um die Hardwareinitialisierung kümmert.

Die beiden Pins `PD2` (`BUTTON0`) und `PD3` (`BUTTON1`) müssen als Eingang konfiguriert und die integrierte Pull-up Widerstände aktiviert werden. Stellen Sie sicher, dass die Hardwareinitialisierung nur bei dem ersten Aufruf einer Schnittstellenfunktion erfolgt.

Achten Sie bei der Implementierung der modulinternen Hilfsfunktionen auf korrekte Sichtbarkeit.

Teilaufgabe b: Callbacks deaktivieren (2 Punkte)

Der Einfachheit halber müssen Sie die Callback Funktionalität nicht implementieren. Die Schnittstellenfunktionen `sb_button_registerCallback()` und `sb_button_unregisterCallback()` sollen allerdings den Wert `-1` an den Aufrufenden zurückliefern, um ihn über die fehlende Funktionalität zu informieren.

Teilaufgabe c: Taster auslesen (5 Punkte)

Implementieren Sie nun die Funktion `sb_button_getState()`, die den aktuellen Zustand (`PRESSED`, `RELEASED`) eines Tasters (`BUTTON0`, `BUTTON1`) an den Aufrufenden zurück gibt. Achten Sie darauf, dass beide Taster `active-low` beschaltet sind. Das Auslesen der Taster soll wie folgt ablaufen:

- Der Taster `BUTTON0` ist in Hardware entprellt, Sie müssen also lediglich prüfen, ob am Pin `PD2` der jeweilige Pegel für die Werte `PRESSED` oder `RELEASED` anliegt.
- Für den Taster `BUTTON1` sollen Sie eine einfach Entprellroutine implementieren. Tasten Sie dafür den Pegel des Pins `PD3` 99 mal im Abstand von $5\mu s$ ab. Geben Sie anschließend den Wert zurück, der zum häufiger aufgetretenen Pegel gehört.
- Verwenden Sie die Funktion `_delay_us()` aus dem Header `<util/delay.h>`, um das aktive Warten zwischen den Abfragen zu realisieren.
- Achten Sie auf korrekte Rückgabewerte bei ungültigen Parametern. Entnehmen Sie dieser der `libspicboard`-Dokumentation.

Teilaufgabe d: Modultest (4 Punkte)

Implementieren Sie nun in der Datei `test-button.c` eine Anwendung, die alle Funktionen des Moduls aus Teilaufgaben a) – c) testet (Modultest). Der Modultest soll insbesondere folgende Eigenschaften haben:

- Fragen Sie jeden Rückgabewert der vom Modul bereitgestellten Funktionen mindestens einmal ab.
- Testen Sie gegebenfalls verschiedene Kombinationen von Übergabeparametern
- Für die Funktionalität der `sb_button_getState()` dürfen Sie in einer Endlosschleife für `BUTTON0` und `BUTTON1` eine LED leuchten lassen, solange der jeweilige Taster gedrückt ist.

Achten Sie darauf, dass Erfolgs- und Fehlerfälle unterschieden werden können.

Allgemeine Hinweise

- Da Sie die Übergabeparameter in den Funktionen `sb_button_registerCallback()` und `sb_button_unregisterCallback()` nicht benötigen, kann es sein, dass der Compiler die Warnung `-Wunused-parameter` anzeigt. Diese Warnung dürfen Sie ignorieren.
- Der Linker verwendet stets die „lokalen“ Funktionen, die in den lokalen Quelldateien definiert sind. Nur wenn eine Funktion dort nicht definiert ist, werden die Bibliotheken durchsucht. Um testweise statt der eigenen Taster-Funktionen in der Datei `button.c` die Taster-Funktionen der `libspicboard` zu verwenden, können Sie einfach Ihre eigene Implementierung auskommentieren, z. B. durch eine Zeile

  `#if 0`

  am Anfang der Datei und eine Zeile

  `#endif`

  am Ende der Datei. Aktivieren können Sie Ihre Implementierung dann wieder, indem Sie die `0` in eine `1` abändern.
- Begründen Sie die Verwendung von allen `volatile` Variablen. Wenn für mehrere Variablen die selbe Begründung gilt, dürfen Sie diese gemeinsam begründen.

Abgabezeitpunkt

alle Gruppen	23.11.2021	18:00:00 Uhr