Aufgabe 2

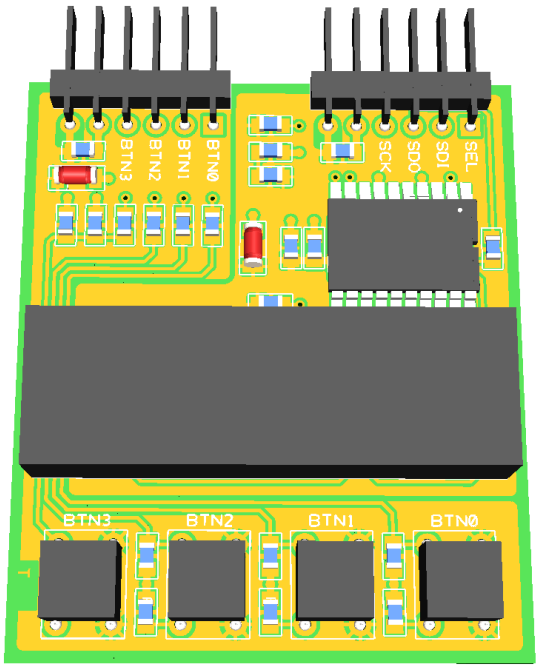
Lernziele: Machen Sie sich mit der Prinzip der seriellen, synchronen Datenübertragung mittels der SPI-Schnittstelle vertraut. In dieser Aufgabe sollen Sie lernen

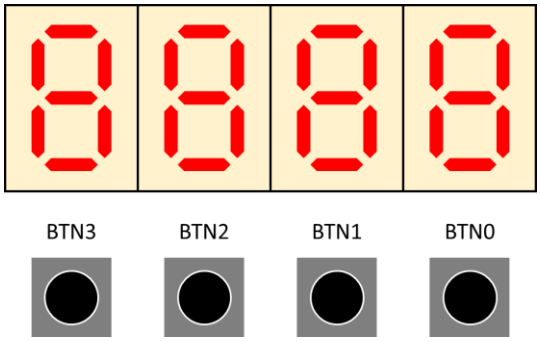
a) wie man Ports des MSP430 als SPI-Schnittstelle konfiguriert,
b) wie man über die SPI-Schnittstelle auf eine externe SPI-Komponente zugreift,
c) wie man auf mehrfache Ereignisse in der Funktion main() reagiert und
d) wie man komplexe Abläufe in einer ereignisgesteuerten Applikation in C programmiert.

Die in der 1. Aufgabe implementierten Funktionalitäten (d.h. die Ausgabe eines Blinkmusters sowie das Entprellen von Tasten) sind auch in der 2. Aufgabe zu behalten und ggf. zu erweitern.

Aufgabenstellung

Das Evaluationsboard mit dem Mikrocontroller MSP430FR5729 lässt sich über fünf PMOD-Schnittstellen um weitere Experimentierplatinen flexibel erweitern.





Eine der Erweiterungsplatinen ist mit einer vierstelligen Siebensegmentanzeige (bestehend aus zwei LTD-4708JR-Modulen) und dem dazugehörigen Treiberbaustein AS1108WL sowie mit vier einfachen Tastern (Buttons BTN0 bis BTN3) bestückt. Auf einer solchen Erweiterungsplatin befinden sich auch zwei PMOD-Schnittstellen: eine vom Typ 2 für eine SPI-Verbindung zum Treiberbaustein und eine zweite vom Typ 1, über die der Zustand der vier Taster abgefragt werden kann. Die vier Taster sind auf der Erweiterungsplatine räumlich direkt vor der vierstelligen Siebensegmentanzeige angeordnet, und zwar so, dass sie mit den einzelnen Zifferpositionen dieser Siebensegmentanzeige korrespondieren.

Im Rahmen der 2. Aufgabe sind zwei Teilaufgaben zu lösen. 1) Die vierstellige Siebensegmentanzeige ist über die SPI-Schnittstelle des Mikrocontrollers so anzusteuern, dass man mit deren Hilfe beliebige vierstellige Dezimalzahlen (von 0000 bis 9999) anzeigen kann. 2) Mit Hilfe der Taster kann man jede der einzelnen Zifferpositionen separat inkrementieren (+1) oder dekrementieren (-1). Die Auswahl der Funktionen Inkrementieren/Dekrementieren wird über den Taster BTN1 der Hauptplatine (mit dem

Mikrocontroller MSP430) gesteuert, wobei die ausgewählte Funktion über die LED1 angezeigt wird. Beim Inkrementieren ist die LED1 ausgeschaltet, und beim Dekrementieren ist sie eingeschaltet. Durch den Klick auf den Taster BTN1 kann man zwischen den beiden Funktionen hin- und herschalten.

Beim Inkrementieren/Dekrementieren einer Zifferposition ist zu beachten, dass ein Überlauf auf die nächsthöhere Zifferposition entstehen kann. Der Überlauf ist auf arithmetisch korrekte Art zu behandeln und ggf. auf die darauffolgenden Zifferpositionen weiter zu leiten. In der unteren Tabelle sind einige Beispiele dargestellt, wie sich das System beim Inkrementieren/Dekrementieren zu verhalten hat.

Inkrementieren					
Anzeige VOR	BTN 3	2	1	0	Anzeige DANACH
0000				X	0001
0001				X	0002
0009				X	0010
1299				X	1300
0030			X		0040
1299			X		1309
9999				X	0000
1234		X			1334
1934		X			2034
9999		X			0099
8765	X				9765
9765	X				0765

Dekrementieren					
Anzeige VOR	BTN 3	2	1	0	Anzeige DANACH
0000				X	9999
0001				X	0000
0009				X	0008
1300				X	1299
0040			X		0030
1309			X		1299
0099		X			9999
1334		X			1234
2034		X			1934
2004			X		1994
0099		X			9999
9765	X				8765

Nach der Reset/Power-On-Phase ist die Funktion Inkrementieren aktiv, die vierstellige Dezimalzahl wird mit 0000 initialisiert und die vierstellige Siebensegmentanzeige zeigt den Wert 0000 an.

Hinweis: Machen Sie sich mit dem Datenblatt des Treiberbausteins AS1108WL vertraut, insbesondere mit der Beschreibung im Kapitel 8. Das Datenblatt ist als PDF-Dokument unter den Namen AS1108_Datasheet_EN_v2.pdf verfügbar.

Hinweise zur Implementierung: 1) Es wird empfohlen, den Ablauf in drei sog. Handler aufzuteilen, die mittels Events und Daten miteinander kommunizieren und kooperieren. Beispielsweise könnte sich ein zustandsloser Handler um die Auswertung von Button-Events kümmern, ein zweiter, ebenfalls zustandsloser Handler um die Aktualisierung der einzelnen Zifferpositionen, und ein dritter, zustandsbehafteter Handler, der mit Hilfe einer Zustandsmaschine implementiert ist, um die Ausgabe der vierstelligen Zahl mittels der SPI-Schnittstelle. Sind alle drei Handler „sauber“ implementiert, so kann man ihre Reihenfolge in der Funktion main() beliebig vertauschen. 2) Warteschleifen (egal wie implementiert, ob mit Hilfe von while, for oder goto) sind in der implementierten Aufgabe nicht zulässig. Von dieser Einschränkung sind Initialisierungssequenzen ausgenommen. Bei der Abnahme der Aufgabe wird darauf geachtet. Werden Warteschleifen gefunden, gilt die Aufgabe als nicht bestanden. Bei der Benutzung von Funktionen aus C-Bibliotheken muss man vorsichtig sein. Auch solche Funktionen beinhalten häufig Schleifen. Auch komplexe Operationen wie Multiplikation, Division oder Modulo werden mit Hilfe von Schleifen realisiert und sind bei der Lösung der Aufgabe nicht zulässig.