Versuch 5: Ausblick

Aufgabe 5.1: CMSIS-Bibliothek

Bei den bisherigen Programmen des Praktikums wurde direkt auf die I/O-Register der verschiedenen I/O-Bausteine wie z.B. GPIOB, USART1, Timer2 usw. zugegriffen.
Für die Mikrocontroller der Cortex-Reihe werden von ARM mehrere Bibliotheken angeboten, die die Verwendung der I/O-Bausteine erleichtern. Eine dieser Library heißt CMSIS (**C**ortex **M**icrocontroller **S**oftware **I**nterface **S**tandard) und ist ein herstellerunabhängiger Hardware Abstraction Layer für die Cortex-M Prozessoren.

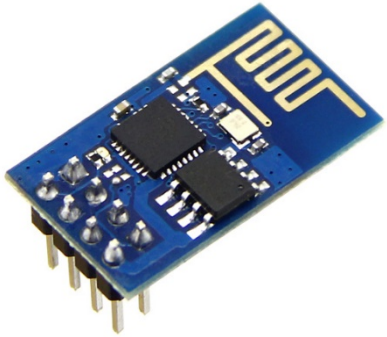
Um den Umgang mit dieser Bibliothek kennenzulernen, soll das Modul `USART2.c` aus dem Versuch 2 mit Hilfe der CMSIS-Bibliothek programmiert werden. Nehmen Sie als Basis das Projekt `V5.1_CMSIS`. Hier ist der Versuch 4.3 komplett mit der Verwendung der CMSIS-Library umgesetzt. Im Modul `USART2.c` müssen nur noch von Ihnen die Funktionen `InitUSART2` und `WriteChar` ergänzt werden.

Vervollständigen Sie diese Funktionen und verwenden Sie die passenden Bibliotheks-Funktionen.

Hinweise:

- Auf der folgenden Internetseite finden Sie eine gute Beschreibung der CMSIS-Funktionen. In den Unterlagen zum Versuch in moodle ist das USART-Kapitel der CMSIS-Dokumentation zur Verfügung gestellt.
  http://www.disca.upv.es/aperles/arm_cortex_m3/llibre/st/STM32F439xx_User_Manual/group__stm32f4xx__ll__driver.html
- Die Idee bei der Verwendung von CMSIS ist, dass nie direkt auf die I/O-Register zugegriffen wird. Zur Initialisierung wird daher zunächst eine Datenstruktur (z.B. `USART_InitStructure`) befüllt und dann mit einer Initialisierungsfunktion die Werte in die eigentlichen I/O-Register übertragen. Dabei wird die korrekte Bitposition im I/O-Register berücksichtigt, die der Programmierer nicht mehr wissen muss. Sie verwenden z.B. die Konstante `USART_Mode_Rx` beim Belegen der `Init-Stucture`, um den Receiver der USART-Schnittstelle einzuschalten. Die CMSIS-Bibliothek setzt daraufhin das Bit 2 im Register `USART_CR1` (siehe Versuch 2).
  Ebenso wird das Einstellen der Baudrate vereinfacht. Sie müssen nur den Wert 9600 angeben, die CMSIS-Bibliothek berechnet daraus die richtige Belegung von Mantisse und Fraktion im BRR-Register.

Aufgabe 5.2: WLAN



Als letzter Teil des Praktikums wird der STM32-Mikrocontroller mit einem WLAN-Modul ESP8266 ergänzt. Dieser Mikrocontroller enthält den kompletten TCP/IP-Stack und ein WLAN-Modul. In unserem Versuch werden wir den ESP8266 dazu verwenden, um die Eingaben über putty durch die Eingaben über einen Webbrowser (Smartphone, Notebook) zu ersetzen.

Binden Sie dazu das Programmmodul `ESP_USART3.c` in Ihr Projekt ein und fügen mittels #include die Datei `ESP_USART3.h` in das main-Modul ein.
Rufen Sie dann die Funktion `Program_to_ESP(PlatzNummer)` als letzten Aufruf im Initialisierungsteil der main-Funktion auf. Als Parameter geben Sie die Nummer Ihres Arbeitsplatzes an.
Beim Start des Programms wird ein WLAN-Accesspoint gestartet mit der SSID `ESP-WLAN x`, wobei x die die Arbeitsplatznummer ist. Das WLAN-Passwort des ESP lautet „1234567890“.

Anschließend müssen Sie auf Ihrem Tablet oder Handy den Webbrowser starten und die URL `192.168.4.1` aufrufen.

Sie können auf einem Android-System alternativ die App `MCT-Input` installieren (siehe moodle->Unterlagen V5)