LABORÜBUNGEN MIKROCONTROLLER – ÜBUNG 5

BEISPIEL 9 – ANALOG-DIGITAL-CONVERTER

Ausgehend von Beispiel 8 soll das Tastverhältnis für das PWM-Signal durch einen analogen Wert vorgegeben werden. Verbinden Sie dazu das Potentiometer am Steckbrett mit GND und VCC vom Board. Stecken Sie das Flachbandkabel von Port A ab und verbinden Sie den Schleifer des Potentiometers mit Pin PA0.

Erstellen Sie die Dateien *ADC.h* und *ADC.c*, binden Sie diese in das Projekt ein und deklarieren Sie die Funktion `InitADC()`. Die Implementierung der Funktion soll den AD-Wandler aktivieren und im Single-Conversion-Modus betreiben. Deklarieren Sie zudem eine Funktion `GetADCValue()`, welche den Pegel von PA0 quantisiert.

Entnehmen Sie die dafür benötigten Register sowie deren Parametrisierung dem Datenblatt des ATmega32, welches im eLearning-System bereitgestellt ist.

Der quantisierte Wert soll in Folge in einen Protzentwert umgerechnet werden und der Funktion `SetPWMDuty(T duty)` aus Beispiel 8 übergeben werden.

BEISPIEL 10 – RINGPUFFER

Implementieren Sie einen Ringpuffer, der 20 Elemente (Byte) fassen kann, indem Sie ein Array entsprechender Größe anlegen und Funktionen zum Hinzufügen sowie zum Entnehmen einzelner Elemente bereitstellen. Fügen Sie der Reihe nach die nachfolgend aufgezählten Elemente dem Ringpuffer hinzu und lesen Sie nach jedem Aufzählungspunkt ein Element aus. Überprüfen Sie die Funktionalität des Ringpuffers durch Debugging im AVR Simulator.

- *0x01* [hinzufügen]
  [1 Element auslesen]
- *0x02, 0x03, 0x04* [hinzufügen]
  [1 Element auslesen]
- *0x05*, *0x06*, *0x07*, *0x08*, *0x09*, *0x0A* [hinzufügen]
  [1 Element auslesen]
- 10-mal *0x10* [hinzufügen]
  [1 Element auslesen]
- *0x0B* [hinzufügen]
  [1 Element auslesen]
- 5-mal *0x20* [hinzufügen]
  [1 Element auslesen]
- *0x0C* [hinzufügen]
  [1 Element auslesen]
- 21-mal *0x30* [hinzufügen]
  [1 Element auslesen]