Version 2.6 vom 14.04.2022

Über dieses Dokument:

Die Einhaltung der in diesem Dokument beschriebenen Vorgaben zur Erstellung der Quelltexte sowie der Dokumentation ist wesentlich für die Erteilung des Testats. Bitte lesen Sie deshalb die folgenden Seiten sorgfältig durch und beachten Sie die dort aufgeführten Anweisungen !

Beachten Sie bitte vor der endgültigen Abgabe die Checkliste, die Sie auf der Homepage des Praktikums Mikrocomputertechnik finden !

Vorgaben zur Erstellung der C-Programme im Praktikum Mikrocomputertechnik:

- Erstellen Sie modulare Programme, schreiben Sie also für jede logische Funktion ein eigenes Unterprogramm (z.B. Tastenabfrage, Display-Ausgabe, A/D-Wandler-Abfrage, etc.). Nutzen Sie dabei auf sinnvolle Art die verschiedenen Möglichkeiten der Variablenübergabe (call by value / call by reference).

- Benutzen Sie globale Variablen nur dann, falls dies wirklich unumgänglich ist. Dies ist eigentlich nur dann der Fall, wenn Informationen in eine Interrupt-Serviceroutine hinein oder aus ihr heraus übergeben werden sollen. **Begründen Sie jede einzelne globale Variable in Ihrer schriftlichen Ausarbeitung.**

- Benutzen Sie sinnvolle Einrückungen, um Ihren Quelltext lesbar zu gestalten.

- Benennen Sie die Variablen sinnvoll, also nicht i1, i2, i3 etc.

- Verwenden Sie aus Performancegründen bevorzugt Integer-Datentypen, deren Größe der Datenbusbreite des verwendeten Mikrocontrollers entspricht. Beim LPC1769 sollten also vorwiegend (u)int32_t – Variablen zum Einsatz kommen. Vermeiden Sie möglichst Variablen vom Typ *float*, da deren Verwendung oft unnötig große und langsame Programme erzeugt.

- Der *#include* – Mechanismus dient ausschließlich dem Einbinden von Header-Dateien, nicht aber von C-Quelltexten ! Die Bibliotheksdateien werden über den Linker eingebunden (über *Project*)

- Jeder Strukturblock besitzt jeweils einen Ein- und Ausgang, Mehrfach-Returns aus Funktionen sind unzulässig ! Verzichten Sie generell auf die Verwendung der C-Statements `continue` und `goto`, verwenden Sie das `break` ausschließlich im Kontext von `switch/case` – Anweisungen !

- **Jedem C-Unterprogramm ist ein Kopf voranzustellen, in dem Folgendes zu dokumentieren ist:**

  - **Was bewirkt die Funktion ?**
  - **Eingangsvariablen, mit möglichem Wertebereich**
  - **Ausgangsvariablen, mit möglichem Wertebereich**
  - **Eventuell verwendete, globale Variablen**

Allgemeine Hinweise zur Erstellung der Dokumentation:

Erstellen Sie zu jeder Aufgabe eine Software-Dokumentation mit folgenden Schwerpunkten:

- Software-Dokumentation mittels Doxygen

  Nutzen Sie die Funktionalität von Doxygen, um über geeignete Kommentierung des Quelltextes eine Software-Dokumentation zu generieren.
  Das auf der MCT-Homepage in Form der ZIP-Datei *DoxyTestSources.zip* abgelegte Beispielprogramm (DoxyTest.c / DoxyTest.h) beinhaltet dabei das Mindestmaß der zu verwendenden Doxygen-Kommentare.

- Textliche Lösungsbeschreibung

  Erklären Sie textlich und ggf. mit Hilfe von Skizzen den Aufbau und den Ablauf Ihrer selbst erstellten Software. Beschreiben Sie dabei auch den Zweck und die Funktionsweise der einzelnen C- Unterprogramme.

- Die folgenden Fragenkomplexe sind **ausführlich** in der Dokumentation **aller B-Aufgaben** zu beantworten:

  1. Beschreiben Sie **detailliert** alle von Ihnen verwendeten Prozessorregister (welche Bedeutung haben die Register bzw. die dort enthaltenen Steuerbits und warum haben Sie welche Bits wie gesetzt ?). Nutzen Sie zur **Verdeutlichung** gern Auszüge aus den Datenblättern.

  2. Beschreiben Sie ausführlich **alle** von Ihnen verwendeten Interrupts. Hierzu gehören die Initialisierung (im Mikrocontroller sowie ggf. im I²C-Baustein), die Interrupt-serviceroutine (welche Flags sind wichtig, wie ist das Zeitverhalten der ISR ?) sowie das Zurücksetzen von Interruptflags und auslösenden Devices.

  3. Beschreiben Sie **detailliert alle von Ihnen verwendeten I²C-Bausteine** und erklären Sie ausführlich die Kommunikation zwischen Mikrocontroller und I²C-Slaves.

     3a. Wie wird die **Adressierung der einzelnen Busteilnehmer sowie innerhalb derselben durchgeführt** und wie kann man auf Bit-Ebene zwischen Lese- und Schreibvorgang unterscheiden ?

     3b. Wie signalisiert ein I²C-Slave, ob weitere Bytes empfangen werden können oder nicht ? Wie signalisiert ein I²C-Master einem Slave, ob gerade das letzte Byte eines Kommunikationsvorgangs transferiert wird oder nicht ?

  Teilweise wird es zu Überschneidungen zwischen den Punkten 1 bis 3 kommen. Es ist dann natürlich nicht erforderlich, diese Sachverhalte doppelt zu dokumentieren, es reicht ein entsprechender Hinweis in der Ausarbeitung.

- Gut kommentierte Quelltexte. Beachten Sie hierbei unbedingt und vollständig die „Vorgaben zur Erstellung der C-Programme“ auf Seite 1

Lassen Sie sich nach Fertigstellung der Software die Funktion sowie ein gutes Verständnis der Materie testieren und senden Sie anschließend Ihr gemäß Kapitel 11.1 unserer Gesamtdokumentation exportiertes Projekt sowie die elektronische Form Ihrer Ausarbeitung (PDF-Format), gemeinsam gepackt in einer ZIP-Datei (kein RAR, ARJ, etc.) an

praktikum@micro.et-inf.fho-emden.de

oder benutzen Sie den hierfür ggf. in Moodle zur Verfügung gestellten Upload-Bereich.

Bitte benennen Sie die PDF- und die ZIP-Datei nach folgendem Schema: <Aufgabennummer>-<Name1>-<Name2>.[pdf|zip], also beispielsweise: B6-Schulze-Schmidt.pdf und B6-Schulze-Schmidt.zip