Friedrich-Alexander-Universität Erlangen-Nürnberg

Schriftliche Prüfung zu dem Modul „Systemnahe Programmierung in C (SPiC)“ in den Bachelor-Studiengängen Mechatronik, Mathematik, Technomathematik

	erreichbare Punkte	erhaltene Punkte							Noten-anteil
Aufgabe 1	**20**								**22 %**
Aufgabe 2	**46**	**A**	**M**	**I**		**D**			**78 %**
Aufgabe 3 bis 5	**24**	**3a**	**3b**	**3c**		**4**		**5**	
Summe Aufgabe 2-5	**70**								

(Name) (Vorname) (Matrikel-Nr.)

(Studiengang) (Semester)

Durch meine Unterschrift bestätige ich

- den Empfang der vollständigen Klausurunterlagen (17 Seiten inkl. Deckblatt sowie ein Blatt mit den Manualseiten zu dir, stat und strings),
- die Kenntnisnahme der Hinweise auf Seite 2.

Erlangen, 26.07.2013 .....

(Unterschrift)

Hinweise

Die folgenden Informationen bitte aufmerksam lesen und die Erklärung auf Seite 1 unterschreiben.

- Die Bearbeitungszeit beträgt 90 Minuten.
- Es sind **keine** eigenen Hilfsmittel zugelassen.
- Die Lösung einer Aufgabe soll auf das Aufgabenblatt in den dafür vorgesehenen Raum geschrieben werden. Beachten Sie bitte, dass der freigelassene Platz großzügig bemessen ist und nicht unbedingt der erwarteten Antwortlänge entspricht. Sollte der Platz nicht ausreichen, können Sie die Rückseiten der Aufgabenblätter mitverwenden. Kennzeichnen Sie dabei die Zugehörigkeit Ihrer Lösung zu einer Aufgabe. Bei Bedarf können zusätzliche Lösungsblätter (weiß) ausgeteilt werden. Vermerken Sie vor deren Verwendung unbedingt Ihren Namen und Ihre Matrikelnummer darauf!
- Die Lösungen müssen dokumentenecht in blau oder schwarz geschrieben werden. Als falsch Erkanntes muss deutlich durchgestrichen werden. Tintenkiller darf nicht verwendet werden. Keinen Bleistift verwenden!
- Schmierpapier (farbige Blätter) darf **nicht** abgegeben werden. Bei Bedarf ist von der Aufsicht weiteres Schmierpapier erhältlich.
- Fragen zu den Prüfungsaufgaben können grundsätzlich nicht beantwortet werden.
- Tragen Sie Ihren Namen und Vornamen, Ihre Matrikelnummer, Studiengang und Fachsemesterzahl auf dem Deckblatt der Klausur ein.
- Bitte legen Sie Ihren Studenten- und einen Lichtbildausweis zur Kontrolle bereit.
- Sie dürfen den Raum nicht verlassen, bevor Ihre Personalien überprüft wurden und Sie die Klausurunterlagen der Aufsicht zurückgegeben haben.
- In den letzten 15 Minuten der Bearbeitungszeit können Sie den Raum nicht mehr verlassen. Bleiben Sie an Ihrem Platz sitzen, bis am Ende alle Klausurunterlagen eingesammelt und nachgezählt sind und die Aufsicht das Zeichen zum Gehen gibt.

Die **Klausurergebnisse** werden in ca. einer Woche im WWW unter der Seite der Vorlesung im SS 2013, Unterpunkt "Ergebnisse" veröffentlicht (Zugriff mit Ihrem Login und Matrikelnummer bzw. Passwort im WAFFEL-System).

Aufgabe 1: (20 Punkte)

Bei den Multiple-Choice-Fragen ist jeweils nur **eine** richtige Antwort eindeutig anzukreuzen. Auf die richtige Antwort gibt es die angegebene Punktzahl.

Wollen Sie eine Multiple-Choice-Antwort korrigieren, streichen Sie bitte die falsche Antwort mit drei waagrechten Strichen durch (≢)und kreuzen die richtige an.

Lesen Sie die Frage genau, bevor Sie antworten.

a) In welcher Situation kann ein lost-update-Problem auftreten? 2 Punkte

- [ ] Wenn eine Interrupt-Sperre zu lange gehalten wird.
- [ ] Wenn sowohl in der main-Funktion als auch in einem Interrupt-Handler modifizierend auf die gleiche `uint16_t`-Variable zugegriffen wird.
- [ ] Bei der Modifikation von lokalen Variablen in zwei Interrupt-Handlern.
- [ ] Wenn während einer Interrupt-Bearbeitung weitere Interrupts gesperrt sind.

b) Gegeben ist folgender Ausdruck: 2 Punkte

```
if ( (a = 5) || (b != 3) ) ...
```

Welche Aussage ist richtig?

- [ ] Der Wert von a hat keinen Einfluss auf das Ergebnis.
- [ ] Falls a den Wert 7 und b den Wert 5 enthält, wird falsch zurückgeliefert.
- [ ] Falls a den Wert 5 und b den Wert 7 enthält, wird falsch zurückgeliefert.
- [ ] Der Compiler meldet einen Fehler, weil dieser Ausdruck nicht zulässig ist.

c) Welche Aussage zu globalen Variablen ist richtig? 2 Punkte

- [ ] Globale Variablen sind bei sehr großen Programmen vorteilhaft, weil man alle Variablendefinitionen an einer Stelle hinschreiben kann und man sie dadurch sehr schnell finden kann.
- [ ] Durch die Verwendung von globalen Variablen kann man den Einsatz von Funktionsparametern vermeiden. Dadurch werden Programme übersichtlicher und leichter wartbar.
- [ ] Durch den Einsatz von globalen Variablen werden vor allem große Programme unübersichtlich und auf Dauer schwer wartbar, da der direkte Bezug zwischen Daten und den Funktionen verloren geht.
- [ ] Man sollte globale Variablen sparsam einsetzen, da sie mehr Speicherplatz benötigen als lokale Variablen.

d) Gegeben sei folgende Enumeration: 2 Punkte

```
enum SPRACHE {Deutsch, Englisch, Russisch};
```

Welche Aussage ist richtig?

Der Compiler meldet einen Fehler, weil den enum-Elementen kein Wert zugewiesen wurde.

Der Wert von Russisch ist 2.

Der Wert von Russisch ist 3.

Der Wert von Russisch ist unbekannt; der Compiler weist zur Übersetzungszeit jedem enum-Element einen zufälligen, aber eindeutigen Wert zu.

e) Welche der folgenden Aussagen zum Binden eines Programms ist richtig? 2 Punkte

Auf einer Mikrocontroller-Plattform wird normalerweise dynamisch gebunden, weil dies Speicherplatz spart.

Der Binder löst `#include`-Anweisungen in einem C-Programm auf, indem er sie durch den Inhalt der einzubindenden Datei ersetzt.

Der Binder fügt Objektdateien und Bibliotheken zu einer ausführbaren Datei zusammen.

Beim Binden werden Zugriffe aus einem C-Modul auf globale `static`-Variablen eines anderen C-Moduls optimiert.

f) Welche der folgenden Aussagen über den C-Präprozessor ist richtig? 2 Punkte

Der Präprozessor optimiert Makros durch Zeigerarithmetik.

Nach dem Übersetzen und dem Binden müssen C-Programme durch den Präprozessor nachbearbeitet werden, um Makros aufzulösen.

Der Präprozessor ist eine Softwarekomponente, welche Java-Klassen durch C-Funktionen ersetzt, die dann von einem C-Compiler übersetzt werden.

Die Syntax von Präprozessoranweisungen ist unabhängig vom Rest der Sprache C.

g) Gegeben sei folgender Programmcode

2 Punkte

```
int a = 11;
int b = 47/(a-11);
```

Welche Aussage ist richtig?

Das Ergebnis von b ist unendlich

Das Ergebnis von b ist 0

Das Ergebnis von b ist NaN (Not a Number)

Auf einem Linux-System wird das Programm mit einem Trap abgebrochen.

h) Welche Aussage zu Prozessen ist richtig?

2 Punkte

Auf Multi-Core-Systemen können mehrere Prozesse echt parallel ausgeführt werden. Jedoch nur so viele, wie Prozessorkerne zur Verfügung stehen.

In einem Prozess können mehrere Programme gleichzeitig ausgeführt werden.

Laufen mehrere Prozesse auf einer CPU, können diese direkt Daten austauschen.

Auf Single-Core-Systemen gibt es keine Nebenläufigkeit.

i) Welche Angaben enthält ein Eintrag eines Katalogs (Verzeichnis) in einem Standard-UNIX-Dateisystem?

2 Punkte

Blocknummer des Inode-Plattenblocks und Dateiname

Inode-Nummer und Dateiname

Dateiname, Dateigröße, Eigentümer und Zugriffsrechte

nur die Inode-Nummer

j) Welche Aussage zum Thema virtueller Adressraum ist richtig?

2 Punkte

Die Umrechnung von virtuellen zu physikalischen Adressen erfolgt beim Übersetzen durch den Compiler.

Dieselbe virtuelle Adresse kann in verschiedenen Prozessen auf unterschiedliche physikalische Adressen abgebildet werden.

Virtuelle Adressen entsprechen Variablennamen in einem C-Programm. In Zeigern werden dagegen physikalische Adressen gespeichert, mit denen man die Abbildung umgehen kann.

Die Abbildung von virtuellen auf physikalische Adressen erfolgt während der Programmlaufzeit durch eine spezielle Softwarekomponente.

Aufgabe 2a: stoppuhr (30 Punkte)

Sie dürfen diese Seite zur besseren Übersicht bei der Programmierung heraustrennen!

Programmieren Sie eine Stoppuhr, die mit Hilfe eines AVR-Mikrocontrollers implementiert ist. Die Stoppuhr hat zwei Taster. Der erste Taster startet und stoppt die Stoppuhr, der zweite Taster setzt sie zurück.

Die Stoppuhr besitzt eine Flüssigkristallanzeige die über eine Bibliotheksfunktion angesteuert wird. Die Anzeige hat eine Genauigkeit von 1 Millisekunde und kann bis zu 999,999 Sekunden anzeigen.

Des Weiteren steht eine Bibliothek zur einfacheren Konfiguration des Timers zur Verfügung.

Im Detail soll Ihr Programm wie folgt funktionieren:

- Initialisieren Sie die Hardware in der Funktion `void init(void);`. Es sollen keine Annahmen über den initialen Zustand der Hardware-Register getroffen werden. Das Display zeigt 0 an.
- Das Programm wartet im Schlafmodus auf Tastendruck oder Timer-Interrupt.
- Wird der Start-Stopp-Taster gedrückt, startet die Stoppuhr. Das Display wird jede Millisekunde aktualisiert.
- Wird der Start-Stopp-Taster erneut gedrückt, stoppt die Stoppuhr. Bei einem weiteren Druck startet sie wieder.
- Wird der Reset-Taster gedrückt, wird die Stoppuhr angehalten und in den Ausgangszustand zurückgesetzt.

Display-Bibliothek:

- `showtime( uint16_t ms);` Zeigt auf der Anzeige `ms` Millisekunden an. Beim ersten Aufruf wird das Display initialisiert. Da dies eine aufwändige Funktion ist, sollen unnötige Aufrufe vermieden werden.

Timer-Bibliothek:

Die Funktionen der Timer-Bibliothek dürfen nicht von einem Interrupt aus aufgerufen werden!

- `void timer_setup(uint16_t ms);` Löst alle `ms` Millisekunden die Interruptquelle `TIMER1_COMPA` aus. ISR-Vektor-Makro: `TIMER1_COMPA_vect`
- `void timer_stop(void);` Deaktiviert den Timer

Information über die Hardware

Taster: **PORTD**, "Start-Stopp"-Taster (Taster 0) = Pin 2, "Reset"-Taster (Taster 1) = Pin 3
- Pin als Eingang konfigurieren: entsprechendes Bit in **DDRD**-Register auf 0
- externe Interruptquellen **INT0** und **INT1**, ISR-Vektor-Makros: **INT0_vect** und **INT1_vect**
- Aktivierung der Interruptquellen erfolgt durch Setzen des **INT0**- bzw. **INT1**-Bits im Register **GICR**.
- die Taster verbinden den Pin mit Masse, es müssen die internen Pullup-Widerstände verwendet werden (entsprechende Bits in **PORTD**-Register auf 1 setzen).

Konfiguration der externen Interruptquellen 0 und 1 (Bits in Register **MCUCR**)

Interrupt 0		**Beschreibung**	**Interrupt 1**	
ISC01	**ISC00**		**ISC11**	**ISC10**
0	0	Interrupt bei low Pegel	0	0
0	1	Interrupt bei beliebiger Flanke	0	1
1	0	Interrupt bei fallender Flanke	1	0
1	1	Interrupt bei steigender Flanke	1	1

Ergänzen Sie das folgende Codegerüst so, dass ein vollständig übersetzbares Programm entsteht.

```
#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/sleep.h>
#include <stdint.h>

/* Funktionsdeklarationen, globale Variablen, etc. */

. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .

/* Unterbrechungsbehandlungsfunktionen */

. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
```

A:

```
/* Funktion main */

   /* Initialisierung und lokale Variablen */

  /* Hauptschleife */

      /* Ruhezustand - warten auf ein Ereignis */

      /* Behandlung des Start-Stop-Tasters */
```

```
    /* Behandlung des Reset-Tasters */
```

```
    /* Behandlung des Timers */
```

```
/* Ende main */
```

M:

```
/* Initialisierungsfunktion */
```

I:

Aufgabe 2b: dirmax (16 Punkte)

Sie dürfen diese Seite zur besseren Übersicht bei der Programmierung heraustrennen!

Schreiben Sie ein Programm `dirmax`, welches die größte reguläre Datei in einem Verzeichnis sucht.

Das Programm bearbeitet jeweils genau ein Verzeichnis, der komplette Pfadname des Verzeichnisses wird beim Aufruf von `dirmax` als Argument übergeben.

Nach dem Durchsuchen des Verzeichnisses gibt `dirmax` das Ergebnis in der folgenden Form aus:

`Groesste Datei: dateiname; n Bytes`

(für *`dateiname`* wird der Name der ermittelten Datei, für *`n`* die Größe dieser Datei ausgegeben)

Sollte es zwei gleich große "größte Dateien" geben, so ist es egal welche der beiden Dateien ausgegeben wird.

Hinweise:

- Sie können davon ausgehen, dass der gesamte Dateipfad (Pfad + Name) nicht länger als 1024 Zeichen sein wird. Andernfalls müsste das Programm nicht mehr korrekt funktionieren. Es darf aber in keinem Fall zur Überschreitung von Feldgrenzen im Programm kommen!
- Der Fall, dass sich keine regulären Dateien in dem Verzeichnis befinden, muss nicht explizit behandelt werden.

Ergänzen Sie das folgende Codegerüst so, dass ein vollständig übersetzbares Programm entsteht.

```
#include <dirent.h>
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <sys/stat.h>
#include <unistd.h>
#include <string.h>

// Funktion main

-------------------------------------------------------------------------------

-------------------------------------------------------------------------------
   // lokale Variablen

-------------------------------------------------------------------------------

-------------------------------------------------------------------------------

-------------------------------------------------------------------------------

-------------------------------------------------------------------------------

-------------------------------------------------------------------------------

-------------------------------------------------------------------------------

-------------------------------------------------------------------------------

   // Aufrufargumente pruefen

-------------------------------------------------------------------------------

-------------------------------------------------------------------------------

-------------------------------------------------------------------------------

-------------------------------------------------------------------------------

-------------------------------------------------------------------------------

-------------------------------------------------------------------------------

-------------------------------------------------------------------------------
```

```
// Verzeichnis oeffnen und durchsuchen
```

D:

```
   // Ressourcen freigeben
```

```
   // Ausgabe
```

```
// Ende main
```

Aufgabe 3: Microcontroller-Programmierung (13 Punkte)

Die folgenden Beschreibungen sollen kurz und prägnant erfolgen (Stichworte, kurze Sätze).

a) Was versteht man unter einem Microcontroller und was sind seine typischen Bestandteile (4 Punkte)

b) Welche zwei Verfahren gibt es zum Zugriff auf die Register eines Peripheriegerätes? Geben Sie eine kurze Beschreibung. Welches Verfahren kommt bei AVR-Microcontrollern zum Einsatz? (3 Punkte)

c) Die **AVRlibc** stellt Makros zum komfortablen Zugriff auf Peripheriegeräte bereit. Erläutern Sie die Funktionsweise des folgenden Makros anhand seiner Bestandteile (6 Punkte)

```
#define PIND ( * (volatile uint8_t*)( 0x10 ) )
```

Aufgabe 4: Java vs. C (6 Punkte)

Die folgenden Beschreibungen sollen kurz und prägnant erfolgen (Stichworte, kurze Sätze).

Java und C sind sich syntaktisch ähnlich, weisen aber große Unterschiede in Handhabung (Idiomatik) und grundlegenden Prinzipien (Philosophie) auf. Stellen Sie derartige Unterschiede in jeweils drei Beispielen gegenüber.

Aufgabe 5: Locks (5 Punkte)

Erklären Sie stichpunktartig die Funktionsweise und Nachteile von Spin-Locks und Sleeping Locks.