Universität Kassel
Fachbereich Elektrotechnik/Informatik
Prof. Dr. B. Sick
Fachgebiet Intelligente Eingebettete Systeme

UNIKASSEL VERSITÄT
Intelligent Embedded Systems

Labor Embedded Systems

Intelligente Eingebettete Systeme

Aufgabenblatt zur Abschlussaufgabe (Vorabversion)

Abschlussaufgabe, Benotungskriterien und Challenge

Betreuer: Benjamin Herwig <bherwig@uni-kassel.de>
und viele großartige Tutoren

Aufgabe 0: Wichtig!

Die Abschlussaufgabe besteht aus mehreren Komponenten (Roboterfunktionalität, Quelltextqualität, Dokumentationsgüte). Bitte lesen Sie dieses Aufgabenblatt vollständig und ggf. mehrmals gründlich durch. Beachten Sie insbesondere auch die letzte Aufgabe zur Form der Abgabe Ihrer Aufgabe und setzen Sie sie – wie auch alle anderen Aufgaben – exakt wie gefordert um!

Aufgabe 1: Roboterfunktionalität

Ihr Roboter soll am Ende des Praktikums drei Runden über einen Linientrack fahren können und dabei unterschiedliche weitere Funktionen haben.

Eine Runde ist dabei so definiert, dass Ihr Roboter sich ab einem Startfeld – dabei handelt es sich um eine $10cm \cdot 20cm$ große schwarze Fläche – die gesamte schwarze Linie entlang zu fahren hat. Die Runde endet, sobald Ihr Roboter nach dem Abfahren der gesamten Linie wieder am Startfeld ankommt. Das Startfeld dient also auch als Zielfeld.

Das bedeutet, dass Ihr Roboter am Anfang der Abschlussvorführung auf dem Startfeld positioniert wird, er dann die Linie abfährt, dabei das Startfeld zwei mal überfährt und schließlich nach der dritten Runde auf dem Startfeld zu stehen kommt.

Das Folgende muss dabei von Ihnen und Ihrem Roboter geleistet werden:

Ihr Roboter muss bei Beginn der Abschlussvorführung in Fahrtrichtung der schwarzen Linie mittig auf dem Startfeld positioniert werden.

Die Vorderseite Ihres Roboters muss dazu in die Richtung zeigen, in die der dicke schwarze Pfeil neben dem Linientrack zeigt.

Schalten Sie Ihren Roboter erst dann ein, wenn Sie ihn auf dem Track innerhalb des Rahmens positioniert haben.

Dann, wenn Ihr Roboter auf dem Linientrack steht, schalten Sie ihn ein und verbinden Sie sich per Bluetooth mit der USART-Schnittstelle des Microcontrollers.

Ihr Roboter soll nach dem Aufsetzen auf den Linientrack, direkt nach dem Anschalten, vor dem Beginn des Abfahrens der drei Runden, abhängig davon, wo genau er steht, Nachrichten ausgeben und seinen Zustand per Leuchtdioden signalisieren:

Ihr Roboter soll das Stehen auf dem Startfeld dadurch signalisieren, dass alle drei auf dem Steckbrett verbauten, am Schieberegister angeschlossenen Leuchtdioden mit einer Frequenz von $5Hz$ blinken und außerdem den folgenden Text senden, und zwar mit einer Geschwindigkeit von $1Hz$:
`On the starting field. Waiting for your instructions... Send ? for help.` Hierbei können Sie annehmen, dass Ihr Roboter vor dem Start des Abfahrens der drei Runden nicht mit allen drei

Linienfolgerelementen auf der Linie, die hier nicht als Teil des Startfeld definiert ist, zu stehen kommen darf/-sollte.

Diese Ausgabe darf jedoch erst dann erfolgen, nachdem Ihr Roboter das Stehen auf dem Startfeld erkannt hat.

Sollte Ihr Roboter (mit den Linienfolgerelementen) nicht auf dem Startfeld stehen, dann müssen Sie ihn eine andere Textnachricht ausgeben lassen, und zwar mit einer Geschwindigkeit von $1Hz$:
`Not on the starting field. Place me there please... Send ? for help.`

Während Ihr Roboter vom Startfeld entfernt ist, das Abfahren der drei Runden noch nicht begonnen hat und er die eben genannte Textnachricht überträgt, müssen die an das Schieberegister angeschlossenen Leuchtdioden den Zustand der Linienfolgerelemente anzeigen, und zwar muss die in Fahrtrichtung linke Leuchtdiode dann leuchten, wenn das linke Linienfolgerelement über einer schwarzen Linie ist, die mittlere Leuchtdiode muss das entsprechend für das mittlere Linienfolgerleement anzeigen und die rechte Leuchtdiode muss entsprechend das rechte Linienfolgerelement abbilden.

Sobald der Roboter wieder auf das Startfeld gesetzt wird, müssen alle an das Schieberegister angeschlossenen Leuchtdioden wieder mit $5Hz$ blinken und der oben genannte Text (`On the starting field. Waiting for your instructions... Send ? for help.`) mit $1Hz$ übertragen werden.

Abhängig von unterschiedlichen Zuständen Ihres Roboters muss außerdem die Möglichkeit bestehen, dass unterschiedliche Steuerungsanweisungen an Ihren Roboter übertragen werden können – Ihr Roboter muss sich dann wie folgt verhalten:

- Steuerungsanweisung `S`
  Ihr Roboter soll damit beginnen, den Linientrack abzufahren, und zwar drei Mal, also in durch den großen schwarzen Pfeil angegebene Fahrtrichtung von Startfeld zu Zielfeld/Startfeld zu Zielfeld/Startfeld zu Zielfeld. Während des Abfahrens des Tracks muss Ihr Roboter mit einer Frequenz von $1Hz$ den Text `Round and round I go, currently rount #$ROUND`, wobei Sie `$ROUND` bitte als Platzhalter verstehen, in den Sie die aktuelle Rundenzahl, in der sich Ihr Roboter befindet, eintragen – zählen Sie die Runden ab 1 beginnend. Beim Start des Abfahrens der ersten Runde, also nach dem Übertragen des Zeichens `S`, muss Ihr Roboter einmalig den Text
  `Here I go again on my own, going down the only round I've ever known...`
  ausgeben. Nach dem Überqueren des Startzielfelds nach Runde 1 muss Ihr Roboter einmalig den Text `YEAH, done round 1, going for round 2/3` ausgeben. Nach dem Überqueren des Startzielfelds nach Runde 2 muss Ihr Roboter einmalig den Text `YEAH YEAH, done round 2, going for round 3/3` ausgeben. Nach dem Erreichen des Zielfelds nach Runde 3 muss Ihr Roboter einmalig den Text `YEAH YEAH YEAH , I really did it my way. ... And what's my purpose and the general sense of my further life now? Type ? for help.` ausgeben uns sich nach 5 Sekunden reseten (per *Watchdog-Timer*) und dann agieren, wie er es nach dem Einschalten des roboters auch tut.
- Steuerungsanweisung `R`
  Ihr Roboter soll zurückgesetzt (resetet) werden. Lassen Sie den Roboter ausgeben, dass er sich innerhalb von 5 Sekunden resetten wird. Ein Reset soll nur dann erlaubt sein, wenn Ihr Roboter sich auf dem Startfeld befindet. Ein Reset soll jedoch auch dann möglich sein, falls `S` noch nicht übertragen wurde. Tipp: Recherchieren Sie (Selbstständig!), wie man einen Reset per *Watchdog-Timmer* durchführt! Nach dem Reset soll Ihr Roboter agieren, wie er es nach dem Einschalten des roboters auch tut.
- Steuerungsanweisung `P`
  Ihr Roboter soll, ganz egal in welchem Modus er sich gerade befindet, pausieren und so lange nichts tun, bis Sie ihm erneut die Steuerungsanweisung `P` senden. Das Einzige, was er während der Pause tun soll, ist, auszugeben, dass er pausiert, und zwar mit folgendem Text, der mit einer Geschwindigkeit von $1Hz$ übertragen wird: `Pause .... zzzZZZzzzZZZzzz .... wake me up with P again.`
- Steuerungsanweisung `X`
  Ihr Roboter soll gar nichts mehr tun, auch nicht auf `R` reagieren. Er soll nur ununterbrochen, bis er ausgeschaltet (oder per Hardware-Taster resetet) wird, den Text `In safe state! Won't react to any instructions! Rescue me!` mit einer Geschwindigkeit von $1Hz$ übertragen. Außerdem sollen alle am Schieberegister angeschlossenen Leuchtdioden mit einem Lauflicht reagieren, das mit $8Hz$ arbeitet (also soll das Leuchtlicht innerhalb einer Sekunde 8 mal hin- und herlaufen).

- Steuerungsanweisung `C`
  Ihr Roboter soll, falls er gerade – egal in welcher Runde – den Linientrack abfährt, nur noch einmalig das Zielfeld anfahren. Währenddessen soll er ausgeben `Returning home, will reset me there.` Sobald er das Zielfeld erreicht hat, soll Ihr Roboter sich reseten und dann agieren, als wäre er neu eingeschaltet worden.
- Steuerungsanweisung `?`
  Ihr Roboter soll, abhängig davon in welchem Modus er sich befindet (*auf dem Startfeld* oder *nicht auf dem Startfeld*), unterschiedliche Hilfetexte übertragen. Falls er auf dem Startfeld steht, soll die Bedeutung aller möglichen Steuerzeichen übertragen werden. Jede Nachricht darf maximal aus 155 ASCII-Zeichen bestehen!

Jeder der übertragenen Texte ist mit einem Zeilenumbruch zu versehen, sodass die Cutecom-Ausgabe *ordentlich und gut lesbar* erscheint.

Aufgabe 2: Softwarequalität

Die Abschlussaufgabe besteht nicht nur aus dem gemäß dieser Anforderungsbeschreibung arbeitenden Roboter. Teil der Abschlussaufgabe ist es auch, den Ihrem Roboterprogramm zugrunde liegenden Quelltext entsprechend hier definierter Coding-Conventions zu gestalten.

- Coding-Conventions exakt gemäß des im moodle bereitgestellten Dokuments.
- Ausschließlich englische Bezeichner und Kommentare.
- Sie müssen Ihr Programm sinnvoll modularisieren:
  - Ihre `main`-Funktion darf 100 *Lines of Code (LOC)* nicht überschreiten (Kommentare nicht mitgezählt).
  - Keine Ihrer Methoden oder Funktionen darf 200 LOC (Kommentare nicht mitgezählt) überschreiten.
  - Sie müssen jedes Modul um eine Header-Datei ergänzen!
- Sie müssen Ihren Code vernünftig strukturieren und Einrückungen immer nutzen.
- Sie dürfen keinesfalls an irgendeiner Stelle einen `if`-Block oder eine Schleife ohne geschweifte Klammern schreiben.
- Handeln Sie absolut streng nach dem Motto „So lokal wie möglich, so global wie nötig“.
- Sie dürfen nicht im Team arbeiten. Duplizierter Code führt zu massivem Punktabzug für alle Beteiligten, was auch schnell zu einem „nicht bestanden“ führen kann!
- Selbstverständlich dürfen Sie algorithmische Ideen austauschen. Sie müssen Ihren Code jedoch a) mit eigenen Worten programmieren und b) mit eigenen Worten kommentieren.
- Kommentieren Sie höchstens beiläufig *was* Sie tun, erklären Sie aber unbedingt *warum* Sie etwas tun, aber ausführlich. Nehmen Sie auch Bezug auf das Datenblatt des Microcontrollers, mit Seitenzahl usw.
- Dokumentationssprache ist *Englisch*!
- Nutzen sie die Startseite der Dokumentation um Ihr Projekt vollständig in einer groben, aber aussagekräftigen Übersicht darzustellen. Dokumentieren Sie außerdem natürlich jedes weitere Modul und jede Funktion. Der Quelltext Ihres Programms muss mit in der Dokumentation auftauchen, für C optimiert.
- Sie müssen den Prozess, wie jemand, der sich mit der zu verwenden Software nicht auskennt, Ihren Quelltext kompilieren und auf den Roboter übertragen kann auf der Startseite der Dokumentation nachvollziehbar beschreiben. Gehen Sie dabei von in der Thematik unerfahrenen Benutzern aus.
- Sprachlich schlechter Ausdruck führt zu einer schlechteren Bewertung des Dokumentationsteils!
- BEREITGESTELLTEN SOURCECODE MÜSSEN SIE SELBSTSTÄNDIG DOKUMENTIEREN! Schreiben Sie ihn am besten um, so dass Sie ihn ordentlich verstehen! Sie müssen ihn auch gemäß der Coding-Conventions aus dem moodle anpassen!
- Sie müssen auch den von uns bereitgestellten Code mit eigenen Worten dokumentieren, so dass Sie zeigen, dass Sie verstanden haben, was jeweils warum gemacht wird.

- Auch müssen Sie bereitgestellten Sourcecode ggf. noch um passende Header-Dateien erweitern.

Aufgabe 3: Dokumentationsrichtlinien

Die Abschlussaufgabe besteht nicht nur aus dem gemäß dieser Anforderungsbeschreibung arbeitenden Roboter und dem entsprechend der Softwarequalitätsanforderungen gestaltetem Quelltext. Weiterer Teil der Abschlussaufgabe ist es, den Ihrem Roboterprogramm zugrunde liegenden Quelltext entsprechend hier definierter Richtlinien zu dokumentieren.

- Dokumentationssprache ist Englisch!
- Doxygen-Dokumentationsformat ist HTML!
- Ihre Dokumentation muss eine Startseite enthalten (idR. erreichbar per `index.html`), die Ihr Projekt vorstellt, auf der Sie Ihren Roboter, die Abschlussaufgabe und eventuelle Zusatzfunktionalitäten ausführlich vorstellen, derart, dass keine Fragen offen bleiben.
- Sie dürfen die generierte Dokumentation nicht mit abgeben, sondern Sie müssen in einem Makefile ein entsprechendes Target definieren, das *auch* per Default-Target angesprungen wird, sodass die Dokumentation erst nach der Abgabe durch einen make-Aufruf ohne Parameter erzeugt wird.

Aufgabe 4: Infrastrukturelle Vorgaben

Sie müssen Ihr komplettes Softwareprojekt so gestalten, dass beim Build und beim Flashen Ihres Projekts auf einen Roboter nur ein makefile genutzt werden braucht. Sie dürfen keinesfalls eine andere Technik verwenden!

Ihr makefile muss auf der obersten Verzeichnisstrukturebene Ihres Softwareprojekts liegen.

Das Default-Target muss dafür sorgen, dass Ihr Quelltext kompiliert und auf den Roboter übertragen wird und dass Ihr Quelltext außerdem in einer HTML-Dokumentation, die in einem Verzeichnis `doc` auf der obersten Verzeichnisstrukturebene zu finden sein muss, beschrieben wird.

Sie dürfen die Dokumentation nicht selbst generieren und mit abgeben. Das `doc`-Verzeichnis darf in Ihrer Abgabe nicht enthalten sein! Aber es muss, neben vielen weiteren Dingen, nach dem Aufruf von `make` in dem Verzeichnis, in dem sich Ihr Makefile befindet, auffindbar sein.

Mit Ihrem Makefile *muss* das Kompilieren, Linken, Flashen usw. Ihres Quelltextes ohne jeden manuellen Eingriff möglich sein! Außerdem muss Ihr Makefile auch die Dokumentation erzeugen.

Sie *müssen*(!)

Bei der Bewertung/Benotung müssen Sie Zusatzfunktionaliþqt (mehr dazu weiter unten) auf der Dokumentations-Startseite bewerben, sodasssie nicht übersehen werden kann. In der Abschlusspräsentation dürfen Sie eventuelle Zusatzfunktionalität nicht vorführen und sie darf die Präsentation

Aufgabe 5: Benotungskriterien

Die Teilnahme an der Abschlusspräsentation ist Voraussetzung zur Benotung. Die während der Präsentation gegebenen Antworten auf Fragen ist aber nicht ausschlaggebend für den Notenwert, wobei es da evtl. eine Korrelation gibt.

Es gilt, dass eine Abschlusspräsentations-Teilnahme genau dann stattgefunden hat, solange er erkennbar die in Aufgabe 1 geforderte Funktionalität erfüllt.

Sollte dies der Fall sein, dann haben Sie die Gesamtveranstaltung bestanden und können mit der Note „4.0“ (wörtliche Notenbedeutung: „ausreichend“) rechnen.

Sollte zusätzlich Ihr Quelltext gemäß der entsprechenden Aufgabenstellung vernünftig gestaltet sein, dann haben Sie die Note „3.0“ (wörtlich: „befriedigend“) sicher. Hier müssen Sie auch ein Makefile mit abgeben, mit dem der Betreuer Ihren Code compilieren und flashen und auch die Dokumentation erzeugen kann.

Sollte zusätzlich Ihre Dokumentation ausführlich und sinnvoll gemäß Aufgabenstellung gestaltet sein, so habe Sie die Note „2.0“ sicher.

Im Notenraum bis zur „2.0“ sind Abstufungen möglich. Beispiel: Eine vorhandene, aber wenig aussagekräftige Dokumentation wird möglicherweise nur mit der Note „2.7“ bewertet oder aber ein funktionierender Quelltext,

der kaum dokumentiert ist und der sich aber nur partiell an die Coding Conventions hält, wird mit einer „3.3“ oder „3.7“ bewertet.

Ein schlecht gestalteter Code *und* eine schlecht gestaltete Dokumentation führt also z. B. zu einer „3.3“ für den Code-Teil und vielleicht zu einer „2.7“ für den Dokumentationsteil. Insgesamt resultiert das dann in der Note „3.0“ für die Abschlussaufgabe. Es werden also Mittelwerte gebildet.

Sollten Sie zusätzlich zur Aufgabenstellung weitere Funktionalitäten in Ihren Roboter integrieren – beachten Sie dazu jedoch unbedingt die Tipps zum Erreichen einer sehr guten Note –, dann können Sie sich über die Notenstufen „1.7“ und „1.3“ zur Note „1.0“ hocharbeiten.

Aufgabe 6: Tipps zum Erreichen einer sehr guten Note

Hocharbeiten können Sie sich dadurch, dass Sie sich Dinge überlegen, die Ihr Roboter *über die in der Aufgabenstellung hinausgehende Funktionalität* hat. Entsprechende Tipps finden Sie im folgenden Block.
Um sich hochzuarbeiten, müssen Sie sich vorstellen, dass das Fachgebiet IES als Ihr *Kunde* auftritt und sie daür belohnt, falls Sie ein besonders hochwertiges Softwareprodukt liefern.
Hochwertigkeit äußert sich einerseits durch eine sinnvolle, ausführliche Dokumentation, andererseits aber auch durch eine sinnvolle, robuste und umfassende (und dokumentierte) Roboterfunktionalität.

Wie Sie wissen, existieren zwei Linientracks. Auf einem kann an einer vorher nicht definierten Position ein nicht verschiebbares Hindernis aufgestellt werden. Sie können als weitere Funktionalität Ihres Roboters beispielsweise eine Hindernis-Ausweich-Algorithmik implementieren. Sie können Ihren Roboter dann auf dem Linientrack testen, auf dem vom Betreuer ab Termin 5 ein Hindernis positioniert wird.

Weitere denkbare Funktionen Ihres Roboters könnten sein:

- Eine Fernsteuerung über die serielle Schnittstelle (z. B. eintipen von `F100` im seriellen Terminal um den Roboter für 100 Millisekunden nach vorne fahren zu lassen) (wie das geht müssen Sie aber selbst herausfinden)
- Eine Erweiterung für eine evtl. von Ihnen entwickelte Fernsteuerung, die eine Python-GUI bereit stellt und *klickbare* Elemente enthält – Tipp: Pyhon+pyserial+tkinter. Als Sprache, um eine GUI zu entwickeln, wird *ausschließlich* Python akzeptiert.
- Einen ausührlichen Software-Test Ihrer Algorithmik (wie das geht müssen Sie aber selbst herausfinden)
- Einen zeitgesteuerten priotätsbezogenen Task-Scheduling-Algorithmus (wie das geht müssen Sie aber selbst herausfinden)
- Sinnvolle zusätzliche Targets für `make`, also ein sinnvolles Makefile mit „noch mehr“ Funktionalität als unbedingt nötig (wie das geht müssen Sie aber selbst herausfinden)
- Ein Profiling Ihres Codes, ggf. mit anschließender Optimierung per Inline-Assembler.
- … überlegen Sie sich selbst etwas. Je mehr Sie machen, desto besser! Seien Sie kreativ, dabei aber natürlich effizient.
- Auf dem Roboter haben Sie auch die Möglichkeit, der Batterispannung zu messen. Machen Sie das ruhig. Vielleicht fällt Ihnen ja etwas ein, wozu das nützlich sein könnte.

Zusätzliche Funktionalität *müssen* Sie an prominenter Stelle in Ihrer Dokumentation *vorstellen* und auch *bewerben*! Nutzen Sie dafür *unbedingt* die Startseite Ihrer Dokumentation. Während der Challenge dürfen Sie die zusätzliche Funktionalität *nicht* vorführen, sondern Ihr Roboter muss genau wie oben unter *Roboterfunktionalität* spezifiziert arbeiten! Auch darf

Aufgabe 7: Organisatorisches

Alles, was Sie im Rahmen der Abschlussaufgabe erstellen (Sourcecode – abgelegt in entsprechender bzw. von Ihrem makefile vorausgesetzter relativer Verzeichnisstruktur –, Makefile, Konfigurationsdateien, ggf. `readme`-Datei)

Die Form der Abgabe, sobald Sie beschließen fertig zu sein (oder: fertig sein müssen, es gibt eine Abgabefrist!) muss wie folgt aussehen:

Alle in Ihrem Projekt benötigten Artefakte (ausschließlich makefile, ggf. nötige Konfigurationsdateien für von Ihnen genutzte Tools und Sourcecode; Sourcecode abgelegt in der Verzeichnisstruktur so, wie sie vom makefile erwartet wird) müssen als ZIP-Archiv gesendet werden. Ihre Abgabe-Mail darf *nur* das ZIP-Archiv als Anlage enthalten.

Der Betreff Ihrer E-Mail *muss* folgendermaßen gestaltet sein:
`Abgabe der Abschlussaufgabe LCESII; $VORNAME $ZUNAME, $MATRIKELNUMMER`

Der Body, also der Textinhalt Ihrer Mail, *muss* folgendermaßen gestaltet sein:

```
Sehr geehrtes Fachgebiet IES

mein Name lautet $VORNAME $NACHNAME und ich habe die
Matrikelnummer $MATRIKELNUMMER. In der Anlage finden Sie
ein ZIP-Archiv mit meiner Abgabe der Abschlussaufgabe
für den zweiten Teil des Moduls Labor C/Embedded Systems
im Semester $JAHR.

Mit freundlichen Grüßen
$VORNAME $NACHNAME
```

Die angehängte Datei muss ein ZIP-Archiv sein und den folgenden Dateinamen haben:
`LCESII_$JAHR_$VORNAME_$NACHAME_$MATRIKELNUMMER.zip`

Im Betreff und im Body ersetzen Sie die Platzhalter (`$VORNAME`, `$NACHNAME`, `$MATRIKELNUMMER` und `$JAHR`) durch Ihren Vornamen, Ihren Nachnamen, Ihre Matrikelnummer und durch das aktuelle Semester (z. B. 2022 im Falle des Sommersemesters 2022, 2022/2023 im Falle des Wintersemesters 2022/2023).

Die Korrektur einer erfolgten Abgabe durch Sie ist *nicht möglich*! Stellen Sie es sich wie am Ende einer Klausur vor – sobald Sie die Klausur abgegeben haben, sind Korrekturen auch nicht mehr möglich.

Die hier gestellte Abschlussaufgabe muss zu einem bestimmten Termin, der Ihnen per moodle mitgeteilt wird, abgegeben werden, und zwar per E-Mail an `bherwig@uni-kassel.de`.

Seien Sie absolut pedantisch und halten Sie sich exakt an all diese Vorgabe. Abweichungen führen zu Punktabzügen! Verspätete Abgaben werden nicht akzeptiert.

Tipp: Setzen Sie sich selbst ins (`B`)CC-Feld der E-Mail, die sie absenden, sodass Sie sich sicher ein können, dass Sie Ihre E-Mail erfolgreich abgeschickt haben.

Noch ein Tipp: Bevor Sie die E-Mail an `bherwig@uni-kassel.de` senden, senden Sie sich selbst eine Test-Mail mit allen Inhalten, die Sie in der Abgabemail sehen wollen/müssen. Prüfen Sie diese Mail dann gemäß der obigen Vorgaben.