Laborversuch 2

Thema:

Interrupt- und Ein-/Ausgabeprogrammierung mit dem HCS12

Inhaltsverzeichnis

Zusätzlich erforderliche Unterlagen:

- Vorlesungsmanuskript zu Kapitel 1 bis 3 mit Anhang Metroworks CodeWarrior HCS12 Entwicklungsumgebung
- Dokumentationspaket für den HCS12 Mikrocontroller und das Dragon12-Entwicklungsboard

1. Überblick

Ziel des Laborversuchs

- Kennenlernen einiger wichtiger Peripheriekomponenten des HCS12 Mikrocontrollers
- Interruptprogrammierung mit dem HCS12

In der ersten Laborübung ist eine Funktion zur Ausgabe von Daten auf dem LCD-Display des Dragon12-Entwicklungsboards erstellt worden. In dieser zweiten Laborübung bauen wir darauf auf und bauen mit Hilfe der nun vorhandenen Software ein etwas komplexeres System: Eine Uhr und ein Thermometer. Den Temperaturfühler simulieren wir über das auf dem Board vorhandene Potentiometer.

Ablauf des Laborversuchs

Der Laborversuch besteht aus einem Vorbereitungsteil und einem Teil, der erst im Labor durchgeführt wird. Der Vorbereitungsteil enthält eine Reihe von Aufgaben, die **vor dem Laborversuch** bearbeitet werden müssen und in diesem Umdruck mit **Vorbereitungsaufgabe** gekennzeichnet sind. Bei der Vorbereitung werden Assemblerprogramme erstellt und, soweit möglich, mit dem Simulator getestet. Im Labor werden diese und weitere Programme auf das Dragon12-Entwicklungsboard portiert und auf der realen Hardware erprobt.

(1) **Notwendige Vorkenntnisse und Vorbereitungsaufwand**: Für die Vorbereitung dieses Laborversuchs muss ausreichend Zeit eingeplant werden. Die im Rahmen des Laborversuchs 1 erstellten Programme, z.B. für die Umwandlung von Dezimal- und Hexadezimalzahlen in ASCII-Strings und die Ausgabe auf dem LCD-Display des Dragon12-Entwicklungsboards werden in diesem Laborversuch wiederverwendet.

(2) **Laborunterlagen: Den vollständigen Laborumdruck sowie vorbereitete CodeWarrior-Projekt-Dateien finden Sie auf der Web-Seite zu dieser Vorlesung**. Entwerfen Sie alle Programme zunächst mit Hilfe eines Programmablaufplans (Flußdiagramm), bevor Sie den Programmcode schreiben. Vergessen Sie nicht, Ihre Programme ausführlich zu kommentieren. Bringen Sie zum Laborversuch sämtliche Vorbereitungsunterlagen inklusive der Programmablaufpläne und der CodeWarrior-Projektdateien mit.

Die Vorbereitung sollte in der Laborgruppe gemeinsam so durchgeführt werden, dass jeder Gruppenteilnehmer in der Lage ist, den Laborbetreuern die Lösung zu sämtlichen Vorbereitungsaufgaben selbständig zu erklären und bei Bedarf sämtliche Programme vorzuführen und zu erläutern.

Bitte führen Sie die Vorbereitung ohne Rückgriff auf Altmeister durch, auch wenn das eine deutliche Zeitersparnis bedeuten würde. In der Klausur werden Sie auch keinen Altmeister haben und die Labors sind in dieser Lehrveranstaltung die allerbeste Prüfungsvorbereitung. Softwareentwicklung in einer beliebigen Programmiersprache kann man nur dann wirklich lernen, wenn man es selbst macht!

Mangelhafte Vorbereitung oder unentschuldigtes Fehlen führt zum Nichtbestehen des Labors. Das nichtbestandene Labor kann dann im folgenden Semester wiederholt werden.

(3) **Labordurchführung:** Während des Laborversuchs benötigen Sie in jedem Fall das Vorlesungsmanuskript sowie die vollständige Dokumentation des HCS12-Mikrocontrollers und seiner Peripheriebausteine sowie des Dragon12-Entwicklungsboards. Bringen Sie diese Unterlagen bitte zum Labor mit. Während des Labors sind die als **Laboraufgabe** gekennzeichneten Fragen zu bearbeiten.

2. Vorarbeiten zum Timer-Interrupt

Vorbereitungsaufgabe 2.1

In der Vorlage „**lab2-tickerVorlage**“ finden Sie ein Testprogramm für die Treiberfunktionen für den Zeitgeber-Interrupt mit dem Peripheriemodul Enhanced Capture Timer ECT. Das Programm soll eine Leuchtdiode am Port B im Sekundentakt blinken lassen. Die Ansteuerung der Leuchtdiode erfolgt innerhalb der Interrupt-Service-Routine **`isrECT4`** (in der Datei **`ticker.asm`**). Die Interrupt-Service-Routine soll durch den Timer periodisch alle 10ms aufgerufen werden, der Sekundentakt wird daraus durch den Softwarezähler mit der Variable **`ticks`** abgeleitet. Die Initialisierung des Timers erfolgt in der Funktion **`initTicker`**. Diese geht davon aus, dass die Busfrequenz der CPU (durch das Monitorprogramm) auf 24MHz eingestellt wurde. Aus dieser Busfrequenz wird durch durch einen Vorteiler (Prescaler) mit einem Teilungsfaktor von 128 die eigentliche Taktfrequenz von 187,5kHz für den Timer abgeleitet, woraus dieser dann die 10ms Interrupt-Periode erzeugt.

Leider haben sich in die Datei **`ticker.asm`** zwei Fehler eingeschlichen, so dass der Timer-Interrupt und damit das Blinken der Leuchtdiode nicht wie gewünscht funktionieren.

Testen Sie das Programm mit dem Simulator, finden und beseitigen Sie die beiden Fehler. Denken Sie daran, dass Sie die Periode, mit der die ISR aufgerufen wird bzw. die LED blinkt, im Simulator nur über die Messung der CPU-Taktzyklen bestimmen können, da der Simulator nicht mit derselben Geschwindigkeit läuft wie die reale CPU.

3. Entwurf der Uhr und des Thermometers

In dieser Laborübung bauen wir eine einfache Uhr mit Thermometer, die folgende Funktionalität haben soll:

- Die Uhr soll die Zeit in der oberen Zeile des LCD-Displays im Format HH:MM:SS anzeigen und im Sekundentakt zählen. Die Stundenanzeige soll den Wertebereich 0 … 23, die Minuten- und Sekundenanzeige den Bereich 0 … 59 umfassen. Gleichzeitig soll die LED am Port B.0 im Sekundentakt blinken.
- Beim Einschalten soll die Uhr bei 00:00:00 beginnen. Durch kurzes Drücken auf den Taster SW2 soll der Einstellmodus für die Uhr ein- und ausgeschaltet werden. Im Einstellmodus sollen durch Betätigen der Taster SW3, SW4 und SW5 die Stunden-, Minuten- bzw. Sekundenwerte inkrementiert werden. Solange der Einstellmodus aktiv ist, soll die LED am Port B.7 dauerhaft leuchten.
- In der zweiten Zeile des LCD-Displays soll die Temperatur angezeigt werden (z.B. 25°C). Dabei wird angenommen, dass an den Analogeingang Port AD.7 ein Temperaturfühler, z.B. ein Spannungsteiler mit einem temperaturabhängigen Widerstand, angeschlossen ist. Dabei soll eine Analogspannung im Bereich 0 … 5V einer Temperatur von -50°C bis +70°C entsprechen. Ihr Programm muss die Analogspannung messen und den Messwert in „°C“ umrechnen. Für den Test des Thermometers im Labor wird der Temperaturfühler durch das Potentiometer VR2 auf dem Dragon12-Entwicklungsboard nachgebildet werden. Die Temperatur soll wie die Uhrzeit im Sekundenrhythmus aktualisiert werden.

Für die Software-Implementierung dieses Systems sollen folgende Randbedingungen gelten:

- Verwenden Sie für den Sekundentakt die Funktionen aus dem Modul **`ticker.asm`** aus Vorbereitungsaufgabe 2.1, für die LCD-Anzeige das Modul **`LCD.asm`** und für die Umwandlung binärer Werte in ASCII-Strings das Modul **`decToASCII.asm`** sowie gegebenenfalls weitere Funktionen aus dem ersten Laborversuch.
- Teilen Sie die Funktionalität auf verschiedene Dateien auf. So sollte z.B. die Bedienung des A/D-Wandlers zur Temperaturmessung in einer anderen Datei (Modul) untergebracht werden als die des Zeitgebers. Isolieren Sie innerhalb der Module zusätzlich diejenigen Funktionen, die unmittelbar auf die Hardware zugreifen von den übrigen Funktionen in separaten Unterprogrammen.
- Ihr Hauptprogramm **`main.asm`** sollte so klein wie möglich gehalten werden. Auch die Interruptroutinen sollten möglichst klein sein und schnell wieder zum unterbrochenen Programm zurückkehren. Stecken Sie keine große Programmlogik in die Interruptroutine(n) und rufen Sie keine zeitaufwändigen Unterprogramme aus Interruptroutinen aus auf. Im wesentlichen sollten Sie in den Interrupt-Service-Routinen lediglich die Timer-Hardware bzw. gegebenenfalls den A/D-Umsetzer bzw. den Tasteninterrupt bedienen und für das Hauptprogramm einige Flags (globale Variable) setzen. Das Hauptprogramm sollte in einer Warteschleife auf diese Flags warten und dann die notwendigen Unterprogramm zu Weiterverarbeitung und Anzeige aufrufen.

Vorbereitungsaufgabe 3.1

Erstellen Sie unter Berücksichtigung der schon vorhandenen Funktionen Flussdiagramme, die das Hauptprogramm, die Unterprogramme und Interrupt-Serviceroutinen und deren Zusammenspiel beschreiben. Geben Sie bei jedem Flussdiagramm die genaue Bezeichnung der einzelnen Module an.

Basierend auf Ihrem Design erstellen Sie nun die entsprechenden Dateien im Projektordner Ihrer Entwicklungsumgebung.

Beschreiben Sie die Schnittstellen Ihrer Module eindeutig, indem Sie für jedes Modul als Kommentar am Anfang der Moduldatei eine Liste erstellen, die angibt:

- Welche Funktionen (Unterprogramme, Interrupt-Service-Routinen) enthält das Modul?
- Welche Funktionen und Variable importiert oder exportiert das Modul (XDEF, XREF)?

Geben Sie als Kommentar am Beginn jedes Unterprogramms bzw. jeder Interrupt-Service-Routine stichwortartig an:

- Name des Unterprogramms
- Funktion des Unterprogramms
- Übergabeparameter (Register, Speicherort) und Wertebereich beim Aufruf und Beschreibung der Rückgabeparameter. Geben Sie zusätzlich an, welche globalen Variable und gegebenenfalls welche Register das Unterprogramm zusätzlich zu den Rückgabeparametern aus Sicht des aufrufenden Programms verändert.

Verwenden Sie als Vorbild für die Schnittstellenbeschreibung die Vorlagen **`LCD.asm`** und **`Ticker.asm`**. Als Ergebnis dieser Vorbereitungsaufgabe sollten saubere, detaillierte Flussdiagrame entstanden sein sowie die Hüllen der einzelnen Module. Bringen Sie die Flussdiagramme zur Laborübung mit.

Vorbereitungsaufgabe 3.2

Basierend auf dem Programmentwurf aus Vorbereitungsaufgabe 3.1 erstellen Sie nun die Software in HCS12-Assembler. Ersetzen Sie die Routinen, die Sie mit dem Simulator nicht richtig testen könne, z.B. die Analog-Digital-Umsetzung oder die Abfrage der Bedientasten, durch Dummy-Routinen.

Versuchen Sie, soviel wie möglich von Ihrer Software mit dem Simulator ohne Hardware zu testen, so dass Sie im Labor keine oder nur noch sehr wenig Software schreiben müssen. Testen Sie Funktion für Funktion und schreiben Sie gegebenenfalls kleine Testprogramme, um einzelne Funktionen testen zu können.

Bringen Sie Ihre (fast) fertige Software zur Laborübung mit. Dort müssen Sie dem Laborbetreuer Ihren Entwurf anhand der Flussdiagramme erläutern und vorführen, wie Sie die einzelnen Funktionen getestet haben.

4. Programmtest auf dem Dragon12-Entwicklungsboard

Laboraufgabe 4.1

Während des betreuten Teils der Laborübung integrieren Sie die von Ihnen zuvor geschriebene Software auf der Hardware. Testen müssen Sie nun besonders die Funktionen, die ohne Hardware nicht gut prüfbar waren.

Testen Sie gründlich, ob Ihre Uhr und das Thermometer wie gewünscht funktionieren.