

## Vorbereitung zu Versuch 1

NAME; VORNAME	DATUM
Hohenstein, Michael	03.11.18

- 1) Was ist eine Assemblersprache? ✓

Eine Assemblersprache ist eine Maschinensprache, die  
1. von Assembler in die Befehle einer Prozessorarchitektur  
übersetzt wird. Sie ist von der verwendeten Prozessorarchitektur

- 2) Wo liegt der wesentliche Unterschied zu höheren Sprachen wie z.B. C? abhängig ✓

höhere Sprachen ~~arbeiten~~ verhalten sich objektorientiert,  
während in Assembler Daten in den Registern oder anderweitig  
abgelegt werden, wobei der Entwickler den Speicherort  
auf der Hardware ~~Speicher~~ zuordnen muss.

Höhere Sprachen werden von einem Compiler in Assembler übersetzt. ✓

- 3) Welche Aufgabe haben die GPIOs des PSoC-Bausteins?

Die GPIOs können sowohl digitale als auch analoge  
Signale entgegennehmen (eintlesen) oder ausgeben

- 4) Nennen Sie jeweils 3 Komponenten der analogen und digitalen Blöcke. ✓

Digital:

- Counter
- Timer
- Pulsweitenmodulatoren (PWM)

Analog:

- AD-Wandler
- DA-Wandler
- Komparatoren

5) Welche internen Register besitzt der M8C und welche Funktion haben sie? ✓

A : Accumulator, Speichert Ergebnisse von Berechnungen  
X : Index, vielfältige Verwendung  
PC : Program-Counter, enthält die Adresse des nächsten Befehls  
SP : Stack Pointer, Adresse des obersten Elements auf dem Stack  
F : Flag, speichert Systemzustände & Betriebsmodi

6) Welchen Zweck erfüllen die Speicher RAM, SROM und Flash-ROM? ✓

RAM: Daten, Variablen und Stack speichern  
ROM: SROM und Flash-ROM  
↳ Bootprogramm      ↳ Anwendungsprogramm

7) Was sind Usermodule? ✓

Implementierungen in der 3DE, durch die grundlegenden analoge & digitale Funktionen einfach realisiert werden können.

8) Aus welchen Komponenten besteht ein Assemblerprogramm? ✓

• Kommentare      • Direktiven      • Instruktionen  
• Label

9) Wozu dient die Direktive AREA? ✓

AREA legt fest, wo Daten & Befehle gespeichert werden

bss → Variablen  
text → Programmcode  
lit → Zeichenketten / Konstanten } (ROM | REL | COM)

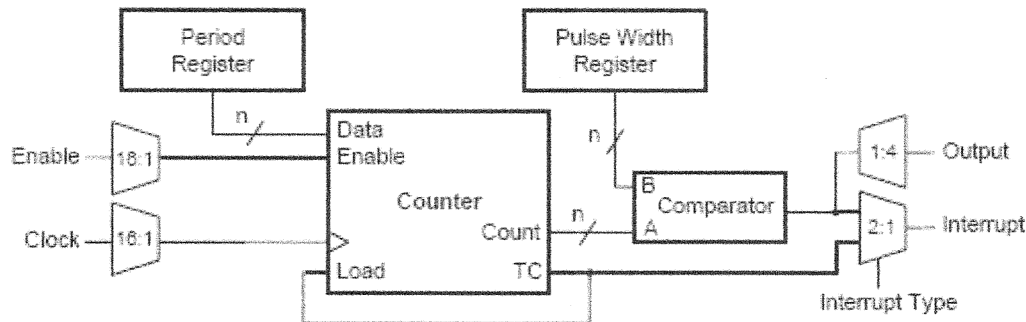
10) Nenne Sie 3 Adressierungsarten ✓

• Source Immediate  
• Source Direct  
• Destination Indexed

Gruppe 8

## 11) Berechnungen zu PWMs

Ein 8-Bit Pulsweitenmodulator ist im PSoC-Baustein wie folgt aufgebaut:



PWM Block Diagram, Data Path width  $n = 8$  or  $16$

Mit dem **Period Register** wird die Periodendauer des PWM-Signals definiert. Mit dem **Pulse Width Register** wird das Tastverhältnis (duty cycle) festgelegt, d.h. wie lange ist der High-Pegel des PWM-Signals im Verhältnis zu einer Periode. Mit jeder steigenden Flanke des Eingangstakts **Clock** wird der zuvor über das **Period Register** geladene **Counter** um 1 dekrementiert. Sobald Null erreicht wird geht der **TC**-Ausgang (Terminal Count) für die Dauer eines Eingangstaktes auf High. Anschließend wird der **Counter** neu geladen um wieder dekrementiert werden zu können. Hat der **Counter** den gleichen Wert wie das **Pulse Width Register**, so geht der **Comparator**-Ausgang auf High. Am Ende der Periode geht er zusammen mit TC wieder auf Low.

Die Periodendauer wird mit

$$\text{OutputPeriod} = (\text{PeriodValue} + 1) t_{\text{CLK}}$$

berechnet.

Das Tastverhältnis mit:

$$\text{DutyCycle} = \text{CompareValue} / (\text{PeriodValue} + 1)$$

Berechnen Sie für einen 8-Bit PWM, mit einem Eingangstakt von 375 Hz, ein PWM-Signal von 1,5 Hz und einem Tastverhältnis von 60 %.

PeriodValue = 249

CompareValue = 150  
(PulseWidth)

## 12) Berechnung zu SleepTimer

Mit dem Usermodul SleepTimer soll eine Programmverzögerung von 1 Sekunde realisiert werden. Geben Sie die Werte für eine Interruptrate von 8 Hz an:

Wert für SleepTimer\_SetInterval : 0,10

Wert für SleepTimer\_TickWait : 2

# Mikrorechnerpraktikum

## Versuch 2: MSR-System Teil 1

### Vorbereitung

Interfakultatives Institut für Anwendungen der Informatik

6. November 2018

Gruppe	Nachname, Vorname	Fachbereich	Datum
8			
8			

1. Welchen Nachteil hat die zeichenorientierte Übertragung.

Man kann nur Zeichen aus dem ASCII-Zeichensatz übertragen. 7

2. Zur Darstellung von Text, z.B. auf dem Hyperterminal, werden Steuerzeichen und druckbare Zeichen verwendet. Wo werden diese definiert? 7

Im ASCII-Zeichensatz

3. Wozu dienen die Steuerzeichen LF und CR? 2

LF: (~~Zeichen~~ / Zeilenverschiebung) Setze den Cursor nach links zurück

CR: Beginn einer neuen Zeile (Zeilenumbruch)

4. Warum werden bei der seriellen Übertragung eines 1-Byte Zahlenwertes mit dem Hyperterminal jeweils 2 Bytes gesendet? 1

Die Bytes werden über das Hyper mit dem Hyperterminal

hexadezimal übertragen, weshalb aus jedem Byte zwei hexadezimale Zeichen werden, die einzeln übertragen werden. 1

5. Was wird mit den Parameterdaten bei dem Befehl COM\_FGW\_SET\_PWM1 eingestellt? 1

Die Leistung der Pumpe effizienter

6. Warum hat der Befehl COM\_FGW\_GET\_PREASSURE keinen Parameter? 2

Weil er keinen benötigt, da schon implementiert ist, welcher Wert ausgelesen und ~~welcher~~ gesendet werden muss.

7. Unter Verwendung der `comlib` soll der PWM2 ein Zahlenwert von dezimal 239 gesendet werden. Wie sieht das dafür zu erzeugende Datenpaket aus? Tragen Sie Ihre Werte in die folgende Tabelle ein. Carriage Return und Line Feed sind nicht zu berücksichtigen. 2/8

Byte- folge	Eingabe am Hyper- terminal	übertragene Zahlenwerte dezimal/hex	Name der Bestandteile des Datenpackets	verwendete Variablenamen
1		2	Befehlsnummer	
2		2	Parameterlänge	
3		14 1E	Parameterdaten	
4		15 1F	Parameterdaten	
5				

*com\_recvParamLength, com\_recvPacketAvailable, com\_recvCommand, com\_recvParamData*

In der letzten Spalte tragen Sie die Namen der Variablen ein, nach dem Empfang des Datenpaketes im PSoC-Baustein verwendet werden.

8. Der Motor der verwendeten Pumpe wird laut Herstellerangaben optimal mit einem PWM-Signal mit einer Frequenz von 3,5 kHz angesteuert. Der Clock-Eingang des PWM-Modules ist mit einem Takt von  $f = \frac{24}{52}$  MHz beschaltet. Welchen Wert muss in das Period-Register des PWM-Moduls geschrieben werden, damit die vom Hersteller vorgegebene Frequenz möglichst gut erreicht wird? Informationen zum PWM-Modul finden Sie auch im Vorbereitungsblatt zu Versuch 1. 7

132

9. Schreiben Sie den Assembler Quelltext für eine Programmschleife auf (evtl. auf extra Blatt), die prüft, ob ein Datenpaket über die serielle Schnittstelle empfangen wurde. Wenn ja, dann sollen die Befehls-Nr. und die Parameterlänge als hexadezimale Zahl auf dem Display ausgegeben werden (siehe auch Versuchsteil 1). 0/5

75/22

# Mikrorechnerpraktikum

## Versuch 3: MSR-System Teil 2

### Vorbereitung

Interfakultatives Institut für Anwendungen der Informatik

2. Dezember 2014

Gruppe	Nachname, Vorname	Datum	Unterschrift
8	Wisser Christian	19.11.14	C. Wisser
P	Hohenstein, Michael	19.11.14	M. Hohenstein

1. Wovon hängt die Drehzahl des Schrittmotors im Praktikum ab?

~~über die Ausgangsrate bestimmt, also von der Laufzeit~~  
 der Schleife Wert für TIMER\_PERIOD

2. Wie wird die Drehrichtung des Schrittmotors festgelegt?

über die globale Variable int RotationDirection  
 0 = Linkslauf 1 = Rechtslauf

3. Welche Funktion hat der Timer16\_SMT in diesem Versuch?

Interrupt auslösen und um Cos-Sinusfunktionswerte über den  
 DA-Wandler mit einer festen, einstellbaren Frequenz auszugeben

4. Welche Auswirkungen hat eine zu geringe Periodendauer des Timer16\_SMT?

Dauerinterrupts und diese führen dazu, dass der  
 Mikrokontroller nicht mehr reagiert

5. Was ist ein Interrupt und durch welches Ereignis wird er in diesem Versuch ausgelöst?

eine gewollte Unterbrechung des Programmlaufs um eine Aktion<sup>2</sup> höherer Priorität durchzuführen.  
durch den Nulldurchlauf des TIMER16-SMT

6. Was ist eine ISR und wie wird sie gestartet bzw. aufgerufen?

Ein Assemblerprogramm, welches über einen Zeiger einen Wert aus der LUT<sub>2</sub> liest, diesen an den DA-Wandler übergibt und den Zeiger inkrementiert.  
Aufruf durch den Interrupt beim Nulldurchgang des TIMER16-SMT

7. Damit der Schrittmotor sich dreht, müssen an die beiden DA-Wandler ständig diskretisierte SIN-/COS-Werte übergeben werden. An welcher Stelle des Quelltextes sollten die entsprechenden Assemblerbefehle stehen, welche die DA-Wandler ansteuern?

Außerhalb der Programmschleife damit die Drehzahl nicht von der Laufzeit oder Schleife abhängt also i.d. ISR

8. Warum sollte der Interrupt-Mode des Timer16\_SMT in der SMT-Initialisierungsroutine nicht aktiviert werden?

Weil der Timer unabhängig vom Interruptmode läuft und somit die routine abbrechen könnte und sich der Motor beginnt sich zu drehen

9. Mit welchem Wert für den DA-Wandler wird am Schrittmotor eine Spannung von Null Volt erreicht, so dass er still steht?

Wert: 31

10. Wie kann der zurückgelegte Weg des Förderbandes bestimmt werden?

theoretisch über die vom Schrittmotor gefahrenen Schritte, gibt Probleme durch<sup>7</sup> überspringung bei hoher drehzahl. Also: Drehscheibe mit 180° wobei die bei durch eine Gebläseklappe geführt werden und dann einen Spannungsimpuls auslösen die am Board mit dem am Ausgang verbundenen Countermodul gezählt werden.

11. In welchen Variablen werden im PSoC-Baustein nach einem empfangenen Befehl die Befehlsnummer und die Parameterdaten abgespeichert?

~~Kommandos~~ In Variablen mit dem Präfix com\_ in<sup>2</sup>

comlib

Befehlsnr: com\_recvCommand

Parameterdaten: com\_recvParamData

# Vorbereitung zu Versuch 1 – LabVIEW

GRUPPE	NAME; VORNAME	DATUM
8	Wisser, Christian	26.11.2018

- 1) In der linken Tabellenspalte stehen die in konventionellen Programmiersprachen (C, Pascal, ...) üblichen Begriffe. Stellen Sie diesen, die in LabVIEW verwendeten Begriffe gegenüber.

Konventionelle Programmiersprache	LabVIEW
Programm-Projekt	VI
Unterprogramm	SUB-VI
Sourcecode	Blockdiagramm
Benutzeroberfläche	Frontpanel

4

- 2) Wie werden die beiden Elemente, die für die Ein- und Ausgabe von Daten zuständig sind, genannt und in welchem Fenster (siehe Hauptbestandteile eines LabVIEW Programmes) werden sie platziert?

Ein-/Ausgabeelemente	Platzierung
Beckenelement <i>controls indicators</i>	Frontpanel der Elementpalette
Anzeigeelement	Frontpanel "

2

- 3) Wie unterscheiden sich lokale und globale Variablen hinsichtlich ihres Gültigkeitsbereich?

Lokale Variable: nur innerhalb des VIs gültig

2

Globale Variable: VI übergreifend gültig

- 4) Schieberegister sind spezielle lokale Variablen. Welche weitere Einschränkung haben diese in ihrem Gültigkeitsbereich?

sie behalten ihre Gültigkeit nur innerhalb einer Schleife

werden wenn erst beim schließen der VIs wieder zurück-gesetzt

1

---bitte wenden





# Vorbereitungsf formular zu Versuch 6 (LabVIEW / Schrittmotor)

Gruppe	Nachname, Vorname	Unterschrift	Fachbereich	Datum
8	Wisse Christian	C. Wisse		10.12.18
	Holenski, Michael			

- 1) Wieviele Vollschr itte sind f ur eine Rotorumdrehung des Schrittmotors notwendig?

200 Vollschr itte

1

- 2) In maximal wieviele Teilschr itte kann eine Rotorumdrehung im Praktikum unterteilt werden?

1 Vollschr it = max 8 Teilschr itte

1600 TS

1

- 3) Wieviel Lichtschrankenimpulse werden gez hlt, wenn sich das F rderband um 5,969 Meter fortbewegt?

17907

1

- 4) Mit der Ausgaberate (update-rate) f ur den D/A-Wandler wird die Drehzahl des Motors festgelegt. Erstellen Sie die Formel zur Berechnung der Ausgaberate  $r$  [1/sec] aus der Drehzahl  $u$  [1/min].

~~$r = \frac{u}{60}$~~

~~$r = \frac{u}{60} \cdot \frac{1600}{200}$~~

~~$r = \frac{u}{60} \cdot \frac{1600}{200}$~~

$r = \frac{1600}{60} u$

2

- 5) Erstellen Sie eine Formel zur Umrechnung der Wegl nge  $S$  [mm] in Anzahl von Lichtschrankenimpulse  $P_{max}$ .

$S: 60_{min} \cdot 180 = P_{max}$

2

- 6) Erstellen Sie eine Formel zur Umrechnung der Ausgaberate  $r$  [1/sec] in den Teilerwert TW f ur den Timer16\_SMT

~~$r = \frac{f_i}{r}$~~

$tw = \frac{f_i}{r} - 1$

1

- 7) Der Puffer mit den diskretisierten Sinus-/Cosinuswerten wird mit der gew nschten Ausgaberate (update-rate) zyklisch ausgelesen und an den D/A-Wandler ubergeben. Hat die Anzahl der gespeicherten Sinusperioden (je 32 diskretisierte Sinuswerte pro Periode) einen Einfluss auf die Motordrehzahl?

Ja die update-rate bestimmt die Drehzahl des Motors  
weniger diskretisierte Werte = schneller weniger Teilschr itte, mehr TS = langsamer

0/1

- 8) Der Schrittmotor soll ub er eine Rampenfunktion die vorgegebene Wegstrecke  $S$  [mm] zur cklegen. Erstellen Sie eine Formel oder ein kleines Programm, das die Drehzahl  $u$  [1/min] in Abh ngigkeit der Lichtschrankenimpulse  $p$  (anstatt des zur ckgelegten Weges) berechnet. Als Eingabeparameter verwenden Sie :

$U_{max}$  maximale Drehzahl

$U_{start}$  Startdrehzahl

$P_{ges}$  Summe der Lichtschrankenimpulse der gew nschten Wegstrecke

~~$(P_{ges} : 3,6) : 50 P_{ges} = \frac{P_{ges}}{50}$~~

~~$\frac{P_{ges}}{180} = u$~~

$u = \begin{cases} \frac{P_{ges}}{p} \cdot U_{max}, & 0 \leq p < P_{ges} \\ 0, & p \geq P_{ges} \end{cases}$

3

12/12

$$r$$

$$1P = 4Vs$$

$$200Vs = 1u$$

$$50p = 1u$$

$$r = \frac{1}{50} u \quad \frac{1}{min} \quad \frac{1}{60s}$$

$$r = \frac{1}{3000} u$$

$$1 \frac{u}{min} \Rightarrow 200Vs \text{ in } 1min$$

$$3,3Vs \text{ pro Sekunde}$$

$$1p = 4Vs$$

~~200Vs~~

$$0,825p$$

$$3,3 \text{ pro}$$

$$P_{ges} = 3,6 = \text{Anzahl an } P$$

$$50 = 1u/min$$

$$1 \frac{u}{min} = 3,3 \text{ pro}$$

$$200 Vs$$

$$180 \text{ Löcher}$$

$$1Vs = 0,9 \text{ Löcher}$$

$$4Vs = 3,6 \text{ Löcher}$$

$$0,9 \text{ Löcher}$$

1

Aufgabe 1:

- a) `int * pt;`  
b) `int * pt = &t1;`  
c) `*pt = 5;`  
d) `t2 = *pt;`

4

Aufgabe 2:

Header-Dateien sind Programmbibliotheken, die zur Auslagerung von Quelltext in eine Datei verwendet werden. Sie werden von dem Präprozessor vor dem Kompilieren an die Stelle des Befehls zum Einbinden eingefügt.

2

Aufgabe 3:

`#include <...>` // für installierte Bibliotheken  
`#include "..."` // für Bibliotheken / Dateien im

Arbeitsverzeichnis

1

Aufgabe 4:

- `private` : für Routinen innerhalb der Klasse
- `public` : für Routinen außerhalb der Klasse
- `protected` : für abgeleitete Klassen

3

Aufgabe 5:

1

CBase test;

Aufgabe 6:

1

fest. pub = \*Wert;

Aufgabe 7:

1

über einen Konstruktor

Aufgabe 8:

Ja, man kann konstruktoren überladen.

1

Aufgabe 9:

Der Parameterkonstruktor erwartet  
bei der Initialisierung der Instanz  
einen Parameter.

1

Aufgabe 10:

7

```
CPunkt::CPunkt (int x, int y) {
```

```
    priv-x = x;
```

```
    priv-y = y;
```

```
}
```

```
CKreis::CKreis (int x, int y, int r, string farbe) {
```

```
CKreis::CKreis (int x, int y, int r, string farbe): CPunkt(x, y) {
```

```
    priv-r = r;
```

```
    priv-farbe = farbe;
```

```
}
```