Änderungen 2:

| Ort | Original | Änderung |
|---|---|---|
| Seite 14 zweiter Absatz | Zu Beginn werden, wie üblich, die verwendeten Register gesichern. Danach werden die Messwerte, welche sich in ADCH befinden in das Register R28 gespeichert. Danach wird geprüft ob ADCC = 1 ist. Hierzu wird das 0. Bit ausgelesen. | Zu Beginn werden, wie üblich, die verwendeten Register gesichern. Danach werden die Messwerte, welche sich in ADCH befinden in das Register R28 gespeichert.<br><br>Als nächstes erfolgt ein rechts schieben des adcc (ADC Counter) um über das Carry-Flag eine Prüfung des 0.Bits vorzunehmen. |
| Seite 14 3. Absatz | Sollte dies der Fall sein werden die Bits im ADCC nach links geschoben und so ein 2 erzeugt. Danach wird das Messergebnis als Zwischenergebnis links (zel) abgespeichert. Danach wird die Front LED abgeschalten.<br>In Folge (ab adci_4) wird der ADC Interrupt deaktiviert und dem Timer0 wird eine Wartezeit zugeteilt. Diese dient dazu abzuwarten bis die LED vollständig erleuchtet, bzw. erloschen ist. Durch die Differenz aus den Werten mit LED und ohne LED kann das Umgebungslicht herausgerechnet werden und damit auf ein abkleben der Messeinrichtung verzichtet | Ist dieses gesetzt wird das adcc auf den Wert 2 gesetzt, um damit anzuzeigen, dass beim nächsten Aufruf des ADCI der Messwert für die Messung 2 vorliegt.<br><br>Danach wird das Messergebnis als Zwischenergebnis links (zel) abgespeichert. Danach wird die Front LED abgeschalten. Anschließen erfolgt ein Sprung nach adci_4. Dort wird der ADC Interrupt deaktiviert und der Timer erneut initialisiert. Dies dient dazu abzuwarten bis die LED vollständig erleuchtet, bzw. erloschen ist. Durch die Differenz aus den Werten mit LED und ohne LED kann das Umgebungslicht herausgerechnet werden und damit auf ein abkleben der |

| | | |
|---|---|---|
| | werden. | Messeinrichtung verzichtet werden. |
| Seite 14 4. Absatz | Sollte in ADCC 2 stehen zeigt dies die zweite Messung an. Der Zähler wird so eingestellt das als nächstes die 3. Messung folgt und die oben angesprochene Subtraktion wird durchgeführt. | Sollte in adcc der Wert 2 stehen zeigt dies die zweite Messung an. Der Zähler wird nun auf den Wert 4 gesetzt, damit beim nächsten Aufruf von ADCI der Messwert der Messauswertung 3 zugeführt wird.<br><br>In der 2. Messauswertung wird die oben angesprochene Subtraktion durchgeführt. |
| Seite 15 1 Absatz | Zu Beginn der 3. Messung wird der ADC Zähler auf 8 gesetzt. Die Verdoppelung der Werte im Zähler wird durch ein Logical Shift Left realisiert. Der Messwert wird nun in den Zwischenspeicher Rechts (zer) gespeichert und die Front LED wieder abgeschalten. | Zu Beginn der 3. Messung wird der adcc auf den Wert 8 gesetzt. Der Messwert wird nun in den Zwischenspeicher Rechts (zer) gespeichert und die Front LED wieder abgeschalten. |
| Seite 15 2. Absatz | Das letzt Messergebnis liefert den Messwert des rechten Sensors ohne LED. Wie bei den linken LEDs wird auch hier wieder die Differenz aus den beiden Werten gebildet. | Das letzte Messergebnis liefert den Messwert des rechten Sensors ohne LED. Wie bei dem linken Sensor wird auch hier wieder die Differenz aus den beiden Werten gebildet. |
| Seite 15 3. Absatz | Anschließend muss der Ausgangszustand der Messauswertung wieder hergestellt werden. Dazu wird wieder auf den linken Sensor gestellt und die Front LED angeschalten. | Anschließend wird der ADC wieder auf den linken Sensor eingestellt, adcc auf 1 gesetzt und die Front-LED eingeschaltet, um einen neuen Messzyklus einzuleiten. Bevor der neue Messzyklus gestartet wird, wird noch die Gesamtauswertung durchgeführt. |

| | | |
|---|---|---|
| Seite 15 4. Absatz | Die Differenzwerte (mit und ohne LED) werden in die Messergebnisse rechts und links (mr und ml)geschrieben. Zudem wird die Differenz zwischen den Messergebnissen des rechten und linken Sensors gebildet. Hierzu wird der Drift (zwischen den Motoren, bzw. Rädern) gerechnet. Dieser Drift ist ein vordefinierter Wert und hängt von der jeweiligen Geräteeigenschaften des Asuros ab. | Dazu werden die Differenzwerte (mit und ohne LED) in die Messergebnissregister rechts und links (mr und ml) geschrieben. Zudem wird die Differenz zwischen den Messergebnissen des rechten und linken Sensors gebildet. Hierzu wird der Drift (zwischen den Motoren, bzw. Rädern) mit eingerechnet. Dieser Drift ist ein vordefinierter Wert und hängt von der jeweiligen Geräteeigenschaften des Asuros ab. Die Gleichung für die Bildung des Differenzwertes ist somit:<br><br>diff = ml – mr + drifft |
| Seite 16 1. Absatz | Beim Transmit, also dem Senden, wird nach dem Sichern der Register überprüft ob überhaupt ein Zeichen zum Senden bereit steht. Sollt dies der Fall sein wird in TRANSMIT_ENABLE gesprungen: | Bei dem Unterprogramm „TRANSMIT“ handelt es sich um das Senden, der in „RECEIVE“ empfangenen Nachricht an den Empfänger.<br><br>Nach dem Sichern der Register wird überprüft ob ein Zeichen zum Senden bereit steht. Sollte dies der Fall sein wird in TRANSMIT_ENABLE gesprungen: |
| Seite 16 letzter Absatz | Hier wird das Empfangen ausgeschalten. Genauso wie die Interrupts bei der Benachrichtigung für ein empfangenes oder gesendetes Zeichen. Zudem wird die Übertragung aktiviert.<br>Zurück in Transmit wird der Zähler auf die Anzahl der gespeicherten Zeichen gesetzt. Danach werden die Zeiger auf die Startadressen der zuvor empfangenen Zeichenfolge gesetzt. | Hier wird das Empfangen ausgeschalten. Genauso wie die Interrupts bei der Benachrichtigung für ein empfangenes oder gesendetes Zeichen. Zudem wird die Übertragung aktiviert.<br><br>Zurück in Transmit wird der Zähler auf die Anzahl der gespeicherten Zeichen gesetzt. Danach wird der Zeiger (Z) auf die Startadresse der zuvor empfangenen |

| | | |
|---|---|---|
| | Nun wird TX-Sequence (siehe nächste Seite) geöffnet, um die Startsequence zu senden.<br>Danach können die Zeichen der Nachricht übermittelt werden. Hierzu muss zunächst solange gewartet werden bis in den Sendepuffer ein Zeichen geschrieben werden kann. Ist dies der Fall wird ein Zeichen (der Reihenfolge nach) in UDR geschrieben und übertragen und der Zähler für die Anzahl der Zeichen wird um 1 zurückgesetzt.<br>Nun wird erneut TX_Sequence aufgerufen, um die Endsequence zu senden. Die Register werden wieder hergestellt und Transmit beendet. | Zeichenfolge gesetzt.<br><br>Nun wird die Startsequence in drei Register abgelegt, damit diese durch das Unterprogramm TX-Sequence (siehe nächste Seite) gesendet wird. Dazu wird nun das Unterprogramm TX-Sequence aufgerufen.<br><br>Danach werden die Zeichen der Nachricht gesendet. Dies erfolgt in einer Schleife, welche zunächst prüft ob der der Sendepuffer beschrieben werden kann. Ist dies der Fall wird aus dem Speicher ein Zeichen geladen und im Sendepuffer abgelegt. Anschließend wird der Zähler dekrementiert und geprüft ob dieser bei 0 angekommen ist. Ist dies nicht der Fall wird die Schleife erneut durchlaufen und das nächste Zeichen gesendet usw.<br><br>Ist der Zähler bei 0 angekommen wird die Endsequence in drei Register abgelegt und mittels des Unterprogrammes TX_Sequence gesendet. Dazu wird als nächstes TX-Sequence aufgerufen.<br><br>Nach dem aussenden der Endsequence werden noch die am Anfang gesicherten Register wiederhergestellt und es erfolgt der Rücksprung in das Hauptprogramm (MAIN). |
| Seite 17 1. Absatz | TX_Sequence sichert die Register R16, count und die Register R20, R21, R22 in welchen End- oder Startsequenz | TX-Sequence sichert zunächst die verwendeten Register (R16,count). Anschließend werden auch die Register R20,R21,R22 , mit der Start- / Endsequence, auf dem Stack abgelegt, um Sie |

| | | |
|---|---|---|
| | abgelegt sind. | später von dort wieder auszulesen. |
| Seite 17 3. Absatz | R20 bis R22 werden nun bei je einem Schleifendurchgang in R16 geschoben und das Zeichen auf dem Stack abgelegt. | Die in R20 bis R22 stehende Start-/Endsequence, welche zuvor auf dem Stack abgelegt wurde, wird nun bei je einem Schleifendurchgang der Reihe nach vom Stack ausgelesen und dem Sendepuffer zurm Senden übergeben. |
| Seite 17 4. Absatz | Sind alle 3 Register auf dem Stack werden das ursprüngliche R16 und count wieder hergestellt. | Anschließend wird der Zähler dekrementiert und geprüft ob er 0 erreicht hat. Hat er Null erreicht wurden alle drei Zeichen der Start-/Endsequence gesendet und die am Anfang von TX_Sequence gesicherten Register (R16, count) werden wieder hersgestellt. |
| Seite 18 2. Absatz | An eine Grenze stößt das Projekt im Bereich der zu speichernden Zeichen. Die maximale Anzahl von Zeichen wären 1000. Dies gilt jedoch nur für den Fall, dass nichts weiter gespeichert würde. Allerdings nutzt auch der Stack den Speicher, genauso wie die Tabelle für die Reglerwerte. | An eine Grenze stößt das Projekt im Bereich der zu speichernden Zeichen. Die maximale Anzahl von Zeichen wird durch die Größe des Speichers begrenzt. Da auch die Tabellen mit den Reglerwerten im Speicher abgelegt werden müssen, wird dieser dadurch noch weiter begrenzt. Ein weiteres Problem stellt der Stack dar, der den gleichen Speicherbereich nutzt. Wodurch man die push und pop – Befehle im Programm zählen müsste, um hier eine Aussage zu treffen, wieviel Speicher noch für die Zeichenfolge übrig bleibt, da aber ein Programm auch erweiterbar sein soll und somit die Anzahl der hinzukommenden zu speichernden Elemente nicht klar ist, haben wir uns auf eine Zeichenanzahl von 250 Zeichen, also 250 Byte festgelegt, welche für eine kurze Nachricht ausreichend sein dürfte. |