hochschule mannheim

Fakultät Elektrotechnik
Studiengang Energietechnik
und erneuerbare Energien

Laborbericht

Mini-Taschenrechner

Johannes Wilhelm

Vorgelegt von	Johannes Wilhelm
am	18. Mai 2021

Inhaltsverzeichnis

Kapitel 1

Aufgabenstellung

Nach dem Einschalten/Reset leuchtet die grüne LED und signalisiert die Eingabebereitschaft
- Der Benutzer drückt n-mal den Taster S1, um somit die erste Zahl n einzugeben
- Danach drückt der Benutzer den Taster S2, um somit das „+-Zeichen“ einzugeben
- Danach drückt der Benutzer m-mal den Taster S1, um die zweite Zahl m einzugeben
- Danach drückt der Benutzer wieder S2, um somit „=“ einzugeben
- Der Mikrocontroller addiert n + m und gibt das Ergebnis durch (n+m)-maliges Blinken der roten LED aus
- Danach schaltet der Taschenrechner wieder in den Bereitschaftsmodus mit der grünen LED und der gleiche Ablauf beginnt erneut

Kapitel 2

Programmplanung

Beim Programmstart wird das Interruptsystem aktiviert und von den beiden Buttons S1 und S2 getriggert. Die grüne LED symbolisiert zu beginn Eingabebereitschaft für den ersten Summand.
Der Programmstatus definiert welcher Summand gerade eingegeben werden soll. Nach Eingabe des ersten Summanden wird er auf 1 gesetzt und nach der Ausgabe wieder auf 0. Nach Eingabe des zweiten Summand erlischt die grüne LED und es wird eine Schleife aufgerufen welche sich so oft wie die Summe wiederholt. Bei jedem Durchgang wird die Rote LED ein und nach einem kurzen Delay wieder ausgeschaltet.
Nach der Schleife werden alle Register zurückgesetzt und das Programm beginnt von neuem.

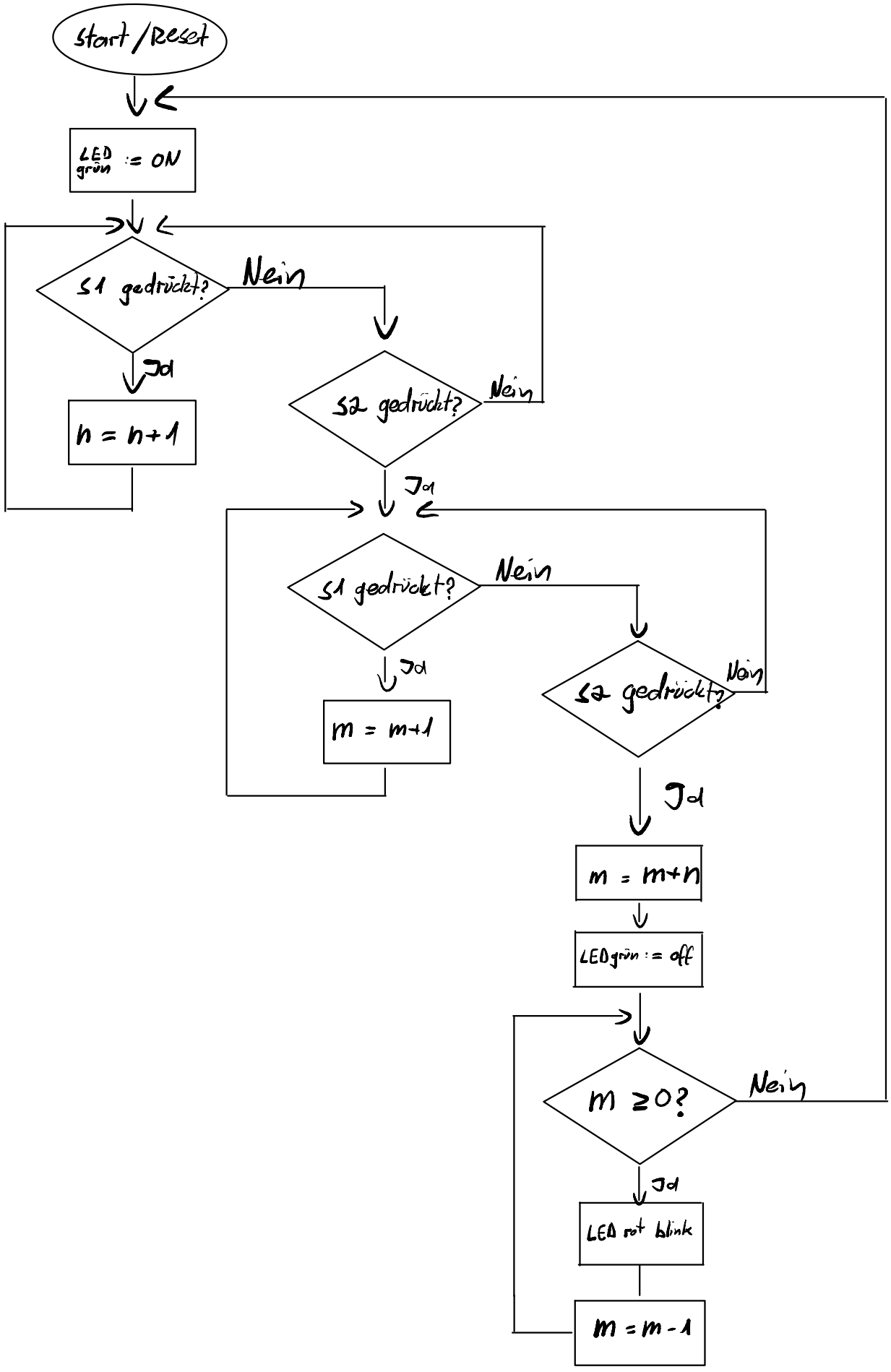
Register	Verwendung	Anfangswert
R5	erster Summand	0x00
R6	zweiter Summand	0x00
R8	Programmstatus	0x00
R15	Schleifenzähler	

Tabelle 2.1: Register

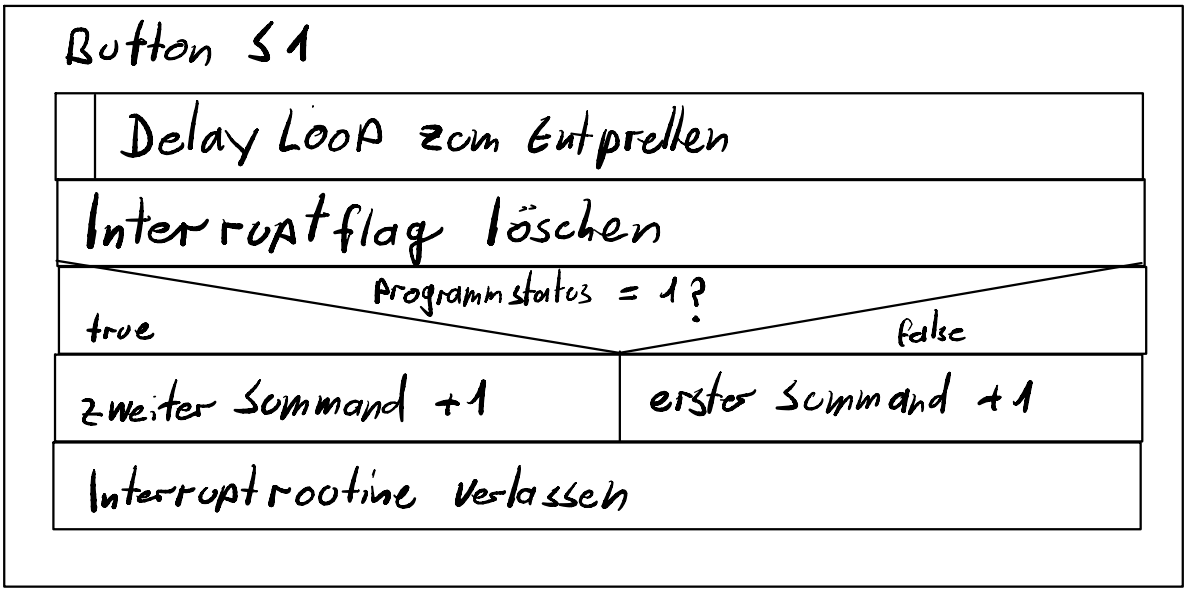
Funktion	Port	Register	Interrupt Vektor	Bemerkung
Button S1	P2.1	P2IFG	0FFD4h	trigger auf negativer Flanke
Button S2	P1.1	P1IFG	0FFDEh	trigger auf negativer Flanke

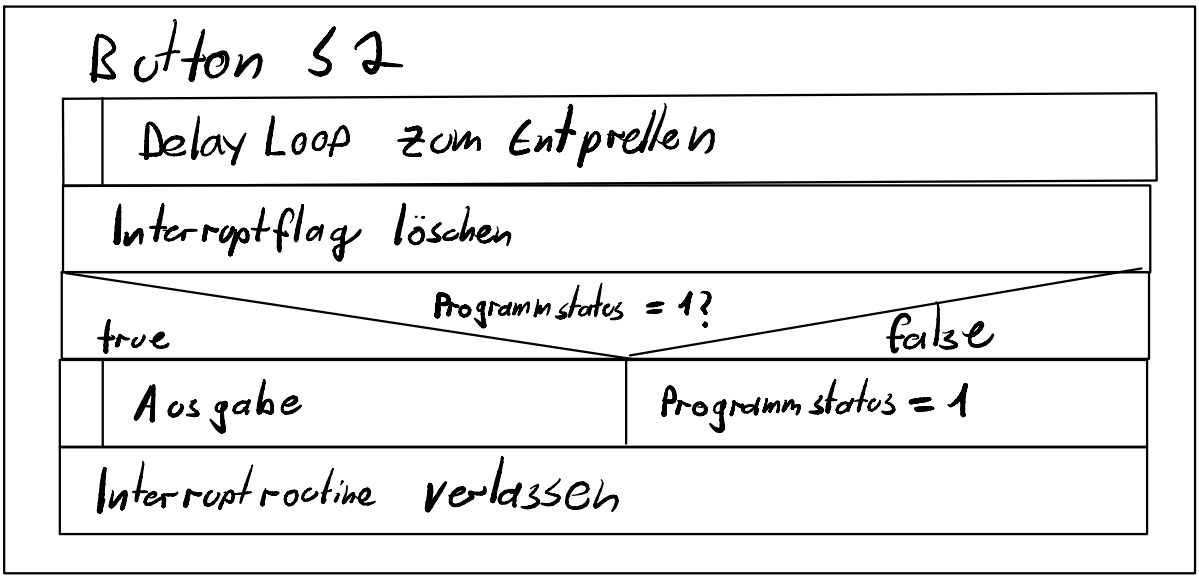
Tabelle 2.2: Interrupts

Labor 1 Mini Taschenrechner



Interrupts


Button S1
Delay Loop zum Entprellen
Interruptflag löschen
Programmstatus = 1?
true
false
zweiter Sommand +1
erster Sommand +1
Interruptroutine verlassen


Button S2
Delay Loop zum Entprellen
Interruptflag löschen
Programmstatus = 1?
true
false
Ausgabe
Programmstatus = 1
Interruptroutine verlassen

Funktionen

Ausgabe

Summanden addieren $n + m = x$

Grüne LED = off

bis die Summe $x = 0$ ist

- Rote LED = on
- Delay Loop
- Rote LED = off
- Delay Loop
- x um 1 verringern

alle Register auf Anfangswert zurücksetzen

DelayLoop

bis Schleifenzähler = 0

- Schleifenzähler um 1 verringern

Kapitel 3

Quellcode

```
1  ;
       ********************************************************************
2  ;  Mini Taschenrechner
3  ;
4  ;  ber  den Button S1 werden zwei Zahlen eingegeben, welche
       anschlieend summiert und nach dem bettigen von Button S2
       durch Blinken ausgegeben werden.
5  ;
6  ;  Johannes Wilhelm
7  ;  Hochschule Mannheim
8  ;  Built with IAR Embedded Workbench
9  ;
       ********************************************************************
10
11 #include "msp430f5529.h"
12
13 Button_S1   SET   0x02                  ; Taster an Port P2.1
14 Button_S2   SET   0x02                  ; Taster an Port P1.1
        (der Name S2 stammt aus dem Schaltplan des LaunchPad,
       siehe LaunchPad-Userguide "slau533b" auf Seite 57)
15
16 ;
       --------------------------------------------------------------------
```

```
17          ORG   04400h            ; Progam Start, Adresse aus
                Tabelle im Datenblatt Seite 22, rechte Spalte (fr
                 MSP430F5529)
18  ;
    -------------------------------------------------------------------------

19  RESET       MOV.w #04400h,SP         ; Stackpointer
    initialisieren, der Stack wchst von oben nach unten !
20  StopWDT     MOV.w #WDTPW+WDTHOLD,&WDTCTL ; Watchdog Timer
    anhalten
21
22  ; +++ Konfiguration der IO-Ports und der Port-Interrupts +++
23
24  Port_1
25              MOV.b #0x00, &P1IFG ; Alle P1-IFG's lschen, falls
                     zufllig gesetzt
26              MOV.b #BIT0, &P1DIR ; nur P1.0 fr rote LED als
                    Ausgang konfigurieren (1=Out, 0=In), der Rest
                    des Ports sind Eingnge (der Taster Button_S2
                    hngt an P1.1)
27              MOV.b #BIT1, &P1OUT ; LED an Port 1.0 ist aus (
                    P1.0=0), PullUp an Port 1.1 fr den Button_S2 (
                    P1.1=1)
28              MOV.b #0x00, &P1SEL ; kompletter Port P1 als
                    normaler IO-Port verfgbar, nichts wird an
                    andere Peripherie abgegeben
29              MOV.b #BIT1, &P1REN ; aktiviere PullUp an P1.1 fr
                     Button_S2
30              MOV.b #0xff, &P1IES ; alle Port 1 Interrupts
                    werden auf negative Flanke getriggert (das ist
                     so, weil die Taster auf dem LaunchPad nach
                    Masse gehen)
31              MOV.b #Button_S2, &P1IE ; Nur Taster "Button_S2"
                    fr Interrupt freigeben, alle anderen
                    Interruptflags von Port 1 unterdrcken
32
33  PORT_2
34              MOV.b #0x00, &P2IFG ; Alle P2-IFG's lschen, falls
                     zufllig gesetzt
35              MOV.b #0x00, &P2DIR ; alle Ports sind Eingnge
```

```
36              MOV.b #BIT1, &P2OUT ; PullUp an Port 2.1 fr den
                   Button_S1 (P2.1=1)
37              MOV.b #0x00, &P2SEL ; kompletter Port P1 als
                   normaler IO-Port verfgbar, nichts wird an
                   andere Peripherie abgegeben
38              MOV.b #BIT1, &P2REN ; aktiviere PullUp an P1.1 fr
                    Button_S1
39              MOV.b #0xff, &P2IES ; alle Port 1 Interrupts
                   werden auf negative Flanke getriggert (das ist
                    so, weil die Taster auf dem LaunchPad nach
                   Masse gehen)
40              MOV.b #Button_S1, &P2IE ; Nur Taster "Button_S1"
                   fr Interrupt freigeben, alle anderen
                   Interruptflags von Port 1 unterdrcken
41
42  Port_4      BIS.b #BIT7,&P4DIR      ; Port P4.7 als Ausgang
       konfigurieren fr grne LED
43              BIS.b #BIT7,&P4OUT      ; LED an Port 4.7
                   einschalten zu Programmbeginn (1 = "an")
44
45
46              MOV   #0x00, R5         ; Initialisierung erster
                   Summand
47              MOV   #0x00, R6         ; Initialisierung zweiter
                    Summand
48              MOV   #0x00, R8         ; Initialisierung
                   Programmstatusregister (0: Eingebe erste
                   Summand, 1: Eingabe zweiter Summand)
49
50  Main
51              BIS.W #GIE, SR          ; global Interruptsystem
                   aktivieren
52              NOP
53
54              JMP   Main              ; Endlosschleife
55              NOP                     ; sinnloser letzter
                   Befehl, damit der IAR Simulator nicht
                   meckert...
56
```

```
57 ;+++ ISR
      ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
58
59 ; +++ Button_S2 +++
60 PORT1_ISR
61
62              MOV.w #50000,R15        ; initialisiere den
                   Schleifenzhler R15 mit dem Startwert 50000
63              CALL  #DelayLoop        ; Schleife als Verzgerung
                    zum Entprellen
64
65              BIC.b #BIT1, &P1IFG     ; das gesetzte
                   Interruptflag lschen, sonst wrde ISR sofort
                   wieder neu ausgelst werden
66
67              CMP   #0x01, R8         ; Vergleiche ob R8<1 ist,
                    also ob der zweite Summand schon eingegeben
                   wurde
68              JZ AUSGABE              ; Wenn ja, springe zur
                   Ausgabe
69
70              MOV   #0x01, R8         ; Wenn nein, nder
                   Programmstatus fr Eingabe des zweiten
                   Summanden
71              RETI                    ; beende Interruptroutine
72
73 ; Ausgabe der Summe durch Blinken der roten LED
74
75 AUSGABE      ADD   R5, R6            ; Addiere die zwei
      Summanten R5+R6 und speicher das Ergebnis in R6
76              BIC.b #BIT7,&P4OUT      ; keine
                   Eingebebereitschaft, Grne LED AUS
77
78 BLINK        CMP   #0x00, R6         ; Vergleiche ob das
      Ergebnis in R6 Null ist
79              JZ RESTART              ; wenn Null: Neustart
80
81              BIS.b #BIT0,&P1OUT      ; LED Rot AN
82
```

```
83              MOV.w #50000,R15        ; initialisiere den
                   Schleifenzhler R15 mit dem Startwert 50000
84              CALL #DelayLoop         ; Verzgerungsschleife
                   aufrufen damit LED nicht sofort aus geht
85
86              BIC.b #BIT0,&P1OUT      ; LED Rot AUS
87
88              MOV.w #50000,R15        ; initialisiere den
                   Schleifenzhler R15 mit dem Startwert 50000
89              CALL #DelayLoop         ; Verzgerungsschleife
                   aufrufen damit LED nicht sofort an geht
90
91              DEC   R6                ; Ergebnis runterzhlen (
                   so oft wird noch geblinkt)
92              JMP BLINK               ; Wiederhole das Blinken
93
94  ; Neustart des Programm nach Ausgabe
95
96  RESTART     MOV   #0x00, R5             ; Erste Summand auf
       null
97              MOV   #0x00, R6             ; Zweiter Summand auf
                    null
98
99              MOV   #0x00, R8             ;
                   Programmstatusregister auf null
100
101             BIS.b #BIT7,&P4OUT          ; Eingebebereitschaft,
                    grne LED AN
102             RETI                        ; beende
                   Interruptroutine
103
104
105 ; +++ Button_S1 +++
106
107 PORT2_ISR
108
109             MOV.w #50000,R15        ; initialisiere den
                   Schleifenzhler R15 mit dem Startwert 50000
110             CALL  #DelayLoop        ; Schleife als Verzgerung
                    zum Entprellen
```

```
111
112
113             BIC.b #BIT1, &P2IFG     ; das gesetzte
                   Interruptflag lschen, sonst wrde ISR sofort
                   wieder neu ausgelst werden
114
115             CMP   #0x01, R8         ; Vergleiche den
                   aktuellen Programmstatus
116             JZ    NEXT              ; Springe zu NEXT wenn
                   der zweite Summand eingegeben wird
117
118             INC   R5                ; sonnst erhhe den ersten
                    Summand um eins
119             RETI                    ; beende Interruptroutine
120
121 NEXT        INC   R6                ; erhhe den zweiten
    Summand um eins
122             RETI                    ; eine Interruptroutine
                   muss immer mit dem Befehl RETI abgeschlossen
                   werden
123
124
125 ; +++ Funktionen +++
126
127 ; Verzgerungsschleife
128
129 DelayLoop   DEC   R15              ; Register R5 um 1
    verringern
130             JNZ DelayLoop           ; Wiederholung bis R15=0
131             RET                     ; Zurck zum
                   Funktionsaufruf
132
133 ;
    ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
134
135 ;
    ----------------------------------------------------------------------
136 ;       Interrupt Vectors
```

```
137 ;-------------------------------------------------------------------------------

138         ORG   0FFFEh            ; MSP430 RESET Vector
139         DW    RESET
140
141         ORG   0FFDEh            ;Interrupt Vektor fr die
                Flags in Register P1IFG
142         DW    PORT1_ISR         ;die Interrupt Vektor
                Adresse 0xFFDE steht im Datenblatt des
                MSP430F5529 auf Seite 21 in der Tabelle
143
144         ORG   0FFD4h            ;Interrupt Vektor fr die
                Flags in Register P2IFG
145         DW    PORT2_ISR
146         END
```