Power manager circuit driver for a handy power-pack over ATtiny4

1.1

Generated by Doxygen 1.8.18

Chapter 1

File Index

1.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 2

File Documentation

2.1 ATtiny4.h File Reference

This header file contains the important definitions for ATtiny4 MCU.

Macros

- #define **SET_BIT**(REG, BIT) REG |= (1<<BIT) /*Sets the bit value to 1*/
- #define **CLEAR_BIT**(REG, BIT) REG &= ~(1<<BIT) /*Clears the bit value to 0*/
- #define **GET_BIT**(REG, BIT) ((REG >> BIT) & 0x01) /*Get the bit value*/
- #define **SREG** (*(volatile u8_t*)(0x3F))
- #define **EICRA** (*(volatile u8_t*)(0x15))
- #define **EIFR** (*(volatile u8_t*)(0x14))
- #define **EIMSK** (*(volatile u8_t*)(0x13))
- #define **PUEB** (*(volatile u8_t*)(0x03))
- #define **PORTB** (*(volatile u8_t*)(0x02))
- #define **DDRB** (*(volatile u8_t*)(0x01))
- #define **PINB** (*(volatile u8_t*)(0x00))
- #define **TCCR0** (*(volatile u16_t*)(0x2D))
- #define **TCNT0** (*(volatile u16_t*)(0x28))
- #define **OCR0A** (*(volatile u16_t*)(0x26))
- #define **TIMSK0** (*(volatile u8_t*)(0x2B))
- #define **SMCR** (*(volatile u8_t*)(0x3A))
- #define **CLKMSR** (*(volatile u8_t*)(0x37))
- #define **CLKPSR** (*(volatile u8_t*)(0x36))
- #define **CCP** (*(volatile u8_t*)(0x3C))
- #define **SREG_IBIT** (7)
- #define **PORTB_PB0** (0)
- #define **PORTB_PB1** (1)
- #define **PORTB_PB2** (2)
- #define **PORTB_PB3** (3)
- #define **DDRB_PB0** (0)
- #define **DDRB_PB1** (1)
- #define **DDRB_PB2** (2)
- #define **DDRB_PB3** (3)
- #define **PINB_PB0** (0)
- #define **PINB_PB1** (1)
- #define **PINB_PB2** (2)
- #define **PINB_PB3** (3)
- #define **EIFR_INTF0** (0)
- #define **SMCR_SE** (0)

Typedefs

- typedef unsigned char **u8_t**
- typedef unsigned short **u16_t**

2.1.1 Detailed Description

This header file contains the important definitions for ATtiny4 MCU.

Author

Ahmed Ashraf (ahmedashrafelnaqeeb@gmail.com)

Version

1.1

Date

2020-07-13

Copyright

2.2 Functionality.c File Reference

This file contains the interfacing functions logic implementation for the power manager application.

```
#include "ATtiny4.h"
#include "Functionality.h"
```

Macros

- #define **EXTI0_ENABLE** (0x01)
- #define **EXTI0_DISABLE** (0x00)
- #define **EXTI0_LOW_LEVEL_TRIGGER** (0x00)
- #define **EXTI0_FALLING_EDGE_TRIGGER** (0x02)
- #define **TIMER0_CTC_MODE_SELECTION** (0x0008)
- #define **TIMER0_50MS_TICK** (50000)
- #define **TIMER0_PRESCALER_8** (0x0002)
- #define **TIMER0_CLEAR_PRESCALER** (0xFFF8)
- #define **TIMER0_OCR0A_INT_EN** (0x02)
- #define **IO_PINS_DIR_INITIALIZATION** (0x01)
- #define **IO_LOW_LEVEL** (0)
- #define **IO_HIGH_LEVEL** (1)
- #define **IO_PB2_PULLUP_ACTIVATE** (0x04)
- #define **POWER_DOWN_MODE_SELECTION** (0x04)

- #define **SYSTEM_OFF_STATUS** (0)
- #define **SYSTEM_ON_STATUS** (1)
- #define **NO_VOLTAGE_PRESENT** (0)
- #define **NO_RESIDUAL_CHARGE** (1)
- #define **VOLTAGE_CHECKING_TRIALS** (2)
- #define **TWO_SEC_DELAY** (2000)
- #define **ONE_MS_DELAY** (2000)
- #define **ONE_SECOND** (20)
- #define **TWO_SECONDS** (40)
- #define **THREE_SECONDS** (60)
- #define **TEN_SECONDS** (200)
- #define **INTERNAL_OSC_SELECT_8MZ** (0x00)
- #define **ENABLE_CHANGE_FOR_IO_REG** (0xD8)
- #define **MAIN_CLK_PRESCALING_BY_1** (0x00)

Functions

- void attiny4_init (void)

  This function is responsible for initializing the ATtiny MCU and activate the power down mode.
- void mainApplication (void)

  This function is responsible for applying the state machine of the power manager system and making a transition from state to another

- void **EXTI0_ISR** (void)
- void **OCR0A_ISR** (void)

Variables

- u8_t **gu8_systemStatus** = 0
- u16_t **gu16_switchCounter** = 0
- u16_t **gu16_checkCounter** = 0
- u8_t **gu8_voltageCheckTrials** = 0

2.2.1 Detailed Description

This file contains the interfacing functions logic implementation for the power manager application.

Author

Ahmed Ashraf (ahmedashrafelnaqeeb@gmail.com)

Version

1.1

Date

2020-07-13

Copyright

Copyright (c) 2020

2.2.2 Function Documentation

2.2.2.1 attiny4_init()

```
void attiny4_init (
             void  )
```

This function is responsible for initializing the ATtiny MCU and activate the power down mode.

Adjusting the MCU CLK section

External interrupt initialization section

Timer initialization section

DIO initialization section

IO Pins initialization by: PB0 -> Output PB1 -> Input PB2 -> Input PB3 -> Input

Activating power down mode

Definition at line 72 of file Functionality.c.

```
73 {
78     /*Disable global interrupts*/
79     CLEAR_BIT(SREG , SREG_IBIT);
80
81     /*Select the internal oscillator of the MCU with 8MHz*/
82     CLKMSR = INTERNAL_OSC_SELECT_8MZ;
83
84     /*Enable writing to the CLKPSR register*/
85     CCP = ENABLE_CHANGE_FOR_IO_REG;
86
87     /*Enable the pre-scaler of the main CLK by 1*/
88     CLKPSR = MAIN_CLK_PRESCALING_BY_1;
89
90
95     /*Disable external interrupt0 (EXTI0)*/
96     EIMSK = EXTI0_ENABLE;
97
98     /*Selecting low level as interrupt trigger*/
99     EICRA = EXTI0_LOW_LEVEL_TRIGGER;
100
101     /*Clear EXTI0 flag*/
102     SET_BIT(EIFR , EIFR_INTF0);
103
104     /*Enable global interrupts*/
105     SET_BIT(SREG , SREG_IBIT);
106
107
112     /*Selecting CTC mode with OCR0A*/
113     TCCR0 = TIMER0_CTC_MODE_SELECTION;
114
115     /*Clearing timer/counter register*/
116     TCNT0 = 0;
117
118     /*Adjusting TIMER0 to fire CTC interrupt every 10ms for 8MHz frequency and prescaler by 8*/
119     OCR0A = TIMER0_50MS_TICK;
120
121     /*Enable CTC mode interrupt*/
122     TIMSK0 = TIMER0_OCR0A_INT_EN;
123
135     DDRB = IO_PINS_DIR_INITIALIZATION;
136
137     /*Activating the pull up resistor for PB2*/
138     PUEB = IO_PB2_PULLUP_ACTIVATE;
139
140     /*Set PB0 to logic zero*/
141     CLEAR_BIT(PORTB , PORTB_PB0);
142
143
148     /*Select the power down mode*/
```

```
149     SMCR = POWER_DOWN_MODE_SELECTION;
150
151     /*Sleep enable*/
152     SET_BIT(SMCR , SMCR_SE);
153
154     /*Execute sleep instruction*/
155     __asm__ __volatile__ ( "sleep" "\n\t" :: );
156
157     return;
158 }
```

2.2.2.2 mainApplication()

```
void mainApplication (
             void  )
```

This function is responsible for applying the state machine of the power manager system and making a transition from state to another

Definition at line 160 of file Functionality.c.

```
161 {
162     /*Applying the state machine of the system*/
163
164     /*Checking if the switch pressed for more than 10 seconds*/
165     if( gu16_switchCounter > TEN_SECONDS )
166     {
167         /*Variable used in delay operations*/
168         u16_t au16_delayVariable = TWO_SEC_DELAY;
169
170         /*Delay for two seconds*/
171         while(au16_delayVariable--)
172         {
173             /*Variable used in for looping*/
174             u16_t i = 0;
175
176             /*Software delay for 1ms approximately*/
177             for (i = 0 ; i < ONE_MS_DELAY ; i++);
178         }
179
180         /*Initialize the system again and enter power down mode*/
181         attiny4_init();
182     }
183
184     /*Checking if the switch is pressed for (1~2) seconds and the system is already in the OFF state*/
185     else if( (gu16_switchCounter >= ONE_SECOND && gu16_switchCounter <= TWO_SECONDS) &&
        (gu8_systemStatus == SYSTEM_OFF_STATUS) )
186     {
187         /*Set PB0 to high level*/
188         SET_BIT(PORTB , PORTB_PB0);
189
190         /*Report that the system has become in ON mode*/
191         gu8_systemStatus = SYSTEM_ON_STATUS;
192
193         /*Reset the voltage checking counter*/
194         gu16_checkCounter = 0;
195
196         /*Reset voltage checking trials counter*/
197         gu8_voltageCheckTrials = 0;
198     }
199
200     /*Checking if the switch is pressed for (1~2) seconds and the system is already in the ON state*/
201     else if( (gu16_switchCounter >= ONE_SECOND && gu16_switchCounter <= TWO_SECONDS) &&
        (gu8_systemStatus == SYSTEM_ON_STATUS) )
202     {
203         /*Report that the system is in OFF mode*/
204         gu8_systemStatus = SYSTEM_OFF_STATUS;
205
206         /*Initialize the system and enter power down mode*/
207         attiny4_init();
208     }
209
210     /*Checking after powering ON by 3 seconds that there's a voltage present or not and applying two
        powering up trials
211       if there's no voltage present*/
```

```
212     else if( (gu16_checkCounter >= THREE_SECONDS) && (GET_BIT(PINB , PINB_PB1) == NO_VOLTAGE_PRESENT) &&
        (gu8_voltageCheckTrials < VOLTAGE_CHECKING_TRIALS) )
213     {
214         /*Variable used in delay operations*/
215         u16_t au16_delayVariable = TWO_SEC_DELAY;
216
217         /*Disable all interrupts*/
218         CLEAR_BIT(SREG , SREG_IBIT);
219
220         /*Set PB0 to low level*/
221         CLEAR_BIT(PORTB , PORTB_PB0);
222
223         /*Delay for two seconds*/
224         while(au16_delayVariable--)
225         {
226             /*Variable used in for looping*/
227             u16_t i = 0;
228
229             /*Software delay for 1ms approximately*/
230             for (i = 0 ; i < ONE_MS_DELAY ; i++);
231         }
232
233         /*Set PB0 to high level*/
234         SET_BIT(PORTB , PORTB_PB0);
235
236         /*Reset the voltage checking counter*/
237         gu16_checkCounter = 0;
238
239         /*Increase voltage checking trials counter*/
240         gu8_voltageCheckTrials++;
241
242         /*Enable all interrupts*/
243         SET_BIT(SREG , SREG_IBIT);
244
245         /*Check if the system attempted two trials or not*/
246         if(gu8_voltageCheckTrials == VOLTAGE_CHECKING_TRIALS)
247         {
248             /*If the trials reached two times initialize the system and enter power down mode*/
249             attiny4_init();
250         }
251         else
252         {
253             /*Do nothing*/
254         }
255     }
256     else
257     {
258         /*Do nothing*/
259     }
260
261     return;
262 }
```

2.3 Functionality.h File Reference

This header file contains power manager interfacing functions' prototypes.

Macros

- #define **EXTI0_ISR** __vector_1
- #define **OCR0A_ISR** __vector_5

Functions

- void attiny4_init (void)

  This function is responsible for initializing the ATtiny MCU and activate the power down mode.
- void mainApplication (void)

  This function is responsible for applying the state machine of the power manager system and making a transition from state to another

2.3.1 Detailed Description

This header file contains power manager interfacing functions' prototypes.

Author

Ahmed Ashraf (ahmedashrafelnaqeeb@gmail.com)

Version

1.0

Date

2020-07-12

Copyright

2.3.2 Function Documentation

2.3.2.1 attiny4_init()

```
void attiny4_init (
             void  )
```

This function is responsible for initializing the ATtiny MCU and activate the power down mode.

Adjusting the MCU CLK section

External interrupt initialization section

Timer initialization section

DIO initialization section

IO Pins initialization by: PB0 -> Output PB1 -> Input PB2 -> Input PB3 -> Input

Activating power down mode

Definition at line 72 of file Functionality.c.

```
73 {
78     /*Disable global interrupts*/
79     CLEAR_BIT(SREG , SREG_IBIT);
80
81     /*Select the internal oscillator of the MCU with 8MHz*/
82     CLKMSR = INTERNAL_OSC_SELECT_8MZ;
83
84     /*Enable writing to the CLKPSR register*/
85     CCP = ENABLE_CHANGE_FOR_IO_REG;
86
87     /*Enable the pre-scaler of the main CLK by 1*/
88     CLKPSR = MAIN_CLK_PRESCALING_BY_1;
89
```

```
90
95      /*Disable external interrupt0 (EXTI0)*/
96      EIMSK = EXTI0_ENABLE;
97
98      /*Selecting low level as interrupt trigger*/
99      EICRA = EXTI0_LOW_LEVEL_TRIGGER;
100
101      /*Clear EXTI0 flag*/
102      SET_BIT(EIFR , EIFR_INTF0);
103
104      /*Enable global interrupts*/
105      SET_BIT(SREG , SREG_IBIT);
106
107
112      /*Selecting CTC mode with OCR0A*/
113      TCCR0 = TIMER0_CTC_MODE_SELECTION;
114
115      /*Clearing timer/counter register*/
116      TCNT0 = 0;
117
118      /*Adjusting TIMER0 to fire CTC interrupt every 10ms for 8MHz frequency and prescaler by 8*/
119      OCR0A = TIMER0_50MS_TICK;
120
121      /*Enable CTC mode interrupt*/
122      TIMSK0 = TIMER0_OCR0A_INT_EN;
123
135      DDRB = IO_PINS_DIR_INITIALIZATION;
136
137      /*Activating the pull up resistor for PB2*/
138      PUEB = IO_PB2_PULLUP_ACTIVATE;
139
140      /*Set PB0 to logic zero*/
141      CLEAR_BIT(PORTB , PORTB_PB0);
142
143
148      /*Select the power down mode*/
149      SMCR = POWER_DOWN_MODE_SELECTION;
150
151      /*Sleep enable*/
152      SET_BIT(SMCR , SMCR_SE);
153
154      /*Execute sleep instruction*/
155      __asm__ __volatile__ ( "sleep" "\n\t" :: );
156
157      return;
158 }
```

2.3.2.2 mainApplication()

```
void mainApplication (
              void  )
```

This function is responsible for applying the state machine of the power manager system and making a transition from state to another

Definition at line 160 of file Functionality.c.

```
161 {
162      /*Applying the state machine of the system*/
163
164      /*Checking if the switch pressed for more than 10 seconds*/
165      if( gu16_switchCounter > TEN_SECONDS )
166      {
167          /*Variable used in delay operations*/
168          u16_t au16_delayVariable = TWO_SEC_DELAY;
169
170          /*Delay for two seconds*/
171          while(au16_delayVariable--)
172          {
173              /*Variable used in for looping*/
174              u16_t i = 0;
175
176              /*Software delay for 1ms approximately*/
177              for (i = 0 ; i < ONE_MS_DELAY ; i++);
178          }
179
```

```
180         /*Initialize the system again and enter power down mode*/
181         attiny4_init();
182     }
183
184     /*Checking if the switch is pressed for (1~2) seconds and the system is already in the OFF state*/
185     else if( (gu16_switchCounter >= ONE_SECOND && gu16_switchCounter <= TWO_SECONDS) &&
    (gu8_systemStatus == SYSTEM_OFF_STATUS) )
186     {
187         /*Set PB0 to high level*/
188         SET_BIT(PORTB , PORTB_PB0);
189
190         /*Report that the system has become in ON mode*/
191         gu8_systemStatus = SYSTEM_ON_STATUS;
192
193         /*Reset the voltage checking counter*/
194         gu16_checkCounter = 0;
195
196         /*Reset voltage checking trials counter*/
197         gu8_voltageCheckTrials = 0;
198     }
199
200     /*Checking if the switch is pressed for (1~2) seconds and the system is already in the ON state*/
201     else if( (gu16_switchCounter >= ONE_SECOND && gu16_switchCounter <= TWO_SECONDS) &&
    (gu8_systemStatus == SYSTEM_ON_STATUS) )
202     {
203         /*Report that the system is in OFF mode*/
204         gu8_systemStatus = SYSTEM_OFF_STATUS;
205
206         /*Initialize the system and enter power down mode*/
207         attiny4_init();
208     }
209
210     /*Checking after powering ON by 3 seconds that there's a voltage present or not and applying two
    powering up trials
211        if there's no voltage present*/
212     else if( (gu16_checkCounter >= THREE_SECONDS) && (GET_BIT(PINB , PINB_PB1) == NO_VOLTAGE_PRESENT) &&
    (gu8_voltageCheckTrials < VOLTAGE_CHECKING_TRIALS) )
213     {
214         /*Variable used in delay operations*/
215         u16_t au16_delayVariable = TWO_SEC_DELAY;
216
217         /*Disable all interrupts*/
218         CLEAR_BIT(SREG , SREG_IBIT);
219
220         /*Set PB0 to low level*/
221         CLEAR_BIT(PORTB , PORTB_PB0);
222
223         /*Delay for two seconds*/
224         while(au16_delayVariable--)
225         {
226             /*Variable used in for looping*/
227             u16_t i = 0;
228
229             /*Software delay for 1ms approximately*/
230             for (i = 0 ; i < ONE_MS_DELAY ; i++);
231         }
232
233         /*Set PB0 to high level*/
234         SET_BIT(PORTB , PORTB_PB0);
235
236         /*Reset the voltage checking counter*/
237         gu16_checkCounter = 0;
238
239         /*Increase voltage checking trials counter*/
240         gu8_voltageCheckTrials++;
241
242         /*Enable all interrupts*/
243         SET_BIT(SREG , SREG_IBIT);
244
245         /*Check if the system attempted two trials or not*/
246         if(gu8_voltageCheckTrials == VOLTAGE_CHECKING_TRIALS)
247         {
248             /*If the trials reached two times initialize the system and enter power down mode*/
249             attiny4_init();
250         }
251         else
252         {
253             /*Do nothing*/
254         }
255     }
256     else
257     {
258         /*Do nothing*/
259     }
260
261     return;
262 }
```

2.4 main.c File Reference

This file contains the starting point (main function) of the power manager application.

```
#include "Functionality.h"
```

Functions

- int main (void)

  This the entry point of the power manager application.

2.4.1 Detailed Description

This file contains the starting point (main function) of the power manager application.

Author

Ahmed Ashraf (ahmedashrafelnaqeeb@gmail.com)

Version

1.0

Date

2020-07-12

Copyright

Copyright (c) 2020

2.4.2 Function Documentation

2.4.2.1 main()

```
int main (
            void  )
```

This the entry point of the power manager application.

Returns

int 0 if everything is good and another value if there's an error

Definition at line 28 of file main.c.

```
29 {
30     /*Initializing the power manager circuit*/
31     attiny4_init();
32
33     while(1)
34     {
35         /*The main operation of the power manager circuit*/
36         mainApplication();
37     }
38     return 0;
39 }
```

Index