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

1.0

Generated by Doxygen 1.8.18

1 File Index **1**

1.1 File List 1

2 File Documentation **3**

2.1 ATtiny4.h File Reference 3

2.1.1 Detailed Description 4

2.2 Functionality.c File Reference 4

2.2.1 Detailed Description 5

2.2.2 Function Documentation 5

2.2.2.1 attiny4_init() 6

2.2.2.2 mainApplication() 7

2.3 Functionality.h File Reference 8

2.3.1 Detailed Description 8

2.3.2 Function Documentation 9

2.3.2.1 attiny4_init() 9

2.3.2.2 mainApplication() 10

2.4 main.c File Reference 11

2.4.1 Detailed Description 11

2.4.2 Function Documentation 12

2.4.2.1 main() 12

Index **13**

Chapter 1

File Index

1.1 File List

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

ATtiny4.h
This header file contains the important definitions for ATtiny4 MCU 3

Functionality.c
This file contains the interfacing functions logic implementation for the power manager application 4

Functionality.h
This header file contains power manager interfacing functions' prototypes 8

main.c
This file contains the starting point (main function) of the power manager application 11

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 **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 **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

1.0

Date

2020-07-13

Copyright

Copyright (c) 2020

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** (0x08)
- #define **IO_LOW_LEVEL** (0)
- #define **IO_HIGH_LEVEL** (1)
- #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 **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)

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

1.0

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.

External interrupt initialization section

Timer initialization section

DIO initialization section

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

Activating power down mode

Definition at line 68 of file Functionality.c.

```
{
    /*Disable external interrupt0 (EXTI0)*/
    EIMSK = EXTI0_ENABLE;

    /*Selecting low level as interrupt trigger*/
    EICRA = EXTI0_LOW_LEVEL_TRIGGER;

    /*Clear EXTI0 flag*/
    SET_BIT(EIFR , EIFR_INTF0);

    /*Enable global interrupts*/
    SET_BIT(SREG , SREG_IBIT);

    /*Selecting CTC mode with OCR0A*/
    TCCR0 = TIMER0_CTC_MODE_SELECTION;

    /*Clearing timer/counter register*/
    TCNT0 = 0;

    /*Adjusting TIMER0 to fire CTC interrupt every 10ms for 8MHz frequency and prescaler by 8*/
    OCR0A = TIMER0_50MS_TICK;

    /*Enable CTC mode interrupt*/
    TIMSK0 = TIMER0_OCR0A_INT_EN;

    DDRB = IO_PINS_DIR_INITIALIZATION;

    /*Set PB3 to logic zero*/
    CLEAR_BIT(PORTB , PORTB_PB3);

    /*Activate pull-up resistor for PB0*/
    SET_BIT(PORTB , PORTB_PB0);

    /*Activate pull-up resistor for PB2*/
    SET_BIT(PORTB , PORTB_PB2);

    /*Select the power down mode*/
    SMCR = POWER_DOWN_MODE_SELECTION;

    /*Sleep enable*/
    SET_BIT(SMCR , SMCR_SE);

    /*Execute sleep instruction*/
    __asm__ __volatile__ ( "sleep" "\n\t" :: );

    return;
}
```

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 140 of file Functionality.c.

```
{
    /*Applying the state machine of the system*/

    /*Checking if there's a residual charge or not in the battery and if the switch pressed for more than 10 seconds*/
    if( (GET_BIT(PINB , PINB_PB0) == NO_RESIDUAL_CHARGE) || (gu16_switchCounter > TEN_SECONDS) )
    {
        /*Variable used in delay operations*/
        u16_t au16_delayVariable = TWO_SEC_DELAY;

        /*Delay for two seconds*/
        while(au16_delayVariable--)
        {
            /*Variable used in for looping*/
            u16_t i = 0;

            /*Software delay for 1ms approximately*/
            for (i = 0 ; i < ONE_MS_DELAY ; i++);
        }

        /*Initialize the system again and enter power down mode*/
        attiny4_init();
    }

    /*Checking if the switch is pressed for (1~2) seconds and the system is already in the OFF state*/
    else if( (gu16_switchCounter >= ONE_SECOND && gu16_switchCounter <= TWO_SECONDS) && (gu8_systemStatus == SYSTEM_OFF_STATUS) )
    {
        /*Set PB3 to high level*/
        SET_BIT(PORTB , PORTB_PB3);

        /*Report that the system has become in ON mode*/
        gu8_systemStatus = SYSTEM_ON_STATUS;

        /*Reset the voltage checking counter*/
        gu16_checkCounter = 0;

        /*Reset voltage checking trials counter*/
        gu8_voltageCheckTrials = 0;
    }

    /*Checking if the switch is pressed for (1~2) seconds and the system is already in the ON state*/
    else if( (gu16_switchCounter >= ONE_SECOND && gu16_switchCounter <= TWO_SECONDS) && (gu8_systemStatus == SYSTEM_ON_STATUS) )
    {
        /*Set PB3 to low level*/
        CLEAR_BIT(PORTB , PORTB_PB3);

        /*Report that the system is in OFF mode*/
        gu8_systemStatus = SYSTEM_OFF_STATUS;
    }

    /*Checking after powering ON by 3 seconds that there's a voltage present or not and applying two powering up trials
      if there's no voltage present*/
    else if( (gu16_checkCounter == THREE_SECONDS) && (GET_BIT(PINB , PINB_PB1) == NO_VOLTAGE_PRESENT) && (gu8_voltageCheckTrials < 2) )
    {
        /*Variable used in delay operations*/
        u16_t au16_delayVariable = TWO_SEC_DELAY;

        /*Disable all interrupts*/
        CLEAR_BIT(SREG , SREG_IBIT);

        /*Set PB3 to low level*/
        CLEAR_BIT(PORTB , PORTB_PB3);

        /*Delay for two seconds*/
        while(au16_delayVariable--)
        {
            /*Variable used in for looping*/
            u16_t i = 0;

            /*Software delay for 1ms approximately*/
            for (i = 0 ; i < ONE_MS_DELAY ; i++);
        }

        /*Set PB3 to high level*/
        SET_BIT(PORTB , PORTB_PB3);

        /*Reset the voltage checking counter*/
        gu16_checkCounter = 0;

        /*Increase voltage checking trials counter*/
        gu8_voltageCheckTrials++;

        /*Enable all interrupts*/
        SET_BIT(SREG , SREG_IBIT);
    }

    /*Any other state happens the system will initialize and power down*/
    else
    {
        /*Initialize the system again and enter power down mode*/
        attiny4_init();
    }

    return;
}
```

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

Copyright (c) 2020

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.

External interrupt initialization section

Timer initialization section

DIO initialization section

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

Activating power down mode

Definition at line 68 of file Functionality.c.

```
{
    /*Disable external interrupt0 (EXTI0)*/
    EIMSK = EXTI0_ENABLE;

    /*Selecting low level as interrupt trigger*/
    EICRA = EXTI0_LOW_LEVEL_TRIGGER;

    /*Clear EXTI0 flag*/
    SET_BIT(EIFR , EIFR_INTF0);

    /*Enable global interrupts*/
    SET_BIT(SREG , SREG_IBIT);

    /*Selecting CTC mode with OCR0A*/
    TCCR0 = TIMER0_CTC_MODE_SELECTION;

    /*Clearing timer/counter register*/
    TCNT0 = 0;

    /*Adjusting TIMER0 to fire CTC interrupt every 10ms for 8MHz frequency and prescaler by 8*/
    OCR0A = TIMER0_50MS_TICK;

    /*Enable CTC mode interrupt*/
    TIMSK0 = TIMER0_OCR0A_INT_EN;

    DDRB = IO_PINS_DIR_INITIALIZATION;

    /*Set PB3 to logic zero*/
    CLEAR_BIT(PORTB , PORTB_PB3);

    /*Activate pull-up resistor for PB0*/
    SET_BIT(PORTB , PORTB_PB0);

    /*Activate pull-up resistor for PB2*/
    SET_BIT(PORTB , PORTB_PB2);

    /*Select the power down mode*/
    SMCR = POWER_DOWN_MODE_SELECTION;

    /*Sleep enable*/
    SET_BIT(SMCR , SMCR_SE);

    /*Execute sleep instruction*/
    __asm__ __volatile__ ( "sleep" "\n\t" :: );

    return;
}
```

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 140 of file Functionality.c.

```
{
    /*Applying the state machine of the system*/

    /*Checking if there’s a residual charge or not in the battery and if the switch pressed for more than 10 seconds*/
    if( (GET_BIT(PINB , PINB_PB0) == NO_RESIDUAL_CHARGE) || (gu16_switchCounter > TEN_SECONDS) )
    {
        /*Variable used in delay operations*/
        u16_t au16_delayVariable = TWO_SEC_DELAY;

        /*Delay for two seconds*/
        while(au16_delayVariable--)
        {
            /*Variable used in for looping*/
            u16_t i = 0;

            /*Software delay for 1ms approximately*/
            for (i = 0 ; i < ONE_MS_DELAY ; i++);
        }

        /*Initialize the system again and enter power down mode*/
        attiny4_init();
    }

    /*Checking if the switch is pressed for (1~2) seconds and the system is already in the OFF state*/
    else if( (gu16_switchCounter >= ONE_SECOND && gu16_switchCounter <= TWO_SECONDS) && (gu8_systemStatus == SYSTEM_OFF_STATUS) )
    {
        /*Set PB3 to high level*/
        SET_BIT(PORTB , PORTB_PB3);

        /*Report that the system has become in ON mode*/
        gu8_systemStatus = SYSTEM_ON_STATUS;

        /*Reset the voltage checking counter*/
        gu16_checkCounter = 0;

        /*Reset voltage checking trials counter*/
        gu8_voltageCheckTrials = 0;
    }

    /*Checking if the switch is pressed for (1~2) seconds and the system is already in the ON state*/
    else if( (gu16_switchCounter >= ONE_SECOND && gu16_switchCounter <= TWO_SECONDS) && (gu8_systemStatus == SYSTEM_ON_STATUS) )
    {
        /*Set PB3 to low level*/
        CLEAR_BIT(PORTB , PORTB_PB3);

        /*Report that the system is in OFF mode*/
        gu8_systemStatus = SYSTEM_OFF_STATUS;
    }

    /*Checking after powering ON by 3 seconds that there’s a voltage present or not and applying two powering up trials
      if there’s no voltage present*/
    else if( (gu16_checkCounter == THREE_SECONDS) && (GET_BIT(PINB , PINB_PB1) == NO_VOLTAGE_PRESENT) && (gu8_voltageCheckTrials < 2) )
    {
        /*Variable used in delay operations*/
        u16_t au16_delayVariable = TWO_SEC_DELAY;

        /*Disable all interrupts*/
        CLEAR_BIT(SREG , SREG_IBIT);

        /*Set PB3 to low level*/
        CLEAR_BIT(PORTB , PORTB_PB3);

        /*Delay for two seconds*/
        while(au16_delayVariable--)
        {
            /*Variable used in for looping*/
            u16_t i = 0;

            /*Software delay for 1ms approximately*/
            for (i = 0 ; i < ONE_MS_DELAY ; i++);
        }

        /*Set PB3 to high level*/
        SET_BIT(PORTB , PORTB_PB3);

        /*Reset the voltage checking counter*/
        gu16_checkCounter = 0;

        /*Increase voltage checking trials counter*/
        gu8_voltageCheckTrials++;

        /*Enable all interrupts*/
        SET_BIT(SREG , SREG_IBIT);
    }

    /*Any other state happens the system will initialize and power down*/
    else
    {
        /*Initialize the system again and enter power down mode*/
        attiny4_init();
    }

    return;
}
```

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

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.

```
{
    /*Initializing the power manager circuit*/
    attiny4_init();

    while(1)
    {
        /*The main operation of the power manager circuit*/
        mainApplication();
    }
    return 0;
}
```

Index

ATtiny4.h, 3
attiny4_init
    Functionality.c, 5
    Functionality.h, 9

Functionality.c, 4
    attiny4_init, 5
    mainApplication, 6
Functionality.h, 8
    attiny4_init, 9
    mainApplication, 9

main
    main.c, 12
main.c, 11
    main, 12
mainApplication
    Functionality.c, 6
    Functionality.h, 9