

A tiny bike sentry

1.0

Generated on Wed Dec 18 2024 11:29:49 for A tiny bike sentry by Doxygen 1.9.8

Wed Dec 18 2024 11:29:49

1 Data Structure Index	1
1.1 Data Structures	1
2 File Index	3
2.1 File List	3
3 Data Structure Documentation	5
3.1 Animation Class Reference	5
3.1.1 Detailed Description	6
3.1.2 Constructor & Destructor Documentation	6
3.1.2.1 Animation()	6
3.1.3 Member Function Documentation	6
3.1.3.1 enter_deep_sleep()	6
3.1.3.2 enter_sentry()	6
3.1.3.3 exit_alarm()	6
3.1.3.4 in_alarm()	7
3.1.3.5 in_attention()	7
3.1.3.6 in_sentry()	7
3.1.4 Field Documentation	7
3.1.4.1 gpio	7
3.1.4.2 led_state	7
3.1.4.3 timing	8
3.2 Gpio Class Reference	8
3.2.1 Detailed Description	8
3.2.2 Constructor & Destructor Documentation	9
3.2.2.1 Gpio()	9
3.2.3 Member Function Documentation	9
3.2.3.1 off()	9
3.2.3.2 on()	9
3.2.3.3 set_pin()	9
3.2.3.4 setup()	11
3.2.3.5 toggle()	11
3.2.4 Field Documentation	11
3.2.4.1 timing	11
3.3 Timing Class Reference	12
3.3.1 Detailed Description	12
3.3.2 Constructor & Destructor Documentation	12
3.3.2.1 Timing()	12
3.3.3 Member Function Documentation	13
3.3.3.1 get_millis()	13
3.3.3.2 wait_ms()	13
3.3.4 Field Documentation	13
3.3.4.1 this_clock_freq_mhz	13

4 File Documentation	15
4.1 src/main/Animation.cpp File Reference	15
4.1.1 Detailed Description	15
4.2 Animation.cpp	15
4.3 src/main/Animation.h File Reference	16
4.3.1 Detailed Description	17
4.4 Animation.h	17
4.5 src/main/defines.h File Reference	17
4.5.1 Detailed Description	18
4.5.2 Macro Definition Documentation	18
4.5.2.1 ALARM_COOLDOWN_MS	18
4.5.2.2 ALARM_TOGGLE_FREQ	18
4.5.2.3 ATTENTION_COOLDOWN_MS	18
4.5.2.4 CLOCK_FREQ_MHZ	19
4.5.2.5 DELAY_BUTTON_DEBOUNCE_MS	19
4.5.2.6 DELAY_PIEZO_MOVED_MS	19
4.5.2.7 PIN_BUTTON	19
4.5.2.8 PIN_BUZZER	19
4.5.2.9 PIN_LED	19
4.5.2.10 PIN_PIEZO	20
4.5.2.11 THRESHOLD_PIEZO	20
4.6 defines.h	20
4.7 src/main/Gpio.cpp File Reference	21
4.7.1 Detailed Description	21
4.8 Gpio.cpp	21
4.9 src/main/Gpio.h File Reference	22
4.9.1 Detailed Description	23
4.10 Gpio.h	23
4.11 src/main/Timing.cpp File Reference	23
4.11.1 Detailed Description	23
4.12 Timing.cpp	24
4.13 src/main/Timing.h File Reference	24
4.13.1 Detailed Description	24
4.14 Timing.h	25
Index	27

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

Animation	Animation class that provides functions for various animations (e.g	5
Gpio	Gpio class that provides functions for direct hardware actions (e.g	8
Timing	Timing class that provides corrected delay and millis functions based on selected clock frequency	12

Chapter 2

File Index

2.1 File List

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

src/main/ Animation.cpp	
Cpp file for Animation class that provides functions for various animations (e.g	15
src/main/ Animation.h	
Header file for Animation class	16
src/main/ defines.h	
This file includes all important declarations and definitions	17
src/main/ Gpio.cpp	
Cpp file for Gpio class that provides functions for direct hardware actions (e.g	21
src/main/ Gpio.h	
Header file for Gpio class	22
src/main/ Timing.cpp	
Cpp file for Timing class that provides corrected delay and millis functions based on selected clock frequency	23
src/main/ Timing.h	
Header file for Timing class	24

Chapter 3

Data Structure Documentation

3.1 Animation Class Reference

[Animation](#) class that provides functions for various animations (e.g.

```
#include <Animation.h>
```

Public Member Functions

- [Animation](#) ()
Constructor for [Animation](#) class.
- void [enter_sentry](#) ()
[Animation](#) to play when entering sentry state.
- void [enter_deep_sleep](#) ()
[Animation](#) to play when entering deep sleep state.
- void [in_sentry](#) ()
[Animation](#) to play when in sentry mode.
- void [in_attention](#) ()
[Animation](#) to play when in attention mode.
- void [exit_alarm](#) ()
[Animation](#) to play when exiting alarm mode.
- void [in_alarm](#) ()
[Animation](#) to play when in alarm mode.

Private Attributes

- [Timing](#) timing
[Timing](#) instance.
- [Gpio](#) gpio
[GPIO](#) instance.
- bool [led_state](#)
current led state in alarm

3.1.1 Detailed Description

[Animation](#) class that provides functions for various animations (e.g. when switching states)

Definition at line 20 of file [Animation.h](#).

3.1.2 Constructor & Destructor Documentation

3.1.2.1 Animation()

```
Animation::Animation ( )
```

Constructor for [Animation](#) class.

Definition at line 21 of file [Animation.cpp](#).

References [gpio](#), [led_state](#), and [Gpio::setup\(\)](#).

3.1.3 Member Function Documentation

3.1.3.1 enter_deep_sleep()

```
void Animation::enter_deep_sleep ( )
```

[Animation](#) to play when entering deep sleep state.

Definition at line 38 of file [Animation.cpp](#).

References [gpio](#), [PIN_LED](#), and [Gpio::toggle\(\)](#).

3.1.3.2 enter_sentry()

```
void Animation::enter_sentry ( )
```

[Animation](#) to play when entering sentry state.

Definition at line 30 of file [Animation.cpp](#).

References [gpio](#), [PIN_LED](#), and [Gpio::toggle\(\)](#).

3.1.3.3 exit_alarm()

```
void Animation::exit_alarm ( )
```

[Animation](#) to play when exiting alarm mode.

Definition at line 62 of file [Animation.cpp](#).

References [gpio](#), [Gpio::off\(\)](#), [PIN_BUZZER](#), and [PIN_LED](#).

3.1.3.4 in_alarm()

```
void Animation::in_alarm ( )
```

[Animation](#) to play when in alarm mode.

Definition at line 71 of file [Animation.cpp](#).

References [gpio](#), [led_state](#), [PIN_BUZZER](#), [PIN_LED](#), and [Gpio::set_pin\(\)](#).

3.1.3.5 in_attention()

```
void Animation::in_attention ( )
```

[Animation](#) to play when in attention mode.

Definition at line 54 of file [Animation.cpp](#).

References [gpio](#), [Gpio::on\(\)](#), and [PIN_LED](#).

3.1.3.6 in_sentry()

```
void Animation::in_sentry ( )
```

[Animation](#) to play when in sentry mode.

Definition at line 46 of file [Animation.cpp](#).

References [gpio](#), [Gpio::off\(\)](#), and [PIN_LED](#).

3.1.4 Field Documentation

3.1.4.1 gpio

```
Gpio Animation::gpio [private]
```

GPIO instance.

Definition at line 32 of file [Animation.h](#).

Referenced by [Animation\(\)](#), [enter_deep_sleep\(\)](#), [enter_sentry\(\)](#), [exit_alarm\(\)](#), [in_alarm\(\)](#), [in_attention\(\)](#), and [in_sentry\(\)](#).

3.1.4.2 led_state

```
bool Animation::led_state [private]
```

current led state in alarm

Definition at line 33 of file [Animation.h](#).

Referenced by [Animation\(\)](#), and [in_alarm\(\)](#).

3.1.4.3 timing

`Timing Animation::timing [private]`

`Timing` instance.

Definition at line 31 of file [Animation.h](#).

The documentation for this class was generated from the following files:

- `src/main/Animation.h`
- `src/main/Animation.cpp`

3.2 Gpio Class Reference

`Gpio` class that provides functions for direct hardware actions (e.g.

```
#include <Gpio.h>
```

Public Member Functions

- `Gpio ()`
Constructor for `Timing` class.
- `void setup ()`
GPIO setup function.
- `void on (byte pin)`
Turns on a pin.
- `void off (byte pin)`
Turns off a pin.
- `void set_pin (byte pin, bool state)`
Sets pin to state.
- `void toggle (byte pin, int iterations, long d)`
Function to toggle a digital pin on and off.

Private Attributes

- `Timing timing`
`Timing` instance.

3.2.1 Detailed Description

`Gpio` class that provides functions for direct hardware actions (e.g.

turning on/off digital pins)

Definition at line 19 of file [Gpio.h](#).

3.2.2 Constructor & Destructor Documentation

3.2.2.1 Gpio()

```
Gpio::Gpio ( )
```

Constructor for [Timing](#) class.

Definition at line 21 of file [Gpio.cpp](#).

References [PIN_BUZZER](#), and [PIN_LED](#).

3.2.3 Member Function Documentation

3.2.3.1 off()

```
void Gpio::off (
    byte pin )
```

Turns off a pin.

Parameters

<i>pin</i>	pin to turn off
------------	-----------------

Definition at line 50 of file [Gpio.cpp](#).

Referenced by [Animation::exit_alarm\(\)](#), and [Animation::in_sentry\(\)](#).

3.2.3.2 on()

```
void Gpio::on (
    byte pin )
```

Turns on a pin.

Parameters

<i>pin</i>	pin to turn on
------------	----------------

Definition at line 41 of file [Gpio.cpp](#).

Referenced by [Animation::in_attention\(\)](#).

3.2.3.3 set_pin()

```
void Gpio::set_pin (
    byte pin,
    bool state )
```

Sets pin to state.

Parameters

<i>pin</i>	pin to set state
<i>state</i>	state to set (true/false)

Definition at line 60 of file [Gpio.cpp](#).

Referenced by [Animation::in_alarm\(\)](#).

3.2.3.4 setup()

```
void Gpio::setup ( )
```

GPIO setup function.

Sets all output pins as outputs.

Definition at line 31 of file [Gpio.cpp](#).

References [PIN_BUZZER](#), and [PIN_LED](#).

Referenced by [Animation::Animation\(\)](#).

3.2.3.5 toggle()

```
void Gpio::toggle (
    byte pin,
    int iterations,
    long d )
```

Function to toggle a digital pin on and off.

Parameters

<i>pin</i>	Pin to toggle on and off
<i>iterations</i>	How often should pin be toggled
<i>d</i>	delay between of/off states (d = 1 / frequency)

Definition at line 71 of file [Gpio.cpp](#).

References [timing](#), and [Timing::wait_ms\(\)](#).

Referenced by [Animation::enter_deep_sleep\(\)](#), and [Animation::enter_sentry\(\)](#).

3.2.4 Field Documentation

3.2.4.1 timing

```
Timing Gpio::timing [private]
```

[Timing](#) instance.

Definition at line 29 of file [Gpio.h](#).

Referenced by [toggle\(\)](#).

The documentation for this class was generated from the following files:

- [src/main/Gpio.h](#)
- [src/main/Gpio.cpp](#)

3.3 Timing Class Reference

[Timing](#) class that provides corrected delay and millis functions based on selected clock frequency.

```
#include <Timing.h>
```

Public Member Functions

- [Timing](#) (byte clock_freq_mhz)
Constructor for [Timing](#) class.
- void [wait_ms](#) (long d)
Own delay function that counteracts different clock speeds.
- long long [get_millis](#) ()
Own millis function that counteracts different clock speeds.

Private Attributes

- byte [this_clock_freq_mhz](#)
selected clock frequency

3.3.1 Detailed Description

[Timing](#) class that provides corrected delay and millis functions based on selected clock frequency.

Definition at line 17 of file [Timing.h](#).

3.3.2 Constructor & Destructor Documentation

3.3.2.1 Timing()

```
Timing::Timing (  
    byte clock_freq_mhz )
```

Constructor for [Timing](#) class.

Parameters

<i>clock_freq_mhz</i>	selected clock frequency.
-----------------------	---------------------------

Definition at line 20 of file [Timing.cpp](#).

References [this_clock_freq_mhz](#).

3.3.3 Member Function Documentation

3.3.3.1 `get_millis()`

```
long long Timing::get_millis ( )
```

Own millis function that counteracts different clock speeds.

Returns

long millis since ATtiny restart

Definition at line 43 of file [Timing.cpp](#).

References [this_clock_freq_mhz](#).

3.3.3.2 `wait_ms()`

```
void Timing::wait_ms (
    long duration )
```

Own delay function that counteracts different clock speeds.

Parameters

<i>duration</i>	delay time in ms
-----------------	------------------

Definition at line 29 of file [Timing.cpp](#).

References [this_clock_freq_mhz](#).

Referenced by [Gpio::toggle\(\)](#).

3.3.4 Field Documentation

3.3.4.1 `this_clock_freq_mhz`

```
byte Timing::this_clock_freq_mhz [private]
```

selected clock frequency

Definition at line 24 of file [Timing.h](#).

Referenced by [get_millis\(\)](#), [Timing\(\)](#), and [wait_ms\(\)](#).

The documentation for this class was generated from the following files:

- [src/main/Timing.h](#)
- [src/main/Timing.cpp](#)

Chapter 4

File Documentation

4.1 src/main/Animation.cpp File Reference

Cpp file for [Animation](#) class that provides functions for various animations (e.g.

```
#include "Animation.h"
#include "Arduino.h"
#include "Gpio.h"
#include "Timing.h"
#include "defines.h"
```

4.1.1 Detailed Description

Cpp file for [Animation](#) class that provides functions for various animations (e.g.

when switching states)

Author

Lukas Krämer

Definition in file [Animation.cpp](#).

4.2 Animation.cpp

[Go to the documentation of this file.](#)

```
00001 /**
00002  * @file Animation.cpp
00003  *
00004  * Cpp file for Animation class that provides
00005  * functions for various animations (e.g. when switching states)
00006  *
00007  * @author Lukas Krämer
00008  */
00009 #pragma once
00010
00011 #include "Animation.h"
00012
```

```

00013 #include "Arduino.h"
00014 #include "Gpio.h"
00015 #include "Timing.h"
00016 #include "defines.h"
00017
00018 /**
00019  * Constructor for Animation class.
00020  */
00021 Animation::Animation() : timing(CLOCK_FREQ_MHZ), gpio() {
00022     gpio.setup();
00023     led_state = false;
00024 }
00025
00026 /**
00027  * Animation to play when entering sentry state.
00028  *
00029  */
00030 void Animation::enter_sentry() {
00031     gpio.toggle(PIN_LED, 3, 100);
00032 }
00033
00034 /**
00035  * Animation to play when entering deep sleep state.
00036  *
00037  */
00038 void Animation::enter_deep_sleep() {
00039     gpio.toggle(PIN_LED, 2, 400);
00040 }
00041
00042 /**
00043  * Animation to play when in sentry mode.
00044  *
00045  */
00046 void Animation::in_sentry() {
00047     gpio.off(PIN_LED);
00048 }
00049
00050 /**
00051  * Animation to play when in attention mode.
00052  *
00053  */
00054 void Animation::in_attention() {
00055     gpio.on(PIN_LED);
00056 }
00057
00058 /**
00059  * Animation to play when exiting alarm mode.
00060  *
00061  */
00062 void Animation::exit_alarm() {
00063     gpio.off(PIN_LED);
00064     gpio.off(PIN_BUZZER);
00065 }
00066
00067 /**
00068  * Animation to play when in alarm mode.
00069  *
00070  */
00071 void Animation::in_alarm() {
00072     gpio.set_pin(PIN_LED, led_state);
00073     gpio.set_pin(PIN_BUZZER, !led_state);
00074     led_state = !led_state;
00075 }

```

4.3 src/main/Animation.h File Reference

Header file for [Animation](#) class.

```

#include "Arduino.h"
#include "Gpio.h"
#include "Timing.h"
#include "defines.h"

```

Data Structures

- class [Animation](#)

[Animation](#) class that provides functions for various animations (e.g.

4.3.1 Detailed Description

Header file for [Animation](#) class.

Author

Lukas Krämer

Definition in file [Animation.h](#).

4.4 Animation.h

[Go to the documentation of this file.](#)

```
00001 /**
00002  * @file Animation.h
00003  *
00004  * Header file for Animation class
00005  *
00006  * @author Lukas Krämer
00007  */
00008 #pragma once
00009
00010 #include "Arduino.h"
00011 #include "Gpio.h"
00012 #include "Timing.h"
00013 #include "defines.h"
00014
00015 /**
00016  * Animation class that provides
00017  * functions for various animations (e.g. when switching states)
00018  *
00019  */
00020 class Animation {
00021 public:
00022     Animation();
00023     void enter_sentry();
00024     void enter_deep_sleep();
00025     void in_sentry();
00026     void in_attention();
00027     void exit_alarm();
00028     void in_alarm();
00029
00030 private:
00031     Timing timing; ///< Timing instance
00032     Gpio gpio;      ///< GPIO instance
00033     bool led_state; ///< current led state in alarm
00034 };
```

4.5 src/main/defines.h File Reference

This file includes all important declarations and definitions.

Macros

- #define [PIN_LED](#) 1
Pin for the LED.
- #define [PIN_BUTTON](#) 2
Pin for the button.
- #define [PIN_PIEZO](#) 3
Pin for the piezo.
- #define [PIN_BUZZER](#) 4

- Pin for the button.*
- `#define CLOCK_FREQ_MHZ 1`
Selected clock frequency in MHz.
- `#define THRESHOLD_PIEZO 30`
Threshold for piezo for movement detection.
- `#define DELAY_BUTTON_DEBOUNCE_MS 50`
Button debounce delay.
- `#define ALARM_TOGGLE_FREQ 20`
Frequency for alarm.
- `#define DELAY_PIEZO_MOVED_MS 500`
When piezo is moved, wait for DELAY_PIEZO_MOVED_MS because it could be just a button press.
- `#define ATTENTION_COOLDOWN_MS 5000`
Cooldown after which state attention is left again.
- `#define ALARM_COOLDOWN_MS 5000`
Cooldown after which state alarm is left again.

4.5.1 Detailed Description

This file includes all important declarations and definitions.

Author

Lukas Krämer

Definition in file [defines.h](#).

4.5.2 Macro Definition Documentation

4.5.2.1 ALARM_COOLDOWN_MS

```
#define ALARM_COOLDOWN_MS 5000
```

Cooldown after which state alarm is left again.

Definition at line 63 of file [defines.h](#).

4.5.2.2 ALARM_TOGGLE_FREQ

```
#define ALARM_TOGGLE_FREQ 20
```

Frequency for alarm.

Definition at line 47 of file [defines.h](#).

4.5.2.3 ATTENTION_COOLDOWN_MS

```
#define ATTENTION_COOLDOWN_MS 5000
```

Cooldown after which state attention is left again.

Definition at line 58 of file [defines.h](#).

4.5.2.4 CLOCK_FREQ_MHZ

```
#define CLOCK_FREQ_MHZ 1
```

Selected clock frequency in MHz.

Definition at line 32 of file [defines.h](#).

4.5.2.5 DELAY_BUTTON_DEBOUNCE_MS

```
#define DELAY_BUTTON_DEBOUNCE_MS 50
```

Button debounce delay.

Definition at line 42 of file [defines.h](#).

4.5.2.6 DELAY_PIEZO_MOVED_MS

```
#define DELAY_PIEZO_MOVED_MS 500
```

When piezo is moved, wait for DELAY_PIEZO_MOVED_MS because it could be just a button press.

Definition at line 53 of file [defines.h](#).

4.5.2.7 PIN_BUTTON

```
#define PIN_BUTTON 2
```

Pin for the button.

Definition at line 17 of file [defines.h](#).

4.5.2.8 PIN_BUZZER

```
#define PIN_BUZZER 4
```

Pin for the button.

Definition at line 27 of file [defines.h](#).

4.5.2.9 PIN_LED

```
#define PIN_LED 1
```

Pin for the LED.

Definition at line 12 of file [defines.h](#).

4.5.2.10 PIN_PIEZO

```
#define PIN_PIEZO 3
```

Pin for the piezo.

Definition at line 22 of file [defines.h](#).

4.5.2.11 THRESHOLD_PIEZO

```
#define THRESHOLD_PIEZO 30
```

Threshold for piezo for movement detection.

Definition at line 37 of file [defines.h](#).

4.6 defines.h

[Go to the documentation of this file.](#)

```
00001 /**
00002  * @file defines.h
00003  *
00004  * This file includes all important declarations and definitions.
00005  *
00006  * @author Lukas Krämer
00007  */
00008
00009 /**
00010  * @brief Pin for the LED.
00011  */
00012 #define PIN_LED 1
00013
00014 /**
00015  * @brief Pin for the button.
00016  */
00017 #define PIN_BUTTON 2
00018
00019 /**
00020  * @brief Pin for the piezo.
00021  */
00022 #define PIN_PIEZO 3
00023
00024 /**
00025  * @brief Pin for the button.
00026  */
00027 #define PIN_BUZZER 4
00028
00029 /**
00030  * @brief Selected clock frequency in MHz.
00031  */
00032 #define CLOCK_FREQ_MHZ 1
00033
00034 /**
00035  * @brief Threshold for piezo for movement detection.
00036  */
00037 #define THRESHOLD_PIEZO 30
00038
00039 /**
00040  * @brief Button debounce delay.
00041  */
00042 #define DELAY_BUTTON_DEBOUNCE_MS 50
00043
00044 /**
00045  * @brief Frequency for alarm.
00046  */
00047 #define ALARM_TOGGLE_FREQ 20
00048
00049 /**
00050  * @brief When piezo is moved, wait
00051  * for DELAY_PIEZO_MOVED_MS because it could be just a button press.
00052  */
```



```

00053 #define DELAY_PIEZO_MOVED_MS 500
00054
00055 /**
00056  * @brief Cooldown after which state attention is left again.
00057  */
00058 #define ATTENTION_COOLDOWN_MS 5000
00059
00060 /**
00061  * @brief Cooldown after which state alarm is left again.
00062  */
00063 #define ALARM_COOLDOWN_MS 5000

```

4.7 src/main/Gpio.cpp File Reference

Cpp file for [Gpio](#) class that provides functions for direct hardware actions (e.g.

```

#include "Gpio.h"
#include "Arduino.h"
#include "Timing.h"
#include "defines.h"

```

4.7.1 Detailed Description

Cpp file for [Gpio](#) class that provides functions for direct hardware actions (e.g.

turning on/off digital pins)

Author

Lukas Krämer

Definition in file [Gpio.cpp](#).

4.8 Gpio.cpp

[Go to the documentation of this file.](#)

```

00001 /**
00002  * @file Gpio.cpp
00003  *
00004  * Cpp file for Gpio class that provides
00005  * functions for direct hardware actions (e.g. turning on/off digital pins)
00006  *
00007  * @author Lukas Krämer
00008  */
00009 #pragma once
00010
00011 #include "Gpio.h"
00012
00013 #include "Arduino.h"
00014 #include "Timing.h"
00015 #include "defines.h"
00016
00017 /**
00018  * Constructor for Timing class.
00019  *
00020  */
00021 Gpio::Gpio() : timing(CLOCK_FREQ_MHZ) {
00022     pinMode(PIN_LED, OUTPUT);
00023     pinMode(PIN_BUZZER, OUTPUT);
00024 }
00025

```

```

00026 /**
00027  * GPIO setup function.
00028  * Sets all output pins as outputs.
00029  *
00030  */
00031 void Gpio::setup() {
00032     pinMode(PIN_LED, OUTPUT);
00033     pinMode(PIN_BUZZER, OUTPUT);
00034 }
00035
00036 /**
00037  * Turns on a pin.
00038  *
00039  * @param pin pin to turn on
00040  */
00041 void Gpio::on(byte pin) {
00042     digitalWrite(pin, HIGH);
00043 }
00044
00045 /**
00046  * Turns off a pin.
00047  *
00048  * @param pin pin to turn off
00049  */
00050 void Gpio::off(byte pin) {
00051     digitalWrite(pin, LOW);
00052 }
00053
00054 /**
00055  * Sets pin to state.
00056  *
00057  * @param pin pin to set state
00058  * @param state state to set (true/false)
00059  */
00060 void Gpio::set_pin(byte pin, bool state) {
00061     digitalWrite(pin, state);
00062 }
00063
00064 /**
00065  * Function to toggle a digital pin on and off.
00066  *
00067  * @param pin Pin to toggle on and off
00068  * @param iterations How often should pin be toggled
00069  * @param d delay between on/off states (d = 1 / frequency)
00070  */
00071 void Gpio::toggle(byte pin, int iterations, long d) {
00072     for (int i = 0; i < iterations; ++i) {
00073         digitalWrite(pin, HIGH);
00074         timing.wait_ms(d);
00075         digitalWrite(pin, LOW);
00076         timing.wait_ms(d);
00077     }
00078 }

```

4.9 src/main/Gpio.h File Reference

Header file for [Gpio](#) class.

```

#include "Arduino.h"
#include "Timing.h"
#include "defines.h"

```

Data Structures

- class [Gpio](#)
[Gpio](#) class that provides functions for direct hardware actions (e.g.

4.9.1 Detailed Description

Header file for [Gpio](#) class.

Author

Lukas Krämer

Definition in file [Gpio.h](#).

4.10 Gpio.h

[Go to the documentation of this file.](#)

```
00001 /**
00002  * @file Gpio.h
00003  *
00004  * Header file for Gpio class
00005  *
00006  * @author Lukas Krämer
00007  */
00008 #pragma once
00009
00010 #include "Arduino.h"
00011 #include "Timing.h"
00012 #include "defines.h"
00013
00014 /**
00015  * Gpio class that provides
00016  * functions for direct hardware actions (e.g. turning on/off digital pins)
00017  *
00018  */
00019 class Gpio {
00020 public:
00021     Gpio();
00022     void setup();
00023     void on(byte pin);
00024     void off(byte pin);
00025     void set_pin(byte pin, bool state);
00026     void toggle(byte pin, int iterations, long d);
00027
00028 private:
00029     Timing timing; ///< Timing instance
00030 };
```

4.11 src/main/Timing.cpp File Reference

Cpp file for [Timing](#) class that provides corrected delay and millis functions based on selected clock frequency.

```
#include "Timing.h"
#include "Arduino.h"
```

4.11.1 Detailed Description

Cpp file for [Timing](#) class that provides corrected delay and millis functions based on selected clock frequency.

Author

Lukas Krämer

Definition in file [Timing.cpp](#).

4.12 Timing.cpp

[Go to the documentation of this file.](#)

```

00001 /**
00002  * @file Timing.cpp
00003  *
00004  * Cpp file for Timing class that provides corrected
00005  * delay and millis functions based on selected clock frequency.
00006  *
00007  * @author Lukas Krämer
00008  */
00009 #pragma once
00010
00011 #include "Timing.h"
00012
00013 #include "Arduino.h"
00014
00015 /**
00016  * Constructor for Timing class.
00017  *
00018  * @param clock_freq_mhz selected clock frequency.
00019  */
00020 Timing::Timing(byte clock_freq_mhz) {
00021     this_clock_freq_mhz = clock_freq_mhz;
00022 }
00023
00024 /**
00025  * Own delay function that counteracts different clock speeds.
00026  *
00027  * @param duration delay time in ms
00028  */
00029 void Timing::wait_ms(long duration) {
00030     if (this_clock_freq_mhz == 1)
00031         delay(duration * 8);
00032     else if (this_clock_freq_mhz == 8)
00033         delay(duration);
00034     else
00035         delay(duration); // Add options for external 16MHz here if needed
00036 }
00037
00038 /**
00039  * Own millis function that counteracts different clock speeds.
00040  *
00041  * @return long millis since ATtiny restart
00042  */
00043 long long Timing::get_millis() {
00044     if (this_clock_freq_mhz == 1) return millis() / 8;
00045     if (this_clock_freq_mhz == 8) return millis();
00046 }

```

4.13 src/main/Timing.h File Reference

Header file for [Timing](#) class.

```
#include "Arduino.h"
```

Data Structures

- class [Timing](#)
Timing class that provides corrected delay and millis functions based on selected clock frequency.

4.13.1 Detailed Description

Header file for [Timing](#) class.

Author

Lukas Krämer

Definition in file [Timing.h](#).

4.14 Timing.h

[Go to the documentation of this file.](#)

```
00001 /**
00002  * @file Timing.h
00003  *
00004  * Header file for Timing class.
00005  *
00006  * @author Lukas Krämer
00007  */
00008 #pragma once
00009
00010 #include "Arduino.h"
00011
00012 /**
00013  * Timing class that provides corrected
00014  * delay and millis functions based on selected clock frequency.
00015  *
00016  */
00017 class Timing {
00018     public:
00019         Timing(byte clock_freq_mhz);
00020         void wait_ms(long d);
00021         long long get_millis();
00022
00023     private:
00024         byte this_clock_freq_mhz; ///< selected clock frequency
00025 };
```


Index

ALARM_COOLDOWN_MS
 [defines.h, 18](#)

ALARM_TOGGLE_FREQ
 [defines.h, 18](#)

Animation, [5](#)
 Animation, [6](#)
 enter_deep_sleep, [6](#)
 enter_sentry, [6](#)
 exit_alarm, [6](#)
 gpio, [7](#)
 in_alarm, [6](#)
 in_attention, [7](#)
 in_sentry, [7](#)
 led_state, [7](#)
 timing, [7](#)

ATTENTION_COOLDOWN_MS
 [defines.h, 18](#)

CLOCK_FREQ_MHZ
 [defines.h, 18](#)

[defines.h](#)
 ALARM_COOLDOWN_MS, [18](#)
 ALARM_TOGGLE_FREQ, [18](#)
 ATTENTION_COOLDOWN_MS, [18](#)
 CLOCK_FREQ_MHZ, [18](#)
 DELAY_BUTTON_DEBOUNCE_MS, [19](#)
 DELAY_PIEZO_MOVED_MS, [19](#)
 PIN_BUTTON, [19](#)
 PIN_BUZZER, [19](#)
 PIN_LED, [19](#)
 PIN_PIEZO, [19](#)
 THRESHOLD_PIEZO, [20](#)

DELAY_BUTTON_DEBOUNCE_MS
 [defines.h, 19](#)

DELAY_PIEZO_MOVED_MS
 [defines.h, 19](#)

enter_deep_sleep
 Animation, [6](#)

enter_sentry
 Animation, [6](#)

exit_alarm
 Animation, [6](#)

get_millis
 Timing, [13](#)

Gpio, [8](#)
 Gpio, [9](#)
 off, [9](#)
 on, [9](#)
 set_pin, [9](#)
 setup, [11](#)
 timing, [11](#)
 toggle, [11](#)

gpio
 Animation, [7](#)

in_alarm
 Animation, [6](#)

in_attention
 Animation, [7](#)

in_sentry
 Animation, [7](#)

led_state
 Animation, [7](#)

off
 Gpio, [9](#)

on
 Gpio, [9](#)

PIN_BUTTON
 [defines.h, 19](#)

PIN_BUZZER
 [defines.h, 19](#)

PIN_LED
 [defines.h, 19](#)

PIN_PIEZO
 [defines.h, 19](#)

set_pin
 Gpio, [9](#)

setup
 Gpio, [11](#)
 src/main/Animation.cpp, [15](#)
 src/main/Animation.h, [16, 17](#)
 src/main/defines.h, [17, 20](#)
 src/main/Gpio.cpp, [21](#)
 src/main/Gpio.h, [22, 23](#)
 src/main/Timing.cpp, [23, 24](#)
 src/main/Timing.h, [24, 25](#)

this_clock_freq_mhz
 Timing, [13](#)

THRESHOLD_PIEZO
 [defines.h, 20](#)

Timing, [12](#)
 get_millis, [13](#)
 this_clock_freq_mhz, [13](#)

- Timing, [12](#)
- wait_ms, [13](#)
- timing
 - Animation, [7](#)
 - Gpio, [11](#)
- toggle
 - Gpio, [11](#)
- wait_ms
 - Timing, [13](#)