Swarm Firmware AVR

Using by Doxygen 1.9.1
page numbering is the reference manual is relative to this page for ease of indexing, hyperef is present.

Chapter 1

License

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 3

File Index

3.1 File List

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

Chapter 4

Data Structure Documentation

4.1 _i_pos Struct Reference

```
#include <__kinematics__.h>
```

Data Fields

- float v

  linear velocity V
- float w

  rotational velocity W

4.1.1 Detailed Description

Inertial pos vector V and W

The documentation for this struct was generated from the following file:

- src/__kinematics__.h

4.2 _omega Struct Reference

```
#include <__kinematics__.h>
```

Data Fields

- float wl

  left motor
- float wr

  right motor

4.2.1 Detailed Description

Struct to hold wheels' velocity rad/s

The documentation for this struct was generated from the following file:

- src/__kinematics__.h

4.3 _theta Struct Reference

```
#include <__kinematics__.h>
```

Data Fields

- float left

  left motor
- float right

  right motor

4.3.1 Detailed Description

struct to hold wheel absolute rotation

The documentation for this struct was generated from the following file:

- src/__kinematics__.h

4.4 mat2 Struct Reference

```
#include <__kinematics__.h>
```

Data Fields

- float a

  11
- float b

  12
- float c

  21
- float d

  22

4.4.1 Detailed Description

square matrix struct [

Parameters

a	

The documentation for this struct was generated from the following file:

- src/__kinematics__.h

4.5 PID_DATA Struct Reference

PID Status.

```
#include <_pid_.h>
```

Data Fields

- int16_t lastProcessValue

  Last process value, used to find derivative of process value.
- int32_t sumError

  Summation of errors, used for integrate calculations.
- int16_t P_Factor

  The Proportional tuning constant, multiplied with SCALING_FACTOR.
- int16_t I_Factor

  The Integral tuning constant, multiplied with SCALING_FACTOR.
- int16_t D_Factor

  The Derivative tuning constant, multiplied with SCALING_FACTOR.
- int16_t maxError

  Maximum allowed error, avoid overflow.
- int32_t maxSumError

  Maximum allowed sumerror, avoid overflow.

4.5.1 Detailed Description

PID Status.

Setpoints and data used by the PID control algorithm

The documentation for this struct was generated from the following file:

- src/_pid_.h

4.6 point Struct Reference

```
#include <__kinematics__.h>
```

Data Fields

- float x
- float y

4.6.1 Detailed Description

point struct (x and y card. coordinates

4.6.2 Field Documentation

4.6.2.1 x

```
float x
```

Parameters

x	

4.6.2.2 y

```
float y
```

Parameters

y	

The documentation for this struct was generated from the following file:

- src/__kinematics__.h

4.7 pos Struct Reference

```
#include <__kinematics__.h>
```

Data Fields

- float x
- float y
- float th

4.7.1 Detailed Description

position struct to hold robot's pos vector

4.7.2 Field Documentation

4.7.2.1 th

```
float th
```

Parameters

Theta	(\theta)

4.7.2.2 x

```
float x
```

Parameters

x	

4.7.2.3 y

```
float y
```

Parameters

y	

The documentation for this struct was generated from the following file:

- src/__kinematics__.h

4.8 pos_dot Struct Reference

```
#include <__kinematics__.h>
```

Data Fields

- float x_dot
- float y_dot
- float th_dot

4.8.1 Detailed Description

position derivative vector struct (\frac{d pos}{dt})

4.8.2 Field Documentation

4.8.2.1 th_dot

```
float th_dot
```

Parameters

4.8.2.2 x_dot

```
float x_dot
```

Parameters

x'	

4.8.2.3 y_dot

```
float y_dot
```

Parameters

y'	

The documentation for this struct was generated from the following file:

- src/__kinematics__.h

Chapter 5

File Documentation

5.1 src/__adc__.h File Reference

Header file for **adc**.c.

```
#include <__swarm_wold__.h>
```

Functions

- void _adc_init (void)
- int _adc_read (uint8_t channel)

5.1.1 Detailed Description

Header file for **adc**.c.

- File: **adc**.h
- Compiler: GCC-AVR
- Supported devices: Tested on 328p
- AppNote: ADC module driver

Author

Swarm robot graduation project workgroub
Mechatronics Program for the Distinguished
$Name$

Revision

1

$RCSfile$

Date

/19/2021 9:30:13 AM

5.1.2 Function Documentation

5.1.2.1 _adc_init()

```
void _adc_init (
            void  )
```

Initialize the ADC module

5.1.2.2 _adc_read()

```
int _adc_read (
            uint8_t channel )
```

Reads from

Parameters

channel	

Parameters

in	*channel*	adc channel from PortC.

5.2 src/__dc_control__.h File Reference

Header file for **dc_control**.c.

```
#include <__swarm_wold__.h>
```

Macros

- #define **MA** 1
- #define **MB** 2
- #define **DEBUG_CONTROLLER** 0
- #define **K_P** 4.0
- #define **K_I** 0.3
- #define **K_D** 0.5

Functions

- int16_t _ref (uint8_t motor)
- void _command (uint8_t motor, int16_t inputValue)
- float _sens (uint8_t motor)
- int _set_speed (uint8_t motor, int value)
- void _break_motor (uint8_t motor)
- void _init_dc_control (void)
- int16_t _update_controller (uint8_t motor)
- int16_t _dc_controller_loop (void)

5.2.1 Detailed Description

Header file for **dc_control**.c.

- File: **dc_control**.h
- Compiler: GCC-AVR
- Supported devices: Tested on 328p
- AppNote: PID for DC motor control

Author

Swarm robot graduation project workgroub
Mechatronics Program for the Distinguished
$Name$

Revision

1

$RCSfile$

Date

/26/2021 9:18:13 PM

5.2.2 Function Documentation

5.2.2.1 _break_motor()

```
void _break_motor (
            uint8_t motor )
```

breaks the

Parameters

motor	

Parameters

in	*motor*	motor MA or MB.

5.2.2.2 _command()

```
void _command (
            uint8_t motor,
            int16_t inputValue )
```

Output command module for

Parameters

motor	of value
inputValue	

Parameters

in	*motor*	motor MA or MB.
in	*inputValue*	input value

5.2.2.3 _dc_controller_loop()

```
int16_t _dc_controller_loop (
            void )
```

PID DC controller loop

5.2.2.4 _init_dc_control()

```
void _init_dc_control (
            void )
```

Initialize the DC PID control module

5.2.2.5 _ref()

```
int16_t _ref (
            uint8_t motor )
```

Reference for

Parameters

motor	

Parameters

in	*motor*	motor MA or MB.

5.2.2.6 _sens()

```
float _sens (
            uint8_t motor )
```

sensor module for

Parameters

motor	

Parameters

in	*motor*	motor MA or MB.

5.2.2.7 _set_speed()

```
int _set_speed (
            uint8_t motor,
            int value )
```

L298 driver module using PWM signals and motor orientation

Parameters

value	can be negative or positive for direction

Parameters

in	*motor*	motor MA or MB.
in	*value*	speed value

5.2.2.8 _update_controller()

```
int16_t _update_controller (
            uint8_t motor )
```

Update a control iteration for a discrete PID controller for the DC motor

Parameters

in	*motor*	motor MA or MB.

5.3 src/__format__.h File Reference

Header file for **format**.c.

```
#include <__swarm_wold__.h>
```

Functions

- char ∗ _float_to_printable (float input)

5.3.1 Detailed Description

Header file for **format**.c.

- File: **format**.h
- Compiler: GCC-AVR
- Supported devices: Tested on 328p
- AppNote: string formatter

Author

Swarm robot graduation project workgroub
Mechatronics Program for the Distinguished
$Name$

Revision

1

$RCSfile$

Date

/5/2021 12:34:35 AM

5.3.2 Function Documentation

5.3.2.1 _float_to_printable()

```
char* _float_to_printable (
            float input )
```

Convert a float varialbe to string

Parameters

in	*input*	input value of float to convert to string

5.4 src/__INT_0_1__.h File Reference

Header file for **INT_0_1**.c.

```
#include <__swarm_wold__.h>
```

Macros

- #define **_INT_LOW_LEVEL** 0
- #define **_INT_CHANGE_LEVEL** 1
- #define **_INT_FALLING_EDGE** 2
- #define **_INT_RISING_EDGE** 3

Functions

- void _interrupt0_enable (uint8_t trigger)

5.4.1 Detailed Description

Header file for **INT_0_1**.c.

- File: **INT_0_1**.h
- Compiler: GCC-AVR
- Supported devices: Tested on 328p
- AppNote: external interrupt driver

Author

Swarm robot graduation project workgroub
Mechatronics Program for the Distinguished
$Name$

Revision

1

$RCSfile$

Date

/11/2021 9:18:06 AM

5.4.2 Function Documentation

5.4.2.1 _interrupt0_enable()

```
void _interrupt0_enable (
            uint8_t trigger )
```

Enable interrupt of pin INT0 (see _pin_map.h)

Parameters

in	*trigger*	trigger mode, see file __INT_0_1__.h

5.5 src/__kinematics__.h File Reference

Header file for **kimenatics**.c.

```
#include <__swarm_wold__.h>
```

Data Structures

- struct point
- struct pos
- struct pos_dot
- struct _i_pos
- struct mat2
- struct _omega
- struct _theta

Macros

- #define **L** 0.06
- #define **r** 0.02
- #define **R_over_L** 0.333
- #define **R_over_2** 0.01

Functions

- struct _i_pos _omega_to_intertial (struct _omega ∗input)
- struct pos_dot _inertial_to_pos_dot (struct _i_pos ∗inertial, struct pos ∗_pos)
- uint16_t _dead_reckon (struct pos ∗_pos, struct pos_dot ∗_pos_dot)
- struct pos _thetaLR_to_pos (struct _theta ∗th)
- void _update_thetaLR (struct _theta ∗input)
- void _update_omega (struct _omega ∗input)

5.5.1 Detailed Description

Header file for **kimenatics**.c.

- File: **kimenatics**.h
- Compiler: GCC-AVR
- Supported devices: Tested on 328p
- AppNote: Kinematics calculations for the mobile robot

Author

Swarm robot graduation project workgroub
Mechatronics Program for the Distinguished
$Name$

Revision

1

$RCSfile$

Date

5/29/2021 8:04:54 PM

5.5.2 Function Documentation

5.5.2.1 _dead_reckon()

```
uint16_t _dead_reckon (
            struct pos * _pos,
            struct pos_dot * _pos_dot )
```

a dead-reckoner for spatial pos estimation, returns debug info

Parameters

in	*_pos*	pos
in	*_pos_dot*	derivative of pos

5.5.2.2 _inertial_to_pos_dot()

```
struct pos_dot _inertial_to_pos_dot (
            struct _i_pos * inertial,
            struct pos * _pos )
```

A function to convert inertial velocities to pos_dot

Parameters

in	*inertial*	struct of inertial pos
in	*_pos*	struct of current pos for \theta

5.5.2.3 _omega_to_intertial()

```
struct _i_pos _omega_to_intertial (
            struct _omega * input )
```

A function to convert wheels rotation velocity to inertial velocities

Parameters

in	*input*	struct of wheels velocity
	_omega	

5.5.2.4 _thetaLR_to_pos()

```
struct pos _thetaLR_to_pos (
            struct _theta * th )
```

A function to convert wheels' absolute rotation angles to pos (not proper)

Parameters

in	*th*	wheels' absolute rotation

5.5.2.5 _update_omega()

```
void _update_omega (
             struct _omega * input )
```

A function to update the wheels' rotation velocity or each wheel from hardware modules

Parameters

in	*input*	wheel's rotational velocity

5.5.2.6 _update_thetaLR()

```
void _update_thetaLR (
             struct _theta * input )
```

A function to update the absolute rotation angle of each wheel from hardware modules

Parameters

in	*input*	wheels' absolute rotation

5.6 src/__odometry__.h File Reference

Header file for **odometry**.c.

```
#include <__swarm_wold__.h>
```

Macros

- #define **__ENC_TICK_THETA_FOR_OMEGA** 190399
- #define **__ENC_TICK_THETA** 0.1904
- #define **__PM_lower_bound** 200
- #define **__PM_upper_bound** 800
- #define **__PM_SAMPLE_COUNT** 5
- #define **__PM_SLOPE** 1
- #define **FORWARD** 1
- #define **BACKWARD** -1

Functions

- float _thetaA (void)
- float _thetaB (void)
- float _omega_from_encA (void)
- float _omega_from_encB (void)
- float _omega_from_PMA (void)
- float _omega_from_PMB (void)
- float _omega_comp_A (void)
- float _omega_comp_B (void)
- int32_t _ticksA ()
- int32_t _ticksB ()
- void _insertion_sort (uint16_t arr[ ], int n)

5.6.1 Detailed Description

Header file for **odometry**.c.

- File: __odometry__h
- Compiler: GCC-AVR
- Supported devices: Tested on 328p
- AppNote: Odometry calculations for the mobile robot

Author

Swarm robot graduation project workgroub
Mechatronics Program for the Distinguished
$Name$

Revision

1

$RCSfile$

Date

/19/2021 9:43:59 AM

5.6.2 Function Documentation

5.6.2.1 _insertion_sort()

```
void _insertion_sort (
             uint16_t arr[ ],
             int n )
```

Insertion sort algorithm (fastest for small array sizes)

Parameters

in	*arr*	pointer to start of sorting
in	*n*	offset from
	arr[ ]	

5.6.2.2 _omega_comp_A()

```
float _omega_comp_A (
            void  )
```

Complementary filter for wheel A angular velocity (encoder ticks and internal potentiometer)

5.6.2.3 _omega_comp_B()

```
float _omega_comp_B (
            void  )
```

Complementary filter for wheel B angular velocity (encoder ticks and internal potentiometer)

5.6.2.4 _omega_from_encA()

```
float _omega_from_encA (
            void  )
```

Returns the A wheel's angular velocity from encoder readings by measuring he time between consecutive encoder ticks

5.6.2.5 _omega_from_encB()

```
float _omega_from_encB (
            void  )
```

Returns the B wheel's angular velocity from encoder readings by measuring he time between consecutive encoder ticks

5.6.2.6 _omega_from_PMA()

```
float _omega_from_PMA (
            void  )
```

Returns the A wheel's angular velocity using the potentiometer readings

5.6.2.7 _omega_from_PMB()

```
float _omega_from_PMB (
             void  )
```

Returns the B wheel's angular velocity using the potentiometer readings

5.6.2.8 _thetaA()

```
float _thetaA (
             void  )
```

Returns the absolute rotation angle of motor A

5.6.2.9 _thetaB()

```
float _thetaB (
             void  )
```

Returns the absolute rotation angle of motor B

5.6.2.10 _ticksA()

```
int32_t _ticksA ( )
```

returns the number of ticks happened since boot to wheel A

5.6.2.11 _ticksB()

```
int32_t _ticksB ( )
```

returns the number of ticks happened since boot to wheel B

5.7 src/__pwm__.h File Reference

Header file for **pwm**.c.

```
#include <__swarm_wold__.h>
```

Functions

- void _set_pwm_0A (int input)
- void _set_pwm_0B (int input)
- void _set_pwm_1A (int input)
- void _set_pwm_1B (int input)

5.7.1 Detailed Description

Header file for **pwm**.c.

- File: **pwm**.h
- Compiler: GCC-AVR
- Supported devices: Tested on 328p
- AppNote: PWM driver

Author

Swarm robot graduation project workgroub
Mechatronics Program for the Distinguished
$Name$

Revision

1

$RCSfile$

Date

/19/2021 9:30:13 AM

5.7.2 Function Documentation

5.7.2.1 _set_pwm_0A()

```
void _set_pwm_0A (
            int input )
```

sets the PWM output of OCR0A (Motor A rear)

Parameters

in	*input*	
	input	a number (0 --> 255)

5.7.2.2 _set_pwm_0B()

```
void _set_pwm_0B (
            int input )
```

sets the PWM output of OCR0B (Motor B rear)

Parameters

in	*input*	
	input	a number (0 --> 255)

5.7.2.3 _set_pwm_1A()

```
void _set_pwm_1A (
            int input )
```

sets the PWM output of OCR1A (Motor A front)

Parameters

in	*input*	
	input	a number (0 --> 255)

5.7.2.4 _set_pwm_1B()

```
void _set_pwm_1B (
            int input )
```

sets the PWM output of OCR1B (Motor B front)

Parameters

in	*input*	
	input	a number (0 --> 255)

5.8 src/__swarm_wold__.h File Reference

Header file for dummy purposes.

```
#include <stdio.h>
#include <asf.h>
#include <avr/io.h>
#include <avr/interrupt.h>
#include <board.h>
#include <conf_board.h>
```

```
#include <util/delay.h>
#include "stdint.h"
#include <__adc__.h>
#include <__INT_0_1__.h>
#include <__pin_map.h>
#include <__timer0__.h>
#include <__timer1__.h>
#include <__timer2__.h>
#include <__usart__.h>
#include <__dc_control__.h>
#include <__odometry__.h>
#include <__pwm__.h>
#include <_pid_.h>
#include <__format__.h>
```

Macros

- #define **F_CPU** 16000000

5.8.1 Detailed Description

Header file for dummy purposes.

- File: **swarm_world**.h
- Compiler: GCC-AVR
- Supported devices: Tested on 328p
- AppNote: Discrete PID controller

Author

Swarm robot graduation project workgroub
Mechatronics Program for the Distinguished
$Name$

Revision

1

$RCSfile$

Date

/19/2021 9:31:42 AM

5.9 src/__timer0__.h File Reference

Header file for **timer0**.c.

```
#include <__swarm_wold__.h>
```

Functions

- void _timer0_init (void)
- void _timer0_init_prescaler (uint16_t prescaler)
- unsigned long _micros0 (void)
- unsigned long _millis0 (void)

5.9.1 Detailed Description

Header file for **timer0**.c.

- File: **timer0**.h
- Compiler: GCC-AVR
- Supported devices: Tested on 328p
- AppNote: Discrete PID controller

Author

Swarm robot graduation project workgroub
Mechatronics Program for the Distinguished
$Name$

Revision

1

$RCSfile$

Date

/19/2021 9:03:51 AM

5.9.2 Function Documentation

5.9.2.1 _micros0()

```
unsigned long _micros0 (
            void  )
```

Returns the microseconds passed since timer initialize

5.9.2.2 _millis0()

```
unsigned long _millis0 (
             void  )
```

Returns the milliseconds passed since timer initialize

5.9.2.3 _timer0_init()

```
void _timer0_init (
             void  )
```

Initialize Timer/Counter 0 with clock prescaler of 1024 (244 Hz overflow at 16 MHz)

5.9.2.4 _timer0_init_prescaler()

```
void _timer0_init_prescaler (
             uint16_t prescaler )
```

Initialize Timer/Counter 0 with costume prescaler (see the datasheet)

Parameters

in	*prescaler*	prescaler 1,2,4,8,255,1024

5.10 src/__timer1__.h File Reference

Header file for **timer1**.c.

```
#include <__swarm_wold__.h>
```

Macros

- #define **_TICK_US_1** 0.0625
- #define **_TICK_MS_1** 0.0000625
- #define **_TICK_US_0** 16
- #define **_TICK_MS_0** 0.016

Functions

- void _timer1_init (void)
- uint64_t **_micros1** (void)
- uint64_t **_millis1** (void)

Variables

- volatile uint8_t **_controler_flag_A**
- volatile uint8_t **_controler_flag_B**

5.10.1 Detailed Description

Header file for **timer1**.c.

- File: **timer1**.h
- Compiler: GCC-AVR
- Supported devices: Tested on 328p
- AppNote: Discrete PID controller

Author

Swarm robot graduation project workgroub
Mechatronics Program for the Distinguished
$Name$

Revision

1

$RCSfile$

Date

/9/2021 10:17:41 AM

5.10.2 Function Documentation

5.10.2.1 _timer1_init()

```
void _timer1_init (
            void  )
```

Initialize Timer/Counter 1 with 1 prescaler (244 MHz overflow frequency at 16 MHz)

5.11 src/__timer2__.h File Reference

Header file for **timer2**.c.

```
#include <__swarm_wold__.h>
```

Functions

- void _timer2_init (void)

5.11.1 Detailed Description

Header file for **timer2**.c.

- File: **timer2**.h
- Compiler: GCC-AVR
- Supported devices: Tested on 328p
- AppNote: Discrete PID controller

Author

> Swarm robot graduation project workgroub
> Mechatronics Program for the Distinguished
> $Name$

Revision

> 1

$RCSfile$

Date

> /20/2021 12:04:32 AM

5.11.2 Function Documentation

5.11.2.1 _timer2_init()

```
void _timer2_init (
            void  )
```

Initialize Timer/Counter 2 with 1 prescaler (244 Hz overflow frequency at 16 MHz)

5.12 src/__usart__.h File Reference

Header file for pid.c.

```
#include <asf.h>
#include <stdio.h>
#include <board.h>
#include <conf_board.h>
#include <util/delay.h>
```

Macros

- #define **BAUD** 57600
- #define **RX_BUFSIZE** 120
- #define **BRC** ((F_CPU/(16UL∗BAUD)) - 1)

Functions

- void usart_init (void)

5.12.1 Detailed Description

Header file for pid.c.

- File: **usart**.h
- Compiler: GCC-AVR
- Supported devices: Tested on 328p
- AppNote: USART module driver

Author

Swarm robot graduation project workgroub
Mechatronics Program for the Distinguished
$Name$

Revision

456

$RCSfile$

Date

021-5

5.12.2 Function Documentation

5.12.2.1 usart_init()

```
void usart_init (
             void  )
```

Initialize the USART module with

Parameters

BAUD	

5.13 src/_pid_.c File Reference

General PID implementation for AVR.

```
#include <_pid_.h>
```

Functions

- void pid_Init (int16_t p_factor, int16_t i_factor, int16_t d_factor, struct PID_DATA *pid)

  Initialisation of PID controller parameters.
- int16_t pid_Controller (int16_t setPoint, int16_t processValue, struct PID_DATA *pid_st)

  PID control algorithm.
- void pid_Reset_Integrator (pidData_t *pid_st)

  Resets the integrator.

5.13.1 Detailed Description

General PID implementation for AVR.

Discrete PID controller implementation. Set up by giving P/I/D terms to Init_PID(), and uses a struct PID_DATA to store internal values.

- File: pid.c
- Compiler: IAR EWAAVR 4.11A
- Supported devices: All AVR devices can be used.
- AppNote: AVR221 - Discrete PID controller

Author

Atmel Corporation: `http://www.atmel.com`
Support email: `avr@atmel.com`

$Name$

Revision

456

$RCSfile$

Date

2006-02-16 12:46:13 +0100 (to, 16 feb 2006)

5.13.2 Function Documentation

5.13.2.1 pid_Controller()

```
int16_t pid_Controller (
            int16_t setPoint,
            int16_t processValue,
            struct PID_DATA * pid_st )
```

PID control algorithm.

Calculates output from setpoint, process value and PID status.

Parameters

setPoint	Desired value.
processValue	Measured value.
pid_st	PID status struct.

5.13.2.2 pid_Init()

```
void pid_Init (
            int16_t p_factor,
            int16_t i_factor,
            int16_t d_factor,
            struct PID_DATA * pid )
```

Initialisation of PID controller parameters.

Initialise the variables used by the PID algorithm.

Parameters

p_factor	Proportional term.
i_factor	Integral term.
d_factor	Derivate term.
pid	Struct with PID status.

5.13.2.3 pid_Reset_Integrator()

```
void pid_Reset_Integrator (
            pidData_t * pid_st )
```

Resets the integrator.

Calling this function will reset the integrator in the PID regulator.

5.14 src/_pid_.h File Reference

Header file for pid.c.

```
#include "stdint.h"
```

Data Structures

- struct PID_DATA

  PID Status.

Macros

- #define **SCALING_FACTOR** 128
- #define MAX_INT INT16_MAX

  Maximum values.
- #define **MAX_LONG** INT32_MAX
- #define **MAX_I_TERM** (MAX_LONG / 2)
- #define **FALSE** 0
- #define **TRUE** 1

Typedefs

- typedef struct PID_DATA pidData_t

  PID Status.

Functions

- void pid_Init (int16_t p_factor, int16_t i_factor, int16_t d_factor, struct PID_DATA ∗pid)

  Initialisation of PID controller parameters.
- int16_t pid_Controller (int16_t setPoint, int16_t processValue, struct PID_DATA ∗pid_st)

  PID control algorithm.
- void pid_Reset_Integrator (pidData_t ∗pid_st)

  Resets the integrator.

Variables

- volatile uint8_t **flag**

5.14.1 Detailed Description

Header file for pid.c.

- File: pid.h
- Compiler: GCC-AVR
- Supported devices: Tested on 328p
- AppNote: Discrete PID controller

Author

Swarm robot graduation project workgroub
Mechatronics Program for the Distinguished
$Name$

Revision

456

$RCSfile$

Date

021

5.14.2 Macro Definition Documentation

5.14.2.1 MAX_INT

```
#define MAX_INT INT16_MAX
```

Maximum values.

Needed to avoid sign/overflow problems

5.14.3 Typedef Documentation

5.14.3.1 pidData_t

```
typedef struct PID_DATA pidData_t
```

PID Status.

Setpoints and data used by the PID control algorithm

5.14.4 Function Documentation

5.14.4.1 pid_Controller()

```
int16_t pid_Controller (
             int16_t setPoint,
             int16_t processValue,
             struct PID_DATA * pid_st )
```

PID control algorithm.

Calculates output from setpoint, process value and PID status.

Parameters

setPoint	Desired value.
processValue	Measured value.
pid_st	PID status struct.

5.14.4.2 pid_Init()

```
void pid_Init (
             int16_t p_factor,
             int16_t i_factor,
             int16_t d_factor,
             struct PID_DATA * pid )
```

Initialisation of PID controller parameters.

Initialise the variables used by the PID algorithm.

Parameters

p_factor	Proportional term.
i_factor	Integral term.
d_factor	Derivate term.
pid	Struct with PID status.

5.14.4.3 pid_Reset_Integrator()

```
void pid_Reset_Integrator (
             pidData_t * pid_st )
```

Resets the integrator.

Calling this function will reset the integrator in the PID regulator.

5.15 src/asf.h File Reference

Autogenerated API include file for the Atmel Software Framework (ASF)

```
#include <user_board.h>
#include <board.h>
#include <interrupt.h>
#include <compiler.h>
#include <status_codes.h>
#include <parts.h>
```

Macros

- #define **F_CPU** 16000000

5.15.1 Detailed Description

Autogenerated API include file for the Atmel Software Framework (ASF)

\asf_license_start

Index