

## Swarm Firmware AVR

Generated by Doxygen 1.9.1



<b>1 License</b>	<b>1</b>
<b>2 Data Structure Index</b>	<b>3</b>
2.1 Data Structures . . . . .	3
<b>3 File Index</b>	<b>5</b>
3.1 File List . . . . .	5
<b>4 Data Structure Documentation</b>	<b>7</b>
4.1 <code>_i_pos</code> Struct Reference . . . . .	7
4.1.1 Detailed Description . . . . .	7
4.2 <code>_omega</code> Struct Reference . . . . .	7
4.2.1 Detailed Description . . . . .	8
4.3 <code>_theta</code> Struct Reference . . . . .	8
4.3.1 Detailed Description . . . . .	8
4.4 <code>mat2</code> Struct Reference . . . . .	8
4.4.1 Detailed Description . . . . .	8
4.5 <code>PID_DATA</code> Struct Reference . . . . .	9
4.5.1 Detailed Description . . . . .	9
4.6 <code>point</code> Struct Reference . . . . .	9
4.6.1 Detailed Description . . . . .	10
4.6.2 Field Documentation . . . . .	10
4.6.2.1 <code>x</code> . . . . .	10
4.6.2.2 <code>y</code> . . . . .	10
4.7 <code>pos</code> Struct Reference . . . . .	10
4.7.1 Detailed Description . . . . .	11
4.7.2 Field Documentation . . . . .	11
4.7.2.1 <code>th</code> . . . . .	11
4.7.2.2 <code>x</code> . . . . .	11
4.7.2.3 <code>y</code> . . . . .	11
4.8 <code>pos_dot</code> Struct Reference . . . . .	11
4.8.1 Detailed Description . . . . .	12
4.8.2 Field Documentation . . . . .	12
4.8.2.1 <code>th_dot</code> . . . . .	12
4.8.2.2 <code>x_dot</code> . . . . .	12
4.8.2.3 <code>y_dot</code> . . . . .	12
<b>5 File Documentation</b>	<b>13</b>
5.1 <code>src/___adc__.h</code> File Reference . . . . .	13
5.1.1 Detailed Description . . . . .	13
5.1.2 Function Documentation . . . . .	14
5.1.2.1 <code>__adc_init()</code> . . . . .	14
5.1.2.2 <code>__adc_read()</code> . . . . .	14
5.2 <code>src/___dc_control__.h</code> File Reference . . . . .	14

5.2.1 Detailed Description . . . . .	15
5.2.2 Function Documentation . . . . .	15
5.2.2.1 _break_motor() . . . . .	15
5.2.2.2 _command() . . . . .	16
5.2.2.3 _dc_controller_loop() . . . . .	16
5.2.2.4 _init_dc_control() . . . . .	16
5.2.2.5 _ref() . . . . .	16
5.2.2.6 _sens() . . . . .	17
5.2.2.7 _set_speed() . . . . .	17
5.2.2.8 _update_controller() . . . . .	18
5.3 src/__format__.h File Reference . . . . .	18
5.3.1 Detailed Description . . . . .	18
5.3.2 Function Documentation . . . . .	19
5.3.2.1 _float_to_printable() . . . . .	19
5.4 src/__INT_0_1__.h File Reference . . . . .	19
5.4.1 Detailed Description . . . . .	19
5.4.2 Function Documentation . . . . .	20
5.4.2.1 _interrupt0_enable() . . . . .	20
5.5 src/__kinematics__.h File Reference . . . . .	20
5.5.1 Detailed Description . . . . .	21
5.5.2 Function Documentation . . . . .	21
5.5.2.1 _dead_reckon() . . . . .	21
5.5.2.2 _inertial_to_pos_dot() . . . . .	22
5.5.2.3 _omega_to_intertial() . . . . .	22
5.5.2.4 _thetaLR_to_pos() . . . . .	22
5.5.2.5 _update_omega() . . . . .	23
5.5.2.6 _update_thetaLR() . . . . .	23
5.6 src/__odometry__.h File Reference . . . . .	23
5.6.1 Detailed Description . . . . .	24
5.6.2 Function Documentation . . . . .	24
5.6.2.1 _insertion_sort() . . . . .	24
5.6.2.2 _omega_comp_A() . . . . .	25
5.6.2.3 _omega_comp_B() . . . . .	25
5.6.2.4 _omega_from_encA() . . . . .	25
5.6.2.5 _omega_from_encB() . . . . .	25
5.6.2.6 _omega_from_PMA() . . . . .	25
5.6.2.7 _omega_from_PMB() . . . . .	26
5.6.2.8 _thetaA() . . . . .	26
5.6.2.9 _thetaB() . . . . .	26
5.6.2.10 _ticksA() . . . . .	26
5.6.2.11 _ticksB() . . . . .	26
5.7 src/__pwm__.h File Reference . . . . .	26

5.7.1 Detailed Description . . . . .	27
5.7.2 Function Documentation . . . . .	27
5.7.2.1 _set_pwm_0A() . . . . .	27
5.7.2.2 _set_pwm_0B() . . . . .	27
5.7.2.3 _set_pwm_1A() . . . . .	28
5.7.2.4 _set_pwm_1B() . . . . .	28
5.8 src/_swarm_wold_.h File Reference . . . . .	28
5.8.1 Detailed Description . . . . .	29
5.9 src/_timer0_.h File Reference . . . . .	29
5.9.1 Detailed Description . . . . .	30
5.9.2 Function Documentation . . . . .	30
5.9.2.1 _micros0() . . . . .	30
5.9.2.2 _millis0() . . . . .	31
5.9.2.3 _timer0_init() . . . . .	31
5.9.2.4 _timer0_init_prescaler() . . . . .	31
5.10 src/_timer1_.h File Reference . . . . .	31
5.10.1 Detailed Description . . . . .	32
5.10.2 Function Documentation . . . . .	32
5.10.2.1 _timer1_init() . . . . .	32
5.11 src/_timer2_.h File Reference . . . . .	32
5.11.1 Detailed Description . . . . .	33
5.11.2 Function Documentation . . . . .	33
5.11.2.1 _timer2_init() . . . . .	33
5.12 src/_usart_.h File Reference . . . . .	33
5.12.1 Detailed Description . . . . .	34
5.12.2 Function Documentation . . . . .	34
5.12.2.1 usart_init() . . . . .	34
5.13 src/_pid_.c File Reference . . . . .	35
5.13.1 Detailed Description . . . . .	35
5.13.2 Function Documentation . . . . .	36
5.13.2.1 pid_Controller() . . . . .	36
5.13.2.2 pid_Init() . . . . .	36
5.13.2.3 pid_Reset_Integrator() . . . . .	36
5.14 src/_pid_.h File Reference . . . . .	37
5.14.1 Detailed Description . . . . .	38
5.14.2 Macro Definition Documentation . . . . .	38
5.14.2.1 MAX_INT . . . . .	38
5.14.3 Typedef Documentation . . . . .	38
5.14.3.1 pidData_t . . . . .	38
5.14.4 Function Documentation . . . . .	39
5.14.4.1 pid_Controller() . . . . .	39
5.14.4.2 pid_Init() . . . . .	39

5.14.4.3 pid_Reset_Integrator() . . . . .	39
5.15 src/asf.h File Reference . . . . .	40
5.15.1 Detailed Description . . . . .	40
<b>Index</b>	<b>41</b>

# Chapter 1

## License

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The name of Atmel may not be used to endorse or promote products derived from this software without specific prior written permission.
4. This software may only be redistributed and used in connection with an Atmel microcontroller product.

THIS SOFTWARE IS PROVIDED BY ATMEL "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT ARE EXPRESSLY AND SPECIFICALLY DISCLAIMED. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

\asf\_license\_stop





## Chapter 2

# Data Structure Index

### 2.1 Data Structures

Here are the data structures with brief descriptions:

<a href="#">_i_pos</a>	7
<a href="#">_omega</a>	7
<a href="#">_theta</a>	8
<a href="#">mat2</a>	8
<a href="#">PID_DATA</a>	
PID Status	9
<a href="#">point</a>	9
<a href="#">pos</a>	10
<a href="#">pos_dot</a>	11



## Chapter 3

# File Index

### 3.1 File List

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

src/	<a href="#">__adc__.h</a>		
	Header file for <b>adc.c</b>	13	
src/	<a href="#">__dc_control__.h</a>		
	Header file for <b>dc_control.c</b>	14	
src/	<a href="#">__format__.h</a>		
	Header file for <b>format.c</b>	18	
src/	<a href="#">__INT_0_1__.h</a>		
	Header file for <b>INT_0_1.c</b>	19	
src/	<a href="#">__kinematics__.h</a>		
	Header file for <b>kinematics.c</b>	20	
src/	<a href="#">__odometry__.h</a>		
	Header file for <b>odometry.c</b>	23	
src/	<a href="#">__pin_map.h</a>		??
src/	<a href="#">__pwm__.h</a>		
	Header file for <b>pwm.c</b>	26	
src/	<a href="#">__swarm_wold__.h</a>		
	Header file for dummy purposes	28	
src/	<a href="#">__timer0__.h</a>		
	Header file for <b>timer0.c</b>	29	
src/	<a href="#">__timer1__.h</a>		
	Header file for <b>timer1.c</b>	31	
src/	<a href="#">__timer2__.h</a>		
	Header file for <b>timer2.c</b>	32	
src/	<a href="#">__usart__.h</a>		
	Header file for <b>pid.c</b>	33	
src/	<a href="#">__pid_.c</a>		
	General PID implementation for AVR	35	
src/	<a href="#">__pid_.h</a>		
	Header file for <b>pid.c</b>	37	
src/	<a href="#">asf.h</a>		
	Autogenerated API include file for the Atmel Software Framework (ASF)	40	



## 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

<i>a</i>	
----------	--

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

<a href="#">x</a>	
-------------------	--

#### 4.6.2.2 y

float [y](#)

##### Parameters

<a href="#">y</a>	
-------------------	--

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

<i>Theta</i>	(\theta)
--------------	----------

#### 4.7.2.2 x

float x

##### Parameters

<i>x</i>	
----------	--

#### 4.7.2.3 y

float y

##### Parameters

<i>y</i>	
----------	--

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 ( $\frac{d \text{ pos}}{dt}$ )

### 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

<i>channel</i>	
----------------	--

Parameters

in	<i>channel</i>	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**

<i>motor</i>	
--------------	--

**Parameters**

in	<i>motor</i>	motor MA or MB.
----	--------------	-----------------

**5.2.2.2 \_command()**

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

Output command module for

**Parameters**

<i>motor</i>	of value
<i>inputValue</i>	

**Parameters**

in	<i>motor</i>	motor MA or MB.
in	<i>inputValue</i>	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

<i>motor</i>	
--------------	--

## Parameters

in	<i>motor</i>	motor MA or MB.
----	--------------	-----------------

### 5.2.2.6 \_\_sens()

```
float __sens (
    uint8_t motor )
```

sensor module for

## Parameters

<i>motor</i>	
--------------	--

## Parameters

in	<i>motor</i>	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

<i>value</i>	can be negative or positive for direction
--------------	---

## Parameters

in	<i>motor</i>	motor MA or MB.
in	<i>value</i>	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

<code>in</code>	<code>motor</code>	motor MA or MB.
-----------------	--------------------	-----------------

## 5.3 `src/___format___` 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 variable to string

#### Parameters

<i>in</i>	<i>input</i>	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	<i>trigger</i>	trigger mode, see file <a href="#">__INT_0_1__.h</a>
----	----------------	--

## 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\\_inertial](#) (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 **kinematics.c**.

- File: **kinematics.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	<i>_pos</i>	pos
in	<i>_pos_dot</i>	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	<i>inertial</i>	struct of inertial pos
in	<i>_pos</i>	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	<i>input</i>	struct of wheels velocity
	<i>_omega</i>	

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	<i>th</i>	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	<i>input</i>	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	<i>input</i>	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	<i>arr</i>	pointer to start of sorting
in	<i>n</i>	offset from
	<i>arr[]</i>	

**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 the 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 the 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	<i>input</i>	
	<i>input</i>	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	<i>input</i>	
	<i>input</i>	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	<i>input</i>	
	<i>input</i>	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	<i>input</i>	
	<i>input</i>	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	<i>prescaler</i>	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](mailto: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

<i>setPoint</i>	Desired value.
<i>processValue</i>	Measured value.
<i>pid_st</i>	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

<i>p_factor</i>	Proportional term.
<i>i_factor</i>	Integral term.
<i>d_factor</i>	Derivate term.
<i>pid</i>	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

<i>setPoint</i>	Desired value.
<i>processValue</i>	Measured value.
<i>pid_st</i>	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

<i>p_factor</i>	Proportional term.
<i>i_factor</i>	Integral term.
<i>d_factor</i>	Derivate term.
<i>pid</i>	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)

Copyright (c) 2012 Atmel Corporation. All rights reserved.

\asf\_license\_start

# Index

- [\\_\\_INT\\_0\\_1\\_\\_.h](#)
  - [\\_interrupt0\\_enable, 20](#)
- [\\_\\_adc\\_\\_.h](#)
  - [\\_adc\\_init, 14](#)
  - [\\_adc\\_read, 14](#)
- [\\_\\_dc\\_control\\_\\_.h](#)
  - [\\_break\\_motor, 15](#)
  - [\\_command, 16](#)
  - [\\_dc\\_controller\\_loop, 16](#)
  - [\\_init\\_dc\\_control, 16](#)
  - [\\_ref, 16](#)
  - [\\_sens, 17](#)
  - [\\_set\\_speed, 17](#)
  - [\\_update\\_controller, 17](#)
- [\\_\\_format\\_\\_.h](#)
  - [\\_float\\_to\\_printable, 19](#)
- [\\_\\_kinematics\\_\\_.h](#)
  - [\\_dead\\_reckon, 21](#)
  - [\\_inertial\\_to\\_pos\\_dot, 22](#)
  - [\\_omega\\_to\\_intertial, 22](#)
  - [\\_thetaLR\\_to\\_pos, 22](#)
  - [\\_update\\_omega, 23](#)
  - [\\_update\\_thetaLR, 23](#)
- [\\_\\_odometry\\_\\_.h](#)
  - [\\_insertion\\_sort, 24](#)
  - [\\_omega\\_comp\\_A, 25](#)
  - [\\_omega\\_comp\\_B, 25](#)
  - [\\_omega\\_from\\_PMA, 25](#)
  - [\\_omega\\_from\\_PMB, 25](#)
  - [\\_omega\\_from\\_encA, 25](#)
  - [\\_omega\\_from\\_encB, 25](#)
  - [\\_thetaA, 26](#)
  - [\\_thetaB, 26](#)
  - [\\_ticksA, 26](#)
  - [\\_ticksB, 26](#)
- [\\_\\_pwm\\_\\_.h](#)
  - [\\_set\\_pwm\\_0A, 27](#)
  - [\\_set\\_pwm\\_0B, 27](#)
  - [\\_set\\_pwm\\_1A, 28](#)
  - [\\_set\\_pwm\\_1B, 28](#)
- [\\_\\_timer0\\_\\_.h](#)
  - [\\_micros0, 30](#)
  - [\\_millis0, 30](#)
  - [\\_timer0\\_init, 31](#)
  - [\\_timer0\\_init\\_prescaler, 31](#)
- [\\_\\_timer1\\_\\_.h](#)
  - [\\_timer1\\_init, 32](#)
- [\\_\\_timer2\\_\\_.h](#)
  - [\\_timer2\\_init, 33](#)
- [\\_\\_usart\\_\\_.h](#)
  - [usart\\_init, 34](#)
- [\\_adc\\_init](#)
  - [\\_\\_adc\\_\\_.h, 14](#)
- [\\_adc\\_read](#)
  - [\\_\\_adc\\_\\_.h, 14](#)
- [\\_break\\_motor](#)
  - [\\_\\_dc\\_control\\_\\_.h, 15](#)
- [\\_command](#)
  - [\\_\\_dc\\_control\\_\\_.h, 16](#)
- [\\_dc\\_controller\\_loop](#)
  - [\\_\\_dc\\_control\\_\\_.h, 16](#)
- [\\_dead\\_reckon](#)
  - [\\_\\_kinematics\\_\\_.h, 21](#)
- [\\_float\\_to\\_printable](#)
  - [\\_\\_format\\_\\_.h, 19](#)
- [\\_i\\_pos, 7](#)
- [\\_inertial\\_to\\_pos\\_dot](#)
  - [\\_\\_kinematics\\_\\_.h, 22](#)
- [\\_init\\_dc\\_control](#)
  - [\\_\\_dc\\_control\\_\\_.h, 16](#)
- [\\_insertion\\_sort](#)
  - [\\_\\_odometry\\_\\_.h, 24](#)
- [\\_interrupt0\\_enable](#)
  - [\\_\\_INT\\_0\\_1\\_\\_.h, 20](#)
- [\\_micros0](#)
  - [\\_\\_timer0\\_\\_.h, 30](#)
- [\\_millis0](#)
  - [\\_\\_timer0\\_\\_.h, 30](#)
- [\\_omega, 7](#)
- [\\_omega\\_comp\\_A](#)
  - [\\_\\_odometry\\_\\_.h, 25](#)
- [\\_omega\\_comp\\_B](#)
  - [\\_\\_odometry\\_\\_.h, 25](#)
- [\\_omega\\_from\\_PMA](#)
  - [\\_\\_odometry\\_\\_.h, 25](#)
- [\\_omega\\_from\\_PMB](#)
  - [\\_\\_odometry\\_\\_.h, 25](#)
- [\\_omega\\_from\\_encA](#)
  - [\\_\\_odometry\\_\\_.h, 25](#)
- [\\_omega\\_from\\_encB](#)
  - [\\_\\_odometry\\_\\_.h, 25](#)
- [\\_omega\\_to\\_intertial](#)
  - [\\_\\_kinematics\\_\\_.h, 22](#)
- [\\_pid.c](#)
  - [pid\\_Controller, 36](#)
  - [pid\\_Init, 36](#)
  - [pid\\_Reset\\_Integrator, 36](#)
- [\\_pid.h](#)

MAX\_INT, 38  
 pid\_Controller, 39  
 pid\_Init, 39  
 pid\_Reset\_Integrator, 39  
 pidData\_t, 38  
 \_ref  
   \_\_dc\_control\_\_.h, 16  
 \_sens  
   \_\_dc\_control\_\_.h, 17  
 \_set\_pwm\_0A  
   \_\_pwm\_\_.h, 27  
 \_set\_pwm\_0B  
   \_\_pwm\_\_.h, 27  
 \_set\_pwm\_1A  
   \_\_pwm\_\_.h, 28  
 \_set\_pwm\_1B  
   \_\_pwm\_\_.h, 28  
 \_set\_speed  
   \_\_dc\_control\_\_.h, 17  
 \_theta, 8  
 \_thetaA  
   \_\_odometry\_\_.h, 26  
 \_thetaB  
   \_\_odometry\_\_.h, 26  
 \_thetaLR\_to\_pos  
   \_\_kinematics\_\_.h, 22  
 \_ticksA  
   \_\_odometry\_\_.h, 26  
 \_ticksB  
   \_\_odometry\_\_.h, 26  
 \_timer0\_init  
   \_\_timer0\_\_.h, 31  
 \_timer0\_init\_prescaler  
   \_\_timer0\_\_.h, 31  
 \_timer1\_init  
   \_\_timer1\_\_.h, 32  
 \_timer2\_init  
   \_\_timer2\_\_.h, 33  
 \_update\_controller  
   \_\_dc\_control\_\_.h, 17  
 \_update\_omega  
   \_\_kinematics\_\_.h, 23  
 \_update\_thetaLR  
   \_\_kinematics\_\_.h, 23  
  
 mat2, 8  
 MAX\_INT  
   \_pid\_.h, 38  
  
 pid\_Controller  
   \_pid\_.c, 36  
   \_pid\_.h, 39  
 PID\_DATA, 9  
 pid\_Init  
   \_pid\_.c, 36  
   \_pid\_.h, 39  
 pid\_Reset\_Integrator  
   \_pid\_.c, 36  
   \_pid\_.h, 39  
  
 pidData\_t  
   \_pid\_.h, 38  
 point, 9  
   x, 10  
   y, 10  
 pos, 10  
   th, 11  
   x, 11  
   y, 11  
 pos\_dot, 11  
   th\_dot, 12  
   x\_dot, 12  
   y\_dot, 12  
  
 src/\_adc\_.h, 13  
 src/\_dc\_control\_\_.h, 14  
 src/\_format\_.h, 18  
 src/\_INT\_0\_1\_.h, 19  
 src/\_kinematics\_\_.h, 20  
 src/\_odometry\_\_.h, 23  
 src/\_pwm\_\_.h, 26  
 src/\_swarm\_wold\_\_.h, 28  
 src/\_timer0\_\_.h, 29  
 src/\_timer1\_\_.h, 31  
 src/\_timer2\_\_.h, 32  
 src/\_usart\_\_.h, 33  
 src/\_pid\_.c, 35  
 src/\_pid\_.h, 37  
 src/asf.h, 40  
  
 th  
   pos, 11  
 th\_dot  
   pos\_dot, 12  
  
 usart\_init  
   \_\_usart\_\_.h, 34  
  
 x  
   point, 10  
   pos, 11  
 x\_dot  
   pos\_dot, 12  
  
 y  
   point, 10  
   pos, 11  
 y\_dot  
   pos\_dot, 12