

DCAITI Robot Hardware

1.0

Generated by Doxygen 1.9.1

1 Hierarchical Index	1
1.1 Class Hierarchy	1
2 Class Index	3
2.1 Class List	3
3 File Index	5
3.1 File List	5
4 Class Documentation	7
4.1 GearedMotor Class Reference	7
4.1.1 Constructor & Destructor Documentation	8
4.1.1.1 GearedMotor()	8
4.1.2 Member Function Documentation	9
4.1.2.1 getGearedSpeedRPM()	9
4.1.2.2 getPosition()	9
4.1.2.3 getRatio()	10
4.1.2.4 setGearedSpeedRPM() [1/2]	10
4.1.2.5 setGearedSpeedRPM() [2/2]	10
4.1.2.6 setRatio()	11
4.2 ISRVars Struct Reference	11
4.2.1 Member Data Documentation	11
4.2.1.1 currDirection	11
4.2.1.2 ISRfunc	12
4.2.1.3 pinIRQ	12
4.2.1.4 pinIRQB	12
4.2.1.5 pulseEndMicros	12
4.2.1.6 pulses	12
4.2.1.7 pulseStartMicros	12
4.2.1.8 speedPPS	12
4.3 Motor Class Reference	13
4.3.1 Constructor & Destructor Documentation	14
4.3.1.1 Motor()	14
4.3.2 Member Function Documentation	15
4.3.2.1 advancePWM()	15
4.3.2.2 backoffPWM()	16
4.3.2.3 debugger()	16
4.3.2.4 delayMS()	16
4.3.2.5 getCurrDir()	17
4.3.2.6 getCurrPulse()	17
4.3.2.7 getDesiredDir()	18
4.3.2.8 getPinDir()	18
4.3.2.9 getPinIRQ()	18

4.3.2.10 getPinIRQB()	18
4.3.2.11 getPinPWM()	18
4.3.2.12 getPWM()	19
4.3.2.13 getSpeedPPS()	19
4.3.2.14 getSpeedRPM()	19
4.3.2.15 PIDDisable()	20
4.3.2.16 PIDEnable()	20
4.3.2.17 PIDGetSpeedRPMDesired()	21
4.3.2.18 PIDGetStatus()	21
4.3.2.19 PIDRegulate()	21
4.3.2.20 PIDReset()	22
4.3.2.21 PIDSetSpeedRPMDesired()	23
4.3.2.22 PIDSetup()	23
4.3.2.23 resetCurrPulse()	24
4.3.2.24 reverseDesiredDir()	25
4.3.2.25 runPWM()	25
4.3.2.26 setCurrDir()	26
4.3.2.27 setCurrPulse()	26
4.3.2.28 setDesiredDir()	27
4.3.2.29 setSpeedRPM() [1/2]	27
4.3.2.30 setSpeedRPM() [2/2]	28
4.3.2.31 setupInterrupt()	28
4.3.3 Member Data Documentation	28
4.3.3.1 isr	28
4.4 MotorWheel Class Reference	29
4.4.1 Constructor & Destructor Documentation	30
4.4.1.1 MotorWheel()	30
4.4.2 Member Function Documentation	30
4.4.2.1 getCirMM()	30
4.4.2.2 getSpeedCMPM()	31
4.4.2.3 getSpeedMMPS()	31
4.4.2.4 setCirMM()	32
4.4.2.5 setGearedRPM()	32
4.4.2.6 setSpeedCMPM() [1/2]	33
4.4.2.7 setSpeedCMPM() [2/2]	33
4.4.2.8 setSpeedMMPS() [1/2]	34
4.4.2.9 setSpeedMMPS() [2/2]	34
4.5 PCintPort::PCintPin Class Reference	34
4.5.1 Constructor & Destructor Documentation	35
4.5.1.1 PCintPin()	35
4.5.2 Member Data Documentation	35
4.5.2.1 PCintFunc	35

4.5.2.2 PCIntMask	35
4.5.2.3 PCIntMode	35
4.5.2.4 pinDataAlloc	36
4.6 PCIntPort Class Reference	36
4.6.1 Member Function Documentation	37
4.6.1.1 addPin()	37
4.6.1.2 attachInterrupt()	37
4.6.1.3 delPin()	38
4.6.1.4 detachInterrupt()	38
4.6.1.5 PCInt()	38
4.6.2 Member Data Documentation	39
4.6.2.1 PCICRbit	39
4.6.2.2 PCIntLast	39
4.6.2.3 pcIntPins	39
4.6.2.4 pcIntPorts	39
4.6.2.5 pcmask	39
4.6.2.6 portInputReg	39
4.7 PID Class Reference	40
4.7.1 Constructor & Destructor Documentation	41
4.7.1.1 PID() [1/2]	41
4.7.1.2 PID() [2/2]	41
4.7.2 Member Function Documentation	42
4.7.2.1 Compute()	42
4.7.2.2 GetD_Param()	42
4.7.2.3 GetI_Param()	42
4.7.2.4 GetINMax()	42
4.7.2.5 GetINMin()	42
4.7.2.6 GetMode()	43
4.7.2.7 GetOUTMax()	43
4.7.2.8 GetOUTMin()	43
4.7.2.9 GetP_Param()	43
4.7.2.10 GetSampleTime()	43
4.7.2.11 JustCalculated()	43
4.7.2.12 Reset()	44
4.7.2.13 SetInputLimits()	44
4.7.2.14 SetMode()	44
4.7.2.15 SetOutputLimits()	45
4.7.2.16 SetSampleTime()	45
4.7.2.17 SetTunings()	45
4.8 UARTCom Class Reference	46
4.8.1 Member Function Documentation	46
4.8.1.1 init()	46

4.8.1.2 read()	47
4.8.1.3 send()	48
4.8.2 Member Data Documentation	48
4.8.2.1 data	49
4.9 UCommands Class Reference	49
4.9.1 Member Function Documentation	49
4.9.1.1 setPIDParameter()	49
4.9.1.2 setSetpoint()	50
4.10 Utils Class Reference	51
4.10.1 Member Function Documentation	51
4.10.1.1 bytesToFloat()	51
4.10.1.2 bytesToInt()	52
4.11 UValue Class Reference	53
4.11.1 Member Function Documentation	53
4.11.1.1 sendBothEncoderValues()	53
4.11.1.2 sendEncoderValue()	54
5 File Documentation	57
5.1 include/communication/UARTCom.hpp File Reference	57
5.1.1 Detailed Description	58
5.2 include/communication/UCommands.hpp File Reference	58
5.2.1 Detailed Description	59
5.3 include/communication/UValue.hpp File Reference	59
5.3.1 Detailed Description	60
5.4 include/constants.h File Reference	61
5.4.1 Macro Definition Documentation	62
5.4.1.1 CHAIN_LENGTH_M	62
5.4.1.2 COMMAND_POS	62
5.4.1.3 COMPONENT_ENDBIT	63
5.4.1.4 COMPONENT_POS	63
5.4.1.5 COMPONENT_STARTBIT	63
5.4.1.6 COUNT_PER_ROTATION	63
5.4.1.7 CP_ALL_M_BV	63
5.4.1.8 CP_DEFAULT_BV	63
5.4.1.9 CP_LEFT_M_BV	63
5.4.1.10 CP_RIGHT_M_BV	63
5.4.1.11 ENC_1_A	64
5.4.1.12 ENC_1_B	64
5.4.1.13 ENC_2_A	64
5.4.1.14 ENC_2_B	64
5.4.1.15 FRAME_TYPE_ENDBIT	64
5.4.1.16 FRAME_TYPE_POS	64

5.4.1.17 FRAME_TYPE_STARTBIT	64
5.4.1.18 FT_REQUEST_BV	64
5.4.1.19 FT_RESPONSE_BV	65
5.4.1.20 FT_VALUE_BV	65
5.4.1.21 KD_1	65
5.4.1.22 KD_2	65
5.4.1.23 KI_1	65
5.4.1.24 KI_2	65
5.4.1.25 KP_1	65
5.4.1.26 KP_2	65
5.4.1.27 LAST	66
5.4.1.28 M1_DIR_PIN	66
5.4.1.29 M1_EN_PIN	66
5.4.1.30 M2_DIR_PIN	66
5.4.1.31 M2_EN_PIN	66
5.4.1.32 MAX_PACKET_LENGTH	66
5.4.1.33 MAX_SPEED	66
5.4.1.34 MID	67
5.4.1.35 NUM_MOTORS	67
5.4.1.36 P_GAIN	67
5.4.1.37 PACKAGE_LENGTH_NO_P	67
5.4.1.38 PACKAGE_PAUSE_MS	67
5.4.1.39 PAYLOAD_LENGTH_POS	67
5.4.1.40 PAYLOAD_POS	67
5.4.1.41 PRINT_RATE_MS	68
5.4.1.42 REQUEST_DRIVE_MOTOR	68
5.4.1.43 REQUEST_RESET_TIMESTAMP	68
5.4.1.44 RX_328_PIN	68
5.4.1.45 SPEED_THRESHOLD	68
5.4.1.46 START_BYTE	68
5.4.1.47 START_BYTE_POS	68
5.4.1.48 TIMESTAMP_POS	68
5.4.1.49 TX_328_PIN	69
5.4.1.50 UART_BAUD_RATE	69
5.4.1.51 UART_TRANSMIT_MS	69
5.4.1.52 VEL2TORQUE_RATIO	69
5.4.1.53 VEL_UPDATE_RATE_MS	69
5.4.2 Typedef Documentation	69
5.4.2.1 U_Component	69
5.4.2.2 U_FrameType	69
5.4.2.3 U_Request	69
5.4.3 Enumeration Type Documentation	69

5.4.3.1 components	69
5.4.3.2 frametypes	70
5.4.3.3 requests	70
5.5 include/misc/Utils.hpp File Reference	70
5.5.1 Detailed Description	71
5.6 lib/MotorWheel/MotorWheel.cpp File Reference	72
5.7 lib/MotorWheel/MotorWheel.h File Reference	72
5.7.1 Macro Definition Documentation	73
5.7.1.1 Baudrate	73
5.7.1.2 CIRMM	74
5.7.1.3 CPR	74
5.7.1.4 debug	74
5.7.1.5 DIR_ADVANCE	74
5.7.1.6 DIR_BACKOFF	74
5.7.1.7 DIR_INVERSE	74
5.7.1.8 irqISR	74
5.7.1.9 KC	75
5.7.1.10 MAX_PWM	75
5.7.1.11 MAX_SPEEDRPM	75
5.7.1.12 MICROS_PER_SEC	75
5.7.1.13 PI	75
5.7.1.14 PIN_UNDEFINED	75
5.7.1.15 REDUCTION_RATIO	75
5.7.1.16 REF_VOLT	75
5.7.1.17 SAMPLETIME	76
5.7.1.18 SEC_PER_MIN	76
5.7.1.19 SPEEDPPS2SPEEDRPM	76
5.7.1.20 TAUD	76
5.7.1.21 TAUI	76
5.7.1.22 TRIGGER	76
5.8 lib/PID_Beta6/fuzzy_table.h File Reference	76
5.9 lib/PID_Beta6/PID_Beta6.cpp File Reference	77
5.10 lib/PID_Beta6/PID_Beta6.h File Reference	77
5.10.1 Macro Definition Documentation	77
5.10.1.1 AUTO	78
5.10.1.2 LIBRARY_VERSION	78
5.10.1.3 MANUAL	78
5.11 lib/PinChangeInt/PinChangeInt.cpp File Reference	78
5.11.1 Function Documentation	78
5.11.1.1 ISR() [1/3]	79
5.11.1.2 ISR() [2/3]	79
5.11.1.3 ISR() [3/3]	79

5.12 lib/PinChangeInt/PinChangeInt.h File Reference	80
5.12.1 Macro Definition Documentation	81
5.12.1.1 INLINE_PCINT	81
5.12.1.2 MAX_PIN_CHANGE_PINS	81
5.12.1.3 PCattachInterrupt	81
5.12.1.4 PCdetachInterrupt	81
5.12.2 Typedef Documentation	81
5.12.2.1 PCIntvoidFuncPtr	81
5.13 lib/PinChangeInt/PinChangeIntConfig.h File Reference	82
5.14 src/communication/UARTCom.cpp File Reference	82
5.14.1 Function Documentation	83
5.14.1.1 clearPackage()	83
5.14.1.2 computeChecksum()	84
5.14.1.3 correctChecksum()	84
5.14.1.4 fromComponent()	85
5.14.1.5 fromFrameType()	86
5.14.1.6 handlePackage()	86
5.14.1.7 handleRequest()	87
5.14.1.8 parseComponent()	88
5.14.1.9 parseFrameType()	88
5.14.1.10 parseRequest()	89
5.14.1.11 parseTimestamp()	89
5.14.1.12 showPackage()	90
5.14.2 Variable Documentation	90
5.14.2.1 currentParseIndex	90
5.14.2.2 currentTimestamp	90
5.14.2.3 payloadLength	90
5.14.2.4 wheelLeft	91
5.14.2.5 wheelRight	91
5.15 src/communication/UCommands.cpp File Reference	91
5.15.1 Variable Documentation	91
5.15.1.1 wheelLeft	91
5.15.1.2 wheelRight	92
5.16 src/communication/UValue.cpp File Reference	92
5.17 src/main.cpp File Reference	92
5.17.1 Macro Definition Documentation	93
5.17.1.1 MICROS_PER_SEC	93
5.17.2 Function Documentation	93
5.17.2.1 irqISR() [1/2]	93
5.17.2.2 irqISR() [2/2]	93
5.17.2.3 loop()	94
5.17.2.4 setup()	94

5.17.3 Variable Documentation	94
5.17.3.1 controlTimer	94
5.17.3.2 speed	95
5.17.3.3 uartTransmitTimer	95
5.17.3.4 wheelLeft	95
5.17.3.5 wheelRight	95
5.18 src/misc/Utils.cpp File Reference	95
Index	97

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

ISRVars	11
PCintPort::PCintPin	34
PCintPort	36
PID	40
Motor	13
GearedMotor	7
MotorWheel	29
UARTCom	46
UCommands	49
Utils	51
UValue	53

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

GearedMotor	7
ISRVars	11
Motor	13
MotorWheel	29
PCintPort::PCintPin	34
PCintPort	36
PID	40
UARTCom	46
UCommands	49
Utils	51
UValue	53

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

include/ constants.h	61
include/communication/ UARTCom.hpp	
Class to manage uart communication	57
include/communication/ UCommands.hpp	
Implementation of commands according to uart protocol	58
include/communication/ UValue.hpp	
Class to send out encoder values	59
include/misc/ Utils.hpp	
Useful converting and computing functions	70
lib/MotorWheel/ MotorWheel.cpp	72
lib/MotorWheel/ MotorWheel.h	72
lib/PID_Beta6/ fuzzy_table.h	76
lib/PID_Beta6/ PID_Beta6.cpp	77
lib/PID_Beta6/ PID_Beta6.h	77
lib/PinChangeInt/ PinChangeInt.cpp	78
lib/PinChangeInt/ PinChangeInt.h	80
lib/PinChangeInt/ PinChangeIntConfig.h	82
src/ main.cpp	92
src/communication/ UARTCom.cpp	82
src/communication/ UCommands.cpp	91
src/communication/ UValue.cpp	92
src/misc/ Utils.cpp	95

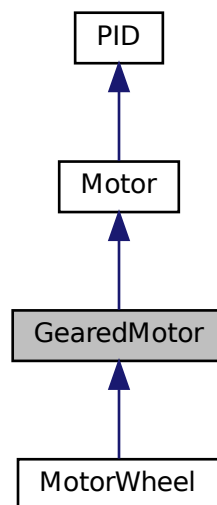
Chapter 4

Class Documentation

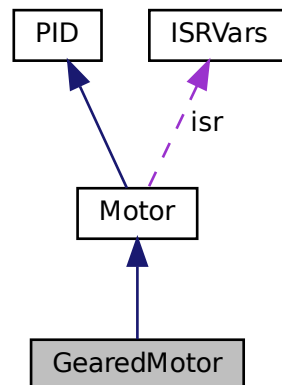
4.1 GearedMotor Class Reference

```
#include <MotorWheel.h>
```

Inheritance diagram for GearedMotor:



Collaboration diagram for GearedMotor:



Public Member Functions

- [GearedMotor](#) (unsigned char _pinPWM, unsigned char _pinDir, unsigned char _pinIRQ, unsigned char _pinIRQB, struct [ISRVars](#) * _isr, unsigned int _ratio=[REDUCTION_RATIO](#))
- float [getGearedSpeedRPM](#) () const
- float [setGearedSpeedRPM](#) (float gearedSpeedRPM, bool dir)
- float [setGearedSpeedRPM](#) (float gearedSpeedRPM)
- unsigned int [getRatio](#) () const
- unsigned int [setRatio](#) (unsigned int ratio=[REDUCTION_RATIO](#))
- float [getPosition](#) ()

Additional Inherited Members

4.1.1 Constructor & Destructor Documentation

4.1.1.1 GearedMotor()

```

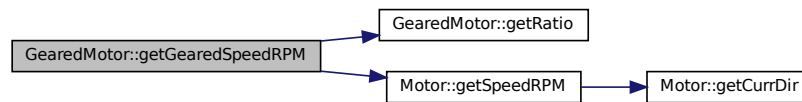
GearedMotor::GearedMotor (
    unsigned char _pinPWM,
    unsigned char _pinDir,
    unsigned char _pinIRQ,
    unsigned char _pinIRQB,
    struct ISRVars * _isr,
    unsigned int _ratio = REDUCTION\_RATIO )
  
```

4.1.2 Member Function Documentation

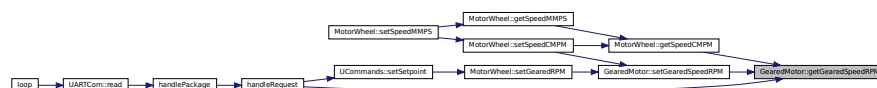
4.1.2.1 getGearedSpeedRPM()

```
float GearedMotor::getGearedSpeedRPM ( ) const
```

Here is the call graph for this function:



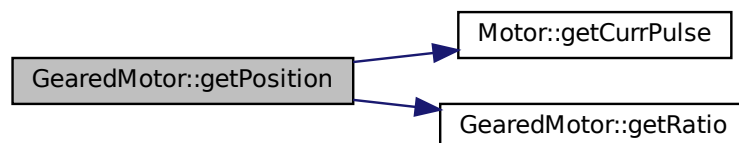
Here is the caller graph for this function:



4.1.2.2 getPosition()

```
float GearedMotor::getPosition ( )
```

Here is the call graph for this function:



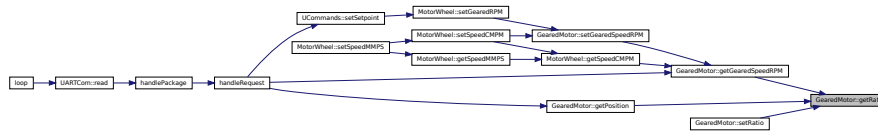
Here is the caller graph for this function:



4.1.2.3 getRatio()

```
unsigned int GearedMotor::getRatio ( ) const
```

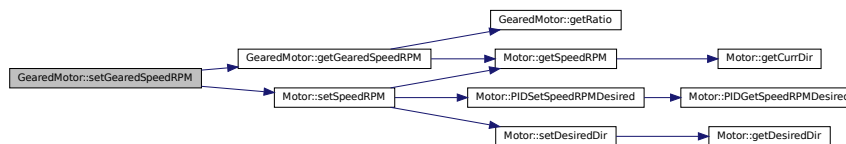
Here is the caller graph for this function:



4.1.2.4 setGearedSpeedRPM() [1/2]

```
float GearedMotor::setGearedSpeedRPM (
    float gearedSpeedRPM )
```

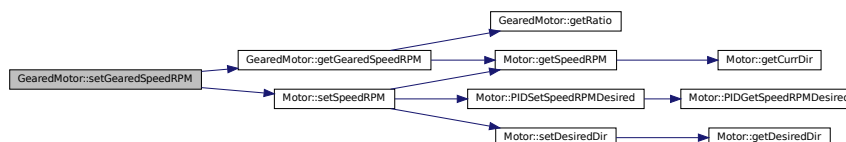
Here is the call graph for this function:



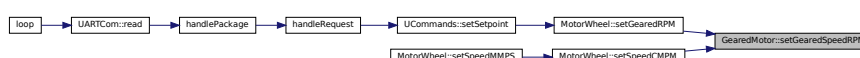
4.1.2.5 setGearedSpeedRPM() [2/2]

```
float GearedMotor::setGearedSpeedRPM (
    float gearedSpeedRPM,
    bool dir )
```

Here is the call graph for this function:



Here is the caller graph for this function:



4.1.2.6 setRatio()

```
unsigned int GearedMotor::setRatio (
    unsigned int ratio = REDUCTION_RATIO )
```

Here is the call graph for this function:



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

- lib/MotorWheel/MotorWheel.h
- lib/MotorWheel/MotorWheel.cpp

4.2 ISRVars Struct Reference

```
#include <MotorWheel.h>
```

Public Attributes

- void(* [ISRfunc](#))()
- volatile long [pulses](#)
- volatile unsigned long [pulseStartMicros](#)
- volatile unsigned long [pulseEndMicros](#)
- volatile unsigned int [speedPPS](#)
- volatile bool [currDirection](#)
- unsigned char [pinIRQB](#)
- unsigned char [pinIRQ](#)

4.2.1 Member Data Documentation

4.2.1.1 currDirection

```
volatile bool ISRVars::currDirection
```

4.2.1.2 ISRfunc

```
void(* ISRVars::ISRfunc) ()
```

4.2.1.3 pinIRQ

```
unsigned char ISRVars::pinIRQ
```

4.2.1.4 pinIRQB

```
unsigned char ISRVars::pinIRQB
```

4.2.1.5 pulseEndMicros

```
volatile unsigned long ISRVars::pulseEndMicros
```

4.2.1.6 pulses

```
volatile long ISRVars::pulses
```

4.2.1.7 pulseStartMicros

```
volatile unsigned long ISRVars::pulseStartMicros
```

4.2.1.8 speedPPS

```
volatile unsigned int ISRVars::speedPPS
```

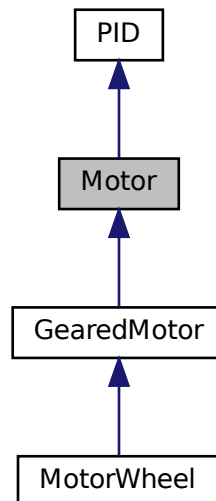
The documentation for this struct was generated from the following file:

- lib/MotorWheel/[MotorWheel.h](#)

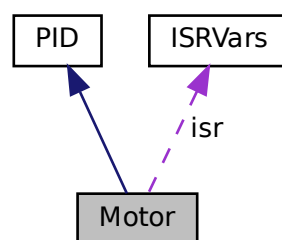
4.3 Motor Class Reference

```
#include <MotorWheel.h>
```

Inheritance diagram for Motor:



Collaboration diagram for Motor:



Public Member Functions

- [Motor](#) (unsigned char _pinPWM, unsigned char _pinDir, unsigned char _pinIRQ, unsigned char _pinIRQB, struct [ISRVars](#) *_isr)
- void [setupInterrupt](#) ()
- unsigned char [getPinPWM](#) () const
- unsigned char [getPinDir](#) () const

- unsigned char [getPinIRQ](#) () const
- unsigned char [getPinIRQB](#) () const
- unsigned int [runPWM](#) (unsigned int PWM, bool dir, bool saveDir=true)
- unsigned int [getPWM](#) () const
- unsigned int [advancePWM](#) (unsigned int PWM)
- unsigned int [backoffPWM](#) (unsigned int PWM)
- bool [setDesiredDir](#) (bool dir)
- bool [getDesiredDir](#) () const
- bool [reverseDesiredDir](#) ()
- bool [setCurrDir](#) ()
- bool [getCurrDir](#) () const
- int [getSpeedRPM](#) () const
- unsigned int [setSpeedRPM](#) (int speedRPM, bool dir)
- int [setSpeedRPM](#) (int speedRPM)
- bool [PIDSetup](#) (float kc=[KC](#), float tau=[TAUI](#), float taud=[TAUD](#), unsigned int sampleTime=1000)
- bool [PIDGetStatus](#) () const
- bool [PIDEnable](#) (float kc=[KC](#), float tau=[TAUI](#), float taud=[TAUD](#), unsigned int sampleTime=1000)
- bool [PIDDisable](#) ()
- bool [PIDReset](#) ()
- bool [PIDRegulate](#) (bool doRegulate=true)
- unsigned int [PIDSetSpeedRPMDesired](#) (unsigned int speedRPM)
- unsigned int [PIDGetSpeedRPMDesired](#) () const
- void [delayMS](#) (unsigned int ms, bool [debug](#)=false)
- void [debugger](#) () const
- int [getSpeedPPS](#) () const
- long [getCurrPulse](#) () const
- long [setCurrPulse](#) (long _pulse)
- long [resetCurrPulse](#) ()

Public Attributes

- struct [ISRVars](#) * [isr](#)

4.3.1 Constructor & Destructor Documentation

4.3.1.1 Motor()

```
Motor::Motor (
    unsigned char _pinPWM,
    unsigned char _pinDir,
    unsigned char _pinIRQ,
    unsigned char _pinIRQB,
    struct ISRVars * _isr )
```


Here is the call graph for this function:



Here is the caller graph for this function:



4.3.2 Member Function Documentation

4.3.2.1 advancePWM()

```
unsigned int Motor::advancePWM (  
    unsigned int PWM )
```

Here is the call graph for this function:



4.3.2.2 backoffPWM()

```
unsigned int Motor::backoffPWM (
    unsigned int PWM )
```

Here is the call graph for this function:



4.3.2.3 debugger()

```
void Motor::debugger ( ) const
```

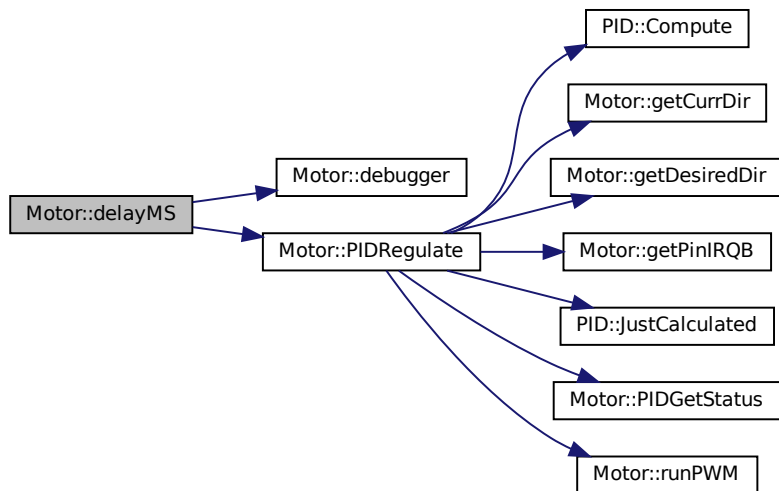
Here is the caller graph for this function:



4.3.2.4 delayMS()

```
void Motor::delayMS (
    unsigned int ms,
    bool debug = false )
```

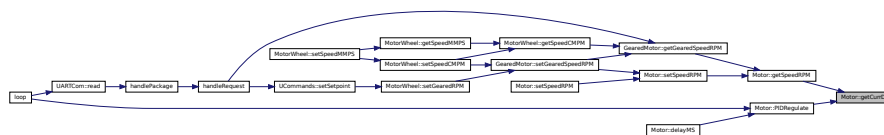
Here is the call graph for this function:



4.3.2.5 getCurrDir()

```
bool Motor::getCurrDir ( ) const
```

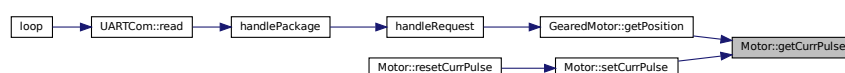
Here is the caller graph for this function:



4.3.2.6 getCurrPulse()

```
long Motor::getCurrPulse ( ) const
```

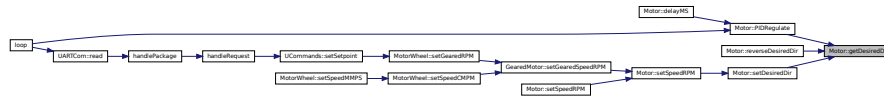
Here is the caller graph for this function:



4.3.2.7 getDesiredDir()

```
bool Motor::getDesiredDir ( ) const
```

Here is the caller graph for this function:



4.3.2.8 getPinDir()

```
unsigned char Motor::getPinDir ( ) const
```

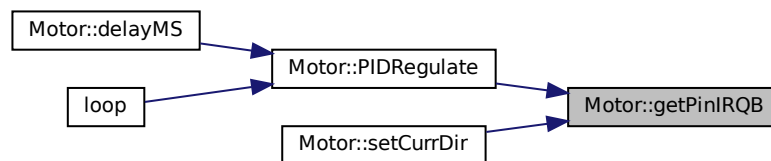
4.3.2.9 getPinIRQ()

```
unsigned char Motor::getPinIRQ ( ) const
```

4.3.2.10 getPinIRQB()

```
unsigned char Motor::getPinIRQB ( ) const
```

Here is the caller graph for this function:



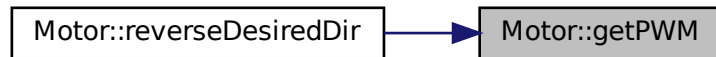
4.3.2.11 getPinPWM()

```
unsigned char Motor::getPinPWM ( ) const
```

4.3.2.12 getPWM()

```
unsigned int Motor::getPWM ( ) const
```

Here is the caller graph for this function:



4.3.2.13 getSpeedPPS()

```
int Motor::getSpeedPPS ( ) const
```

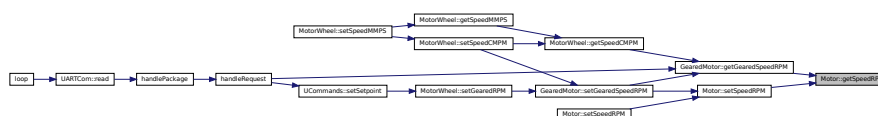
4.3.2.14 getSpeedRPM()

```
int Motor::getSpeedRPM ( ) const
```

Here is the call graph for this function:



Here is the caller graph for this function:



4.3.2.15 PIDDisable()

```
bool Motor::PIDDisable ( )
```

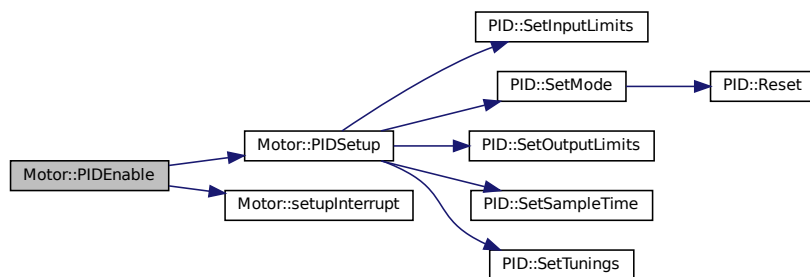
Here is the caller graph for this function:



4.3.2.16 PIDEnable()

```
bool Motor::PIDEnable (
    float kc = KC,
    float tauI = TAUI,
    float tauD = TAUD,
    unsigned int sampleTime = 1000 )
```

Here is the call graph for this function:



Here is the caller graph for this function:



4.3.2.17 PIDGetSpeedRPMDesired()

```
unsigned int Motor::PIDGetSpeedRPMDesired ( ) const
```

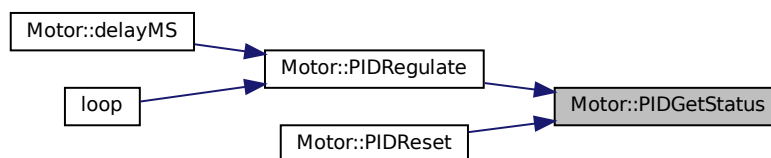
Here is the caller graph for this function:



4.3.2.18 PIDGetStatus()

```
bool Motor::PIDGetStatus ( ) const
```

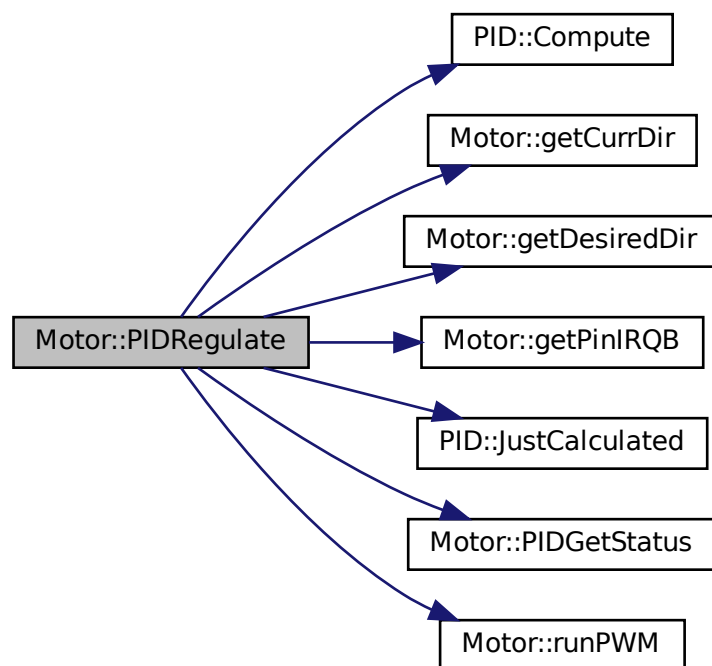
Here is the caller graph for this function:



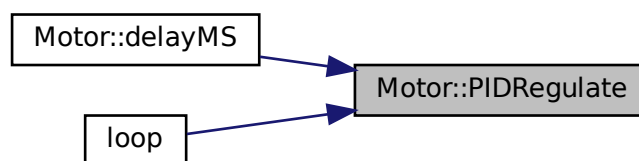
4.3.2.19 PIDRegulate()

```
bool Motor::PIDRegulate (
    bool doRegulate = true )
```

Here is the call graph for this function:



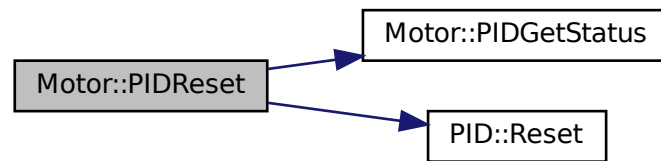
Here is the caller graph for this function:



4.3.2.20 PIDReset()

```
bool Motor::PIDReset ( )
```


Here is the call graph for this function:



4.3.2.21 PIDSetSpeedRPMDesired()

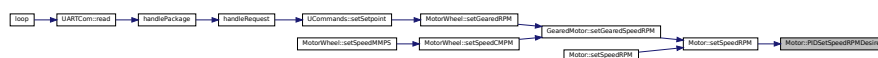
```

unsigned int Motor::PIDSetSpeedRPMDesired (
    unsigned int speedRPM )
  
```

Here is the call graph for this function:



Here is the caller graph for this function:

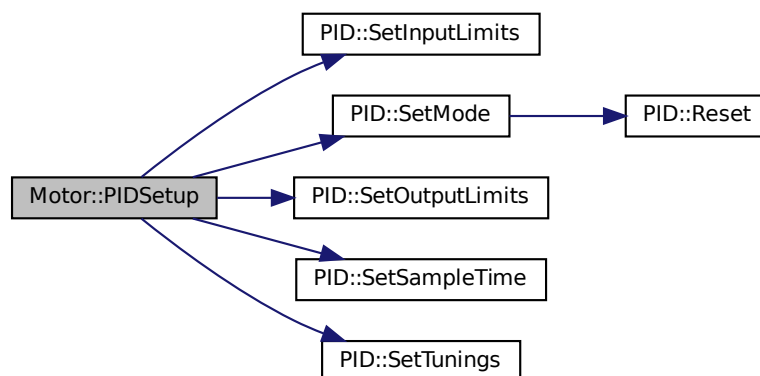


4.3.2.22 PIDSetup()

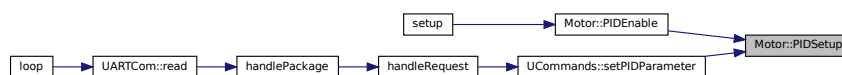
```

bool Motor::PIDSetup (
    float kc = KC,
    float tauI = TAUI,
    float tauD = TAUD,
    unsigned int sampleTime = 1000 )
  
```

Here is the call graph for this function:



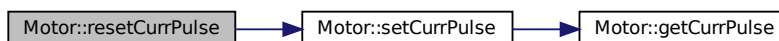
Here is the caller graph for this function:



4.3.2.23 resetCurrPulse()

```
long Motor::resetCurrPulse ( )
```

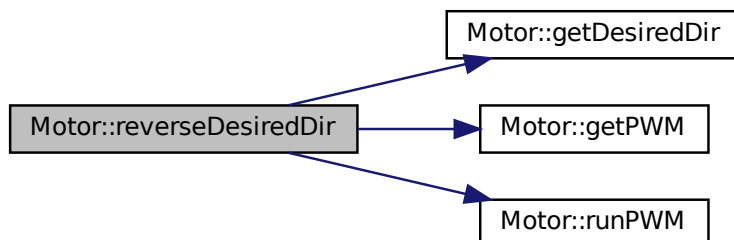
Here is the call graph for this function:



4.3.2.24 reverseDesiredDir()

```
bool Motor::reverseDesiredDir ( )
```

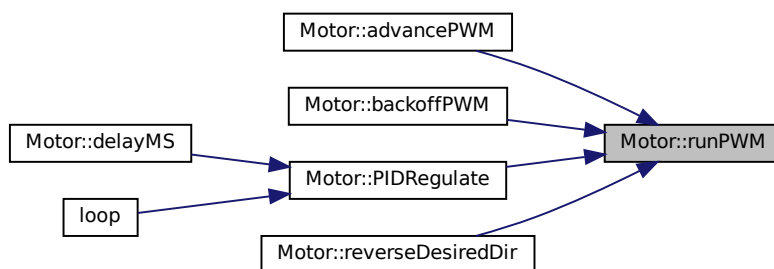
Here is the call graph for this function:



4.3.2.25 runPWM()

```
unsigned int Motor::runPWM (
    unsigned int PWM,
    bool dir,
    bool saveDir = true )
```

Here is the caller graph for this function:



4.3.2.26 setCurrDir()

```
bool Motor::setCurrDir ( )
```

Here is the call graph for this function:



4.3.2.27 setCurrPulse()

```
long Motor::setCurrPulse (
    long _pulse )
```

Here is the call graph for this function:



Here is the caller graph for this function:



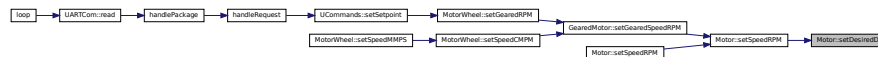
4.3.2.28 setDesiredDir()

```
bool Motor::setDesiredDir (
    bool dir )
```

Here is the call graph for this function:



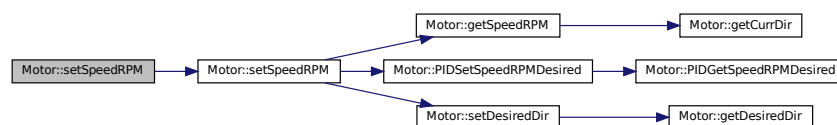
Here is the caller graph for this function:



4.3.2.29 setSpeedRPM() [1/2]

```
int Motor::setSpeedRPM (
    int speedRPM )
```

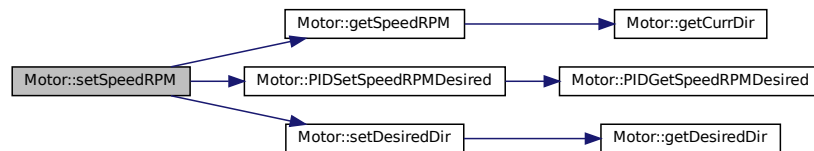
Here is the call graph for this function:



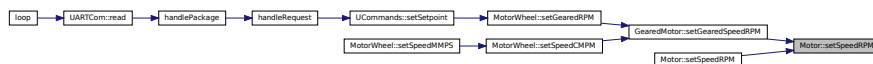
4.3.2.30 setSpeedRPM() [2/2]

```
unsigned int Motor::setSpeedRPM (
    int speedRPM,
    bool dir )
```

Here is the call graph for this function:



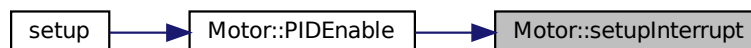
Here is the caller graph for this function:



4.3.2.31 setupInterrupt()

```
void Motor::setupInterrupt ( )
```

Here is the caller graph for this function:



4.3.3 Member Data Documentation

4.3.3.1 isr

```
struct ISRVars* Motor::isr
```

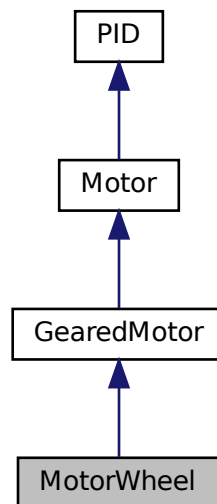
The documentation for this class was generated from the following files:

- lib/MotorWheel/MotorWheel.h
- lib/MotorWheel/MotorWheel.cpp

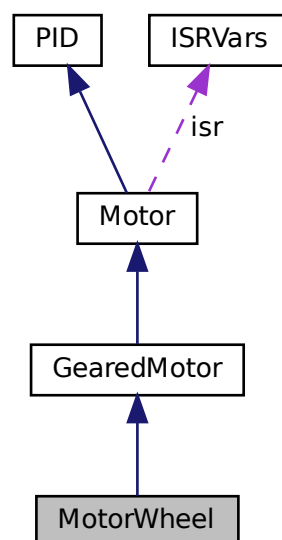
4.4 MotorWheel Class Reference

```
#include <MotorWheel.h>
```

Inheritance diagram for MotorWheel:



Collaboration diagram for MotorWheel:



Public Member Functions

- [MotorWheel](#) (unsigned char _pinPWM, unsigned char _pinDir, unsigned char _pinIRQ, unsigned char _pinIRQB, struct [ISRVars](#) *_isr, unsigned int ratio=[REDUCTION_RATIO](#), unsigned int cirMM=[CIRMM](#))
- unsigned int [getCirMM](#) () const
- unsigned int [setCirMM](#) (unsigned int cirMM=[CIRMM](#))
- int [getSpeedCMPM](#) () const
- int [setSpeedCMPM](#) (unsigned int cm, bool dir)
- int [setSpeedCMPM](#) (int cm)
- int [getSpeedMMPS](#) () const
- int [setSpeedMMPS](#) (unsigned int mm, bool dir)
- int [setSpeedMMPS](#) (int mm)
- void [setGearedRPM](#) (float rpm, bool dir)

Additional Inherited Members

4.4.1 Constructor & Destructor Documentation

4.4.1.1 MotorWheel()

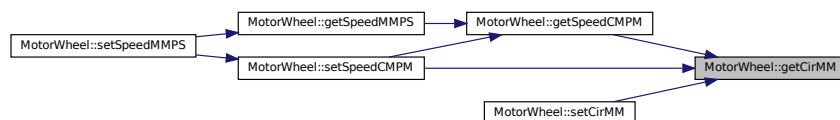
```
MotorWheel::MotorWheel (
    unsigned char _pinPWM,
    unsigned char _pinDir,
    unsigned char _pinIRQ,
    unsigned char _pinIRQB,
    struct ISRVars *_isr,
    unsigned int ratio = REDUCTION\_RATIO,
    unsigned int cirMM = CIRMM )
```

4.4.2 Member Function Documentation

4.4.2.1 getCirMM()

```
unsigned int MotorWheel::getCirMM ( ) const
```

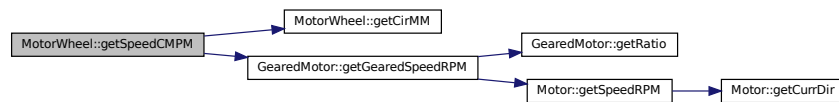
Here is the caller graph for this function:



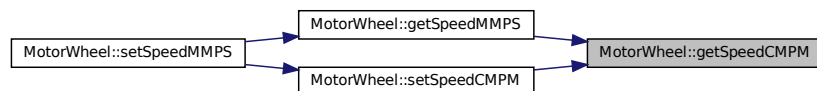
4.4.2.2 getSpeedCMPM()

```
int MotorWheel::getSpeedCMPM ( ) const
```

Here is the call graph for this function:



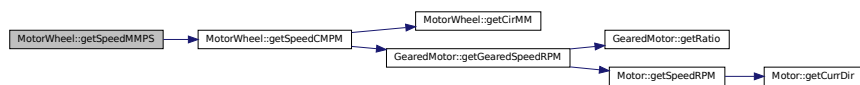
Here is the caller graph for this function:



4.4.2.3 getSpeedMMPS()

```
int MotorWheel::getSpeedMMPS ( ) const
```

Here is the call graph for this function:



Here is the caller graph for this function:



4.4.2.4 setCirMM()

```
unsigned int MotorWheel::setCirMM (
    unsigned int cirMM = CIRMM )
```

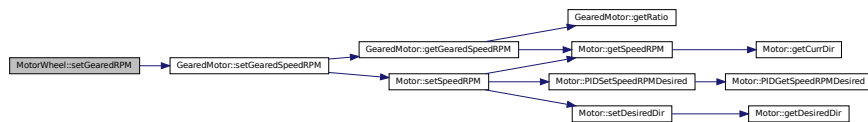
Here is the call graph for this function:



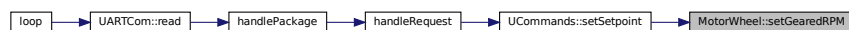
4.4.2.5 setGearedRPM()

```
void MotorWheel::setGearedRPM (
    float rpm,
    bool dir )
```

Here is the call graph for this function:



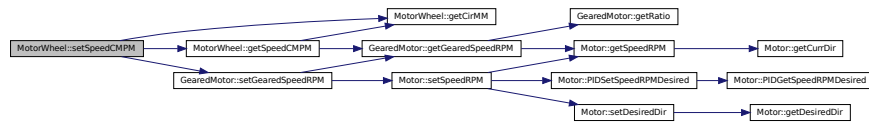
Here is the caller graph for this function:



4.4.2.6 setSpeedCMPM() [1/2]

```
int MotorWheel::setSpeedCMPM (
    int cm )
```

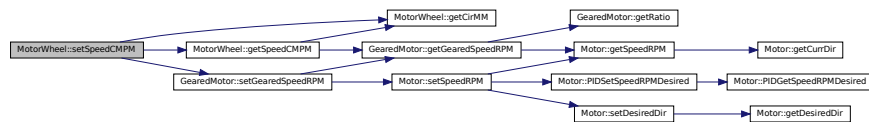
Here is the call graph for this function:



4.4.2.7 setSpeedCMPM() [2/2]

```
int MotorWheel::setSpeedCMPM (
    unsigned int cm,
    bool dir )
```

Here is the call graph for this function:



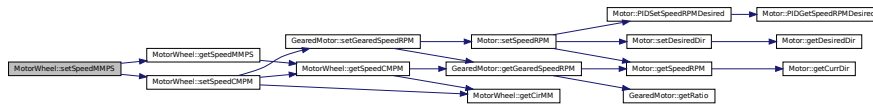
Here is the caller graph for this function:



4.4.2.8 setSpeedMMPS() [1/2]

```
int MotorWheel::setSpeedMMPS (
    int mm )
```

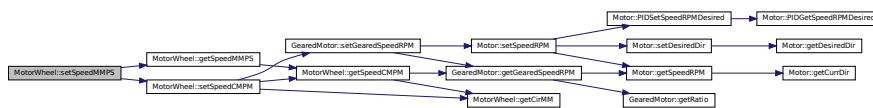
Here is the call graph for this function:



4.4.2.9 setSpeedMMPS() [2/2]

```
int MotorWheel::setSpeedMMPS (
    unsigned int mm,
    bool dir )
```

Here is the call graph for this function:



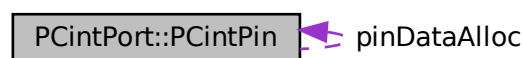
The documentation for this class was generated from the following files:

- lib/MotorWheel/[MotorWheel.h](#)
- lib/MotorWheel/[MotorWheel.cpp](#)

4.5 PCintPort::PCintPin Class Reference

```
#include <PinChangeInt.h>
```

Collaboration diagram for PCintPort::PCintPin:



Public Member Functions

- [PCIntPin\(\)](#)

Public Attributes

- [PCIntVoidFuncPtr](#) [PCIntFunc](#)
- [uint8_t](#) [PCIntMode](#)
- [uint8_t](#) [PCIntMask](#)

Static Public Attributes

- static [PCIntPin](#) [pinDataAlloc](#) [[MAX_PIN_CHANGE_PINS](#)]

4.5.1 Constructor & Destructor Documentation

4.5.1.1 PCIntPin()

```
PCIntPort::PCIntPin::PCIntPin ( ) [inline]
```

4.5.2 Member Data Documentation

4.5.2.1 PCIntFunc

```
PCIntVoidFuncPtr PCIntPort::PCIntPin::PCIntFunc
```

4.5.2.2 PCIntMask

```
uint8\_t PCIntPort::PCIntPin::PCIntMask
```

4.5.2.3 PCIntMode

```
uint8\_t PCIntPort::PCIntPin::PCIntMode
```

4.5.2.4 pinDataAlloc

```
PCintPort::PCintPin PCintPort::PCintPin::pinDataAlloc [static]
```

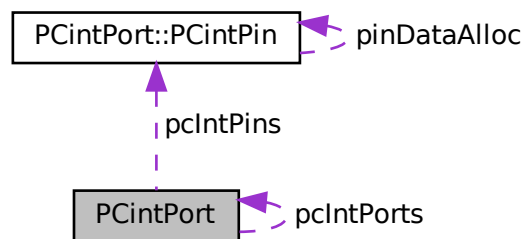
The documentation for this class was generated from the following files:

- lib/PinChangeInt/[PinChangeInt.h](#)
- lib/PinChangeInt/[PinChangeInt.cpp](#)

4.6 PCintPort Class Reference

```
#include <PinChangeInt.h>
```

Collaboration diagram for PCintPort:



Classes

- class [PCintPin](#)

Public Member Functions

- [INLINE_PCINT](#) void [PCint](#) ()

Static Public Member Functions

- static void [attachInterrupt](#) (uint8_t pin, [PCintvoidFuncPtr](#) userFunc, int mode)
- static void [detachInterrupt](#) (uint8_t pin)

Static Public Attributes

- static [PCintPort](#) [pcIntPorts](#) []

Protected Member Functions

- void [addPin](#) (uint8_t mode, uint8_t mask, [PCIntvoidFuncPtr](#) userFunc)
- void [delPin](#) (uint8_t mask)

Protected Attributes

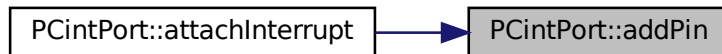
- volatile uint8_t & [portInputReg](#)
- volatile uint8_t & [pcmask](#)
- const uint8_t [PCICRbit](#)
- uint8_t [PCintLast](#)
- [PCIntPin](#) * [pcIntPins](#) [9]

4.6.1 Member Function Documentation

4.6.1.1 addPin()

```
void PCIntPort::addPin (
    uint8_t mode,
    uint8_t mask,
    PCIntvoidFuncPtr userFunc ) [protected]
```

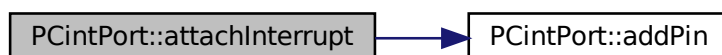
Here is the caller graph for this function:



4.6.1.2 attachInterrupt()

```
void PCIntPort::attachInterrupt (
    uint8_t pin,
    PCIntvoidFuncPtr userFunc,
    int mode ) [static]
```

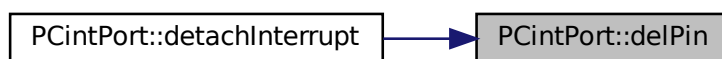
Here is the call graph for this function:



4.6.1.3 delPin()

```
void PCintPort::delPin (
    uint8_t mask ) [protected]
```

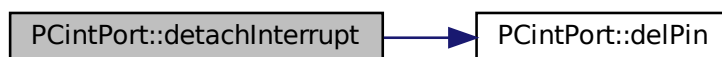
Here is the caller graph for this function:



4.6.1.4 detachInterrupt()

```
void PCintPort::detachInterrupt (
    uint8_t pin ) [static]
```

Here is the call graph for this function:



4.6.1.5 PCint()

```
void PCintPort::PCint ( )
```

Here is the caller graph for this function:



4.6.2 Member Data Documentation

4.6.2.1 PCICRbit

```
const uint8_t PCIntPort::PCICRbit [protected]
```

4.6.2.2 PCintLast

```
uint8_t PCIntPort::PCintLast [protected]
```

4.6.2.3 pcIntPins

```
PCintPin* PCIntPort::pcIntPins[9] [protected]
```

4.6.2.4 pcIntPorts

```
PCintPort PCIntPort::pcIntPorts [static]
```

Initial value:

```
= {  
    PCintPort(0, PCMSK0),  
    PCintPort(1, PCMSK1),  
    PCintPort(2, PCMSK2)  
}
```

4.6.2.5 pcmask

```
volatile uint8_t& PCIntPort::pcmask [protected]
```

4.6.2.6 portInputReg

```
volatile uint8_t& PCIntPort::portInputReg [protected]
```

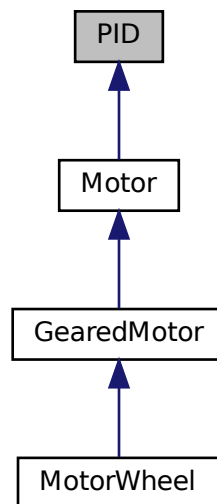
The documentation for this class was generated from the following files:

- lib/PinChangeInt/[PinChangeInt.h](#)
- lib/PinChangeInt/[PinChangeInt.cpp](#)

4.7 PID Class Reference

```
#include <PID_Beta6.h>
```

Inheritance diagram for PID:



Public Member Functions

- [PID](#) (int *, int *, int *, float, float, float)
- [PID](#) (int *, int *, int *, int *, float, float, float)
- void [SetMode](#) (int Mode)
- void [Compute](#) ()
- void [SetInputLimits](#) (int, int)
- void [SetOutputLimits](#) (int, int)
- void [SetTunings](#) (float, float, float)
- void [SetSampleTime](#) (int)
- void [Reset](#) ()
- bool [JustCalculated](#) ()
- int [GetMode](#) ()
- int [GetINMin](#) ()
- int [GetINMax](#) ()
- int [GetOUTMin](#) ()
- int [GetOUTMax](#) ()
- int [GetSampleTime](#) ()
- float [GetP_Param](#) ()
- float [GetI_Param](#) ()
- float [GetD_Param](#) ()

4.7.1 Constructor & Destructor Documentation

4.7.1.1 PID() [1/2]

```
PID::PID (
    int * Input,
    int * Output,
    int * Setpoint,
    float Kc,
    float TauI,
    float TauD )
```

Here is the call graph for this function:



4.7.1.2 PID() [2/2]

```
PID::PID (
    int * Input,
    int * Output,
    int * Setpoint,
    int * FFBias,
    float Kc,
    float TauI,
    float TauD )
```

Here is the call graph for this function:

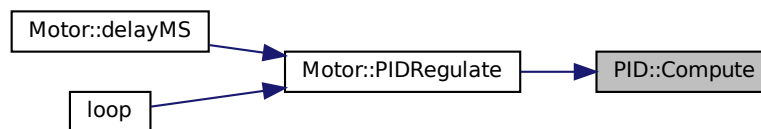


4.7.2 Member Function Documentation

4.7.2.1 Compute()

```
void PID::Compute ( )
```

Here is the caller graph for this function:



4.7.2.2 GetD_Param()

```
float PID::GetD_Param ( )
```

4.7.2.3 GetI_Param()

```
float PID::GetI_Param ( )
```

4.7.2.4 GetINMax()

```
int PID::GetINMax ( )
```

4.7.2.5 GetINMin()

```
int PID::GetINMin ( )
```

4.7.2.6 GetMode()

```
int PID::GetMode ( )
```

4.7.2.7 GetOUTMax()

```
int PID::GetOUTMax ( )
```

4.7.2.8 GetOUTMin()

```
int PID::GetOUTMin ( )
```

4.7.2.9 GetP_Param()

```
float PID::GetP_Param ( )
```

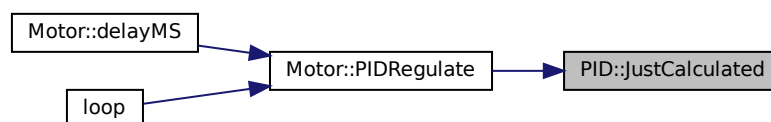
4.7.2.10 GetSampleTime()

```
int PID::GetSampleTime ( )
```

4.7.2.11 JustCalculated()

```
bool PID::JustCalculated ( )
```

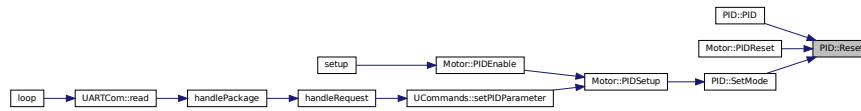
Here is the caller graph for this function:



4.7.2.12 Reset()

```
void PID::Reset ( )
```

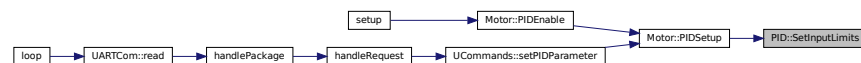
Here is the caller graph for this function:



4.7.2.13 SetInputLimits()

```
void PID::SetInputLimits (
    int INMin,
    int INMax )
```

Here is the caller graph for this function:



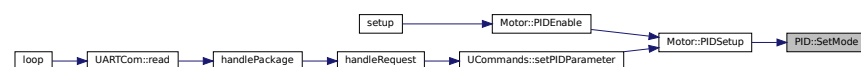
4.7.2.14 SetMode()

```
void PID::SetMode (
    int Mode )
```

Here is the call graph for this function:



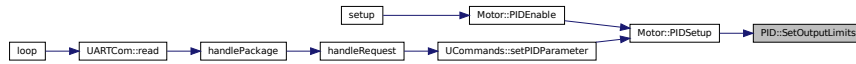
Here is the caller graph for this function:



4.7.2.15 SetOutputLimits()

```
void PID::SetOutputLimits (
    int OUTMin,
    int OUTMax )
```

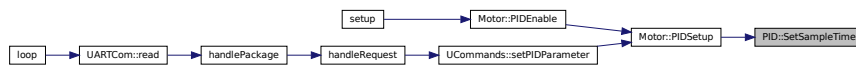
Here is the caller graph for this function:



4.7.2.16 SetSampleTime()

```
void PID::SetSampleTime (
    int NewSampleTime )
```

Here is the caller graph for this function:



4.7.2.17 SetTunings()

```
void PID::SetTunings (
    float Kc,
    float TauI,
    float TauD )
```

Here is the caller graph for this function:



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

- lib/PID_Beta6/PID_Beta6.h
- lib/PID_Beta6/PID_Beta6.cpp

4.8 UARTCom Class Reference

```
#include <UARTCom.hpp>
```

Static Public Member Functions

- static void [init](#) (uint8_t *d)
initializes uart communication
- static void [read](#) (uint8_t *d)
reads uart interface and stores the data into an uint8_t array
- static void [send](#) (U_FrameType ft, uint8_t pll, [U_Component](#) c, uint8_t *pl)
assembles and sends a uart package

Static Public Attributes

- static uint8_t [data](#) [[MAX_PACKET_LENGTH](#)]

4.8.1 Member Function Documentation

4.8.1.1 [init\(\)](#)

```
void UARTCom::init (
    uint8_t * d ) [static]
```

initializes uart communication

initializes UART communication

Parameters

<i>d</i>	data package to be initialized
<i>d</i>	data package

Here is the call graph for this function:



Here is the caller graph for this function:



4.8.1.2 read()

```
void UARTCom::read (
    uint8_t * d ) [static]
```

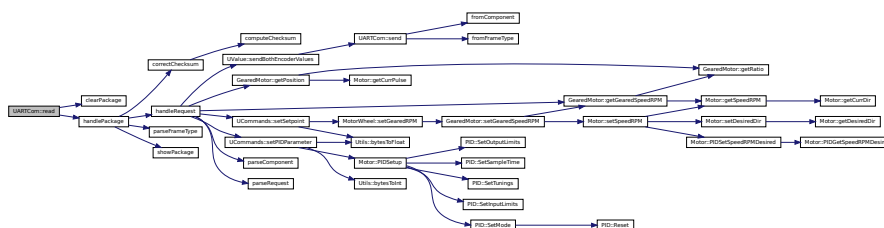
reads uart interface and stores the data into an uint8_t array

```
reads incoming bytes from serial port
```

Parameters

d	array to store the data into
d	data package

Here is the call graph for this function:



Here is the caller graph for this function:



4.8.1.3 send()

```
void UARTCom::send (
    U_FrameType ft,
    uint8_t pll,
    U_Component cp,
    uint8_t * pl ) [static]
```

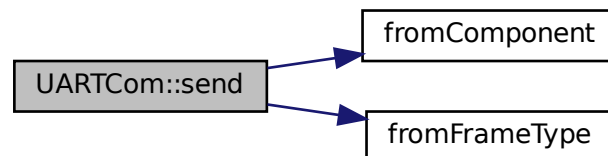
assembles and sends a uart package

sends data package

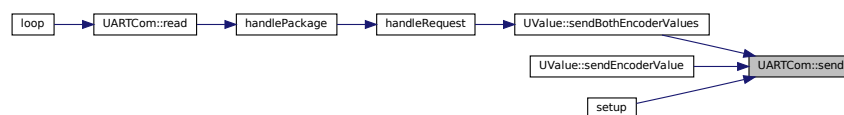
Parameters

<i>ft</i>	frametype of message
<i>pll</i>	payload length
<i>c</i>	component
<i>pl</i>	payload
<i>ft</i>	frametype
<i>pll</i>	payload length
<i>cp</i>	component
<i>pl</i>	payload

Here is the call graph for this function:



Here is the caller graph for this function:



4.8.2 Member Data Documentation

4.8.2.1 data

```
uint8_t UARTCom::data [static]
```

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

- include/communication/[UARTCom.hpp](#)
- src/communication/[UARTCom.cpp](#)

4.9 UCommands Class Reference

```
#include <UCommands.hpp>
```

Static Public Member Functions

- static void [setSetpoint](#) (uint8_t *d, [U_Component](#) c)
sets the desired velocity of a motor
- static void [setPIDParameter](#) (uint8_t *d, [U_Component](#) c)
sets the [PID](#) parameters

4.9.1 Member Function Documentation

4.9.1.1 setPIDParameter()

```
void UCommands::setPIDParameter (
    uint8_t * d,
    U\_Component c ) [static]
```

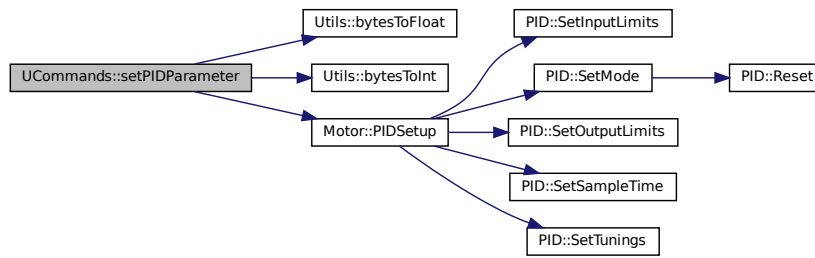
sets the [PID](#) parameters

handles incoming requests for [PID](#) parameters

Parameters

<i>d</i>	uart data package to read the pid parameters from
<i>c</i>	motor to apply the value to
<i>d</i>	payload
<i>c</i>	component

Here is the call graph for this function:



Here is the caller graph for this function:



4.9.1.2 setSetpoint()

```

void UCommands::setSetpoint (
    uint8_t * d,
    U_Component c ) [static]
  
```

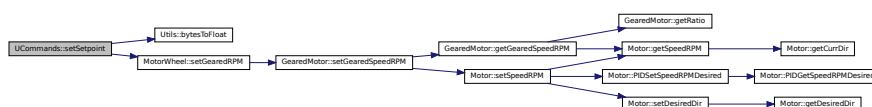
sets the desired velocity of a motor

handles incoming requests for motor speed

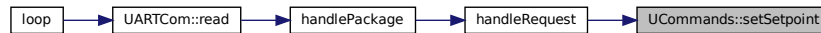
Parameters

<i>d</i>	uart data package to read the velocity from
<i>c</i>	motor to apply the value to
<i>d</i>	payload
<i>c</i>	component

Here is the call graph for this function:



Here is the caller graph for this function:



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

- include/communication/[UCommands.hpp](#)
- src/communication/[UCommands.cpp](#)

4.10 Utils Class Reference

```
#include <Utils.hpp>
```

Static Public Member Functions

- static float [bytesToFloat](#) (uint8_t b0, uint8_t b1, uint8_t b2, uint8_t b3)
converts 4 bytes to float (big endian)
- static uint32_t [bytesToInt](#) (uint8_t b0, uint8_t b1, uint8_t b2, uint8_t b3)
converts 4 bytes to uint32_t (big endian)

4.10.1 Member Function Documentation

4.10.1.1 bytesToFloat()

```
float Utils::bytesToFloat (
    uint8_t b0,
    uint8_t b1,
    uint8_t b2,
    uint8_t b3 ) [static]
```

converts 4 bytes to float (big endian)

converts a float to a byte array

Parameters

<i>b0</i>	first byte
<i>b1</i>	second byte
<i>b2</i>	third byte
<i>b3</i>	fourth byte

Returns

float

Parameters

<i>b0</i>	
<i>b1</i>	
<i>b2</i>	
<i>b3</i>	

Returns

float

Here is the caller graph for this function:

**4.10.1.2 bytesToInt()**

```

uint32_t Utils::bytesToInt (
    uint8_t b0,
    uint8_t b1,
    uint8_t b2,
    uint8_t b3 ) [static]

```

converts 4 bytes to uint32_t (big endian)

converts a byte array to a float

Parameters

<i>b0</i>	first byte
<i>b1</i>	second byte
<i>b2</i>	third byte
<i>b3</i>	fourth byte

Returns

uint32_t

Parameters

<i>b0</i>	
<i>b1</i>	
<i>b2</i>	
<i>b3</i>	

Returns

uint32_t

Here is the caller graph for this function:



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

- include/misc/[Utils.hpp](#)
- src/misc/[Utils.cpp](#)

4.11 UValue Class Reference

```
#include <UValue.hpp>
```

Static Public Member Functions

- static void [sendEncoderValue](#) (U_Component cp, float value)
sends encoder value via uart
- static void [sendBothEncoderValues](#) (float speed_right, float speed_left, float pos_right, float pos_left)
send both encoder values via uart

4.11.1 Member Function Documentation

4.11.1.1 sendBothEncoderValues()

```
void UValue::sendBothEncoderValues (
    float speed_right,
    float speed_left,
    float pos_right,
    float pos_left ) [static]
```

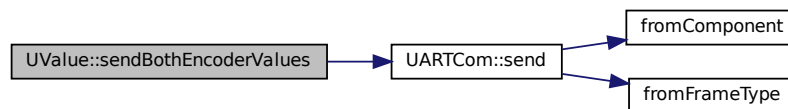
send both encoder values via uart

sends the current speed of both motors

Parameters

<i>value_left</i>	encoder value left
<i>value_right</i>	encoder value right
<i>speed_right</i>	speed of right motor
<i>speed_left</i>	speed of left motor
<i>pos_right</i>	current integrated position of right rotor
<i>pos_left</i>	current integrated position of left rotor

Here is the call graph for this function:



Here is the caller graph for this function:



4.11.1.2 sendEncoderValue()

```

void UValue::sendEncoderValue (
    U_Component cp,
    float value ) [static]
  
```

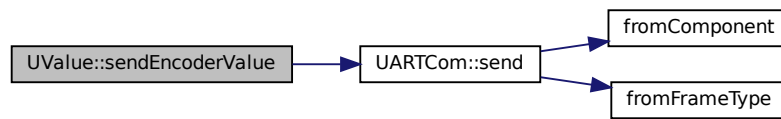
sends encoder value via uart

sends the current speed of both motors

Parameters

<i>cp</i>	component to which the value refers to
<i>value</i>	encoder value
<i>cp</i>	component
<i>value</i>	speed of motor

Here is the call graph for this function:



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

- [include/communication/UValue.hpp](#)
- [src/communication/UValue.cpp](#)

Chapter 5

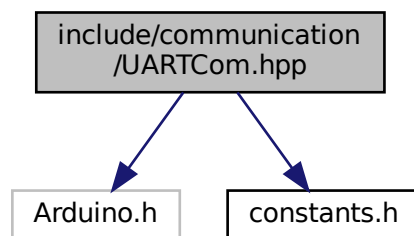
File Documentation

5.1 include/communication/UARTCom.hpp File Reference

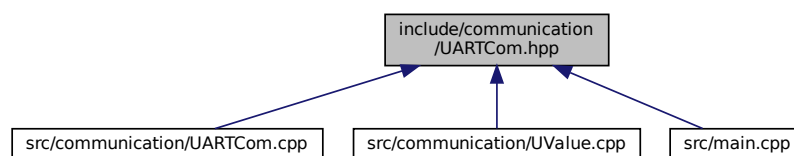
class to manage uart communication

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

Include dependency graph for UARTCom.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [UARTCom](#)

5.1.1 Detailed Description

class to manage uart communication

Author

your name (you@domain.com)

Version

0.1

Date

2021-07-22

Copyright

Copyright (c) 2021

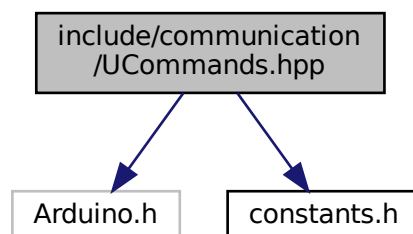
5.2 include/communication/UCommands.hpp File Reference

implementation of commands according to uart protocol

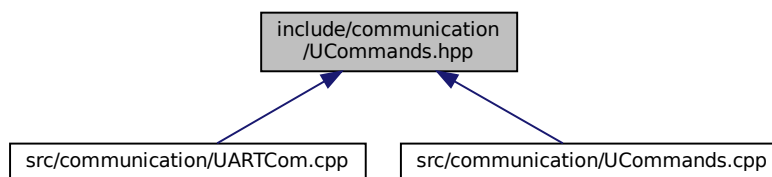
```
#include "Arduino.h"
```

```
#include "constants.h"
```

Include dependency graph for UCommands.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [UCommands](#)

5.2.1 Detailed Description

implementation of commands according to uart protocol

Author

your name ([you@domain.com](#))

Version

0.1

Date

2021-07-22

Copyright

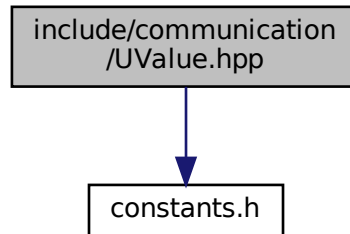
Copyright (c) 2021

5.3 include/communication/UValue.hpp File Reference

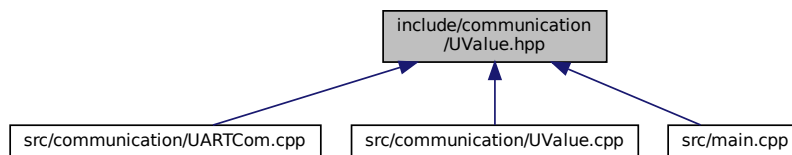
class to send out encoder values

```
#include "constants.h"
```

Include dependency graph for UValue.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [UValue](#)

5.3.1 Detailed Description

class to send out encoder values

Author

your name (you@domain.com)

Version

0.1

Date

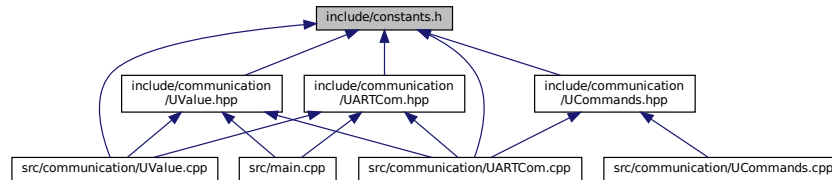
2021-07-22

Copyright

Copyright (c) 2021

5.4 include/constants.h File Reference

This graph shows which files directly or indirectly include this file:



Macros

- #define `NUM_MOTORS` 2
- all constants and custom data types*
- #define `M1_EN_PIN` 10
- #define `M1_DIR_PIN` 11
- #define `M2_EN_PIN` 9
- #define `M2_DIR_PIN` 8
- #define `SPEED_THRESHOLD` 8
- #define `ENC_1_A` 6
- #define `ENC_1_B` 7
- #define `ENC_2_A` 5
- #define `ENC_2_B` 4
- #define `COUNT_PER_ROTATION` 3533
- #define `CHAIN_LENGTH_M` 0.741
- #define `VEL_UPDATE_RATE_MS` 20
- #define `MAX_SPEED` 0.6
- #define `KP_1` 10
- #define `KD_1` 0.0
- #define `KI_1` 0.0
- #define `KP_2` 10
- #define `KD_2` 0.0
- #define `KI_2` 0.0
- #define `VEL2TORQUE_RATIO` 1.0
- #define `P_GAIN` 1.0
- #define `PRINT_RATE_MS` 500
- #define `UART_BAUD_RATE` 57600
- #define `MAX_PACKET_LENGTH` 50
- #define `PACKAGE_LENGTH_NO_P` 10
- #define `RX_328_PIN` 6
- #define `TX_328_PIN` 7
- #define `UART_TRANSMIT_MS` 10
- #define `PACKAGE_PAUSE_MS` 10
- #define `START_BYTE_POS` 0
- #define `FRAME_TYPE_POS` 1
- #define `PAYLOAD_LENGTH_POS` 2
- #define `COMPONENT_POS` 3
- #define `TIMESTAMP_POS` 4
- #define `COMMAND_POS` 8

- `#define PAYLOAD_POS 8`
- `#define FRAME_TYPE_STARTBIT 7`
- `#define FRAME_TYPE_ENDBIT 5`
- `#define COMPONENT_STARTBIT 7`
- `#define COMPONENT_ENDBIT 4`
- `#define START_BYTE 0xAA`
- `#define REQUEST_DRIVE_MOTOR 0x01`
- `#define REQUEST_RESET_TIMESTAMP 0x02`
- `#define FT_VALUE_BV 0`
- `#define FT_REQUEST_BV 32`
- `#define FT_RESPONSE_BV 64`
- `#define CP_DEFAULT_BV 0`
- `#define CP_LEFT_M_BV 16`
- `#define CP_RIGHT_M_BV 32`
- `#define CP_ALL_M_BV 48`
- `#define LAST(k, n) ((k) & ((1<<(n))-1))`
- `#define MID(k, m, n) LAST((k)>>(m),((n)-(m)))`

Typedefs

- `typedef enum frametypes U_FrameType`
- `typedef enum components U_Component`
- `typedef enum requests U_Request`

Enumerations

- `enum frametypes { VALUE , REQUEST , RESPONSE }`
- `enum components { DEFAULT_COMPONENT , LEFT_MOTOR , RIGHT_MOTOR , ALL_MOTORS }`
- `enum requests {
 DEFAULT_REQUEST , DRIVE_MOTOR , RESET_TIMESTAMP , DRIVE_ENCODER ,
 PID_PARAMETER }`

5.4.1 Macro Definition Documentation

5.4.1.1 CHAIN_LENGTH_M

```
#define CHAIN_LENGTH_M 0.741
```

5.4.1.2 COMMAND_POS

```
#define COMMAND_POS 8
```


5.4.1.3 COMPONENT_ENDBIT

```
#define COMPONENT_ENDBIT 4
```

5.4.1.4 COMPONENT_POS

```
#define COMPONENT_POS 3
```

5.4.1.5 COMPONENT_STARTBIT

```
#define COMPONENT_STARTBIT 7
```

5.4.1.6 COUNT_PER_ROTATION

```
#define COUNT_PER_ROTATION 3533
```

5.4.1.7 CP_ALL_M_BV

```
#define CP_ALL_M_BV 48
```

5.4.1.8 CP_DEFAULT_BV

```
#define CP_DEFAULT_BV 0
```

5.4.1.9 CP_LEFT_M_BV

```
#define CP_LEFT_M_BV 16
```

5.4.1.10 CP_RIGHT_M_BV

```
#define CP_RIGHT_M_BV 32
```

5.4.1.11 ENC_1_A

```
#define ENC_1_A 6
```

5.4.1.12 ENC_1_B

```
#define ENC_1_B 7
```

5.4.1.13 ENC_2_A

```
#define ENC_2_A 5
```

5.4.1.14 ENC_2_B

```
#define ENC_2_B 4
```

5.4.1.15 FRAME_TYPE_ENDBIT

```
#define FRAME_TYPE_ENDBIT 5
```

5.4.1.16 FRAME_TYPE_POS

```
#define FRAME_TYPE_POS 1
```

5.4.1.17 FRAME_TYPE_STARTBIT

```
#define FRAME_TYPE_STARTBIT 7
```

5.4.1.18 FT_REQUEST_BV

```
#define FT_REQUEST_BV 32
```

5.4.1.19 FT_RESPONSE_BV

```
#define FT_RESPONSE_BV 64
```

5.4.1.20 FT_VALUE_BV

```
#define FT_VALUE_BV 0
```

5.4.1.21 KD_1

```
#define KD_1 0.0
```

5.4.1.22 KD_2

```
#define KD_2 0.0
```

5.4.1.23 KI_1

```
#define KI_1 0.0
```

5.4.1.24 KI_2

```
#define KI_2 0.0
```

5.4.1.25 KP_1

```
#define KP_1 10
```

5.4.1.26 KP_2

```
#define KP_2 10
```

5.4.1.27 LAST

```
#define LAST(  
    k,  
    n )  ((k) & ((1<<(n))-1))
```

5.4.1.28 M1_DIR_PIN

```
#define M1_DIR_PIN 11
```

5.4.1.29 M1_EN_PIN

```
#define M1_EN_PIN 10
```

5.4.1.30 M2_DIR_PIN

```
#define M2_DIR_PIN 8
```

5.4.1.31 M2_EN_PIN

```
#define M2_EN_PIN 9
```

5.4.1.32 MAX_PACKET_LENGTH

```
#define MAX_PACKET_LENGTH 50
```

5.4.1.33 MAX_SPEED

```
#define MAX_SPEED 0.6
```

5.4.1.34 MID

```
#define MID(  
    k,  
    m,  
    n ) LAST( (k)>>(m), ( (n) - (m) ) )
```

5.4.1.35 NUM_MOTORS

```
#define NUM_MOTORS 2
```

all constants and custom data types

5.4.1.36 P_GAIN

```
#define P_GAIN 1.0
```

5.4.1.37 PACKAGE_LENGTH_NO_P

```
#define PACKAGE_LENGTH_NO_P 10
```

5.4.1.38 PACKAGE_PAUSE_MS

```
#define PACKAGE_PAUSE_MS 10
```

5.4.1.39 PAYLOAD_LENGTH_POS

```
#define PAYLOAD_LENGTH_POS 2
```

5.4.1.40 PAYLOAD_POS

```
#define PAYLOAD_POS 8
```

5.4.1.41 PRINT_RATE_MS

```
#define PRINT_RATE_MS 500
```

5.4.1.42 REQUEST_DRIVE_MOTOR

```
#define REQUEST_DRIVE_MOTOR 0x01
```

5.4.1.43 REQUEST_RESET_TIMESTAMP

```
#define REQUEST_RESET_TIMESTAMP 0x02
```

5.4.1.44 RX_328_PIN

```
#define RX_328_PIN 6
```

5.4.1.45 SPEED_THRESHOLD

```
#define SPEED_THRESHOLD 8
```

5.4.1.46 START_BYTE

```
#define START_BYTE 0xAA
```

5.4.1.47 START_BYTE_POS

```
#define START_BYTE_POS 0
```

5.4.1.48 TIMESTAMP_POS

```
#define TIMESTAMP_POS 4
```

5.4.1.49 TX_328_PIN

```
#define TX_328_PIN 7
```

5.4.1.50 UART_BAUD_RATE

```
#define UART_BAUD_RATE 57600
```

5.4.1.51 UART_TRANSMIT_MS

```
#define UART_TRANSMIT_MS 10
```

5.4.1.52 VEL2TORQUE_RATIO

```
#define VEL2TORQUE_RATIO 1.0
```

5.4.1.53 VEL_UPDATE_RATE_MS

```
#define VEL_UPDATE_RATE_MS 20
```

5.4.2 Typedef Documentation

5.4.2.1 U_Component

```
typedef enum components U_Component
```

5.4.2.2 U_FrameType

```
typedef enum frametypes U_FrameType
```

5.4.2.3 U_Request

```
typedef enum requests U_Request
```

5.4.3 Enumeration Type Documentation

5.4.3.1 components

```
enum components
```

Enumerator

DEFAULT_COMPONENT	
LEFT_MOTOR	
RIGHT_MOTOR	
ALL_MOTORS	

5.4.3.2 frametypes

enum `frametypes`

Enumerator

VALUE	
REQUEST	
RESPONSE	

5.4.3.3 requests

enum `requests`

Enumerator

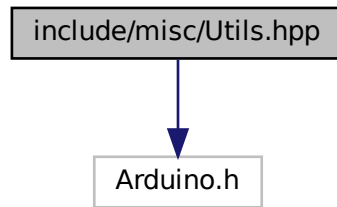
DEFAULT_REQUEST	
DRIVE_MOTOR	
RESET_TIMESTAMP	
DRIVE_ENCODER	
PID_PARAMETER	

5.5 include/misc/Utils.hpp File Reference

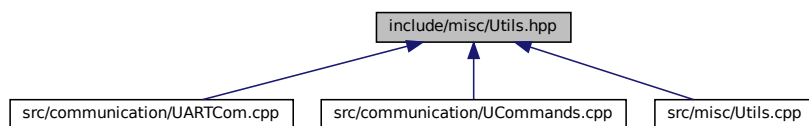
useful converting and computing functions


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

Include dependency graph for Utils.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [Utils](#)

5.5.1 Detailed Description

useful converting and computing functions

Author

your name (you@domain.com)

Version

0.1

Date

2021-07-22

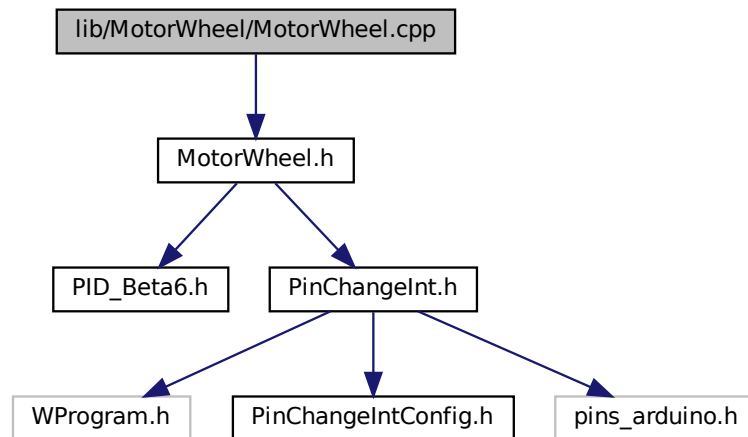
Copyright

Copyright (c) 2021

5.6 lib/MotorWheel/MotorWheel.cpp File Reference

```
#include <MotorWheel.h>
```

Include dependency graph for MotorWheel.cpp:

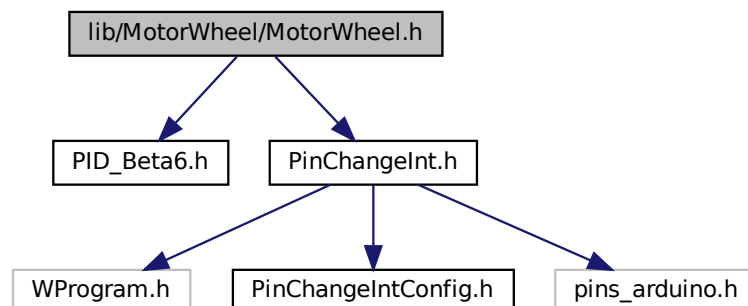


5.7 lib/MotorWheel/MotorWheel.h File Reference

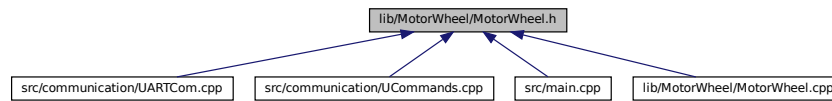
```
#include <PID_Beta6.h>
```

```
#include <PinChangeInt.h>
```

Include dependency graph for MotorWheel.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [ISRVars](#)
- class [Motor](#)
- class [GearedMotor](#)
- class [MotorWheel](#)

Macros

- `#define` [DIR_ADVANCE](#) HIGH
- `#define` [DIR_BACKOFF](#) LOW
- `#define` [PIN_UNDEFINED](#) 255
- `#define` [REF_VOLT](#) 12
- `#define` [MAX_PWM](#) 255
- `#define` [TRIGGER](#) CHANGE
- `#define` [CPR](#) 24
- `#define` [DIR_INVERSE](#) !
- `#define` [REDUCTION_RATIO](#) 64
- `#define` [MAX_SPEEDRPM](#) 8000
- `#define` [SEC_PER_MIN](#) 60
- `#define` [MICROS_PER_SEC](#) 1000000
- `#define` [SPEEDPPS2SPEEDRPM](#)(freq) ((unsigned long)(freq)*([SEC_PER_MIN](#))/([CPR](#)))
- `#define` [KC](#) 0.31
- `#define` [TAUI](#) 0.02
- `#define` [TAUD](#) 0.00
- `#define` [SAMPLETIME](#) 5
- `#define` [Baudrate](#) 19200
- `#define` [debug](#)() {}
- `#define` [irqISR](#)(y, x)
- `#define` [PI](#) 3.1416
- `#define` [CIRMM](#) 314

5.7.1 Macro Definition Documentation

5.7.1.1 Baudrate

```
#define Baudrate 19200
```

5.7.1.2 CIRMM

```
#define CIRMM 314
```

5.7.1.3 CPR

```
#define CPR 24
```

5.7.1.4 debug

```
#define debug( ) {}
```

5.7.1.5 DIR_ADVANCE

```
#define DIR_ADVANCE HIGH
```

5.7.1.6 DIR_BACKOFF

```
#define DIR_BACKOFF LOW
```

5.7.1.7 DIR_INVERSE

```
#define DIR_INVERSE !
```

5.7.1.8 irqISR

```
#define irqISR(  
    y,  
    x )
```

Value:

```
void x(); \
struct ISRVars y={x}; \
void x() { \
    static bool first_pulse=true; \
    y.pulseEndMicros=micros(); \
    if(first_pulse==false && y.pulseEndMicros>y.pulseStartMicros) { \
        y.speedPPS=MICROS_PER_SEC/(y.pulseEndMicros-y.pulseStartMicros); \
        /* y.accPPS=(y.speedPPS-y.lastSpeedPPS)*y.speedPPS; */ \
    } else first_pulse=false; \
    y.pulseStartMicros=y.pulseEndMicros; \
    /* y.lastSpeedPPS=y.speedPPS; */ \
    if(y.pinIRQB!=PIN_UNDEFINED) \
        y.currDirection=DIR_INVERSE(digitalRead(y.pinIRQ)^digitalRead(y.pinIRQB)); \
    y.currDirection==DIR_ADVANCE?++y.pulses:--y.pulses; \
}
```

5.7.1.9 KC

```
#define KC 0.31
```

5.7.1.10 MAX_PWM

```
#define MAX_PWM 255
```

5.7.1.11 MAX_SPEEDRPM

```
#define MAX_SPEEDRPM 8000
```

5.7.1.12 MICROS_PER_SEC

```
#define MICROS_PER_SEC 1000000
```

5.7.1.13 PI

```
#define PI 3.1416
```

5.7.1.14 PIN_UNDEFINED

```
#define PIN_UNDEFINED 255
```

5.7.1.15 REDUCTION_RATIO

```
#define REDUCTION_RATIO 64
```

5.7.1.16 REF_VOLT

```
#define REF_VOLT 12
```

5.7.1.17 SAMPLETIME

```
#define SAMPLETIME 5
```

5.7.1.18 SEC_PER_MIN

```
#define SEC_PER_MIN 60
```

5.7.1.19 SPEEDPPS2SPEEDRPM

```
#define SPEEDPPS2SPEEDRPM(  
    freq ) ((unsigned long) (freq)*(SEC_PER_MIN)/(CPR))
```

5.7.1.20 TAUD

```
#define TAUD 0.00
```

5.7.1.21 TAU1

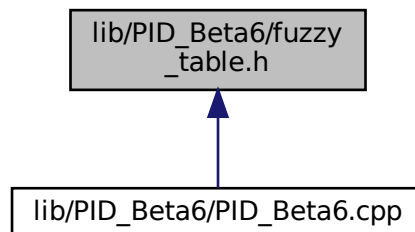
```
#define TAU1 0.02
```

5.7.1.22 TRIGGER

```
#define TRIGGER CHANGE
```

5.8 lib/PID_Beta6/fuzzy_table.h File Reference

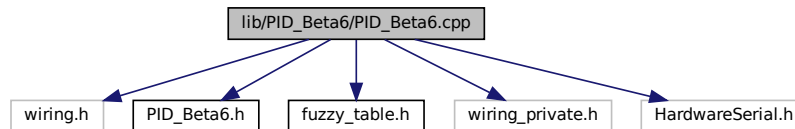
This graph shows which files directly or indirectly include this file:



5.9 lib/PID_Beta6/PID_Beta6.cpp File Reference

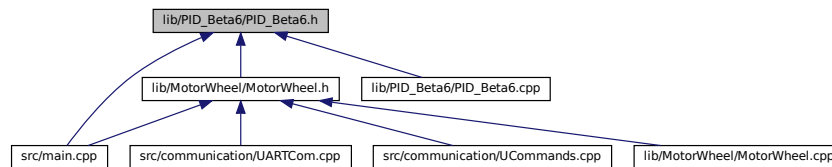
```
#include "wiring.h"
#include <PID_Beta6.h>
#include "fuzzy_table.h"
#include <wiring_private.h>
#include <HardwareSerial.h>
```

Include dependency graph for PID_Beta6.cpp:



5.10 lib/PID_Beta6/PID_Beta6.h File Reference

This graph shows which files directly or indirectly include this file:



Classes

- class [PID](#)

Macros

- #define [AUTO](#) 1
- #define [MANUAL](#) 0
- #define [LIBRARY_VERSION](#) 0.6

5.10.1 Macro Definition Documentation

5.10.1.1 AUTO

```
#define AUTO 1
```

5.10.1.2 LIBRARY_VERSION

```
#define LIBRARY_VERSION 0.6
```

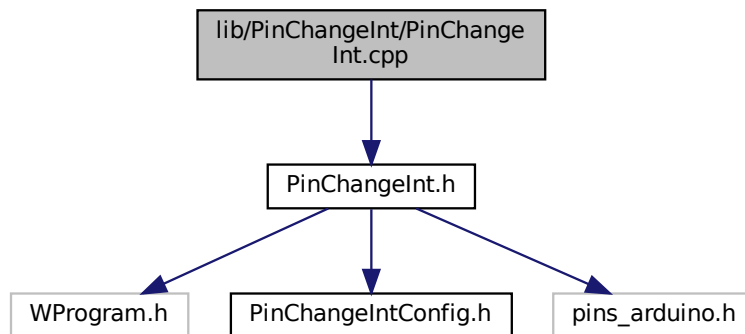
5.10.1.3 MANUAL

```
#define MANUAL 0
```

5.11 lib/PinChangeInt/PinChangeInt.cpp File Reference

```
#include <PinChangeInt.h>
```

Include dependency graph for PinChangeInt.cpp:



Functions

- [ISR](#) (PCINT0_vect)
- [ISR](#) (PCINT1_vect)
- [ISR](#) (PCINT2_vect)

5.11.1 Function Documentation

5.11.1.1 ISR() [1/3]

```
ISR (
    PCINT0_vect )
```

Here is the call graph for this function:



5.11.1.2 ISR() [2/3]

```
ISR (
    PCINT1_vect )
```

Here is the call graph for this function:



5.11.1.3 ISR() [3/3]

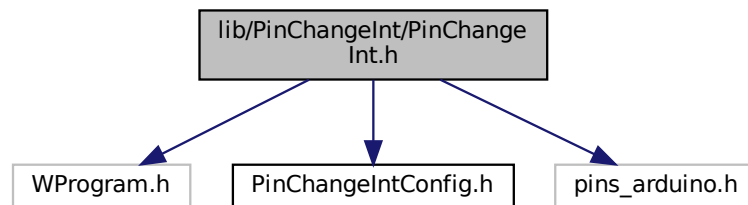
```
ISR (
    PCINT2_vect )
```

Here is the call graph for this function:

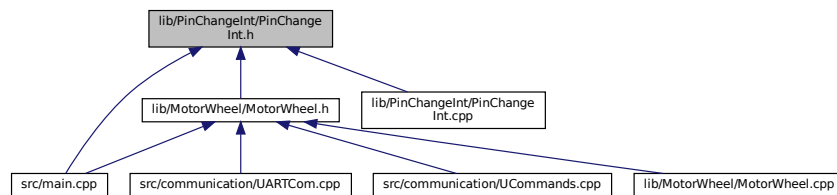


5.12 lib/PinChangeInt/PinChangeInt.h File Reference

```
#include "WProgram.h"
#include "PinChangeIntConfig.h"
#include "pins_arduino.h"
Include dependency graph for PinChangeInt.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [PCIntPort](#)
- class [PCIntPort::PCIntPin](#)

Macros

- `#define` [MAX_PIN_CHANGE_PINS](#) 8
- `#define` [INLINE_PCINT](#)
- `#define` [PCdetachInterrupt](#)(pin) [PCIntPort::detachInterrupt](#)(pin)
- `#define` [PCattachInterrupt](#)(pin, userFunc, mode) [PCIntPort::attachInterrupt](#)(pin, userFunc, mode)

Typedefs

- typedef void(* [PCIntvoidFuncPtr](#)) (void)

5.12.1 Macro Definition Documentation

5.12.1.1 INLINE_PCINT

```
#define INLINE_PCINT
```

5.12.1.2 MAX_PIN_CHANGE_PINS

```
#define MAX_PIN_CHANGE_PINS 8
```

5.12.1.3 PCattachInterrupt

```
#define PCattachInterrupt(  
    pin,  
    userFunc,  
    mode ) PCintPort::attachInterrupt(pin, userFunc,mode)
```

5.12.1.4 PCdetachInterrupt

```
#define PCdetachInterrupt(  
    pin ) PCintPort::detachInterrupt(pin)
```

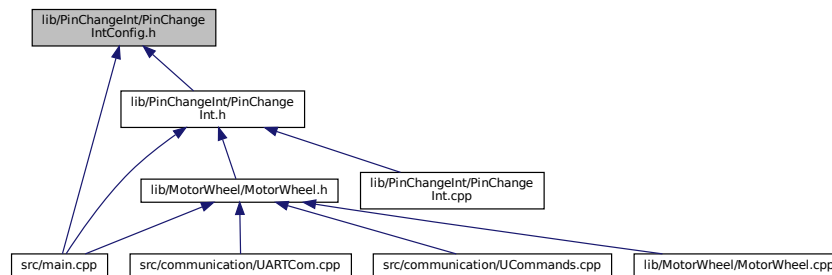
5.12.2 Typedef Documentation

5.12.2.1 PCIntvoidFuncPtr

```
typedef void(* PCIntvoidFuncPtr) (void)
```

5.13 lib/PinChangeInt/PinChangeIntConfig.h File Reference

This graph shows which files directly or indirectly include this file:



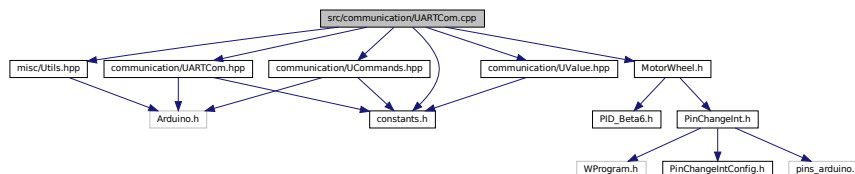
5.14 src/communication/UARTCom.cpp File Reference

```

#include "communication/UARTCom.hpp"
#include "communication/UCommands.hpp"
#include "communication/UValue.hpp"
#include "misc/Utils.hpp"
#include "constants.h"
#include "MotorWheel.h"

```

Include dependency graph for UARTCom.cpp:



Functions

- [U_FrameType parseFrameType](#) (uint8_t *d)
extracts frametype from data package
- [uint8_t fromFrameType](#) (U_FrameType ft)
translates frametype to corresponding byte value
- [U_Component parseComponent](#) (uint8_t *d)
extract component information
- [uint8_t fromComponent](#) (U_Component cp)
translates component into byte value
- [U_Request parseRequest](#) (uint8_t *d)
extracts request information from data package
- [uint32_t parseTimestamp](#) (uint8_t *d)
extracts timestamp from data package

- uint16_t `computeChecksum` (uint8_t *d, uint8_t pl)
computes checksum from data package
- int `correctChecksum` (uint8_t *d)
checks if checksum inside package and computed checksum from package data are the same
- void `showPackage` (uint8_t *d, int pl)
debugging function, prints out package data
- void `clearPackage` (uint8_t *d)
erases all package data and sets the array to 0
- void `handleRequest` (uint8_t *d)
executes commands according to incoming request
- void `handlePackage` (uint8_t *d, int pl)
package handler, checks for errors first, then proceeds to extract information from the data package

Variables

- uint8_t `currentParseIndex`
- int `payloadLength`
- uint32_t `currentTimestamp` = 0
- `MotorWheel` `wheelLeft`
- `MotorWheel` `wheelRight`

5.14.1 Function Documentation

5.14.1.1 `clearPackage()`

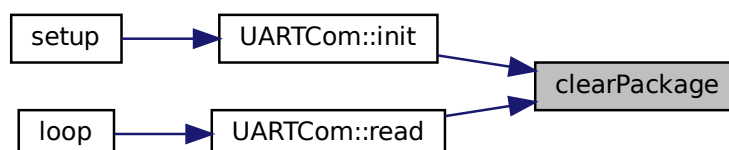
```
void clearPackage (
    uint8_t * d )
```

erases all package data and sets the array to 0

Parameters

<i>d</i>	data package
----------	--------------

Here is the caller graph for this function:



5.14.1.2 computeChecksum()

```
uint16_t computeChecksum (  
    uint8_t * d,  
    uint8_t pl )
```

computes checksum from data package

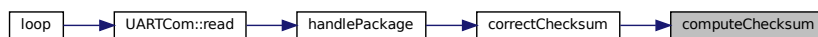
Parameters

<i>d</i>	data package
<i>pl</i>	payload length

Returns

uint16_t

Here is the caller graph for this function:



5.14.1.3 correctChecksum()

```
int correctChecksum (  
    uint8_t * d )
```

checks if checksum inside package and computed checksum from package data are the same

Parameters

<i>d</i>	data package
----------	--------------

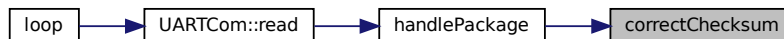
Returns

int

Here is the call graph for this function:



Here is the caller graph for this function:

**5.14.1.4 fromComponent()**

```
uint8_t fromComponent (
    U_Component cp )
```

translates component into byte value

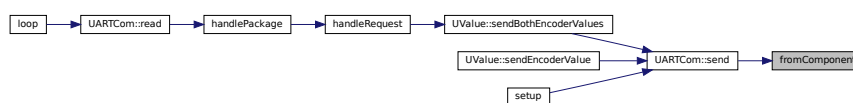
Parameters

<i>cp</i>	component
-----------	-----------

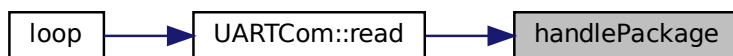
Returns

uint8_t

Here is the caller graph for this function:



Here is the caller graph for this function:



5.14.1.7 `handleRequest()`

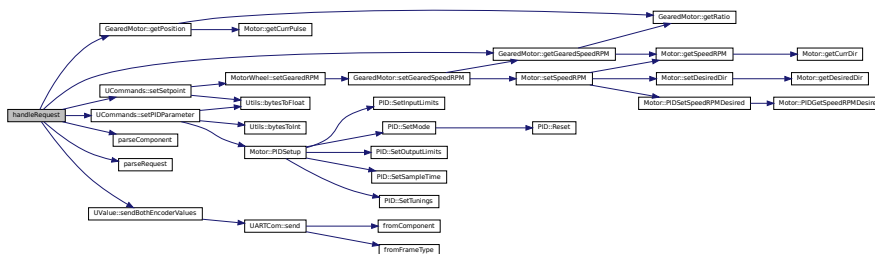
```
void handleRequest (
    uint8_t * d )
```

executes commands according to incoming request

Parameters

d	data package
-----	--------------

Here is the call graph for this function:



Here is the caller graph for this function:



5.14.1.8 parseComponent()

```
U_Component parseComponent (
    uint8_t * d )
```

extract component information

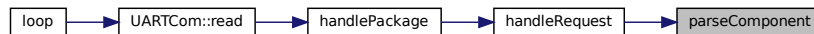
Parameters

<i>d</i>	data package
----------	--------------

Returns

U_Component

Here is the caller graph for this function:



5.14.1.9 parseFrameType()

```
U_FrameType parseFrameType (
    uint8_t * d )
```

extracts frametype from data package

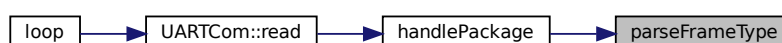
Parameters

<i>d</i>	data package
----------	--------------

Returns

U_FrameType

Here is the caller graph for this function:



5.14.1.10 parseRequest()

```
U_Request parseRequest (  
    uint8_t * d )
```

extracts request information from data package

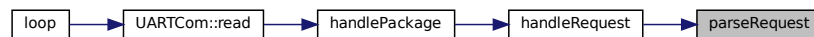
Parameters

<i>d</i>	data package
----------	--------------

Returns

U_Request

Here is the caller graph for this function:



5.14.1.11 parseTimestamp()

```
uint32_t parseTimestamp (  
    uint8_t * d )
```

extracts timestamp from data package

Parameters

<i>d</i>	data package
----------	--------------

Returns

uint32_t

Here is the call graph for this function:



5.14.1.12 showPackage()

```
void showPackage (
    uint8_t * d,
    int pl )
```

debugging function, prints out package data

Parameters

<i>d</i>	data package
<i>pl</i>	payload length

Here is the caller graph for this function:



5.14.2 Variable Documentation

5.14.2.1 currentParseIndex

```
uint8_t currentParseIndex
```

5.14.2.2 currentTimestamp

```
uint32_t currentTimestamp = 0
```

5.14.2.3 payloadLength

```
int payloadLength
```

5.14.2.4 wheelLeft

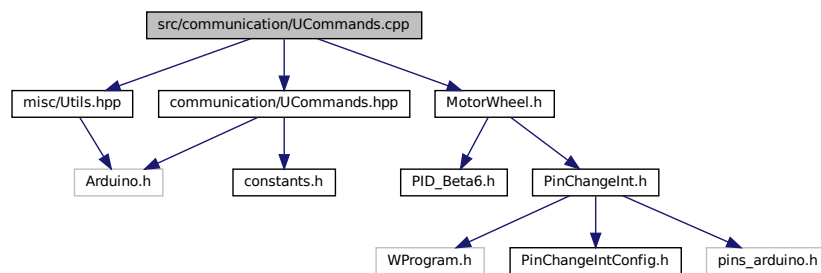
`MotorWheel wheelLeft [extern]`

5.14.2.5 wheelRight

`MotorWheel wheelRight [extern]`

5.15 src/communication/UCommands.cpp File Reference

```
#include "communication/UCommands.hpp"
#include "misc/Utils.hpp"
#include "MotorWheel.h"
Include dependency graph for UCommands.cpp:
```



Variables

- `MotorWheel wheelLeft`
- `MotorWheel wheelRight`

5.15.1 Variable Documentation

5.15.1.1 wheelLeft

`MotorWheel wheelLeft [extern]`

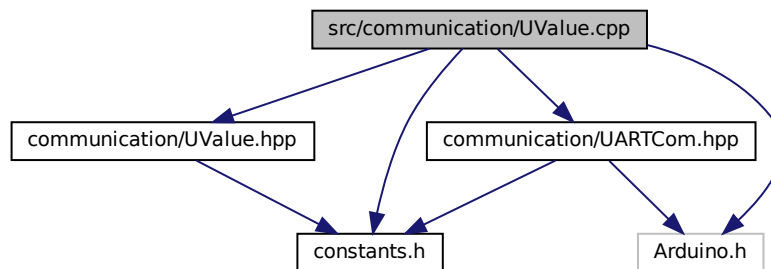
5.15.1.2 wheelRight

```
MotorWheel wheelRight [extern]
```

5.16 src/communication/UValue.cpp File Reference

```
#include "communication/UValue.hpp"
#include "communication/UARTCom.hpp"
#include "Arduino.h"
#include "constants.h"
```

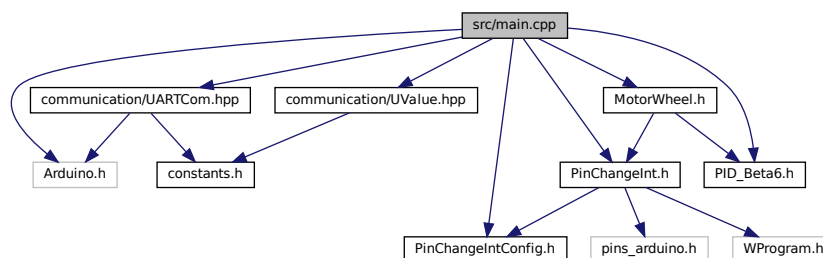
Include dependency graph for UValue.cpp:



5.17 src/main.cpp File Reference

```
#include <Arduino.h>
#include "communication/UARTCom.hpp"
#include "communication/UValue.hpp"
#include <PinChangeInt.h>
#include <PinChangeIntConfig.h>
#include <PID_Beta6.h>
#include <MotorWheel.h>
```

Include dependency graph for main.cpp:



Macros

- `#define` `MICROS_PER_SEC` 1000000

Functions

- `irqISR` (`irq1`, `isr1`)
- `irqISR` (`irq2`, `isr2`)
- `void` `setup` (`void`)
- `void` `loop` (`void`)

Variables

- `long` `controlTimer` = 0
- `long` `uartTransmitTimer` = 0
- `int` `speed` = 0
- `MotorWheel` `wheelLeft` (9, 8, 4, 5, &`irq1`, `REDUCTION_RATIO`, 300)
- `MotorWheel` `wheelRight` (10, 11, 6, 7, &`irq2`, `REDUCTION_RATIO`, 300)

5.17.1 Macro Definition Documentation

5.17.1.1 MICROS_PER_SEC

```
#define MICROS_PER_SEC 1000000
```

5.17.2 Function Documentation

5.17.2.1 irqISR() [1/2]

```
irqISR (  
    irq1 ,  
    isr1 )
```

5.17.2.2 irqISR() [2/2]

```
irqISR (  
    irq2 ,  
    isr2 )
```


5.17.3.2 speed

```
int speed = 0
```

5.17.3.3 uartTransmitTimer

```
long uartTransmitTimer = 0
```

5.17.3.4 wheelLeft

```
MotorWheel wheelLeft(9, 8, 4, 5, &irq1, REDUCTION_RATIO, 300) (  
    9 ,  
    8 ,  
    4 ,  
    5 ,  
    & irq1,  
    REDUCTION_RATIO ,  
    300 )
```

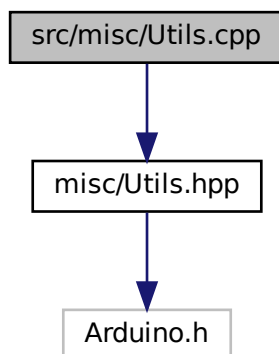
5.17.3.5 wheelRight

```
MotorWheel wheelRight(10, 11, 6, 7, &irq2, REDUCTION_RATIO, 300) (  
    10 ,  
    11 ,  
    6 ,  
    7 ,  
    & irq2,  
    REDUCTION_RATIO ,  
    300 )
```

5.18 src/misc/Utils.cpp File Reference

```
#include "misc/Utils.hpp"
```

Include dependency graph for Utils.cpp:



Index

- addPin
 - PCintPort, [37](#)
- advancePWM
 - Motor, [15](#)
- ALL_MOTORS
 - constants.h, [70](#)
- attachInterrupt
 - PCintPort, [37](#)
- AUTO
 - PID_Beta6.h, [77](#)
- backoffPWM
 - Motor, [15](#)
- Baudrate
 - MotorWheel.h, [73](#)
- bytesToFloat
 - Utils, [51](#)
- bytesToInt
 - Utils, [52](#)
- CHAIN_LENGTH_M
 - constants.h, [62](#)
- CIRMM
 - MotorWheel.h, [73](#)
- clearPackage
 - UARTCom.cpp, [83](#)
- COMMAND_POS
 - constants.h, [62](#)
- COMPONENT_ENDBIT
 - constants.h, [62](#)
- COMPONENT_POS
 - constants.h, [63](#)
- COMPONENT_STARTBIT
 - constants.h, [63](#)
- components
 - constants.h, [69](#)
- Compute
 - PID, [42](#)
- computeChecksum
 - UARTCom.cpp, [84](#)
- constants.h
 - ALL_MOTORS, [70](#)
 - CHAIN_LENGTH_M, [62](#)
 - COMMAND_POS, [62](#)
 - COMPONENT_ENDBIT, [62](#)
 - COMPONENT_POS, [63](#)
 - COMPONENT_STARTBIT, [63](#)
 - components, [69](#)
 - COUNT_PER_ROTATION, [63](#)
 - CP_ALL_M_BV, [63](#)
 - CP_DEFAULT_BV, [63](#)
 - CP_LEFT_M_BV, [63](#)
 - CP_RIGHT_M_BV, [63](#)
 - DEFAULT_COMPONENT, [70](#)
 - DEFAULT_REQUEST, [70](#)
 - DRIVE_ENCODER, [70](#)
 - DRIVE_MOTOR, [70](#)
 - ENC_1_A, [63](#)
 - ENC_1_B, [64](#)
 - ENC_2_A, [64](#)
 - ENC_2_B, [64](#)
 - FRAME_TYPE_ENDBIT, [64](#)
 - FRAME_TYPE_POS, [64](#)
 - FRAME_TYPE_STARTBIT, [64](#)
 - frametypes, [70](#)
 - FT_REQUEST_BV, [64](#)
 - FT_RESPONSE_BV, [64](#)
 - FT_VALUE_BV, [65](#)
 - KD_1, [65](#)
 - KD_2, [65](#)
 - KI_1, [65](#)
 - KI_2, [65](#)
 - KP_1, [65](#)
 - KP_2, [65](#)
 - LAST, [65](#)
 - LEFT_MOTOR, [70](#)
 - M1_DIR_PIN, [66](#)
 - M1_EN_PIN, [66](#)
 - M2_DIR_PIN, [66](#)
 - M2_EN_PIN, [66](#)
 - MAX_PACKET_LENGTH, [66](#)
 - MAX_SPEED, [66](#)
 - MID, [66](#)
 - NUM_MOTORS, [67](#)
 - P_GAIN, [67](#)
 - PACKAGE_LENGTH_NO_P, [67](#)
 - PACKAGE_PAUSE_MS, [67](#)
 - PAYLOAD_LENGTH_POS, [67](#)
 - PAYLOAD_POS, [67](#)
 - PID_PARAMETER, [70](#)
 - PRINT_RATE_MS, [67](#)
 - REQUEST, [70](#)
 - REQUEST_DRIVE_MOTOR, [68](#)
 - REQUEST_RESET_TIMESTAMP, [68](#)
 - requests, [70](#)
 - RESET_TIMESTAMP, [70](#)
 - RESPONSE, [70](#)
 - RIGHT_MOTOR, [70](#)
 - RX_328_PIN, [68](#)

- SPEED_THRESHOLD, 68
- START_BYTE, 68
- START_BYTE_POS, 68
- TIMESTAMP_POS, 68
- TX_328_PIN, 68
- U_Component, 69
- U_FrameType, 69
- U_Request, 69
- UART_BAUD_RATE, 69
- UART_TRANSMIT_MS, 69
- VALUE, 70
- VEL2TORQUE_RATIO, 69
- VEL_UPDATE_RATE_MS, 69
- controlTimer
 - main.cpp, 94
- correctChecksum
 - UARTCom.cpp, 84
- COUNT_PER_ROTATION
 - constants.h, 63
- CP_ALL_M_BV
 - constants.h, 63
- CP_DEFAULT_BV
 - constants.h, 63
- CP_LEFT_M_BV
 - constants.h, 63
- CP_RIGHT_M_BV
 - constants.h, 63
- CPR
 - MotorWheel.h, 74
- currDirection
 - ISRVars, 11
- currentParseIndex
 - UARTCom.cpp, 90
- currentTimestamp
 - UARTCom.cpp, 90
- data
 - UARTCom, 48
- debug
 - MotorWheel.h, 74
- debugger
 - Motor, 16
- DEFAULT_COMPONENT
 - constants.h, 70
- DEFAULT_REQUEST
 - constants.h, 70
- delayMS
 - Motor, 16
- delPin
 - PCintPort, 37
- detachInterrupt
 - PCintPort, 38
- DIR_ADVANCE
 - MotorWheel.h, 74
- DIR_BACKOFF
 - MotorWheel.h, 74
- DIR_INVERSE
 - MotorWheel.h, 74
- DRIVE_ENCODER
 - constants.h, 70
- DRIVE_MOTOR
 - constants.h, 70
- ENC_1_A
 - constants.h, 63
- ENC_1_B
 - constants.h, 64
- ENC_2_A
 - constants.h, 64
- ENC_2_B
 - constants.h, 64
- FRAME_TYPE_ENDBIT
 - constants.h, 64
- FRAME_TYPE_POS
 - constants.h, 64
- FRAME_TYPE_STARTBIT
 - constants.h, 64
- frametypes
 - constants.h, 70
- fromComponent
 - UARTCom.cpp, 85
- fromFrameType
 - UARTCom.cpp, 85
- FT_REQUEST_BV
 - constants.h, 64
- FT_RESPONSE_BV
 - constants.h, 64
- FT_VALUE_BV
 - constants.h, 65
- GearedMotor, 7
 - GearedMotor, 8
 - getGearedSpeedRPM, 9
 - getPosition, 9
 - getRatio, 9
 - setGearedSpeedRPM, 10
 - setRatio, 10
- getCirMM
 - MotorWheel, 30
- getCurrDir
 - Motor, 17
- getCurrPulse
 - Motor, 17
- GetD_Param
 - PID, 42
- getDesiredDir
 - Motor, 17
- getGearedSpeedRPM
 - GearedMotor, 9
- GetI_Param
 - PID, 42
- GetINMax
 - PID, 42
- GetINMin
 - PID, 42
- GetMode
 - PID, 42

- GetOUTMax
 - PID, [43](#)
- GetOUTMin
 - PID, [43](#)
- GetP_Param
 - PID, [43](#)
- getPinDir
 - Motor, [18](#)
- getPinIRQ
 - Motor, [18](#)
- getPinIRQB
 - Motor, [18](#)
- getPinPWM
 - Motor, [18](#)
- getPosition
 - GearedMotor, [9](#)
- getPWM
 - Motor, [18](#)
- getRatio
 - GearedMotor, [9](#)
- GetSampleTime
 - PID, [43](#)
- getSpeedCMPM
 - MotorWheel, [30](#)
- getSpeedMMPS
 - MotorWheel, [31](#)
- getSpeedPPS
 - Motor, [19](#)
- getSpeedRPM
 - Motor, [19](#)
- handlePackage
 - UARTCom.cpp, [86](#)
- handleRequest
 - UARTCom.cpp, [87](#)
- include/communication/UARTCom.hpp, [57](#)
- include/communication/UCommands.hpp, [58](#)
- include/communication/UValue.hpp, [59](#)
- include/constants.h, [61](#)
- include/misc/Utils.hpp, [70](#)
- init
 - UARTCom, [46](#)
- INLINE_PCINT
 - PinChangeInt.h, [81](#)
- irqISR
 - main.cpp, [93](#)
 - MotorWheel.h, [74](#)
- ISR
 - PinChangeInt.cpp, [78](#), [79](#)
- isr
 - Motor, [28](#)
- ISRfunc
 - ISRVars, [11](#)
- ISRVars, [11](#)
 - currDirection, [11](#)
 - ISRfunc, [11](#)
 - pinIRQ, [12](#)
 - pinIRQB, [12](#)
 - pulseEndMicros, [12](#)
 - pulses, [12](#)
 - pulseStartMicros, [12](#)
 - speedPPS, [12](#)
- JustCalculated
 - PID, [43](#)
- KC
 - MotorWheel.h, [74](#)
- KD_1
 - constants.h, [65](#)
- KD_2
 - constants.h, [65](#)
- KI_1
 - constants.h, [65](#)
- KI_2
 - constants.h, [65](#)
- KP_1
 - constants.h, [65](#)
- KP_2
 - constants.h, [65](#)
- LAST
 - constants.h, [65](#)
- LEFT_MOTOR
 - constants.h, [70](#)
- lib/MotorWheel/MotorWheel.cpp, [72](#)
- lib/MotorWheel/MotorWheel.h, [72](#)
- lib/PID_Beta6/fuzzy_table.h, [76](#)
- lib/PID_Beta6/PID_Beta6.cpp, [77](#)
- lib/PID_Beta6/PID_Beta6.h, [77](#)
- lib/PinChangeInt/PinChangeInt.cpp, [78](#)
- lib/PinChangeInt/PinChangeInt.h, [80](#)
- lib/PinChangeInt/PinChangeIntConfig.h, [82](#)
- LIBRARY_VERSION
 - PID_Beta6.h, [78](#)
- loop
 - main.cpp, [93](#)
- M1_DIR_PIN
 - constants.h, [66](#)
- M1_EN_PIN
 - constants.h, [66](#)
- M2_DIR_PIN
 - constants.h, [66](#)
- M2_EN_PIN
 - constants.h, [66](#)
- main.cpp
 - controlTimer, [94](#)
 - irqISR, [93](#)
 - loop, [93](#)
 - MICROS_PER_SEC, [93](#)
 - setup, [94](#)
 - speed, [94](#)
 - uartTransmitTimer, [95](#)
 - wheelLeft, [95](#)
 - wheelRight, [95](#)
- MANUAL

- PID_Beta6.h, 78
- MAX_PACKET_LENGTH
 - constants.h, 66
- MAX_PIN_CHANGE_PINS
 - PinChangeInt.h, 81
- MAX_PWM
 - MotorWheel.h, 75
- MAX_SPEED
 - constants.h, 66
- MAX_SPEEDRPM
 - MotorWheel.h, 75
- MICROS_PER_SEC
 - main.cpp, 93
 - MotorWheel.h, 75
- MID
 - constants.h, 66
- Motor, 13
 - advancePWM, 15
 - backoffPWM, 15
 - debugger, 16
 - delayMS, 16
 - getCurrDir, 17
 - getCurrPulse, 17
 - getDesiredDir, 17
 - getPinDir, 18
 - getPinIRQ, 18
 - getPinIRQB, 18
 - getPinPWM, 18
 - getPWM, 18
 - getSpeedPPS, 19
 - getSpeedRPM, 19
 - isr, 28
 - Motor, 14
 - PIDDisable, 19
 - PIDEnable, 20
 - PIDGetSpeedRPMDesired, 20
 - PIDGetStatus, 21
 - PIDRegulate, 21
 - PIDReset, 22
 - PIDSetSpeedRPMDesired, 23
 - PIDSetup, 23
 - resetCurrPulse, 24
 - reverseDesiredDir, 24
 - runPWM, 25
 - setCurrDir, 25
 - setCurrPulse, 26
 - setDesiredDir, 26
 - setSpeedRPM, 27
 - setupInterrupt, 28
- MotorWheel, 29
 - getCirMM, 30
 - getSpeedCMPM, 30
 - getSpeedMMPS, 31
 - MotorWheel, 30
 - setCirMM, 31
 - setGearedRPM, 32
 - setSpeedCMPM, 32, 33
 - setSpeedMMPS, 33, 34
- MotorWheel.h
 - Baudrate, 73
 - CIRMM, 73
 - CPR, 74
 - debug, 74
 - DIR_ADVANCE, 74
 - DIR_BACKOFF, 74
 - DIR_INVERSE, 74
 - irqISR, 74
 - KC, 74
 - MAX_PWM, 75
 - MAX_SPEEDRPM, 75
 - MICROS_PER_SEC, 75
 - PI, 75
 - PIN_UNDEFINED, 75
 - REDUCTION_RATIO, 75
 - REF_VOLT, 75
 - SAMPLETIME, 75
 - SEC_PER_MIN, 76
 - SPEEDPPS2SPEEDRPM, 76
 - TAUD, 76
 - TAUI, 76
 - TRIGGER, 76
- NUM_MOTORS
 - constants.h, 67
- P_GAIN
 - constants.h, 67
- PACKAGE_LENGTH_NO_P
 - constants.h, 67
- PACKAGE_PAUSE_MS
 - constants.h, 67
- parseComponent
 - UARTCom.cpp, 87
- parseFrameType
 - UARTCom.cpp, 88
- parseRequest
 - UARTCom.cpp, 88
- parseTimestamp
 - UARTCom.cpp, 89
- PAYLOAD_LENGTH_POS
 - constants.h, 67
- PAYLOAD_POS
 - constants.h, 67
- payloadLength
 - UARTCom.cpp, 90
- PCattachInterrupt
 - PinChangeInt.h, 81
- PCdetachInterrupt
 - PinChangeInt.h, 81
- PCICRbit
 - PCintPort, 39
- PCint
 - PCintPort, 38
- PCintFunc
 - PCintPort::PCintPin, 35
- PCintLast
 - PCintPort, 39

- PCIntMask
 - PCIntPort::PCIntPin, 35
- PCIntMode
 - PCIntPort::PCIntPin, 35
- PCIntPin
 - PCIntPort::PCIntPin, 35
- pcIntPins
 - PCIntPort, 39
- PCIntPort, 36
 - addPin, 37
 - attachInterrupt, 37
 - delPin, 37
 - detachInterrupt, 38
 - PCICRbit, 39
 - PCint, 38
 - PCintLast, 39
 - pcIntPins, 39
 - pcIntPorts, 39
 - pcmask, 39
 - portInputReg, 39
- PCIntPort::PCIntPin, 34
 - PCintFunc, 35
 - PCIntMask, 35
 - PCIntMode, 35
 - PCintPin, 35
 - pinDataAlloc, 35
- pcIntPorts
 - PCIntPort, 39
- PCIntvoidFuncPtr
 - PinChangeInt.h, 81
- pcmask
 - PCIntPort, 39
- PI
 - MotorWheel.h, 75
- PID, 40
 - Compute, 42
 - GetD_Param, 42
 - GetI_Param, 42
 - GetINMax, 42
 - GetINMin, 42
 - GetMode, 42
 - GetOUTMax, 43
 - GetOUTMin, 43
 - GetP_Param, 43
 - GetSampleTime, 43
 - JustCalculated, 43
 - PID, 41
 - Reset, 43
 - SetInputLimits, 44
 - SetMode, 44
 - SetOutputLimits, 44
 - SetSampleTime, 45
 - SetTunings, 45
- PID_Beta6.h
 - AUTO, 77
 - LIBRARY_VERSION, 78
 - MANUAL, 78
- PID_PARAMETER
 - constants.h, 70
- PIDDisable
 - Motor, 19
- PIDEnable
 - Motor, 20
- PIDGetSpeedRPMDesired
 - Motor, 20
- PIDGetStatus
 - Motor, 21
- PIDRegulate
 - Motor, 21
- PIDReset
 - Motor, 22
- PIDSetSpeedRPMDesired
 - Motor, 23
- PIDSetup
 - Motor, 23
- PIN_UNDEFINED
 - MotorWheel.h, 75
- PinChangeInt.cpp
 - ISR, 78, 79
- PinChangeInt.h
 - INLINE_PCINT, 81
 - MAX_PIN_CHANGE_PINS, 81
 - PCattachInterrupt, 81
 - PCdetachInterrupt, 81
 - PCIntvoidFuncPtr, 81
- pinDataAlloc
 - PCIntPort::PCIntPin, 35
- pinIRQ
 - ISRVars, 12
- pinIRQB
 - ISRVars, 12
- portInputReg
 - PCIntPort, 39
- PRINT_RATE_MS
 - constants.h, 67
- pulseEndMicros
 - ISRVars, 12
- pulses
 - ISRVars, 12
- pulseStartMicros
 - ISRVars, 12
- read
 - UARTCom, 47
- REDUCTION_RATIO
 - MotorWheel.h, 75
- REF_VOLT
 - MotorWheel.h, 75
- REQUEST
 - constants.h, 70
- REQUEST_DRIVE_MOTOR
 - constants.h, 68
- REQUEST_RESET_TIMESTAMP
 - constants.h, 68
- requests
 - constants.h, 70
- Reset

- PID, [43](#)
- RESET_TIMESTAMP
 - constants.h, [70](#)
- resetCurrPulse
 - Motor, [24](#)
- RESPONSE
 - constants.h, [70](#)
- reverseDesiredDir
 - Motor, [24](#)
- RIGHT_MOTOR
 - constants.h, [70](#)
- runPWM
 - Motor, [25](#)
- RX_328_PIN
 - constants.h, [68](#)
- SAMPLETIME
 - MotorWheel.h, [75](#)
- SEC_PER_MIN
 - MotorWheel.h, [76](#)
- send
 - UARTCom, [47](#)
- sendBothEncoderValues
 - UValue, [53](#)
- sendEncoderValue
 - UValue, [54](#)
- setCirMM
 - MotorWheel, [31](#)
- setCurrDir
 - Motor, [25](#)
- setCurrPulse
 - Motor, [26](#)
- setDesiredDir
 - Motor, [26](#)
- setGearedRPM
 - MotorWheel, [32](#)
- setGearedSpeedRPM
 - GearedMotor, [10](#)
- SetInputLimits
 - PID, [44](#)
- SetMode
 - PID, [44](#)
- SetOutputLimits
 - PID, [44](#)
- setPIDParameter
 - UCommands, [49](#)
- setRatio
 - GearedMotor, [10](#)
- SetSampleTime
 - PID, [45](#)
- setSetpoint
 - UCommands, [50](#)
- setSpeedCMPM
 - MotorWheel, [32, 33](#)
- setSpeedMMPS
 - MotorWheel, [33, 34](#)
- setSpeedRPM
 - Motor, [27](#)
- SetTunings
 - PID, [45](#)
- setup
 - main.cpp, [94](#)
- setupInterrupt
 - Motor, [28](#)
- showPackage
 - UARTCom.cpp, [90](#)
- speed
 - main.cpp, [94](#)
- SPEED_THRESHOLD
 - constants.h, [68](#)
- speedPPS
 - ISRVars, [12](#)
- SPEEDPPS2SPEEDRPM
 - MotorWheel.h, [76](#)
- src/communication/UARTCom.cpp, [82](#)
- src/communication/UCommands.cpp, [91](#)
- src/communication/UValue.cpp, [92](#)
- src/main.cpp, [92](#)
- src/misc/Utils.cpp, [95](#)
- START_BYTE
 - constants.h, [68](#)
- START_BYTE_POS
 - constants.h, [68](#)
- TAUD
 - MotorWheel.h, [76](#)
- TAUI
 - MotorWheel.h, [76](#)
- TIMESTAMP_POS
 - constants.h, [68](#)
- TRIGGER
 - MotorWheel.h, [76](#)
- TX_328_PIN
 - constants.h, [68](#)
- U_Component
 - constants.h, [69](#)
- U_FrameType
 - constants.h, [69](#)
- U_Request
 - constants.h, [69](#)
- UART_BAUD_RATE
 - constants.h, [69](#)
- UART_TRANSMIT_MS
 - constants.h, [69](#)
- UARTCom, [46](#)
 - data, [48](#)
 - init, [46](#)
 - read, [47](#)
 - send, [47](#)
- UARTCom.cpp
 - clearPackage, [83](#)
 - computeChecksum, [84](#)
 - correctChecksum, [84](#)
 - currentParseIndex, [90](#)
 - currentTimestamp, [90](#)
 - fromComponent, [85](#)
 - fromFrameType, [85](#)

- handlePackage, [86](#)
- handleRequest, [87](#)
- parseComponent, [87](#)
- parseFrameType, [88](#)
- parseRequest, [88](#)
- parseTimestamp, [89](#)
- payloadLength, [90](#)
- showPackage, [90](#)
- wheelLeft, [90](#)
- wheelRight, [91](#)
- uartTransmitTimer
 - main.cpp, [95](#)
- UCommands, [49](#)
 - setPIDParameter, [49](#)
 - setSetpoint, [50](#)
- UCommands.cpp
 - wheelLeft, [91](#)
 - wheelRight, [91](#)
- Utils, [51](#)
 - bytesToFloat, [51](#)
 - bytesToInt, [52](#)
- UValue, [53](#)
 - sendBothEncoderValues, [53](#)
 - sendEncoderValue, [54](#)
- VALUE
 - constants.h, [70](#)
- VEL2TORQUE_RATIO
 - constants.h, [69](#)
- VEL_UPDATE_RATE_MS
 - constants.h, [69](#)
- wheelLeft
 - main.cpp, [95](#)
 - UARTCom.cpp, [90](#)
 - UCommands.cpp, [91](#)
- wheelRight
 - main.cpp, [95](#)
 - UARTCom.cpp, [91](#)
 - UCommands.cpp, [91](#)