Daimtronics

Generated by Doxygen 1.8.11

Contents

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Namespace Documentation

5.1 ros Namespace Reference

Namespaces

- message_operations
- message_traits
- serialization

5.2 ros::message_operations Namespace Reference

Classes

- struct Printer< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >
- struct Printer< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >

5.3 ros::message_traits Namespace Reference

Classes

- struct DataType< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >
- struct DataType< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >
- struct Definition< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >
- struct Definition< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >
- struct HasHeader< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >
- struct HasHeader< ::semi_truck::Teensy_Actuators_< ContainerAllocator > const >
- struct HasHeader< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >
- struct HasHeader< ::semi_truck::Teensy_Sensors_< ContainerAllocator > const >
- struct IsFixedSize< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >
- struct IsFixedSize< ::semi_truck::Teensy_Actuators_< ContainerAllocator > const >
- struct IsFixedSize< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >
- struct IsFixedSize< ::semi_truck::Teensy_Sensors_< ContainerAllocator > const >
- struct IsMessage< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >
- struct IsMessage< ::semi_truck::Teensy_Actuators_< ContainerAllocator > const >
- struct IsMessage< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >
- struct IsMessage< ::semi_truck::Teensy_Sensors_< ContainerAllocator > const >
- struct MD5Sum< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >
- struct MD5Sum< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >

5.4 ros::serialization Namespace Reference

Classes

- struct Serializer< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >
- struct Serializer< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >

5.5 semi_truck Namespace Reference

Classes

- struct Teensy_Actuators_
- struct Teensy_Sensors_

Typedefs

- typedef ::semi_truck::Teensy_Actuators_< std::allocator< void > > Teensy_Actuators
- typedef boost::shared_ptr< ::semi_truck::Teensy_Actuators > Teensy_ActuatorsPtr
- typedef boost::shared_ptr< ::semi_truck::Teensy_Actuators const > Teensy_ActuatorsConstPtr
- typedef ::semi_truck::Teensy_Sensors_< std::allocator< void > > Teensy_Sensors
- typedef boost::shared_ptr< ::semi_truck::Teensy_Sensors > Teensy_SensorsPtr
- typedef boost::shared_ptr< ::semi_truck::Teensy_Sensors const > Teensy_SensorsConstPtr

Functions

- template<typename ContainerAllocator >
  std::ostream & operator<< (std::ostream &s, const ::semi_truck::Teensy_Actuators_< ContainerAllocator > &v)
- template<typename ContainerAllocator >
  std::ostream & operator<< (std::ostream &s, const ::semi_truck::Teensy_Sensors_< ContainerAllocator > &v)

5.5.1 Typedef Documentation

5.5.1.1 typedef ::semi_truck::Teensy_Actuators_<std::allocator<void> > semi_truck::Teensy_Actuators

5.5.1.2 typedef boost::shared_ptr< ::semi_truck::Teensy_Actuators const> semi_truck::Teensy_Actuators↩ ConstPtr

5.5.1.3 typedef boost::shared_ptr< ::semi_truck::Teensy_Actuators > semi_truck::Teensy_ActuatorsPtr

5.5.1.4 typedef ::semi_truck::Teensy_Sensors_<std::allocator<void> > semi_truck::Teensy_Sensors

5.5.1.5 typedef boost::shared_ptr< ::semi_truck::Teensy_Sensors const> semi_truck::Teensy_SensorsConstPtr

5.5.1.6 typedef boost::shared_ptr< ::semi_truck::Teensy_Sensors > semi_truck::Teensy_SensorsPtr

5.5.2 Function Documentation

5.5.2.1 template<typename ContainerAllocator > std::ostream& semi_truck::operator<< (std::ostream & *s*, const ::semi_truck::Teensy_Actuators_< ContainerAllocator > & *v*)

5.5.2.2 template<typename ContainerAllocator > std::ostream& semi_truck::operator<< (std::ostream & *s*, const ::semi_truck::Teensy_Sensors_< ContainerAllocator > & *v*)

Chapter 6

Class Documentation

6.1 actuator_data_t Struct Reference

`#include <system_data.h>`

Public Attributes

- int16_t motor_output
- int16_t steer_output
- int16_t fifth_output

6.1.1 Member Data Documentation

6.1.1.1 int16_t actuator_data_t::fifth_output

6.1.1.2 int16_t actuator_data_t::motor_output

6.1.1.3 int16_t actuator_data_t::steer_output

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

- daimtronics/teensy_chibios/src/main/include/system_data.h

6.2 ros::message_traits::DataType< ::semi_truck::Teensy_Actuators_< ContainerAllocator > > Struct Template Reference

`#include <Teensy_Actuators.h>`

Static Public Member Functions

- static const char * value ()
- static const char * value (const ::semi_truck::Teensy_Actuators_< ContainerAllocator > &)

6.2.1 Member Function Documentation

6.2.1.1 template<class ContainerAllocator > static const char* ros::message_traits::DataType< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >::value ( ) [inline], [static]

6.2.1.2 template<class ContainerAllocator > static const char* ros::message_traits::DataType< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >::value (const ::semi_truck::Teensy_Actuators_< ContainerAllocator > &) [inline], [static]

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

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Actuators.h

6.3 ros::message_traits::DataType< ::semi_truck::Teensy_Sensors_< ContainerAllocator > > Struct Template Reference

`#include <Teensy_Sensors.h>`

Static Public Member Functions

- static const char * value ()
- static const char * value (const ::semi_truck::Teensy_Sensors_< ContainerAllocator > &)

6.3.1 Member Function Documentation

6.3.1.1 template<class ContainerAllocator > static const char* ros::message_traits::DataType< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >::value ( ) [inline], [static]

6.3.1.2 template<class ContainerAllocator > static const char* ros::message_traits::DataType< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >::value (const ::semi_truck::Teensy_Sensors_< ContainerAllocator > &) [inline], [static]

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

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Sensors.h

6.4 ros::message_traits::Definition< ::semi_truck::Teensy_Actuators_< ContainerAllocator > > Struct Template Reference

`#include <Teensy_Actuators.h>`

Static Public Member Functions

- static const char * value ()
- static const char * value (const ::semi_truck::Teensy_Actuators_< ContainerAllocator > &)

6.4.1 Member Function Documentation

6.4.1.1 template<class ContainerAllocator > static const char* ros::message_traits::Definition< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >::value ( ) [inline], [static]

6.4.1.2 template<class ContainerAllocator > static const char* ros::message_traits::Definition< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >::value (const ::semi_truck::Teensy_Actuators_< ContainerAllocator > &) [inline], [static]

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

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Actuators.h

6.5 ros::message_traits::Definition< ::semi_truck::Teensy_Sensors_< ContainerAllocator > > Struct Template Reference

`#include <Teensy_Sensors.h>`

Static Public Member Functions

- static const char * value ()
- static const char * value (const ::semi_truck::Teensy_Sensors_< ContainerAllocator > &)

6.5.1 Member Function Documentation

6.5.1.1 template<class ContainerAllocator > static const char* ros::message_traits::Definition< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >::value ( ) [inline], [static]

6.5.1.2 template<class ContainerAllocator > static const char* ros::message_traits::Definition< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >::value (const ::semi_truck::Teensy_Sensors_< ContainerAllocator > &) [inline], [static]

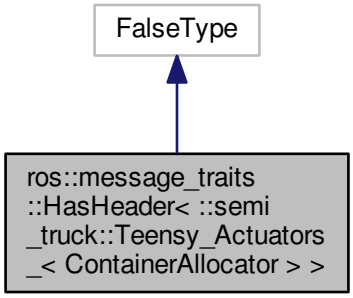
The documentation for this struct was generated from the following file:

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Sensors.h

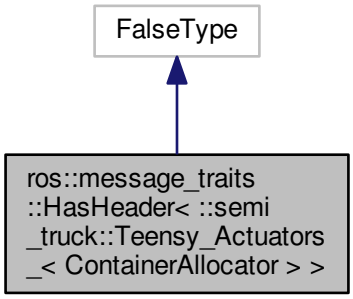
6.6 ros::message_traits::HasHeader< ::semi_truck::Teensy_Actuators_< ContainerAllocator > > Struct Template Reference

`#include <Teensy_Actuators.h>`

Inheritance diagram for ros::message_traits::HasHeader< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >:



Collaboration diagram for ros::message_traits::HasHeader< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >:



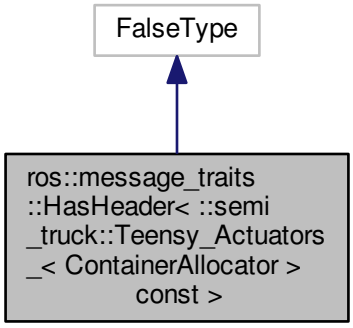
The documentation for this struct was generated from the following file:

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Actuators.h

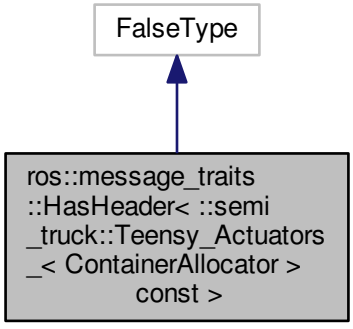
6.7 ros::message_traits::HasHeader< ::semi_truck::Teensy_Actuators_< ContainerAllocator > const > Struct Template Reference

`#include <Teensy_Actuators.h>`

Inheritance diagram for ros::message_traits::HasHeader< ::semi_truck::Teensy_Actuators_< ContainerAllocator > const >:



Collaboration diagram for ros::message_traits::HasHeader< ::semi_truck::Teensy_Actuators_< ContainerAllocator > const >:



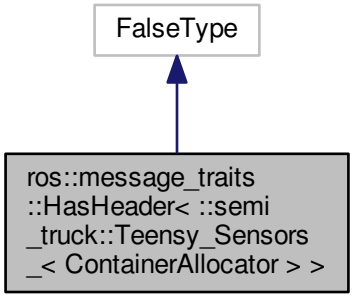
The documentation for this struct was generated from the following file:

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Actuators.h

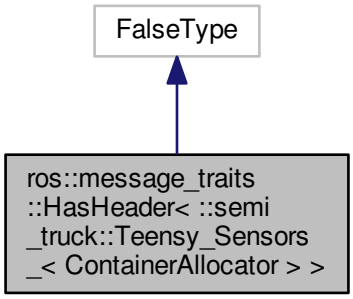
6.8 ros::message_traits::HasHeader< ::semi_truck::Teensy_Sensors_< ContainerAllocator > > Struct Template Reference

`#include <Teensy_Sensors.h>`

Inheritance diagram for ros::message_traits::HasHeader< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >:



Collaboration diagram for ros::message_traits::HasHeader< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >:



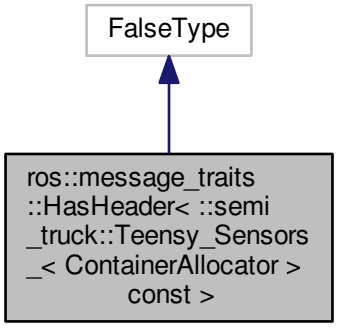
The documentation for this struct was generated from the following file:

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Sensors.h

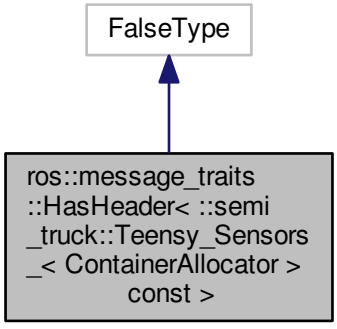
6.9 ros::message_traits::HasHeader< ::semi_truck::Teensy_Sensors_< ContainerAllocator > const > Struct Template Reference

`#include <Teensy_Sensors.h>`

Inheritance diagram for ros::message_traits::HasHeader< ::semi_truck::Teensy_Sensors_< ContainerAllocator > const >:



Collaboration diagram for ros::message_traits::HasHeader< ::semi_truck::Teensy_Sensors_< ContainerAllocator > const >:



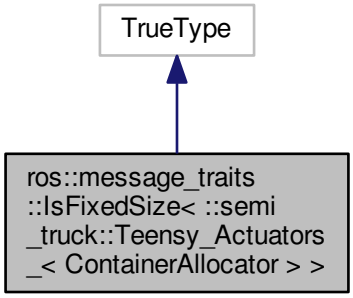
The documentation for this struct was generated from the following file:

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Sensors.h

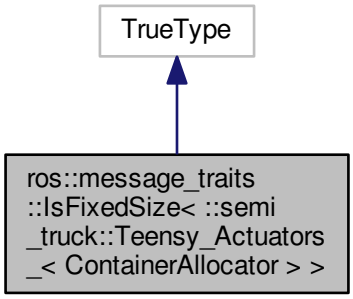
6.10 ros::message_traits::IsFixedSize< ::semi_truck::Teensy_Actuators_< ContainerAllocator > > Struct Template Reference

`#include <Teensy_Actuators.h>`

Inheritance diagram for ros::message_traits::IsFixedSize< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >:



Collaboration diagram for ros::message_traits::IsFixedSize< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >:



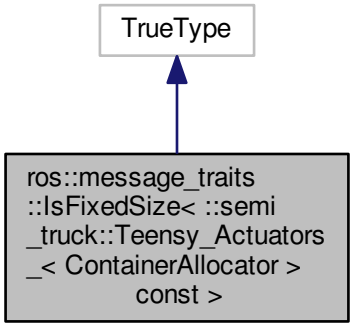
The documentation for this struct was generated from the following file:

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Actuators.h

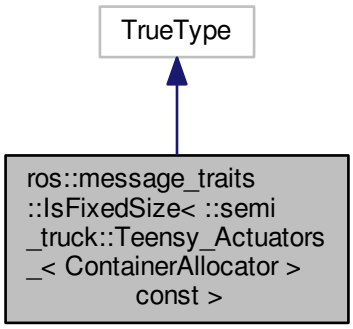
6.11 ros::message_traits::IsFixedSize< ::semi_truck::Teensy_Actuators_< ContainerAllocator > const > Struct Template Reference

`#include <Teensy_Actuators.h>`

Inheritance diagram for ros::message_traits::IsFixedSize< ::semi_truck::Teensy_Actuators_< ContainerAllocator > const >:



Collaboration diagram for ros::message_traits::IsFixedSize< ::semi_truck::Teensy_Actuators_< ContainerAllocator > const >:



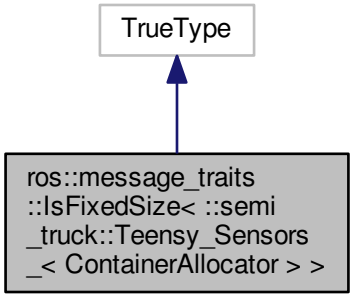
The documentation for this struct was generated from the following file:

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Actuators.h

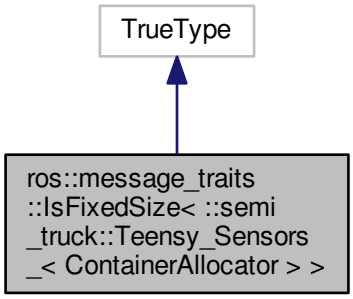
6.12 ros::message_traits::IsFixedSize< ::semi_truck::Teensy_Sensors_< ContainerAllocator > > Struct Template Reference

`#include <Teensy_Sensors.h>`

Inheritance diagram for ros::message_traits::IsFixedSize< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >:



Collaboration diagram for ros::message_traits::IsFixedSize< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >:



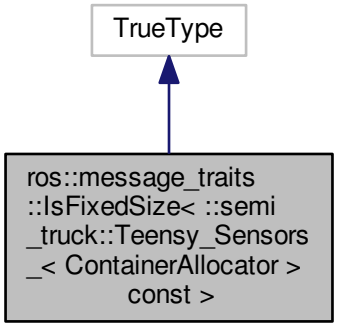
The documentation for this struct was generated from the following file:

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Sensors.h

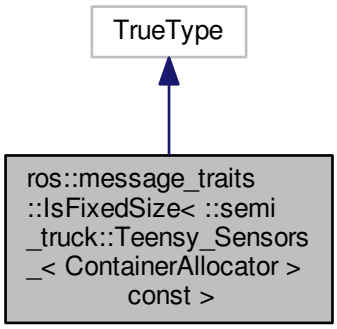
6.13 ros::message_traits::IsFixedSize< ::semi_truck::Teensy_Sensors_< ContainerAllocator > const > Struct Template Reference

`#include <Teensy_Sensors.h>`

Inheritance diagram for ros::message_traits::IsFixedSize< ::semi_truck::Teensy_Sensors_< ContainerAllocator > const >:



Collaboration diagram for ros::message_traits::IsFixedSize< ::semi_truck::Teensy_Sensors_< ContainerAllocator > const >:



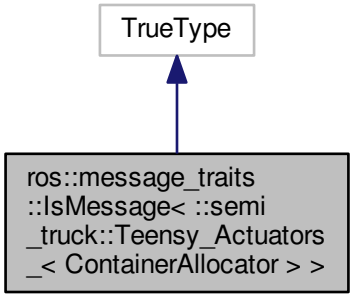
The documentation for this struct was generated from the following file:

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Sensors.h

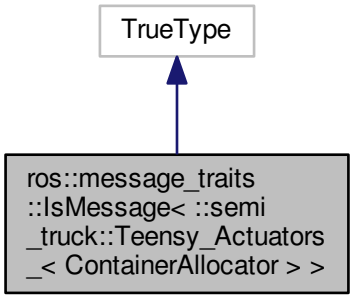
6.14 ros::message_traits::IsMessage< ::semi_truck::Teensy_Actuators_< ContainerAllocator > > Struct Template Reference

`#include <Teensy_Actuators.h>`

Inheritance diagram for ros::message_traits::IsMessage< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >:



Collaboration diagram for ros::message_traits::IsMessage< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >:



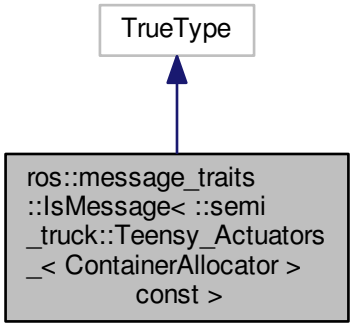
The documentation for this struct was generated from the following file:

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Actuators.h

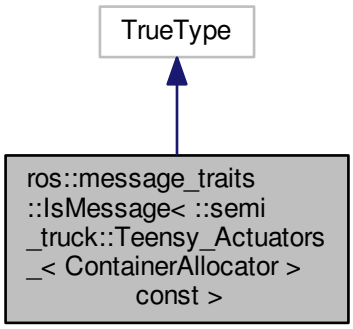
6.15 ros::message_traits::IsMessage< ::semi_truck::Teensy_Actuators_< ContainerAllocator > const > Struct Template Reference

```
#include <Teensy_Actuators.h>
```

Inheritance diagram for ros::message_traits::IsMessage< ::semi_truck::Teensy_Actuators_< ContainerAllocator > const >:



Collaboration diagram for ros::message_traits::IsMessage< ::semi_truck::Teensy_Actuators_< ContainerAllocator > const >:



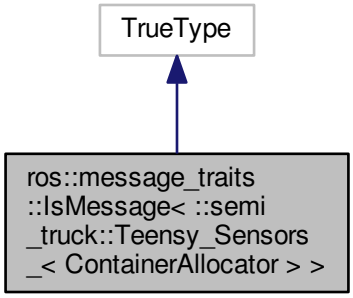
The documentation for this struct was generated from the following file:

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Actuators.h

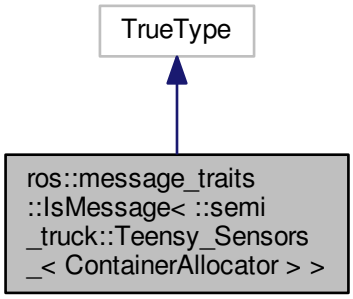
6.16 ros::message_traits::IsMessage< ::semi_truck::Teensy_Sensors_< ContainerAllocator > > Struct Template Reference

`#include <Teensy_Sensors.h>`

Inheritance diagram for ros::message_traits::IsMessage< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >:



Collaboration diagram for ros::message_traits::IsMessage< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >:



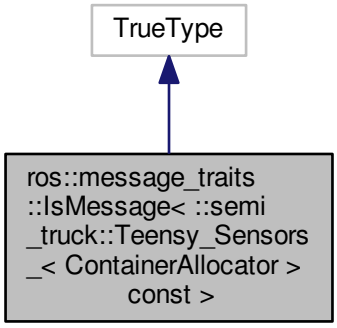
The documentation for this struct was generated from the following file:

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Sensors.h

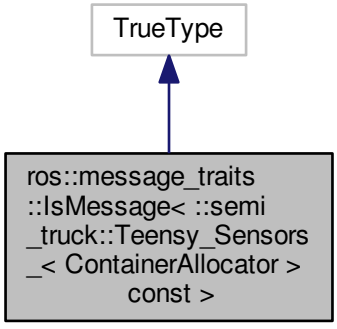
6.17 ros::message_traits::IsMessage< ::semi_truck::Teensy_Sensors_< ContainerAllocator > const > Struct Template Reference

`#include <Teensy_Sensors.h>`

Inheritance diagram for ros::message_traits::IsMessage< ::semi_truck::Teensy_Sensors_< ContainerAllocator > const >:



Collaboration diagram for ros::message_traits::IsMessage< ::semi_truck::Teensy_Sensors_< ContainerAllocator > const >:



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

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Sensors.h

6.18 ros::message_traits::MD5Sum< ::semi_truck::Teensy_Actuators_< ContainerAllocator > > Struct Template Reference

`#include <Teensy_Actuators.h>`

Static Public Member Functions

- static const char * value ()
- static const char * value (const ::semi_truck::Teensy_Actuators_< ContainerAllocator > &)

Static Public Attributes

- static const uint64_t static_value1 = 0x0d131da7355e429dULL
- static const uint64_t static_value2 = 0x9d8b9cc6b2375149ULL

6.18.1 Member Function Documentation

6.18.1.1 template<class ContainerAllocator > static const char* ros::message_traits::MD5Sum< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >::value ( ) [inline], [static]

6.18.1.2 template<class ContainerAllocator > static const char* ros::message_traits::MD5Sum< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >::value (const ::semi_truck::Teensy_Actuators_< ContainerAllocator > &) [inline], [static]

6.18.2 Member Data Documentation

6.18.2.1 template<class ContainerAllocator > const uint64_t ros::message_traits::MD5Sum< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >::static_value1 = 0x0d131da7355e429dULL [static]

6.18.2.2 template<class ContainerAllocator > const uint64_t ros::message_traits::MD5Sum< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >::static_value2 = 0x9d8b9cc6b2375149ULL [static]

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

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Actuators.h

6.19 ros::message_traits::MD5Sum< ::semi_truck::Teensy_Sensors_< ContainerAllocator > > Struct Template Reference

`#include <Teensy_Sensors.h>`

Static Public Member Functions

- static const char * value ()
- static const char * value (const ::semi_truck::Teensy_Sensors_< ContainerAllocator > &)

Static Public Attributes

- static const uint64_t static_value1 = 0x9623202c8fe03b3aULL
- static const uint64_t static_value2 = 0xc3d8a917b632561eULL

6.19.1 Member Function Documentation

6.19.1.1 template<class ContainerAllocator > static const char* ros::message_traits::MD5Sum< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >::value ( ) [inline], [static]

6.19.1.2 template<class ContainerAllocator > static const char* ros::message_traits::MD5Sum< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >::value (const ::semi_truck::Teensy_Sensors_< ContainerAllocator > &) [inline], [static]

6.19.2 Member Data Documentation

6.19.2.1 template<class ContainerAllocator > const uint64_t ros::message_traits::MD5Sum< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >::static_value1 = 0x9623202c8fe03b3aULL [static]

6.19.2.2 template<class ContainerAllocator > const uint64_t ros::message_traits::MD5Sum< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >::static_value2 = 0xc3d8a917b632561eULL [static]

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

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Sensors.h

6.20 ros::message_operations::Printer< ::semi_truck::Teensy_Actuators_< ContainerAllocator > > Struct Template Reference

`#include <Teensy_Actuators.h>`

Static Public Member Functions

- template<typename Stream >
  static void stream (Stream &s, const std::string &indent, const ::semi_truck::Teensy_Actuators_< ContainerAllocator > &v)

6.20.1 Member Function Documentation

6.20.1.1 template<class ContainerAllocator > template<typename Stream > static void ros::message_operations::Printer< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >::stream (Stream & *s*, const std::string & *indent*, const ::semi_truck::Teensy_Actuators_< ContainerAllocator > & *v*) [inline], [static]

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

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Actuators.h

6.21 ros::message_operations::Printer< ::semi_truck::Teensy_Sensors_< ContainerAllocator > > Struct Template Reference

`#include <Teensy_Sensors.h>`

Static Public Member Functions

- template<typename Stream >
  static void stream (Stream &s, const std::string &indent, const ::semi_truck::Teensy_Sensors_< ContainerAllocator > &v)

6.21.1 Member Function Documentation

6.21.1.1 template<class ContainerAllocator > template<typename Stream > static void ros::message_operations::Printer< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >::stream (Stream & *s*, const std::string & *indent*, const ::semi_truck::Teensy_Sensors_< ContainerAllocator > & *v*) [inline], [static]

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

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Sensors.h

6.22 sensor_data_t Struct Reference

`#include <system_data.h>`

Public Attributes

- int16_t imu_angle
- int16_t wheel_speed
- int16_t right_TOF
- int16_t left_TOF
- int16_t rear_TOF

6.22.1 Member Data Documentation

6.22.1.1 int16_t sensor_data_t::imu_angle

6.22.1.2 int16_t sensor_data_t::left_TOF

6.22.1.3 int16_t sensor_data_t::rear_TOF

6.22.1.4 int16_t sensor_data_t::right_TOF

6.22.1.5 int16_t sensor_data_t::wheel_speed

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

- daimtronics/teensy_chibios/src/main/include/system_data.h

6.23 ros::serialization::Serializer< ::semi_truck::Teensy_Actuators_< ContainerAllocator > > Struct Template Reference

```
#include <Teensy_Actuators.h>
```

Static Public Member Functions

- template<typename Stream , typename T >
  static void allInOne (Stream &stream, T m)

6.23.1 Member Function Documentation

6.23.1.1 template<class ContainerAllocator > template<typename Stream , typename T > static void ros::serialization::Serializer< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >::allInOne (Stream & *stream,* T *m*) [inline], [static]

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

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Actuators.h

6.24 ros::serialization::Serializer< ::semi_truck::Teensy_Sensors_< ContainerAllocator > > Struct Template Reference

```
#include <Teensy_Sensors.h>
```

Static Public Member Functions

- template<typename Stream , typename T >
  static void allInOne (Stream &stream, T m)

6.24.1 Member Function Documentation

6.24.1.1 template<class ContainerAllocator > template<typename Stream , typename T > static void ros::serialization::Serializer< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >::allInOne (Stream & *stream*, T *m*) [inline], [static]

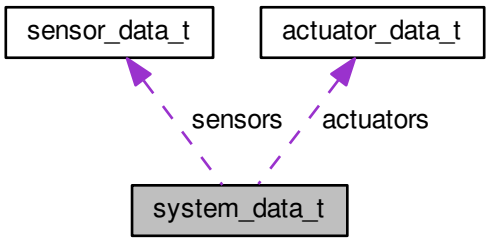
The documentation for this struct was generated from the following file:

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Sensors.h

6.25 system_data_t Struct Reference

`#include <system_data.h>`

Collaboration diagram for system_data_t:



Public Attributes

- int16_t wheel_speed
- int16_t imu_angle
- uint16_t right_TOF
- uint16_t left_TOF
- uint16_t rear_TOF
- uint16_t drive_mode_1
- uint16_t drive_mode_2
- int16_t motor_output
- int16_t steer_output
- uint16_t fifth_output
- bool updated
- int16_t deadman
- int16_t drive_mode
- sensor_data_t sensors
- actuator_data_t actuators

6.25.1 Member Data Documentation

6.25.1.1 actuator_data_t system_data_t::actuators

6.25.1.2 int16_t system_data_t::deadman

6.25.1.3 int16_t system_data_t::drive_mode

6.25.1.4 uint16_t system_data_t::drive_mode_1

6.25.1.5 uint16_t system_data_t::drive_mode_2

6.25.1.6 uint16_t system_data_t::fifth_output

6.25.1.7 int16_t system_data_t::imu_angle

6.25.1.8 uint16_t system_data_t::left_TOF

6.25.1.9 int16_t system_data_t::motor_output

6.25.1.10 uint16_t system_data_t::rear_TOF

6.25.1.11 uint16_t system_data_t::right_TOF

6.25.1.12 sensor_data_t system_data_t::sensors

6.25.1.13 int16_t system_data_t::steer_output

6.25.1.14 bool system_data_t::updated

6.25.1.15 int16_t system_data_t::wheel_speed

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

- daimtronics/semi_catkin_ws/src/semi_truck/include/system_data.h

6.26 semi_truck::Teensy_Actuators_< ContainerAllocator > Struct Template Reference

`#include <Teensy_Actuators.h>`

Public Types

- typedef Teensy_Actuators_< ContainerAllocator > Type
- typedef int16_t _motor_output_type
- typedef int16_t _steer_output_type
- typedef int16_t _fifth_output_type
- typedef boost::shared_ptr< ::semi_truck::Teensy_Actuators_< ContainerAllocator > > Ptr
- typedef boost::shared_ptr< ::semi_truck::Teensy_Actuators_< ContainerAllocator > const > ConstPtr

Public Member Functions

- Teensy_Actuators_ ()
- Teensy_Actuators_ (const ContainerAllocator &_alloc)

Public Attributes

- _motor_output_type motor_output
- _steer_output_type steer_output
- _fifth_output_type fifth_output

6.26.1 Member Typedef Documentation

6.26.1.1 template<class ContainerAllocator > typedef int16_t semi_truck::Teensy_Actuators_< ContainerAllocator >::_fifth_output_type

6.26.1.2 template<class ContainerAllocator > typedef int16_t semi_truck::Teensy_Actuators_< ContainerAllocator >::_motor_output_type

6.26.1.3 template<class ContainerAllocator > typedef int16_t semi_truck::Teensy_Actuators_< ContainerAllocator >::_steer_output_type

6.26.1.4 template<class ContainerAllocator > typedef boost::shared_ptr< ::semi_truck::Teensy_Actuators_<ContainerAllocator> const> semi_truck::Teensy_Actuators_< ContainerAllocator >::ConstPtr

6.26.1.5 template<class ContainerAllocator > typedef boost::shared_ptr< ::semi_truck::Teensy_Actuators_<ContainerAllocator> > semi_truck::Teensy_Actuators_< ContainerAllocator >::Ptr

6.26.1.6 template<class ContainerAllocator > typedef Teensy_Actuators_<ContainerAllocator> semi_truck::Teensy_Actuators_< ContainerAllocator >::Type

6.26.2 Constructor & Destructor Documentation

6.26.2.1 template<class ContainerAllocator > semi_truck::Teensy_Actuators_< ContainerAllocator >::Teensy_Actuators_ ( ) [inline]

6.26.2.2 template<class ContainerAllocator > semi_truck::Teensy_Actuators_< ContainerAllocator >::Teensy_Actuators_ (const ContainerAllocator & *_alloc*) [inline]

6.26.3 Member Data Documentation

6.26.3.1 template<class ContainerAllocator > _fifth_output_type semi_truck::Teensy_Actuators_< ContainerAllocator >::fifth_output

6.26.3.2 template<class ContainerAllocator > _motor_output_type semi_truck::Teensy_Actuators_< ContainerAllocator >::motor_output

6.26.3.3 template<class ContainerAllocator > _steer_output_type semi_truck::Teensy_Actuators_< ContainerAllocator >::steer_output

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

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Actuators.h

6.27 semi_truck::Teensy_Sensors_< ContainerAllocator > Struct Template Reference

`#include <Teensy_Sensors.h>`

Public Types

- typedef Teensy_Sensors_< ContainerAllocator > Type
- typedef int16_t _wheel_speed_type
- typedef int16_t _imu_angle_type
- typedef int16_t _right_TOF_type
- typedef int16_t _left_TOF_type
- typedef int16_t _rear_TOF_type
- typedef int16_t _drive_mode_1_type
- typedef int16_t _drive_mode_2_type
- typedef boost::shared_ptr< ::semi_truck::Teensy_Sensors_< ContainerAllocator > > Ptr
- typedef boost::shared_ptr< ::semi_truck::Teensy_Sensors_< ContainerAllocator > const > ConstPtr

Public Member Functions

- Teensy_Sensors_ ()
- Teensy_Sensors_ (const ContainerAllocator &_alloc)

Public Attributes

- _wheel_speed_type wheel_speed
- _imu_angle_type imu_angle
- _right_TOF_type right_TOF
- _left_TOF_type left_TOF
- _rear_TOF_type rear_TOF
- _drive_mode_1_type drive_mode_1
- _drive_mode_2_type drive_mode_2

6.27.1 Member Typedef Documentation

6.27.1.1 template<class ContainerAllocator > typedef int16_t semi_truck::Teensy_Sensors_< ContainerAllocator >::_drive_mode_1_type

6.27.1.2 template<class ContainerAllocator > typedef int16_t semi_truck::Teensy_Sensors_< ContainerAllocator >::_drive_mode_2_type

6.27.1.3 template<class ContainerAllocator > typedef int16_t semi_truck::Teensy_Sensors_< ContainerAllocator >::_imu_angle_type

6.27.1.4 template<class ContainerAllocator > typedef int16_t semi_truck::Teensy_Sensors_< ContainerAllocator >::_left_TOF_type

6.27.1.5 template<class ContainerAllocator > typedef int16_t semi_truck::Teensy_Sensors_< ContainerAllocator >::_rear_TOF_type

6.27.1.6 template<class ContainerAllocator > typedef int16_t semi_truck::Teensy_Sensors_< ContainerAllocator >::_right_TOF_type

6.27.1.7 template<class ContainerAllocator > typedef int16_t semi_truck::Teensy_Sensors_< ContainerAllocator >::_wheel_speed_type

6.27.1.8 template<class ContainerAllocator > typedef boost::shared_ptr< ::semi_truck::Teensy_Sensors_<ContainerAllocator> const> semi_truck::Teensy_Sensors_< ContainerAllocator >::ConstPtr

6.27.1.9 template<class ContainerAllocator > typedef boost::shared_ptr< ::semi_truck::Teensy_Sensors_<ContainerAllocator> > semi_truck::Teensy_Sensors_< ContainerAllocator >::Ptr

6.27.1.10 template<class ContainerAllocator > typedef Teensy_Sensors_<ContainerAllocator> semi_truck::Teensy_Sensors_< ContainerAllocator >::Type

6.27.2 Constructor & Destructor Documentation

6.27.2.1 template<class ContainerAllocator > semi_truck::Teensy_Sensors_< ContainerAllocator >::Teensy_Sensors_ ( ) [inline]

6.27.2.2 template<class ContainerAllocator > semi_truck::Teensy_Sensors_< ContainerAllocator >::Teensy_Sensors_ (const ContainerAllocator & *_alloc*) [inline]

6.27.3 Member Data Documentation

6.27.3.1 template<class ContainerAllocator > _drive_mode_1_type semi_truck::Teensy_Sensors_< ContainerAllocator >::drive_mode_1

6.27.3.2 template<class ContainerAllocator > _drive_mode_2_type semi_truck::Teensy_Sensors_< ContainerAllocator >::drive_mode_2

6.27.3.3 template<class ContainerAllocator > _imu_angle_type semi_truck::Teensy_Sensors_< ContainerAllocator >::imu_angle

6.27.3.4 template<class ContainerAllocator > _left_TOF_type semi_truck::Teensy_Sensors_< ContainerAllocator >::left_TOF

6.27.3.5 template<class ContainerAllocator > _rear_TOF_type semi_truck::Teensy_Sensors_< ContainerAllocator >::rear_TOF

6.27.3.6 template<class ContainerAllocator > _right_TOF_type semi_truck::Teensy_Sensors_< ContainerAllocator >::right_TOF

6.27.3.7 template<class ContainerAllocator > _wheel_speed_type semi_truck::Teensy_Sensors_< ContainerAllocator >::wheel_speed

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

- daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Sensors.h

Chapter 7

File Documentation

7.1 daimtronics/semi_catkin_ws/src/CMakeLists.txt File Reference

Functions

- cmake_minimum_required (VERSION 2.8.3) set(CATKIN_TOPLEVEL TRUE) set(_cmd"catkin_find_↩ pkg""catkin""$
- execute_process (COMMAND ${_cmd}RESULT_VARIABLE _res OUTPUT_VARIABLE _out ERROR_↩ VARIABLE _err OUTPUT_STRIP_TRAILING_WHITESPACE ERROR_STRIP_TRAILING_WHITESPACE) if(NOT _res EQUAL 0 AND NOT _res EQUAL 2) string(REPLACE"

Variables

- _cmd_str

7.1.1 Function Documentation

7.1.1.1 cmake_minimum_required (VERSION 2.8. *3*)

7.1.1.2 execute_process (COMMAND ${_cmd}RESULT_VARIABLE _res OUTPUT_VARIABLE _out ERROR_VARIABLE _err OUTPUT_STRIP_TRAILING_WHITESPACE *ERROR_STRIP_TRAILING_WHITESPACE*)

7.1.2 Variable Documentation

7.1.2.1 _cmd_str

Initial value:

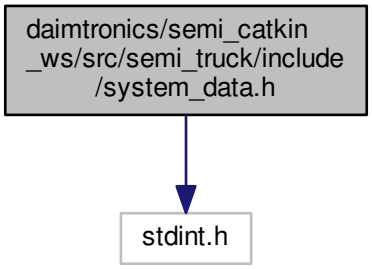
```
{_cmd}")
  message(FATAL_ERROR "Search for 'catkin' in workspace failed (${_cmd_str}): ${_err}")
endif()


if(_res EQUAL 0)
  set(catkin_EXTRAS_DIR "${CMAKE_SOURCE_DIR}/${_out}/cmake")

  include(${catkin_EXTRAS_DIR}/all.cmake NO_POLICY_SCOPE)
  add_subdirectory("${_out}")

else()


  if(NOT DEFINED CMAKE_PREFIX_PATH)
    if(NOT "$ENV{CMAKE_PREFIX_PATH}" STREQUAL "")
      string(REPLACE ":" "
```

7.2 daimtronics/semi_catkin_ws/src/semi_truck/CMakeLists.txt File Reference

7.3 daimtronics/semi_catkin_ws/src/semi_truck/include/system_data.h File Reference

```
#include <stdint.h>
```

Include dependency graph for system_data.h:



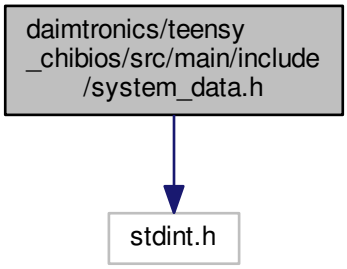
Classes

- struct system_data_t

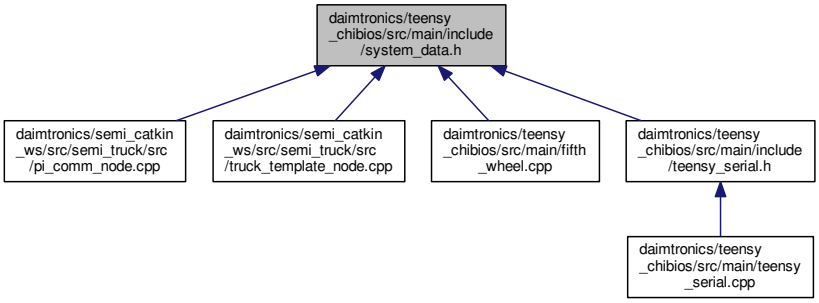
7.4 daimtronics/teensy_chibios/src/main/include/system_data.h File Reference

```
#include <stdint.h>
```

Include dependency graph for system_data.h:



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



Classes

- struct sensor_data_t
- struct actuator_data_t
- struct system_data_t

Typedefs

- typedef struct sensor_data_t sensor_data_t
- typedef struct actuator_data_t actuator_data_t
- typedef struct system_data_t system_data_t

7.4.1 Typedef Documentation

7.4.1.1 typedef struct actuator_data_t actuator_data_t

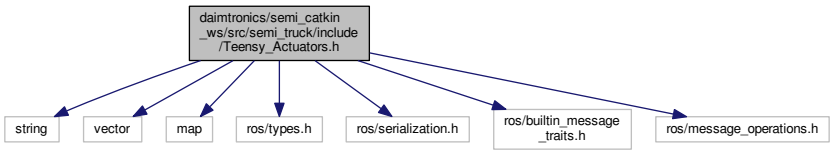
7.4.1.2 typedef struct sensor_data_t sensor_data_t

7.4.1.3 typedef struct system_data_t system_data_t

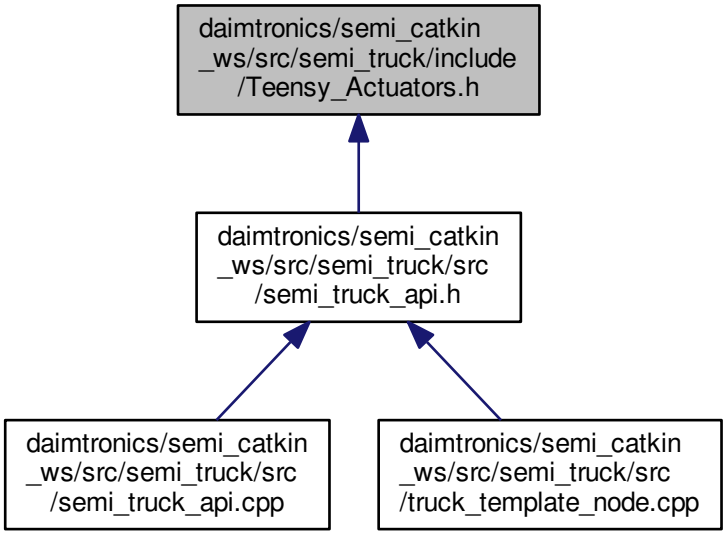
7.5 daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Actuators.h File Reference

```
#include <string>
#include <vector>
#include <map>
#include <ros/types.h>
#include <ros/serialization.h>
#include <ros/builtin_message_traits.h>
#include <ros/message_operations.h>
```

Include dependency graph for Teensy_Actuators.h:



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



Classes

- struct semi_truck::Teensy_Actuators_< ContainerAllocator >
- struct ros::message_traits::IsFixedSize< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >
- struct ros::message_traits::IsFixedSize< ::semi_truck::Teensy_Actuators_< ContainerAllocator > const >
- struct ros::message_traits::IsMessage< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >
- struct ros::message_traits::IsMessage< ::semi_truck::Teensy_Actuators_< ContainerAllocator > const >
- struct ros::message_traits::HasHeader< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >
- struct ros::message_traits::HasHeader< ::semi_truck::Teensy_Actuators_< ContainerAllocator > const >
- struct ros::message_traits::MD5Sum< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >
- struct ros::message_traits::DataType< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >
- struct ros::message_traits::Definition< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >
- struct ros::serialization::Serializer< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >
- struct ros::message_operations::Printer< ::semi_truck::Teensy_Actuators_< ContainerAllocator > >

Namespaces

- semi_truck
- ros
- ros::message_traits
- ros::serialization
- ros::message_operations

Typedefs

- typedef ::semi_truck::Teensy_Actuators_< std::allocator< void > > semi_truck::Teensy_Actuators
- typedef boost::shared_ptr< ::semi_truck::Teensy_Actuators > semi_truck::Teensy_ActuatorsPtr
- typedef boost::shared_ptr< ::semi_truck::Teensy_Actuators const > semi_truck::Teensy_ActuatorsConstPtr

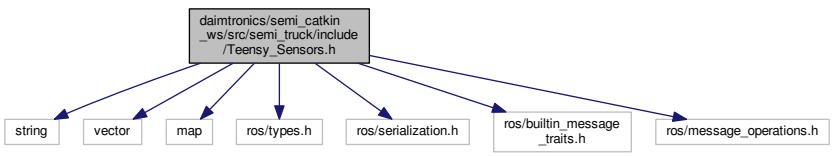
Functions

- template<typename ContainerAllocator >
  std::ostream & semi_truck::operator<< (std::ostream &s, const ::semi_truck::Teensy_Actuators_< ContainerAllocator > &v)

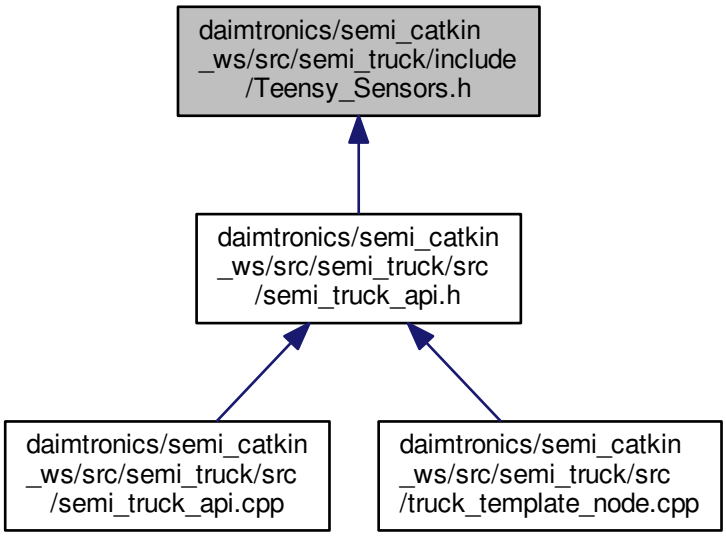
7.6 daimtronics/semi_catkin_ws/src/semi_truck/include/Teensy_Sensors.h File Reference

```
#include <string>
#include <vector>
#include <map>
#include <ros/types.h>
#include <ros/serialization.h>
#include <ros/builtin_message_traits.h>
#include <ros/message_operations.h>
```

Include dependency graph for Teensy_Sensors.h:



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



Classes

- struct semi_truck::Teensy_Sensors_< ContainerAllocator >
- struct ros::message_traits::IsFixedSize< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >
- struct ros::message_traits::IsFixedSize< ::semi_truck::Teensy_Sensors_< ContainerAllocator > const >
- struct ros::message_traits::IsMessage< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >
- struct ros::message_traits::IsMessage< ::semi_truck::Teensy_Sensors_< ContainerAllocator > const >
- struct ros::message_traits::HasHeader< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >
- struct ros::message_traits::HasHeader< ::semi_truck::Teensy_Sensors_< ContainerAllocator > const >
- struct ros::message_traits::MD5Sum< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >
- struct ros::message_traits::DataType< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >
- struct ros::message_traits::Definition< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >
- struct ros::serialization::Serializer< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >
- struct ros::message_operations::Printer< ::semi_truck::Teensy_Sensors_< ContainerAllocator > >

Namespaces

- semi_truck
- ros
- ros::message_traits
- ros::serialization
- ros::message_operations

Typedefs

- typedef ::semi_truck::Teensy_Sensors_< std::allocator< void > > semi_truck::Teensy_Sensors
- typedef boost::shared_ptr< ::semi_truck::Teensy_Sensors > semi_truck::Teensy_SensorsPtr
- typedef boost::shared_ptr< ::semi_truck::Teensy_Sensors const > semi_truck::Teensy_SensorsConstPtr

Functions

- template<typename ContainerAllocator >
  std::ostream & semi_truck::operator<< (std::ostream &s, const ::semi_truck::Teensy_Sensors_↩
  < ContainerAllocator > &v)

7.7 daimtronics/semi_catkin_ws/src/semi_truck/msg/Teensy_Actuators.msg File Reference

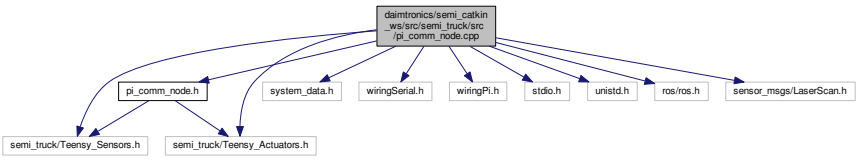
7.8 daimtronics/semi_catkin_ws/src/semi_truck/msg/Teensy_Sensors.msg File Reference

7.9 daimtronics/semi_catkin_ws/src/semi_truck/package.xml File Reference

7.10 daimtronics/semi_catkin_ws/src/semi_truck/src/pi_comm_node.cpp File Reference

```
#include "pi_comm_node.h"
#include "system_data.h"
#include "semi_truck/Teensy_Sensors.h"
#include "semi_truck/Teensy_Actuators.h"
#include <wiringSerial.h>
#include <wiringPi.h>
#include <stdio.h>
#include <unistd.h>
#include <ros/ros.h>
#include <sensor_msgs/LaserScan.h>
```

Include dependency graph for pi_comm_node.cpp:



Macros

- #define SHORT_SIZE 2
- #define RELAY_PIN_1 7
- #define RELAY_PIN_2 0
- #define SYNC_VALUE -32000
- #define SENSOR_DATA_SIZE_W_SYNC 14
- #define SENSOR_DATA_SIZE 12
- #define UART "/dev/ttyS0"
- #define BAUDRATE 9600
- #define LOOP_FREQUENCY 20

Functions

- int main (int argc, char **argv)

  The main function for the ROS node to communicate with the Teensy over UART. It receives sensor data from the Teensy and publishes this data to the teensy_sensor_data topic. It also subscribes to the teensy_actuator_data topic and writes the values it gets to the Teensy over UART.
- void pi_sync ()

  Called before reading sensor data from the Teensy. It will read the buffer until it encounters the SYNC_VALUE that has been defined. Once this happens, the next set of bytes are the set of sensor values.
- short read_sensor_msg (int serial, char num_bytes)

  Reads a single value from the UART communication buffer.
- void read_from_teensy (int serial, semi_truck::Teensy_Sensors &sensors)

  Reads an entire set of Teensy sensor data by calling read_sensor_msg on each sensor data in the UART message.
- void write_actuator_msg (int serial, short actuator_val, char num_bytes)

  Writes a single actuator value to the Teensy.
- void write_to_teensy (int serial, const semi_truck::Teensy_Actuators &actuators)

  Writes an entire set of actuator data to the Teensy via UART.
- void print_sensors (const semi_truck::Teensy_Sensors &sensors)

  A function useful for debugging serial communication. Prints the entire set of sensor data to the console.
- void print_actuators (const semi_truck::Teensy_Actuators &actuators)

  A function useful for debugging serial communication. Prints the entire set of actuator data to the console.
- void actuator_cb (const semi_truck::Teensy_Actuators &msg)

  The callback function the the subscriber to the teensy_actuator_data topic. This function will run every time ROS::spinOnce is called. It reads the set of data from the topic and immediately writes these values to the Teensy via UART.

7.10.1 Macro Definition Documentation

7.10.1.1 #define BAUDRATE 9600

7.10.1.2 #define LOOP_FREQUENCY 20

7.10.1.3 #define RELAY_PIN_1 7

7.10.1.4 #define RELAY_PIN_2 0

7.10.1.5 #define SENSOR_DATA_SIZE 12

7.10.1.6 #define SENSOR_DATA_SIZE_W_SYNC 14

7.10.1.7 #define SHORT_SIZE 2

7.10.1.8 #define SYNC_VALUE -32000

7.10.1.9 #define UART "/dev/ttyS0"

7.10.2 Function Documentation

7.10.2.1 void actuator_cb (const semi_truck::Teensy_Actuators & *msg*)

The callback function the the subscriber to the teensy_actuator_data topic. This function will run every time ROS::spinOnce is called. It reads the set of data from the topic and immediately writes these values to the Teensy via UART.

Parameters

msg	A set of actuator data that has come from the actuator topic and needs to be written to the Teensy.

7.10.2.2 int main (int *argc,* char ∗∗ *argv*)

The main function for the ROS node to communicate with the Teensy over UART. It receives sensor data from the Teensy and publishes this data to the teensy_sensor_data topic. It also subscribes to the teensy_actuator_data topic and writes the values it gets to the Teensy over UART.

7.10.2.3 void pi_sync ( )

Called before reading sensor data from the Teensy. It will read the buffer until it encounters the SYNC_VALUE that has been defined. Once this happens, the next set of bytes are the set of sensor values.

7.10.2.4 void print_actuators (const semi_truck::Teensy_Actuators & *actuators*)

A function useful for debugging serial communication. Prints the entire set of actuator data to the console.

Parameters

actuators	The object that holds all of the actuator data

7.10.2.5 void print_sensors (const semi_truck::Teensy_Sensors & *sensors*)

A function useful for debugging serial communication. Prints the entire set of sensor data to the console.

Parameters

sensors	The object that holds all of the sensor data

7.10.2.6 void read_from_teensy (int *serial,* semi_truck::Teensy_Sensors & *sensors*)

Reads an entire set of Teensy sensor data by calling read_sensor_msg on each sensor data in the UART message.

7.10.2.7 short read_sensor_msg (int *serial,* char *num_bytes*)

Reads a single value from the UART communication buffer.

Parameters

serial	The serial file descriptor to read from
num_bytes	The number of bytes to read from

Returns

The value of the bytes that have been read

7.10.2.8 void write_actuator_msg (int *serial*, short *actuator_val*, char *num_bytes*)

Writes a single actuator value to the Teensy.

Parameters

serial	The serial file descriptor to write to
actuator_val	The value that is written through UART to the Teensy
num_bytes	Number of bytes to write to the UART

7.10.2.9 void write_to_teensy (int *serial*, const semi_truck::Teensy_Actuators & *actuators*)

Writes an entire set of actuator data to the Teensy via UART.

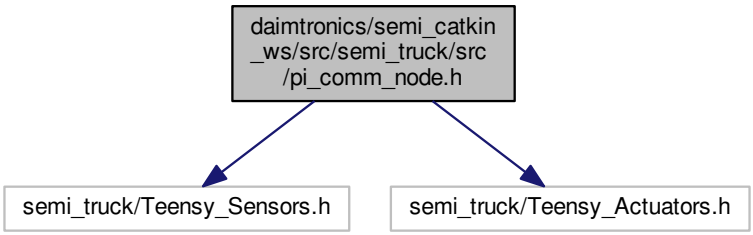
Parameters

serial	The serial file descriptor to write to
actuators	The object that holds all of the actuator data

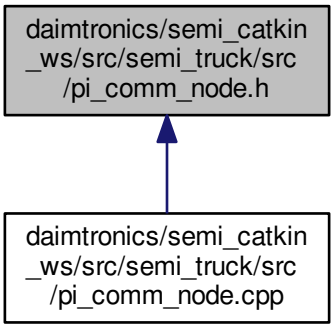
7.11 daimtronics/semi_catkin_ws/src/semi_truck/src/pi_comm_node.h File Reference

```
#include "semi_truck/Teensy_Sensors.h"
#include "semi_truck/Teensy_Actuators.h"
```

Include dependency graph for pi_comm_node.h:



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



Functions

- short read_sensor_msg (int serial, char num_bytes)

  Reads a single value from the UART communication buffer.
- void read_from_teensy (int serial, semi_truck::Teensy_Sensors &sensors)

  Reads an entire set of Teensy sensor data by calling read_sensor_msg on each sensor data in the UART message.
- void write_sensor_msg (int serial, short sensor_val, char num_bytes)
- void write_to_teensy (int serial, const semi_truck::Teensy_Actuators &actuators)

  Writes an entire set of actuator data to the Teensy via UART.
- void update_sensors (semi_truck::Teensy_Sensors &sensors)
- void pi_sync ()

  Called before reading sensor data from the Teensy. It will read the buffer until it encounters the SYNC_VALUE that has been defined. Once this happens, the next set of bytes are the set of sensor values.
- void print_sensors (const semi_truck::Teensy_Sensors &sensors)

  A function useful for debugging serial communication. Prints the entire set of sensor data to the console.
- void print_actuators (const semi_truck::Teensy_Actuators &actuators)

  A function useful for debugging serial communication. Prints the entire set of actuator data to the console.
- void sensor_cb (const semi_truck::Teensy_Sensors &msg)
- void actuator_cb (const semi_truck::Teensy_Actuators &msg)

  The callback function the the subscriber to the teensy_actuator_data topic. This function will run every time ROS↩S::spinOnce is called. It reads the set of data from the topic and immediately writes these values to the Teensy via UART.

7.11.1 Function Documentation

7.11.1.1 void actuator_cb (const semi_truck::Teensy_Actuators & *msg*)

The callback function the the subscriber to the teensy_actuator_data topic. This function will run every time ROS↩::spinOnce is called. It reads the set of data from the topic and immediately writes these values to the Teensy via UART.

Parameters

msg	A set of actuator data that has come from the actuator topic and needs to be written to the Teensy.

7.11.1.2 void pi_sync ( )

Called before reading sensor data from the Teensy. It will read the buffer until it encounters the SYNC_VALUE that has been defined. Once this happens, the next set of bytes are the set of sensor values.

7.11.1.3 void print_actuators (const semi_truck::Teensy_Actuators & *actuators*)

A function useful for debugging serial communication. Prints the entire set of actuator data to the console.

Parameters

actuators	The object that holds all of the actuator data

7.11.1.4 void print_sensors (const semi_truck::Teensy_Sensors & *sensors*)

A function useful for debugging serial communication. Prints the entire set of sensor data to the console.

Parameters

sensors	The object that holds all of the sensor data

7.11.1.5 void read_from_teensy (int *serial,* semi_truck::Teensy_Sensors & *sensors*)

Reads an entire set of Teensy sensor data by calling read_sensor_msg on each sensor data in the UART message.

7.11.1.6 short read_sensor_msg (int *serial,* char *num_bytes*)

Reads a single value from the UART communication buffer.

Parameters

serial	The serial file descriptor to read from
num_bytes	The number of bytes to read from

Returns

The value of the bytes that have been read

7.11.1.7 void sensor_cb (const semi_truck::Teensy_Sensors & *msg*)

7.11.1.8 void update_sensors (semi_truck::Teensy_Sensors & *sensors*)

7.11.1.9 void write_sensor_msg (int *serial*, short *sensor_val*, char *num_bytes*)

7.11.1.10 void write_to_teensy (int *serial*, const semi_truck::Teensy_Actuators & *actuators*)

Writes an entire set of actuator data to the Teensy via UART.

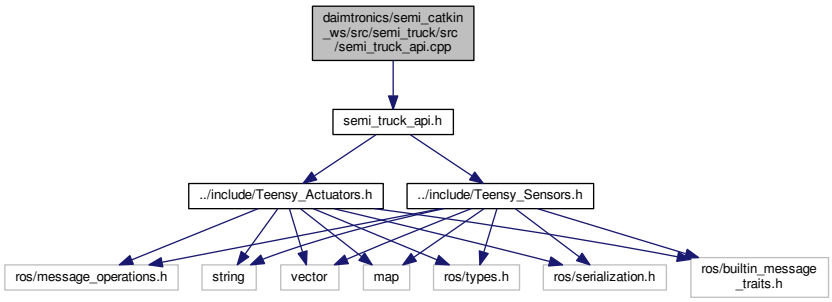
Parameters

serial	The serial file descriptor to write to
actuators	The object that holds all of the actuator data

7.12 daimtronics/semi_catkin_ws/src/semi_truck/src/semi_truck_api.cpp File Reference

```
#include "semi_truck_api.h"
```

Include dependency graph for semi_truck_api.cpp:



Functions

- void set_motor_output (semi_truck::Teensy_Actuators &actuators, int16_t motor_output)
  sets the motor output of the actuators object to the passed in value
- void set_steer_output (semi_truck::Teensy_Actuators &actuators, int16_t steer_output)
  sets the steer output of the actuators object to the passed in value
- void set_fifth_output (semi_truck::Teensy_Actuators &actuators, uint16_t fifth_output)
  sets the fifth wheel of the actuators object to the passed in value
- int16_t get_wheel_speed (semi_truck::Teensy_Sensors &sensors)
  reads the wheel speed of a TeensySensors object
- int16_t get_imu_angle (semi_truck::Teensy_Sensors &sensors)
  reads the imu angle (degrees) of a TeensySensors object
- int16_t get_right_TOF (semi_truck::Teensy_Sensors &sensors)
  reads the right TOF distance (cm) of a TeensySensors object

- int16_t get_left_TOF (semi_truck::Teensy_Sensors &sensors)
  reads the left TOF distance (cm) of a TeensySensors object
- int16_t get_rear_TOF (semi_truck::Teensy_Sensors &sensors)
  reads the rear TOF distance (cm) of a TeensySensors object

7.12.1 Function Documentation

7.12.1.1 int16_t get_imu_angle (semi_truck::Teensy_Sensors & *sensors*)

reads the imu angle (degrees) of a TeensySensors object

Parameters

sensors	a reference to a TeensySensors object to read from.

7.12.1.2 int16_t get_left_TOF (semi_truck::Teensy_Sensors & *sensors*)

reads the left TOF distance (cm) of a TeensySensors object

Parameters

sensors	a reference to a TeensySensors object to read from.

7.12.1.3 int16_t get_rear_TOF (semi_truck::Teensy_Sensors & *sensors*)

reads the rear TOF distance (cm) of a TeensySensors object

Parameters

sensors	a reference to a TeensySensors object to read from.

7.12.1.4 int16_t get_right_TOF (semi_truck::Teensy_Sensors & *sensors*)

reads the right TOF distance (cm) of a TeensySensors object

Parameters

sensors	a reference to a TeensySensors object to read from.

7.12.1.5 int16_t get_wheel_speed (semi_truck::Teensy_Sensors & *sensors*)

reads the wheel speed of a TeensySensors object

Parameters

sensors	a reference to a TeensySensors object to read from.

7.12.1.6 void set_fifth_output (semi_truck::Teensy_Actuators & *actuators,* uint16_t *value*)

sets the fifth wheel of the actuators object to the passed in value

Parameters

actuators	a reference to a TeensyActuators object to alter
value	the value to update the actuators with. 0 for locked and 1 for unlocked.

7.12.1.7 void set_motor_output (semi_truck::Teensy_Actuators & *actuators,* int16_t *value*)

sets the motor output of the actuators object to the passed in value

Parameters

actuators	a reference to a TeensyActuators object to alter
value	the value to update the actuators with. This value should range from 0 for full reverse power to 180 for full forwards power.

7.12.1.8 void set_steer_output (semi_truck::Teensy_Actuators & *actuators,* int16_t *value*)

sets the steer output of the actuators object to the passed in value

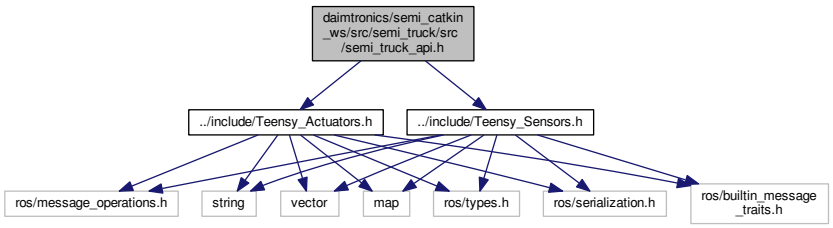
Parameters

actuators	a reference to a TeensyActuators object to alter
value	the value to update the actuators with. This value should range from 0 for a 20 degree angle left to 180 for a 20 degree angle right.

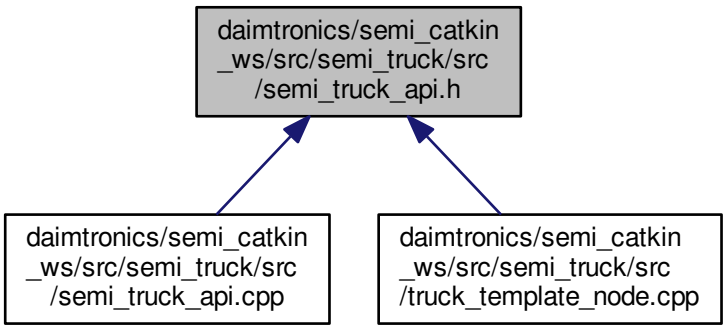
7.13 daimtronics/semi_catkin_ws/src/semi_truck/src/semi_truck_api.h File Reference

```
#include "../include/Teensy_Actuators.h"
#include "../include/Teensy_Sensors.h"
```

Include dependency graph for semi_truck_api.h:



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



Functions

- void set_motor_output (semi_truck::Teensy_Actuators &actuators, int16_t value)

  sets the motor output of the actuators object to the passed in value
- void set_steer_output (semi_truck::Teensy_Actuators &actuators, int16_t value)

  sets the steer output of the actuators object to the passed in value
- void set_fifth_output (semi_truck::Teensy_Actuators &actuators, uint16_t value)

  sets the fifth wheel of the actuators object to the passed in value
- int16_t get_wheel_speed (semi_truck::Teensy_Sensors &sensors)

  reads the wheel speed of a TeensySensors object
- int16_t get_imu_angle (semi_truck::Teensy_Sensors &sensors)

  reads the imu angle (degrees) of a TeensySensors object
- int16_t get_right_TOF (semi_truck::Teensy_Sensors &sensors)

  reads the right TOF distance (cm) of a TeensySensors object
- int16_t get_left_TOF (semi_truck::Teensy_Sensors &sensors)

  reads the left TOF distance (cm) of a TeensySensors object
- int16_t get_rear_TOF (semi_truck::Teensy_Sensors &sensors)

  reads the rear TOF distance (cm) of a TeensySensors object

7.13.1 Function Documentation

7.13.1.1 int16_t get_imu_angle (semi_truck::Teensy_Sensors & *sensors*)

reads the imu angle (degrees) of a TeensySensors object

Parameters

sensors	a reference to a TeensySensors object to read from.

7.13.1.2 int16_t get_left_TOF (semi_truck::Teensy_Sensors & *sensors*)

reads the left TOF distance (cm) of a TeensySensors object

Parameters

sensors	a reference to a TeensySensors object to read from.

7.13.1.3 int16_t get_rear_TOF (semi_truck::Teensy_Sensors & *sensors*)

reads the rear TOF distance (cm) of a TeensySensors object

Parameters

sensors	a reference to a TeensySensors object to read from.

7.13.1.4 int16_t get_right_TOF (semi_truck::Teensy_Sensors & *sensors*)

reads the right TOF distance (cm) of a TeensySensors object

Parameters

sensors	a reference to a TeensySensors object to read from.

7.13.1.5 int16_t get_wheel_speed (semi_truck::Teensy_Sensors & *sensors*)

reads the wheel speed of a TeensySensors object

Parameters

sensors	a reference to a TeensySensors object to read from.

7.13.1.6 void set_fifth_output (semi_truck::Teensy_Actuators & *actuators,* uint16_t *value*)

sets the fifth wheel of the actuators object to the passed in value

Parameters

actuators	a reference to a TeensyActuators object to alter
value	the value to update the actuators with. 0 for locked and 1 for unlocked.

7.13.1.7 void set_motor_output (semi_truck::Teensy_Actuators & *actuators,* int16_t *value*)

sets the motor output of the actuators object to the passed in value

Parameters

actuators	a reference to a TeensyActuators object to alter
value	the value to update the actuators with. This value should range from 0 for full reverse power to 180 for full forwards power.

7.13.1.8 void set_steer_output (semi_truck::Teensy_Actuators & *actuators,* int16_t *value*)

sets the steer output of the actuators object to the passed in value

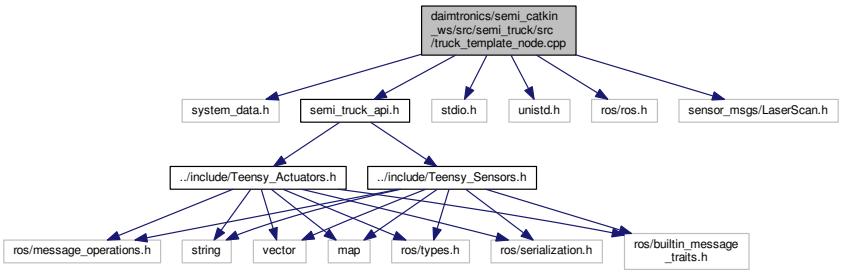
Parameters

actuators	a reference to a TeensyActuators object to alter
value	the value to update the actuators with. This value should range from 0 for a 20 degree angle left to 180 for a 20 degree angle right.

7.14 daimtronics/semi_catkin_ws/src/semi_truck/src/truck_template_node.cpp File Reference

```
#include "system_data.h"
#include "semi_truck_api.h"
#include <stdio.h>
#include <unistd.h>
#include <ros/ros.h>
#include <sensor_msgs/LaserScan.h>
```

Include dependency graph for truck_template_node.cpp:



Functions

- void rplidar_cb (const sensor_msgs::LaserScan &msg)
- void lidar_lite_cb (const sensor_msgs::LaserScan &msg)
- void teensy_sensors_cb (const semi_truck::Teensy_Sensors &msg)
- int main (int argc, char **argv)

7.14.1 Function Documentation

7.14.1.1 void lidar_lite_cb (const sensor_msgs::LaserScan & *msg*)

7.14.1.2 int main (int *argc,* char ** *argv*)

7.14.1.3 void rplidar_cb (const sensor_msgs::LaserScan & *msg*)

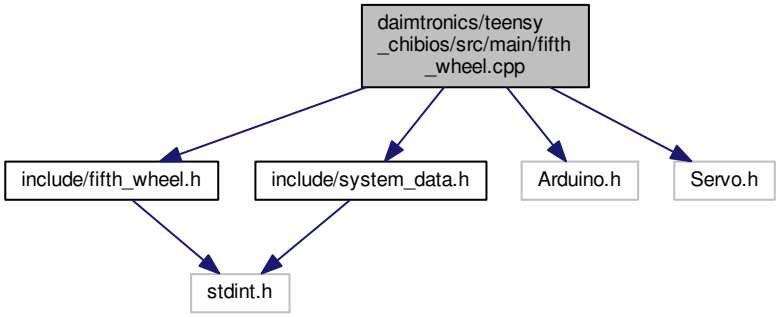
7.14.1.4 void teensy_sensors_cb (const semi_truck::Teensy_Sensors & *msg*)

7.15 daimtronics/teensy_chibios/src/.DS_Store File Reference

7.16 daimtronics/teensy_chibios/src/main/fifth_wheel.cpp File Reference

```
#include "include/fifth_wheel.h"
#include "include/system_data.h"
#include <Arduino.h>
#include <Servo.h>
```

Include dependency graph for fifth_wheel.cpp:



Macros

- #define LOCKED_ANGLE 0
- #define UNLOCKED_ANGLE 180

Functions

- void fifth_wheel_loop_fn (int16_t fifth_output)
- void fifth_wheel_setup (short fifth_wheel_pin)

  Set up the fifth wheel task to write to the pin attached to the fifth wheel, and to be in the locked position.

7.16.1 Macro Definition Documentation

7.16.1.1 #define LOCKED_ANGLE 0

7.16.1.2 #define UNLOCKED_ANGLE 180

7.16.2 Function Documentation

7.16.2.1 void fifth_wheel_loop_fn (int16_t *fifth_output*)

The is the primary function controlling the fifth wheel. It reads the fifth_output value form the system data and writes to the Servo for actuating between the two different angles.

Parameters

fifth_output	the output to the fifth wheel, which will be one of two values, either locked or unlocked

7.16.2.2 void fifth_wheel_setup (short *fifth_wheel_pin*)

Set up the fifth wheel task to write to the pin attached to the fifth wheel, and to be in the locked position.

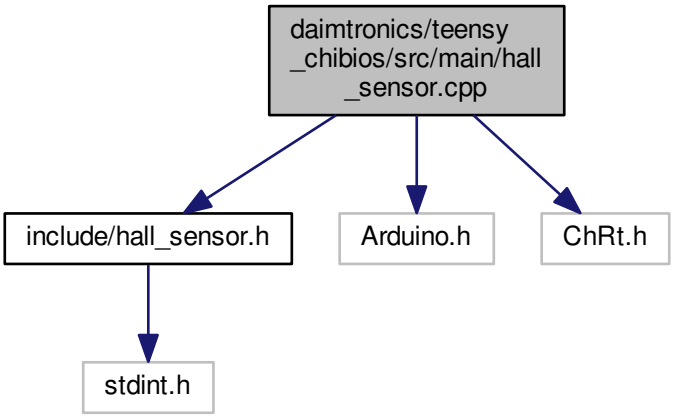
Parameters

fifth_wheel_pin	The pin that signals a PWM to the fifth wheel servo.

7.17 daimtronics/teensy_chibios/src/main/hall_sensor.cpp File Reference

```
#include "include/hall_sensor.h"
#include <Arduino.h>
#include <ChRt.h>
```

Include dependency graph for hall_sensor.cpp:



Functions

- int16_t hall_sensor_loop_fn (short PhaseB_pin, short PhaseC_pin)

  This is the primary function controlling the wheel speed sensor. It reads a Hall sensor that is built into the Tekin R8 Motor. This is a three phased motor (PhaseA, PhaseB, PhaseC). Each phase is offset 120 degrees from each other. Ticks will be incremented if the motor rotates forwards one full revolution, and decremented if it rotates backwards one revolution.
- void hall_sensor_setup (short PhaseA_pin, short PhaseB_pin, short PhaseC_pin)

  Set up the wheel speed task to read high or low values from the three pins attached to the Hall sensor (one for each phase).

Variables

- int16_t ticks = 0

7.17.1 Function Documentation

7.17.1.1 int16_t hall_sensor_loop_fn (short *PhaseB_pin,* short *PhaseC_pin*)

This is the primary function controlling the wheel speed sensor. It reads a Hall sensor that is built into the Tekin R8 Motor. This is a three phased motor (PhaseA, PhaseB, PhaseC). Each phase is offset 120 degrees from each other. Ticks will be incremented if the motor rotates forwards one full revolution, and decremented if it rotates backwards one revolution.

Returns

the current number of ticks that the sensor reads.

7.17.1.2 void hall_sensor_setup (short *PhaseA_pin,* short *PhaseB_pin,* short *PhaseC_pin*)

Set up the wheel speed task to read high or low values from the three pins attached to the Hall sensor (one for each phase).

Parameters

PhaseA_pin	An interrupt is triggered every time the Teensy read a leading edge for this phase.
PhaseB_pin	If this pin is high when the interrupt is triggered, the truck is going in reverse.
PhaseC_pin	If this pin is high when the interrupt is triggered, the truck is going forwards.

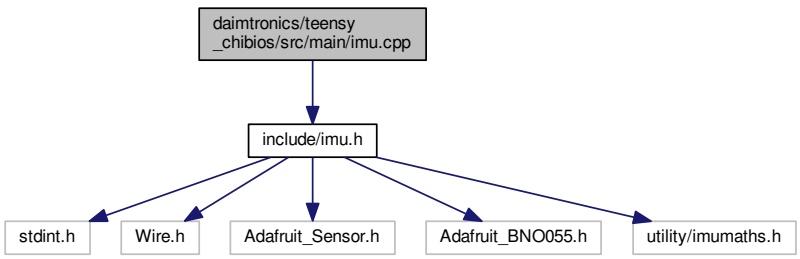
7.17.2 Variable Documentation

7.17.2.1 int16_t ticks = 0

7.18 daimtronics/teensy_chibios/src/main/imu.cpp File Reference

```
#include "include/imu.h"
```

Include dependency graph for imu.cpp:



Functions

- int16_t imu_loop_fn ()

  The primary function for the IMU that reads and returns heading data of the BNO055.The Adafruit_BNO055 library does most of the work here. This function simply sets up a sets up an "event" variable to hold the IMU data.
- void imu_setup ()

  Initializes the BNO055 sensor.
- void print_imu_data (sensors_event_t ∗event)

  A debugging function used to print out IMU orientation to the USB serial device (usually the Arduino Serial Monitor).

Variables

- Adafruit_BNO055 bno = Adafruit_BNO055(55)

  A global variable, for the only Adafruit_BNO055 object in the system.

7.18.1 Function Documentation

7.18.1.1 int16_t imu_loop_fn ( )

The primary function for the IMU that reads and returns heading data of the BNO055.The Adafruit_BNO055 library does most of the work here. This function simply sets up a sets up an "event" variable to hold the IMU data.

Returns

an integer representing the heading angle in degrees

7.18.1.2 void imu_setup ( )

Initializes the BNO055 sensor.

7.18.1.3 void print_imu_data (sensors_event_t ∗ *event*)

A debugging function used to print out IMU orientation to the USB serial device (usually the Arduino Serial Monitor).

7.18.2 Variable Documentation

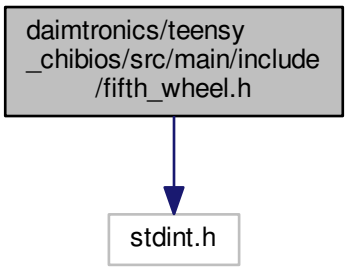
7.18.2.1 Adafruit_BNO055 bno = Adafruit_BNO055(55)

A global variable, for the only Adafruit_BNO055 object in the system.

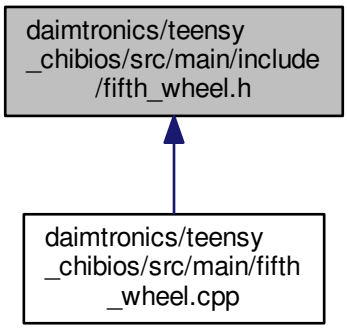
7.19 daimtronics/teensy_chibios/src/main/include/fifth_wheel.h File Reference

```
#include <stdint.h>
```

Include dependency graph for fifth_wheel.h:



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



Macros

- #define LOCKED 1
- #define UNLOCKED 0

Functions

- void fifth_wheel_loop_fn (int16_t fifth_output)
- void fifth_wheel_setup (short fifth_wheel_pin)

  Set up the fifth wheel task to write to the pin attached to the fifth wheel, and to be in the locked position.

7.19.1 Macro Definition Documentation

7.19.1.1 #define LOCKED 1

7.19.1.2 #define UNLOCKED 0

7.19.2 Function Documentation

7.19.2.1 void fifth_wheel_loop_fn (int16_t *fifth_output*)

The is the primary function controlling the fifth wheel. It reads the fifth_output value form the system data and writes to the Servo for actuating between the two different angles.

Parameters

fifth_output	the output to the fifth wheel, which will be one of two values, either locked or unlocked

7.19.2.2 void fifth_wheel_setup (short *fifth_wheel_pin*)

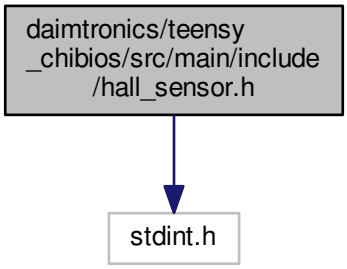
Set up the fifth wheel task to write to the pin attached to the fifth wheel, and to be in the locked position.

Parameters

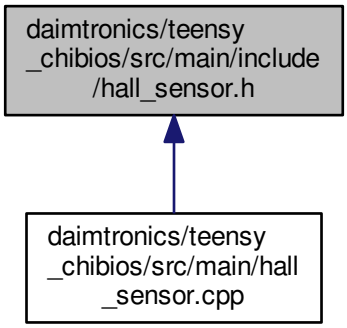
fifth_wheel_pin	The pin that signals a PWM to the fifth wheel servo.

7.20 daimtronics/teensy_chibios/src/main/include/hall_sensor.h File Reference

`#include <stdint.h>`
Include dependency graph for hall_sensor.h:



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



Functions

- int16_t hall_sensor_loop_fn (short PhaseB_pin, short PhaseC_pin)

  This is the primary function controlling the wheel speed sensor. It reads a Hall sensor that is built into the Tekin R8 Motor. This is a three phased motor (PhaseA, PhaseB, PhaseC). Each phase is offset 120 degrees from each other. Ticks will be incremented if the motor rotates forwards one full revolution, and decremented if it rotates backwards one revolution.
- void hall_sensor_setup (short PhaseA_pin, short PhaseB_pin, short PhaseC_pin)

  Set up the wheel speed task to read high or low values from the three pins attached to the Hall sensor (one for each phase).

7.20.1 Function Documentation

7.20.1.1 int16_t hall_sensor_loop_fn (short *PhaseB_pin,* short *PhaseC_pin*)

This is the primary function controlling the wheel speed sensor. It reads a Hall sensor that is built into the Tekin R8 Motor. This is a three phased motor (PhaseA, PhaseB, PhaseC). Each phase is offset 120 degrees from each other. Ticks will be incremented if the motor rotates forwards one full revolution, and decremented if it rotates backwards one revolution.

Returns

the current number of ticks that the sensor reads.

7.20.1.2 void hall_sensor_setup (short *PhaseA_pin,* short *PhaseB_pin,* short *PhaseC_pin*)

Set up the wheel speed task to read high or low values from the three pins attached to the Hall sensor (one for each phase).

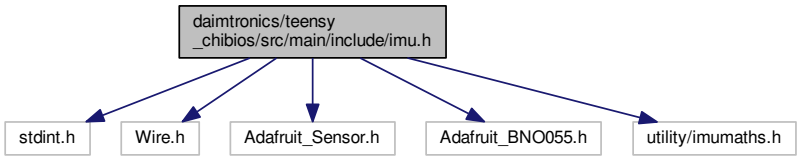
Parameters

PhaseA_pin	An interrupt is triggered every time the Teensy read a leading edge for this phase.
PhaseB_pin	If this pin is high when the interrupt is triggered, the truck is going in reverse.
PhaseC_pin	If this pin is high when the interrupt is triggered, the truck is going forwards.

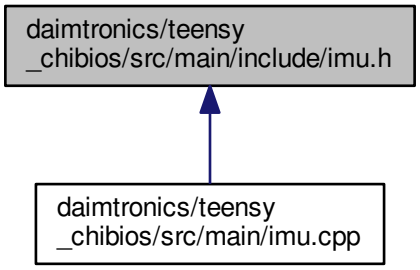
7.21 daimtronics/teensy_chibios/src/main/include/imu.h File Reference

```
#include <stdint.h>
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BNO055.h>
#include <utility/imumaths.h>
```

Include dependency graph for imu.h:



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



Functions

- short imu_loop_fn ()

  The primary function for the IMU that reads and returns heading data of the BNO055.The Adafruit_BNO055 library does most of the work here. This function simply sets up a sets up an "event" variable to hold the IMU data.
- void imu_setup ()

  Initializes the BNO055 sensor.
- void print_imu_data (sensors_event_t ∗event)

  A debugging function used to print out IMU orientation to the USB serial device (usually the Arduino Serial Monitor).

7.21.1 Function Documentation

7.21.1.1 short imu_loop_fn ( )

The primary function for the IMU that reads and returns heading data of the BNO055.The Adafruit_BNO055 library does most of the work here. This function simply sets up a sets up an "event" variable to hold the IMU data.

Returns

an integer representing the heading angle in degrees

7.21.1.2 void imu_setup ( )

Initializes the BNO055 sensor.

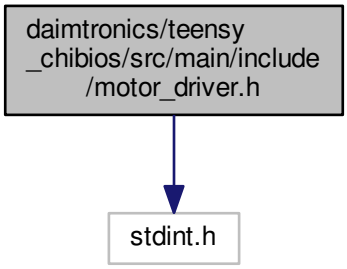
7.21.1.3 void print_imu_data (sensors_event_t * *event*)

A debugging function used to print out IMU orientation to the USB serial device (usually the Arduino Serial Monitor).

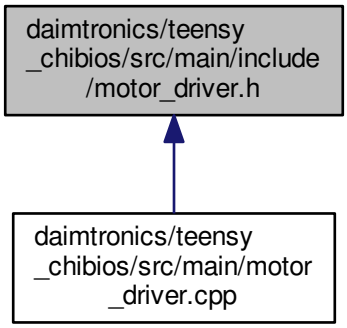
7.22 daimtronics/teensy_chibios/src/main/include/motor_driver.h File Reference

```
#include <stdint.h>
```

Include dependency graph for motor_driver.h:



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



Functions

- void motor_driver_loop_fn (int16_t motor_output)

  This is the primary function controlling the motor. It reads the motor output value from the system data and controls the motor with this value.
- int16_t scale_output (int16_t motor_output)

  Scales a value coming from the pi between -100 and 100 to between 0 and 180 for what the Servo library requires.
- void motor_driver_setup (short motor_pin)

  Set up the motor driver task to write to the pin attached to the motor, and to be in the locked position. It also outputs a value corresponding to zero torque.
- int16_t stop_motor (int16_t wheel_speed, int16_t time_step)

  Runs a control loop to stop the motor based on the reported wheel speed, and returns a value to be output to the motor.

7.22.1 Function Documentation

7.22.1.1 void motor_driver_loop_fn (int16_t *motor_output*)

This is the primary function controlling the motor. It reads the motor output value from the system data and controls the motor with this value.

Parameters

motor_output	the output to the motor

7.22.1.2 void motor_driver_setup (short *motor_pin*)

Set up the motor driver task to write to the pin attached to the motor, and to be in the locked position. It also outputs a value corresponding to zero torque.

Parameters

motor_pin	The pin sends a PWM signal to the motor.

7.22.1.3 int16_t scale_output (int16_t *motor_output*)

Scales a value coming from the pi between -100 and 100 to between 0 and 180 for what the Servo library requires.

Parameters

motor_output	An input value, ideally between -100 and 100 (although it will be limited to -100 or 100 if outside this range

Returns

a value to output to the motor between 0 and 180

7.22.1.4 int16_t stop_motor (int16_t *wheel_speed,* int16_t *time_step*)

Runs a control loop to stop the motor based on the reported wheel speed, and returns a value to be output to the motor.

Parameters

wheel_speed	speed of the truck read by the wheel speed sensor
time_step	number of millis since the last time this task ran; used in integral control

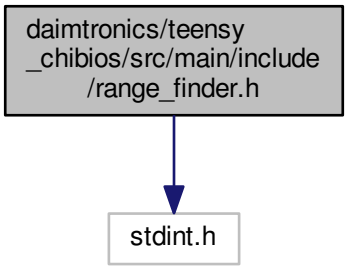
Returns

the output to the motor

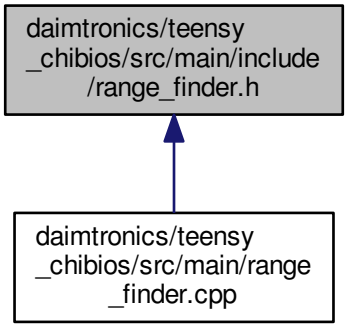
7.23 daimtronics/teensy_chibios/src/main/include/range_finder.h File Reference

```
#include <stdint.h>
```

Include dependency graph for range_finder.h:



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



Functions

- long range_finder_loop_fn (short urf_echo_pin)
- void range_finder_ping (short urf_trig_pin)
- void range_finder_setup (short urf_trig_pin)

7.23.1 Function Documentation

7.23.1.1 long range_finder_loop_fn (short *urf_echo_pin*)

This is the primary function controlling the URFs. It reads a value representing a distance to an object from the URF sensor and returns that value

Returns

distance to object

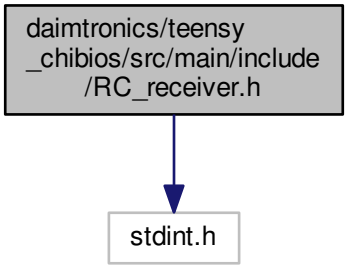
7.23.1.2 void range_finder_ping (short *urf_trig_pin*)

7.23.1.3 void range_finder_setup (short *urf_trig_pin*)

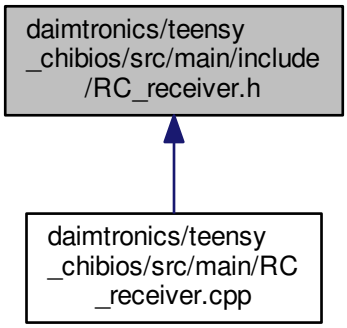
7.24 daimtronics/teensy_chibios/src/main/include/RC_receiver.h File Reference

```
#include <stdint.h>
```

Include dependency graph for RC_receiver.h:



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



Functions

- int16_t RC_receiver_SW1_fn (short PWM_PIN)

  This is the primary function reading Switch 1 on the receiver. It reads a PWM signal that the RC receiver receives from the RC controller. Based on the specific timing of the PWM, a deadman mode (either deadman switch pressed or not pressed) is selected to send to the rest of the platform.
- int16_t RC_receiver_SW2_fn (short PWM_PIN)

- int16_t RC_receiver_SW3_fn (short PWM_PIN)

  This is the primary function controlling the RC receiver. It reads a PWM signal that the RC receiver receives from the RC controller. Based on the specific timing of the PWM, a drive mode (either manual or one of the autonomous algorithms on the Pi) is selected to control the vehicle.
- void RC_receiver_setup ()

7.24.1 Function Documentation

7.24.1.1 void RC_receiver_setup ( )

7.24.1.2 int16_t RC_receiver_SW1_fn (short *PWM_PIN*)

This is the primary function reading Switch 1 on the receiver. It reads a PWM signal that the RC receiver receives from the RC controller. Based on the specific timing of the PWM, a deadman mode (either deadman switch pressed or not pressed) is selected to send to the rest of the platform.

Returns

the deadman mode of the semi-truck based on RC receiver signal

7.24.1.3 int16_t RC_receiver_SW2_fn (short *PWM_PIN*)

7.24.1.4 int16_t RC_receiver_SW3_fn (short *PWM_PIN*)

This is the primary function controlling the RC receiver. It reads a PWM signal that the RC receiver receives from the RC controller. Based on the specific timing of the PWM, a drive mode (either manual or one of the autonomous algorithms on the Pi) is selected to control the vehicle.

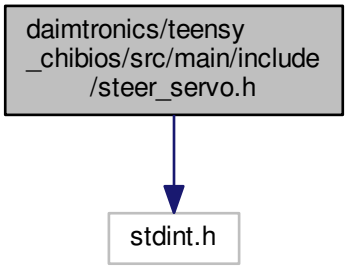
Returns

the driving mode of the semi-truck based on RC receiver signal

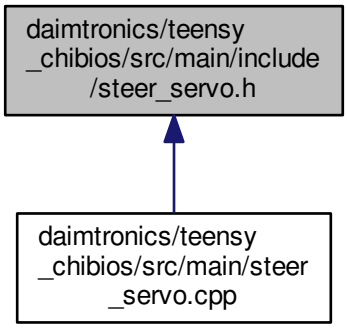
7.25 daimtronics/teensy_chibios/src/main/include/steer_servo.h File Reference

```
#include <stdint.h>
```

Include dependency graph for steer_servo.h:



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



Functions

- void steer_servo_loop_fn (int16_t steer_output)
- void steer_servo_setup (short servo_pin)

  Set up the steer servo task to write to the pin attached to the servo controlling angle of the trucks's front axis.

7.25.1 Function Documentation

7.25.1.1 void steer_servo_loop_fn (int16_t *steer_output*)

The is the primary function controlling the steering servo wheel. It reads a value form the system data and controls the steering servo based on what the system data contains.

Parameters

steer_output	the output to the steering servo that controls angle

7.25.1.2 void steer_servo_setup (short *servo_pin*)

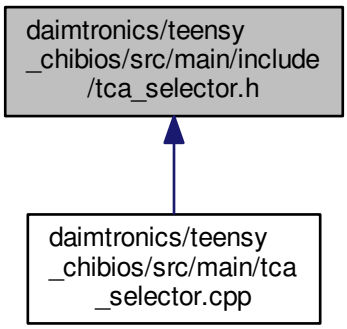
Set up the steer servo task to write to the pin attached to the servo controlling angle of the trucks's front axis.

Parameters

servo_pin	The pin that signals a PWM to the steering servo.

7.26 daimtronics/teensy_chibios/src/main/include/tca_selector.h File Reference

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



Functions

- void tcaselect (short i)

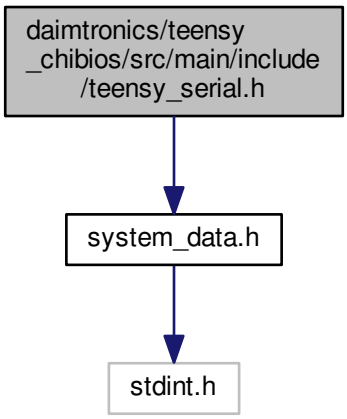
7.26.1 Function Documentation

7.26.1.1 void tcaselect (short *i*)

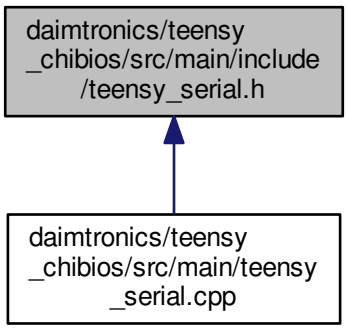
7.27 daimtronics/teensy_chibios/src/main/include/teensy_serial.h File Reference

```
#include "system_data.h"
```

Include dependency graph for teensy_serial.h:



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



Macros

- #define HWSERIAL Serial1

Functions

- void teensy_serial_loop_fn (system_data_t *system_data)

  The primary function for communicating between the Teensy and the Pi over the Serial UART port.
- void teensy_serial_setup ()

  Sets up the serial communication for the teensy to output data to both the Pi (through UART) and a PC console (through USB).
- void set_sensor_msg (int user_input, sensor_data_t *data_ptr)
- void teensy_sync ()

  Ensures that the data being read from the Pi is synced. It clears the serial buffer up until it reads the designated SYNC_VALUE.
- void clear_buffer ()

  Clears the serial buffer from all of its data. This is used primarily when the Teensy is restarted, and the Pi has been sending data that the Teensy does not need to process.
- void read_from_pi (actuator_data_t *actuators_ptr)
- void print_sensor_msg (sensor_data_t *sensors_ptr)
- void print_actuator_msg (actuator_data_t *actuators_ptr)

7.27.1 Macro Definition Documentation

7.27.1.1 #define HWSERIAL Serial1

7.27.2 Function Documentation

7.27.2.1 void clear_buffer ( )

Clears the serial buffer from all of its data. This is used primarily when the Teensy is restarted, and the Pi has been sending data that the Teensy does not need to process.

7.27.2.2 void print_actuator_msg (actuator_data_t * *actuators_ptr*)

7.27.2.3 void print_sensor_msg (sensor_data_t * *sensors_ptr*)

7.27.2.4 void read_from_pi (actuator_data_t * *actuators_ptr*)

7.27.2.5 void set_sensor_msg (int *user_input,* sensor_data_t * *data_ptr*)

7.27.2.6 void teensy_serial_loop_fn (system_data_t * *system_data*)

The primary function for communicating between the Teensy and the Pi over the Serial UART port.

Parameters

system_data	a pointer to the system data that is declared statically in main.ino.

7.27.2.7 void teensy_serial_setup ( )

Sets up the serial communication for the teensy to output data to both the Pi (through UART) and a PC console (through USB).

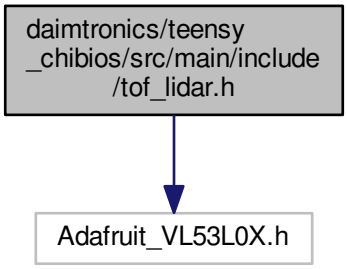
7.27.2.8 void teensy_sync ( )

Ensures that the data being read from the Pi is synced. It clears the serial buffer up until it reads the designated SYNC_VALUE.

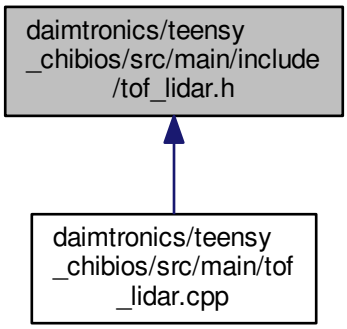
7.28 daimtronics/teensy_chibios/src/main/include/tof_lidar.h File Reference

```
#include <Adafruit_VL53L0X.h>
```

Include dependency graph for tof_lidar.h:



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



Functions

- int16_t tof_loop_fn ()

  This is the primary function controlling the ToF Lidar for reading distance measurements on the sensors.
- void tof_lidar_setup ()

  Initializes the VL53L0X sensor.

7.28.1 Function Documentation

7.28.1.1 void tof_lidar_setup ( )

Initializes the VL53L0X sensor.

7.28.1.2 int16_t tof_loop_fn ( )

This is the primary function controlling the ToF Lidar for reading distance measurements on the sensors.

The Adafruit_VL53L0X library does most of the work and the function here calls the measure.RangeMilliMeter instruction and stores the distance here.

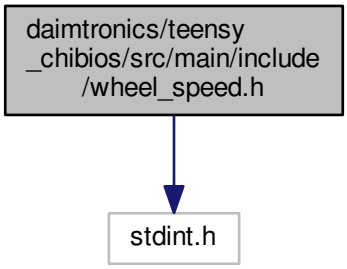
Returns

an integer representing distance the sensor detected in millimeters

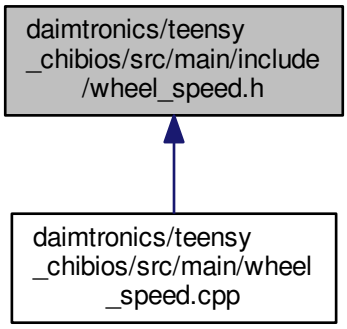
7.29 daimtronics/teensy_chibios/src/main/include/wheel_speed.h File Reference

```
#include <stdint.h>
```

Include dependency graph for wheel_speed.h:



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



Functions

- int16_t wheel_speed_loop_fn (int16_t ticks)

  This function reads the motor ticks that have been determined by the hall sensor and converts this value into a speed for the truck.
- void wheel_speed_setup ()

7.29.1 Function Documentation

7.29.1.1 int16_t wheel_speed_loop_fn (int16_t *ticks*)

This function reads the motor ticks that have been determined by the hall sensor and converts this value into a speed for the truck.

Parameters

ticks	The number of ticks that is kept track of by the hall_sensor task.

Returns

the speed of the truck

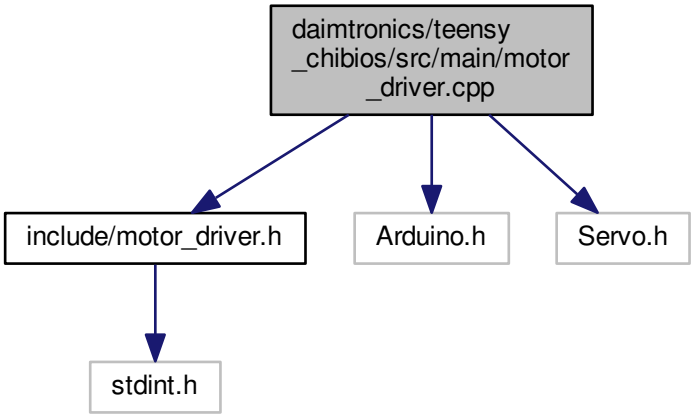
7.29.1.2 void wheel_speed_setup ( )

7.30 daimtronics/teensy_chibios/src/main/main.ino File Reference

7.31 daimtronics/teensy_chibios/src/main/motor_driver.cpp File Reference

```
#include "include/motor_driver.h"
#include <Arduino.h>
#include <Servo.h>
```

Include dependency graph for motor_driver.cpp:



Macros

- #define WHEEL_SPEED_STOP 0
- #define MOTOR_STOP 90
- #define FORWARDS 120
- #define INIT_VALUE 68
- #define KP 1
- #define KI 0.05f
- #define SAT_ERROR 1000
- #define MAX_TIME_STEP 500
- #define WHEEL_SPEED_RANGE 1000
- #define MOTOR_RANGE 180
- #define FULL_REVERSE 1087
- #define FULL_FORWARD 1660

Functions

- void motor_driver_loop_fn (int16_t motor_output)

  This is the primary function controlling the motor. It reads the motor output value from the system data and controls the motor with this value.
- int16_t scale_output (int16_t motor_output)

  Scales a value coming from the pi between -100 and 100 to between 0 and 180 for what the Servo library requires.
- void motor_driver_setup (short motor_pin)

  Set up the motor driver task to write to the pin attached to the motor, and to be in the locked position. It also outputs a value corresponding to zero torque.
- int16_t stop_motor (int16_t wheel_speed, int16_t time_step)

  Runs a control loop to stop the motor based on the reported wheel speed, and returns a value to be output to the motor.

7.31.1 Macro Definition Documentation

7.31.1.1 #define FORWARDS 120

7.31.1.2 #define FULL_FORWARD 1660

7.31.1.3 #define FULL_REVERSE 1087

7.31.1.4 #define INIT_VALUE 68

7.31.1.5 #define KI 0.05f

7.31.1.6 #define KP 1

7.31.1.7 #define MAX_TIME_STEP 500

7.31.1.8 #define MOTOR_RANGE 180

7.31.1.9 #define MOTOR_STOP 90

7.31.1.10 #define SAT_ERROR 1000

7.31.1.11 #define WHEEL_SPEED_RANGE 1000

7.31.1.12 #define WHEEL_SPEED_STOP 0

7.31.2 Function Documentation

7.31.2.1 void motor_driver_loop_fn (int16_t *motor_output*)

This is the primary function controlling the motor. It reads the motor output value from the system data and controls the motor with this value.

Parameters

motor_output	the output to the motor

7.31.2.2 void motor_driver_setup (short *motor_pin*)

Set up the motor driver task to write to the pin attached to the motor, and to be in the locked position. It also outputs a value corresponding to zero torque.

Parameters

motor_pin	The pin sends a PWM signal to the motor.

7.31.2.3 int16_t scale_output (int16_t *motor_output*)

Scales a value coming from the pi between -100 and 100 to between 0 and 180 for what the Servo library requires.

Parameters

motor_output	An input value, ideally between -100 and 100 (although it will be limited to -100 or 100 if outside this range

Returns

a value to output to the motor between 0 and 180

7.31.2.4 int16_t stop_motor (int16_t *wheel_speed,* int16_t *time_step*)

Runs a control loop to stop the motor based on the reported wheel speed, and returns a value to be output to the motor.

Parameters

wheel_speed	speed of the truck read by the wheel speed sensor
time_step	number of millis since the last time this task ran; used in integral control

Returns

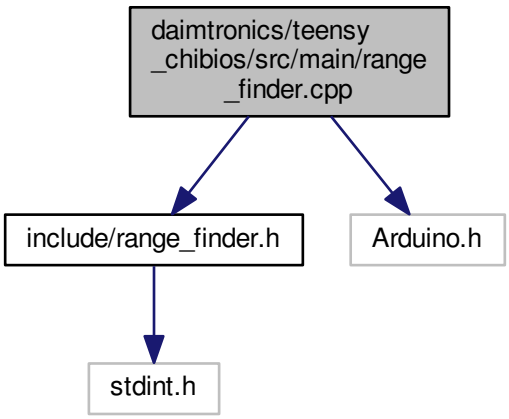
the output to the motor

7.32 daimtronics/teensy_chibios/src/main/range_finder.cpp File Reference

```
#include "include/range_finder.h"
```

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

Include dependency graph for range_finder.cpp:



Functions

- long range_finder_loop_fn (short urf_echo_pin)
- void range_finder_ping (short urf_trig_pin)
- void range_finder_setup (short urf_trig_pin)

Variables

- int val = 0
- unsigned long high_time = 0
- long distance = 0
- unsigned long time = 0

7.32.1 Function Documentation

7.32.1.1 long range_finder_loop_fn (short *urf_echo_pin*)

This is the primary function controlling the URFs. It reads a value representing a distance to an object from the URF sensor and returns that value

Returns

distance to object

7.32.1.2 void range_finder_ping (short *urf_trig_pin*)

7.32.1.3 void range_finder_setup (short *urf_trig_pin*)

7.32.2 Variable Documentation

7.32.2.1 long distance = 0

7.32.2.2 unsigned long high_time = 0

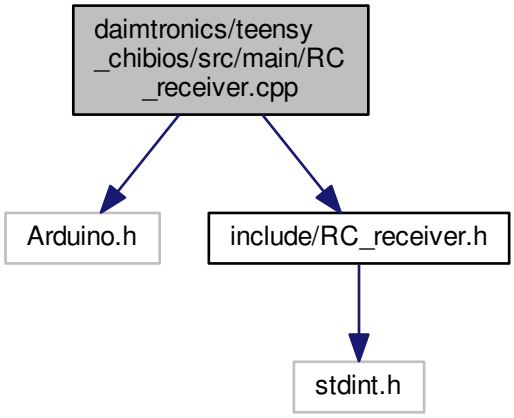
7.32.2.3 unsigned long time = 0

7.32.2.4 int val = 0

7.33 daimtronics/teensy_chibios/src/main/RC_receiver.cpp File Reference

```
#include <Arduino.h>
#include "include/RC_receiver.h"
```

Include dependency graph for RC_receiver.cpp:



Functions

- int16_t RC_receiver_SW1_fn (short PWM_PIN)

  This is the primary function reading Switch 1 on the receiver. It reads a PWM signal that the RC receiver receives from the RC controller. Based on the specific timing of the PWM, a deadman mode (either deadman switch pressed or not pressed) is selected to send to the rest of the platform.
- int16_t RC_receiver_SW3_fn (short PWM_PIN)

  This is the primary function controlling the RC receiver. It reads a PWM signal that the RC receiver receives from the RC controller. Based on the specific timing of the PWM, a drive mode (either manual or one of the autonomous algorithms on the Pi) is selected to control the vehicle.
- void RC_receiver_setup ()

Variables

- float SW1_high_time = 0
- volatile unsigned long SW1_time = 0
- short SW1_mode = 0
- float SW3_high_time = 0
- volatile unsigned long SW3_time = 0
- short SW3_mode = 0

7.33.1 Function Documentation

7.33.1.1 void RC_receiver_setup ( )

7.33.1.2 int16_t RC_receiver_SW1_fn (short *PWM_PIN*)

This is the primary function reading Switch 1 on the receiver. It reads a PWM signal that the RC receiver receives from the RC controller. Based on the specific timing of the PWM, a deadman mode (either deadman switch pressed or not pressed) is selected to send to the rest of the platform.

Returns

the deadman mode of the semi-truck based on RC receiver signal

7.33.1.3 int16_t RC_receiver_SW3_fn (short *PWM_PIN*)

This is the primary function controlling the RC receiver. It reads a PWM signal that the RC receiver receives from the RC controller. Based on the specific timing of the PWM, a drive mode (either manual or one of the autonomous algorithms on the Pi) is selected to control the vehicle.

Returns

the driving mode of the semi-truck based on RC receiver signal

7.33.2 Variable Documentation

7.33.2.1 float SW1_high_time = 0

7.33.2.2 short SW1_mode = 0

7.33.2.3 volatile unsigned long SW1_time = 0

7.33.2.4 float SW3_high_time = 0

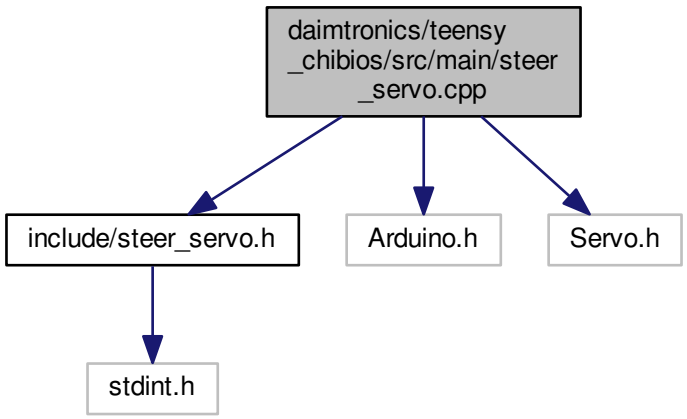
7.33.2.5 short SW3_mode = 0

7.33.2.6 volatile unsigned long SW3_time = 0

7.34 daimtronics/teensy_chibios/src/main/steer_servo.cpp File Reference

```
#include "include/steer_servo.h"
#include <Arduino.h>
#include <Servo.h>
```

Include dependency graph for steer_servo.cpp:



Macros

- #define STRAIGHT 90
- #define MIN_ANGLE 1400
- #define MAX_ANGLE 1800

Functions

- void steer_servo_loop_fn (int16_t steer_output)
- void steer_servo_setup (short servo_pin)

  Set up the steer servo task to write to the pin attached to the servo controlling angle of the trucks's front axis.

7.34.1 Macro Definition Documentation

7.34.1.1 #define MAX_ANGLE 1800

7.34.1.2 #define MIN_ANGLE 1400

7.34.1.3 #define STRAIGHT 90

7.34.2 Function Documentation

7.34.2.1 void steer_servo_loop_fn (int16_t *steer_output*)

The is the primary function controlling the steering servo wheel. It reads a value form the system data and controls the steering servo based on what the system data contains.

Parameters

steer_output	the output to the steering servo that controls angle

7.34.2.2 void steer_servo_setup (short *servo_pin*)

Set up the steer servo task to write to the pin attached to the servo controlling angle of the trucks's front axis.

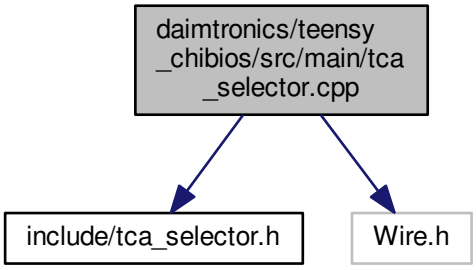
Parameters

servo_pin	The pin that signals a PWM to the steering servo.

7.35 daimtronics/teensy_chibios/src/main/tca_selector.cpp File Reference

```
#include "include/tca_selector.h"
#include <Wire.h>
```

Include dependency graph for tca_selector.cpp:



Macros

- #define TCAADDR 0x70

Functions

- void tcaselect (short i)

7.35.1 Macro Definition Documentation

7.35.1.1 #define TCAADDR 0x70

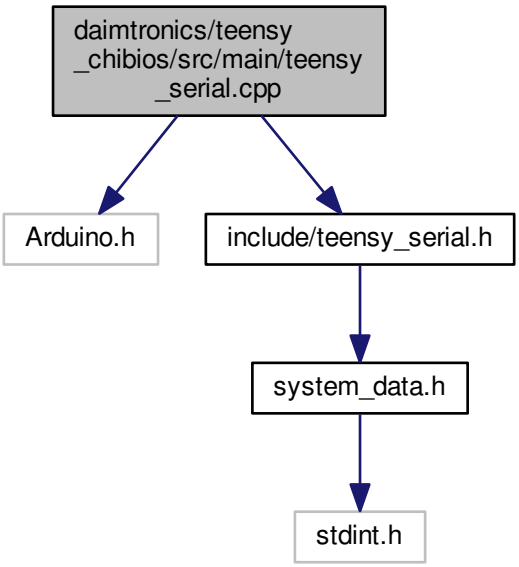
7.35.2 Function Documentation

7.35.2.1 void tcaselect (short *i*)

7.36 daimtronics/teensy_chibios/src/main/teensy_serial.cpp File Reference

```
#include <Arduino.h>
#include "include/teensy_serial.h"
```

Include dependency graph for teensy_serial.cpp:



Macros

- #define SHORT_SIZE 2
- #define ACT_DATA_SIZE_W_SYNC 8
- #define ACT_DATA_SIZE 6
- #define READ_CYCLES 2
- #define SYNC_VALUE -32000

Functions

- void teensy_serial_loop_fn (system_data_t *system_data)

  The primary function for communicating between the Teensy and the Pi over the Serial UART port.
- void teensy_serial_setup ()

  Sets up the serial communication for the teensy to output data to both the Pi (through UART) and a PC console (through USB).
- void clear_buffer ()

  Clears the serial buffer from all of its data. This is used primarily when the Teensy is restarted, and the Pi has been sending data that the Teensy does not need to process.
- void teensy_sync ()

  Ensures that the data being read from the Pi is synced. It clears the serial buffer up until it reads the designated SYNC_VALUE.
- void read_from_pi (actuator_data_t *actuators_ptr)
- void print_sensor_msg (sensor_data_t *sensors_ptr)
- void print_actuator_msg (actuator_data_t *actuators_ptr)

7.36.1 Macro Definition Documentation

7.36.1.1 #define ACT_DATA_SIZE 6

7.36.1.2 #define ACT_DATA_SIZE_W_SYNC 8

7.36.1.3 #define READ_CYCLES 2

7.36.1.4 #define SHORT_SIZE 2

7.36.1.5 #define SYNC_VALUE -32000

7.36.2 Function Documentation

7.36.2.1 void clear_buffer ( )

Clears the serial buffer from all of its data. This is used primarily when the Teensy is restarted, and the Pi has been sending data that the Teensy does not need to process.

7.36.2.2 void print_actuator_msg (actuator_data_t * *actuators_ptr*)

7.36.2.3 void print_sensor_msg (sensor_data_t * *sensors_ptr*)

7.36.2.4 void read_from_pi (actuator_data_t * *actuators_ptr*)

7.36.2.5 void teensy_serial_loop_fn (system_data_t * *system_data*)

The primary function for communicating between the Teensy and the Pi over the Serial UART port.

Parameters

system_data	a pointer to the system data that is declared statically in main.ino.

7.36.2.6 void teensy_serial_setup ( )

Sets up the serial communication for the teensy to output data to both the Pi (through UART) and a PC console (through USB).

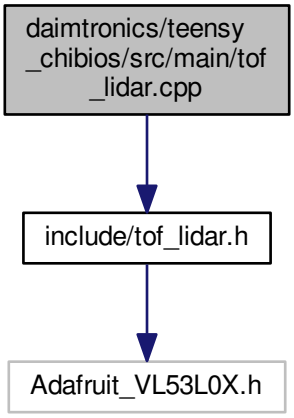
7.36.2.7 void teensy_sync ( )

Ensures that the data being read from the Pi is synced. It clears the serial buffer up until it reads the designated SYNC_VALUE.

7.37 daimtronics/teensy_chibios/src/main/tof_lidar.cpp File Reference

```
#include "include/tof_lidar.h"
```

Include dependency graph for tof_lidar.cpp:



Macros

- #define TCAADDR 0x70

Functions

- void tcaselect (uint8_t i)
- int16_t tof_loop_fn ()

  This is the primary function controlling the ToF Lidar for reading distance measurements on the sensors.
- void tof_lidar_setup ()

  Initializes the VL53L0X sensor.

Variables

- Adafruit_VL53L0X sensor1 = Adafruit_VL53L0X()

  A global variable that sets up the sensors to be used here.
- Adafruit_VL53L0X sensor2 = Adafruit_VL53L0X()
- Adafruit_VL53L0X sensor3 = Adafruit_VL53L0X()

7.37.1 Macro Definition Documentation

7.37.1.1 #define TCAADDR 0x70

7.37.2 Function Documentation

7.37.2.1 void tcaselect (uint8_t *i*)

7.37.2.2 void tof_lidar_setup ( )

Initializes the VL53L0X sensor.

7.37.2.3 int16_t tof_loop_fn ( )

This is the primary function controlling the ToF Lidar for reading distance measurements on the sensors.

The Adafruit_VL53L0X library does most of the work and the function here calls the measure.RangeMilliMeter instruction and stores the distance here.

Returns

an integer representing distance the sensor detected in millimeters

7.37.3 Variable Documentation

7.37.3.1 Adafruit_VL53L0X sensor1 = Adafruit_VL53L0X()

A global variable that sets up the sensors to be used here.

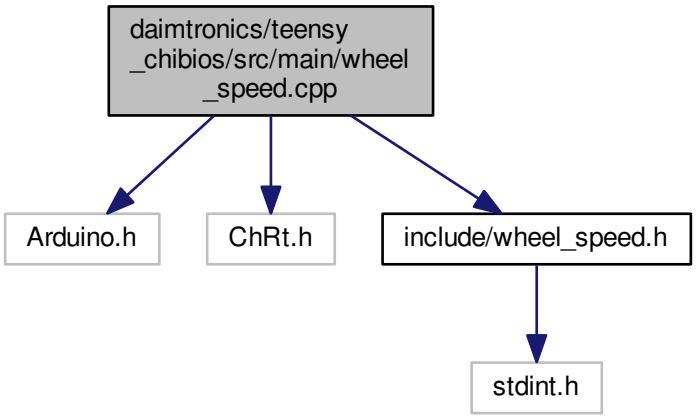
7.37.3.2 Adafruit_VL53L0X sensor2 = Adafruit_VL53L0X()

7.37.3.3 Adafruit_VL53L0X sensor3 = Adafruit_VL53L0X()

7.38 daimtronics/teensy_chibios/src/main/wheel_speed.cpp File Reference

```
#include <Arduino.h>
#include <ChRt.h>
#include "include/wheel_speed.h"
```

Include dependency graph for wheel_speed.cpp:



Macros

- #define SCALE 1
- #define MAX_CHANGE 32768

Functions

- int16_t wheel_speed_loop_fn (int16_t ticks)

  This function reads the motor ticks that have been determined by the hall sensor and converts this value into a speed for the truck.
- void wheel_speed_setup ()

Variables

- int16_t speed
- int16_t prev_ticks =0
- uint16_t prev_time = chVTGetSystemTime()

7.38.1 Macro Definition Documentation

7.38.1.1 #define MAX_CHANGE 32768

7.38.1.2 #define SCALE 1

7.38.2 Function Documentation

7.38.2.1 int16_t wheel_speed_loop_fn (int16_t *ticks*)

This function reads the motor ticks that have been determined by the hall sensor and converts this value into a speed for the truck.

Parameters

ticks	The number of ticks that is kept track of by the hall_sensor task.

Returns

the speed of the truck

7.38.2.2 void wheel_speed_setup ( )

7.38.3 Variable Documentation

7.38.3.1 int16_t prev_ticks =0

7.38.3.2 uint16_t prev_time = chVTGetSystemTime()

7.38.3.3 int16_t speed