BeagleBoardCommandStation

Generated by Doxygen 1.8.13

Contents

1 Introduction

This is the program for the Beagle Board LCC/DCC/Railcom command station. It uses OpenMRN to interface with tha LCC/OpenLCB network to implement a LCC/OpenLCB aware command station node that can operate DCC/Railcom locomotives. It should work with any of the Beagle Board addon boards I have designed. It uses the AM335X's PRUs to generate the DCC signals. It uses the OpenMRN Console class to communicate over a Tcp/Ip channel with a Tcl/Tk coded GUI program to provide a user friendly point-and-click high level user interface.

2 Building

This program needs the OpenMRN library installed in a "standard" place: /opt/openmrn or ∼/openmrn or else in the location specified by the environment variable OPENMRNPATH.

Various build options can be controlled in the Hardware.hxx header file in the target directory. Specificly, the GPIO pin assignments, whether to build a binary OpenLCB Tcp/Ip connected node, a GridConnect connected node, or a LCC CAN connected node. And if a network connected node the default port and host to connect to, also the console port to or to use a terminal console (for debugging).

These build options include:

- USE_OPENLCB_TCP_HOST Use a binary OpenLCB over Tcp/Ip connection. Normally NOT defined.

- DEFAULT_OPENLCB_TCP_HOST Default OpenLCB over Tcp/Ip host to connect to normally "localhost".
- DEFAULT_OPENLCB_TCP_PORT Default OpenLCB over Tcp/Ip port – normally 12020.

- USE_GRIDCONNECT_HOST Use a binary GridConnect over Tcp/Ip connection. Normally NOT defined.
- DEFAULT_TCP_GRIDCONNECT_HOST Default GridConnect over Tcp/Ip host to connect to – normally "localhost".
- DEFAULT_TCP_GRIDCONNECT_PORT Default GridConnect over Tcp/Ip port – normally 12021.

- PRINT_ALL_PACKETS Print all LCC Packets. Normally NOT defined.

- USE_SOCKET_CAN_PORT Use a hardware CAN connection. Normally defined.
- DEFAULT_CAN_SOCKET CAN family socket name. Normally "can1".

- START_GCTCP_HUB Start a Grid Connect Hub server
- DEFAULT_GRIDCONNECT_HUB_PORT Default port the Grid Connect Hub server should listen on – normally 12021.

- TERMINALCONSOLE Use a terminal console. Normally NOT defined – debug use only.
- CONSOLEPORT Console port to listen on – normally 9900.

3 Configuration

There are three configuration sections, one for each of the DCC outputs (Main and Programming) and one for the fan control.

The two DCC outputs have these configuration options:

- The event to send when there is a short.
- The event to send when short is cleared.
- The event to send when the command station is shutdown due to over current.
- The event to send when the shutdown is cleared.
- The event to send when the thermal flag goes on.
- The event to send when the thermal flag goes off.

The fan control section has these configuration options:

- The alarm temperature threshold, in tenths of degree centitrade.
- The event to send when the temperature excedes the alarm temperature threshold.
- The event to send when the temperature drops below the alarm temperature threshold.
- The fan temperature threshold, in tenths of degree centitrade.
- The event to send when the temperature excedes the fan temperature threshold.
- The event to send when the temperature drops below the fan temperature threshold.

4 Todo List

Member ExtendedRingBuffer$<$ T $>$::get (T $*$buf, size_t items)

(Stuart Baker) significant optimization opportunity

Member ExtendedRingBuffer$<$ T $>$::put (const T $*$buf, size_t items)

(Stuart Baker) significant optimization opportunity

5 Module Index

5.1 Modules

Here is a list of all modules:

6 Hierarchical Index

6.1 Class Hierarchy

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

7 Class Index

7.1 Class List

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

8 File Index

8.1 File List

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

9 Module Documentation

9.1 Beagle Board Command Station

9.1.1 SYNOPSIS

BBBCommandStationOpenMRN [options]

9.1.2 DESCRIPTION

This is the program for the Beagle Board LCC/DCC/Railcom command station. It uses OpenMRN to interface with tha LCC/OpenLCB network to implement a LCC/OpenLCB aware command station node that can operate DCC/Railcom locomotives. It should work with any of the Beagle Board addon boards I have designed. It uses the AM335X's PRUs to generate the DCC signals. It ises the OpenMRN Console class to communicate over a Tcp/Ip channel with a Tcl/Tk coded GUI program to provide a user friendly point-and-click high level user interface.

9.1.3 OPTIONS

- -e EEPROM_file_path is the path to use to implement the EEProm device.
- -t Persistent_Train_file_path is the path to use to the implement the train persistent data.
- -u upstream_host is the host name for an upstream hub.
- -q upstream_port is the port number for the upstream hub.
- -c can_socketname is the name of the CAN socket.
- -M mainPRUfirmware is the path to the Main (PRU0) firmware
- -P progPRUfirmware is the path to the Prog (PRU1) firmware
- -W name:port Start a WiThrottle named name on port (if :port is ommited, on the default port).

The -u and -q options are only available if the program was built to support either a OpenLCB Tcp host or a GRIDCONNECT host. The -c option is only available if the program was built to support CAN Sockets.

9.1.4 PARAMETERS

None.

9.1.5 FILES

9.1.6 AUTHOR

Author

Robert Heller

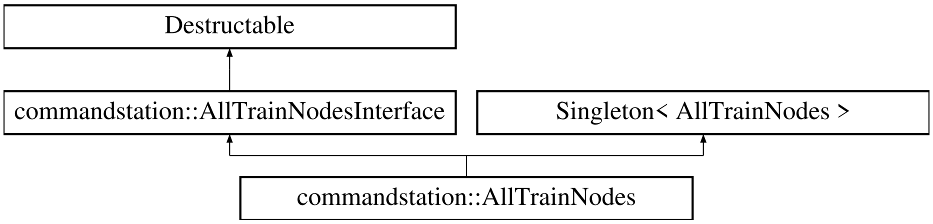
Date

29 Apr 2021

10 Class Documentation

10.1 commandstation::AllTrainNodes Class Reference

Inheritance diagram for commandstation::AllTrainNodes:


Destructable
commandstation::AllTrainNodesInterface
Singleton< AllTrainNodes >
commandstation::AllTrainNodes

Public Member Functions

- **AllTrainNodes** (TrainDb ∗db, openlcb::TrainService ∗traction_service, openlcb::SimpleInfoFlow ∗info_flow, openlcb::MemoryConfigHandler ∗memory_config, openlcb::MemorySpace ∗train_cdi, openlcb::MemorySpace ∗tmp_train_cdi)
- void remove_train_impl (uint32_t address)

  Removes a TrainImpl for the requested address if it exists.
- openlcb::TrainImpl ∗ **get_train_impl** (openlcb::NodeID id, bool allocate=true)
- openlcb::TrainImpl ∗ get_train_impl (DccMode drive_type, uint32_t address)

  Finds or creates a TrainImpl for the requested address and drive_type.
- std::shared_ptr< TrainDbEntry > get_traindb_entry (size_t id) override

  Returns a traindb entry or nullptr if the id is too high.
- openlcb::NodeID get_train_node_id (size_t id) override

  Returns a node id or 0 if the id is not known to be a train.
- openlcb::NodeID **get_train_node_id_ext** (size_t id, bool allocate=true)
- openlcb::NodeID allocate_node (DccMode drive_type, unsigned address) override

  Creates a new train node based on the given address and drive mode.
- size_t size ()

  Return the maximum number of locomotives currently being serviced.
- bool is_valid_train_node (openlcb::Node ∗node)
- bool is_valid_train_node (openlcb::NodeID node_id, bool allocate=true)

Private Member Functions

- Impl ∗ find_node (openlcb::Node ∗node)

  A child can look up if a local node is actually a Train node.
- Impl ∗ find_node (openlcb::NodeID node_id, bool allocate=true)

  Extension to the find_node implementation that exposes the option to not allocate a node when no existing node is found.
- Impl ∗ create_impl (int train_id, DccMode mode, int address)

  Helper function to create lok objects.

Private Attributes

- TrainDb ∗ **db_**
- openlcb::MemoryConfigHandler ∗ **memoryConfigService_**
- openlcb::MemorySpace ∗ **ro_train_cdi_**
- openlcb::MemorySpace ∗ **ro_tmp_train_cdi_**
- std::vector< Impl ∗ > trains_

  All train nodes that we know about.
- OSMutex trainsLock_

  Lock to protect trains_.
- std::unique_ptr< FindProtocolServer > **findProtocolServer_**
- std::unique_ptr< TrainSnipHandler > **snipHandler_**
- std::unique_ptr< TrainPipHandler > **pipHandler_**
- std::unique_ptr< TrainFDISpace > **fdiSpace_**
- std::unique_ptr< TrainConfigSpace > **configSpace_**
- std::unique_ptr< TrainCDISpace > **cdiSpace_**
- std::unique_ptr< TrainIdentifyHandler > **trainIdentHandler_**

Friends

- class **FindProtocolServer**
- class **TrainSnipHandler**
- class **TrainPipHandler**
- class **TrainFDISpace**
- class **TrainConfigSpace**
- class **TrainCDISpace**
- class **TrainIdentifyHandler**

Additional Inherited Members

10.1.1 Member Function Documentation

10.1.1.1 allocate_node()

```
openlcb::NodeID commandstation::AllTrainNodes::allocate_node (
            DccMode drive_type,
            unsigned address )  [override], [virtual]
```

Creates a new train node based on the given address and drive mode.

Parameters

drive_type	describes what kind of train node this should be
address	is the hardware (legacy) address

Returns

0 if the allocation fails (invalid arguments)

Implements commandstation::AllTrainNodesInterface.

10.1.1.2 create_impl()

```
Impl* commandstation::AllTrainNodes::create_impl (
             int train_id,
             DccMode mode,
             int address )  [private]
```

Helper function to create lok objects.

Adds a new Impl structure to impl_.

10.1.1.3 find_node() [1/2]

```
Impl* commandstation::AllTrainNodes::find_node (
             openlcb::Node * node )  [private]
```

A child can look up if a local node is actually a Train node.

If so, the Impl structure will be returned. If the node is not known (or not a train node maintained by this object), we return nullptr.

10.1.1.4 find_node() [2/2]

```
Impl* commandstation::AllTrainNodes::find_node (
             openlcb::NodeID node_id,
             bool allocate = true )  [private]
```

Extension to the find_node implementation that exposes the option to not allocate a node when no existing node is found.

10.1.1.5 get_train_impl()

```
openlcb::TrainImpl* commandstation::AllTrainNodes::get_train_impl (
             DccMode drive_type,
             uint32_t address )
```

Finds or creates a TrainImpl for the requested address and drive_type.

Parameters

drive_type	is the drive type for the loco to create if it doesn't exist.
address	is the legacy address of the loco to find or create.

10.1.1.6 is_valid_train_node() [1/2]

```
bool commandstation::AllTrainNodes::is_valid_train_node (
             openlcb::Node * node )
```

Returns

true if the provided node is a known/active train.

10.1.1.7 is_valid_train_node() [2/2]

```
bool commandstation::AllTrainNodes::is_valid_train_node (
             openlcb::NodeID node_id,
             bool allocate = true )
```

Returns

true if the provided node id is a known/active train.

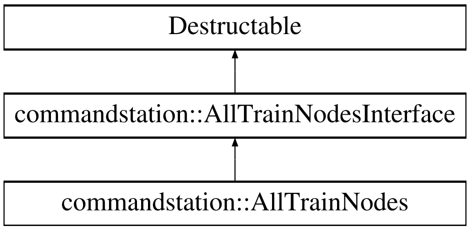
The documentation for this class was generated from the following file:

- AllTrainNodes.hxx

10.2 commandstation::AllTrainNodesInterface Class Reference

Abstract class for the AllTrainNodes that prevents pulling in transitive dependencies.

Inheritance diagram for commandstation::AllTrainNodesInterface:


Destructable
commandstation::AllTrainNodesInterface
commandstation::AllTrainNodes

Public Member Functions

- AllTrainNodesInterface (openlcb::TrainService ∗service)

  Constructor.
- openlcb::TrainService ∗ train_service ()
- virtual size_t size ()=0
- virtual std::shared_ptr< TrainDbEntry > get_traindb_entry (size_t index)=0
- virtual openlcb::NodeID get_train_node_id (size_t index)=0
- virtual openlcb::NodeID allocate_node (DccMode mode, unsigned address)=0

  Allocates a new legacy train node.

Protected Attributes

- openlcb::TrainService ∗ trainService_

  Pointer to the traction service instance. Externally owned.

10.2.1 Detailed Description

Abstract class for the AllTrainNodes that prevents pulling in transitive dependencies.

10.2.2 Constructor & Destructor Documentation

10.2.2.1 AllTrainNodesInterface()

```
commandstation::AllTrainNodesInterface::AllTrainNodesInterface (
             openlcb::TrainService ∗ service )  [inline]
```

Constructor.

Parameters

service	points to the traction service. Externally owned (ownership is not taken).

10.2.3 Member Function Documentation

10.2.3.1 allocate_node()

```
virtual openlcb::NodeID commandstation::AllTrainNodesInterface::allocate_node (
             DccMode mode,
             unsigned address )  [pure virtual]
```

Allocates a new legacy train node.

Parameters

mode	which protocol mode to use.
address	legacy address (to be interpreted for the given protocol mode).

Returns

the openlcb train node ID, or 0 if the arguments are not valid.

Implemented in commandstation::AllTrainNodes.

10.2.3.2 get_train_node_id()

```
virtual openlcb::NodeID commandstation::AllTrainNodesInterface::get_train_node_id (
            size_t index )  [pure virtual]
```

Returns

the openlcb train node ID for a given train index, or 0 if the train index is not valid.

Parameters

index	0..size() - 1.

Implemented in commandstation::AllTrainNodes.

10.2.3.3 get_traindb_entry()

```
virtual std::shared_ptr<TrainDbEntry> commandstation::AllTrainNodesInterface::get_traindb_entry (
            size_t index )  [pure virtual]
```

Returns

the train database entry for a given train index, or nullptr if the train index is not valid.

Parameters

index	0..size() - 1.

Implemented in commandstation::AllTrainNodes.

10.2.3.4 size()

```
virtual size_t commandstation::AllTrainNodesInterface::size ( )  [pure virtual]
```

Returns

maximum (or current) number of trains managed by this service. Trains are indexed 0..size().

Implemented in commandstation::AllTrainNodes.

10.2.3.5 train_service()

```
openlcb::TrainService* commandstation::AllTrainNodesInterface::train_service ( )  [inline]
```

Returns

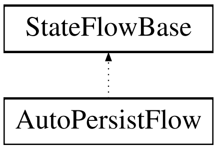
the traction service instance.

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

- AllTrainNodesInterface.hxx

10.3 AutoPersistFlow Class Reference

Inheritance diagram for AutoPersistFlow:



Public Member Functions

- **AutoPersistFlow** (Service ∗service, uint64_t interval, std::function< void(void)> callback)
- void **stop** ()

Private Member Functions

- StateFlowBase::Action **sleep_and_persist** ()
- StateFlowBase::Action **persist** ()

Private Attributes

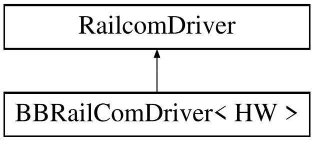
- StateFlowTimer **timer_** {this}
- uint64_t **interval_**
- std::function< void(void)> **callback_**

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

- AutoPersistCallbackFlow.hxx

10.4 BBRailComDriver< HW > Class Template Reference

Inheritance diagram for BBRailComDriver< HW >:



Public Types

- enum **RailComPhase** : uint8_t { **PRE_CUTOUT**, **CUTOUT_PHASE1**, **CUTOUT_PHASE2** }

Public Member Functions

- **BBRailComDriver** (size_t queue_size)
- void **hw_init** (dcc::RailcomHubFlow ∗hubFlow)
- void **disable_output** ()
- void **enable_output** ()
- void **start_cutout** () override
- size_t **rx_to_buf** (uint8_t ∗buf, size_t max_len)
- void **middle_cutout** () override
- void **end_cutout** () override
- void **no_cutout** ()
- void **set_feedback_key** (uint32_t key) override
- void **feedback_sample** () override
- RailComPhase **railcom_phase** ()
- dcc::RailcomHubData ∗ **railcom_buffer** ()
- void **advance_railcom_buffer** ()
- void **timer_tick** ()

Static Private Member Functions

- static void **railcom_timer_tick** (union sigval sv)

Private Attributes

- int **uart_fd_**
- uintptr_t **railcomFeedbackKey_** {0}
- dcc::RailcomHubFlow ∗ **railComHubFlow_**
- ExtendedRingBuffer< dcc::RailcomHubData > ∗ **railComFeedbackBuffer_**
- RailComPhase **railcomPhase_** {RailComPhase::PRE_CUTOUT}
- bool **enabled_** {false}
- timer_t **timerid_**

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

- BBRailComDriver.hxx

10.5 BeagleCS::BeaglePersistentTrainData Struct Reference

Public Member Functions

- **BeaglePersistentTrainData** (uint16_t address, std::string name="unknown", std::string description="", DccMode mode=DccMode::DCC_128)

Public Attributes

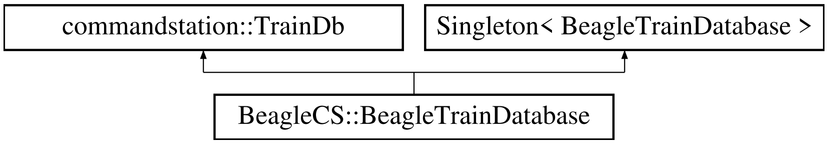
- uint16_t **address**
- std::string **name**
- std::string **description**
- bool **automatic_idle**
- bool **show_on_limited_throttles**
- uint8_t **mode**
- std::vector< uint8_t > **functions**

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

- BeagleTrainDatabase.hxx

10.6 BeagleCS::BeagleTrainDatabase Class Reference

Inheritance diagram for BeagleCS::BeagleTrainDatabase:


commandstation::TrainDb
Singleton< BeagleTrainDatabase >
BeagleCS::BeagleTrainDatabase

Public Member Functions

- **BeagleTrainDatabase** (openlcb::SimpleStackBase *stack)
- void **stop** ()
- size_t size () override
- int **get_index** (unsigned address)
- bool is_train_id_known (unsigned train_id) override

  Returns true if a train of a specific identifier is known to the traindb.
- bool is_train_id_known (openlcb::NodeID train_id) override

  Returns true if a train of a specific identifier is known to the traindb.
- std::shared_ptr< commandstation::TrainDbEntry > **create_if_not_found** (unsigned address, std::string name="unknown", std::string description="", DccMode mode=DccMode::DCC_128)
- void **delete_entry** (unsigned address)
- std::shared_ptr< commandstation::TrainDbEntry > get_entry (unsigned train_id) override

  Returns a train DB entry if the train ID is known, otherwise nullptr.

- std::shared_ptr< commandstation::TrainDbEntry > find_entry (openlcb::NodeID traction_node_id, unsigned hint=0) override

  Searches for an entry by the traction node ID.
- unsigned add_dynamic_entry (uint16_t address, DccMode mode) override

  Inserts a given entry into the train database.
- std::set< uint16_t > **get_default_train_addresses** (uint16_t limit)
- void **set_train_name** (unsigned address, std::string name)
- void **set_train_description** (unsigned address, std::string description)
- void **set_train_auto_idle** (unsigned address, bool idle)
- void **set_train_show_on_limited_throttle** (unsigned address, bool show)
- void **set_train_function_label** (unsigned address, uint8_t fn_id, Symbols label)
- void **set_train_drive_mode** (unsigned address, DccMode mode)
- std::string **get_all_entries_as_json** ()
- std::string **get_all_entries_as_list** ()
- std::string **get_entry_as_json** (unsigned address)
- DccMode **get_train_mode** (unsigned address)
- std::string **get_train_name** (unsigned address)
- std::string **get_train_description** (unsigned address)
- openlcb::MemorySpace ∗ **get_train_cdi** ()
- openlcb::MemorySpace ∗ **get_temp_train_cdi** ()
- void **persist** ()

Private Member Functions

- std::string **get_entry_as_json_locked** (unsigned address)

Private Attributes

- openlcb::SimpleStackBase ∗ **stack_**
- bool **entryDeleted_** {false}
- OSMutex **knownTrainsLock_**
- std::vector< std::shared_ptr< BeagleTrainDbEntry > > **knownTrains_**
- std::unique_ptr< openlcb::MemorySpace > **trainCdiFile_**
- std::unique_ptr< openlcb::MemorySpace > **tempTrainCdiFile_**
- uninitialized< AutoPersistFlow > **persistFlow_**

10.6.1 Member Function Documentation

10.6.1.1 add_dynamic_entry()

```
unsigned BeagleCS::BeagleTrainDatabase::add_dynamic_entry (
            uint16_t address,
            DccMode mode )  [override], [virtual]
```

Inserts a given entry into the train database.

Parameters

address	the locomotive address to create.
mode	the operating mode for the new locomotive.

Returns

the new train_id for the given entry.

Implements commandstation::TrainDb.

10.6.1.2 find_entry()

```
std::shared_ptr<commandstation::TrainDbEntry> BeagleCS::BeagleTrainDatabase::find_entry (
             openlcb::NodeID traction_node_id,
             unsigned hint = 0 )  [override], [virtual]
```

Searches for an entry by the traction node ID.

Returns nullptr if not found.

Parameters

hint	is a train_id that might be a match.

Implements commandstation::TrainDb.

10.6.1.3 get_entry()

```
std::shared_ptr<commandstation::TrainDbEntry> BeagleCS::BeagleTrainDatabase::get_entry (
             unsigned train_id )  [override], [virtual]
```

Returns a train DB entry if the train ID is known, otherwise nullptr.

The ownership of the entry is not transferred.

Implements commandstation::TrainDb.

10.6.1.4 is_train_id_known() [1/2]

```
bool BeagleCS::BeagleTrainDatabase::is_train_id_known (
             unsigned train_id )  [inline], [override], [virtual]
```

Returns true if a train of a specific identifier is known to the traindb.

Parameters

train↩ _id	is the train identifier. Valid values: anything. Typical values: 0..NUM_TRAINS

Implements commandstation::TrainDb.

10.6.1.5 is_train_id_known() [2/2]

```
bool BeagleCS::BeagleTrainDatabase::is_train_id_known (
             openlcb::NodeID train_id )  [override], [virtual]
```

Returns true if a train of a specific identifier is known to the traindb.

Parameters

train↩ _id	is the node id of the train being queried.

Implements commandstation::TrainDb.

10.6.1.6 size()

```
size_t BeagleCS::BeagleTrainDatabase::size ( )  [inline], [override], [virtual]
```

Returns

the number of traindb entries. The valid train IDs will then be 0 <= id < size().

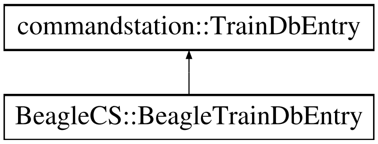
Implements commandstation::TrainDb.

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

- BeagleTrainDatabase.hxx

10.7 BeagleCS::BeagleTrainDbEntry Class Reference

Inheritance diagram for BeagleCS::BeagleTrainDbEntry:



Public Member Functions

- **BeagleTrainDbEntry** (BeaglePersistentTrainData, bool persist=true)
- std::string identifier () override

  Returns an internal identifier that uniquely defines where this traindb entry was allocated from.
- openlcb::NodeID get_traction_node () override

  Retrieves the NMRAnet NodeID for the virtual node that represents a particular train known to the database.
- std::string get_train_name () override

  Retrieves the name of the train.
- void **set_train_name** (std::string name)
- std::string get_train_description () override

  Retrieves the description of the train.
- void **set_train_description** (std::string description)
- uint16_t get_legacy_address () override

  Retrieves the legacy address of the train.
- void **set_legacy_address** (uint16_t address)
- DccMode get_legacy_drive_mode () override

  Retrieves the traction drive mode of the train.
- void **set_legacy_drive_mode** (DccMode mode)
- unsigned get_function_label (unsigned fn_id) override

  Retrieves the label assigned to a given function, or FN_NONEXISTANT if the function does not exist.
- void **set_function_label** (unsigned fn_id, Symbols label)
- int get_max_fn () override

  Returns the largest valid function ID for this train, or -1 if the train has no functions.
- int file_offset () override

  If non-negative, represents a file offset in the openlcb CONFIG_FILENAME file where this train has its data stored.
- void start_read_functions () override

  Notifies that we are going to read all functions.
- BeaglePersistentTrainData **get_data** ()
- void **set_auto_idle** (bool idle)
- void **set_show_on_limited_throttles** (bool show)
- bool **is_dirty** ()
- void **reset_dirty** (bool dirty=false)
- bool **is_persisted** ()
- bool **is_auto_idle** ()
- bool **is_show_on_limited_throttles** ()

Private Member Functions

- void **recalcuate_max_fn** ()

Private Attributes

- BeaglePersistentTrainData **data_**
- uint8_t **maxFn_**
- bool **dirty_**
- bool **persist_**

10.7.1 Member Function Documentation

10.7.1.1 file_offset()

```
int BeagleCS::BeagleTrainDbEntry::file_offset ( )  [override], [virtual]
```

If non-negative, represents a file offset in the openlcb CONFIG_FILENAME file where this train has its data stored.

Reimplemented from commandstation::TrainDbEntry.

10.7.1.2 get_function_label()

```
unsigned BeagleCS::BeagleTrainDbEntry::get_function_label (
            unsigned fn_id )  [override], [virtual]
```

Retrieves the label assigned to a given function, or FN_NONEXISTANT if the function does not exist.

Implements commandstation::TrainDbEntry.

10.7.1.3 get_legacy_address()

```
uint16_t BeagleCS::BeagleTrainDbEntry::get_legacy_address ( )  [inline], [override], [virtual]
```

Retrieves the legacy address of the train.

Implements commandstation::TrainDbEntry.

10.7.1.4 get_legacy_drive_mode()

```
DccMode BeagleCS::BeagleTrainDbEntry::get_legacy_drive_mode ( )  [inline], [override], [virtual]
```

Retrieves the traction drive mode of the train.

Implements commandstation::TrainDbEntry.

10.7.1.5 get_max_fn()

```
int BeagleCS::BeagleTrainDbEntry::get_max_fn ( )  [inline], [override], [virtual]
```

Returns the largest valid function ID for this train, or -1 if the train has no functions.

Implements commandstation::TrainDbEntry.

10.7.1.6 get_train_description()

```
std::string BeagleCS::BeagleTrainDbEntry::get_train_description ( )  [inline], [override], [virtual]
```

Retrieves the description of the train.

Implements commandstation::TrainDbEntry.

10.7.1.7 get_train_name()

```
std::string BeagleCS::BeagleTrainDbEntry::get_train_name ( )  [inline], [override], [virtual]
```

Retrieves the name of the train.

Implements commandstation::TrainDbEntry.

10.7.1.8 identifier()

```
std::string BeagleCS::BeagleTrainDbEntry::identifier ( )  [override], [virtual]
```

Returns an internal identifier that uniquely defines where this traindb entry was allocated from.

Implements commandstation::TrainDbEntry.

10.7.1.9 start_read_functions()

```
void BeagleCS::BeagleTrainDbEntry::start_read_functions ( )  [inline], [override], [virtual]
```

Notifies that we are going to read all functions.

Sometimes a re-initialization is helpful at this point.

Implements commandstation::TrainDbEntry.

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

- BeagleTrainDatabase.hxx

10.8 CDIHelper Class Reference

Static Public Member Functions

- template<class ConfigDef >
  static void create_config_descriptor_xml (const ConfigDef &config, const char *filename, openlcb::SimpleStack↩ Base *stack=nullptr)

  Creates the XML representation of the configuration structure and saves it to a file on the filesystem.

10.8.1 Member Function Documentation

10.8.1.1 create_config_descriptor_xml()

```
template<class ConfigDef >
static void CDIHelper::create_config_descriptor_xml (
              const ConfigDef & config,
              const char * filename,
              openlcb::SimpleStackBase * stack = nullptr )  [inline], [static]
```

Creates the XML representation of the configuration structure and saves it to a file on the filesystem.

Must be called after SPIFFS.begin() but before calling the {} method. The config file will be re-written whenever there was a change in the contents. It is also necessary to declare the static compiled-in CDI to be empty: namespace openlcb { // This will stop openlcb from exporting the CDI memory space // upon start. extern const char CDI_DATA[] = ""; } // namespace openlcb cfg is the global configuration instance (usually called cfg). filename is where the xml file can be stored on the filesystem. For example "/spiffs/cdi.xml".

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

- CDIHelper.hxx

10.9 DCCPacket::cmd_t Struct Reference

Specifies the meaning of the command byte for meta-commands to send.

Public Attributes

- uint8_t is_pkt: 1

  Always 1.
- uint8_t cmd: 7

  Command identifier.

10.9.1 Detailed Description

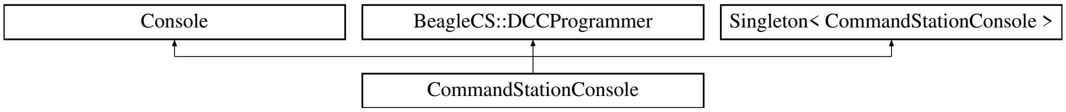
Specifies the meaning of the command byte for meta-commands to send.

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

- dccpacket.h

10.10 CommandStationConsole Class Reference

Inheritance diagram for CommandStationConsole:



Public Member Functions

- **CommandStationConsole** (openlcb::SimpleStackBase ∗stack, openlcb::TrainService ∗tractionService, ExecutorBase ∗executor, uint16_t port)
- **CommandStationConsole** (openlcb::SimpleStackBase ∗stack, openlcb::TrainService ∗tractionService, ExecutorBase ∗executor, int fd_in, int fd_out, int port=-1)

Static Public Member Functions

- static void **Begin** (openlcb::SimpleStackBase ∗stack, openlcb::TrainService ∗tractionService, const HBridgeControlConfig &maincfg, const HBridgeControlConfig &progcfg, const FanControlConfig &fancfg, const char ∗mainPRUfirmware="MainTrackDCC.out", const char ∗progPRUfirmware="ProgTrackDCC.out")
- static void **initiate_estop** ()

Private Member Functions

- CommandStatus **define_command** (FILE ∗fp, int argc, const char ∗argv[ ])
- CommandStatus **undefine_command** (FILE ∗fp, int argc, const char ∗argv[ ])
- CommandStatus **list_command** (FILE ∗fp, int argc, const char ∗argv[ ])
- CommandStatus **describe_command** (FILE ∗fp, int argc, const char ∗argv[ ])
- CommandStatus **status_command** (FILE ∗fp, int argc, const char ∗argv[ ])
- CommandStatus **readcv_command** (FILE ∗fp, int argc, const char ∗argv[ ])
- CommandStatus **readcvword_command** (FILE ∗fp, int argc, const char ∗argv[ ])
- CommandStatus **writeprogcvbyte_command** (FILE ∗fp, int argc, const char ∗argv[ ])
- CommandStatus **writeprogcvword_command** (FILE ∗fp, int argc, const char ∗argv[ ])
- CommandStatus **writeprogcvbit_command** (FILE ∗fp, int argc, const char ∗argv[ ])
- CommandStatus **writeopscvbyte_command** (FILE ∗fp, int argc, const char ∗argv[ ])
- CommandStatus **writeopscvbit_command** (FILE ∗fp, int argc, const char ∗argv[ ])
- void **putTclBraceString** (FILE ∗fp, const char ∗s) const

Static Private Member Functions

- static bool **is_ops_track_output_enabled** ()
- static void **enable_ops_track_output** ()
- static void **disable_ops_track_output** ()
- static void **disable_track_outputs** ()
- static void **enable_prog_track_output** ()
- static void **disable_prog_track_output** ()
- static CommandStatus **define_command** (FILE ∗fp, int argc, const char ∗argv[ ], void ∗context)
- static CommandStatus **undefine_command** (FILE ∗fp, int argc, const char ∗argv[ ], void ∗context)
- static CommandStatus **list_command** (FILE ∗fp, int argc, const char ∗argv[ ], void ∗context)
- static CommandStatus **describe_command** (FILE ∗fp, int argc, const char ∗argv[ ], void ∗context)
- static CommandStatus **status_command** (FILE ∗fp, int argc, const char ∗argv[ ], void ∗context)
- static CommandStatus **power_command** (FILE ∗fp, int argc, const char ∗argv[ ], void ∗context)
- static CommandStatus **estop_command** (FILE ∗fp, int argc, const char ∗argv[ ], void ∗context)
- static CommandStatus **shutdown_command** (FILE ∗fp, int argc, const char ∗argv[ ], void ∗context)
- static CommandStatus **readcv_command** (FILE ∗fp, int argc, const char ∗argv[ ], void ∗context)
- static CommandStatus **readcvword_command** (FILE ∗fp, int argc, const char ∗argv[ ], void ∗context)
- static CommandStatus **writeprogcvbyte_command** (FILE ∗fp, int argc, const char ∗argv[ ], void ∗context)
- static CommandStatus **writeprogcvword_command** (FILE ∗fp, int argc, const char ∗argv[ ], void ∗context)
- static CommandStatus **writeprogcvbit_command** (FILE ∗fp, int argc, const char ∗argv[ ], void ∗context)
- static CommandStatus **writeopscvbyte_command** (FILE ∗fp, int argc, const char ∗argv[ ], void ∗context)
- static CommandStatus **writeopscvbit_command** (FILE ∗fp, int argc, const char ∗argv[ ], void ∗context)

Private Attributes

- openlcb::SimpleStackBase ∗ **stack_**
- openlcb::TrainService ∗ **traction_service_**

Static Private Attributes

- static std::unique_ptr< HBridgeControl > **mains**
- static std::unique_ptr< HBridgeControl > **progtrack**
- static std::unique_ptr< FanControl > **fan**
- static std::unique_ptr< openlcb::RefreshLoop > **cs_poller**
- static std::unique_ptr< dcc::RailcomHubFlow > **railcom_hub**
- static std::unique_ptr< dcc::RailcomPrintfFlow > **railcom_dumper**
- static std::unique_ptr< BeagleCS::BeagleTrainDatabase > **trainDb**
- static std::unique_ptr< commandstation::AllTrainNodes > **trainNodes**
- static std::unique_ptr< CommandStationDCCPRUTrack< 0 > > **mainDCC**
- static std::unique_ptr< CommandStationDCCPRUTrack< 1 > > **progDCC**
- static std::unique_ptr< BeagleCS::DuplexedTrackIf > **track**
- static std::unique_ptr< dcc::SimpleUpdateLoop > **dccUpdateLoop**
- static std::unique_ptr< PoolToQueueFlow< Buffer< dcc::Packet > > > **pool_translator**
- static std::unique_ptr< ProgrammingTrackBackend > **prog_track_backend**
- static std::unique_ptr< BeagleCS::EStopHandler > **estop_handler**

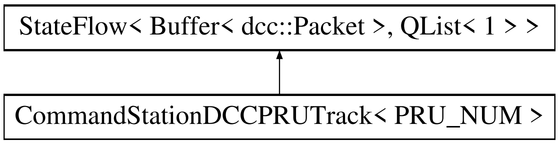
Additional Inherited Members

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

- CommandStationConsole.hxx

10.11 CommandStationDCCPRUTrack< PRU_NUM > Class Template Reference

Inheritance diagram for CommandStationDCCPRUTrack< PRU_NUM >:


StateFlow< Buffer< dcc::Packet >, QList< 1 > >
CommandStationDCCPRUTrack< PRU_NUM >

Public Member Functions

- **CommandStationDCCPRUTrack** (Service ∗service, int pool_size, const char ∗firmwareName)
- FixedPool ∗ **pool** () OVERRIDE

Static Public Attributes

- static constexpr const uint8_t **PRU** = PRU_NUM

Protected Member Functions

- Action **entry** () OVERRIDE
- Action finish ()

Protected Attributes

- FixedPool pool_

  Packet pool from which to allocate packets.

Private Member Functions

- void **StartPRU** (const char ∗firmwareName)

Private Attributes

- char **pruFirmware** [30]
- char **pruState** [36]
- char **pruMessageDevice** [20]

Static Private Attributes

- static constexpr char const ∗ **pruFirmwareFMT** = "/lib/firmware/am335x-pru%d-fw"
- static constexpr char const ∗ **pruStateFMT** = "/dev/remoteproc/pruss-core%d/state"
- static constexpr char const ∗ **pruMessageDeviceFMT** = "/dev/rpmsg_pru3%d"
- static bool **hasInstance_**

10.11.1 Member Function Documentation

10.11.1.1 finish()

```
template<uint8_t PRU_NUM>
Action CommandStationDCCPRUTrack< PRU_NUM >::finish ( )  [inline], [protected]
```

Returns

next action.

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

- CommandStationDCCPRUTrack.hxx

10.12 DCCPacket Struct Reference

Stores a DCC packet in memory.

Classes

- struct cmd_t

  Specifies the meaning of the command byte for meta-commands to send.
- struct pkt_t

  Specifies the meaning of the command byte for packets to send.

Public Attributes

- struct DCCPacket::pkt_t **pkt**
- struct DCCPacket::cmd_t **cmd**
-
  ```
  union {
     uint8_t header_raw_data
     struct pkt_t packet_header
     struct cmd_t command_header
  };
  ```
- uint8_t dlc

  Specifies the number of used payload bytes.
- uint8_t payload [DCC_PACKET_MAX_PAYLOAD]

  Packet payload bytes.
- uintptr_t feedback_key

  An opaque key used by the hardware driver to attribute feedback information to the source of the packet.

10.12.1 Detailed Description

Stores a DCC packet in memory.

Used to send data from the packet generation (usually the command station refresh loop flows) to the DCC track driver.

10.12.2 Member Data Documentation

10.12.2.1 dlc

```
uint8_t DCCPacket::dlc
```

Specifies the number of used payload bytes.

10.12.2.2 feedback_key

```
uintptr_t DCCPacket::feedback_key
```

An opaque key used by the hardware driver to attribute feedback information to the source of the packet.

This key will be sent back in the dcc::Feedback structure. If the key is non-zero it is guaranteed that some feedback (maybe empty) will be sent back after the packet is transmitted to the track.

10.12.2.3 payload

```
uint8_t DCCPacket::payload[DCC_PACKET_MAX_PAYLOAD]
```

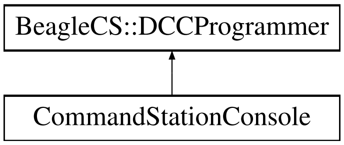
Packet payload bytes.

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

- dccpacket.h

10.13 BeagleCS::DCCProgrammer Class Reference

Inheritance diagram for BeagleCS::DCCProgrammer:



Public Types

- enum **CV_NAMES** {
  SHORT_ADDRESS = 1, **DECODER_VERSION** = 7, **DECODER_MANUFACTURER** = 8, **ACCESSORY_DEC**↩ **ODER_MSB_ADDRESS** = 9,
  LONG_ADDRESS_MSB_ADDRESS = 17, **LONG_ADDRESS_LSB_ADDRESS** = 18, **CONSIST_ADDRESS** = 19, **CONSIST_FUNCTION_CONTROL_F1_F8** = 21,
  CONSIST_FUNCTION_CONTROL_FL_F9_F12 = 22, **DECODER_CONFIG** = 29 }
- enum **DECODER_CONFIG_BITS** {
  LOCOMOTIVE_DIRECTION = 0, **FL_CONTROLLED_BY_SPEED** = 1, **POWER_CONVERSION** = 2, **BIDIRE**↩ **CTIONAL_COMMUNICATION** = 3,
  SPEED_TABLE = 4, **SHORT_OR_LONG_ADDRESS** = 5, **ACCESSORY_ADDRESS_MODE** = 6, **DECODER**↩ **_TYPE** = 7 }
- enum **CONSIST_FUNCTION_CONTROL_F1_F8_BITS** {
  F1_BIT = 0, **F2_BIT** = 1, **F3_BIT** = 2, **F4_BIT** = 3,
  F5_BIT = 4, **F6_BIT** = 5, **F7_BIT** = 6, **F8_BIT** = 7 }
- enum **CONSIST_FUNCTION_CONTROL_FL_F9_F12_BITS** {
  FL_BIT = 0, **F9_BIT** = 1, **F10_BIT** = 2, **F11_BIT** = 3,
  F12_BIT = 4 }

Public Member Functions

- int16_t **readCV** (const uint16_t)
- bool **writeProgCVByte** (const uint16_t, const uint8_t)
- bool **writeProgCVBit** (const uint16_t, const uint8_t, const bool)
- void **writeOpsCVByte** (const uint16_t, const uint16_t, const uint8_t)
- void **writeOpsCVBit** (const uint16_t, const uint16_t, const uint8_t, const bool)

Static Public Attributes

- static constexpr uint8_t **CONSIST_ADDRESS_REVERSED_ORIENTATION** = 0x80
- static constexpr uint8_t **CONSIST_ADDRESS_NO_ADDRESS** = 0x00

Private Member Functions

- bool **enterServiceMode** ()
- void **leaveServiceMode** ()
- bool **sendServiceModeDecoderReset** ()
- bool **sendServiceModePacketWithAck** (dcc::Packet pkt)
- bool **executeProgTrackWriteRequest** (dcc::Packet pkt)

Static Private Attributes

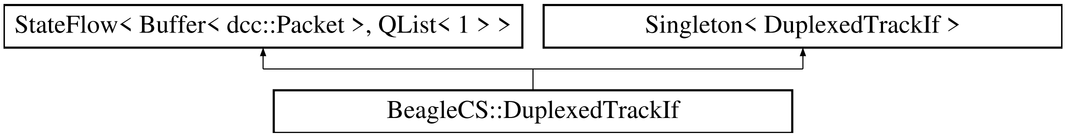
- static constexpr uint8_t **PROG_TRACK_CV_ATTEMPTS** = 3

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

- DCCProgrammer.hxx

10.14 BeagleCS::DuplexedTrackIf Class Reference

Inheritance diagram for BeagleCS::DuplexedTrackIf:


StateFlow< Buffer< dcc::Packet >, QList< 1 > >
Singleton< DuplexedTrackIf >
BeagleCS::DuplexedTrackIf

Public Member Functions

- **DuplexedTrackIf** (Service ∗service, int pool_size, CommandStationDCCMainTrack ∗ops, CommandStationDCCProgTrack ∗prog)
- FixedPool ∗ **pool** () override

Protected Member Functions

- Action **entry** () override
- Action **finish** ()

Protected Attributes

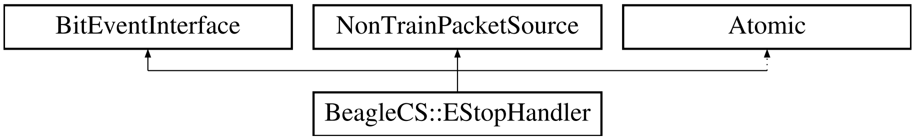
- CommandStationDCCMainTrack ∗ **ops_**
- CommandStationDCCProgTrack ∗ **prog_**
- FixedPool **pool_**

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

- DuplexedTrackIf.hxx

10.15 BeagleCS::EStopHandler Class Reference

Inheritance diagram for BeagleCS::EStopHandler:



Public Member Functions

- **EStopHandler** (openlcb::Node ∗node)
- openlcb::EventState **get_current_state** () override
- void **set_state** (bool new_value) override
- void **get_next_packet** (unsigned code, dcc::Packet ∗packet)
- openlcb::Node ∗ **node** () override

Private Attributes

- openlcb::BitEventPC **pc_** {this}
- openlcb::Node ∗ **node_**
- int16_t **remaining_**

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

- EStopHandler.hxx

10.16 ExtendedRingBuffer< T > Class Template Reference

Implements an extended ring buffer.

Public Member Functions

- void destroy ()

  Destroy an existing ring buffer instance.
- size_t put (const T ∗buf, size_t items)

  Insert a number of items to the buffer.
- size_t get (T ∗buf, size_t items)

  remove a number of items from the buffer.
- size_t items ()

  Number of items in the buffer.
- size_t size ()

  Size of buffer in number of items.
- size_t space ()

  space left in buffer of buffer in number items.
- size_t advance (size_t items)

  Add a number of items to the buffer by advancing the writeIndex.
- size_t data_read_pointer (T ∗∗buf)

  Get a reference to the current location in the buffer for read.
- size_t data_write_pointer (T ∗∗buf)

  Get a reference to the current location in the buffer for write.

Static Public Member Functions

- static ExtendedRingBuffer ∗ create (size_t size)

  Factory method to create a ring buffer instance.

Private Member Functions

- ExtendedRingBuffer (size_t size)

  Constructor.
- ExtendedRingBuffer ()

  Default Constructor.
- ∼ExtendedRingBuffer ()

  Default destructor.
- **DISALLOW_COPY_AND_ASSIGN** (ExtendedRingBuffer)

Private Attributes

- size_t _size

  size in items of ring buffer
- size_t count

  total number of items in ring buffer
- size_t readIndex

  read index
- size_t writeIndex

  write index
- T data [ ]

  ring buffer data

10.16.1 Detailed Description

template<typename T>
class ExtendedRingBuffer< T >

Implements an extended ring buffer.

(Based on RingBuffer & DevicedBuffer.)

10.16.2 Constructor & Destructor Documentation

10.16.2.1 ExtendedRingBuffer()

```
template<typename T>
ExtendedRingBuffer< T >::ExtendedRingBuffer (
              size_t size )  [inline], [private]
```

Constructor.

Parameters

size	size in bytes for the ring buffer

10.16.3 Member Function Documentation

10.16.3.1 advance()

```
template<typename T>
size_t ExtendedRingBuffer< T >::advance (
              size_t items )  [inline]
```

Add a number of items to the buffer by advancing the writeIndex.

Parameters

items	total number of items to add

Returns

total number of items added

10.16.3.2 create()

```
template<typename T>
static ExtendedRingBuffer* ExtendedRingBuffer< T >::create (
              size_t size )  [inline], [static]
```

Factory method to create a ring buffer instance.

Parameters

size	size in items for the ring buffer

Returns

the newly cleated ExtendedRingBuffer object.

10.16.3.3 data_read_pointer()

```
template<typename T>
size_t ExtendedRingBuffer< T >::data_read_pointer (
             T ** buf )  [inline]
```

Get a reference to the current location in the buffer for read.

Parameters

buf	location to store resulting reference

Returns

number of items in continuous memory. May be less than total number of items in the buffer.

10.16.3.4 data_write_pointer()

```
template<typename T>
size_t ExtendedRingBuffer< T >::data_write_pointer (
             T ** buf )  [inline]
```

Get a reference to the current location in the buffer for write.

Parameters

buf	location to store resulting reference

Returns

amount of space in continuous memory. May be less than total amount of space avaiable.

10.16.3.5 get()

```
template<typename T>
size_t ExtendedRingBuffer< T >::get (
             T * buf,
             size_t items )  [inline]
```

remove a number of items from the buffer.

Parameters

buf	reference to the data removed
items	total number of items to remove

Returns

total number of items removed

Todo (Stuart Baker) significant optimization opportunity

10.16.3.6 items()

```
template<typename T>
size_t ExtendedRingBuffer< T >::items ( ) [inline]
```

Number of items in the buffer.

Returns

number of items in the buffer

10.16.3.7 put()

```
template<typename T>
size_t ExtendedRingBuffer< T >::put (
            const T * buf,
            size_t items ) [inline]
```

Insert a number of items to the buffer.

Parameters

buf	reference to the first item to insert
items	total number of items to insert

Returns

total number of items inserted

Todo (Stuart Baker) significant optimization opportunity

10.16.3.8 size()

```
template<typename T>
size_t ExtendedRingBuffer< T >::size ( ) [inline]
```

Size of buffer in number of items.

Returns

size of buffer in number of items

10.16.3.9 space()

```
template<typename T>
size_t ExtendedRingBuffer< T >::space ( )  [inline]
```

space left in buffer of buffer in number items.

Returns

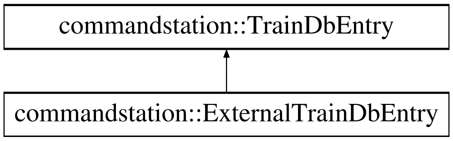
space left in buffer in number of items

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

- ExtendedRingBuffer.hxx

10.17 commandstation::ExternalTrainDbEntry Class Reference

Inheritance diagram for commandstation::ExternalTrainDbEntry:


commandstation::TrainDbEntry
commandstation::ExternalTrainDbEntry

Public Member Functions

- **ExternalTrainDbEntry** (const string &name, int address, DccMode mode=DCC_28, const string &descr="")
- string identifier () override

  Returns an internal identifier that uniquely defines where this traindb entry was allocated from.
- openlcb::NodeID get_traction_node () override

  Retrieves the NMRAnet NodeID for the virtual node that represents a particular train known to the database.
- string get_train_name () override

  Retrieves the name of the train.
- string get_train_description () override

  Retrieves the description of the train.
- uint16_t get_legacy_address () override

  Retrieves the legacy address of the train.
- DccMode get_legacy_drive_mode () override

  Retrieves the traction drive mode of the train.
- unsigned get_function_label (unsigned fn_id) override

  Retrieves the label assigned to a given function, or FN_NONEXISTANT if the function does not exist.
- int get_max_fn () override

  Returns the largest valid function ID for this train, or -1 if the train has no functions.
- void start_read_functions () override

  Setup for get_max_fn().

Public Attributes

- string **name_**
- string **descr_**
- int **address_**
- DccMode **mode_**

10.17.1 Member Function Documentation

10.17.1.1 get_function_label()

```
unsigned commandstation::ExternalTrainDbEntry::get_function_label (
             unsigned fn_id )  [inline], [override], [virtual]
```

Retrieves the label assigned to a given function, or FN_NONEXISTANT if the function does not exist.

Implements commandstation::TrainDbEntry.

10.17.1.2 get_legacy_address()

```
uint16_t commandstation::ExternalTrainDbEntry::get_legacy_address ( )  [inline], [override], [virtual]
```

Retrieves the legacy address of the train.

Implements commandstation::TrainDbEntry.

10.17.1.3 get_legacy_drive_mode()

```
DccMode commandstation::ExternalTrainDbEntry::get_legacy_drive_mode ( )  [inline], [override],
[virtual]
```

Retrieves the traction drive mode of the train.

Implements commandstation::TrainDbEntry.

10.17.1.4 get_max_fn()

```
int commandstation::ExternalTrainDbEntry::get_max_fn ( )  [inline], [override], [virtual]
```

Returns the largest valid function ID for this train, or -1 if the train has no functions.

Implements commandstation::TrainDbEntry.

10.17.1.5 get_train_description()

```
string commandstation::ExternalTrainDbEntry::get_train_description ( )  [inline], [override],
[virtual]
```

Retrieves the description of the train.

Implements commandstation::TrainDbEntry.

10.17.1.6 get_train_name()

```
string commandstation::ExternalTrainDbEntry::get_train_name ( )  [inline], [override], [virtual]
```

Retrieves the name of the train.

Implements commandstation::TrainDbEntry.

10.17.1.7 identifier()

```
string commandstation::ExternalTrainDbEntry::identifier ( )  [inline], [override], [virtual]
```

Returns an internal identifier that uniquely defines where this traindb entry was allocated from.

Implements commandstation::TrainDbEntry.

10.17.1.8 start_read_functions()

```
void commandstation::ExternalTrainDbEntry::start_read_functions ( )  [inline], [override], [virtual]
```

Setup for get_max_fn().

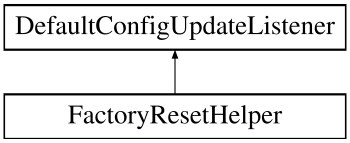
Implements commandstation::TrainDbEntry.

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

- ExternalTrainDbEntry.hxx

10.18 FactoryResetHelper Class Reference

Inheritance diagram for FactoryResetHelper:



Public Member Functions

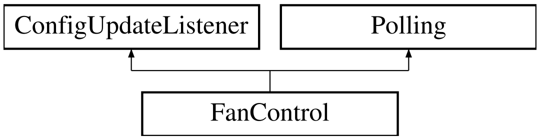
- UpdateAction **apply_configuration** (int fd, bool initial_load, BarrierNotifiable ∗done) OVERRIDE
- void **factory_reset** (int fd) override

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

- main.cxx

10.19 FanControl Class Reference

Inheritance diagram for FanControl:



Public Member Functions

- **FanControl** (openlcb::Node ∗node, const FanControlConfig &cfg, uint8_t temperatureAIN, const Gpio ∗fanGpio)
- template< class FAN >
  FanControl (openlcb::Node ∗node, const FanControlConfig &cfg, uint8_t temperatureAIN, const FAN &, const Gpio ∗fanGpio=FAN::instance())
- virtual void **poll_33hz** (openlcb::WriteHelper ∗helper, Notifiable ∗done)
- virtual UpdateAction **apply_configuration** (int fd, bool initial_load, BarrierNotifiable ∗done)
- virtual void **factory_reset** (int fd)
- bool **FanOn** () const
- bool **AlarmOn** () const
- openlcb::Polling ∗ **polling** ()
- uint32_t **getLastReading** ()

Private Attributes

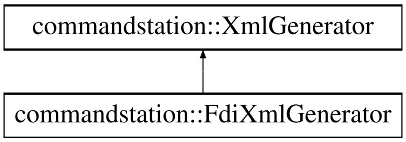
- openlcb::Node ∗ **node_**
- const FanControlConfig **cfg_**
- uint8_t **temperatureAIN_**
- const Gpio ∗ **fanGpio_**
- uint16_t **alarmthresh_** {350}
- uint16_t **fanthresh_** {250}
- openlcb::MemoryBit< uint8_t > **alarmBit_**
- openlcb::MemoryBit< uint8_t > **fanBit_**
- openlcb::BitEventProducer **alarmProducer_**
- openlcb::BitEventProducer **fanProducer_**
- uint8_t **fanon_** {0}
- uint8_t **alarmon_** {0}
- uint16_t **lastReading_** {0}

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

- FanControl.hxx

10.20 commandstation::FdiXmlGenerator Class Reference

Inheritance diagram for commandstation::FdiXmlGenerator:



Public Member Functions

- void reset (std::shared_ptr< TrainDbEntry > entry)

  Call this after the lokdb on entry was overwritten with the new loco's data.

Private Types

- enum **State** {
  STATE_START = 0, **STATE_XMLHEAD** = STATE_START, **STATE_START_FN**, **STATE_FN_NAME**,
  STATE_FN_NUMBER, **STATE_FN_END**, **STATE_NO_MORE_FN**, **STATE_EOF** }

Private Member Functions

- void generate_more () override

  This function will be called repeatedly in order to fill in the output buffer.

Private Attributes

- State **state_**
- std::shared_ptr< TrainDbEntry > **entry_**
- int **nextFunction_**

Additional Inherited Members

10.20.1 Member Function Documentation

10.20.1.1 generate_more()

```
void commandstation::FdiXmlGenerator::generate_more ( )  [override], [private], [virtual]
```

This function will be called repeatedly in order to fill in the output buffer.

Each call must call add_to_output at least once unless the EOF is reached.

Implements commandstation::XmlGenerator.

10.20.1.2 reset()

```
void commandstation::FdiXmlGenerator::reset (
            std::shared_ptr< TrainDbEntry > entry )
```

Call this after the lokdb on entry was overwritten with the new loco's data.

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

- FdiXmlGenerator.hxx

10.21 commandstation::FindProtocolDefs Struct Reference

Public Types

- enum { **TRAIN_FIND_BASE** = 0x090099FF00000000U }
- enum {
  TRAIN_FIND_MASK = 32, **TRAIN_FIND_MASK_LOW** = 8, **ALLOCATE** = 0x80, **EXACT** = 0x40,
  ADDRESS_ONLY = 0x20, **MATCH_ANY** = 0x01 }
- enum {
  NIBBLE_UNUSED = 0xf, **NIBBLE_SPACE** = 0xe, **NIBBLE_STAR** = 0xd, **NIBBLE_QN** = 0xc,
  NIBBLE_HASH = 0xb }

Static Public Member Functions

- static bool is_find_event (openlcb::EventId event)
- static bool match_event_to_drive_mode (openlcb::EventId event, DccMode mode)

  Compares an incoming search query's drive mode bits to an actual drive mode of a locomotive.
- static uint8_t match_query_to_node (openlcb::EventId event, TrainDbEntry ∗train)

  Compares an incoming search query to a given train node.
- static uint8_t match_query_to_train (openlcb::EventId event, const string &name, unsigned address, DccMode mode)

  Compares an incoming search query to a train node described by the major parameters only.
- static unsigned query_to_address (openlcb::EventId query, DccMode ∗mode)

  Converts a find protocol query to an address and desired DccMode information.
- static openlcb::EventId address_to_query (unsigned address, bool exact, DccMode mode)

  Translates an address as punched in by a (dumb) throttle to a query to issue on the OpenLCB bus as a find protocol request.
- static openlcb::EventId input_to_search (const string &input)

  Translates a sequence of input digits punched in by a throttle to a query to issue on the OpenLCB bus as a find protocol request.
- static openlcb::EventId input_to_allocate (const string &input)

  Translates a sequence of input digits punched in by a throttle to an allocate request to issue on the OpenLCB bus.

Static Public Attributes

- static uint8_t DEFAULT_DRIVE_MODE

    Specifies what kind of train to allocate when the drive mode is left as default / unspecified.
- static uint8_t DEFAULT_MARKLIN_DRIVE_MODE

    Specifies what kind of train to allocate when the drive mode is set as MARKLIN_ANY.
- static uint8_t DEFAULT_DCC_DRIVE_MODE

    Specifies what kind of train to allocate when the drive mode is set as DCC_ANY.

Static Private Member Functions

- static openlcb::EventId input_to_event (const string &input)

    Helper function for the input_to_* calls.

10.21.1 Member Function Documentation

10.21.1.1 address_to_query()

```
static openlcb::EventId commandstation::FindProtocolDefs::address_to_query (
             unsigned address,
             bool exact,
             DccMode mode )  [static]
```

Translates an address as punched in by a (dumb) throttle to a query to issue on the OpenLCB bus as a find protocol request.

Parameters

address	is the numeric value that the user typed.
exact	should be true if only exact matches shall be retrieved.
mode	should be set most of the time to OLCBUSER to specify that we don't care about the address type, but can also be set to DCC_LONG_ADDRESS.

10.21.1.2 input_to_allocate()

```
static openlcb::EventId commandstation::FindProtocolDefs::input_to_allocate (
             const string & input )  [static]
```

Translates a sequence of input digits punched in by a throttle to an allocate request to issue on the OpenLCB bus.

Parameters

input	is the sequence of numbers that the user typed. This is expected to have form like '415' or '021' or '474014'. You can add a leading zero to force DCC long address, a trailing M to force a Marklin locomotive.

Returns

an event ID representing the search. This event ID will be zero if the user input is invalid.

10.21.1.3 input_to_search()

```
static openlcb::EventId commandstation::FindProtocolDefs::input_to_search (
             const string & input )  [static]
```

Translates a sequence of input digits punched in by a throttle to a query to issue on the OpenLCB bus as a find protocol request.

Parameters

input	is the sequence of numbers that the user typed. This is expected to have form like '415' or '021' or '474014'

Returns

an event ID representing the search. This event ID could be IS_TRAIN_EVENT.

10.21.1.4 is_find_event()

```
static bool commandstation::FindProtocolDefs::is_find_event (
             openlcb::EventId event )  [inline], [static]
```

Parameters

event	is an openlcb event ID

Returns

true if that event ID belong to the find protocol event range.

10.21.1.5 match_event_to_drive_mode()

```
static bool commandstation::FindProtocolDefs::match_event_to_drive_mode (
             openlcb::EventId event,
             DccMode mode )  [static]
```

Compares an incoming search query's drive mode bits to an actual drive mode of a locomotive.

Decides whether they match using tri-state logic, i.e. taking into account "no restriction" queries.

Parameters

event	the incoming query
mode	the drive mode of a locomotive

Returns

true if this locomotive matches the restrictions in the query (true if there were no restrictions in the query).

10.21.1.6 match_query_to_node()

```
static uint8_t commandstation::FindProtocolDefs::match_query_to_node (
             openlcb::EventId event,
             TrainDbEntry * train )  [static]
```

Compares an incoming search query to a given train node.

Returns 0 for a no-match. Returns a bitfield of match types for a match. valid bits are MATCH_ANY (always set), ADDRESS_ONLY (set when the match occurred in the address), EXACT (clear for prefix match).

10.21.1.7 match_query_to_train()

```
static uint8_t commandstation::FindProtocolDefs::match_query_to_train (
             openlcb::EventId event,
             const string & name,
             unsigned address,
             DccMode mode )  [static]
```

Compares an incoming search query to a train node described by the major parameters only.

mode should be set to 0 for ignore, or DCC_LONG_ADDRESS. Returns a bitfield of match types for a match. valid bits are MATCH_ANY (always set), ADDRESS_ONLY (set when the match occurred in the address), EXACT (clear for prefix match).

10.21.1.8 query_to_address()

```
static unsigned commandstation::FindProtocolDefs::query_to_address (
             openlcb::EventId query,
             DccMode * mode )  [static]
```

Converts a find protocol query to an address and desired DccMode information.

Will take into account prefix zeros for forcing a dcc long address, as well as all mode and flag bits coming in via the query.

Parameters

mode	(can't be null) will be filled in with the Dcc Mode: the bottom 3 bits as specified by the incoming query, or zero if the query did not specify a preference. If the query started with a prefix of zero (typed by the user) or DCC_FORCE_LONG_ADDRESS was set in the query, the DccMode will have the force long address bit set.

Returns

the new legacy_address.

10.21.2 Member Data Documentation

10.21.2.1 DEFAULT_DCC_DRIVE_MODE

```
uint8_t commandstation::FindProtocolDefs::DEFAULT_DCC_DRIVE_MODE  [static]
```

Specifies what kind of train to allocate when the drive mode is set as DCC_ANY.

10.21.2.2 DEFAULT_DRIVE_MODE

```
uint8_t commandstation::FindProtocolDefs::DEFAULT_DRIVE_MODE  [static]
```

Specifies what kind of train to allocate when the drive mode is left as default / unspecified.

10.21.2.3 DEFAULT_MARKLIN_DRIVE_MODE

```
uint8_t commandstation::FindProtocolDefs::DEFAULT_MARKLIN_DRIVE_MODE  [static]
```

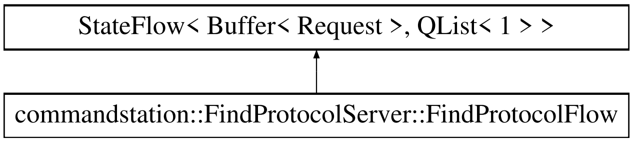
Specifies what kind of train to allocate when the drive mode is set as MARKLIN_ANY.

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

- FindProtocolDefs.hxx

10.22 commandstation::FindProtocolServer::FindProtocolFlow Class Reference

Inheritance diagram for commandstation::FindProtocolServer::FindProtocolFlow:


StateFlow< Buffer< Request >, QList< 1 > >
commandstation::FindProtocolServer::FindProtocolFlow

Public Member Functions

- **FindProtocolFlow** (FindProtocolServer ∗parent)
- Action **entry** () override
- Action **iterate** ()
- Action **send_response** ()
- Action **next_iterate** ()
- Action **iteration_done** ()
- Action wait_for_new_node ()

  Yields until the new node is initialized and we are allowed to send traffic out from it.
- Action **new_node_reply** ()
- Action **send_new_node_response** ()

Private Member Functions

- AllTrainNodesInterface ∗ **nodes** ()
- openlcb::If ∗ **iface** ()

Private Attributes

- FindProtocolServer ∗ **parent_**
- openlcb::EventId **eventId_**
- union {
    unsigned **nextTrainId_**
    openlcb::NodeID **newNodeId_**
  };
- BarrierNotifiable **bn_**
- bool hasMatches_: 1

  True if we found any matches during the iteration.
- bool isGlobal_: 1

  True if the current iteration has to touch every node.
- StateFlowTimer **timer_** {this}

10.22.1 Member Function Documentation

10.22.1.1 wait_for_new_node()

```
Action commandstation::FindProtocolServer::FindProtocolFlow::wait_for_new_node ( )  [inline]
```

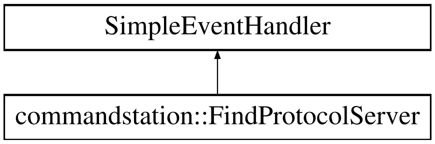
Yields until the new node is initialized and we are allowed to send traffic out from it.

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

- FindProtocolServer.hxx

10.23 commandstation::FindProtocolServer Class Reference

Inheritance diagram for commandstation::FindProtocolServer:



Classes

- class FindProtocolFlow
- struct Request

Public Member Functions

- **FindProtocolServer** (AllTrainNodesInterface ∗nodes)
- void **handle_identify_global** (const EventRegistryEntry &registry_entry, EventReport ∗event, BarrierNotifiable ∗done) override
- void **handle_identify_producer** (const EventRegistryEntry &registry_entry, EventReport ∗event, Barrier←Notifiable ∗done) override

Private Types

- enum { **REQUEST_GLOBAL_IDENTIFY** = 0x0001000000000000U, **IS_TRAIN_EVENT** = openlcb::Traction←Defs::IS_TRAIN_EVENT, **USER_ARG_FIND** = 1, **USER_ARG_ISTRAIN** = 2 }

Private Member Functions

- openlcb::If ∗ iface ()
- openlcb::TrainService ∗ service ()
- AllTrainNodesInterface ∗ nodes ()

Private Attributes

- AllTrainNodesInterface ∗ nodes_

  Pointer to the AllTrainNodes instance. Externally owned.
- uint8_t pendingGlobalIdentify_ {false}

  Set to true when a global identify message is received.
- uint8_t pendingIsTrain_ {false}

  Same as pendingGlobalIdentify_ for the IS_TRAIN event producer.
- FindProtocolFlow **flow_** {this}

10.23.1 Member Function Documentation

10.23.1.1 iface()

```
openlcb::If* commandstation::FindProtocolServer::iface ( )  [inline], [private]
```

Returns

the openlcb interface to which the train nodes (and the traction service) are bound.

10.23.1.2 nodes()

```
AllTrainNodesInterface* commandstation::FindProtocolServer::nodes ( )  [inline], [private]
```

Returns

the AllTrainNodes instance.

10.23.1.3 service()

```
openlcb::TrainService* commandstation::FindProtocolServer::service ( )  [inline], [private]
```

Returns

the openlcb Traction Service.

10.23.2 Member Data Documentation

10.23.2.1 pendingGlobalIdentify_

```
uint8_t commandstation::FindProtocolServer::pendingGlobalIdentify_ {false}  [private]
```

Set to true when a global identify message is received.

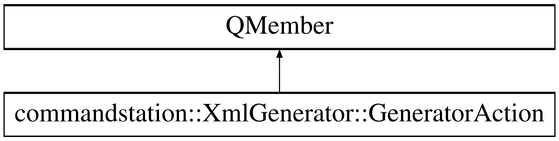
When a global identify starts processing, it shall be set to false. If a global identify request arrives with no pending←GlobalIdentify_, that is a duplicate request that can be ignored.

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

- FindProtocolServer.hxx

10.24 commandstation::XmlGenerator::GeneratorAction Struct Reference

Inheritance diagram for commandstation::XmlGenerator::GeneratorAction:



Public Attributes

- uint8_t **type**
-

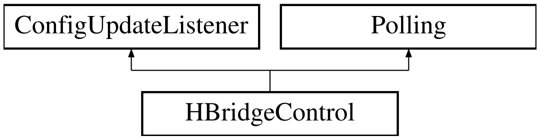
  union {
    const void ∗ **pointer**
    int **integer**
  };

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

- XmlGenerator.hxx

10.25 HBridgeControl Class Reference

Inheritance diagram for HBridgeControl:



Public Types

- enum **STATE** : uint8_t { **STATE_OVERCURRENT** = BIT(0), **STATE_SHUTDOWN** = BIT(1), **STATE_ON** = B↩ IT(2), **STATE_OFF** = BIT(3) }

Public Member Functions

- **HBridgeControl** (openlcb::Node ∗node, const HBridgeControlConfig &cfg, uint8_t currentAIN, const uint32_t limitMilliAmps, const uint32_t maxMilliAmps, const Gpio ∗enableGpio, const Gpio ∗thermFlagGpio=NULL)
- **HBridgeControl** (openlcb::Node ∗node, const HBridgeControlConfig &cfg, uint8_t currentAIN, const uint32_t maxMilliAmps, const Gpio ∗enableGpio, const Gpio ∗thermFlagGpio=NULL)
- virtual void **poll_33hz** (openlcb::WriteHelper ∗helper, Notifiable ∗done)
- virtual UpdateAction **apply_configuration** (int fd, bool initial_load, BarrierNotifiable ∗done)
- virtual void **factory_reset** (int fd)
- bool **EnabledP** () const
- bool **ThermalFlagP** () const
- bool **OverCurrentP** () const
- openlcb::Polling ∗ **polling** ()
- uint32_t **getMaxMilliAmps** ()
- uint32_t **getLastReading** ()
- bool **isProgrammingTrack** ()
- void **enable_prog_response** (bool enable)

Private Attributes

- openlcb::Node ∗ **node_**
- const HBridgeControlConfig **cfg_**
- const uint8_t **currentAIN_**
- const uint8_t **adcSampleCount_** {32}
- const uint8_t **overCurrentRetryCount_** {3}
- const Gpio ∗ **enableGpio_**
- const Gpio ∗ **thermFlagGpio_**
- const uint32_t **maxMilliAmps_**
- const uint32_t **overCurrentLimit_**
- const uint32_t **shutdownLimit_**
- bool **isProgTrack_**
- const uint32_t **progAckLimit_**
- openlcb::MemoryBit< uint8_t > **shortBit_**
- openlcb::MemoryBit< uint8_t > **shutdownBit_**
- openlcb::MemoryBit< uint8_t > **thermalFlagBit_**
- openlcb::BitEventProducer **shortProducer_**
- openlcb::BitEventProducer **shutdownProducer_**
- openlcb::BitEventProducer **thermalFlagProducer_**
- bool **progEnable_** {false}
- uint8_t **state_** {STATE_OFF}
- uint8_t **overCurrentCheckCount_** {0}
- uint32_t **lastReading_** {0}
- uint8_t **thermalFlag_** {0}

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

- HBridgeControl.hxx

10.26 my_resource_table Struct Reference

Public Attributes

- struct resource_table **base**
- uint32_t **offset** [2]
- struct fw_rsc_vdev **rpmsg_vdev**
- struct fw_rsc_vdev_vring **rpmsg_vring0**
- struct fw_rsc_vdev_vring **rpmsg_vring1**
- struct fw_rsc_custom **pru_ints**

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

- resource_table_0.h
- resource_table_1.h
- resource_table_empty.h

10.27 DCCPacket::pkt_t Struct Reference

Specifies the meaning of the command byte for packets to send.

Public Attributes

- uint8_t is_pkt: 1

  Always 0.
- uint8_t is_marklin: 1

  0: DCC packet, 1: motorola packet.
- uint8_t skip_ec: 1

  typically for DCC packets: 1: do NOT append an EC byte to the end of the packet.
- uint8_t send_long_preamble: 1

  1: send long preamble instead of packet.
- uint8_t sense_ack: 1

  1: wait for service mode ack and report it back to the host.
- uint8_t rept_count: 2

  The packet will be sent 1 + rept_count times to the wire.
- uint8_t reserved: 1

  reserved for future use.

10.27.1 Detailed Description

Specifies the meaning of the command byte for packets to send.

10.27.2 Member Data Documentation

10.27.2.1 rept_count

```
uint8_t DCCPacket::pkt_t::rept_count
```

The packet will be sent 1 + rept_count times to the wire.

default: 0.

10.27.2.2 send_long_preamble

```
uint8_t DCCPacket::pkt_t::send_long_preamble
```

1: send long preamble instead of packet.

0: send normal preamble and pkt.

10.27.2.3 skip_ec

```
uint8_t DCCPacket::pkt_t::skip_ec
```

typically for DCC packets: 1: do NOT append an EC byte to the end of the packet.

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

- dccpacket.h

10.28 commandstation::FindProtocolServer::Request Struct Reference

Public Member Functions

- void **reset** (openlcb::EventId event)

Public Attributes

- EventId **event_**

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

- FindProtocolServer.hxx

10.29 commandstation::ProgrammingTrackSpaceConfig::Shadow Struct Reference

This shadow structure is declared to be parallel to the CDI entries.

Public Attributes

- uint32_t **mode**
- uint32_t **cv**
- uint32_t **value**
- uint32_t **bit_write_value**
- char **bit_value_string** [24]
- uint32_t **verify_repeats**
- uint32_t **verify_cooldown_repeats**

10.29.1 Detailed Description

This shadow structure is declared to be parallel to the CDI entries.

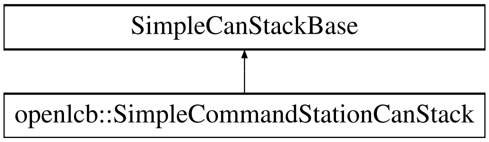
The documentation for this struct was generated from the following file:

- ProgrammingTrackSpaceConfig.hxx

10.30 openlcb::SimpleCommandStationCanStack Class Reference

CAN-based Command Station stack.

Inheritance diagram for openlcb::SimpleCommandStationCanStack:



Public Member Functions

- SimpleCommandStationCanStack (const openlcb::NodeID node_id)

  Creates a Traction Proxy OpenLCB stack.
- Node ∗ node () override
- TrainService ∗ **traction_service** ()

Private Member Functions

- void **start_node** () override

Private Attributes

- DefaultNode node_

  The actual node.
- ProtocolIdentificationHandler pipHandler_ {&node_, PIP_RESPONSE}

  Handles PIP requests.
- SNIPHandler snipHandler_ {iface(), &node_, &infoFlow_}

  Handles SNIP requests.
- TrainService **traction_service_**

Static Private Attributes

- static const auto **PIP_RESPONSE**

10.30.1 Detailed Description

CAN-based Command Station stack.

10.30.2 Member Function Documentation

10.30.2.1 node()

```
Node* openlcb::SimpleCommandStationCanStack::node ( )  [inline], [override]
```

Returns

the virtual node pointer of the main virtual node of the stack (as defined by the NodeID argument of the constructor).

10.30.3 Member Data Documentation

10.30.3.1 PIP_RESPONSE

```
const auto openlcb::SimpleCommandStationCanStack::PIP_RESPONSE  [static], [private]
```

Initial value:

```
= Defs::EVENT_EXCHANGE | Defs::DATAGRAM |
        Defs::MEMORY_CONFIGURATION | Defs::ABBREVIATED_DEFAULT_CDI |
        Defs::SIMPLE_NODE_INFORMATION | Defs::CDI
```

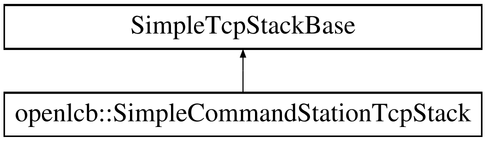
The documentation for this class was generated from the following file:

- CommandStationStack.hxx

10.31 openlcb::SimpleCommandStationTcpStack Class Reference

Tcp-based Command Station stack.

Inheritance diagram for openlcb::SimpleCommandStationTcpStack:



Public Member Functions

- **SimpleCommandStationTcpStack** (const openlcb::NodeID node_id)
- Node ∗ node () override
- TrainService ∗ **traction_service** ()

Private Member Functions

- void **start_node** () override

Private Attributes

- DefaultNode node_

  The actual node.
- ProtocolIdentificationHandler pipHandler_ {&node_, PIP_RESPONSE}

  Handles PIP requests.
- SNIPHandler snipHandler_ {iface(), &node_, &infoFlow_}

  Handles SNIP requests.
- openlcb::TrainService **traction_service_**

Static Private Attributes

- static const auto **PIP_RESPONSE**

10.31.1 Detailed Description

Tcp-based Command Station stack.

10.31.2 Member Function Documentation

10.31.2.1 node()

```
Node* openlcb::SimpleCommandStationTcpStack::node ( )  [inline], [override]
```

Returns

the virtual node pointer of the main virtual node of the stack (as defined by the NodeID argument of the constructor).

10.31.3 Member Data Documentation

10.31.3.1 PIP_RESPONSE

```
const auto openlcb::SimpleCommandStationTcpStack::PIP_RESPONSE  [static], [private]
```

Initial value:

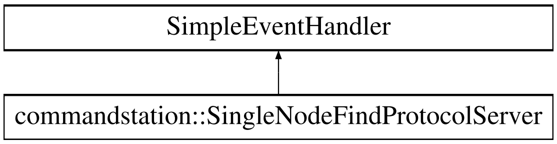
```
= Defs::EVENT_EXCHANGE | Defs::DATAGRAM |
        Defs::MEMORY_CONFIGURATION | Defs::ABBREVIATED_DEFAULT_CDI |
        Defs::SIMPLE_NODE_INFORMATION | Defs::CDI
```

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

- CommandStationStack.hxx

10.32 commandstation::SingleNodeFindProtocolServer Class Reference

Inheritance diagram for commandstation::SingleNodeFindProtocolServer:



Public Types

- using **Node** = openlcb::Node

Public Member Functions

- **SingleNodeFindProtocolServer** (Node ∗node, TrainDbEntry ∗db_entry)
- void **handle_identify_global** (const EventRegistryEntry &registry_entry, EventReport ∗event, BarrierNotifiable ∗done) override
- void **handle_identify_producer** (const EventRegistryEntry &registry_entry, EventReport ∗event, BarrierNotifiable ∗done) override

Private Attributes

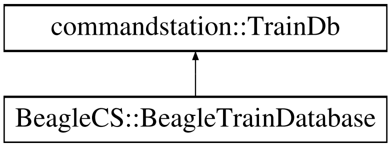
- Node ∗ **node_**
- TrainDbEntry ∗ **dbEntry_**

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

- FindProtocolServer.hxx

10.33 commandstation::TrainDb Class Reference

Inheritance diagram for commandstation::TrainDb:



Public Member Functions

- virtual size_t size ()=0
- virtual bool is_train_id_known (unsigned train_id)=0

  Returns true if a train of a specific identifier is known to the traindb.
- virtual bool is_train_id_known (openlcb::NodeID train_id)=0

  Returns true if a train of a specific identifier is known to the traindb.
- virtual std::shared_ptr< TrainDbEntry > get_entry (unsigned train_id)=0

  Returns a train DB entry if the train ID is known, otherwise nullptr.
- virtual std::shared_ptr< TrainDbEntry > find_entry (openlcb::NodeID traction_node_id, unsigned hint=0)=0

  Searches for an entry by the traction node ID.
- virtual unsigned add_dynamic_entry (uint16_t address, DccMode mode)=0

  Inserts a given entry into the train database.

10.33.1 Member Function Documentation

10.33.1.1 add_dynamic_entry()

```
virtual unsigned commandstation::TrainDb::add_dynamic_entry (
            uint16_t address,
            DccMode mode )  [pure virtual]
```

Inserts a given entry into the train database.

Parameters

address	the locomotive address to create.
mode	the operating mode for the new locomotive.

Returns

the new train_id for the given entry.

Implemented in BeagleCS::BeagleTrainDatabase.

10.33.1.2 find_entry()

```
virtual std::shared_ptr<TrainDbEntry> commandstation::TrainDb::find_entry (
                openlcb::NodeID traction_node_id,
                unsigned hint = 0 )  [pure virtual]
```

Searches for an entry by the traction node ID.

Returns nullptr if not found.

Parameters

hint	is a train_id that might be a match.

Implemented in BeagleCS::BeagleTrainDatabase.

10.33.1.3 get_entry()

```
virtual std::shared_ptr<TrainDbEntry> commandstation::TrainDb::get_entry (
                unsigned train_id )  [pure virtual]
```

Returns a train DB entry if the train ID is known, otherwise nullptr.

The ownership of the entry is not transferred.

Implemented in BeagleCS::BeagleTrainDatabase.

10.33.1.4 is_train_id_known() [1/2]

```
virtual bool commandstation::TrainDb::is_train_id_known (
                unsigned train_id )  [pure virtual]
```

Returns true if a train of a specific identifier is known to the traindb.

Parameters

train← _id	is the train identifier. Valid values: anything. Typical values: 0..NUM_TRAINS

Implemented in BeagleCS::BeagleTrainDatabase.

10.33.1.5 is_train_id_known() [2/2]

```
virtual bool commandstation::TrainDb::is_train_id_known (
            openlcb::NodeID train_id )  [pure virtual]
```

Returns true if a train of a specific identifier is known to the traindb.

Parameters

train← _id	is the node id of the train being queried.

Implemented in BeagleCS::BeagleTrainDatabase.

10.33.1.6 size()

```
virtual size_t commandstation::TrainDb::size ( )  [pure virtual]
```

Returns

the number of traindb entries. The valid train IDs will then be 0 <= id < size().

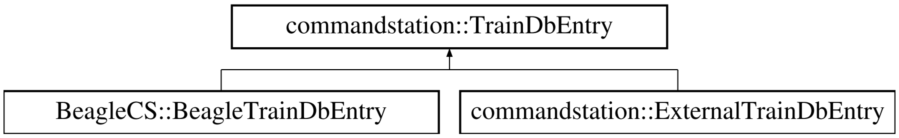
Implemented in BeagleCS::BeagleTrainDatabase.

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

- TrainDb.hxx

10.34 commandstation::TrainDbEntry Class Reference

Inheritance diagram for commandstation::TrainDbEntry:


commandstation::TrainDbEntry
BeagleCS::BeagleTrainDbEntry
commandstation::ExternalTrainDbEntry

Public Member Functions

- virtual string identifier ()=0

    Returns an internal identifier that uniquely defines where this traindb entry was allocated from.
- virtual openlcb::NodeID get_traction_node ()=0

    Retrieves the NMRAnet NodeID for the virtual node that represents a particular train known to the database.
- virtual string get_train_name ()=0

    Retrieves the name of the train.
- virtual string get_train_description ()=0

    Retrieves the description of the train.
- virtual uint16_t get_legacy_address ()=0

    Retrieves the legacy address of the train.
- virtual DccMode get_legacy_drive_mode ()=0

    Retrieves the traction drive mode of the train.
- virtual unsigned get_function_label (unsigned fn_id)=0

    Retrieves the label assigned to a given function, or FN_NONEXISTANT if the function does not exist.
- virtual int get_max_fn ()=0

    Returns the largest valid function ID for this train, or -1 if the train has no functions.
- virtual int file_offset ()

    If non-negative, represents a file offset in the openlcb CONFIG_FILENAME file where this train has its data stored.
- virtual void start_read_functions ()=0

    Notifies that we are going to read all functions.

10.34.1 Member Function Documentation

10.34.1.1 file_offset()

```
virtual int commandstation::TrainDbEntry::file_offset ( )  [inline], [virtual]
```

If non-negative, represents a file offset in the openlcb CONFIG_FILENAME file where this train has its data stored.

Reimplemented in BeagleCS::BeagleTrainDbEntry.

10.34.1.2 get_function_label()

```
virtual unsigned commandstation::TrainDbEntry::get_function_label (
                unsigned fn_id )  [pure virtual]
```

Retrieves the label assigned to a given function, or FN_NONEXISTANT if the function does not exist.

Implemented in BeagleCS::BeagleTrainDbEntry, and commandstation::ExternalTrainDbEntry.

10.34.1.3 get_legacy_address()

```
virtual uint16_t commandstation::TrainDbEntry::get_legacy_address ( )  [pure virtual]
```

Retrieves the legacy address of the train.

Implemented in BeagleCS::BeagleTrainDbEntry, and commandstation::ExternalTrainDbEntry.

10.34.1.4 get_legacy_drive_mode()

```
virtual DccMode commandstation::TrainDbEntry::get_legacy_drive_mode ( )  [pure virtual]
```

Retrieves the traction drive mode of the train.

Implemented in BeagleCS::BeagleTrainDbEntry, and commandstation::ExternalTrainDbEntry.

10.34.1.5 get_max_fn()

```
virtual int commandstation::TrainDbEntry::get_max_fn ( )  [pure virtual]
```

Returns the largest valid function ID for this train, or -1 if the train has no functions.

Implemented in BeagleCS::BeagleTrainDbEntry, and commandstation::ExternalTrainDbEntry.

10.34.1.6 get_train_description()

```
virtual string commandstation::TrainDbEntry::get_train_description ( )  [pure virtual]
```

Retrieves the description of the train.

Implemented in BeagleCS::BeagleTrainDbEntry, and commandstation::ExternalTrainDbEntry.

10.34.1.7 get_train_name()

```
virtual string commandstation::TrainDbEntry::get_train_name ( )  [pure virtual]
```

Retrieves the name of the train.

Implemented in BeagleCS::BeagleTrainDbEntry, and commandstation::ExternalTrainDbEntry.

10.34.1.8 identifier()

```
virtual string commandstation::TrainDbEntry::identifier ( )  [pure virtual]
```

Returns an internal identifier that uniquely defines where this traindb entry was allocated from.

Implemented in BeagleCS::BeagleTrainDbEntry, and commandstation::ExternalTrainDbEntry.

10.34.1.9 start_read_functions()

```
virtual void commandstation::TrainDbEntry::start_read_functions ( )  [pure virtual]
```

Notifies that we are going to read all functions.

Sometimes a re-initialization is helpful at this point.

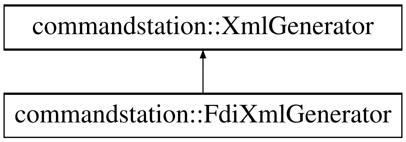
Implemented in BeagleCS::BeagleTrainDbEntry, and commandstation::ExternalTrainDbEntry.

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

- TrainDb.hxx

10.35 commandstation::XmlGenerator Class Reference

Inheritance diagram for commandstation::XmlGenerator:



Classes

- struct GeneratorAction

Public Member Functions

- ssize_t read (size_t offset, void ∗buf, size_t len)

  Reads from the buffer, or generates more data to read.
- size_t **file_offset** ()

Protected Member Functions

- virtual void generate_more ()=0

  This function will be called repeatedly in order to fill in the output buffer.
- void internal_reset ()

  Call this method from the driver API in order to.
- void add_to_output (GeneratorAction ∗action)

  Call this function from generate_more to extend the output buffer.
- GeneratorAction ∗ **from_const_string** (const char ∗data)
- GeneratorAction ∗ **from_integer** (int data)

Private Types

- enum **ActionType** { **CONST_LITERAL**, **RENDER_INT** }

Private Member Functions

- void init_front_action ()

  Sets up the internal structures needed based on the action in the front of the pendingQueue_.
- const char ∗ get_front_buffer ()

  Returns the pointer to the data representing the front action.

Private Attributes

- TypedQueue< GeneratorAction > pendingActions_

  Actions that were generated by the last call of generate_more().
- size_t fileOffset_

  The offset (in the file) of the first byte of the first Action in pendingActions_.
- unsigned **bufferOffset_**
- char buffer_ [16]

  For rendering integers.

Friends

- class **TestEmptyXmlGenerator**

10.35.1 Member Function Documentation

10.35.1.1 generate_more()

```
virtual void commandstation::XmlGenerator::generate_more ( )  [protected], [pure virtual]
```

This function will be called repeatedly in order to fill in the output buffer.

Each call must call add_to_output at least once unless the EOF is reached.

Implemented in commandstation::FdiXmlGenerator.

10.35.1.2 init_front_action()

```
void commandstation::XmlGenerator::init_front_action ( )  [private]
```

Sets up the internal structures needed based on the action in the front of the pendingQueue_.

10.35.1.3 read()

```
ssize_t commandstation::XmlGenerator::read (
             size_t offset,
             void * buf,
             size_t len )
```

Reads from the buffer, or generates more data to read.

Returns the number of bytes written to buf. Returns a short read (including 0) if and only if EOF is reached.

10.35.2 Member Data Documentation

10.35.2.1 fileOffset_

```
size_t commandstation::XmlGenerator::fileOffset_  [private]
```

The offset (in the file) of the first byte of the first Action in pendingActions_.

10.35.2.2 pendingActions_

```
TypedQueue<GeneratorAction> commandstation::XmlGenerator::pendingActions_  [private]
```

Actions that were generated by the last call of generate_more().

Note that the order of these action is REVERSED during the call to generate_more().

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

- XmlGenerator.hxx

11 File Documentation

11.1 AllTrainNodes.hxx File Reference

Classes

- class commandstation::AllTrainNodes

Functions

- openlcb::TrainImpl ∗ **commandstation::create_train_node_helper** (DccMode mode, int address)

11.1.1 Detailed Description

Copyright

A class that instantiates every train node from the TrainDb.

Author

Balazs Racz

Date

20 May 2014

11.2 AllTrainNodesInterface.hxx File Reference

Classes

- class commandstation::AllTrainNodesInterface

  Abstract class for the AllTrainNodes that prevents pulling in transitive dependencies.

11.2.1 Detailed Description

Copyright

Abstract class for the AllTrainNodes that prevents pulling in transitive dependencies.

Author

Balazs Racz

Date

8 Aug 2020

11.3 AnalogReadSysFS.h File Reference

Macros

- #define **MAX_BUF** 255
- #define **SYSFS_ADC_DIR** "/sys/devices/platform/ocp/44e0d000.tscadc/TI-am335x-adc.0.auto/iio:device0/"
- #define **AIN0** 0
- #define **AIN1** 1
- #define **AIN2** 2
- #define **AIN3** 3
- #define **AIN4** 4
- #define **AIN5** 5
- #define **AIN6** 6

Functions

- uint32_t **sysfs_adc_getvalue** (uint32_t channel)

11.3.1 Detailed Description

Copyright

Reads the analog inputs, using the SysFS filesystem.

11.4 AutoPersistCallbackFlow.hxx File Reference

Classes

- class AutoPersistFlow

11.4.1 Detailed Description

Copyright

11.5 BBRailComDriver.hxx File Reference

Classes

- class BBRailComDriver< HW >

11.5.1 Detailed Description

Copyright

Beagle Board Railcom driver. Uses a Beagle Board UART driver.

11.6 BeagleTrainDatabase.hxx File Reference

Classes

- struct BeagleCS::BeaglePersistentTrainData
- class BeagleCS::BeagleTrainDbEntry
- class BeagleCS::BeagleTrainDatabase

Macros

- #define **CONFIG_ROSTER_AUTO_IDLE_NEW_LOCOS** false
- #define **CONFIG_ROSTER_PERSISTENCE_INTERVAL_SEC** 10
- #define **CONFIG_ROSTER_AUTO_CREATE_ENTRIES**

Variables

- const char ∗const **BeagleCS::TRAIN_DB_JSON_FILE** = persistenttrainfile

11.6.1 Detailed Description

Copyright

11.7 CDIHelper.hxx File Reference

Classes

- class CDIHelper

11.7.1 Detailed Description

Copyright

11.8 CommandStationConsole.hxx File Reference

Classes

- class CommandStationConsole

11.8.1 Detailed Description

Copyright

> Copyright (C) 2019 Robert Heller D/B/A Deepwoods Software 51 Locke Hill Road Wendell, MA 01379-9728

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Implements a Comand Console for the Command Station, generally connecting a "front end" (eg GUI or Web Based) via a Tcp/Ip socket.

11.9 CommandStationDCCPRUTrack.hxx File Reference

Classes

- class CommandStationDCCPRUTrack< PRU_NUM >

11.9.1 Detailed Description

Copyright

> Copyright (C) 2021 Robert Heller D/B/A Deepwoods Software 51 Locke Hill Road Wendell, MA 01379-9728

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Defines a class that interfaces with the PRU firmware using the RPMsg virtual I/O device. The constructor generates the file pathnames for the partitular PRU (0 or 1) and then loads the PRU firmware into the PRU. Both PRUs run the same program, with some compile-time differences – RPMsg channel, GPIO bits, and preamble length, etc. The basic DCC waveform is the same for both the main and prog tracks.

The PRU firmware just loops sending the last DCC packet over and over again. This class sends new packets as they become available.

11.10 CommandStationStack.hxx File Reference

Classes

- class openlcb::SimpleCommandStationCanStack

  CAN-based Command Station stack.
- class openlcb::SimpleCommandStationTcpStack

  Tcp-based Command Station stack.

11.10.1 Detailed Description

Copyright

Copyright (C) 2019 Robert Heller D/B/A Deepwoods Software 51 Locke Hill Road Wendell, MA 01379-9728

Command Station node stacks (based on SimpleStack).

11.11 config.hxx File Reference

Functions

- openlcb::CDI_GROUP (IoBoardSegment, Name(HARDWARE_IMPL), Segment(MemoryConfigDefs::SPACE_↩ CONFIG), Offset(128))

  Defines the main segment in the configuration CDI.
- openlcb::CDI_GROUP_ENTRY (internal_config, InternalConfigData)

  Each entry declares the name of the current entry, then the type and then optional arguments list.
- **openlcb::CDI_GROUP_ENTRY** (maindcc, HBridgeControlConfig, Name("Main DCC"))
- **openlcb::CDI_GROUP_ENTRY** (progdcc, HBridgeControlConfig, Name("Programming Track DCC"))
- **openlcb::CDI_GROUP_ENTRY** (fancontrol, FanControlConfig, Name("Fan Control"))
- **openlcb::CDI_GROUP_END** ()
- openlcb::CDI_GROUP (VersionSeg, Segment(MemoryConfigDefs::SPACE_CONFIG), Name("Version information"))

  This segment is only needed temporarily until there is program code to set the ACDI user data version byte.
- **openlcb::CDI_GROUP_ENTRY** (acdi_user_version, Uint8ConfigEntry, Name("ACDI User Data version"), Description("Set to 2 and do not change."))
- **openlcb::CDI_GROUP_ENTRY** (buildrevisions, BuildRevisions)
- openlcb::CDI_GROUP (ConfigDef, MainCdi())

  The main structure of the CDI.

- openlcb::CDI_GROUP_ENTRY (ident, Identification)

  Adds the <identification> tag with the values from SNIP_STATIC_DATA above.
- openlcb::CDI_GROUP_ENTRY (acdi, Acdi)

  Adds an <acdi> tag.
- openlcb::CDI_GROUP_ENTRY (userinfo, UserInfoSegment, Name("User Info"))

  Adds a segment for changing the values in the ACDI user-defined space.
- openlcb::CDI_GROUP_ENTRY (seg, IoBoardSegment)

  Adds the main configuration segment.
- openlcb::CDI_GROUP_ENTRY (version, VersionSeg)

  Adds the versioning segment.

Variables

- const SimpleNodeStaticValues openlcb::SNIP_STATIC_DATA

  Defines the identification information for the node.
- static constexpr uint16_t openlcb::CANONICAL_VERSION = 0x9000

  Used for detecting when the config file stems from a different config.hxx version and needs to be factory reset before using.

11.11.1 Detailed Description

Copyright

Define the configuration structure (CDI)

11.11.2 Function Documentation

11.11.2.1 CDI_GROUP() [1/3]

```
openlcb::CDI_GROUP (
             IoBoardSegment ,
             Name(HARDWARE_IMPL) ,
             Segment(MemoryConfigDefs::SPACE_CONFIG) ,
             Offset(128)  )
```

Defines the main segment in the configuration CDI.

This is laid out at origin 128 to give space for the ACDI user data at the beginning.

11.11.2.2 CDI_GROUP() [2/3]

```
openlcb::CDI_GROUP (
             VersionSeg ,
             Segment(MemoryConfigDefs::SPACE_CONFIG) ,
             Name("Version information")  )
```

This segment is only needed temporarily until there is program code to set the ACDI user data version byte.

11.11.2.3 CDI_GROUP() [3/3]

```
openlcb::CDI_GROUP (
             ConfigDef ,
             MainCdi()  )
```

The main structure of the CDI.

ConfigDef is the symbol we use in main.cxx to refer to the configuration defined here.

11.11.2.4 CDI_GROUP_ENTRY() [1/2]

```
openlcb::CDI_GROUP_ENTRY (
             internal_config ,
             InternalConfigData  )
```

Each entry declares the name of the current entry, then the type and then optional arguments list.

11.11.2.5 CDI_GROUP_ENTRY() [2/2]

```
openlcb::CDI_GROUP_ENTRY (
             userinfo ,
             UserInfoSegment ,
             Name("User Info")  )
```

Adds a segment for changing the values in the ACDI user-defined space.

UserInfoSegment is defined in the system header.

11.11.3 Variable Documentation

11.11.3.1 CANONICAL_VERSION

```
constexpr uint16_t openlcb::CANONICAL_VERSION = 0x9000  [static]
```

Used for detecting when the config file stems from a different config.hxx version and needs to be factory reset before using.

Change every time that the config eeprom file's layout changes.

11.11.3.2 SNIP_STATIC_DATA

```
const SimpleNodeStaticValues openlcb::SNIP_STATIC_DATA
```

Initial value:

```
= {
    4,               "Deepwoods Software", HARDWARE_IMPL,
    "linux.armv7a", "1.00"}
```

Defines the identification information for the node.

The arguments are:

- 4 (version info, always 4 by the standard
- Manufacturer name
- Model name
- Hardware version
- Software version

This data will be used for all purposes of the identification:

- the generated cdi.xml will include this data
- the Simple Node Ident Info Protocol will return this data
- the ACDI memory space will contain this data.

11.12 dccpacket.h File Reference

Classes

- struct DCCPacket

  Stores a DCC packet in memory.
- struct DCCPacket::pkt_t

  Specifies the meaning of the command byte for packets to send.
- struct DCCPacket::cmd_t

  Specifies the meaning of the command byte for meta-commands to send.

Macros

- #define DCC_PACKET_MAX_PAYLOAD (6)

  Maximum number of payload bytes.
- #define ONEBitTime 11600

  1/2 bit times in units of cycles (5 nanoseconds)
- #define **ZEROBitTime** 20000

11.12.1 Detailed Description

Copyright

Packet structure on the PRUs.

11.12.2 Macro Definition Documentation

11.12.2.1 DCC_PACKET_MAX_PAYLOAD

```
#define DCC_PACKET_MAX_PAYLOAD (6)
```

Maximum number of payload bytes.

11.13 DCCProgrammer.hxx File Reference

Classes

- class BeagleCS::DCCProgrammer

11.13.1 Detailed Description

Copyright

> Copyright (C) 2021 Robert Heller D/B/A Deepwoods Software 51 Locke Hill Road Wendell, MA 01379-9728

DCC Programmer

11.14 DuplexedTrackIf.hxx File Reference

Classes

- class BeagleCS::DuplexedTrackIf

Typedefs

- using **CommandStationDCCMainTrack** = CommandStationDCCPRUTrack< 0 >
- using **CommandStationDCCProgTrack** = CommandStationDCCPRUTrack< 1 >

11.14.1 Detailed Description

Copyright

> Copyright (C) 2021 Robert Heller D/B/A Deepwoods Software 51 Locke Hill Road Wendell, MA 01379-9728

Track IF duplexer.

11.15 EStopHandler.hxx File Reference

Classes

- class BeagleCS::EStopHandler

11.15.1 Detailed Description

Copyright

Emergency stop handler.

11.16 ExtendedRingBuffer.hxx File Reference

Classes

- class ExtendedRingBuffer< T >

  Implements an extended ring buffer.

11.16.1 Detailed Description

Copyright

11.17 ExternalTrainDbEntry.hxx File Reference

Classes

- class commandstation::ExternalTrainDbEntry

11.17.1 Detailed Description

Copyright

Copyright (c) 2014-2016, Balazs Racz All rights reserved.

Author

Balazs Racz

Date

Jul 2 2016

11.18 FanControl.hxx File Reference

Classes

- class FanControl

Functions

- CDI_GROUP (FanControlConfig)

  CDI Configuration for a FanControl.
- **CDI_GROUP_ENTRY** (alarmtemperaturethresh, openlcb::Uint16ConfigEntry, Name("Alarm Temperature threshold, in tenths of degrees Centitrade."), Default(350), Min(250), Max(500), Description("This is the temperature level to issue an event."))
- **CDI_GROUP_ENTRY** (alarmon, openlcb::EventConfigEntry, Name("Alarm On Event"))
- **CDI_GROUP_ENTRY** (alarmoff, openlcb::EventConfigEntry, Name("Alarm Off Event"))
- **CDI_GROUP_ENTRY** (fantemperaturethresh, openlcb::Uint16ConfigEntry, Name("Fan Temperature threshold, in tenths of degrees Centitrade."), Default(250), Min(250), Max(500), Description("This is the temperature level to turn on the fan."))
- **CDI_GROUP_ENTRY** (fanon, openlcb::EventConfigEntry, Name("Fan On Event"))
- **CDI_GROUP_ENTRY** (fanoff, openlcb::EventConfigEntry, Name("Fan Off Event"))
- **CDI_GROUP_END** ()

11.18.1 Detailed Description

Copyright

Thermal control code.

11.19 FdiXmlGenerator.hxx File Reference

Classes

- class commandstation::FdiXmlGenerator

11.19.1 Detailed Description

Copyright

Train FDI generator.

Author

Balazs Racz

Date

16 Jan 2016

11.20 FindProtocolDefs.hxx File Reference

Classes

- struct commandstation::FindProtocolDefs

11.20.1 Detailed Description

Copyright

Definitions for the train node find protocol.

Author

Balazs Racz

Date

18 Feb 2016

11.21 FindProtocolServer.hxx File Reference

Classes

- class commandstation::FindProtocolServer
- struct commandstation::FindProtocolServer::Request
- class commandstation::FindProtocolServer::FindProtocolFlow
- class commandstation::SingleNodeFindProtocolServer

11.21.1 Detailed Description

Copyright

Copyright (c) 2014-2016, Balazs Racz All rights reserved.

Implementation of the find protocol event handler.

Author

Balazs Racz

Date

18 Feb 2016

11.22 HBridgeControl.hxx File Reference

Classes

- class HBridgeControl

Macros

- #define **BIT**(n) (1 << n)

Functions

- CDI_GROUP (HBridgeControlConfig)

  CDI Configuration for a HBridgeControl.
- **CDI_GROUP_ENTRY** (event_short, openlcb::EventConfigEntry, Name("Short Detected"), Description("This event will be produced when a short has " "been detected on the track output."))
- **CDI_GROUP_ENTRY** (event_short_cleared, openlcb::EventConfigEntry, Name("Short Cleared"), Description("This event will be produced when a short has " "been cleared on the track output."))
- **CDI_GROUP_ENTRY** (event_shutdown, openlcb::EventConfigEntry, Name("H-Bridge Shutdown"), Description("This event will be produced when the track " "output power has exceeded the safety threshold " "of the H-Bridge."))
- **CDI_GROUP_ENTRY** (event_shutdown_cleared, openlcb::EventConfigEntry, Name("H-Bridge Shutdown Cleared"), Description("This event will be produced when the track " "output power has returned to safe levels."))
- **CDI_GROUP_ENTRY** (event_thermflagon, openlcb::EventConfigEntry, Name("Thermal Flag on"))
- **CDI_GROUP_ENTRY** (event_thermflagoff, openlcb::EventConfigEntry, Name("Thermal Flag off"))
- **CDI_GROUP_END** ()

11.22.1 Detailed Description

Copyright

Copyright (C) 2019 Robert Heller D/B/A Deepwoods Software 51 Locke Hill Road Wendell, MA 01379-9728

Manage a H-Bridge.

11.23 JsonConstants.hxx File Reference

Variables

- constexpr const char ∗ **JSON_FILE_NODE** = "file"
- constexpr const char ∗ **JSON_NAME_NODE** = "name"
- constexpr const char ∗ **JSON_STATE_NODE** = "state"
- constexpr const char ∗ **JSON_USAGE_NODE** = "usage"
- constexpr const char ∗ **JSON_MODE_NODE** = "mode"
- constexpr const char ∗ **JSON_COUNT_NODE** = "count"
- constexpr const char ∗ **JSON_ADDRESS_NODE** = "address"
- constexpr const char ∗ **JSON_SUB_ADDRESS_NODE** = "subAddress"
- constexpr const char ∗ **JSON_BOARD_ADDRESS_NODE** = "boardAddress"

- constexpr const char ∗ **JSON_SPEED_NODE** = "speed"
- constexpr const char ∗ **JSON_DIRECTION_NODE** = "dir"
- constexpr const char ∗ **JSON_ORIENTATION_NODE** = "orientation"
- constexpr const char ∗ **JSON_DESCRIPTION_NODE** = "description"
- constexpr const char ∗ **JSON_TYPE_NODE** = "type"
- constexpr const char ∗ **JSON_IDLE_NODE** = "idle"
- constexpr const char ∗ **JSON_IDLE_ON_STARTUP_NODE** = "idleOnStartup"
- constexpr const char ∗ **JSON_DEFAULT_ON_THROTTLE_NODE** = "defaultOnThrottles"
- constexpr const char ∗ **JSON_FUNCTIONS_NODE** = "functions"
- constexpr const char ∗ **JSON_LOCOS_NODE** = "locos"
- constexpr const char ∗ **JSON_LOCO_NODE** = "loco"
- constexpr const char ∗ **JSON_CONSIST_NODE** = "consist"
- constexpr const char ∗ **JSON_CONSISTS_NODE** = "consists"
- constexpr const char ∗ **JSON_DECODER_ASSISTED_NODE** = "decoderAssisted"
- constexpr const char ∗ **JSON_OUTPUTS_NODE** = "outputs"
- constexpr const char ∗ **JSON_ID_NODE** = "id"
- constexpr const char ∗ **JSON_PIN_NODE** = "pin"
- constexpr const char ∗ **JSON_FLAGS_NODE** = "flags"
- constexpr const char ∗ **JSON_INVERTED_NODE** = "inverted"
- constexpr const char ∗ **JSON_FORCE_STATE_NODE** = "forceState"
- constexpr const char ∗ **JSON_DEFAULT_STATE_NODE** = "defaultState"
- constexpr const char ∗ **JSON_SENSORS_NODE** = "sensors"
- constexpr const char ∗ **JSON_PULLUP_NODE** = "pullUp"
- constexpr const char ∗ **JSON_TURNOUTS_NODE** = "turnouts"
- constexpr const char ∗ **JSON_TURNOUTS_READABLE_STRINGS_NODE** = "readableStrings"
- constexpr const char ∗ **JSON_S88_NODE** = "s88"
- constexpr const char ∗ **JSON_S88_SENSOR_BASE_NODE** = "sensorIDBase"
- constexpr const char ∗ **JSON_PROG_ON_MAIN** = "pom"
- constexpr const char ∗ **JSON_CV_NODE** = "cv"
- constexpr const char ∗ **JSON_VALUE_NODE** = "value"
- constexpr const char ∗ **JSON_CV_BIT_NODE** = "bit"
- constexpr const char ∗ **JSON_IDENTIFY_NODE** = "identify"
- constexpr const char ∗ **JSON_ADDRESS_MODE_NODE** = "addressMode"
- constexpr const char ∗ **JSON_SPEED_TABLE_NODE** = "speedTable"
- constexpr const char ∗ **JSON_DECODER_VERSION_NODE** = "version"
- constexpr const char ∗ **JSON_DECODER_MANUFACTURER_NODE** = "manufacturer"
- constexpr const char ∗ **JSON_CREATE_NODE** = "create"
- constexpr const char ∗ **JSON_OVERALL_STATE_NODE** = "overallState"
- constexpr const char ∗ **JSON_LAST_UPDATE_NODE** = "lastUpdate"
- constexpr const char ∗ **JSON_LCC_NODE** = "lcc"
- constexpr const char ∗ **JSON_LCC_FORCE_RESET_NODE** = "reset"
- constexpr const char ∗ **JSON_LCC_NODE_ID_NODE** = "id"
- constexpr const char ∗ **JSON_LCC_CAN_NODE** = "can"
- constexpr const char ∗ **JSON_WIFI_NODE** = "wifi"
- constexpr const char ∗ **JSON_WIFI_MODE_NODE** = "mode"
- constexpr const char ∗ **JSON_WIFI_SSID_NODE** = "ssid"
- constexpr const char ∗ **JSON_WIFI_PASSWORD_NODE** = "password"
- constexpr const char ∗ **JSON_WIFI_SOFTAP_NODE** = "softap"
- constexpr const char ∗ **JSON_WIFI_STATION_NODE** = "station"
- constexpr const char ∗ **JSON_WIFI_STATION_IP_NODE** = "ip"
- constexpr const char ∗ **JSON_WIFI_STATION_GATEWAY_NODE** = "gateway"

- constexpr const char ∗ **JSON_WIFI_STATION_NETMASK_NODE** = "netmask"
- constexpr const char ∗ **JSON_WIFI_DNS_NODE** = "dns"
- constexpr const char ∗ **JSON_WIFI_RSSI_NODE** = "rssi"
- constexpr const char ∗ **JSON_WIFI_AUTH_NODE** = "auth"
- constexpr const char ∗ **JSON_HC12_NODE** = "hc12"
- constexpr const char ∗ **JSON_HC12_ENABLED_NODE** = "enabled"
- constexpr const char ∗ **JSON_HC12_UART_NODE** = "uart"
- constexpr const char ∗ **JSON_HC12_RX_NODE** = "rx"
- constexpr const char ∗ **JSON_HC12_TX_NODE** = "tx"
- constexpr const char ∗ **JSON_HBRIDGES_NODE** = "hbridges"
- constexpr const char ∗ **JSON_HBRIDGE_ENABLE_PIN_NODE** = "enable"
- constexpr const char ∗ **JSON_HBRIDGE_SIGNAL_PIN_NODE** = "signal"
- constexpr const char ∗ **JSON_HBRIDGE_PREAMBLE_BITS_NODE** = "preamble"
- constexpr const char ∗ **JSON_HBRIDGE_THERMAL_PIN_NODE** = "thermal"
- constexpr const char ∗ **JSON_HBRIDGE_SENSE_PIN_NODE** = "sense"
- constexpr const char ∗ **JSON_HBRIDGE_RMT_CHANNEL_NODE** = "rmt"
- constexpr const char ∗ **JSON_RAILCOM_NODE** = "railcom"
- constexpr const char ∗ **JSON_RAILCOM_ENABLE_PIN_NODE** = "enable"
- constexpr const char ∗ **JSON_RAILCOM_BRAKE_PIN_NODE** = "brake"
- constexpr const char ∗ **JSON_RAILCOM_SHORT_PIN_NODE** = "short"
- constexpr const char ∗ **JSON_RAILCOM_UART_NODE** = "uart"
- constexpr const char ∗ **JSON_RAILCOM_RX_NODE** = "rx"
- constexpr const char ∗ **JSON_CDI_NODE** = "cdi"
- constexpr const char ∗ **JSON_CDI_UPLINK_NODE** = "uplink"
- constexpr const char ∗ **JSON_CDI_UPLINK_RECONNECT_NODE** = "reconnect"
- constexpr const char ∗ **JSON_CDI_UPLINK_MODE_NODE** = "mode"
- constexpr const char ∗ **JSON_CDI_UPLINK_AUTO_HOST_NODE** = "auto_host"
- constexpr const char ∗ **JSON_CDI_UPLINK_AUTO_SERVICE_NODE** = "auto_service"
- constexpr const char ∗ **JSON_CDI_UPLINK_MANUAL_HOST_NODE** = "manual_host"
- constexpr const char ∗ **JSON_CDI_UPLINK_MANUAL_PORT_NODE** = "manual_port"
- constexpr const char ∗ **JSON_CDI_HUB_NODE** = "hub"
- constexpr const char ∗ **JSON_CDI_HUB_ENABLE_NODE** = "enable"
- constexpr const char ∗ **JSON_CDI_HUB_PORT_NODE** = "port"
- constexpr const char ∗ **JSON_CDI_HUB_SERVICE_NODE** = "service"
- constexpr const char ∗ **JSON_CDI_HBRIDGE_SHORT_EVENT_NODE** = "short"
- constexpr const char ∗ **JSON_CDI_HBRIDGE_SHORT_CLEAR_EVENT_NODE** = "short_clear"
- constexpr const char ∗ **JSON_CDI_HBRIDGE_SHUTDOWN_EVENT_NODE** = "shutdown"
- constexpr const char ∗ **JSON_CDI_HBRIDGE_SHUTDOWN_CLEAR_EVENT_NODE** = "shutdown_clear"
- constexpr const char ∗ **JSON_CDI_HBRIDGE_THERMAL_EVENT_NODE** = "thermal"
- constexpr const char ∗ **JSON_CDI_HBRIDGE_THERMAL_CLEAR_EVENT_NODE** = "thermal_clear"
- constexpr const char ∗ **JSON_VALUE_STATION_IP_MODE_STATIC** = "static"
- constexpr const char ∗ **JSON_VALUE_STATION_IP_MODE_DHCP** = "dhcp"
- constexpr const char ∗ **JSON_VALUE_WIFI_MODE_SOFTAP_ONLY** = "softap"
- constexpr const char ∗ **JSON_VALUE_WIFI_MODE_SOFTAP_STATION** = "softap-station"
- constexpr const char ∗ **JSON_VALUE_WIFI_MODE_STATION_ONLY** = "station"
- constexpr const char ∗ **JSON_VALUE_FORWARD** = "FWD"
- constexpr const char ∗ **JSON_VALUE_REVERSE** = "REV"
- constexpr const char ∗ **JSON_VALUE_TRUE** = "true"
- constexpr const char ∗ **JSON_VALUE_FALSE** = "false"
- constexpr const char ∗ **JSON_VALUE_NORMAL** = "Normal"
- constexpr const char ∗ **JSON_VALUE_OFF** = "Off"

- constexpr const char ∗ **JSON_VALUE_ON** = "On"
- constexpr const char ∗ **JSON_VALUE_FAULT** = "Fault"
- constexpr const char ∗ **JSON_VALUE_ERROR** = "Error"
- constexpr const char ∗ **JSON_VALUE_THROWN** = "Thrown"
- constexpr const char ∗ **JSON_VALUE_CLOSED** = "Closed"
- constexpr const char ∗ **JSON_VALUE_LONG_ADDRESS** = "Long Address"
- constexpr const char ∗ **JSON_VALUE_SHORT_ADDRESS** = "Short Address"
- constexpr const char ∗ **JSON_VALUE_MOBILE_DECODER** = "Mobile Decoder"
- constexpr const char ∗ **JSON_VALUE_STATIONARY_DECODER** = "Stationary Decoder"

11.23.1 Detailed Description

Copyright

11.24 main.cxx File Reference

Classes

- class FactoryResetHelper

Macros

- #define **DefaultNODEID** 0x050101012200ULL
- #define **OPTSTRING** "hn:e:t:M:P:W:"

Functions

- **OVERRIDE_CONST** (local_nodes_count, 50)
- **OVERRIDE_CONST** (num_memory_spaces, 7)
- **OVERRIDE_CONST** (gc_generate_newlines, 1)
- **OVERRIDE_CONST** (main_thread_stack_size, 2500)
- openlcb::ConfigDef **cfg** (0)
- void **usage** (const char ∗e)
- openlcb::NodeID **parseNodeID** (const char ∗nidstring)
- void **parse_args** (int argc, char ∗argv[ ])
- int appl_main (int argc, char ∗argv[ ])

  Entry point to application.

Variables

- static openlcb::NodeID **NODE_ID** = DefaultNODEID
- char **pathnamebuffer** [256]
- char **persistenttrainfile** [256]
- static char **mainPRUfirmware** [256] = "MainTrackDCC.out"
- static char **progPRUfirmware** [256] = "ProgTrackDCC.out"
- static bool **start_WiThrottle** = false
- static char **WiThrottle_Name** [256] = "PocketBeagle"
- int **WiThrottle_port** = -1
- withrottle::Server ∗ **WiThrottleServer**

11.24.1 Detailed Description

Copyright

Main file for the BBBCommandStationOpenMRN program.

Author

Robert Heller

Date

3 Feb 2019 – 11 May 2021

11.24.2 Function Documentation

11.24.2.1 appl_main()

```
int appl_main (
            int argc,
            char * argv[ ] )
```

Entry point to application.

Parameters

argc	number of command line arguments
argv	array of command line arguments

Returns

0, should never return

11.25 ProgrammingTrackSpaceConfig.hxx File Reference

Classes

- struct commandstation::ProgrammingTrackSpaceConfig::Shadow

  This shadow structure is declared to be parallel to the CDI entries.

Macros

- #define **SHADOW_OFFSETOF**(entry) ((uintptr_t) & ((ProgrammingTrackSpaceConfig::Shadow∗)nullptr)->entry)

Enumerations

- enum **OperatingMode** { **PROG_DISABLED** = 0, **DIRECT_MODE** = 1, **POM_MODE** = 2, **ADVANCED** = 10 }

Functions

- **commandstation::CDI_GROUP** (ProgrammingTrackSpaceConfigAdvanced)
- **commandstation::CDI_GROUP_ENTRY** (repeat_verify, openlcb::Uint32ConfigEntry, Name("Repeat count for verify packets"), Description("How many times a direct mode bit verify packet needs to be " "repeated for an acknowledgement to be generated."), Default(3), Min(0), Max(255))
- **commandstation::CDI_GROUP_ENTRY** (repeat_cooldown_reset, openlcb::Uint32ConfigEntry, Name("Repeat count for reset packets after verify"), Description("How many reset packets to send after a verify."), Default(6), Min(0), Max(255))
- **commandstation::CDI_GROUP_END** ()
- **commandstation::CDI_GROUP** (ProgrammingTrackSpaceConfig, Segment(openlcb::MemoryConfigDefs::S↩ PACE_DCC_CV), Offset(0x7F100000), Name("Programming track operation"), Description("Use this component to read and write CVs on the " "programming track of the command station."))
- **commandstation::CDI_GROUP_ENTRY** (mode, openlcb::Uint32ConfigEntry, Name("Operating mode"), Map↩ Values(OPERATING_MODE_MAP_VALUES))
- **commandstation::CDI_GROUP_ENTRY** (cv, openlcb::Uint32ConfigEntry, Name("CV number"), Description("Number of CV to read or write (1..1024)."), Default(0), Min(0), Max(1024))
- **commandstation::CDI_GROUP_ENTRY** (value, openlcb::Uint32ConfigEntry, Name("CV value"), Description("Set 'Operating mode' and 'CV number' first, then: hit 'Refresh' to " "read the entire CV, or enter a value and hit 'Write' to set the CV."), Default(0), Min(0), Max(255))
- **commandstation::CDI_GROUP_ENTRY** (bit_write_value, openlcb::Uint32ConfigEntry, Name("Bit change"), Description("Set 'Operating mode' and 'CV number' first, then: write 1064 to set " "the single bit whose value is 64, or 2064 to clear that bit. Write " "100 to 107 to set bit index 0 to 7, or 200 to 207 to clear bit 0 to " "7. Values outside of these two ranges do nothing."), Default(1000), Min(100), Max(2128))
- **commandstation::CDI_GROUP_ENTRY** (bit_value_string, openlcb::StringConfigEntry< 24 >, Name("Read bits decomposition"), Description("Hit Refresh on this line after reading a CV value " "to see which bits are set."))
- **commandstation::CDI_GROUP_ENTRY** (advanced, ProgrammingTrackSpaceConfigAdvanced, Name("Advanced settings"))

Variables

- static const char **commandstation::OPERATING_MODE_MAP_VALUES** [ ]

11.25.1 Detailed Description

Copyright

CDI configuration for the CV space to access the programming track flow.

Author

Balazs Racz

Date

2 June 2018

11.25.2 Variable Documentation

11.25.2.1 OPERATING_MODE_MAP_VALUES

```
const char commandstation::OPERATING_MODE_MAP_VALUES[ ]  [static]
```

Initial value:

```
= R"(
<relation><property>0</property><value>Disabled</value></relation>
<relation><property>1</property><value>Direct mode</value></relation>
<relation><property>2</property><value>POM mode</value></relation>
<relation><property>10</property><value>Advanced mode</value></relation>
)"
```

11.26 resource_table_0.h File Reference

Classes

- struct my_resource_table

Macros

- #define **PRU_RPMSG_VQ0_SIZE** 16
- #define **PRU_RPMSG_VQ1_SIZE** 16
- #define **VIRTIO_RPMSG_F_NS** 0
- #define **RPMSG_PRU_C0_FEATURES** (1 << VIRTIO_RPMSG_F_NS)
- #define **HOST_UNUSED** 255

Variables

- struct ch_map **pru_intc_map** [ ]
- struct my_resource_table **resourceTable**

11.26.1 Detailed Description

Copyright

Resources for PRU #0 (Mains DCC)

11.26.2 Variable Documentation

11.26.2.1 pru_intc_map

```
struct ch_map pru_intc_map[]
```

Initial value:

```
= { {16, 2},
                                        {17, 0},
}
```

11.27 resource_table_1.h File Reference

Classes

- struct my_resource_table

Macros

- #define **PRU_RPMSG_VQ0_SIZE** 16
- #define **PRU_RPMSG_VQ1_SIZE** 16
- #define **VIRTIO_RPMSG_F_NS** 0
- #define **RPMSG_PRU_C0_FEATURES** (1 << VIRTIO_RPMSG_F_NS)
- #define **HOST_UNUSED** 255

Variables

- struct ch_map **pru_intc_map** [ ]
- struct my_resource_table **resourceTable**

11.27.1 Detailed Description

Copyright

Resources for PRU #1 (Programming track DCC)

11.27.2 Variable Documentation

11.27.2.1 pru_intc_map

```
struct ch_map pru_intc_map[ ]
```

Initial value:

```
= { {18, 3},
                                          {19, 1},
}
```

11.28 resource_table_empty.h File Reference

Classes

- struct my_resource_table

Variables

- struct my_resource_table **pru_remoteproc_ResourceTable**

11.28.1 Detailed Description

Copyright

Empty resources.

11.28.2 Variable Documentation

11.28.2.1 pru_remoteproc_ResourceTable

```
struct my_resource_table pru_remoteproc_ResourceTable
```

Initial value:

```
= {
        1,
        0,
        0, 0,
        0,
}
```

11.29 TrainDb.hxx File Reference

Classes

- class commandstation::TrainDbEntry
- class commandstation::TrainDb

11.29.1 Detailed Description

Copyright

> Copyright (c) 2014, Balazs Racz All rights reserved.

Interface for accessing a train database for the mobile station lookup.

Added description field accessor. Robert Heller Sun Apr 25 13:43:20 2021

Author

> Balazs Racz

Date

> 18 May 2014

11.30 TrainDbCdi.hxx File Reference

Typedefs

- using **commandstation::TrainDbCdiRepFunctionGroup** = openlcb::RepeatedGroup< TrainDbCdiFunction↩ Group, DCC_MAX_FN - 1 >

Functions

- **commandstation::CDI_GROUP** (TrainDbCdiFunctionGroup, Name("Functions"), Description("Defines what each function button does."))
- **commandstation::CDI_GROUP_ENTRY** (icon, openlcb::Uint8ConfigEntry, Name("Display"), Description("Defines how throttles display this function."), Default(FN_NONEXISTANT), MapValues(FNDISPLAY_MAP))
- **commandstation::CDI_GROUP_ENTRY** (is_momentary, openlcb::Uint8ConfigEntry, Name("Momentary"), Description("Momentary functions are automatically turned off when you " "release the respective button on the throttles."), MapValues(MOMENTARY_MAP), Default(0))
- **commandstation::CDI_GROUP_END** ()
- **commandstation::CDI_GROUP** (F0Group, Name("F0"), Description("F0 is permanently assigned to Light."))
- **commandstation::CDI_GROUP_ENTRY** (blank, openlcb::EmptyGroup< TrainDbCdiFunctionGroup::size()>)
- **commandstation::CDI_GROUP** (TrainDbCdiAllFunctionGroup)
- **commandstation::CDI_GROUP_ENTRY** (f0, F0Group)
- **commandstation::CDI_GROUP_ENTRY** (all_functions, TrainDbCdiRepFunctionGroup, RepName("Fn"))
- **commandstation::CDI_GROUP** (TrainDbCdiEntry, Description("Configures a single train"))
- **commandstation::CDI_GROUP_ENTRY** (address, openlcb::Uint16ConfigEntry, Name("Address"), Description("Track protocol address of the train."), Default(0))
- **commandstation::CDI_GROUP_ENTRY** (mode, openlcb::Uint8ConfigEntry, Name("Protocol"), Description("Protocol to use on the track for driving this train."), MapValues(DCC_DRIVE_MODE_MAP), Default(DCC↩ _28))
- **commandstation::CDI_GROUP_ENTRY** (name, openlcb::StringConfigEntry< 16 >, Name("Name"), Description("Identifies the train node on the LCC bus."))
- **commandstation::CDI_GROUP_ENTRY** (functions, TrainDbCdiAllFunctionGroup)
- **commandstation::CDI_GROUP** (TrainSegment, Segment(openlcb::MemoryConfigDefs::SPACE_CONFIG))
- **commandstation::CDI_GROUP_ENTRY** (train, TrainDbCdiEntry)
- **commandstation::CDI_GROUP** (TrainConfigDef, MainCdi())
- **commandstation::CDI_GROUP_ENTRY** (ident, openlcb::Identification, Model("Virtual train node"))
- **commandstation::CDI_GROUP_ENTRY** (train, TrainSegment)
- **commandstation::CDI_GROUP_ENTRY** (cv, ProgrammingTrackSpaceConfig)
- **commandstation::CDI_GROUP** (TmpTrainSegment, Segment(openlcb::MemoryConfigDefs::SPACE_CONF↩ IG), Offset(0), Name("Non-stored train"), Description("This train is not part of the train database, thus no " "configuration settings can be changed on it."))
- commandstation::CDI_GROUP (TrainTmpConfigDef, MainCdi())

  This alternate CDI for a virtual train node will be in use for trains that are not coming from the database.
- **commandstation::CDI_GROUP_ENTRY** (train, TmpTrainSegment)

Variables

- static const char **commandstation::MOMENTARY_MAP** [ ]
- static const char **commandstation::FNDISPLAY_MAP** [ ]
- static const char **commandstation::DCC_DRIVE_MODE_MAP** [ ]

11.30.1 Detailed Description

Copyright

CDI entry defining the commandstation traindb entry.

Author

Balazs Racz

Date

8 Feb 2016

11.30.2 Function Documentation

11.30.2.1 CDI_GROUP()

```
commandstation::CDI_GROUP (
             TrainTmpConfigDef ,
             MainCdi()  )
```

This alternate CDI for a virtual train node will be in use for trains that are not coming from the database.

It will not offer any settings for the user.

11.30.3 Variable Documentation

11.30.3.1 DCC_DRIVE_MODE_MAP

```
const char commandstation::DCC_DRIVE_MODE_MAP[ ]  [static]
```

Initial value:

```
=
    "<relation><property>0</property><value>Unused</value></relation>"
    "<relation><property>10</property><value>DCC 28-step</value></relation>"
    "<relation><property>11</property><value>DCC 128-step</value></relation>"
    "<relation><property>5</property><value>Marklin-Motorola "
    "I</value></relation>"
    "<relation><property>6</property><value>Marklin-Motorola "
    "II</value></relation>"
    "<relation><property>14</property><value>DCC 28-step (forced long "
    "address)</value></relation>"
    "<relation><property>15</property><value>DCC 128-step (forced long "
    "address)</value></relation>"
```

11.30.3.2 FNDISPLAY_MAP

```
const char commandstation::FNDISPLAY_MAP[ ]  [static]
```

Initial value:

```
=
    "<relation><property>0</property><value>Unavailable</value></relation>"
    "<relation><property>1</property><value>Light</value></relation>"
    "<relation><property>2</property><value>Beamer</value></relation>"
    "<relation><property>3</property><value>Bell</value></relation>"
    "<relation><property>4</property><value>Horn</value></relation>"
    "<relation><property>5</property><value>Shunting mode</value></relation>"
    "<relation><property>6</property><value>Pantograph</value></relation>"
    "<relation><property>7</property><value>Smoke</value></relation>"
    "<relation><property>8</property><value>Momentum off</value></relation>"
    "<relation><property>9</property><value>Whistle</value></relation>"
    "<relation><property>10</property><value>Sound</value></relation>"
    "<relation><property>11</property><value>F</value></relation>"
    "<relation><property>12</property><value>Announce</value></relation>"
    "<relation><property>13</property><value>Engine</value></relation>"
    "<relation><property>14</property><value>Light1</value></relation>"
    "<relation><property>15</property><value>Light2</value></relation>"
    "<relation><property>17</property><value>Uncouple</value></relation>"
    "<relation><property>255</property><value>Unavailable_</value></relation>"
```

11.30.3.3 MOMENTARY_MAP

```
const char commandstation::MOMENTARY_MAP[ ]  [static]
```

Initial value:

```
=
    "<relation><property>0</property><value>Latching</value></relation>"
    "<relation><property>1</property><value>Momentary</value></relation>"
```

11.31 TrainDbDefs.hxx File Reference

Macros

- #define **DCC_MAX_FN** 29

Enumerations

- enum **Symbols** {
  FN_NONEXISTANT = 0, **LIGHT** = 1, **BEAMER** = 2, **BELL** = 3,
  HORN = 128 + 4, **SHUNT** = 5, **PANTO** = 6, **SMOKE** = 7,
  ABV = 8, **WHISTLE** = 128 + 9, **SOUND** = 10, **FNT11** = 11,
  SPEECH = 128 + 12, **ENGINE** = 13, **LIGHT1** = 14, **LIGHT2** = 15,
  TELEX = 128 + 17, **FN_UNKNOWN** = 127, **MOMENTARY** = 128, **FNP** = 139,
  SOUNDP = 141, **FN_UNINITIALIZED** = 255 }
- enum commandstation::DccMode {
  DCCMODE_DEFAULT = 0, **DCCMODE_FAKE_DRIVE** = 1, **DCCMODE_OLCBUSER** = 1, commandstation::MARKLIN_ANY = 0b00100,
  commandstation::MARKLIN_ANY_MASK = 0b11100, commandstation::MARKLIN_DEFAULT = MARKLIN_ANY,
  commandstation::MARKLIN_OLD = MARKLIN_ANY | 1, commandstation::MARKLIN_NEW = MARKLIN_ANY | 2,
  commandstation::MARKLIN_TWOADDR = MARKLIN_ANY | 3, commandstation::MFX = MARKLIN_NEW,
  commandstation::DCC_ANY = 0b01000, commandstation::DCC_ANY_MASK = 0b11000,
  commandstation::DCC_DEFAULT = DCC_ANY, commandstation::DCC_LONG_ADDRESS = 0b00100,
  commandstation::DCC_SS_MASK = 0b00011, commandstation::DCC_DEFAULT_SS = DCC_DEFAULT,
  commandstation::DCC_14 = DCC_ANY | 1, commandstation::DCC_28 = DCC_ANY | 2, commandstation::DCC_128 = DCC_ANY | 3, commandstation::DCC_14_LONG_ADDRESS = DCC_14 | DCC_LONG_ADDRESS,
  commandstation::DCC_28_LONG_ADDRESS = DCC_28 | DCC_LONG_ADDRESS, commandstation::DCC_128_LONG_ADDRESS = DCC_128 | DCC_LONG_ADDRESS, commandstation::DCCMODE_PROTOCOL_MASK = 0b11111 }

Functions

- dcc::TrainAddressType commandstation::dcc_mode_to_address_type (DccMode mode, uint32_t address)

  Converts a DccMode bit mask and a legacy address into a TrainAddressType enum.
- DccMode commandstation::dcc_mode_to_protocol (DccMode mode)

  Converts a DccMode bit mask down to a protocol enumeration, i.e.

11.31.1 Detailed Description

Copyright

- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

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

Common definitions for the train database.

Author

Balazs Racz

Date

13 Feb 2016

11.31.2 Enumeration Type Documentation

11.31.2.1 DccMode

```
enum commandstation::DccMode
```

Enumerator

MARKLIN_ANY	Value for testing whether the protocol is a Markin-Motorola protocol variant.
MARKLIN_ANY_MASK	Mask for testing whether the protocol is a Markin-Motorola protocol variant.
MARKLIN_DEFAULT	Acquisition for a Marklin locomotive with default setting.
MARKLIN_OLD	Force MM protocol version 1 (F0 only).
MARKLIN_NEW	Force MM protocol version 2 (F0-F4).
MARKLIN_TWOADDR	Force MM protocol version 2 with subsequent address for more functions (F0-F8).
MFX	Alias for MFX locmotives (to be driven with Marklin v2 protocol for now).
DCC_ANY	value for testing whether the protocol is a DCC variant
DCC_ANY_MASK	mask for testing whether the protocol is a DCC variant
DCC_DEFAULT	Acquisition for DCC locomotive with default settings.
DCC_LONG_ADDRESS	Force long address for DCC. If clear, uses default address type by number.
DCC_SS_MASK	Mask for the DCC speed step setting.
DCC_DEFAULT_SS	Unpecified / default speed step setting.
DCC_14	Force 14 SS mode.
DCC_28	Force 28 SS mode.
DCC_128	Force 128 SS mode.
DCC_14_LONG_ADDRESS	Force 14 SS mode & long address (this is meaningless).
DCC_28_LONG_ADDRESS	Force 28 SS mode & long address.
DCC_128_LONG_ADDRESS	Force 128 SS mode & long address.

11.31.3 Function Documentation

11.31.3.1 dcc_mode_to_address_type()

```
dcc::TrainAddressType commandstation::dcc_mode_to_address_type (
            DccMode mode,
            uint32_t address )  [inline]
```

Converts a DccMode bit mask and a legacy address into a TrainAddressType enum.

Parameters

mode	the legacy drive mode (e.g. from a TrainDb entry or from a search query)
address	is the legacy address.

Returns

an enum value which together with the address uniquely represents an addressable entity on the track. May return UNSPECIFIED if DccMode == DCCMODE_DEFAULT (usually a query did not specify any restriction) or UNSUPPORTED if we did not recognize the code in the DccMode bitfield.

11.31.3.2 dcc_mode_to_protocol()

```
DccMode commandstation::dcc_mode_to_protocol (
            DccMode mode )  [inline]
```

Converts a DccMode bit mask down to a protocol enumeration, i.e.

DCC, Marklin or OpenLCB.

Parameters

mode	the detailed mode bit field.

Returns

a stripped down mode bit field which does not specify any details about the protocol variant.

11.32 XmlGenerator.hxx File Reference

Classes

- class commandstation::XmlGenerator
- struct commandstation::XmlGenerator::GeneratorAction

11.32.1 Detailed Description

Copyright

Interface for generating XML files on-the-fly on small-memory machines.

Author

Balazs Racz

Date

16 Jan 2016

Index