artdaq_core_demo

v1_06_04a

Generated by Doxygen 1.8.5

Mon Dec 11 2017 10:51:03

Contents

1 Namespace Index **1**

1.1 Namespace List . . . 1

2 Hierarchical Index **3**

2.1 Class Hierarchy . . . 3

3 Class Index **5**

3.1 Class List . . . 5

4 Namespace Documentation **7**

4.1 coredemo Namespace Reference . . . 7

4.1.1 Detailed Description . . . 7

4.2 demo Namespace Reference . . . 7

4.2.1 Detailed Description . . . 8

4.2.2 Function Documentation . . . 8

4.2.2.1 fragmentTypeToString . . . 8

4.2.2.2 makeFragmentTypeMap . . . 9

4.2.2.3 operator<< . . . 9

4.2.2.4 toFragmentType . . . 9

4.3 demo::detail Namespace Reference . . . 9

4.3.1 Detailed Description . . . 9

5 Class Documentation **11**

5.1 demo::AsciiFragment Class Reference . . . 11

5.1.1 Detailed Description . . . 12

5.1.2 Constructor & Destructor Documentation . . . 12

5.1.2.1 AsciiFragment . . . 12

5.1.3 Member Function Documentation . . . 12

5.1.3.1 chars_per_word_ . . . 12

5.1.3.2 hdr_event_size . . . 13

5.1.3.3 hdr_line_number . . . 13
5.1.3.4 hdr_size_words . . . 13
5.1.3.5 header_ . . . 13
5.2 demo::AsciiFragmentWriter Class Reference . . . 14
5.2.1 Detailed Description . . . 14
5.2.2 Constructor & Destructor Documentation . . . 14
5.2.2.1 AsciiFragmentWriter . . . 14
5.2.3 Member Function Documentation . . . 15
5.2.3.1 dataBegin . . . 15
5.2.3.2 dataEnd . . . 15
5.2.3.3 header_ . . . 15
5.2.3.4 resize . . . 15
5.2.3.5 set_hdr_line_number . . . 15
5.3 coredemo::GetPackageBuildInfo Struct Reference . . . 16
5.3.1 Detailed Description . . . 16
5.3.2 Member Function Documentation . . . 16
5.3.2.1 getPackageBuildInfo . . . 16
5.4 demo::AsciiFragment::Header Struct Reference . . . 16
5.4.1 Detailed Description . . . 17
5.5 demo::ToyFragment::Header Struct Reference . . . 17
5.5.1 Detailed Description . . . 18
5.6 demo::UDPFragment::Header Struct Reference . . . 18
5.6.1 Detailed Description . . . 19
5.7 demo::ToyFragment::Metadata Struct Reference . . . 19
5.7.1 Detailed Description . . . 20
5.8 demo::UDPFragment::Metadata Struct Reference . . . 20
5.8.1 Detailed Description . . . 21
5.9 demo::AsciiFragment::Metadata Struct Reference . . . 21
5.9.1 Detailed Description . . . 21
5.9.2 Member Typedef Documentation . . . 21
5.9.2.1 data_t . . . 21
5.10 demo::ToyFragment Class Reference . . . 22
5.10.1 Detailed Description . . . 23
5.10.2 Constructor & Destructor Documentation . . . 23
5.10.2.1 ToyFragment . . . 23
5.10.3 Member Function Documentation . . . 23
5.10.3.1 adc_range . . . 23

5.10.3.2 adc_value . . . 24
5.10.3.3 adcs_per_word_ . . . 24
5.10.3.4 checkADCData . . . 24
5.10.3.5 dataBeginADCs . . . 24
5.10.3.6 dataEndADCs . . . 25
5.10.3.7 fastVerify . . . 25
5.10.3.8 findBadADC . . . 25
5.10.3.9 hdr_event_size . . . 25
5.10.3.10 hdr_size_words . . . 26
5.10.3.11 hdr_trigger_number . . . 26
5.10.3.12 header_ . . . 26
5.10.3.13 total_adc_values . . . 26
5.11 demo::UDPFragment Class Reference . . . 26
5.11.1 Detailed Description . . . 28
5.11.2 Constructor & Destructor Documentation . . . 28
5.11.2.1 UDPFragment . . . 28
5.11.3 Member Function Documentation . . . 28
5.11.3.1 bytes_per_word_ . . . 28
5.11.3.2 dataBegin . . . 28
5.11.3.3 dataEnd . . . 28
5.11.3.4 hdr_data_type . . . 29
5.11.3.5 hdr_event_size . . . 29
5.11.3.6 hdr_size_words . . . 29
5.11.3.7 header_ . . . 29
5.11.3.8 udp_data_words . . . 29
5.12 demo::UDPFragmentWriter Class Reference . . . 30
5.12.1 Detailed Description . . . 30
5.12.2 Constructor & Destructor Documentation . . . 30
5.12.2.1 UDPFragmentWriter . . . 30
5.12.3 Member Function Documentation . . . 31
5.12.3.1 dataBegin . . . 31
5.12.3.2 dataEnd . . . 31
5.12.3.3 header_ . . . 31
5.12.3.4 resize . . . 31
5.12.3.5 set_hdr_type . . . 31

Index 33

Chapter 1

Namespace Index

1.1 Namespace List

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

coredemo
: Namespace used to differentiate the artdaq_core_demo version of GetPackageBuildInfo from other versions present in the system 7

demo
: The artdaq_demo namespace 7

demo::detail
: Implementation details namespace 9

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

demo::AsciiFragment 11
  demo::AsciiFragmentWriter 14
coredemo::GetPackageBuildInfo 16
demo::AsciiFragment::Header 16
demo::ToyFragment::Header 17
demo::UDPFragment::Header 18
demo::ToyFragment::Metadata 19
demo::UDPFragment::Metadata 20
demo::AsciiFragment::Metadata 21
demo::ToyFragment 22
demo::UDPFragment 26
  demo::UDPFragmentWriter 30

Chapter 3

Class Index

3.1 Class List

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

demo::AsciiFragment
An artdaq::Fragment overlay class designed to hold string data for pedagogical purposes 11

demo::AsciiFragmentWriter
Class derived from AsciiFragment which allows writes to the data (for simulation purposes) 14

coredemo::GetPackageBuildInfo
Wrapper around the coredemo::GetPackageBuildInfo::getPackageBuildInfo function 16

demo::AsciiFragment::Header
The AsciiFragment::Header contains information about the payload size and the "line number" of the data contained in the AsciiFragment 16

demo::ToyFragment::Header
The "Header" struct is used to interpret the header information that is created by the hardware and *is* part of the data blob that is read from the hardware for each event. Not to be confused with the Fragment::Header or the ToyFragment::Metadata objects 17

demo::UDPFragment::Header
The UDPFragment::Header contains information about the payload size and the "data type" of the UDP data 18

demo::ToyFragment::Metadata
The ToyFragment::Metadata struct holds data about the readout that may not be present in the primary data stream 19

demo::UDPFragment::Metadata
Metadata describing the contents of the UDPFragment 20

demo::AsciiFragment::Metadata
Metadata describing the contents of the AsciiFragment 21

demo::ToyFragment
An artdaq::Fragment overlay class used for pedagogical purposes 22

demo::UDPFragment
A Fragment designed to contain data received from the network in UDP datagrams 26

demo::UDPFragmentWriter
Class derived from UDPFragment which allows writes to the data 30

Chapter 4

Namespace Documentation

4.1 coredemo Namespace Reference

Namespace used to differentiate the artdaq_core_demo version of GetPackageBuildInfo from other versions present in the system.

Classes

- struct GetPackageBuildInfo

  Wrapper around the coredemo::GetPackageBuildInfo::getPackageBuildInfo function.

4.1.1 Detailed Description

Namespace used to differentiate the artdaq_core_demo version of GetPackageBuildInfo from other versions present in the system.

4.2 demo Namespace Reference

The artdaq_demo namespace.

Namespaces

- detail

  Implementation details namespace.

Classes

- class AsciiFragment

  An artdaq::Fragment overlay class designed to hold string data for pedagogical purposes.
- class AsciiFragmentWriter

  Class derived from AsciiFragment which allows writes to the data (for simulation purposes).
- class ToyFragment

An artdaq::Fragment overlay class used for pedagogical purposes.

- class UDPFragment

  A Fragment designed to contain data received from the network in UDP datagrams.

- class UDPFragmentWriter

  Class derived from UDPFragment which allows writes to the data.

Functions

- std::ostream & operator<< (std::ostream &, AsciiFragment const &)

  Dumps the AsciiFragment's data (text) to given stream.

- FragmentType toFragmentType (std::string t_string)

  Lookup the type code for a fragment by its string name.

- std::string fragmentTypeToString (FragmentType val)

  Look up the name of the given FragmentType.

- std::map
  < artdaq::Fragment::type_t,
  std::string > makeFragmentTypeMap ()

  Create a list of all Fragment types defined by this package, in the format that RawInput expects.

- std::ostream & operator<< (std::ostream &, ToyFragment const &)

  Let the "<<" operator dump the ToyFragment's data to stdout.

- std::ostream & operator<< (std::ostream &, UDPFragment const &)

  Let the "<<" operator dump the UDPFragment's data to stdout.

Variables

- std::vector< std::string > const names {"MISSED", "TOY1", "TOY2", "ASCII", "UDP", "UNKNOWN"}

  List of names (in the order defined below) of the User types defined in artdaq_core_demo.

4.2.1 Detailed Description

The artdaq_demo namespace.

4.2.2 Function Documentation

4.2.2.1 std::string demo::fragmentTypeToString (FragmentType *val*)

Look up the name of the given FragmentType.

Parameters

val	FragmentType to look up

Returns

Name of the given type (from the names vector)

Definition at line 23 of file FragmentType.cc.

4.2.2.2 std::map< artdaq::Fragment::type_t, std::string > demo::makeFragmentTypeMap ( )

Create a list of all Fragment types defined by this package, in the format that RawInput expects.

Returns

A list of all Fragment types defined by this package, in the format that RawInput expects

Definition at line 35 of file FragmentType.cc.

4.2.2.3 std::ostream & demo::operator<< (std::ostream & *os*, AsciiFragment const & *f*)

Dumps the AsciiFragment's data (text) to given stream.

Returns

Reference to stream

Definition at line 15 of file AsciiFragment.cc.

4.2.2.4 demo::FragmentType demo::toFragmentType (std::string *t_string*)

Lookup the type code for a fragment by its string name.

Parameters

t_string	Name of the Fragment type to lookup

Returns

artdaq::Fragment::type_t corresponding to string, or INVALID if not found

Definition at line 9 of file FragmentType.cc.

4.3 demo::detail Namespace Reference

Implementation details namespace.

Enumerations

- enum FragmentType : artdaq::Fragment::type_t {
  MISSED = artdaq::Fragment::FirstUserFragmentType, **TOY1**, **TOY2**, **ASCII**, **UDP**, **INVALID** }

  User type codes defined by artdaq_core_demo.

4.3.1 Detailed Description

Implementation details namespace.

Chapter 5

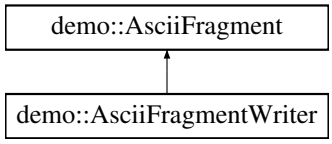
Class Documentation

5.1 demo::AsciiFragment Class Reference

An artdaq::Fragment overlay class designed to hold string data for pedagogical purposes.

`#include <artdaq-core-demo/Overlays/AsciiFragment.hh>`

Inheritance diagram for demo::AsciiFragment:



Classes

- struct Header

  The AsciiFragment::Header contains information about the payload size and the "line number" of the data contained in the AsciiFragment.
- struct Metadata

  Metadata describing the contents of the AsciiFragment.

Public Member Functions

- AsciiFragment (artdaq::Fragment const &f)

  The AsciiFragment constructor.
- Header::event_size_t hdr_event_size () const

  Gets the event_size from the AsciiFragment::Header.
- Header::line_number_t hdr_line_number () const

  Gets the line_number from the AsciiFragment::Header.
- size_t total_line_characters () const

  The number of characters in the line.
- char const ∗ dataBegin () const

  Start of the line, returned as a pointer to the char type.

- char const ∗ dataEnd () const

  End of the line, returned as a pointer to the char type.

Static Public Member Functions

- static constexpr size_t hdr_size_words ()

  Gets the size_words variable from the artdaq::Header.

Protected Member Functions

- Header const ∗ header_ () const

  header_() simply takes the address of the start of this overlay's data (i.e., where the AsciiFragment::Header object begins) and casts it as a pointer to AsciiFragment::Header

Static Protected Member Functions

- static constexpr size_t chars_per_word_ ()

  Translates between the size of the character type and the artda::Fragment::value_type.

5.1.1 Detailed Description

An artdaq::Fragment overlay class designed to hold string data for pedagogical purposes.

AsciiFragment is an artdaq::Fragment overlay class designed to hold string data. It serves both as an educational tool for showing how Fragment overlays works, and as a way to showcase artdaq's data-handling ability, especially how the input data is replicated bit-for-bit in the output.

Definition at line 34 of file AsciiFragment.hh.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 demo::AsciiFragment::AsciiFragment (artdaq::Fragment const & *f*) [inline],[explicit]

The AsciiFragment constructor.

Parameters

f	The raw artdaq::Fragment object to overlay

The constructor simply sets its const private member "artdaq_Fragment_" to refer to the artdaq::Fragment object

Definition at line 97 of file AsciiFragment.hh.

5.1.3 Member Function Documentation

5.1.3.1 static constexpr size_t demo::AsciiFragment::chars_per_word_ ( ) [inline],[static],[protected]

Translates between the size of the character type and the artda::Fragment::value_type.

Returns

The number of characters in a artdaq::Fragment::value_type

Definition at line 141 of file AsciiFragment.hh.

5.1.3.2 Header::event_size_t demo::AsciiFragment::hdr_event_size ( ) const [inline]

Gets the event_size from the AsciiFragment::Header.

Returns

The number of characters in the contained line

Definition at line 105 of file AsciiFragment.hh.

5.1.3.3 Header::line_number_t demo::AsciiFragment::hdr_line_number ( ) const [inline]

Gets the line_number from the AsciiFragment::Header.

Returns

The "line number" string of the AsciiFragment::Header

Definition at line 110 of file AsciiFragment.hh.

5.1.3.4 static constexpr size_t demo::AsciiFragment::hdr_size_words ( ) [inline], [static]

Gets the size_words variable from the artdaq::Header.

Returns

The size of the Fragment payload

Definition at line 115 of file AsciiFragment.hh.

5.1.3.5 Header const* demo::AsciiFragment::header_ ( ) const [inline], [protected]

header_() simply takes the address of the start of this overlay's data (i.e., where the AsciiFragment::Header object begins) and casts it as a pointer to AsciiFragment::Header

Returns

Pointer to the AsciiFragment::Header object in the data payload

Definition at line 152 of file AsciiFragment.hh.

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

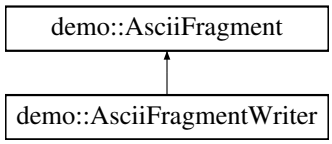
- artdaq_core_demo/artdaq-core-demo/Overlays/AsciiFragment.hh

5.2 demo::AsciiFragmentWriter Class Reference

Class derived from AsciiFragment which allows writes to the data (for simulation purposes).

```
#include <artdaq-core-demo/Overlays/AsciiFragmentWriter.hh>
```

Inheritance diagram for demo::AsciiFragmentWriter:



Public Member Functions

- AsciiFragmentWriter (artdaq::Fragment &f)

  The constructor will expect the artdaq::Fragment object it's been passed to contain the artdaq::Fragment header + the AsciiFragment::Metadata object, otherwise it throws.
- char * dataBegin ()

  Return a non-const (for writing) pointer to the beginning of the Fragment payload.
- char * dataEnd ()

  Return a non-const (for writing) pointer to the end of the Fragment payload.
- Header * header_ ()

  Returns a pointer to the AsciiFragment::Header object, non-const for writing.
- void set_hdr_line_number (Header::line_number_t line_number)

  Sets the line number in the AsciiFragment::Header to the given (ASCII-encoded) value.
- void resize (size_t nChars)

  Resize the Fragment so that it can contain nChars characters.

Additional Inherited Members

5.2.1 Detailed Description

Class derived from AsciiFragment which allows writes to the data (for simulation purposes).

Class derived from AsciiFragment which allows writes to the data (for simulation purposes). Note that for this reason it contains non-const members which hide the const members in its parent class, AsciiFragment, including its reference to the artdaq::Fragment object, artdaq_Fragment_, as well as its functions pointing to the beginning and end of the line in the fragment, dataBegin() and dataEnd()

Definition at line 24 of file AsciiFragmentWriter.hh.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 demo::AsciiFragmentWriter::AsciiFragmentWriter (artdaq::Fragment & *f*) [inline],[explicit]

The constructor will expect the artdaq::Fragment object it's been passed to contain the artdaq::Fragment header + the AsciiFragment::Metadata object, otherwise it throws.

Parameters

f	artdaq::Fragment object to overlay

Exceptions

cet::exception	if input Fragment does not contain an AsciiFragment::Metadata object

Definition at line 91 of file AsciiFragmentWriter.hh.

5.2.3 Member Function Documentation

5.2.3.1 char * demo::AsciiFragmentWriter::dataBegin ( ) [inline]

Return a non-const (for writing) pointer to the beginning of the Fragment payload.

Returns

A pointer to the beginning of the Fragment payload

Definition at line 105 of file AsciiFragmentWriter.hh.

5.2.3.2 char * demo::AsciiFragmentWriter::dataEnd ( ) [inline]

Return a non-const (for writing) pointer to the end of the Fragment payload.

Returns

A pointer to the end of the Fragment payload

Definition at line 112 of file AsciiFragmentWriter.hh.

5.2.3.3 Header* demo::AsciiFragmentWriter::header_ ( ) [inline]

Returns a pointer to the AsciiFragment::Header object, non-const for writing.

Returns

A pointer to the AsciiFragment::Header object

Definition at line 57 of file AsciiFragmentWriter.hh.

5.2.3.4 void demo::AsciiFragmentWriter::resize (size_t *nChars*) [inline]

Resize the Fragment so that it can contain nChars characters.

Parameters

nChars	Number of characters in resized Fragment

Definition at line 118 of file AsciiFragmentWriter.hh.

5.2.3.5 void demo::AsciiFragmentWriter::set_hdr_line_number (Header::line_number_t *line_number*) [inline]

Sets the line number in the AsciiFragment::Header to the given (ASCII-encoded) value.

Parameters

line_number	Set the line number (ASCII-encoded) to this value

Definition at line 67 of file AsciiFragmentWriter.hh.

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

- artdaq_core_demo/artdaq-core-demo/Overlays/AsciiFragmentWriter.hh

5.3 coredemo::GetPackageBuildInfo Struct Reference

Wrapper around the coredemo::GetPackageBuildInfo::getPackageBuildInfo function.

```
#include <artdaq-core-demo/BuildInfo/GetPackageBuildInfo.hh>
```

Static Public Member Functions

- static artdaq::PackageBuildInfo getPackageBuildInfo ()

  Gets the version number and build timestmap for artdaq_core_demo.

5.3.1 Detailed Description

Wrapper around the coredemo::GetPackageBuildInfo::getPackageBuildInfo function.

Definition at line 17 of file GetPackageBuildInfo.hh.

5.3.2 Member Function Documentation

5.3.2.1 static artdaq::PackageBuildInfo coredemo::GetPackageBuildInfo::getPackageBuildInfo ( ) [static]

Gets the version number and build timestmap for artdaq_core_demo.

Returns

An artdaq::PackageBuildInfo object containing the version number and build timestamp for artdaq_core_demo

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

- artdaq_core_demo/artdaq-core-demo/BuildInfo/GetPackageBuildInfo.hh

5.4 demo::AsciiFragment::Header Struct Reference

The AsciiFragment::Header contains information about the payload size and the "line number" of the data contained in the AsciiFragment.

```
#include <artdaq-core-demo/Overlays/AsciiFragment.hh>
```

Public Types

- typedef char data_t

  The data type of the payload.
- typedef uint64_t event_size_t

  Type for the event_size field.
- typedef uint64_t line_number_t

  Type for the line_number field.

Public Attributes

- event_size_t event_size: 28

  The number of characters in the contained string.
- event_size_t unused_1: 36

  Unused.
- line_number_t line_number

  The line number of the string (in ASCII). Equal to Event number in artdaq-demo/Generators/AsciiSimulator_generator.cc.

Static Public Attributes

- static size_t const size_words = 16ul

  Size of the Header object, in units of Header::data_t.

5.4.1 Detailed Description

The AsciiFragment::Header contains information about the payload size and the "line number" of the data contained in the AsciiFragment.

The AsciiFragment::Header contains information about the payload size and the "line number" of the data contained in the AsciiFragment. Note that the AsciiFragment::Header is distinct from the AsciiFragment::Metadata and the Fragment::Header. AsciiFragment::Header is a convienience struct to decode the first 16 bytes of the data payload.

Definition at line 74 of file AsciiFragment.hh.

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

- artdaq_core_demo/artdaq-core-demo/Overlays/AsciiFragment.hh

5.5 demo::ToyFragment::Header Struct Reference

The "Header" struct is used to interpret the header information that is created by the hardware and *is* part of the data blob that is read from the hardware for each event. Not to be confused with the Fragment::Header or the ToyFragment::Metadata objects.

```
#include <artdaq-core-demo/Overlays/ToyFragment.hh>
```

Public Types

- typedef uint32_t data_t

  Fundamental unit of data represenation within the ToyFragment::Header.
- typedef uint32_t event_size_t

  Type used for the event size.
- typedef uint32_t trigger_number_t

  Type used for the trigger number.

Public Attributes

- event_size_t event_size: 28

  The size of the event, in data_t words.
- event_size_t unused_1: 4

  Unused.
- trigger_number_t trigger_number: 32

  The "trigger number" from the simulated hardware.

Static Public Attributes

- static size_t const size_words = 2ul

  Size of the ToyFragment::Header class, in units of Header::data_t.

5.5.1 Detailed Description

The "Header" struct is used to interpret the header information that is created by the hardware and *is* part of the data blob that is read from the hardware for each event. Not to be confused with the Fragment::Header or the ToyFragment::Metadata objects.

Definition at line 77 of file ToyFragment.hh.

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

- artdaq_core_demo/artdaq-core-demo/Overlays/ToyFragment.hh

5.6 demo::UDPFragment::Header Struct Reference

The UDPFragment::Header contains information about the payload size and the "data type" of the UDP data.

```
#include <artdaq-core-demo/Overlays/UDPFragment.hh>
```

Public Types

- typedef uint32_t data_t

  The fundamental unit of Header data.
- typedef uint32_t event_size_t

  Type of the event_size field.
- typedef uint32_t data_type_t

  Type of the type field.

Public Attributes

- event_size_t event_size: 28

  The size of the payload, in words.
- event_size_t type: 4

  The type of the payload data, 0: Raw, 1: JSON, 2: String.

Static Public Attributes

- static size_t const size_words = 1ul

  Size of the UDPFragment::Header, in units of Header::data_t.

5.6.1 Detailed Description

The UDPFragment::Header contains information about the payload size and the "data type" of the UDP data.

The UDPFragment::Header contains information about the payload size and the "data type" of the UDP data. Currently-defined data types are Raw, String and JSON (defined in demo::DataType enum in artdaq-demo/Generators/UDP-Receiver.hh). Note that the UDPFragment::Header is distinct from the UDPFragment::Metadata and the Fragment::-Header. UDPFragment::Header is a convienience struct to decode the first 32 bytes of the data payload.

Notice only the first 28 bits of the first 32-bit unsigned integer in the Header is used to hold the event_size; this means that you can't represent a fragment larger than 2∗∗28 units of data_t, or 1,073,741,824 bytes

Definition at line 58 of file UDPFragment.hh.

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

- artdaq_core_demo/artdaq-core-demo/Overlays/UDPFragment.hh

5.7 demo::ToyFragment::Metadata Struct Reference

The ToyFragment::Metadata struct holds data about the readout that may not be present in the primary data stream.

```
#include <artdaq-core-demo/Overlays/ToyFragment.hh>
```

Public Types

- typedef uint64_t data_t

  Fundamental unit of data representation within the ToyFragment::Metadata.

Public Attributes

- data_t board_serial_number: 16

  Serial number of the simulated board.
- data_t num_adc_bits: 8

  Number of ADC bits this simulated board uses.
- data_t unused: 40

  Unused.

Static Public Attributes

- static size_t const size_words = 1ul

  Size of the ToyFragment::Metadata struct, in units of Metadata::data_t.

5.7.1 Detailed Description

The ToyFragment::Metadata struct holds data about the readout that may not be present in the primary data stream.

The "Metadata" struct declared here is used to define the set of information that is stored in the "metadata" section of artdaq::Fragments that contain ToyFragments in their payload. This struct is not used by the ToyFragment Overlay class itself; it is defined here simply as a convenience. The types of information that are typically stored in artdaq::Fragment metadata blocks are things like hardware serial numbers or values of an interesting configuration parameter. These things are generally *not* part of the data read out from the electronics hardware in each event, hence the need to store them outside the block of data that the hardware provides.

Definition at line 46 of file ToyFragment.hh.

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

- artdaq_core_demo/artdaq-core-demo/Overlays/ToyFragment.hh

5.8 demo::UDPFragment::Metadata Struct Reference

Metadata describing the contents of the UDPFragment.

```
#include <artdaq-core-demo/Overlays/UDPFragment.hh>
```

Public Types

- typedef uint64_t data_t

  The fundamental unit of Metadata data.

Public Attributes

- data_t port: 16

  The port on which the data was received.
- data_t address: 32

  The IPv4 address the data was received on.
- data_t unused: 16

  Unused bits of the data_t field.

Static Public Attributes

- static size_t const size_words = 1ull

  Size of the UDPFragment::Metadata object, in units of Metadata::data_t.

5.8.1 Detailed Description

Metadata describing the contents of the UDPFragment.

The AsciiFragment::Metadata struct is used to store information about the upstream environment from where the fragment came. In the case of UDPFragment, this is the port and address at which the data was received.

Definition at line 32 of file UDPFragment.hh.

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

- artdaq_core_demo/artdaq-core-demo/Overlays/UDPFragment.hh

5.9 demo::AsciiFragment::Metadata Struct Reference

Metadata describing the contents of the AsciiFragment.

```
#include <artdaq-core-demo/Overlays/AsciiFragment.hh>
```

Public Types

- typedef char data_t
- typedef uint32_t chars_in_line_t

  Type used to represent the length in characters of the AsciiFragment data.

Public Attributes

- chars_in_line_t charsInLine

  How many characters are in the data payload.

Static Public Attributes

- static size_t const size_words = 4ul

  Size of the Metadata object, in units of Metadata::data_t.

5.9.1 Detailed Description

Metadata describing the contents of the AsciiFragment.

The AsciiFragment::Metadata struct is used to store information about the upstream environment from where the fragment came. In the case of AsciiFragment, this is simply a count of the characters in the data payload.

Definition at line 45 of file AsciiFragment.hh.

5.9.2 Member Typedef Documentation

5.9.2.1 typedef char demo::AsciiFragment::Metadata::data_t

"data_t" is a typedef of the fundamental unit of data the metadata structure thinks of itself as consisting of; it can give its size via the static "size_words" variable (AsciiFragment::Metadata::size_words)

Definition at line 52 of file AsciiFragment.hh.

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

- artdaq_core_demo/artdaq-core-demo/Overlays/AsciiFragment.hh

5.10 demo::ToyFragment Class Reference

An artdaq::Fragment overlay class used for pedagogical purposes.

```
#include <artdaq-core-demo/Overlays/ToyFragment.hh>
```

Classes

- struct Header

  The "Header" struct is used to interpret the header information that is created by the hardware and is part of the data blob that is read from the hardware for each event. Not to be confused with the Fragment::Header or the ToyFragment::Metadata objects.
- struct Metadata

  The ToyFragment::Metadata struct holds data about the readout that may not be present in the primary data stream.

Public Types

- typedef uint16_t adc_t

  ToyFragment is intended to represent/interpret data that has an inherent size of 16 bits (unsigned). This is represented by the adc_t type that is declared here.

Public Member Functions

- ToyFragment (artdaq::Fragment const &f)

  The ToyFragment constructor.
- Header::event_size_t hdr_event_size () const

  Gets the event_size from the ToyFragment::Header.
- Header::trigger_number_t hdr_trigger_number () const

  Gets the trigger_number from the ToyFragment::Header.
- size_t total_adc_values () const

  Get the number of ADC values describing data beyond the header.
- adc_t adc_value (uint32_t index)

  Retrieve the given ADC value from the Fragment.
- adc_t const * dataBeginADCs () const

  Start of the ADC values, returned as a const pointer to the ADC type.
- adc_t const * dataEndADCs () const

  End of the ADC values, returned as a const pointer to the ADC type.
- adc_t const * findBadADC (int daq_adc_bits) const

  Look through stored ADC values and determine if any are out-of-range.
- bool fastVerify (int daq_adc_bits) const

  Do any ADC values in the ToyFragment appear to be out-of-range?
- void checkADCData (int daq_adc_bits) const

  Throw if any ADC value appears corrupt.

Static Public Member Functions

- static constexpr size_t hdr_size_words ()

  Gets the size_words variable from the artdaq::Header.
- static size_t adc_range (int daq_adc_bits)

  Get the largest ADC value possible for a given number of ADC bits.

Protected Member Functions

- Header const * header_ () const

  Get a pointer to the ToyFragment::Header object.

Static Protected Member Functions

- static constexpr size_t adcs_per_word_ ()

  Translates between the size of an ADC and the size in Header::data_t words.

5.10.1 Detailed Description

An artdaq::Fragment overlay class used for pedagogical purposes.

The intention of this class is to provide an example of how an Overlay can be constructed for a hypothetical 16-bit ADC module. In this example, the ADC data can have either 12 or 14 bits of meaningful information per 16-bit word. (This is just a characteristic of this example. Separate classes could certainly be used to overlay the two types of ADC data.)

Definition at line 26 of file ToyFragment.hh.

5.10.2 Constructor & Destructor Documentation

5.10.2.1 demo::ToyFragment::ToyFragment (artdaq::Fragment const & *f*) [inline],[explicit]

The ToyFragment constructor.

Parameters

f	The raw artdaq::Fragment object to overlay

The constructor simply sets its const private member "artdaq_Fragment_" to refer to the artdaq::Fragment object

Definition at line 112 of file ToyFragment.hh.

5.10.3 Member Function Documentation

5.10.3.1 static size_t demo::ToyFragment::adc_range (int *daq_adc_bits*) [inline],[static]

Get the largest ADC value possible for a given number of ADC bits.

Parameters

daq_adc_bits	Number of bits in the ADC field

Returns

Largest value possible with the given number of ADC bits

Definition at line 211 of file ToyFragment.hh.

5.10.3.2 adc_t demo::ToyFragment::adc_value (uint32_t *index*) [inline]

Retrieve the given ADC value from the Fragment.

Parameters

index	Index of the ADC value

Returns

ADC value at given index, or 0xFFFF if out-of-bounds

Definition at line 144 of file ToyFragment.hh.

5.10.3.3 static constexpr size_t demo::ToyFragment::adcs_per_word_ ( ) [inline], [static], [protected]

Translates between the size of an ADC and the size in Header::data_t words.

Returns

Number of adc_t counts per Header::data_t word

Definition at line 222 of file ToyFragment.hh.

5.10.3.4 void demo::ToyFragment::checkADCData (int *daq_adc_bits*) const

Throw if any ADC value appears corrupt.

Parameters

daq_adc_bits	Number of bits for ADC values

Exceptions

cet::exception	if any ADC value is out-of-range

Definition at line 15 of file ToyFragment.cc.

5.10.3.5 adc_t const* demo::ToyFragment::dataBeginADCs ( ) const [inline]

Start of the ADC values, returned as a const pointer to the ADC type.

Returns

const adc_t pointer to the start of the payload

Definition at line 155 of file ToyFragment.hh.

5.10.3.6 adc_t const* demo::ToyFragment::dataEndADCs ( ) const [inline]

End of the ADC values, returned as a const pointer to the ADC type.

Returns

const adc_t pointer to the end of the payload

Definition at line 164 of file ToyFragment.hh.

5.10.3.7 bool demo::ToyFragment::fastVerify (int *daq_adc_bits*) const [inline]

Do any ADC values in the ToyFragment appear to be out-of-range?

Parameters

daq_adc_bits	Number of bits for ADC values

Returns

Whether any of the ADC values in the ToyFragment are out-of-range

Definition at line 194 of file ToyFragment.hh.

5.10.3.8 adc_t const* demo::ToyFragment::findBadADC (int *daq_adc_bits*) const [inline]

Look through stored ADC values and determine if any are out-of-range.

Parameters

daq_adc_bits	Number of bits for ADC values

Returns

const adc_t pointer to first bad ADC value (if any), or dataEndADCs otherwise

findBadADC() checks to make sure that the ADC type (adc_t) variable holding the ADC value doesn't contain bits beyond the expected range, i.e., can't be evaluated to a larger value than the max permitted ADC value

Definition at line 180 of file ToyFragment.hh.

5.10.3.9 Header::event_size_t demo::ToyFragment::hdr_event_size ( ) const [inline]

Gets the event_size from the ToyFragment::Header.

Returns

The number of Header::data_t words in the ToyFragment

Definition at line 118 of file ToyFragment.hh.

5.10.3.10 static constexpr size_t demo::ToyFragment::hdr_size_words ( ) [inline], [static]

Gets the size_words variable from the artdaq::Header.

Returns

The size of the Fragment payload

Definition at line 128 of file ToyFragment.hh.

5.10.3.11 Header::trigger_number_t demo::ToyFragment::hdr_trigger_number ( ) const [inline]

Gets the trigger_number from the ToyFragment::Header.

Returns

The "trigger number" of the ToyFragment::Header

Definition at line 123 of file ToyFragment.hh.

5.10.3.12 Header const* demo::ToyFragment::header_ ( ) const [inline], [protected]

Get a pointer to the ToyFragment::Header object.

Returns

Const pointer to the ToyFragment::Header object

header_() simply takes the address of the start of this overlay's data (i.e., where the ToyFragment::Header object begins) and casts it as a const pointer to ToyFragment::Header

Definition at line 235 of file ToyFragment.hh.

5.10.3.13 size_t demo::ToyFragment::total_adc_values ( ) const [inline]

Get the number of ADC values describing data beyond the header.

Returns

The number of ADC values describing data beyond the header

Definition at line 134 of file ToyFragment.hh.

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

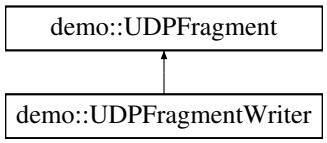
- artdaq_core_demo/artdaq-core-demo/Overlays/ToyFragment.hh
- artdaq_core_demo/artdaq-core-demo/Overlays/ToyFragment.cc

5.11 demo::UDPFragment Class Reference

A Fragment designed to contain data received from the network in UDP datagrams.

```
#include <artdaq-core-demo/Overlays/UDPFragment.hh>
```

Inheritance diagram for demo::UDPFragment:



Classes

- struct Header

  The UDPFragment::Header contains information about the payload size and the "data type" of the UDP data.
- struct Metadata

  Metadata describing the contents of the UDPFragment.

Public Member Functions

- UDPFragment (artdaq::Fragment const &f)

  The UDPFragment constructor.
- Header::event_size_t hdr_event_size () const

  Get the current value of the Header::event_size field.
- Header::data_type_t hdr_data_type () const

  Get the current value of the Header::type field.
- size_t udp_data_words () const

  Get the number of words in the UDPFragment payload.
- uint8_t const * dataBegin () const

  Returns a const pointer to the start of the UDP payload.
- uint8_t const * dataEnd () const

  Returns a const pointer to the end of the UDP payload.

Static Public Member Functions

- static constexpr size_t hdr_size_words ()

  Gets the size_words variable from the artdaq::Header.

Protected Member Functions

- Header const * header_ () const

  Get a const pointer to the UDPFragment::Header.

Static Protected Member Functions

- static constexpr size_t bytes_per_word_ ()

  Translates between data words (type Header::data_t) and bytes.

5.11.1 Detailed Description

A Fragment designed to contain data received from the network in UDP datagrams.

Definition at line 21 of file UDPFragment.hh.

5.11.2 Constructor & Destructor Documentation

5.11.2.1 demo::UDPFragment::UDPFragment (artdaq::Fragment const & *f*) [inline],[explicit]

The UDPFragment constructor.

Parameters

f	The raw artdaq::Fragment object to overlay

The constructor simply sets its const private member "artdaq_Fragment_" to refer to the artdaq::Fragment object

Definition at line 79 of file UDPFragment.hh.

5.11.3 Member Function Documentation

5.11.3.1 static constexpr size_t demo::UDPFragment::bytes_per_word_ ( ) [inline],[static],[protected]

Translates between data words (type Header::data_t) and bytes.

Returns

The number of bytes in each Header::data_t word

Definition at line 130 of file UDPFragment.hh.

5.11.3.2 uint8_t const* demo::UDPFragment::dataBegin ( ) const [inline]

Returns a const pointer to the start of the UDP payload.

Returns

const byte pointer to the start of the UDP payload

Definition at line 110 of file UDPFragment.hh.

5.11.3.3 uint8_t const* demo::UDPFragment::dataEnd ( ) const [inline]

Returns a const pointer to the end of the UDP payload.

Returns

const byte pointer to the end of the UDP payload

Definition at line 119 of file UDPFragment.hh.

5.11.3.4 Header::data_type_t demo::UDPFragment::hdr_data_type ( ) const [inline]

Get the current value of the Header::type field.

Returns

The current value of the Header::type field (const)

Definition at line 90 of file UDPFragment.hh.

5.11.3.5 Header::event_size_t demo::UDPFragment::hdr_event_size ( ) const [inline]

Get the current value of the Header::event_size field.

Returns

The current value of the Header::event_size field (const)

Definition at line 85 of file UDPFragment.hh.

5.11.3.6 static constexpr size_t demo::UDPFragment::hdr_size_words ( ) [inline], [static]

Gets the size_words variable from the artdaq::Header.

Returns

The size of the Fragment payload

Definition at line 95 of file UDPFragment.hh.

5.11.3.7 Header const* demo::UDPFragment::header_ ( ) const [inline], [protected]

Get a const pointer to the UDPFragment::Header.

Returns

A const pointer to the UDPFragment:Header

header_() simply takes the address of the start of this overlay's data (i.e., where the UDPFragment::Header object begins) and casts it as a pointer to UDPFragment::Header

Definition at line 143 of file UDPFragment.hh.

5.11.3.8 size_t demo::UDPFragment::udp_data_words ( ) const [inline]

Get the number of words in the UDPFragment payload.

Returns

The number of Header::data_t words in the UDPFragment payload (const)

Definition at line 101 of file UDPFragment.hh.

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

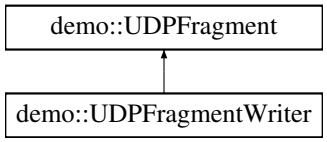
- artdaq_core_demo/artdaq-core-demo/Overlays/UDPFragment.hh

5.12 demo::UDPFragmentWriter Class Reference

Class derived from UDPFragment which allows writes to the data.

`#include <artdaq-core-demo/Overlays/UDPFragmentWriter.hh>`

Inheritance diagram for demo::UDPFragmentWriter:



Public Member Functions

- UDPFragmentWriter (artdaq::Fragment &f)

  UDPFragmentWriter constructor.
- uint8_t * dataBegin ()

  Get a pointer to the start of the UDP payload.
- uint8_t * dataEnd ()

  Get a pointer to the end of the UDP payload.
- Header * header_ ()

  Get a pointer to the UDPFragment::Header object for writing.
- void set_hdr_type (Header::data_type_t dataType)

  Setter for the Header::type field.
- void resize (size_t nBytes)

  Resize the UDP payload to the given number of bytes.

Additional Inherited Members

5.12.1 Detailed Description

Class derived from UDPFragment which allows writes to the data.

Class derived from UDPFragment which allows writes to the data. Note that for this reason it contains non-const members which hide the const members in its parent class, UDPFragment, including its reference to the artdaq::-Fragment object, artdaq_Fragment_, as well as its functions pointing to the beginning and end of the line in the fragment, dataBegin() and dataEnd(). This is necessary as the UDP data is not coming from "hardware" but the UDP stack.

Definition at line 25 of file UDPFragmentWriter.hh.

5.12.2 Constructor & Destructor Documentation

5.12.2.1 demo::UDPFragmentWriter::UDPFragmentWriter (artdaq::Fragment & *f*) [inline],[explicit]

UDPFragmentWriter constructor.

Parameters

f	The artdaq::Fragment to overlay

Exceptions

cet::exception	if the Fragment does not contain UDPFragment::Metadata and nothing else

Definition at line 93 of file UDPFragmentWriter.hh.

5.12.3 Member Function Documentation

5.12.3.1 uint8_t * demo::UDPFragmentWriter::dataBegin ( ) [inline]

Get a pointer to the start of the UDP payload.

Returns

A byte pointer to the start of the UDP payload

Definition at line 107 of file UDPFragmentWriter.hh.

5.12.3.2 uint8_t * demo::UDPFragmentWriter::dataEnd ( ) [inline]

Get a pointer to the end of the UDP payload.

Returns

A byte pointer to the end of the UDP payload

Definition at line 114 of file UDPFragmentWriter.hh.

5.12.3.3 Header* demo::UDPFragmentWriter::header_ ( ) [inline]

Get a pointer to the UDPFragment::Header object for writing.

Returns

A pointer to the UDPFragment::Header object

Definition at line 53 of file UDPFragmentWriter.hh.

5.12.3.4 void demo::UDPFragmentWriter::resize (size_t *nBytes*) [inline]

Resize the UDP payload to the given number of bytes.

Parameters

nBytes	Number of bytes to request for the UDP payload

Definition at line 120 of file UDPFragmentWriter.hh.

5.12.3.5 void demo::UDPFragmentWriter::set_hdr_type (Header::data_type_t *dataType*) [inline]

Setter for the Header::type field.

Parameters

dataType	Value to set to Header::type

Definition at line 63 of file UDPFragmentWriter.hh.

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

- artdaq_core_demo/artdaq-core-demo/Overlays/UDPFragmentWriter.hh

Index

adc_range
  demo::ToyFragment, 23
adc_value
  demo::ToyFragment, 24
adcs_per_word_
  demo::ToyFragment, 24
AsciiFragment
  demo::AsciiFragment, 12
AsciiFragmentWriter
  demo::AsciiFragmentWriter, 14

bytes_per_word_
  demo::UDPFragment, 28

chars_per_word_
  demo::AsciiFragment, 12
checkADCData
  demo::ToyFragment, 24
coredemo, 7
coredemo::GetPackageBuildInfo, 16
  getPackageBuildInfo, 16

data_t
  demo::AsciiFragment::Metadata, 21
dataBegin
  demo::AsciiFragmentWriter, 15
  demo::UDPFragment, 28
  demo::UDPFragmentWriter, 31
dataBeginADCs
  demo::ToyFragment, 24
dataEnd
  demo::AsciiFragmentWriter, 15
  demo::UDPFragment, 28
  demo::UDPFragmentWriter, 31
dataEndADCs
  demo::ToyFragment, 24
demo, 7
  fragmentTypeToString, 8
  makeFragmentTypeMap, 8
  operator<<, 9
  toFragmentType, 9
demo::AsciiFragment, 11
  AsciiFragment, 12
  chars_per_word_, 12
  hdr_event_size, 13
  hdr_line_number, 13
  hdr_size_words, 13
  header_, 13
demo::AsciiFragment::Header, 16
demo::AsciiFragment::Metadata, 21
  data_t, 21
demo::AsciiFragmentWriter, 14
  AsciiFragmentWriter, 14
  dataBegin, 15
  dataEnd, 15
  header_, 15
  resize, 15
  set_hdr_line_number, 15
demo::ToyFragment, 22
  adc_range, 23
  adc_value, 24
  adcs_per_word_, 24
  checkADCData, 24
  dataBeginADCs, 24
  dataEndADCs, 24
  fastVerify, 25
  findBadADC, 25
  hdr_event_size, 25
  hdr_size_words, 25
  hdr_trigger_number, 26
  header_, 26
  total_adc_values, 26
  ToyFragment, 23
demo::ToyFragment::Header, 17
demo::ToyFragment::Metadata, 19
demo::UDPFragment, 26
  bytes_per_word_, 28
  dataBegin, 28
  dataEnd, 28
  hdr_data_type, 28
  hdr_event_size, 29
  hdr_size_words, 29
  header_, 29
  UDPFragment, 28
  udp_data_words, 29
demo::UDPFragment::Header, 18
demo::UDPFragment::Metadata, 20
demo::UDPFragmentWriter, 30
  dataBegin, 31
  dataEnd, 31
  header_, 31

resize, 31
set_hdr_type, 31
UDPFragmentWriter, 30
demo::detail, 9

fastVerify
demo::ToyFragment, 25
findBadADC
demo::ToyFragment, 25
fragmentTypeToString
demo, 8

getPackageBuildInfo
coredemo::GetPackageBuildInfo, 16

hdr_data_type
demo::UDPFragment, 28
hdr_event_size
demo::AsciiFragment, 13
demo::ToyFragment, 25
demo::UDPFragment, 29
hdr_line_number
demo::AsciiFragment, 13
hdr_size_words
demo::AsciiFragment, 13
demo::ToyFragment, 25
demo::UDPFragment, 29
hdr_trigger_number
demo::ToyFragment, 26
header_
demo::AsciiFragment, 13
demo::AsciiFragmentWriter, 15
demo::ToyFragment, 26
demo::UDPFragment, 29
demo::UDPFragmentWriter, 31

makeFragmentTypeMap
demo, 8

operator<<
demo, 9

resize
demo::AsciiFragmentWriter, 15
demo::UDPFragmentWriter, 31

set_hdr_line_number
demo::AsciiFragmentWriter, 15
set_hdr_type
demo::UDPFragmentWriter, 31

toFragmentType
demo, 9
total_adc_values
demo::ToyFragment, 26
ToyFragment
demo::ToyFragment, 23

UDPFragment
demo::UDPFragment, 28
UDPFragmentWriter
demo::UDPFragmentWriter, 30
udp_data_words
demo::UDPFragment, 29