USB Flight Recorder

RTLib Reference

Release 2021-A – May 2021

dSPACE

How to Contact dSPACE

Mail:	dSPACE GmbH Rathenaustraße 26 33102 Paderborn Germany
Tel.:	+49 5251 1638-0
Fax:	+49 5251 16198-0
E-mail:	info@dspace.de
Web:	http://www.dspace.com

How to Contact dSPACE Support

If you encounter a problem when using dSPACE products, contact your local dSPACE representative:

- Local dSPACE companies and distributors: http://www.dspace.com/go/locations
- For countries not listed, contact dSPACE GmbH in Paderborn, Germany.
  Tel.: +49 5251 1638-941 or e-mail: support@dspace.de

You can also use the support request form: http://www.dspace.com/go/supportrequest. If you are logged on to mydSPACE, you are automatically identified and do not need to add your contact details manually.

If possible, always provide the relevant dSPACE License ID or the serial number of the CmContainer in your support request.

Software Updates and Patches

dSPACE strongly recommends that you download and install the most recent patches for your current dSPACE installation. Visit http://www.dspace.com/go/patches for software updates and patches.

Important Notice

Contents

About This Reference

Content

The USB Flight Recorder Real-Time Library (RTLib) provides the C functions and macros you need to handcode the USB flight recorder feature in your real-time applications.

The following hardware is supported:

- MicroAutoBox II
- MicroLabBox
- DS1007 PPC Processor Board

Symbols

dSPACE user documentation uses the following symbols:

Symbol	Description
⚠ DANGER	Indicates a hazardous situation that, if not avoided, will result in death or serious injury.
⚠ WARNING	Indicates a hazardous situation that, if not avoided, could result in death or serious injury.
⚠ CAUTION	Indicates a hazardous situation that, if not avoided, could result in minor or moderate injury.
NOTICE	Indicates a hazard that, if not avoided, could result in property damage.
Note	Indicates important information that you should take into account to avoid malfunctions.
Tip	Indicates tips that can make your work easier.
⍰	Indicates a link that refers to a definition in the glossary, which you can find at the end of the document unless stated otherwise.
📖	Precedes the document title in a link that refers to another document.

Naming conventions

dSPACE user documentation uses the following naming conventions:

%name% Names enclosed in percent signs refer to environment variables for file and path names.

< > Angle brackets contain wildcard characters or placeholders for variable file and path names, etc.

Special folders

Some software products use the following special folders:

Common Program Data folder A standard folder for application-specific configuration data that is used by all users.

`%PROGRAMDATA%\dSPACE\<InstallationGUID>\<ProductName>`

or

`%PROGRAMDATA%\dSPACE\<ProductName>\<VersionNumber>`

Documents folder A standard folder for user-specific documents.

`%USERPROFILE%\Documents\dSPACE\<ProductName>\<VersionNumber>`

Local Program Data folder A standard folder for application-specific configuration data that is used by the current, non-roaming user.

`%USERPROFILE%\AppData\Local\dSPACE\<InstallationGUID>\<ProductName>`

Accessing dSPACE Help and PDF Files

After you install and decrypt dSPACE software, the documentation for the installed products is available in dSPACE Help and as PDF files.

dSPACE Help (local) You can open your local installation of dSPACE Help:

- On its home page via Windows Start Menu
- On specific content using context-sensitive help via **F1**

dSPACE Help (Web) You can access the Web version of dSPACE Help at www.dspace.com.

To access the Web version, you must have a *mydSPACE* account.

PDF files You can access PDF files via the icon in dSPACE Help. The PDF opens on the first page.

USB Flight Recorder

Introduction

With the USB Flight Recorder you can do long-term data acquisition. During the simulation, the values of selectable variables are written to the connected USB mass storage device.

Where to go from here

Information in this section

Introduction to the USB Flight Recorder

Purpose

With the USB Flight Recorder you can do long-term data acquisition. During the simulation, the values of selectable variables are written to a connected USB mass storage device.

Description

Any standard USB mass storage device can be used like, for example, a USB memory stick or an external USB hard drive with or without separate power supply. The USB device must be formatted with the Microsoft FAT32 file system and must be directly connected to the hardware. Connection via USB hubs is not supported.

The recorded data is written to a sequence of files stored in the root directory of the USB device. The file names are generated automatically and contain the name of the real-time model as well as the creation date and time of the file. Only one file is written at a time. The file grows until it reaches a user-defined maximum size (refer to **`dsflrec_usb_initialize`** on page 9). After that, the file is closed and a new output file is created. As long as the USB Flight Recording session runs, new files are generated until the maximum number of files has been reached. The maximum file number is given by the size of the USB device divided by the maximum file size.

When the maximum number of files is reached, either the oldest file is deleted or the USB Flight Recording session is stopped (refer to **`dsflrec_usb_initialize`** on page 9).

On multicore platforms such as the DS1007 PPC Processor Board or MicroLabBox, the USB Flight Recorder is separately configured for each real-time application running on the board. Each instantiated USB Flight Recorder generates its own output files.

The real-time model continues to run in any case, even if the USB Flight Recording session is stopped.

Note

To prevent data loss, you should remove the USB device by using the **`dsflrec_usb_eject`** function before you unplug the USB memory stick or hard drive.

Characteristics

For information on the USB Flight Recorder's characteristics, for example, the maximum data rate or the maximum number of variables, refer to one of the following descriptions:

- USB Flight Recorder (DS1007 Features 🕮)
- USB Flight Recorder (MicroAutoBox II Features 🕮)
- USB Flight Recorder (MicroLabBox Features 🕮)

There you find also instructions how to use the USB Flight Recorder.

dsflrec_usb_eject

Syntax

```
void dsflrec_usb_eject()
```

Include file

`DsFlRecUSB.h`

Purpose

To safely eject a connected USB mass storage device.

Description

This function can be used to safely eject the USB mass storage device used for flight recording while an application is running to avoid data loss.

The function can be called if an external event is detected which must result in the USB device being ejected. For example, the application could monitor an I/O pin which could be connected to a user-defined *Eject button*.

The real-time application has to contain the RTLIB_BACKGROUND_SERVICE call.

Parameters

None

Return value

None

Related topics

References

dsflrec_usb_initialize

Syntax

```
Int16 dsflrec_usb_initialize(
   UInt16 RecordingMode,
   UInt16 MaxFileSizeMB)
```

Include file

`DsFlRecUSB.h`

Purpose

To initialize the USB Flight Recorder.

Description

Upon initialization of the USB Flight Recorder the user must specify the maximum size of a single file on the USB device as well the recording mode.

The maximum file size specifies the maximum size to which a single USB Flight Recording file can grow. After that, the file is closed and a a new file is opened.

The recording mode specifies the behavior of the USB Flight Recorder when the maximum number of files is reached. The maximum number of files is results from the total memory size of the USB device divided by the maximum file size. If the device runs out of space, the USB Flight Recorder can either discard the oldest data by deleting the oldest file or discard the newest data by stopping any further recording.

Parameters

RecordingMode Specifies the behavior of the USB Flight Recorder when the maximum number of files stored on the USB device is reached.

Symbol	Meaning
DSFLREC_USB_MODE_DISCARD_NEW_DATA	The USB Flight Recording session is stopped. This effectively discards any new data.
DSFLREC_USB_MODE_OVERWRITE	The oldest file is deleted before continuing. This effectively discards the oldest data.

MaxFileSizeMB Specifies the maximum size in MB of a single file created during USB Flight Recording. Minimum value is 1 MB, maximum value is 256 MB. The recommended value is 32 MB.

Return value

This function returns an error code. The following symbols are predefined:

Symbol	Meaning
DSFLREC_USB_ALREADY_INITIALIZED	The USB Flight Recorder can only be used once within an application.
DSFLREC_USB_ILLEGAL_FILESIZE	The specified value for MaxFileSizeMB is not valid. For the valid range, refer to the description of the `MaxFileSizeMB` parameter.
DSFLREC_USB_NO_ERROR	The USB Flight Recorder was successfully initialized.

Example #1

Suppose you have a 4 GB USB memory stick connected to the board.

```
Int16 Ierror;
...
Ierror = dsflrec_usb_initialize(
   DSFLREC_USB_MODE_OVERWRITE,
   32);
```

The USB Flight Recorder generates a sequence of files. A single file can grow to a size of at most 32 MB. At most 128 files (= 4 GB /32 MB) can be stored on the memory stick. Every time this number is reached, the oldest file is deleted and the USB Flight Recording session continues.

Example #2

Suppose you have a 4 GB USB memory stick connected to the board.

```
Int16 Ierror;
...
Ierror = dsflrec_usb_initialize(
   DSFLREC_USB_MODE_DISCARD_NEW_DATA,
   16);
```

The USB Flight Recorder generates a sequence of files. A single file can grow to a size of at most 16 MB. At most 256 files (= 4 GB / 16 MB) can be stored on the memory stick. When this number is reached, the USB Flight Recording session is stopped.

Related topics

References

dsflrec_usb_add_variable

Syntax

```
Int16 dsflrec_usb_add_variable(
   char *VarName,
   UInt16 DataTypeId,
   UInt16 *VarIndex)
```

Include file

`DsFlRecUSB.h`

Purpose

To add a variable to the USB Flight Recorder session.

Description

The function is used to add a new variable to a USB Flight Recorder session. The variable is specified by its name and data type. By adding a variable to a USB Flight Recorder session the variable is associated with a unique identifier, the so-called variable index, which is used to address this variable when recording data with the **`dsflrec_usb_write_int32`** and **`dsflrec_usb_write_float32`** functions. All the variables added to a session must have different names.

Parameters

VarName Specifies the pointer to the C-string that holds the name of the variable to be recorded.

DataTypeId Specifies the identifier for the variable's data type.

Symbol	Meaning
DSFLREC_DATA_TYPE_INT8	8-bit integer
DSFLREC_DATA_TYPE_UINT8	8-bit unsigned integer
DSFLREC_DATA_TYPE_INT16	16-bit integer
DSFLREC_DATA_TYPE_UINT16	16-bit unsigned integer
DSFLREC_DATA_TYPE_INT32	32-bit integer
DSFLREC_DATA_TYPE_UINT32	32-bit unsigned integer
DSFLREC_DATA_TYPE_FLOAT32	32-bit float
DSFLREC_DATA_TYPE_FLOAT64	64-bit float

VarIndex Specifies the pointer to the memory location where the new variable variable index is returned. Use the returned value when calling **`dsflrec_usb_write_int32`** or **`dsflrec_usb_write_float32`**.

Return value

This function returns an error code:

Symbol	Meaning
DSFLREC_USB_NO_ERROR	The variable has been successfully added to the USB Flight Recorder session.
DSFLREC_USB_INIT_TABLE_FULL	The variable has not been added to the USB Flight Recorder session because the maximum number of variables (250) has been reached.
DSFLREC_USB_VAR_ALREADY_EXIST	The variable has not been added to the USB Flight Recorder session because a variable with the same name already exists.

Example

This example shows how to add a variable to the USB Flight Recorder.

```
char MyVar="ModelRoot\Sensor1\Out";
Int16 Ierror;
Int16 VarIndex;
...
Ierror = dsflrec_usb_add_variable(
   MyVar,
   DSFLREC_DATA_TYPE_INT32,
   &VarIndex);
```

Related topics

References

dsflrec_usb_write_int32

Syntax

```
Int16 dsflrec_usb_write_int32(
   UInt16 VarIndex,
   UInt32 Value)
```

Include file

`DsFlRecUSB.h`

Purpose

To write a variable's 32-bit integer value to the USB Flight Recorder.

Description

The function writes a variable's 32-bit integer value to the USB Flight Recorder. The variable is specified by its unique variable index as returned by **`dsflrec_usb_add_variable`** on page 11. If the variable actually has a different integer type, you have to cast it to `UInt32` before calling this function.

Parameters

VarIndex Specifies the unique variable index as returned by `dsflrec_usb_add_variable`.

Value Specifies the value of the variable that is to be written.

Return value

This function returns an error code:

Symbol	Meaning
DSFLREC_USB_NO_ERROR	The function has been performed without error.
DSFLREC_USB_ILLEGAL_VARINDEX	The variable index is not within the allowed range of 1 … 250.
DSFLREC_USB_WRITE_FAILED	Writing failed due to an overload situation (data rate is too high).

Example

This example shows how to write a 16-bit integer value to the USB Flight Recorder.

```
UInt16   TestDummy = 100;
UInt16   VarIndex;
dsflrec_usb_initialize(
      DSFLREC_USB_MODE_OVERWRITE,
      128);
dsflrec_usb_add_variable(
      "TestDummy",
      DSFLREC_DATA_TYPE_UINT16,
      &VarIndex);
dsflrec_usb_write_int32 (VarIndex, (Int32) TestDummy);
```

Related topics

References

dsflrec_usb_add_variable	11
dsflrec_usb_initialize	9
dsflrec_usb_write_float32	14

dsflrec_usb_write_float32

Syntax

```
Int16 dsflrec_usb_write_float32(
   UInt16 VarIndex,
   Float32 Value)
```

Include file

`DsFlRecUSB.h`

Purpose

To write a variable's 32-bit float value to the USB Flight Recorder.

Description

The function writes a variable's 32-bit float value to the USB Flight Recorder. The variable is specified by its unique variable index as returned by **`dsflrec_usb_add_variable`** on page 11. If the variable actually has a different integer type, you have to cast it to `Float32` before calling this function.

Parameters

VarIndex Specifies the unique variable index as returned by `dsflrec_usb_add_variable`.

Value Specifies the value of the variable that is to be written.

Return value

This function returns an error code:

Symbol	Meaning
DSFLREC_USB_NO_ERROR	The function has been performed without error.
DSFLREC_USB_ILLEGAL_VARINDEX	The variable index is not within the allowed range of 1 … 250.
DSFLREC_USB_WRITE_FAILED	Writing failed due to an overload situation (data rate is too high).

Example

This example shows how to write a 16-bit float value to the USB Flight Recorder.

```
Float64   TestDummy = 100.0;
UInt16   VarIndex;
dsflrec_usb_initialize(
      DSFLREC_USB_MODE_OVERWRITE,
      128);
dsflrec_usb_add_variable(
      "TestDummy",
      DSFLREC_DATA_TYPE_UINT16,
      &VarIndex);
dsflrec_usb_write_float32 (VarIndex, (Float32) TestDummy);
```

Related topics

References