

libcin

Generated by Doxygen 1.8.6

Thu Jul 6 2017 18:37:58



# Contents

<b>1</b>	<b>Module Index</b>	<b>1</b>
1.1	Modules . . . . .	1
<b>2</b>	<b>Class Index</b>	<b>3</b>
2.1	Class List . . . . .	3
<b>3</b>	<b>File Index</b>	<b>5</b>
3.1	File List . . . . .	5
<b>4</b>	<b>Module Documentation</b>	<b>7</b>
4.1	Cin Control Routines . . . . .	7
4.1.1	Detailed Description . . . . .	8
4.1.2	Function Documentation . . . . .	8
4.1.2.1	<a href="#">cin_ctl_destroy</a> . . . . .	8
4.1.2.2	<a href="#">cin_ctl_init</a> . . . . .	8
4.1.2.3	<a href="#">cin_ctl_read</a> . . . . .	9
4.1.2.4	<a href="#">cin_ctl_stream_write</a> . . . . .	10
4.1.2.5	<a href="#">cin_ctl_write</a> . . . . .	10
4.1.2.6	<a href="#">cin_ctl_write_with_readback</a> . . . . .	10
4.2	CIN Data Framestore Functions . . . . .	11
<b>5</b>	<b>Class Documentation</b>	<b>13</b>
5.1	<a href="#">cin_ctl Struct Reference</a> . . . . .	13
5.2	<a href="#">cin_ctl_config Struct Reference</a> . . . . .	13
5.3	<a href="#">cin_ctl_id Struct Reference</a> . . . . .	14
5.4	<a href="#">cin_ctl_listener Struct Reference</a> . . . . .	14
5.5	<a href="#">cin_ctl_pwr_mon_t Struct Reference</a> . . . . .	14
5.6	<a href="#">cin_ctl_pwr_val Struct Reference</a> . . . . .	14
5.7	<a href="#">cin_data Struct Reference</a> . . . . .	15
5.8	<a href="#">cin_data_callbacks Struct Reference</a> . . . . .	15
5.9	<a href="#">cin_data_frame Struct Reference</a> . . . . .	16
5.10	<a href="#">cin_data_packet Struct Reference</a> . . . . .	16
5.11	<a href="#">cin_data_proc Struct Reference</a> . . . . .	16

5.12	<a href="#">cin_data_stats Struct Reference</a>	16
5.13	<a href="#">cin_data_threads Struct Reference</a>	17
5.14	<a href="#">cin_map_t Struct Reference</a>	17
5.15	<a href="#">cin_port Struct Reference</a>	17
5.16	<a href="#">descramble_map_t Struct Reference</a>	18
5.17	<a href="#">fifo Struct Reference</a>	18
<b>6</b>	<b>File Documentation</b>	<b>19</b>
6.1	<a href="#">src/cin.h File Reference</a>	19
6.1.1	<a href="#">Detailed Description</a>	24
6.1.2	<a href="#">LICENSE</a>	24
6.1.3	<a href="#">DESCRIPTION</a>	24
6.1.4	<a href="#">Function Documentation</a>	24
6.1.4.1	<a href="#">cin_data_framestore_disable</a>	24
6.1.4.2	<a href="#">cin_data_framestore_skip</a>	25
6.1.4.3	<a href="#">cin_data_framestore_trigger</a>	25
6.1.4.4	<a href="#">cin_data_framestore_trigger_enable</a>	25
6.1.4.5	<a href="#">cin_data_get_framestore_counter</a>	26
6.1.4.6	<a href="#">cin_data_init</a>	26
	<b>Index</b>	<b>27</b>

# Chapter 1

## Module Index

### 1.1 Modules

Here is a list of all modules:

Cin Control Routines . . . . .	<a href="#">7</a>
CIN Data Framestore Functions . . . . .	<a href="#">11</a>



## Chapter 2

# Class Index

### 2.1 Class List

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

<a href="#">cin_ctl</a>	13
<a href="#">cin_ctl_config</a>	13
<a href="#">cin_ctl_id</a>	14
<a href="#">cin_ctl_listener</a>	14
<a href="#">cin_ctl_pwr_mon_t</a>	14
<a href="#">cin_ctl_pwr_val</a>	14
<a href="#">cin_data</a>	15
<a href="#">cin_data_callbacks</a>	15
<a href="#">cin_data_frame</a>	16
<a href="#">cin_data_packet</a>	16
<a href="#">cin_data_proc</a>	16
<a href="#">cin_data_stats</a>	16
<a href="#">cin_data_threads</a>	17
<a href="#">cin_map_t</a>	17
<a href="#">cin_port</a>	17
<a href="#">descramble_map_t</a>	18
<a href="#">fifo</a>	18





## Chapter 3

# File Index

### 3.1 File List

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

src/ <a href="#">cin.h</a> . . . . .	19
src/ <b>cin_register_map.h</b> . . . . .	??
src/ <b>cinregisters.h</b> . . . . .	??
src/ <b>common.h</b> . . . . .	??
src/ <b>config.h</b> . . . . .	??
src/ <b>control.h</b> . . . . .	??
src/ <b>data.h</b> . . . . .	??
src/ <b>descramble.h</b> . . . . .	??
src/ <b>descramble_map.h</b> . . . . .	??
src/ <b>fclk_program.h</b> . . . . .	??
src/ <b>fifo.h</b> . . . . .	??
src/ <b>report.h</b> . . . . .	??
src/ <b>version.h</b> . . . . .	??



## Chapter 4

# Module Documentation

### 4.1 Cin Control Routines

#### Functions

- int [cin\\_ctl\\_init](#) ([cin\\_ctl\\_t](#) \*cin, const char \*ipaddr, uint16\_t oport, uint16\_t ippor, uint16\_t soport, uint16\_t siport)
- int [cin\\_ctl\\_destroy](#) ([cin\\_ctl\\_t](#) \*cin)
- int [cin\\_ctl\\_read](#) ([cin\\_ctl\\_t](#) \*cin, uint16\_t reg, uint16\_t \*val)
- int [cin\\_ctl\\_write](#) ([cin\\_ctl\\_t](#) \*cin, uint16\_t reg, uint16\_t val, int wait)
- int [cin\\_ctl\\_stream\\_write](#) ([cin\\_ctl\\_t](#) \*cin, char \*val, int size)
- int [cin\\_ctl\\_write\\_with\\_readback](#) ([cin\\_ctl\\_t](#) \*cin, uint16\_t reg, uint16\_t val)
- int [cin\\_ctl\\_pwr](#) ([cin\\_ctl\\_t](#) \*cin, int pwr)
- int [cin\\_ctl\\_fp\\_pwr](#) ([cin\\_ctl\\_t](#) \*cin, int pwr)
- int [cin\\_ctl\\_fo\\_test\\_pattern](#) ([cin\\_ctl\\_t](#) \*cin, int on\_off)
- int [cin\\_ctl\\_load\\_config](#) ([cin\\_ctl\\_t](#) \*cin, char \*filename)
- int [cin\\_ctl\\_load\\_firmware](#) ([cin\\_ctl\\_t](#) \*cin, char \*filename)
- int [cin\\_ctl\\_set\\_fclk](#) ([cin\\_ctl\\_t](#) \*cin, int clkfreq)
- int [cin\\_ctl\\_get\\_fclk](#) ([cin\\_ctl\\_t](#) \*cin, int \*clkfreq)
- int [cin\\_ctl\\_freeze\\_dco](#) ([cin\\_ctl\\_t](#) \*cin, int freeze)
- int [cin\\_ctl\\_get\\_cfg\\_fpga\\_status](#) ([cin\\_ctl\\_t](#) \*cin, uint16\_t \*\_val)
- int [cin\\_ctl\\_get\\_id](#) ([cin\\_ctl\\_t](#) \*cin, [cin\\_ctl\\_id\\_t](#) \*\_val)
- void [cin\\_ctl\\_display\\_id](#) (FILE \*out, [cin\\_ctl\\_id\\_t](#) val)
- void [cin\\_ctl\\_display\\_fpga\\_status](#) (FILE \*out, uint16\_t val)
- int [cin\\_ctl\\_get\\_dcm\\_status](#) ([cin\\_ctl\\_t](#) \*cin, uint16\_t \*\_val)
- void [cin\\_ctl\\_display\\_dcm\\_status](#) (FILE \*out, uint16\_t \*\_val)
- double [cin\\_ctl\\_current\\_calc](#) (uint16\_t val)
- int [cin\\_ctl\\_get\\_power\\_status](#) ([cin\\_ctl\\_t](#) \*cin, int full, int \*pwr, [cin\\_ctl\\_pwr\\_mon\\_t](#) \*values)
- void [cin\\_ctl\\_display\\_pwr](#) (FILE \*out, [cin\\_ctl\\_pwr\\_mon\\_t](#) \*values)
- void [cin\\_ctl\\_display\\_pwr\\_line](#) (FILE \*out, const char \*msg, [cin\\_ctl\\_pwr\\_val\\_t](#) val)
- int [cin\\_ctl\\_calc\\_vi\\_status](#) ([cin\\_ctl\\_t](#) \*cin, uint16\_t vreg, uint16\_t ireg, double vfact, [cin\\_ctl\\_pwr\\_val\\_t](#) \*vi)
- int [cin\\_ctl\\_get\\_camera\\_pwr](#) ([cin\\_ctl\\_t](#) \*cin, int \*val)
- int [cin\\_ctl\\_set\\_camera\\_pwr](#) ([cin\\_ctl\\_t](#) \*cin, int val)
- int [cin\\_ctl\\_set\\_bias](#) ([cin\\_ctl\\_t](#) \*cin, int val)
- int [cin\\_ctl\\_get\\_bias](#) ([cin\\_ctl\\_t](#) \*cin, int \*val)
- int [cin\\_ctl\\_set\\_clocks](#) ([cin\\_ctl\\_t](#) \*cin, int val)
- int [cin\\_ctl\\_get\\_clocks](#) ([cin\\_ctl\\_t](#) \*cin, int \*val)
- int [cin\\_ctl\\_set\\_trigger](#) ([cin\\_ctl\\_t](#) \*cin, int val)
- int [cin\\_ctl\\_get\\_trigger](#) ([cin\\_ctl\\_t](#) \*cin, int \*val)
- int [cin\\_ctl\\_set\\_focus](#) ([cin\\_ctl\\_t](#) \*cin, int val)

- int **cin\_ctl\_get\_focus** (cin\_ctl\_t \*cin, int \*val)
- int **cin\_ctl\_get\_triggering** (cin\_ctl\_t \*cin, int \*trigger)
- int **cin\_ctl\_int\_trigger\_start** (cin\_ctl\_t \*cin, int nimages)
- int **cin\_ctl\_int\_trigger\_stop** (cin\_ctl\_t \*cin)
- int **cin\_ctl\_ext\_trigger\_start** (cin\_ctl\_t \*cin, int trigger\_mode)
- int **cin\_ctl\_ext\_trigger\_stop** (cin\_ctl\_t \*cin)
- int **cin\_ctl\_set\_exposure\_time** (cin\_ctl\_t \*cin, float e\_time)
- int **cin\_ctl\_set\_trigger\_delay** (cin\_ctl\_t \*cin, float t\_time)
- int **cin\_ctl\_set\_cycle\_time** (cin\_ctl\_t \*cin, float ftime)
- int **cin\_ctl\_frame\_count\_reset** (cin\_ctl\_t \*cin)
- int **cin\_ctl\_set\_mux** (cin\_ctl\_t \*cin, int setting)
- int **cin\_ctl\_get\_mux** (cin\_ctl\_t \*cin, int \*setting)
- int **cin\_ctl\_set\_fcric\_gain** (cin\_ctl\_t \*cin, int gain)
- int **cin\_ctl\_set\_fabric\_address** (cin\_ctl\_t \*cin, char \*ip)
- int **cin\_ctl\_reg\_dump** (cin\_ctl\_t \*cin, FILE \*fp)
- int **cin\_ctl\_get\_bias\_voltages** (cin\_ctl\_t \*cin, float \*voltage)
- int **cin\_ctl\_set\_bias\_voltages** (cin\_ctl\_t \*cin, float \*voltage)
- int **cin\_ctl\_set\_fcric\_clamp** (cin\_ctl\_t \*cin, int clamp)

#### 4.1.1 Detailed Description

#### 4.1.2 Function Documentation

##### 4.1.2.1 int cin\_ctl\_destroy ( cin\_ctl\_t \* cin )

Destroy (close) the cin control library

Close connections, free memory and exit library

Parameters

<i>cin</i>	handle to cin library
------------	-----------------------

Returns

Returns 0 on success non-zero if error

##### 4.1.2.2 int cin\_ctl\_init ( cin\_ctl\_t \* cin, const char \* ipaddr, uint16\_t oport, uint16\_t iport, uint16\_t soport, uint16\_t siport )

Initialize the cin control library

Initialize the control structures and communications with the CIN via the control interface. This function opens the UDP ports and starts a listening thread to receive packets from the CIN.

Parameters

<i>cin</i>	handle to cin library
<i>ipaddr</i>	ip address of CIN base address
<i>oport</i>	output udp port of cin
<i>iport</i>	input udp port of cin
<i>soport</i>	stream output udp port of cin
<i>siport</i>	stream input udp port of cin

Returns

Returns 0 on success non-zero if error

4.1.2.3 `int cin_ctl_read ( cin_ctl_t * cin, uint16_t reg, uint16_t * val )`

Read register from CIN

**Parameters**

<i>cin</i>	handle to cin library
<i>reg</i>	register to read
<i>val</i>	variable to read value of register to

**Returns**

Returns 0 on success non-zero if error

**4.1.2.4 int cin\_ctl\_stream\_write ( cin\_ctl\_t \* cin, char \* val, int size )**

Write stream data to CIN

**Parameters**

<i>cin</i>	handle to cin library
<i>val</i>	array of values to write
<i>size</i>	size of array pointed to by val

Write stream data to cin in form of 16 bit array.

**Returns**

Returns 0 on success non-zero if error

**4.1.2.5 int cin\_ctl\_write ( cin\_ctl\_t \* cin, uint16\_t reg, uint16\_t val, int wait )**

Write register to CIN

**Parameters**

<i>cin</i>	handle to cin library
<i>reg</i>	register to write to
<i>val</i>	value to write to register
<i>wait</i>	if non-zero

Write register value to CIN. If wait is non-zero then wait a sleep time of i CIN\_CTL\_WRITE\_SLEEP before releasing the mutex to add flow control to the cin.

**Returns**

Returns 0 on success non-zero if error

**4.1.2.6 int cin\_ctl\_write\_with\_readback ( cin\_ctl\_t \* cin, uint16\_t reg, uint16\_t val )**

Write register to CIN with readback verification

**Parameters**

<i>cin</i>	handle to cin library
<i>reg</i>	register to write to
<i>val</i>	value to write to register

Write register value to CIN. Follow write with read of register and compare value. CIN\_CTL\_WRITE\_SLEEP before releasing the mutex to add flow control to the cin.

**Returns**

Returns 0 on success non-zero if error

## 4.2 CIN Data Framestore Functions

Data group





## Chapter 5

# Class Documentation

### 5.1 cin\_ctl Struct Reference

#### Public Attributes

- [cin\\_port\\_t](#) **ctl\_port**
- [cin\\_port\\_t](#) **stream\_port**
- [cin\\_ctl\\_config\\_t](#) **config**
- [cin\\_ctl\\_listener\\_t](#) \* **listener**
- [pthread\\_mutex\\_t](#) **access**
- [pthread\\_mutexattr\\_t](#) **access\_attr**

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

- [src/cin.h](#)

### 5.2 cin\_ctl\_config Struct Reference

#### Public Attributes

- char **name** [CIN\_CONFIG\_MAX\_STRING]
- char **firmware\_filename** [CIN\_CONFIG\_MAX\_STRING]
- int **overscan**
- int **columns**
- int **fclk**
- uint16\_t **timing** [CIN\_CONFIG\_MAX\_DATA][2]
- int **timing\_len**
- uint16\_t **fcric** [CIN\_CONFIG\_MAX\_DATA][2]
- int **fcric\_len**
- uint16\_t **bias** [CIN\_CONFIG\_MAX\_DATA][2]
- int **bias\_len**

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

- [src/cin.h](#)

## 5.3 cin\_ctl\_id Struct Reference

### Public Attributes

- uint16\_t **board\_id**
- uint16\_t **serial\_no**
- uint16\_t **fpga\_ver**

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

- src/[cin.h](#)

## 5.4 cin\_ctl\_listener Struct Reference

### Public Attributes

- struct [cin\\_port](#) \* **cp**
- [fifo](#) **ctl\_fifo**
- pthread\_t **thread\_id**

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

- src/[cin.h](#)

## 5.5 cin\_ctl\_pwr\_mon\_t Struct Reference

### Public Attributes

- [cin\\_ctl\\_pwr\\_val\\_t](#) **bus\_12v0**
- [cin\\_ctl\\_pwr\\_val\\_t](#) **mgmt\_3v3**
- [cin\\_ctl\\_pwr\\_val\\_t](#) **mgmt\_2v5**
- [cin\\_ctl\\_pwr\\_val\\_t](#) **mgmt\_1v2**
- [cin\\_ctl\\_pwr\\_val\\_t](#) **enet\_1v0**
- [cin\\_ctl\\_pwr\\_val\\_t](#) **s3e\_3v3**
- [cin\\_ctl\\_pwr\\_val\\_t](#) **gen\_3v3**
- [cin\\_ctl\\_pwr\\_val\\_t](#) **gen\_2v5**
- [cin\\_ctl\\_pwr\\_val\\_t](#) **v6\_0v9**
- [cin\\_ctl\\_pwr\\_val\\_t](#) **v6\_1v0**
- [cin\\_ctl\\_pwr\\_val\\_t](#) **v6\_2v5**
- [cin\\_ctl\\_pwr\\_val\\_t](#) **fp**

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

- src/[cin.h](#)

## 5.6 cin\_ctl\_pwr\_val Struct Reference

### Public Attributes

- double **i**

- double **v**

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

- [src/cin.h](#)

## 5.7 cin\_data Struct Reference

### Public Attributes

- [fifo](#) \* **packet\_fifo**
- [fifo](#) \* **frame\_fifo**
- [fifo](#) \* **image\_fifo**
- [cin\\_data\\_threads\\_t](#) **listen\_thread**
- [cin\\_data\\_threads\\_t](#) **assembler\_thread**
- [cin\\_data\\_threads\\_t](#) **descramble\_thread**
- [pthread\\_mutex\\_t](#) **listen\_mutex**
- [pthread\\_mutex\\_t](#) **assembler\_mutex**
- [pthread\\_mutex\\_t](#) **descramble\_mutex**
- [pthread\\_mutex\\_t](#) **stats\_mutex**
- [pthread\\_mutex\\_t](#) **framestore\_mutex**
- [cin\\_data\\_callbacks\\_t](#) **callbacks**
- [cin\\_port\\_t](#) **dp**
- struct timespec **framerate**
- unsigned long int **dropped\_packets**
- unsigned long int **malformed\_packets**
- uint16\_t **last\_frame**
- [descramble\\_map\\_t](#) **map**
- int **framestore\_mode**
- struct timespec **framestore\_trigger**
- int **framestore\_counter**

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

- [src/cin.h](#)

## 5.8 cin\_data\_callbacks Struct Reference

### Public Attributes

- void (\*)(**push**)([cin\\_data\\_frame\\_t](#) \*)
- void (\*)(**pop**)([cin\\_data\\_frame\\_t](#) \*)
- [cin\\_data\\_frame\\_t](#) \* **frame**

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

- [src/cin.h](#)

## 5.9 cin\_data\_frame Struct Reference

### Public Attributes

- uint16\_t \* **data**
- uint16\_t **number**
- struct timespec **timestamp**
- int **size\_x**
- int **size\_y**
- void \* **usr\_ptr**

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

- [src/cin.h](#)

## 5.10 cin\_data\_packet Struct Reference

### Public Attributes

- unsigned char \* **data**
- int **size**
- struct timespec **timestamp**

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

- [src/data.h](#)

## 5.11 cin\_data\_proc Struct Reference

### Public Attributes

- void (\*)(**input\_get**)(void \*, int)
- void (\*)(**input\_put**)(void \*, int)
- void \* **input\_args**
- int **reader**
- void (\*)(**output\_put**)(void \*)
- void (\*)(**output\_get**)(void \*)
- void \* **output\_args**
- [cin\\_data\\_t](#) \* **parent**

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

- [src/data.h](#)

## 5.12 cin\_data\_stats Struct Reference

### Public Attributes

- int **last\_frame**
- double **framerate**

- double **datarate**
- double **packet\_percent\_full**
- double **frame\_percent\_full**
- double **image\_percent\_full**
- long int **packet\_overruns**
- long int **frame\_overruns**
- long int **image\_overruns**
- long int **packet\_used**
- long int **frame\_used**
- long int **image\_used**
- long int **dropped\_packets**
- long int **mallformed\_packets**

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

- [src/cin.h](#)

## 5.13 cin\_data\_threads Struct Reference

### Public Attributes

- pthread\_t **thread\_id**
- int **started**

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

- [src/cin.h](#)

## 5.14 cin\_map\_t Struct Reference

### Public Attributes

- char \* **name**
- uint16\_t **reg**

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

- [src/cinregisters.h](#)

## 5.15 cin\_port Struct Reference

### Public Attributes

- char \* **srvaddr**
- char \* **cliaddr**
- uint16\_t **srvport**
- uint16\_t **cliport**
- int **sockfd**
- struct timeval **tv**
- struct sockaddr\_in **sin\_srv**
- struct sockaddr\_in **sin\_cli**

- socklen\_t **slen**
- int **rcvbuf**
- int **rcvbuf\_rb**

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

- src/[cin.h](#)

## 5.16 descramble\_map\_t Struct Reference

### Public Attributes

- uint32\_t \* **map**
- int **size\_x**
- int **size\_y**
- int **overscan**
- int **rows**

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

- src/[cin.h](#)

## 5.17 fifo Struct Reference

### Public Attributes

- void \* **data**
- void \* **head**
- void \* **tail** [FIFO\_MAX\_READERS]
- void \* **end**
- int **readers**
- long int **size**
- int **elem\_size**
- int **full**
- long int **overruns**
- pthread\_mutex\_t **mutex**
- pthread\_cond\_t **signal**

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

- src/[cin.h](#)

## Chapter 6

# File Documentation

### 6.1 src/cin.h File Reference

```
#include <stdint.h>
#include <stdio.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netinet/ip.h>
#include <sys/time.h>
#include <pthread.h>
```

#### Classes

- struct [cin\\_ctl\\_config](#)
- struct [fifo](#)
- struct [cin\\_ctl\\_listener](#)
- struct [cin\\_port](#)
- struct [cin\\_ctl](#)
- struct [cin\\_data\\_frame](#)
- struct [cin\\_data\\_stats](#)
- struct [cin\\_data\\_threads](#)
- struct [cin\\_data\\_callbacks](#)
- struct [descramble\\_map\\_t](#)
- struct [cin\\_data](#)
- struct [cin\\_ctl\\_id](#)
- struct [cin\\_ctl\\_pwr\\_val](#)
- struct [cin\\_ctl\\_pwr\\_mon\\_t](#)

#### Macros

- `#define CIN_CTL_IP "192.168.1.207"`
- `#define CIN_CTL_SVR_PORT 49200`
- `#define CIN_CTL_CLI_PORT 50200`
- `#define CIN_CTL_SVR_FRMW_PORT 49202`
- `#define CIN_CTL_CLI_FRMW_PORT 50202`
- `#define CIN_CTL_MAX_READ_TRIES 10`
- `#define CIN_CTL_MAX_WRITE_TRIES 5`
- `#define CIN_CTL_WRITE_SLEEP 2000`

- #define **CIN\_CTL\_POWER\_ENABLE** 0x001F
- #define **CIN\_CTL\_POWER\_DISABLE** 0x0000
- #define **CIN\_CTL\_FP\_POWER\_ENABLE** 0x0020
- #define **CIN\_CTL\_DCM\_LOCKED** 0x0001
- #define **CIN\_CTL\_DCM\_PSDONE** 0x0002
- #define **CIN\_CTL\_DCM\_STATUS0** 0x0004
- #define **CIN\_CTL\_DCM\_STATUS1** 0x0008
- #define **CIN\_CTL\_DCM\_STATUS2** 0x0010
- #define **CIN\_CTL\_DCM\_TX1\_READY** 0x0020
- #define **CIN\_CTL\_DCM\_TX2\_READY** 0x0040
- #define **CIN\_CTL\_DCM\_ATCA\_ALARM** 0x0080
- #define **CIN\_CTL\_TRIG\_INTERNAL** 0x0000
- #define **CIN\_CTL\_TRIG\_EXTERNAL\_1** 0x0001
- #define **CIN\_CTL\_TRIG\_EXTERNAL\_2** 0x0002
- #define **CIN\_CTL\_TRIG\_EXTERNAL\_BOTH** 0x0003
- #define **CIN\_CTL\_FOCUS\_BIT** 0x0002
- #define **CIN\_CTL\_FCLK\_125** 0x0000
- #define **CIN\_CTL\_FCLK\_200** 0x0001
- #define **CIN\_CTL\_FCLK\_250** 0x0002
- #define **CIN\_CTL\_FCLK\_125\_C** 0x0003
- #define **CIN\_CTL\_FCLK\_200\_C** 0x0004
- #define **CIN\_CTL\_FCLK\_250\_C** 0x0005
- #define **CIN\_CTL\_FCLK\_156\_C** 0x0006
- #define **CIN\_CTL\_FPGA\_STS\_CFG** 0x8000
- #define **CIN\_CTL\_FPGA\_STS\_FP\_PWR** 0x0008
- #define **CIN\_CTL\_DCM\_STS\_ATCA** 0x0080
- #define **CIN\_CTL\_DCM\_STS\_LOCKED** 0x0001
- #define **CIN\_CTL\_DCM\_STS\_OVERRIDE** 0x0800
- #define **CIN\_CTL\_MUX1\_VCLK1** 0x0001
- #define **CIN\_CTL\_MUX1\_VCLK2** 0x0002
- #define **CIN\_CTL\_MUX1\_VCLK3** 0x0003
- #define **CIN\_CTL\_MUX1\_ATG** 0x0004
- #define **CIN\_CTL\_MUX1\_VFCLK1** 0x0005
- #define **CIN\_CTL\_MUX1\_VFCLK2** 0x0006
- #define **CIN\_CTL\_MUX1\_VFCLK3** 0x0007
- #define **CIN\_CTL\_MUX1\_HCLK1** 0x0008
- #define **CIN\_CTL\_MUX1\_HCLK2** 0x0009
- #define **CIN\_CTL\_MUX1\_OSW** 0x000A
- #define **CIN\_CTL\_MUX1\_RST** 0x000B
- #define **CIN\_CTL\_MUX1\_CONVERT** 0x000C
- #define **CIN\_CTL\_MUX1\_SHUTTER** 0x000D
- #define **CIN\_CTL\_MUX1\_SWTRIGGER** 0x000E
- #define **CIN\_CTL\_MUX1\_TRIGMON** 0x000F
- #define **CIN\_CTL\_MUX1\_EXPOSE** 0x0000
- #define **CIN\_CTL\_MUX2\_VCLK1** 0x0010
- #define **CIN\_CTL\_MUX2\_VCLK2** 0x0020
- #define **CIN\_CTL\_MUX2\_VCLK3** 0x0030
- #define **CIN\_CTL\_MUX2\_ATG** 0x0040
- #define **CIN\_CTL\_MUX2\_VFCLK1** 0x0050
- #define **CIN\_CTL\_MUX2\_VFCLK2** 0x0060
- #define **CIN\_CTL\_MUX2\_VFCLK3** 0x0070
- #define **CIN\_CTL\_MUX2\_HCLK1** 0x0080
- #define **CIN\_CTL\_MUX2\_HCLK2** 0x0090
- #define **CIN\_CTL\_MUX2\_HCLK3** 0x00A0
- #define **CIN\_CTL\_MUX2\_OSW** 0x00B0



- #define **CIN\_CTL\_MUX2\_RST** 0x00C0
- #define **CIN\_CTL\_MUX2\_CONVERT** 0x00D0
- #define **CIN\_CTL\_MUX2\_SAVE** 0x00E0
- #define **CIN\_CTL\_MUX2\_HWTRIG** 0x00F0
- #define **CIN\_CTL\_MUX2\_EXPOSE** 0x0000
- #define **CIN\_CTL\_FO\_REG1** 0x821D
- #define **CIN\_CTL\_FO\_REG2** 0x821E
- #define **CIN\_CTL\_FO\_REG3** 0x821F
- #define **CIN\_CTL\_FO\_REG4** 0x8001
- #define **CIN\_CTL\_FO\_REG5** 0x8211
- #define **CIN\_CTL\_FO\_REG6** 0x8212
- #define **CIN\_CTL\_FO\_REG7** 0x8213
- #define **CIN\_DATA\_IP** "10.0.5.207"
- #define **CIN\_DATA\_PORT** 49201
- #define **CIN\_DATA\_CTL\_PORT** 49203
- #define **CIN\_DATA\_MAX\_MTU** 9000
- #define **CIN\_DATA\_UDP\_HEADER** 8
- #define **CIN\_DATA\_MAGIC\_PACKET** UINT64\_C(0x0000F4F3F2F1F000)
- #define **CIN\_DATA\_MAGIC\_PACKET\_MASK** UINT64\_C(0x0000FFFFFFFFFFFF00)
- #define **CIN\_DATA\_TAIL\_MAGIC\_PACKET** UINT64\_C(0x010DF0ADDEF2F1F0)
- #define **CIN\_DATA\_TAIL\_MAGIC\_PACKET\_MASK** UINT64\_C(0xFFFFFFFFFFFFFFF0)
- #define **CIN\_DATA\_DROPPED\_PACKET\_VAL** 0x2000
- #define **CIN\_DATA\_DATA\_MASK** 0x1FFF
- #define **CIN\_DATA\_CTRL\_MASK** 0xE000
- #define **CIN\_DATA\_SIGN\_MASK** 0x1000
- #define **CIN\_DATA\_GAIN\_8** 0xC000
- #define **CIN\_DATA\_GAIN\_4** 0x4000
- #define **CIN\_DATA\_PACKET\_LEN** 8184
- #define **CIN\_DATA\_MAX\_PACKETS** 542
- #define **CIN\_DATA\_RCVBUF** 100
- #define **CIN\_DATA\_MAX\_FRAME\_X** 1152
- #define **CIN\_DATA\_MAX\_FRAME\_Y** 2050
- #define **CIN\_DATA\_MAX\_STREAM** 2400000
- #define **CIN\_DATA\_CCD\_COLS** 96
- #define **CIN\_DATA\_CCD\_COLS\_PER\_CHAN** 10
- #define **CIN\_DATA\_PIPELINE\_FLUSH** 1344
- #define **CIN\_DATA\_MODE\_CALLBACK** 0x01
- #define **NUM\_BIAS\_VOLTAGE** 20
- #define **pt\_posH** 0
- #define **pt\_negH** 1
- #define **pt\_posRG** 2
- #define **pt\_negRG** 3
- #define **pt\_posSW** 4
- #define **pt\_negSW** 5
- #define **pt\_posV** 6
- #define **pt\_negV** 7
- #define **pt\_posTG** 8
- #define **pt\_negTG** 9
- #define **pt\_posVF** 10
- #define **pt\_negVF** 11
- #define **pt\_NEDGE** 12
- #define **pt\_OTG** 13
- #define **pt\_VDDR** 14
- #define **pt\_VDD\_OUT** 15
- #define **pt\_BUF\_Base** 16

- `#define pt_BUF_Delta 17`
- `#define pt_Spare1 18`
- `#define pt_Spare2 19`
- `#define DEBUG_PRINT(fmt,...) if(_debug_print_flag) { fprintf(stderr, "%s:%d:%s(): " fmt, __FILE__, __LINE__, __func__, __VA_ARGS__); }`
- `#define DEBUG_COMMENT(fmt) if(_debug_print_flag) { fprintf(stderr, "%s:%d:%s(): " fmt, __FILE__, __LINE__, __func__); }`
- `#define ERROR_COMMENT(fmt) if(_error_print_flag) { fprintf(stderr, "%s:%d:%s(): " fmt, __FILE__, __LINE__, __func__); }`
- `#define ERROR_PRINT(fmt,...) if(_error_print_flag) { fprintf(stderr, "%s:%d:%s(): " fmt, __FILE__, __LINE__, __func__, __VA_ARGS__); }`
- `#define CIN_CONFIG_MAX_STRING 256`
- `#define CIN_CONFIG_MAX_DATA 5000`
- `#define FIFO_MAX_READERS 10`

## Typedefs

- `typedef struct cin_ctl_config cin_ctl_config_t`
- `typedef struct cin_ctl_listener cin_ctl_listener_t`
- `typedef struct cin_port cin_port_t`
- `typedef struct cin_ctl cin_ctl_t`
- `typedef struct cin_data_frame cin_data_frame_t`
- `typedef struct cin_data_stats cin_data_stats_t`
- `typedef struct cin_data_threads cin_data_threads_t`
- `typedef struct cin_data_callbacks cin_data_callbacks_t`
- `typedef struct cin_data cin_data_t`
- `typedef void(* cin_data_callback)(cin_data_frame_t *)`
- `typedef struct cin_ctl_id cin_ctl_id_t`
- `typedef struct cin_ctl_pwr_val cin_ctl_pwr_val_t`

## Functions

- `void cin_set_debug_print(int debug)`
- `void cin_set_error_print(int error)`
- `void cin_report(FILE *fp, int details)`
- `int cin_ctl_init(cin_ctl_t *cin, const char *ipaddr, uint16_t oport, uint16_t iport, uint16_t soport, uint16_t siport)`
- `int cin_ctl_destroy(cin_ctl_t *cin)`
- `int cin_ctl_read(cin_ctl_t *cin, uint16_t reg, uint16_t *val)`
- `int cin_ctl_write(cin_ctl_t *cin, uint16_t reg, uint16_t val, int wait)`
- `int cin_ctl_stream_write(cin_ctl_t *cin, char *val, int size)`
- `int cin_ctl_write_with_readback(cin_ctl_t *cin, uint16_t reg, uint16_t val)`
- `int cin_ctl_pwr(cin_ctl_t *cin, int pwr)`
- `int cin_ctl_fp_pwr(cin_ctl_t *cin, int pwr)`
- `int cin_ctl_fo_test_pattern(cin_ctl_t *cin, int on_off)`
- `int cin_ctl_load_config(cin_ctl_t *cin, char *filename)`
- `int cin_ctl_load_firmware(cin_ctl_t *cin, char *filename)`
- `int cin_ctl_set_fclk(cin_ctl_t *cin, int clkfreq)`
- `int cin_ctl_get_fclk(cin_ctl_t *cin, int *clkfreq)`
- `int cin_ctl_freeze_dco(cin_ctl_t *cin, int freeze)`
- `int cin_ctl_get_cfg_fpga_status(cin_ctl_t *cin, uint16_t *_val)`
- `int cin_ctl_get_id(cin_ctl_t *cin, cin_ctl_id_t *_val)`
- `void cin_ctl_display_id(FILE *out, cin_ctl_id_t val)`
- `void cin_ctl_display_fpga_status(FILE *out, uint16_t val)`

- int **cin\_ctl\_get\_dcm\_status** ([cin\\_ctl\\_t](#) \*cin, uint16\_t \*\_val)
- void **cin\_ctl\_display\_dcm\_status** (FILE \*out, uint16\_t \*\_val)
- double **cin\_ctl\_current\_calc** (uint16\_t val)
- int **cin\_ctl\_get\_power\_status** ([cin\\_ctl\\_t](#) \*cin, int full, int \*pwr, [cin\\_ctl\\_pwr\\_mon\\_t](#) \*values)
- void **cin\_ctl\_display\_pwr** (FILE \*out, [cin\\_ctl\\_pwr\\_mon\\_t](#) \*values)
- void **cin\_ctl\_display\_pwr\_line** (FILE \*out, const char \*msg, [cin\\_ctl\\_pwr\\_val\\_t](#) val)
- int **cin\_ctl\_calc\_vi\_status** ([cin\\_ctl\\_t](#) \*cin, uint16\_t vreg, uint16\_t ireg, double vfact, [cin\\_ctl\\_pwr\\_val\\_t](#) \*vi)
- int **cin\_ctl\_get\_camera\_pwr** ([cin\\_ctl\\_t](#) \*cin, int \*val)
- int **cin\_ctl\_set\_camera\_pwr** ([cin\\_ctl\\_t](#) \*cin, int val)
- int **cin\_ctl\_set\_bias** ([cin\\_ctl\\_t](#) \*cin, int val)
- int **cin\_ctl\_get\_bias** ([cin\\_ctl\\_t](#) \*cin, int \*val)
- int **cin\_ctl\_set\_clocks** ([cin\\_ctl\\_t](#) \*cin, int val)
- int **cin\_ctl\_get\_clocks** ([cin\\_ctl\\_t](#) \*cin, int \*val)
- int **cin\_ctl\_set\_trigger** ([cin\\_ctl\\_t](#) \*cin, int val)
- int **cin\_ctl\_get\_trigger** ([cin\\_ctl\\_t](#) \*cin, int \*val)
- int **cin\_ctl\_set\_focus** ([cin\\_ctl\\_t](#) \*cin, int val)
- int **cin\_ctl\_get\_focus** ([cin\\_ctl\\_t](#) \*cin, int \*val)
- int **cin\_ctl\_get\_triggering** ([cin\\_ctl\\_t](#) \*cin, int \*trigger)
- int **cin\_ctl\_int\_trigger\_start** ([cin\\_ctl\\_t](#) \*cin, int nimages)
- int **cin\_ctl\_int\_trigger\_stop** ([cin\\_ctl\\_t](#) \*cin)
- int **cin\_ctl\_ext\_trigger\_start** ([cin\\_ctl\\_t](#) \*cin, int trigger\_mode)
- int **cin\_ctl\_ext\_trigger\_stop** ([cin\\_ctl\\_t](#) \*cin)
- int **cin\_ctl\_set\_exposure\_time** ([cin\\_ctl\\_t](#) \*cin, float e\_time)
- int **cin\_ctl\_set\_trigger\_delay** ([cin\\_ctl\\_t](#) \*cin, float t\_time)
- int **cin\_ctl\_set\_cycle\_time** ([cin\\_ctl\\_t](#) \*cin, float ftime)
- int **cin\_ctl\_frame\_count\_reset** ([cin\\_ctl\\_t](#) \*cin)
- int **cin\_ctl\_set\_mux** ([cin\\_ctl\\_t](#) \*cin, int setting)
- int **cin\_ctl\_get\_mux** ([cin\\_ctl\\_t](#) \*cin, int \*setting)
- int **cin\_ctl\_set\_fcric\_gain** ([cin\\_ctl\\_t](#) \*cin, int gain)
- int **cin\_ctl\_set\_fabric\_address** ([cin\\_ctl\\_t](#) \*cin, char \*ip)
- int **cin\_ctl\_reg\_dump** ([cin\\_ctl\\_t](#) \*cin, FILE \*fp)
- int **cin\_ctl\_get\_bias\_voltages** ([cin\\_ctl\\_t](#) \*cin, float \*voltage)
- int **cin\_ctl\_set\_bias\_voltages** ([cin\\_ctl\\_t](#) \*cin, float \*voltage)
- int **cin\_ctl\_set\_fcric\_clamp** ([cin\\_ctl\\_t](#) \*cin, int clamp)
- int **cin\_config\_read\_file** ([cin\\_ctl\\_t](#) \*cin, const char \*file)
- int **cin\_data\_init** ([cin\\_data\\_t](#) \*cin, int mode, int packet\_buffer\_len, int frame\_buffer\_len, char \*ipaddr, uint16\_t port, char \*cin\_ipaddr, uint16\_t [cin\\_port](#), int rcvbuf, cin\_data\_callback push\_callback, cin\_data\_callback pop\_callback, void \*usr\_ptr)
- void **cin\_data\_wait\_for\_threads** ([cin\\_data\\_t](#) \*cin)
- void **cin\_data\_stop\_threads** ([cin\\_data\\_t](#) \*cin)
- void **cin\_data\_framestore\_trigger** ([cin\\_data\\_t](#) \*cin, int count)
- void **cin\_data\_framestore\_skip** ([cin\\_data\\_t](#) \*cin, int count)
- int **cin\_data\_get\_framestore\_counter** ([cin\\_data\\_t](#) \*cin)
- void **cin\_data\_framestore\_disable** ([cin\\_data\\_t](#) \*cin)
- void **cin\_data\_framestore\_trigger\_enable** ([cin\\_data\\_t](#) \*cin)
- struct [cin\\_data\\_frame](#) \* **cin\_data\_get\_next\_frame** ([cin\\_data\\_t](#) \*cin)
- void **cin\_data\_release\_frame** ([cin\\_data\\_t](#) \*cin, int free\_mem)
- struct [cin\\_data\\_frame](#) \* **cin\_data\_get\_buffered\_frame** (void)
- void **cin\_data\_release\_buffered\_frame** (void)
- void **cin\_data\_compute\_stats** ([cin\\_data\\_t](#) \*cin, [cin\\_data\\_stats\\_t](#) \*stats)
- void **cin\_data\_show\_stats** (FILE \*fp, [cin\\_data\\_stats\\_t](#) stats)
- void **cin\_data\_reset\_stats** ([cin\\_data\\_t](#) \*cin)
- int **cin\_data\_set\_descramble\_params** ([cin\\_data\\_t](#) \*cin, int rows, int overscan)
- void **cin\_data\_get\_descramble\_params** ([cin\\_data\\_t](#) \*cin, int \*rows, int \*overscan, int \*xsize, int \*ysize)

## Variables

- const char \* **cin\_build\_git\_time**
- const char \* **cin\_build\_git\_sha**
- const char \* **cin\_build\_version**
- int **\_debug\_print\_flag**
- int **\_error\_print\_flag**

### 6.1.1 Detailed Description

#### Author

Stuart B. Wilkins [swilkins@bnl.gov](mailto:swilkins@bnl.gov)

### 6.1.2 LICENSE

Copyright (c) 2014, Stuart B. Wilkins All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. 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 OWNER 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.

The views and conclusions contained in the software and documentation are those of the authors and should not be interpreted as representing official policies, either expressed or implied, of the FreeBSD Project.

### 6.1.3 DESCRIPTION

header file for CIN communications

### 6.1.4 Function Documentation

#### 6.1.4.1 void cin\_data\_framestore\_disable ( cin\_data\_t \* cin )

Disable the framestore modes

This function disables the framestore modes (software trigger and skip). If the camera is hardware triggering then the images will start to be processed.

## Parameters

<i>cin</i>	Handle to the cin library
------------	---------------------------

## See Also

[cin\\_data\\_framestore\\_trigger](#) [cin\\_data\\_framestore\\_skip](#)

## 6.1.4.2 void cin\_data\_framestore\_skip ( cin\_data\_t \* cin, int count )

Enable framestore skip mode

Enable the framestore skip mode. This function should be called before hardware triggering the camera. This causes the data processing to skip

## Parameters

<i>count</i>	frames from the first images to be read. This is usually done to stop the first few frames from being over exposed.
<i>cin</i>	handle to the <a href="#">cin_data</a> library

## See Also

[cin\\_data\\_set\\_framestore\\_mode](#), [cin\\_data\\_set\\_framestore\\_counter](#)

## 6.1.4.3 void cin\_data\_framestore\_trigger ( cin\_data\_t \* cin, int count )

Send a framestore (software) trigger

Send a software trigger to the CIN by timestamping the request time and allow images to be processed when recieved after this time. The function is enabled by setting the framestore mode to CIN\_DATA\_FRAMESTORE\_TRIGGER. The count option sets the number of frames to trigger. A value of -1 indicated that the trigger should not count images but run indefinitely after the trigger has occurred.

## Parameters

<i>cin</i>	handle to the <a href="#">cin_data</a> library
<i>count</i>	[in] number of frames to trigger

## See Also

[cin\\_data\\_set\\_framestore\\_disable](#)

## 6.1.4.4 void cin\_data\_framestore\_trigger\_enable ( cin\_data\_t \* cin )

Enable the framestore trigger mode

This function enables the framestore trigger mode. It cases the images to not be processed pending a call to the function to (software) trigger the camera.

## Parameters

<i>cin</i>	Handle to the cin library
------------	---------------------------

## See Also

[cin\\_data\\_framestore\\_trigger](#)

#### 6.1.4.5 int cin\_data\_get\_framestore\_counter ( cin\_data\_t \* cin )

Get the value of the framestore counter

Return the number of frames in the framestore counter. In trigger mode, this returns the number of frames to go. In skip mode, this returns the number of frames that have to be skipped.

##### Parameters

<i>cin</i>	handle to the <a href="#">cin_data</a> library
------------	--

##### Returns

Number of frames to go in trigger

##### See Also

[cin\\_data\\_framestore\\_trigger](#)

#### 6.1.4.6 int cin\_data\_init ( cin\_data\_t \* cin, int mode, int packet\_buffer\_len, int frame\_buffer\_len, char \* ipaddr, uint16\_t port, char \* cin\_ipaddr, uint16\_t cin\_port, int rcvbuf, cin\_data\_callback push\_callback, cin\_data\_callback pop\_callback, void \* usr\_ptr )

Initialize the cin data library

Initialize the data handling routines and start the threads for listening. mode should be set for the desired output. The packet\_buffer\_len is the length of the packet FIFO in number of packets. The frame\_buffer\_len is the number of data frames to buffer.

##### Parameters

<i>cin</i>	Handle to cin data library
------------	----------------------------

# Index

CIN Data Framestore Functions, [11](#)

Cin Control Routines, [7](#)

    cin\_ctl\_destroy, [8](#)

    cin\_ctl\_init, [8](#)

    cin\_ctl\_read, [8](#)

    cin\_ctl\_stream\_write, [10](#)

    cin\_ctl\_write, [10](#)

    cin\_ctl\_write\_with\_readback, [10](#)

cin.h

    cin\_data\_framestore\_disable, [24](#)

    cin\_data\_framestore\_skip, [25](#)

    cin\_data\_framestore\_trigger, [25](#)

    cin\_data\_framestore\_trigger\_enable, [25](#)

    cin\_data\_get\_framestore\_counter, [25](#)

    cin\_data\_init, [26](#)

cin\_ctl, [13](#)

cin\_ctl\_config, [13](#)

cin\_ctl\_destroy

    Cin Control Routines, [8](#)

cin\_ctl\_id, [14](#)

cin\_ctl\_init

    Cin Control Routines, [8](#)

cin\_ctl\_listener, [14](#)

cin\_ctl\_pwr\_mon\_t, [14](#)

cin\_ctl\_pwr\_val, [14](#)

cin\_ctl\_read

    Cin Control Routines, [8](#)

cin\_ctl\_stream\_write

    Cin Control Routines, [10](#)

cin\_ctl\_write

    Cin Control Routines, [10](#)

cin\_ctl\_write\_with\_readback

    Cin Control Routines, [10](#)

cin\_data, [15](#)

cin\_data\_callbacks, [15](#)

cin\_data\_frame, [16](#)

cin\_data\_framestore\_disable

    cin.h, [24](#)

cin\_data\_framestore\_skip

    cin.h, [25](#)

cin\_data\_framestore\_trigger

    cin.h, [25](#)

cin\_data\_framestore\_trigger\_enable

    cin.h, [25](#)

cin\_data\_get\_framestore\_counter

    cin.h, [25](#)

cin\_data\_init

    cin.h, [26](#)

cin\_data\_packet, [16](#)

cin\_data\_proc, [16](#)

cin\_data\_stats, [16](#)

cin\_data\_threads, [17](#)

cin\_map\_t, [17](#)

cin\_port, [17](#)

descramble\_map\_t, [18](#)

fifo, [18](#)

src/cin.h, [19](#)