ADC-ZESOI Server Reference Manual

0.2.3

Generated by Doxygen 1.2.7

Wed Feb 6 16:45:11 2002

Contents

1 ADC-ZESOI Server File Index

1.1 ADC-ZESOI Server File List

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

2 ADC-ZESOI Server File Documentation

2.1 server.c File Reference

ADC-ZESOI protocol server.

```
#include <server.h>
```

Defines

- #define **TEST_MODE**

Functions

- int **main** (int argc, char *argv[])
- char* **readcommand** (int fd)

  Read command form file.

- **command_t parsecommand** (const char *input, **command_t** status)

  Parse command.

- **command_t executecommand** (**command_t** c, **command_t** status, int fd)

  Executes command.

- void **lostconnection** (int sig)

  SIGPIPE handler.

- int **acquireandsendsamples** (int fd, int no_of_samples, int channel, char resolution)

  Acquires and sends samples.

- void **stopstream** (int sig)

  SIGTERM handler.

- void **cleanallchildren** (int sig)

  SIGCHLD handler.

Variables

- const char* **help**

2.1.1 Detailed Description

ADC-ZESOI protocol server.

Author(s):
Tomislav Petković , Darko Vasić

Date:
2001-12-09

Definition in file **server.c**.

2.1.2 Define Documentation

2.1.2.1 #define TEST_MODE

Value:

Definition at line 19 of file server.c.

2.1.3 Function Documentation

2.1.3.1 int acquireandsendsamples (int *fd*, int *no_of_samples*, int *channel*, char *resolution*)

Acquires and sends samples.

Acquire and send samples. TODO: This function should be replaced with a call to new program that continously stores data samples and proviedes them at request. That project layer is not yet completly implemented (Why? Ask project coordinator Zvonko Kostanjcar at `zkostanj@diana.zesoi.fer.hr`).

Parameters:

fd File descriptor.

no_of_samples Number of samples to send.

channel Requested channel number.

resolution Requested resolution.

Returns:

Returns ACQUIRE_SUCCESS if successful.

Definition at line 1436 of file server.c.

2.1.3.2 void cleanallchildren (int *sig*)

SIGCHLD handler.

Cleans all child processes.

Parameters:

sig Signal number.

Definition at line 1564 of file server.c.

2.1.3.3 command_t executecommand (command_t *c*, command_t *status*, int *fd*)

Executes command.

Executes command and prints useful messages about execution.

Parameters:

c Command to execute.

status Server status.

fd File descriptor.

Returns:

Returns new server status.

Definition at line 843 of file server.c.

2.1.3.4 void lostconnection (int *sig*)

SIGPIPE handler.

Handles SIGPIPE signal (broken pipe). This signal is received when client terminates connection. Terminates child proces and cleans up everything.

Parameters:

sig Signal number.

Definition at line 1406 of file server.c.

2.1.3.5 command_t parsecommand (const char * *input*, command_t *status*)

Parse command.

Input buffer is parsed for ADC-ZESOI protocol commands.

Parameters:

input Input buffer.

status Status of the server.

Returns:

Returns a structure which contains command number and additional parameters. It's a copy of command_t status with command_no and error_code changed appropriately. Other parameters are changed if specified by parsed command.

Definition at line 503 of file server.c.

2.1.3.6 char * readcommand (int *fd*)

Read command form file.

Reads an aribtrary length command form file specified by file descriptor.

Parameters:

fd A file descriptor.

Returns:

Returns a pointer to a uninterpreted command, or NULL pointer if unsuccessful.

Definition at line 451 of file server.c.

2.1.3.7 void stopstream (int *sig*)

SIGTERM handler.

Handles SIGTERM signal for child process when child process is sending data stream.

Parameters:

sig Signal number.

Definition at line 1531 of file server.c.

2.1.4 Variable Documentation

2.1.4.1 const char * help

Initial value:

```
"ADC-ZESOI server v" VERSION ".\n"
"Usage:\n"
" server [options]\n"
"Options are:\n"
" -h --help      Prints this help message.\n"

" -t --test      Test mode - server creates random data (default).\n"
"    --notest    Test mode disabled. Can't be used together with -t.\n"



" -p <port>, --port=<port>   Sets port to <port>. If ommited default port\n"
"                for ADC-ZESOI protocol (port 7777) is used.\n"
" -a --auth      Requires authentification. Default is no\n"
"                authentfication required (option -n).\n"
" -n --noauth    Disable authentification (default). Can't be\n"
"                used together with -a.\n"
"    --nullok    Allow access for passwordless accounts.\n"
"    --maxretries=<retries>   Sets maximum number of login attempts to\n"
"                given number. Default value is 3 login attempts.\n"
" -v --version   Show version number.\n"
```

Definition at line 26 of file server.c.

2.2 server.h File Reference

Header file for **server.c**.

```
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <string.h>
#include <limits.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <signal.h>
#include <unistd.h>
#include <ctype.h>
#include <crypt.h>
```

Data Structures

- struct **_command_t**

Defines

- #define **VERSION** "0.2.3"
- #define **TRUE** (1==1)
- #define **FALSE** (1!=1)
- #define **PORT** 7777
- #define **BACKLOG** 10
- #define **ROOT_UID** 0
- #define **MAX_LOGIN** 3
- #define **LOGIN_SLEEP_TIME** 15
- #define **DEFAULT_CHANNEL** 1
- #define **DEFAULT_RESOLUTION** 'M'
- #define **DEFAULT_NO_SAMPLES** 25
- #define **MIN_CHANNEL** 1
- #define **MAX_CHANNEL** 8
- #define **DATA_STREAM** -1
- #define **BUFFER_SIZE** 256
- #define **MAX_LENGTH** 256
- #define **DELIMITERS** " \t\n\x0A\x0D"
- #define **NOAUTH** 0
- #define **AUTH** 1
- #define **AUTHORISED** (1+2)
- #define **NOTAUTHORISED** (1+4)
- #define **AUTHNOTPROCESSED** 8
- #define **MODENOTPROCESSED** (1+2)
- #define **MODEPROCESSED** (1+4)
- #define **ACQUIRE_SUCCESS** 0
- #define **ACQUIRE_FAILURE** 1
- #define **HIGH_RESOLUTION_STEP** 1
- #define **MEDIUM_RESOLUTION_STEP** 2
- #define **LOW_RESOLUTION_STEP** 5
- #define **STREAM_CHUNK** 128
- #define **GET_STREAM_ARGUMENT** "STREAM"
- #define **RESOLUTION_HIGH** "HIGH"
- #define **RESOLUTION_H** "H"
- #define **RESOLUTION_MEDIUM** "MEDIUM"
- #define **RESOLUTION_M** "M"
- #define **RESOLUTION_LOW** "LOW"
- #define **RESOLUTION_L** "L"
- #define **RESOLUTION_HC** 'H'
- #define **RESOLUTION_MC** 'M'
- #define **RESOLUTION_LC** 'L'

Typedefs

- typedef enum _command_no_t **command_no_t**

  Command number definitions.

- typedef enum _error_code_t **error_code_t**

  Error codes definitions.

- typedef enum _ok_message_no_t **_ok_message_no_t**
- typedef struct _command_t **command_t**

  Command structure definition.

Enumerations

- enum **_command_no_t** { **CN_ERROR** = -2, **CN_NO_COMMAND** = -1, **CN_USER** = 0, **CN_PASS**, **CN_SET**, **CN_GET**, **CN_RESOLUTION**, **CN_START**, **CN_STOP**, **CN_HELP**, **CN_BYE**, **CN_EXIT**, **CN_QUIT**, **MAX_CN_NO** }
- enum **_error_code_t** { **EC_PARSE_FAILURE** = -2, **EC_NO_ERROR** = -1, **EC_UNKNOWN_COMMAND** = 0, **EC_USER_ERROR**, **EC_USER_UNKNOWN**, **EC_USER_OVERRUN**, **EC_USER_NO_NAME**, **EC_USER_NO_AUTHORISATION_REQUIRED**, **EC_PASS_ERROR**, **EC_PASS_INCORRECT**, **EC_PASS_EXPIRED**, **EC_PASS_NO_USER_NAME**, **EC_PASS_ALREADY_AUTHORISED**, **EC_PASS_OVERRUN**, **EC_PASS_NO_PASS**, **EC_PASS_NO_AUTHORISATION_REQUIRED**, **EC_PASS_DISCONNECTED**, **EC_PASS_PASSWORDLESS**, **EC_PASS_DENIED**, **EC_PASS_LOCKED**, **EC_SET_ERROR**, **EC_SET_INVALID**, **EC_SET_UNAVAILABE**, **EC_SET_UNAUTHORISED**, **EC_SET_NO_CHANNEL**, **EC_GET_ERROR**, **EC_GET_INVALID_NO_OF_SAMPLES**, **EC_GET_UNAUTHORISED**, **EC_GET_NO_ARGUMENT**, **EC_RESOLUTION_ERROR**, **EC_RESOLUTION_INVALID**, **EC_RESOLUTION_UNAUTHORISED**, **EC_RESOLUTION_NO_ARGUMENT**, **EC_START_ERROR**, **EC_START_UNAUTHORISED**, **EC_STOP_ERROR**, **EC_STOP_NO_STREAM**, **EC_STOP_UNAUTHORISED**, **EC_HELP_INVALID_COMMAND** }
- enum **_ok_message_no_t** { **MN_USER** = 0, **MN_PASS**, **MN_SET**, **MN_GET_STREAM**, **MN_GET_SAMPLES**, **MN_RESOLUTION**, **MN_START_STREAM**, **MN_START_SAMPLES**, **MN_START_TEST_SAMPLES**, **MN_START_TEST_STREAM**, **MN_START_RESOLUTION**, **MN_STOP**, **MN_TRANSFER_COMPLETED**, **MN_WELCOME**, **MN_GOODBYE** }

Functions

- char* **readcommand** (int)
  Read command form file.
- **command_t parsecommand** (const char *, **command_t**)
  Parse command.
- **command_t executecommand** (**command_t**, **command_t**, int)
  Executes command.
- int **acquireandsendsamples** (int, int, int, char)
  Acquires and sends samples.
- void **lostconnection** (int)
  SIGPIPE handler.
- void **stopstream** (int)
  SIGTERM handler.
- void **cleanallchildren** (int)
  SIGCHLD handler.

Variables

- int **global_fd**
- unsigned int **global_max_login_no**
- short int **global_null_ok**
- pid_t **global_pid**
- pid_t **global_stream_pid**
- short int **global_test_mode** = FALSE

2.2.1 Detailed Description

Header file for **server.c**.

Author(s):
Tomislav Petković , Darko Vasić

Date:
2001-12-09

Definition in file **server.h**.

2.2.2 Typedef Documentation

2.2.2.1 typedef enum _command_no_t command_no_t

Command number definitions.

Defined as enum. All command number but CN_ERROR and CN_NO_-COMMAND are indices into a static const char *command[] and can be used for retriving commands. They can also be used as indices into *help_message[] to retrive help for specified command.

2.2.2.2 typedef struct _command_t command_t

Command structure definition.

Contanis command number, error code and all the necesary informations needed for execution of commands. Negative number of samples inicates data stream. authorised, stream and exit are used as flags (values are TRUE and FALSE).

2.2.2.3 typedef enum _error_code_t error_code_t

Error codes definitions.

Defined as enum. All error codes but EC_NO_ERROR are indices into a static const char *error_message[] and can be used for retriving messages, ie. error_-message[EC_GET_ERROR] is a message for unsuccessful GET command.

2.2.3 Function Documentation

2.2.3.1 int acquireandsendsamples (int *fd*, int *no_of_samples*, int *channel*, char *resolution*)

Acquires and sends samples.

Acquire and send samples. TODO: This function should be replaced with a call to new program that continously stores data samples and proviedes them at request. That project layer is not yet completly implemented (Why? Ask project coordinator Zvonko Kostanjcar at `zkostanj@diana.zesoi.fer.hr`).

Parameters:

fd File descriptor.

no_of_samples Number of samples to send.

channel Requested channel number.

resolution Requested resolution.

Returns:

Returns ACQUIRE_SUCCESS if successful.

Definition at line 1436 of file server.c.

Referenced by executecommand().

2.2.3.2 void cleanallchildren (int *sig*)

SIGCHLD handler.

Cleans all child processes.

Parameters:
sig Signal number.

Definition at line 1564 of file server.c.

2.2.3.3 command_t executecommand (command_t *c*, command_t *status*, int *fd*)

Executes command.

Executes command and prints useful messages about execution.

Parameters:
c Command to execute.

status Server status.

fd File descriptor.

Returns:
Returns new server status.

Definition at line 843 of file server.c.

2.2.3.4 void lostconnection (int *sig*)

SIGPIPE handler.

Handles SIGPIPE signal (broken pipe). This signal is received when client terminates connection. Terminates child proces and cleans up everything.

Parameters:
sig Signal number.

Definition at line 1406 of file server.c.

2.2.3.5 command_t parsecommand (const char * *input*, command_t *status*)

Parse command.

Input buffer is parsed for ADC-ZESOI protocol commands.

Parameters:
input Input buffer.

status Status of the server.

Returns:
Returns a structure which contains command number and additional parameters. It's a copy of command_t status with command_no and error_code changed appropriately. Other parameters are changed if specified by parsed command.

Definition at line 503 of file server.c.

2.2.3.6 char* readcommand (int *fd*)

Read command form file.

Reads an aribtrary length command form file specified by file descriptor.

Parameters:
fd A file descriptor.

Returns:
Returns a pointer to a uninterpreted command, or NULL pointer if unsuccessful.

Definition at line 451 of file server.c.

2.2.3.7 void stopstream (int *sig*)

SIGTERM handler.

Handles SIGTERM signal for child process when child process is sending data stream.

Parameters:
sig Signal number.

Definition at line 1531 of file server.c.