Register daemon

Generated by Doxygen 1.8.6

Sun Dec 7 2014 19:04:44

Contents

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

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

Chapter 3

Class Documentation

3.1 Cqlex Class Reference

Query lexer class.

Public Types

- enum **lex_type** {
  READ, **WRITE**, **DESC**, **ATTR**,
  NUMBER, **LIST**, **SEMICOLON**, **END**,
  ERR, **READERR** }

Public Member Functions

- Cqlex (int fd, int timeout=0)

  Constructor.
- ∼Cqlex ()

  Destructor.
- lex_type get_lex ()

  Automat to recognition individually lexems from input characters.
- int get_lex_num ()

  Function to get number from lexem if lexem type is NUMBER.
- void get_c ()

  Function to get next character from file descriptor Also use to initialise lexer.

3.1.1 Detailed Description

Query lexer class.

Lexer for input clien queryes.

3.1.2 Constructor & Destructor Documentation

3.1.2.1 Cqlex::Cqlex (int *fd*, int *timeout* = 0) `[inline]`

Constructor.

Parameters

in	*fd*	Client file descriptor
in	*timeout*	Set timeout during reading, 0 means without timeout

3.1.3 Member Function Documentation

3.1.3.1 lex_type Cqlex::get_lex ( ) [inline]

Automat to recognition individually lexems from input characters.

Return values

lex_type	Lexem.

3.1.3.2 int Cqlex::get_lex_num ( ) [inline]

Function to get number from lexem if lexem type is NUMBER.

Return values

int	number.

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

- src/query_lex.cpp

3.2 Cregarray Class Reference

Public Member Functions

- Cregarray ()

  Constructor.
- ∼Cregarray ()

  Destructor.
- int add (const int reg, const char ∗desc, const int a, const int b, const int c, const int d, const int flag)

  Add register method.
- int write (int reg, int value)

  Write value to the register.
- int read (int reg)

  Read value from the register.
- const char ∗ desc (int reg)

  Read register description.
- int a (int reg)

  Read register 'a' parameter.
- int b (int reg)

  Read register 'b' parameter.
- int c (int reg)

  Read register 'c' parameter.
- int d (int reg)

  Read register 'd' parameter.
- int flag (int reg)

  Read register flag parameter.
- char ∗ list ()

  Read list of available registers.

3.2.1 Member Function Documentation

3.2.1.1 int Cregarray::a (int *reg*)

Read register 'a' parameter.

Parameters

in	*reg*	Register number

Return values

int	Register parameter value.

3.2.1.2 int Cregarray::add (const int *reg,* const char ∗ *desc,* const int *a,* const int *b,* const int *c,* const int *d,* const int *flag*)

Add register method.

Parameters

in	*reg*	Register number
in	*desc*	Register description
in	*a*	Parameter a
in	*b*	Parameter b
in	*c*	Parameter c
in	*d*	Parameter d
in	*flag*	Register RO|RW flag

Return values

0	Successfully.
1	With error, duplicit name.

3.2.1.3 int Cregarray::b (int *reg*)

Read register 'b' parameter.

Parameters

in	*reg*	Register number

Return values

int	Register parameter value.

3.2.1.4 int Cregarray::c (int *reg*)

Read register 'c' parameter.

Parameters

in	*reg*	Register number

Return values

int	Register parameter value.

3.2.1.5 int Cregarray::d (int *reg*)

Read register 'd' parameter.

Parameters

in	*reg*	Register number

Return values

int	Register parameter value.

3.2.1.6 const char ∗ Cregarray::desc (int *reg*)

Read register description.

Parameters

in	*reg*	Register number

Return values

char∗	Register description.

3.2.1.7 int Cregarray::flag (int *reg*)

Read register flag parameter.

Parameters

in	*reg*	Register number

Return values

0	Register is RW
1	Register is RO

3.2.1.8 char ∗ Cregarray::list ( )

Read list of available registers.

return register number separated with semicolon

Return values

char∗	register list

3.2.1.9 int Cregarray::read (int *reg*)

Read value from the register.

Parameters

`in`	*reg*	Register number

Return values

0	Error.
int	Register value.

3.2.1.10 int Cregarray::write (int *reg*, int *value*)

Write value to the register.

Parameters

`in`	*reg*	Register number
`in`	*value*	Value

Return values

1	Successfully.
0	Error.

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

- src/regarray.hpp
- src/regarray.cpp

3.3 lex_symbol Struct Reference

Public Attributes

- symbol_type **type**
- char ∗ **ident**
- int **number**
- double **decimal**

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

- src/settings.cpp

3.4 Tconfig Struct Reference

Public Attributes

- int **m_daemon**
- char ∗ **m_socket_name**
- char ∗ **m_pid_file**
- char ∗ **m_device**
- int **m_bound**
- int **m_parity**
- int **m_stopbit**

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

- src/settings.hpp
- src/settings.cpp

Chapter 4

File Documentation

4.1 src/database.cpp File Reference

database add and query functions

```
#include "database.hpp"
#include "regarray.hpp"
#include "query_lex.cpp"
#include "logger.hpp"
#include <stdio.h>
#include <unistd.h>
#include <string.h>
```

Functions

- void lex_compare (Cqlex &lex, Cqlex::lex_type type)

  Lexer compare function.
- int query (Cqlex &lex, int fd, int rw)

  Query parser.
- int reg_add (const int reg, const char ∗desc, const int a, const int b, const int c, const int d, const int flag)

  Interface to add register to database.
- void init_database ()

  Initialize database.
- void delete_database ()

  Free database.

Variables

- Cregarray ∗ **g_database**

4.1.1 Detailed Description

database add and query functions Bohdan Vico (vicobohd@fit.cvut.cz)

Date

November, 2014

4.1.2 Function Documentation

4.1.2.1 void delete_database ( )

Free database.

Function free database

4.1.2.2 void init_database ( )

Initialize database.

Function alloc database

4.1.2.3 void lex_compare (Cqlex & *lex*, Cqlex::lex_type *type*)

Lexer compare function.

This function compare two lexems and throw exception when they different

Parameters

in	*lex*	input lexer
in	*lex_type*	expect lex type

4.1.2.4 int query (Cqlex & *lex*, int *fd*, int *rw*)

Query parser.

This function parse query and send answer

Parameters

in	*lex*	Input lexer
in	*fd*	Client file descriptor
in	*rw*	Accept change RO row in database

Return values

0	Successfully.
1	With error.

4.1.2.5 int reg_add (const int *reg*, const char ∗ *desc*, const int *a*, const int *b*, const int *c*, const int *d*, const int *flag*)

Interface to add register to database.

Parameters

in	*reg*	register number
in	*desc*	register description
in	*a*	parameter a
in	*b*	parameter b
in	*c*	parameter c
in	*d*	parameter d

`in`	*flag*	RW/RO flag

Return values

0	Successfully.
1	With error, duplicit name.

4.2 src/database.hpp File Reference

database add and query functions

`#include "query_lex.cpp"`

Functions

- void init_database ()

  Initialize database.
- void delete_database ()

  Free database.
- int query (Cqlex &lex, int fd, int rw=0)

  Query parser.
- int reg_add (const int reg, const char ∗desc, const int a, const int b, const int c, const int d, const int flag)

  Interface to add register to database.

4.2.1 Detailed Description

database add and query functions Bohdan Vico (`vicobohd@fit.cvut.cz`)

Date

November, 2014

4.2.2 Function Documentation

4.2.2.1 void delete_database ( )

Free database.

Function free database

4.2.2.2 void init_database ( )

Initialize database.

Function alloc database

4.2.2.3 int query (Cqlex & *lex,* int *fd,* int *rw*)

Query parser.

This function parse query and send answer

Parameters

in	*lex*	Input lexer
in	*fd*	Client file descriptor
in	*rw*	Accept change RO row in database

Return values

0	Successfully.
1	With error.

4.2.2.4 int reg_add (const int *reg*, const char ∗ *desc*, const int *a*, const int *b*, const int *c*, const int *d*, const int *flag*)

Interface to add register to database.

Parameters

in	*reg*	register number
in	*desc*	register description
in	*a*	parameter a
in	*b*	parameter b
in	*c*	parameter c
in	*d*	parameter d
in	*flag*	RW/RO flag

Return values

0	Successfully.
1	With error, duplicit name.

4.3 src/listeners.cpp File Reference

function to create listeners

```
#include "listeners.hpp"
#include <sys/un.h>
#include <sys/socket.h>
#include <fcntl.h>
#include <errno.h>
#include <termios.h>
#include <stdio.h>
#include <unistd.h>
#include "database.hpp"
#include "query_lex.cpp"
```

Functions

- int unix_socket (const Tconfig ∗config)

  Function to create local unix socket.
- int serial_link (const Tconfig ∗config)

  Function to alloc serial link.
- void socket_listener (int fd)

  Listener function to listen on local unix socket.
- void serial_listener (int fd)

Listener function to listen on serial link.

- void ∗ socket_pthread (void ∗fd)

  Crete unix socket listener thread for parallel processing.

- void ∗ **serial_pthread** (void ∗fd)

4.3.1 Detailed Description

function to create listeners Bohdan Vico (`vicobohd@fit.cvut.cz`)

Date

November, 2014

4.3.2 Function Documentation

4.3.2.1 int serial_link (const Tconfig ∗ *config*)

Function to alloc serial link.

Parameters

`in`	*config*	Program configuration.

Return values

int	Success, return file descriptor.
0	Fail.

4.3.2.2 void serial_listener (int *fd*)

Listener function to listen on serial link.

Parameters

`in`	*fd*	File descriptor to serial link.

4.3.2.3 void socket_listener (int *fd*)

Listener function to listen on local unix socket.

Parameters

`in`	*fd*	File descriptor to socket.

4.3.2.4 void∗ socket_pthread (void ∗ *fd*)

Crete unix socket listener thread for parallel processing.

Parameters

`in`	*fd*	File descriptor to serial link.

4.3.2.5 int unix_socket (const Tconfig ∗ *config*)

Function to create local unix socket.

Parameters

`in`	*config*	Program configuration.

Return values

int	Success, return file descriptor.
0	Fail.

4.4 src/listeners.hpp File Reference

function to create listeners

```
#include "settings.hpp"
#include "logger.hpp"
```

Functions

- int unix_socket (const Tconfig ∗config)

  Function to create local unix socket.
- int serial_link (const Tconfig ∗config)

  Function to alloc serial link.
- void socket_listener (int fd)

  Listener function to listen on local unix socket.
- void serial_listener (int fd)

  Listener function to listen on serial link.
- void ∗ socket_pthread (void ∗fd)

  Crete unix socket listener thread for parallel processing.
- void ∗ **serial_pthread** (void ∗fd)

4.4.1 Detailed Description

function to create listeners Bohdan Vico (`vicobohd@fit.cvut.cz`)

Date

November, 2014

4.4.2 Function Documentation

4.4.2.1 int serial_link (const Tconfig ∗ *config*)

Function to alloc serial link.

Parameters

`in`	*config*	Program configuration.

Return values

int	Success, return file descriptor.
0	Fail.

4.4.2.2 void serial_listener (int *fd*)

Listener function to listen on serial link.

Parameters

`in`	*fd*	File descriptor to serial link.

4.4.2.3 void socket_listener (int *fd*)

Listener function to listen on local unix socket.

Parameters

`in`	*fd*	File descriptor to socket.

4.4.2.4 void∗ socket_pthread (void ∗ *fd*)

Crete unix socket listener thread for parallel processing.

Parameters

`in`	*fd*	File descriptor to serial link.

4.4.2.5 int unix_socket (const **Tconfig** ∗ *config*)

Function to create local unix socket.

Parameters

`in`	*config*	Program configuration.

Return values

int	Success, return file descriptor.
0	Fail.

4.5 src/logger.hpp File Reference

function to log errors

Functions

- void emerg (const char ∗msg)

  Emergenci error, program can not continue.
- void error (const char ∗msg)

  Error, program continue.
- void warn (const char ∗msg)

  Warning, program continue.

- void info (const char ∗msg)

  Info message, program continue.

4.5.1 Detailed Description

function to log errors Bohdan Vico (`vicobohd@fit.cvut.cz`)

Date

November, 2014

4.5.2 Function Documentation

4.5.2.1 void emerg (const char ∗ *msg*)

Emergenci error, program can not continue.

Parameters

`in`	*message*	

4.5.2.2 void error (const char ∗ *msg*)

Error, program continue.

Parameters

`in`	*message*	

4.5.2.3 void info (const char ∗ *msg*)

Info message, program continue.

Parameters

`in`	*message*	

4.5.2.4 void warn (const char ∗ *msg*)

Warning, program continue.

Parameters

`in`	*message*	

4.6 src/main.cpp File Reference

main function to run program

```
#include <sys/types.h>
#include <sys/stat.h>
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <errno.h>
#include <unistd.h>
#include <syslog.h>
#include <string.h>
#include <signal.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <pthread.h>
#include <pwd.h>
#include <grp.h>
#include "settings.hpp"
#include "logger.hpp"
#include "listeners.hpp"
#include "database.hpp"
```

Functions

- void signal_sigterm_handler (int signum)

  Signals handler function.
- int run_as_daemon ()

  Run program in background.
- int main (int argv, char **argc)

  Main function.

Variables

- Tconfig * **config**
- int **sfd**
- int **ufd**

4.6.1 Detailed Description

main function to run program Bohdan Vico (vicobohd@fit.cvut.cz)

Date

November, 2014

4.6.2 Function Documentation

4.6.2.1 int main (int *argv,* char ** *argc*)

Main function.

This function run first

Return values

0	success
1	fail

4.6.2.2 int run_as_daemon ( )

Run program in background.

This function run program in background, as daemon

Return values

int	Number of new pid.

4.6.2.3 void signal_sigterm_handler (int *signum*)

Signals handler function.

Parameters

`in`	*signum*	Signal number

4.7 src/query_lex.cpp File Reference

query lexer, use in database.cpp and listeners.cpp

```
#include <stdlib.h>
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <sys/select.h>
```

Classes

- class Cqlex

  Query lexer class.

4.7.1 Detailed Description

query lexer, use in database.cpp and listeners.cpp Bohdan Vico (`vicobohd@fit.cvut.cz`)

Date

November, 2014

4.8 src/regarray.cpp File Reference

register array database class

```
#include "regarray.hpp"
#include <string.h>
#include <stdlib.h>
#include <math.h>
#include <stdio.h>
#include "logger.hpp"
```

Macros

- #define **SAVE_FILE** "/regsave"

4.8.1 Detailed Description

register array database class Bohdan Vico (vicobohd@fit.cvut.cz)

Date

November, 2014

4.9 src/regarray.hpp File Reference

register array database class

```
#include <stdio.h>
#include <pthread.h>
```

Classes

- class Cregarray

4.9.1 Detailed Description

register array database class Bohdan Vico (vicobohd@fit.cvut.cz)

Date

November, 2014

4.10 src/settings.cpp File Reference

parse switches and configuration file, generate config struct

```
#include "string.h"
#include "stdio.h"
#include "stdlib.h"
#include "settings.hpp"
#include "logger.hpp"
#include "database.hpp"
```

Classes

- struct lex_symbol

Typedefs

- typedef struct lex_symbol **lex_symbol**

Enumerations

- enum **symbol_type** {
  PID_FILE, **SOCKET_NAME**, **RUN_BACKGROUND**, **DEVICE**,
  BOUND, **PARITY**, **IDENT**, **ENDFILE**,
  EQ, **INTEGER**, **DECIMAL**, **REGISTER**,
  RW, **RO** }

Functions

- void sw_error (const char ∗msg)

  Function to print error from switch.
- char get_sw (const char ∗x)

  Function check switch and return switch type.
- void lex_get_input ()

  Function get next character from configuration file.
- void is_keyword ()

  Function check if word is keyword.
- void lex_ident_add (char c)

  Function concat new character to ident string.
- void get_lex ()

  Automat to get next lexem from configuration file.
- void compare (symbol_type type)

  Function compare expect lexem with get lexem.
- int conf_flag_parse ()

  Function to parse RW|RO flag.
- double conf_float_parse ()

  Function to parse number.
- Tconfig ∗ conf_parser (const char ∗path)

  Function to parse configuration file.
- Tconfig ∗ switch_parser (int argv, char ∗∗argc)

  Function to parse switches.

Variables

- FILE ∗ **g_conf_file**
- lex_symbol **g_lex_symbol**
- char **g_char**
- int **g_linenum**

4.10.1 Detailed Description

parse switches and configuration file, generate config struct Bohdan Vico (`vicobohd@fit.cvut.cz`)

Date

November, 2014

4.10.2 Function Documentation

4.10.2.1 void compare (symbol_type *type*)

Function compare expect lexem with get lexem.

if get lexem not equal witch expect throw expection

4.10.2.2 int conf_flag_parse ( )

Function to parse RW|RO flag.

Return values

0	If RW
1	If RO

4.10.2.3 double conf_float_parse ( )

Function to parse number.

Return values

double	Number

4.10.2.4 Tconfig∗ conf_parser (const char ∗ *path*)

Function to parse configuration file.

parse configuration file and save directive to struct Tconfig

Return values

struct	Tconfig∗ Configuration file structure.

4.10.2.5 void get_lex ( )

Automat to get next lexem from configuration file.

Automat set variable g_lex_symbol with new lexem

4.10.2.6 char get_sw (const char ∗ *x*)

Function check switch and return switch type.

Parameters

in	*x*	Argument pointer

Return values

char	Switch type.

4.10.2.7 void is_keyword ( )

Function check if word is keyword.

if word is keyword set lex type to that keyword, other way set IDENT type

4.10.2.8 void lex_ident_add (char *c*)

Function concat new character to ident string.

Parameters

in	*c*	New character

4.10.2.9 void sw_error (const char ∗ *msg*)

Function to print error from switch.

Parameters

in	*msg*	message to print

4.10.2.10 Tconfig∗ switch_parser (int *argv,* char ∗∗ *argc*)

Function to parse switches.

Return values

struct	Tconfig∗ Configuration file structure.

4.11 src/settings.hpp File Reference

parse switches and configuration file, generate config struct

Classes

- struct Tconfig

Typedefs

- typedef struct Tconfig **Tconfig**

Functions

- Tconfig ∗ switch_parser (int argv, char ∗∗argc)

  Function to parse switches.
- void **delete_config** (Tconfig ∗x)

4.11.1 Detailed Description

parse switches and configuration file, generate config struct Bohdan Vico (`vicobohd@fit.cvut.cz`)

Date

November, 2014

4.11.2 Function Documentation

4.11.2.1 Tconfig∗ switch_parser (int *argv*, char ∗∗ *argc*)

Function to parse switches.

Return values

struct	Tconfig∗ Configuration file structure.