Multichannel data-acquisition systems

LTR Crate System

Software for LTR System under Windows, Linux, FreeBSD and QNX OS

Revision 1.0.5
November 2016

L-CARD

http://en.lcard.ru
en@lcard.ru

DAQ SYSTEMS DESIGN, MANUFACTURING & DISTRIBUTION

Author of the manual:
Alexey Borisov

L-Card LLC
117105, Moscow, Varshavskoye shosse, 5, block 4, bld. 2

tel.: +7 (495) 785-95-19
fax: +7 (495) 785-95-14

Internet contacts:

http://en.lcard.ru/

E-Mail:

Sales department: en@lcard.ru
Customer care: en@lcard.ru

Table 1: Current document revisions

Revision	Date	Description
1.0.0	February 12, 2013	The document first revision
1.0.1	April 22, 2013	The description of utilities for configuring PC communication interface and firmware update for LTR-EU crates was added
1.0.2	April 16, 2014	The Qltrd program's name was changed to LTR Manager. The description of possibility of ltrd log programmed obtain was added, the description of possibility of running ltrd as Windows OS service was added. The description of LTR-EU crates interface configuration was added from LTR Manager.
1.0.3	July 27, 2016	The description was updated taking into account the fact that cross-platform software is the main version and under Windows OS, as well as the appearance of the document "Starting operating the LTR crate system. Software issues". The information on possibility to operate under FreeBSD was added. The software installation section was separated and the section on capabilities comparison was revised. The appearance of LTR-CEU crates was taken into account and the capability of firmware update from LTR Manager was described. Also some general corrections were made to the text
1.0.4	October 12, 2016	Some of the information on ltrapi was included in the updated manual for this library. The description of crate synchronization settings control was added in the LTR Manager program
1.0.5	November 08, 2016	The information on reconnection flag in case of connection fault was added for entries with crates IP addresses and corresponding information in ltrd configuration

Contents

Chapter 1

What this document is about

This document is a description of cross-platform software for operating crates and LTR modules and its usage manual. This version of the software is the main supported version for all operating systems, including Windows OS, for which the new version is a compatible replacement for previously used software under Windows (based on the LTR Server program and ltrapi libraries, version 1.27 and below). The software version described in this document is recommended for use in all projects, but for older projects, see section 3.2, which describes possible incompatibilities in some rare cases (if this is the case, we recommend contacting the support desk).

The document assumes that the user has read the document "Starting operating the LTR crate system. Software issues", providing an overview of the software.

This document describes in detail the programs that are part of the basic software, their operation features for different supported operating systems, etc.

This document does not discuss the hardware capabilities of LTR crates and modules, as this information is contained in the "LTR User Guide".

Chapter 2

Installation

For some of the supported systems the pre-built tools are offered for software installation. The way the pre-built software is installed depends on the operating system and is described in the relevant sections below.

Also, for any operating system, it is possible to build the software from the source codes, as described in Section 2.6.

2.1 Windows

The pre-built installers are created for Windows OS (versions with Windows XP and higher are supported). The installation order is described in the document "Starting operating the LTR crate system. Software issues".

2.2 Linux

For some Linux distributions, L-Card provides repositories with pre-built packages. How to connect the L-Card repository as an external repository in supported Linux distributions or install packages manually is described in the document "Using external L-Card repositories for Linux distributions".

Here you can find a specific list of packages, corresponding to the LTR operating software.

- ltrd — installs the daemon, runs it, and also allows it to run at system startup. It can depend on packages with libxml2 and libusb-1.0 libraries and on some systems also on libdaemon or systemd.

- Packages for installing ltrapi libraries. In .rpm and .deb distributions, it is common to separate packages for libraries into packages with runtime libraries that are used by the pre-built programs, and packages for developers (with headers and symbolic links) that are used in writing their programs[1]. In addition, for each library, there must be a package with a name that corresponds to the library name. This will allow to track automatically the application dependencies on the packages of the libraries used. Thus, the ltrapi files are divided into the following packages:

  – The libltrapi1 package with a shared library for access to ltrd and crate control.

[1] There is no such practice for Arch Linux, and everything is included in one package named ltrapi

  - Per package for each library to operate the module named libltrXXXapi1 (libltr11api1, libltr22api1, etc.). These packages depend on libltrapi1. Packages for modules, to which the firmware of the controller or FPGA needs to be written each time; in addition to the .so file the firmware file, which is installed normally in /usr/share/ltrapi/ltrXXX is also included.
  - libltrapi1-dev or libltrapi1-devel - a common package with headers for all libraries, depending on all the library packages mentioned above.

  If you are installing from the connected repository, the dependencies should be resolved automatically and you can install directly the development package for applications development. If you install packages manually, you must install the packages in the specified order.

- ltrctl — installs the corresponding utility. Depends on libltrapi1
- ltrmanager — installs the LTR Manger program. Depends on Qt4 or Qt5. It uses ltrapi libraries, as well as the lboot program for updating the LTR-CEU crates firmware.
- ltreu-config — a package with the command line utility for configuring the LTR-EU crates interface (see Section 4.5.1).
- ltreu-firm-update — a package with a command line utility for updating the LTR-EU crates CPU firmware (see Section 4.5.1).

2.3 FreeBSD

The ltrd daemon, the ltrapi library, and the ltrctl utility are ported for FreeBSD. They can be installed in two different ways as described below.

2.3.1 Pre-built binary packages installation

For supported versions of FreeBSD and supported architectures, the pre-built packages are provided. They can be downloaded through the link https://bitbucket.org/lcard/ltr_cross_sdk/downloads/ltr-freebsd-packages.tar.gz.

After unzipping, you can install the packages using the pkg utility (see the description of using the utility on the pagehttps://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/pkgng-intro.html), getting the packages from the desired directory corresponding to the desired version and OS architecture (denoted as <ltr_pkg_dir>):

```
pkg install <ltr_pkg_dir>/*
```

2.3.2 Installation through Ports Collection

This option, unlike others, does not install the pre-built binary packages, but performs the installation by building from the sources using Ports Collection.

This keeps track of the package dependencies, installed files as well as other features, as with the installation of packages.
If the FreeBSD ports collection is not installed, you need to install it first:

```
portsnap fetch extract
```

And if installed, then to update it to the latest version:

```
portsnap fetch update
```

Next, you need to download the version of the ports for LTR from the archive by the link: https://bitbucket.org/lcard/ltr_cross_sdk/downloads/ltr-freebsd-ports.tar.gz.

The contents of the ports directory of the archive should be copied to the directory with the installed collection of ports (/usr/ports, can be redefined via $(PORTSDIR)).

Then from the directory of each port (for example, /usr/ports/science/ltrd) perform the port installation:

```
make install
```

The port can always be then deleted by executing from its directory:

```
make deinstall
```

The source codes of the utilities and libraries are not part of the port, and the source code archive is downloaded from the site during the installation. If you need to download archives with sources in a separate stage, so that further building is performed without the need for the Internet access, you can execute from the port directory

```
make fetch
```

Or you can display a list of downloaded files through (or rather the commands that are executed in the fetch stage):

```
make fetch-list
```

and manually copy the files and transfer them to the /usr/ports/distfiles directory.

2.4 QNX6.5

A binary build of utilities and libraries for x86 architecture can be downloaded through the link: https://bitbucket.org/lcard/ltr_cross_sdk/downloads/ltr_qnx6.tar.gz. After that, just copy the contents of the archive to the target system. Running ltrd or adding to autorun should be done manually.

2.5 QNX4.25

A binary build of utilities and libraries for x86 architecture can be downloaded through the link: https://bitbucket.org/lcard/ltr_cross_sdk/downloads/ltr_qnx4.zip. After

that, just copy the contents of the archive to the target system. Running ltrd or adding to autorun should be done manually.

2.6 Source codes build

You can download the source archive from the repository page of this project. However, to get the latest version of the source codes, it is recommended to clone the repository by the Mercurial program. More information about using the repository can be find in the document "Using open L-Card source repositories" at bitbucket.org.

To build, you need the cmake utility, which configures the project depending on the system, as well as the supported C/C ++ compiler.

The build process is described in the INSTALL.txt file of the source code archive.

Chapter 3

Differences in supported capabilities

3.1 Differences in support for different OS

- USB interface support is available only for Windows OS and for Linux OS with the libusb-1.0 library. For the rest OS, only crate operation via the Ethernet interface through the TCP/IP protocol is supported (i.e. only crates with this interface can be used). Thereof, only the OS with USB support enable the utilities for operations that are possible only via USB interface (interface configuration and firmware update for LTR-EU crates).

Important! This means, in particular, that setting up the crate for operating via Ethernet interface and assigning the desired IPv4 address must be pre-done under Windows or Linux OS

- The LTR Manager graphical utility is available only for Windows and Linux OS.
- For the QNX OS, ltrd operates as a normal console program, that needs to be run in the background manually or to be added to the autorun scripts (for the rest OS, the run of ltrd as a service/daemon is implemented in the system's regular manner).
- For QNX6/QNX4 OS, it should be taken into account that the version uses the same source code and the same ideology of interaction with modules as the version under the rest OS of general purpose, and is not specialized software that takes into account all features of the real-time operating system. The user should evaluate the applicability of this software depending on the task he is solving.

3.2 New version limitations in comparison with the old version of operating through LTR Server

The cross-platform software option for LTR provides features similar to those that previously used version for Windows OS provides, with the exception of the following items:

- The capability of receiving "raw" primary ltrd data using LTR_GetCrateRawData() is not supported.
- The capability of creating a special user virtual slot for receiving user data from the crate is not supported

- The ltrd operation priority change using
- LTR_SetServerProcessPriority() is not supported
- Some debugging modes are not included
- There may not be compatibility with the software, which explicitly requires a running program with the name LTR Server (for example, it displays a window or a process with that name or runs explicitly LTR Server program itself)
- Differences of the new ltrapi version are described in the “ltrapi library programmer's manual”.

Chapter 4

Software configuration

To operate LTR and write your own programs the following software components are used:

- ltrd — running in the background service program (service or daemon) which serves crates connections, parses data stream from the crate and gets data stream ready for the crate. All interaction with the crates is done through ltrd.
- ltrapi — a set of libraries providing functions for modules operating, for crates control and the ltrd itself operation control. To perform these actions, a client connection with the already running ltrd is established in library functions.
- LTR Manager — a utility with a graphical interface for receiving information on connected crates, statistics, ltrd log, as well as for connecting crates via Ethernet, crates setting up, firmware upgrading and other optional actions.
- ltrctl — a utility with a command line interface that performs functions similar to LTR Manager
- additional utilities for operating certain crates or modules

In general, the running ltrd program is required to operating the modules. The ltrapi libraries, that enable the program to establish a connection to ltrd and operate the required modules (each module has its own connection), must be installed on a computer with a user program (it may not be necessarily the same computer on which ltrd is running).

It should be noted that one ltrd instance can establish a connection with several crates, while only one ltrd service can work with one crate at a time. If it is necessary to operate different modules of the same crate from different computers, then ltrd must be run on one machine only, while programs running on the second computer must establish a connection with ltrd, running on the first.

The difference when working with the remote (as opposed to working with the local) ltrd is only that when you call the corresponding LTR_Open () function, you need to set the IP address of the computer running ltrd, not LTRD_ADDR_DEFAULT, as with the local connection.

It should also be noted that the interface between ltrapi and ltrd is compatible with the corresponding interface of the previously used LTR Server for Windows OS. Therefore, the same program can work with both ltrd and LTR Server programs. It should be noted that some commands are available only in ltrd (getting statistics,

resetting the module, setting the ltrd parameters), while some rarely used LTR Server features are not implemented in ltrd (see section 3.2).

4.1 ltrd

The ltrd service is required for all other software operation, for interaction with LTR modules and crates. ltrd establishes a connection with the crates, thereafter the other programs can operate the necessary modules and crates through the client connections with ltrd. The ltrd program runs in the background, normally starts along with the system and does not contain a user interface.

To provide an interface for ltrd control, the additional programs (frontends) are used: ltrctl (provides a command line interface) and LTR Manager (provides a graphical interface using the Qt4/Qt5 library). These programs use only the ltrapi functions, so if necessary, you can write your own interfaces in addition to those provided.

4.1.1 Configuration file

ltrd stores its settings in an XML configuration file. All these settings can be changed programmatically through the ltrapi functions or through LTR Manger, as well as directly by editing the configuration file, as will be described below. If the user does not plan to change the configuration file manually, then this section can be skipped.

In standard packages for Linux OS, the configuration file is installed in /etc/ltrd/config.xml, and in the Windows OS - into the directory together with ltrd. The path to the file is specified using the --config-file option when starting ltrd. An example of this file is given in the source code with the program.

The corresponding hierarchical list of tags with a description of the parameters to be set is given below.

The document tag must always be <ltrsrv_config>. It contains the following tags:

- <log>. Log Settings
  - <level>. Specifies the level of the log output messages from 0 to 7 (see section 4.1.3).
- <crate_ip_entries>. Contains entries with crates IP addresses (see section 4.1.4.1)
  - <ip>. These tags can be any number. Each describes its entry. The description contains a line with the LTR-crate IPv4 address. Each tag can have the following attributes:
    - "autocon" — specifies whether to automatically connect to this crate when starting ltrd (see section 4.1.4.3). Can take the values "1" or "0".
    - “reconnect” — specifies whether to retry the connection if it fails to connect to the crate (see section 4.1.4.4). Can take the values "1" or "0".
- <intf_eth_params>. Parameters for advanced configuration of the interface between ltrd and crate when working via Ethernet. In their absence the default values are used. In most cases, the user does not need to change these settings.

– <intf_check_time>. The interval (in ms) between checks of the addresses of the local machine interfaces. Used in determining the time to start the auto-connection of the crate.

– <crate_poll_time>. Interval (in ms) of crates poll. If there was no exchange with the crate during this interval, a command will be sent to it to verify that the crate is still on the network. If this value is set to 0, then crates poll will not be performed.

– <crate_connection_timeout>. Timeout (in ms) for the time of establishing the TCP connection with the crate

– <crate_ctlcmd_timeout>. Timeout (in ms) for the time of operation of the control commands sent to the crate (the time limit for waiting for a response to the command).

– <crate_reconnect_time>. Time (in ms) after a connection with the crate is failed, upon which a re-connection attempt will be made (if the corresponding attribute is set for this record).

When changing these settings using the API functions of ltrapi, ltrctl or LTR Manager, ltrd immediately saves the modified file. To save the changes, you need to make sure that the user on whose behalf ltrd is running has the right to write the configuration file (otherwise the changes will not be saved between the ltrd starts).

If necessary, you can edit the configuration file manually when ltrd is stopped (otherwise, ltrd will overwrite the file when it is shut down). Also for OS that support the sending of POSIX signals (all support, except for Windows), it is possible to change the configuration file manually when ltrd is running. To do this, after changing the configuration file, you need to send a ltrd signal SIG_HUP, by which ltrd reads the settings from the file again.

4.1.2 Running ltrd

When installing through the installer on Windows OS, installing packages for supported Linux distributions or a package/port on FreeBSD, all necessary settings for the autorun of the service at system startup are performed automatically during installation. In this case, the user does not need to know the details of the service run described in this section.

The following information on how the run is performed is rather for educational purposes, for the case if you need to manually stop/start the service, and also for ltrd run on the OS for which there are no pre-built packages.

4.1.2.1 Running ltrd as a daemon

In Linux, FreeBSD and other OS, daemons are usually used to perform background tasks. The daemons are started along with the system, not connected to any session and user terminal, and their parent process is the init process responsible for starting the system.

For running in daemon mode, the --daemon option is used.

When running in daemon mode, ltrd executes the following standard for the daemon steps:

- If the --user and/or --group options were specified, then ltrd changes the user and the group on whose behalf it is executing to the specified values. This is used to get rid of the superuser (root) privileges, in the case where ltrd is run on behalf of the superuser at system startup. When installing packages from the repository, a special user and a group named ltrd, on whose behalf the daemon is executed, is created
- Reset all signal handlers
- Check if another copy of the daemon is running on the pid file specified with the --pid-file option. When installing from packages, the file /var/run/ltrd/pid is used
- Creating a child process independent of the terminal
- The child process is made the session leader
- The child process sets the root directory (/) as the current directory
- The child process closes all file descriptors
- The child process creates a pid file specified by the --pid-file option so that the daemon can not be started several times. In this regard, the user on whose behalf the daemon is running must have the appropriate rights
- The child process informs the parent process that the initialization was successful.
- The parent process is completed and only the child process is left and it performs all the work
- Upon completion, the child process deletes the pid file

To start the daemons, special init scripts are used. Their organization depends on the specific distribution.

In working packages, init scripts are used in Debian-based distributions. These distributions use System V scripts (SysV-style init scripts). The ltrd init-script is installed in them in /etc/init.d. For automatic run and stop, symbolic links to /etc/init.d/ltrd are created in the directories /etc/rcN.d with the names S??ltrd (for running) or K??ltrd (for stopping), where N is the corresponding runlevel, and ?? is a two-digit number that determines the sequence of running scripts. The symbolic links are created with the help of the system script update-rc.d, which can determine the runlevels and the sequence of running scripts by reference to the header in the init script itself.

Manual run of the daemon is performed using:

```
sudo invoke-rc.d ltrd start
```

Manual demon stop:

```
sudo invoke-rc.d ltrd stop
```

4.1.2.2 Running ltrd using systemd

Distributions such as Fedora, OpenSuse and Arch Linux have been upgraded from traditional init scripts to the use of systemd for system startup and services control.

Unlike traditional demons, systemd-daemons do not perform all the above steps themselves, since these actions are already performed for them by systemd. Instead of scripts, a declarative file describing the service ltrd.service is used (normally installed in /lib/systemd/system or /usr/lib/systemd/system when installing from repositories, and /etc/systemd/system is used for manual installation, generally).

Upon that, in order to support systemd additional features, ltrd is run with the --systemd-mode key.

When operating systemd, the following additional features are used:

- ltrd restart in case it was completed with an error
- Watchdog-timer: ltrd periodically sends a notification to systemd. If it is not sent within the specified interval (which can happen only when the daemon hangs), systemd will restart the daemon.
- ltrd messages are sent to the systemd log, where they are usually redirected to the system log via syslog. The systemd log can be viewed using journalctl (see example below) Manual start of the service is performed using:

  ```
  sudo systemctl start ltrd.service
  ```

  Manual stop:

  ```
  sudo systemctl stop ltrd.service
  ```

  Run permissions with system startup:

  ```
  sudo systemctl enable ltrd.service
  ```

  Prohibition of run with system startup:

  ```
  sudo systemctl disable ltrd.service
  ```

  Check the operation status:

  ```
  sudo systemctl status ltrd.service
  ```

The output of the last 20 entries related to ltrd from the systemd log (the q key exits the log view):

```
sudo journalctl --lines=20 _SYSTEMD_UNIT=ltrd.service
```

4.1.2.3 Running ltrd as FreeBSD daemon

In FreeBSD OS, ltrd is run as a daemon in the same way as described in section 4.1.2.1, using libdaemon and with the similar actions on startup. The difference is the location and configuration of the running scripts that are used in FreeBSD to control daemons.

In FreeBSD, the running scripts are located in the directories /etc/rc.d or /usr/local/etc/rc.d. The latter is used by ports and it is in it where the ltrd running script is installed. After installation, the daemon is accepted and running. It can be forbidden, if necessary, via rc.conf, by adding a variable

```
ltrd_enabled="NO"
```

To control use the utility service, as for other services (as described in the corresponding section of the FreeBSD documentation): Checking whether the service is allowed:

```
service ltrd rcvar
```

Checking whether the service is running:

```
service ltrd status
```

Manual service stop:

```
service ltrd onestop
```

Manual service run:

```
service ltrd onestart
```

4.1.2.4 Running ltrd as a Windows service

Under Windows OS ltrd with the --daemon key runs as a Windows service. All necessary actions can be performed using standard Windows tools. However, some of the actions are implemented for convenience within the ltrd program and are executed by running ltrd with special keys:

- --install — the ltrd service system registration. Before running the service, it must be registered.
- --remove — deleting information about the registered ltrd service
- --start — manual ltrd service run
- --stop — manual ltrd service stop
- --autostart-enable — enables the ltrd service autorun along with the system
- --autostart-disable — disables the ltrd service autorun along with the system

When installed by a standard installer, the service is immediately started and its run along with the system is enabled, therefore no further action is required. In the same way shortcuts for the service manual run/stop and enable/disable of its startup along with the system can be created.

Important! The ltrd service and the LTR Server program can not be run simultaneously on the same machine. For correct ltrd operation, the LTR Server program must be closed, and the ltrd service must be stopped to run LTR Server ltrd

4.1.2.5 Running ltrd as a regular console background program

ltrd can be run as a common console program to simplify the run on systems where all the necessary settings are not performed automatically by installation of packages (not mentioned Linux distributions and in QNX). All it takes is specifying the configuration file when running ltrd, (for the case when the configuration file is located in /etc/ltrd/config.xml):

```
ltrd --config-file=/etc/ltrd/config.xml &
```

If you want to change the user (to lower the privileges, if the run comes from the superuser), you can use the options --user =<username> and --group=<group>.

4.1.3 ltrd log

ltrd displays messages about its work (log) in one of the following ways:

- Output to stdout - this option is used when starting ltrd as a common console program
- Under Windows OS, when running as a service, the log messages are saved to the system event log.
- Saving to the system log with the syslog() call for OS that support this function — this option can be used both in the child daemon process and at running ltrd as a console program. The log messages are saved by syslogd (or its analog), usually to one of the files in the /var/log directory (for example /var/log/messages) and can be viewed by standard system tools for system log viewing.
- When running with the --systemd-mode key, the log is saved to the systemd log (from which is usually also transferred to the system log) and can be viewed via journalctl (as mentioned in section 3.2).
- The log messages are also saved to the circular buffer (if it's were allowed during the configuration) and can be received by the client program through the functions of the ltrlogapi library, which establishes a special type of control connection with ltrd. In particular, this library is used by the LTR Manager program to output the ltrd log to a special window. In the examples, there is also an option for implementing a console program that outputs ltrd messages to the standard output.

Each output message is assigned an importance level from 0 to 7 (0 - critical errors, 7 - the lowest level for debugging). For ltrd, there is a setting for the level of output messages, which allows you not to output messages of higher levels (less critical). It should be noted that in normal ltrd operation, you should not set a high output level of debug messages (levels 6,7) in order not to unnecessarily load the system log. A typical level is 3, at which the output of the main messages on the progress of work and all warnings/errors is performed, or 4 - where additional detailed information is added to the mentioned.

The log level can be configured in the configuration file or changed when working in one of the following ways:

- ltrapi — using the LTR_SetLogLevel() function.
- ltrctl — to set the level 4:

```
ltrctl setloglvl 4 perm
```

- LTR Manager — select the desired connection with ltrd from the active list. Select "ltrd settings..." from the menu "ltrd" or from the context menu. In the opened window, set the desired level and click "Ok". In LTR Manager, the displayed level of messages in the program window is also individually set, i.e. it accepts all messages that ltrd send, but can display in the window only those that satisfy the configured display level. The message is displayed only if it is equally no less critical than the level set in ltrd and the display level in the LTR Manger window

4.1.4 Managing crate connections via Ethernet interface

The general principle of operating crates via Ethernet is described in the document"Starting operating the LTR crate system. Software issues". A detailed description of each stage is given here.

4.1.4.1 Crates address list

ltrd stores a list of entries with the IP addresses of the crates, with which, if desired, a connection *can be* established. The list of entries is in itself only for storing these addresses. In addition to the address itself, the entry may contain additional flags (sections 4.1.4.3 and 4.1.4.4). Ltrd loads the list of crate addresses from the configuration file at startup and saves it at the end of the operation (except when the changes were made without saving).

During operation, new entries with crates addresses can be added or deleted:

- ltrapi — using the LTR_AddIPCrate() and LTR_DeleteIPCrate() functions respectively.
- ltrctl:
  - adding an address 192.168.0.250:

```
ltrctl ipadd 192.168.0.250 perm
```

  - deleting an address 192.168.0.250:

```
ltrctl iprem 192.168.0.250 perm
```

- LTR Manager
  - To add, click the "Add IP-address" button in the "crate IP address" panel, enter the address in the appeared window and click "Ok"

– To delete, select the desired address in the "crate IP address" panel and click "Delete IP address"

4.1.4.2 Establishing a connection with the crate

You can manually establish connections between ltrd and the crate over the Ethernet interface using one of the IP addresses from the configured list in ltrd. That is, before making a connection, the entry with the desired crate IP address should already be pre-added to the list.

You can start the process of establishing a connection with the crate in the following way:

- ltrapi — using the LTR_ConnectIPCrate() function
- ltrctl: — connection to the address 192.168.0.250:

```
ltrctl ipcon 192.168.0.250
```

- LTR Manager Double-click on the line with the address in the "crates IP addresses" panel, for which there is no connection, or highlight the address and click "Establish connection with the crate"

It should be taken into account that the successful execution of the above operation only indicates the successful execution of the command at the beginning of the connection, i.e. the connection itself is not yet completed at this point.

After the command to connect, ltrd performs the following actions:

- Attempts to establish control connections to the crate using TCP protocol
- Reads information about the crate
- Establishes a TCP connection to the crate for data transferring

After the successful completion of all steps, the crate is added to the active crates list, and then a client connection can be established with it or its modules.

The status of the connection can also be detected by the status of the connection related to the corresponding entry, which can be obtained as follows:

- ltrapi — using the LTR_GetListOfIPCrates() function ·
- ltrctl:

```
ltrctl iplist
```

- LTR Manager The list of addresses with the connection status is always displayed in the table in the "crates IP addresses"

After starting the connection, the entry status changes to "Connecting...", then it goes either to the "Online" status on successful connection completion, or to the "Error" state when a connection error occurs.

At any time, you can break the established (or in the process of establishing) connection using:

- ltrapi - connection with a given address can be broken using the function LTR_DisconnectIPCrate (), and break all connections using LTR_DisconnectAllIPCrates ().
- ltrctl:
  - disconnect the crate with address 192.168.0.250:

```
ltrctl ipdisc 192.168.0.250
```

  - disconect all crates

```
ltrctl ipdisc all
```

- LTR Manager Double-click on the line with the address in the "crates IP addresses" panel, with which the connection is established or highlight the address and click "Disconnect the crate". To break all connections use the button "Disconnect all crates"

4.1.4.3 Automatic crate connection at startup

Records with crates addresses can be marked with the flag "Auto-connection". In addition to being able to manually connect to the crate using these entries, ltrd also tries to establish a connection for these entries automatically in the following cases:

- When running the ltrd program, if there is at least one valid IP address of the host on which ltrd is running
- If a new host address with running ltrd is detected, ltrd tries to establish a connection to all addresses with the "Auto-connection" flag, to which a successful connection has not yet been established. For this, ltrd periodically (the period is a configurable parameter) gets the local machine address.
- On command to connect all crates entries of which are marked with the "Auto-connection" flag (see below), if the connection to the crate at the time of the command is not already established.

The second option allows you to resolve the situation, when the host address is actually obtained after running ltrd. A typical example of this is the option, when ltrd is run along with the system startup, and the machine address is obtained automatically through DHCP.

Note: If the crate connection fails or the crate connection will be broken later due to an error, ltrd changes the entry status to "Error" and by default does not try to establish a connection to this address any more, even if there is the "Auto-connection" flag (except when the host address is changed), and the connection can be established after that manually only. To enable automatic reconnection in this case, use the special flag described in section 4.1.4.4. Thus, by default (without the reconnection flag), to

perform auto-connection, it is necessary that the crate is already on the network at the time of running ltrd.

You can specify that auto connection is required you can by adding a new entry:

- ltrapi — using the flag LTR_CRATE_IP_FLAG_AUTOCONNECT at LTR_AddIPCrate() function call
- ltrctl: —adding an address 192.168.0.250 with auto connection flag:

```
ltrctl ipadd 192.168.0.250 auto perm
```

- LTR Manager — fill the "Automatically connect on startup" checkbox in the Adding Address dialog

You can also change the state of the auto-connection flag for an entry that has been already added:

- ltrapi — by LTR_SetIPCrateFlags() function with flag LTR_CRATE_IP_FLAG_AUTOCONNECT.
- ltrctl:

  – setting the auto-connection flag for the address 192.168.0.250:

```
ltrctl ipflag 192.168.0.250 auto set perm
```

  – reset auto-connection flag for the address 192.168.0.250:

```
ltrctl ipflag 192.168.0.250 auto clear perm
```

- LTR Manager — to check or uncheck the "A" column in the address table of the "crates IP addresses" panel

Note: Setting the auto-connection flag when adding an entry or for an entry that has already been added, does not lead to a connection at the corresponding address, but only tells ltrd that the auto-connection must be performed the next time the program is started or at the another event for auto-connection described earlier.

You can also use the auto-connection flag to manually start by one command a connection to all crates for which entries with IP addresses are flagged with this flag:

- ltrapi — using the function LTR_ConnectAllAutoIPCrates()
- ltrctl:

```
ltrctl ipcon auto
```

- LTR Manager — click the button “Crates auto-connection” on the panel “Crate IP address”,

4.1.4.4 Reconnecting on error

For entries with crates addresses, the "Reconnect on error" flag can be set. This feature is implemented in ltrd since version 2.1.5.0 and this feature control is added to ltrapi since version 1.31.1, to LTR Manager since version 1.5.1 and to ltrctl since version 1.0.1.

This feature indicates that if you fail to connect to the crate or if the already connected crate is disconnected due to an error (i.e. in all cases when the IP entry goes into the "Error" state), you must arrange a new connection attempt. If the reconnection fails, the next attempt will be arranged and so forth until you can establish a connection or one of the following actions will be performed explicitly:

- The command to disconnect the crate (the "Error" state will change to "Disabled" and no repeated attempts to connect will be made).
- Removing the reconnection flag for this entry (see below)
- Deleting the entry

The time upon which the retransmission will be arranged is set by a special additional configuration parameter of ltrd service and it does not change with each new attempt (however, it can vary from one to one and a half of the interval set in the configuration).

This feature allows you to automatically restore a broken connection to the crate caused by a temporary error (reload of the crate, temporary loss of communication due to a connection failure, etc.), and also allows you to run the connection process while the crate is not yet connected to the network without the need to implement this logic in the application.

This flag can be used in conjunction with the "Auto-connection" flag, or with a manual connection.

You can specify that you want to reconnect if an error occurs for a new entry when adding an entry:

- ltrapi — using the flag LTR_CRATE_IP_FLAG_RECONNECT at LTR_AddIPCrate() function call
- ltrctl: — adding an address 192.168.0.250 with the reconnection flag:

```
ltrctl ipadd 192.168.0.250 reconnect perm
```

- LTR Manager — fill the "Reconnection on startup" checkbox in the Adding Address dialog

You can also change the state of this flag for an entry that has already been added:

- ltrapi — by function LTR_SetIPCrateFlags() with the flag LTR_CRATE_IP_FLAG_RECONNECT.
- ltrctl:
  - setting the reconnection flag for the address 192.168.0.250:

```
ltrctl ipflag 192.168.0.250 reconnect set perm
```

– reset the reconnection flag for address 192.168.0.250:

```
ltrctl ipflag 192.168.0.250 reconnect clear perm
```

- LTR Manager — to check or uncheck the "R" column in the address table of the "crates IP addresses" panel

Note: Setting the reconnection flag for an entry, for which the connection state already corresponds to an error, causes a reconnection attempt to be arranged within the specified interval.

4.1.5 Crate connection establishment via the USB interface

The USB interface is currently supported in ltrd under Linux OS and Windows OS.

In Windows OS, the "L-Card" driver for lcomp USB devices is used, and it must be separately installed.

In Linux OS, the library libusb (version 1.0.8 and higher) is available in most distributions. In most distributions, by default, only root has access to unknown USB devices. To allow access the ltr-crates.rules file is used with rules for udev. This file is installed when installing from the pre-built packages to the directory /usr/lib/udev/rules.d/ (on some distributions you can use /lib/udev/rules.d/), and for manual installation, the directory /etc/udev/rules.d. usually is used. The specified file makes the ltrd user the owner of the USB crate device files and it is this user to whom it gives an access to the supported devices. That is, if you run ltrd manually, then you need to either run it from the ltrd user name (you can use the --user and --group options if the user and group are already added to the system), or modify the rule file accordingly to give access to all. Also, when installing the rules file manually, in oder to not reload the system or reconnect the already connected crate, you can update the udev rules and generate the events for the desired devices:

```
sudo udevadm control --reload-rules

sudo udevadm trigger --subsystem-match=usb
```

When installing packages, all the necessary actions are already performed during the installation.

When a new crate connected via USB is found, ltrd establishes a connection with it and initializes it automatically. No additional settings are required. Only after the completion of the entire initialization the crate falls into the list of active crates.

It should be noted that when connecting via USB to a crate configured to operate via Ethernet, it will be seen via USB, but in a special mode without modules, that is used only for setting up the crate. You can track this state by looking at the crate statistics:

- ltrapi — after calling LTR_GetCrateStatistic() on the crate_mode field (one of the constants LTR_CRATE_MODE_).

- ltrctl — in the line “Crate mode” at the statistics output

```
ltrctl cstat
```

- LTR Manager — is displayed in the statistics panel in the crate data as "Operating mode"

4.1.6 List of active crates

ltrd contains the list of active crates with which ltrd has successfully established connections. Clients can establish client connections only with the crates from this list and their modules. The crates are included into this list after the successful completion of the connection initialization.

Removing from the active crates list is performed in the following cases:

- With an explicit command to shutdown the crate by IP address (in the case of the Ethernet interface)
- If a control command failed to be sent to the crate or a data transmission/reception error occurred (indicates a communication failure)
- If the crate disappears in the system

When removing a crate from the active list, ltrd closes the connections of all clients connected to this crate or its modules.

To verify the validity of the crate connection via Ethernet, ltrd can perform periodic poll of the crate modules configuration. If there is no answer to the command for the specified time, it is considered that the connection with the crate is not valid. The poll period and timeout are configurable parameters of ltrd.

The list of active crates and their modules can be obtained as follows:

- ltrapi — using the function LTR_GetCratesEx(), which immediately returns a list of active crates with information on them. You can also use for this purpose the function LTR_GetCrates () (which was entered earlier and left for compatibility), you can get a list of the serial numbers of the crates, after which these numbers can be used to connect to the crates using LTR_Open () and get from each a list of modules through LTR_GetCrateModules (). However, the second option can only be used to obtain the first LTR_CRATES_MAX crates and does not distinguish the case of simultaneous connection of the crate via two interfaces.
- ltrctl:

```
ltrctl clist
```

- LTR Manager — in the form of a tree in the main program window, for each instance of ltrd a list of its active crates with a list of modules is displayed

4.1.7 Client connections

To operate crates or modules, and to control the operation of ltrd, the application programs establish a client connection with ltrd. There are three possible types of connections:

- Control connection with ltrd. This connection allows you to control the ltrd itself and receive information from it. It does not require the presence of active crates. Commands on this channel are handled by ltrd itself.

- Control crate connection. Allows you to control the corresponding crate (for example, getting a list of modules or configuring sync-tags for crates). ltrd converts the control commands for this interface to the crate corresponding commands or processes itself.

- Connection to the module inserted in the crate. With this connection, the client operates the module using the protocol of a specific module. Unlike previous connections, the exchange on this channel represents an opaque data stream to input or output for ltrd. ltrd does not know the protocols of the modules and only performs multiplexing/demultiplexing of data. For transmission, data is inserted into the general data stream to the corresponding crate, only the fields that indicate which slot this data is assigned for are changed. When receiving data from the crate by the corresponding field, it is determined from which module these words have come, and these words are transmitted to the client, who installed the connection with the module, unchanged (or discarded if there are no clients).

Application programs establish client connections using the functions of the ltrapi library.

While there may be several control connections (both to ltrd, and to the crate) at a time, the client connection with each module in the standard mode of operation may be the only one. In an attempt to connect to the module connection with which is already established, ltrd will create a connection, but will return a warning that the module is already in use. Information on how this warning is processed in ltrapi can be found in the programmer's manual for ltrapi library.

If you want to reset all client connections related to the given module so that you can open a new connection without this warning, you can reset the specific module:

- ltrapi — using the function LTR_ResetModule() for the established control connection with ltrd.

- ltrctl:

  – Resetting the slot 6 in the first found crate:

```
ltrctl mrst 6
```

  – Resetting the slot 6 in the crate 1R234567

```
ltrctl mrst 6 1R234567
```

- LTR Manager — right-click on the desired module and select "Reset module" from the context menu. Or select the desired module and select "Module -> Reset module" in the program menu.

4.2 ltrapi

To write custom programs that interact with the ltrd service, crates and/or modules, a set of libraries is provided in the C language. The interface of the cross-platform libraries is fully compatible with the interface of the previously created version of the libraries under Windows OS. This makes it easy to transfer existing programs from under the Windows OS (of course, if the program itself does not use the Windows-specific API for its purposes) and write cross-platform software that runs under any of the supported OS.

The older only-for-Windows version correspond to libraries of versions 1.27 and below (which can be downloaded at http://en.lcard.ru/download/ltrdll_1_27.exe with the old description http://en.lcard.ru/download/ltrapi_v1.pdf). This version is currently not supported, and all systems use shared libraries of newer versions.

The set of libraries consists of a base library with the same name ltrapi, in which functions are implemented for creating client connections with ltrd, as well as general functions of control connections. A detailed description of the basic library, including general principles of operation, a description of all functions and definitions, can be found in the corresponding programmer's manual: http://en.lcard.ru/download/ltrapi.pdf.

To operate each LTR module, separate libraries with the names ltrXXXapi are created, where XXX corresponds to the module number. These libraries use within themselves functions from the core library to establish a client connection and data transfer. Therefore, specifically for the task of operating a particular module that explicit use of the base library functions is not required. The corresponding libraries are described in the documents ltrXXXapi.pdf, which can be downloaded from the "File Library" or on the page of the corresponding module in the "Documentation" section.

The connection of the libraries for Windows OS is described in the document "Starting operating the LTR crate system. Software issues".

For Linux, FreeBSD, and QNX6 OS, the ltrapi library is a collection of shared libraries (.so files) that must be installed along with the program. In this case, the separation to the file of the library itself, the name of which includes the full version and which is necessary for the operation of the program using the library, and to the symbolic link with no version to the first file, that is used for compilation. This is done for the potential to have multiple versions of the library on the same machine as needed.

For QNX4, static libraries (with the .lib extension) are provided, which are included in the program code when compiled.

The differences in the behavior of libraries are described below.

4.3 ltrctl

The ltrctl utility provides the user with a command-line interface for ltrd control and receiving the necessary information from it. To perform its actions ltrctl uses the ltrapi functions.

All the actions that need to be performed are specified using commands, each of which can have its own list of arguments. ltrctl can operate in two modes:

- You can specify a command with arguments as parameters when running ltrctl. In this case, ltrctl will connect to ltrd, execute the actions corresponding to the command, then disconnect and exit. For example, to display a list of ip-addresses stored in ltrd, you can write in the command line:

```
ltrctl iplist
```

- If the command is not specified in the ltrctl parameters, then ltrctl establishes a connection with ltrd, and then enters the command input mode. The user can enter the desired command and press the "Enter" key. After executing the command, ltrd will prompt you to enter a new one (and so on, before the exit command is not entered). All commands use the same connection.

A full list of supported commands with a brief description can be received by calling:

```
ltrctl --help
```

The syntax of the commands is described by the following rules:

- Arguments in triangular brackets mean that the actual values of the parameter are substituted instead of them, while words without triangular brackets are used as they are. So "ipcon <ip_addr>" means that <ip_addr> should be replaced with a real address, for example "ipcon 192.168.1.4".

- Arguments in square brackets are optional. For example, "ipadd <ip_addr> [auto] [perm]" means that the correct options are "ipadd <ip_addr>", "ipadd <ip_addr> auto", "ipadd <ip_addr> perm" or "ipadd <ip_addr> auto perm".

- A vertical line between two words means that one of two words can be used. For example, "ipcon <ip_addr> | auto" means that the actual option of the call is either" ipcon <ip_addr>" or "ipcon auto".

Some of the ltrd settings can be changed either for the duration of the current session (in this case, the changes will be reset the next time you start ltrd), or can be immediately saved to the configuration file. To save to a file, you must specify the perm argument (from permanently).

ltrctl by default establishes a connection with ltrd, running on the same computer. However, if desired, you can also connect to the remote ltrd by specifying the -ip-addr key when calling ltrctl. For example, to connect to ltrd, running on a machine with the address 192.168.1.10, you need to call:

```
ltrctl --ip-addr=192.168.1.10
```

4.3.1 IP address control commands

- Output of information on all IP-address entries:

```
ltrctl iplist
```

For each IP-entry, its address, status (whether a connection of the crate to this address is established), the serial number of the corresponding crate (if connection is established) are displayed as well as whether the auto-connection flag and the reconnection flag are set.

- View the address list by specified mask:

```
ltrctl iplist <ip_addr> <msk>
```

The same information is displayed as by the previous command, but only for addresses that match with the <ip_addr> address when putting the <msk> mask. For example, you can use it to display addresses only from a specific subnet.

- Adding an IP address only for the current session:

```
ltrctl ipadd <ip_addr>
```

- Adding an IP address only for the duration of the current session with the auto-connection flag set:

```
ltrctl ipadd <ip_addr> auto
```

- Adding an IP address for the duration of the current session only with the reconnection flag set:

```
ltrctl ipadd <ip_addr> reconnect
```

- Adding an IP address with saving to a configuration file:

```
ltrctl ipadd <ip_addr> perm
```

- Adding an IP address with the auto-connection flag set with saving to the configuration file:

```
ltrctl ipadd <ip_addr> auto perm
```

- Adding an IP address with the reconnection flag set with saving to the configuration file:

```
ltrctl ipadd <ip_addr> reconnect perm
```

- Deleting an entry with the specified IP address only for the current session:

```
ltrctl iprem <ip_addr>
```

A client connection must not be established with the crate.

- Deleting an entry with the specified IP address with saving the change to the file:

```
ltrctl iprem <ip_addr> perm
```

A client connection must not be established with the crate.

- Setting the auto-connection flag for an already-added entry with the IP address with saving in the configuration file (without perm - only for the session):

```
ltrctl ipflag <ip_addr> auto set perm
```

- Reset the auto-connection flag for an already-added entry with the IP address with saving in the configuration file (without perm-only for the session):

```
ltrctl ipflag <ip_addr> auto clear perm
```

- Setting the reconnect flag for an entry already added with an IP address saved in the configuration file (without perm - only for the session):

```
ltrctl ipflag <ip_addr> reconnect set perm
```

- Reset the auto-connection flag for an already-added entry with the IP address with saving in the configuration file (without perm-only for the session):

```
ltrctl ipflag <ip_addr> reconnect clear perm
```

4.3.2 IP addresses connections control commands

- Starting connection at the specified IP address.

```
ltrctl ipcon <ip_addr>
```

The address must already have been added before using ipadd. Check when the connection is completed and with what result by the status of the IP address using the iplist command

- Establish a connection with all addresses that are marked with an auto-connection flag

```
ltrctl ipcon auto
```

The connection is maintained with the crates that are already connected. If the connection is done, it is restarted.

- Disconnect from the specified IP address.

```
ltrctl ipdisc <ip_addr>
```

- Disconnect from all crates that has a connection or are in the process of establishing a connection

```
ltrctl ipdisc all
```

4.3.3 Receiving information on active crates and modules

- Output of the list of all active crates with the modules set:

```
ltrctl clist
```

- Output of statistics for the first active crate

```
ltrctl cstat
```

- Output of statistics for the active crate with the specified serial number

```
ltrctl cstat <serial>
```

- Output of statistics on the module in the specified slot (starting from 1) in the first found crate.

```
ltrctl mstat <slot>
```

<slot> — number from 1 to 16

- Output of statistics on the module in the specified slot (starting from 1), located in the crate with the specified serial number.

```
ltrctl mstat <slot> <serial>
```

<slot> — number from 1 to 16

4.3.4 ltrd parametric control

- To set the log level with saving to a file (without perm - for the current session):

```
ltrctl setloglvl <lvl> perm
```

<lvl> is a number from 0 to 7, indicating the level, messages with a level above which (with a lower priority) will not be output. Level 0 corresponds to the output of only critical error messages, 7 - the output of all messages, including debugging messages. Typical level 3 - output of the main messages on operation progress and all warnings/errors.

- Receiving the current log level

```
ltrctl getloglvl <lvl>
```

4.3.5 Additional commands

- Reset the module in the specified slot in the first found crate:

```
ltrctl mrst <slot>
```

A reset command is sent to the module and all statistics are reset to zero.

- Reset the module in the specified slot in the crate with the specified serial number:

```
ltrctl mrst <slot> <serial>
```

- Restarting ltrd with a break of all connections

```
ltrctl restart
```

- Request for ltrd shutdown.

```
ltrctl shutdown
```

4.4 LTR Manager

LTR Manager is a graphical utility for controlling ltrd and outputting information about the connected crates.

LTR Manager is not available for all supported OS, but you can run this program from a computer running Linux or Windows OS to configure and monitor remote (running on a machine with another OS) ltrd daemon.

Figure 4.1 provides a screenshot of the LTR Manager graphical interface.

The main differences between LTR Manager and the previously used LTR Server:

- LTR Manager is only an interface to the ltrd service and serves to view statistics and control. When you exit the LTR Manager, the ltrd service continues to work and all connections are saved. Also for operating crates, the run of LTR Manager is optional.

- LTR Manager can be connected with remote ltrd and even with several ltrd simultaneously.

- LTR Manager allows you to perform some additional service operations (setting up the crate, updating the firmware, resetting the modules, controlling crate synchronization, etc.)

The interface represents the main window and a set of toolbars that can be placed in the desired place of the main window, rendered as separate windows or hidden at the user's request (visibility can be controlled either from the "Window" menu or from the toolbar).

The main window shows a tree in which for each ltrd with a established connection, a list of active crates with the modules set is shown. For each crate there is an icon

corresponding to the interface on which the connection is established with the crate. The content of the remaining panels depends on the selected item in this tree. Each node can be minimized as desired.

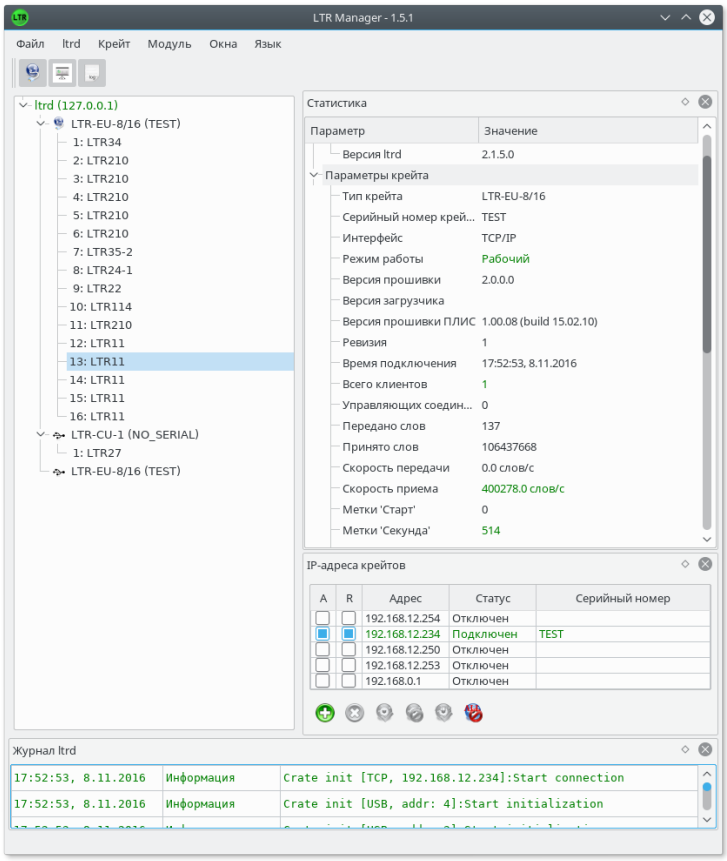

Fig. 4.1: The interface of the LTR Manager program running under the KDE5 desktop

The statistics panel shows information on the ltrd connection dedicated in the main window to the crate and the module. When you hover the cursor on the desired parameter and hold it, a tooltip appears that describes in more detail what the given parameter means.

The "IP addresses" panel displays a table with all the IP addresses of the crates for the chosen ltrd connection and allows performing all available actions with the IP addresses described in section 4.1.4. When you hover the mouse over the column header, a tooltip with a column description pops up. To perform an action, you can select the desired line with the IP address (if the action refers to a specific entry) and click the button corresponding to the desired action below the table (the name of the action is displayed in the tooltip when you hover the mouse over the button). You can also do this by right-clicking on the desired entry for an action pertaining to a specific entry, or for any place in the table for common actions and selecting the desired action from the context menu. The value of the "Auto-connection" and "Reconnection on error" flags can be changed by unchecking or checking the "A" or "R" column, respectively. Double left-clicking on the line with the IP address changes the connection status at the given address (leads to starting the connection to the address if there is no connection, and to the break if the connection is established or the connection process is in progress).

The "ltrd log" panel displays the messages of the ltrd log corresponding to the selected ltrd. The log message level is indicated by the color of the message. Through the context menu (right click of the mouse), you can clear the log messages received by LTR Manager and change the level of displayed messages.

Important! Note that there are two log level configurations. One of them is the ltrd configuration, which determines which level of messages the ltrd service will send (the level is taken into account at the time the message occurs, and lower-level messages will be discarded). The second is the LTR Manager configuration, which determines which messages from the received (already filtered by the first level) from ltrd will be displayed. In this case, changing the level results in a change in the composition of the messages that have already been displayed. Naturally, setting the log level in LTR Manager higher (for resolving less critical messages) than ltrd configuration, will not result in additional messages.

When the main window is closed, LTR Manager collapses and remains in the tray (this corresponds to the usual operating mode of programs that can be minimized to the tray). For complete closing, you need to select "File -> Close" in the menu.

4.4.1 Configuring LTR-EU/LTR-CEU crates interface

The LTR Manager program allows you to configure the operating interface and its parameters for crates that support the Ethernet interface (LTR-EU, LTR-CEU). For the LTR-EU, the crate must be connected via USB, regardless of the interface used (if the crate is configured for USB, all modules will be visible, if for Ethernet the modules will not be displayed, and the crate state will be "Setup Only"). For LTR-CEU crates, it is also possible to change settings when connected via Ethernet.

To change the settings, you must right-click on the desired crate and select the menu item "Crate Settings...", then the dialog box shown in Figure 4.2 will appear. In this window, the current crate settings written in the non-volatile memory of the crate.

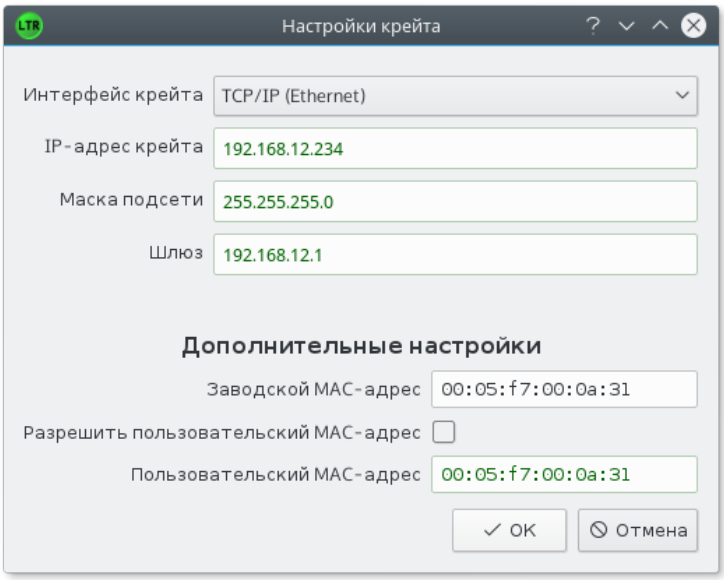

Fig. 4.2: Crate interface configuration dialog

You need to select the desired interface and for TCP/IP select the crate IPv4 address, the network mask and the gateway address, then click "OK" and confirm the settings entry. Real settings will only be enabled after the crate has been reset. For LTR-EU, this must be done either by pressing the reset button on the front panel or by turning off the power switch, and the message will be displayed after the settings are recorded. For LTR-CEU, there is a software reset command and the LTR Manger can perform the reset of the crate by itself after changing the settings (this will be offered).

For LTR-CEU crates, changing settings can also be protected with a simple password. If the password is not set, then when you change the settings, you can enter nothing in the request and immediately click "Ok". This is done so that you can protect the crate from changing settings remotely, since LTR-CEU supports changing them not only via USB. In this case, if the password is forgotten, you can change the settings via the USB interface by entering the serial number of the crate as the password.

It should be noted, that when entering configuration dialog, LTR Manager reads the current settings in the memory of the crate (which may not yet take effect if there has not been a crate reset since the last modification) and displays them in the dialog fields, that is, that can be used to view the current settings of the crate without changing them.

4.4.2 Crates synchronization settings control

Since version 1.5.0, the ability to control the synchronization (generation of sync-tags) for the LTR-EU, LTR-CEU and LTR-CU crates was added to the program.

This mechanism is described in the chapter "Principle of synchronization of data collection in the LTR system" in the LTR user manual.

To configure sync-tags generation, you need to right-click on the desired crate and select the menu item "Crate synchronization control...", as a result the dialog box shown in Figure 4.3. will appear.

In this window, the same sync-tags generation control settings are available, which are also provided by the ltrapi functions. These settings can be performed automatically by finished software that controls synchronous data collection from multiple modules. However, for those cases when it may be convenient to manually configure the sync-tags generation (if these functions are not included in the finished software, it is necessary to control the synchronization of the modules, when they are operated by individual programs, when developing your own programs, etc.), such opportunity is provided additionally through this dialog.

Opening this dialog does not prohibit further operation with LTR Manger and, if necessary, you can open several such dialog boxes, each for your crate.

It should be taken into account that since the crate's firmware allows only to set sync-tags generating settings, but does not allow reading them, while these settings can be set in parallel from other programs, it is not possible to display the current settings of the crate. Thus, the displayed values when opening the dialog are not related to the real settings of the crate.

This dialog allows to make the following settings:

- To set which signals will be broadcast to the DIGOUT1 and DIGOUT2 outputs of the synchronization connector. To do this, fill out the parameters in the "SYNC connector outputs" section and click the "Set" button in this section.

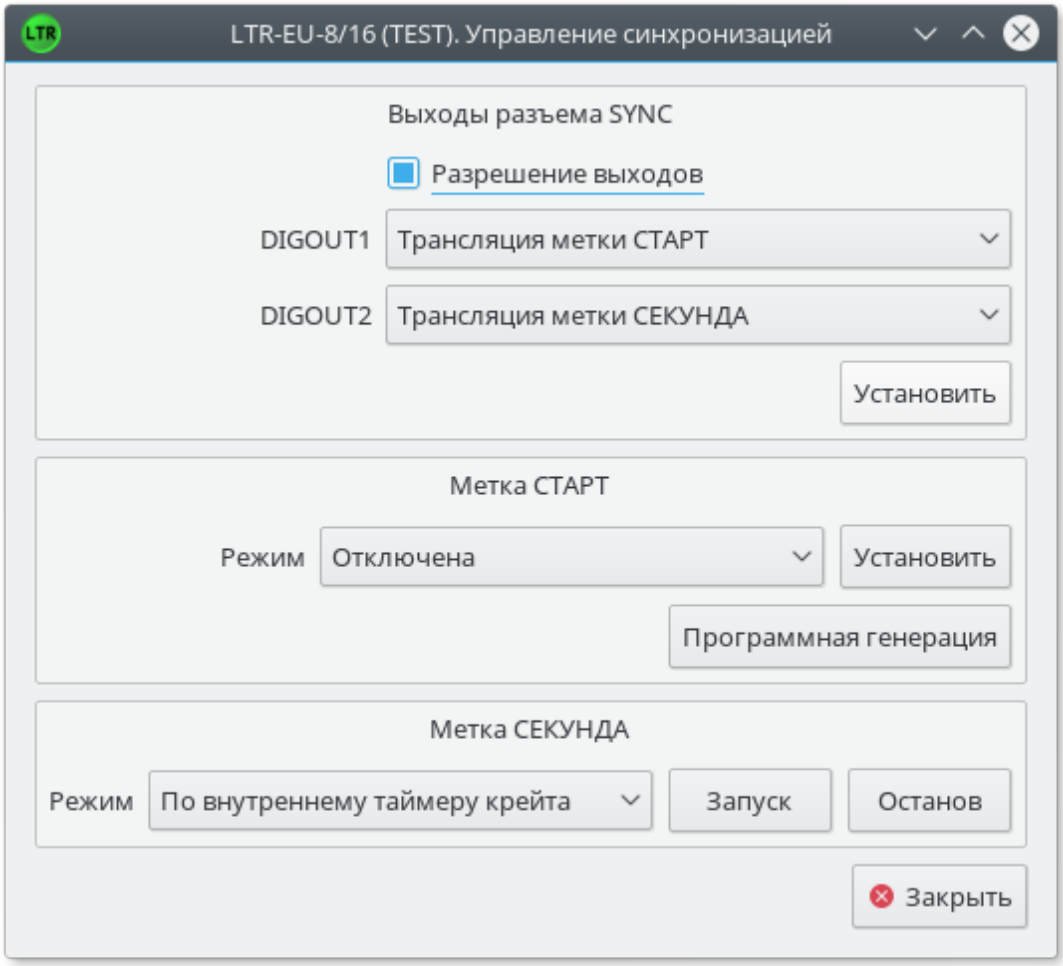

Fig. 4.3: Crates synchronization control dialog

If "Output permission" is not checked, the outputs will always be in the third state, otherwise they will be receiving the signal in accordance with the function selected for each output. In particular, this can be used to broadcast sync-tags that are generated by crate. Accordingly, the signals from these outputs can be used as sources of sync-tags for other crates.

- To set the generation mode of the "START" tag from an external source. To do this, select the mode of sync-tags generation in the section "START tag" and press the "Set" button to the right of the mode. After that, the crate will wait for the specified event and for each such event generate the "START" tag. If the tag generation through an external event is no longer needed, then select the "Disabled" mode and click "Set".

- Perform manual generation of a single "START" tag. By clicking the "Software generation" button in the "START tag" section, the crate will generate one START tag (regardless of the current settings), after which the crate will stop waiting any events of generation of the "START" tag, i.e. will go into the "Disabled" mode, regardless of what was set up before.

- Start the generation of a second tag in one of the modes. To do this, you need to set the desired mode in the "SEC tag" section and click "Start". If the mode "By internal crate timer" is set, the crate will immediately generate a tag once per second. If a different mode is selected, the crate begins to wait for the corresponding external event and generate a "SEC" tag for each event of the selected type.

- To stop the generation of the sec tag. To do this, click the "Stop" button in the "SEC tag" section.

4.4.3 Processor firmware update

The LTR Manager program allows to update the processor firmware for LTR-EU (USB only) and LTR-CEU crates (both USB and Ethernet). To do this, right-click on the desired crate and select the menu item "Crate firmware update". After that you need to select a file with the processor firmware and it will be loaded. For LTR-EU crates after that, you must also manually reset the crate (as with the settings update), and for LTR-CEU crates the firmware will take effect immediately after the update.

The latest firmware versions can be downloaded from the corresponding crate page on the website in the section "Software" -> "Firmware and BIOS".

Important! For LTR-EU crates, it should be explicitly considered that the firmware for LTREU-2 differs from the LTR-EU-8/16 firmwares. It is necessary to make sure that the firmware to be written corresponds to the type of crate, as unlike LTR-CEU, there is no explicit protection against writing an incorrect crate firmware.

Important! When the LTR-CEU firmware is updated via USB, the device goes into bootloader mode (and is seen as a "Boot device" rather than a crate), which remains until the update is completed. Also, if there were no calls for the specified time, the crate returns to the operating mode. In Windows OS when updated for the first time, a driver for the boot device may be requested and its updating may take longer than waiting for the device to appear in the LTR Manger and then the firmware will fail. In this case, you need to install the driver, wait until the crate returns to the operating mode (appears in LTR Manger) and retry the update. Under Linux OS operating with a boot device on behalf of an unprivileged user requires its udev rules, which are installed with the lboot utility automatically when installing the package that is used in this case for updating.

4.4.4 Operating remote ltrd

By default, LTR Manager connects at startup to the local ltrd daemon (marked with the address "127.0.0.1"), which is what required in many cases. However, it is possible to control ltrd from one GUI running on other machines. You can add the addresses of the remote instances of ltrd, much like the addition of crate addresses to ltrd. To do this, use the menu "ltrd->Add a new connection with ltrd". In the menu that appears, select the "Remote connection" flag and enter the address. As for crate IP addresses,

you can specify whether you need to connect to ltrd at this address automatically at the LTR Manger next startup.

Note: Please note that the ltrd address list is the setting for the LTR Manager program itself, while the crate address list is the setting for each ltrd and LTR Manager provides only an interface for modifying them.

The new connection will appear as a new top-level node in the tree displaying the list of crates and modules (each crate belongs to the ltrd node corresponding to which ltrd instance supports the active connection to this crate).

When operating several ltrd, please note that the information displayed in the "Crate IP addresses" panel and the "ltrd" menu items refer specifically to the currently dedicated ltrd connection.

4.5 Additional utilities

4.5.1 LTR-EU crates firmware updating and configuring utilities

The command line utilities ltreu-config and ltreu-firmupdate were integrated in ltr_cross_sdk to set the interface configuration and update the BlackFin or FPGA firmware, respectively. To perform these operations, it is required that the crate is connected to the PC via USB.

In this case, even when the crate is configured to operate on Ethernet, it is possible to connect it via USB interface to perform operations of configuring and updating the firmware.

Also, these features are available from the graphical interface of the LTR Manager.

It is important to note that unlike previously used utilities under Windows OS, these utilities work through ltrd, and not directly with the crate, which allows not to stop the ltrd operation. If necessary, both utilities can even work with a remote running ltrd, for this you need to pass the address of the machine on which ltrd is running, using the --srv-addr option.

Both programs by default connect to the first LTR-EU crate, which is connected via USB. However, it is possible to explicitly specify the desired crate by serial number using the option --csn.

After successful operations both utilities require, to remove and reconnect the power to the crate for the changes to take effect.

The main use cases are described below.

4.5.1.1 Setting interface configuration

To set the interface configuration, the ltreu-config utility is used.

To set, you must specify the interface on which the crate should work, using the option --intf or --iface (these are just two names for the same option). Available interfaces:

- usb — operating the crate via USB-interface
- eth — operating the crate via Ethernet interface using the protocol over TCP/IP.

For the USB interface, you do not need to specify anything else, while for Ethernet it is required to specify the IP address of the crate itself (the -ip-addr option), the subnet mask (-ip-mask option) and the gateway address in the network (option --gate).

Thus, setting the crate to operate via the USB interface is as follows:

```
ltreu-config --intf=usb
```

To set the crate to operate on the Ethernet interface with the crate address 192.168.1.10, mask 255.255.255.0 and the gateway address 192.168.1.1, you need to call:

```
ltreu-config --intf=eth --ip-addr=192.168.1.10 --ip-
                mask=255.255.255.0

                --gate=192.168.1.1
```

In order for the changes to take effect, you need to remove the power of the crate and reconnect it.

4.5.1.2 Reading current interface configuration

Calling the ltreu-config utility without the --intf or --iface option causes the current module settings to be read and displayed on the screen.

4.5.1.3 Crate firmware update

Ltrau-firm-update utility is used to update the firmware. It allows you to write a new firmware into the crate for both the BlackFin signal processor (indicated by the -dsp-firm option) and for the FPGA (option -fpga-firm). What will be written is determined by the presence of the specified options (you can specify both BlackFin and FPGA for updating and writing).

Important! It is important to keep in mind that the firmware is written in the crate, designed specifically for the type of crate that is being updated. Otherwise, the crate can become inoperative!

For additional checking the type of the crate, you must explicitly specify the type of crate-controller for which the firmware is intended, using the --crate-type option. Possible variants of crate-controller type:

- ltr030 — for the LTR-EU-8/16 crate
- ltr031 — for the LTR-EU-2 crate

The names of operating firmwares always begin with the name of the crate-controller for which they are intended.

It should also be noted the presence of the option --all, at which ltreu-firm-update performs a serial update of the firmware for all the crates of the specified type, connected via the USB interface.

When installing utilities from packages, you can use the ltr030-firm-update-all.sh and ltr031-firm-updata-all.sh scripts to update the latest firmware of all LTR-EU-8/16 or LTR-EU-2 crates, respectively.