

Evertz Router Controller

EVERTZ_CTL.exe

Version 6.3.2

Written by Paul Wilkins

Contents

Contents	2
1 Overview	3
1.1 Description	3
1.2 BNCS configuration	3
1.3 CSI version	3
2 Driver setup.....	4
2.1 Introduction	4
2.2 [Controller] Section.....	5
2.3 [Routers] Section	6
2.4 [Locked_Dests_Router_a] Sections	7
2.5 [MultiRouting] Section	7
2.6 Mapped Sources and Destinations	9
2.7 Server Type.....	9
2.8 IP Connectivity	9
2.9 Park Source	10
2.10 Update SC1000 / Quartz Source or Destination names	10
2.11 Use of BNCS Infodriver for destination Lock Status	10
2.12 Driver Menu bar – Options tab.....	10
2.13 Sample Ini.....	10
3 Resilience and redundancy	11
4 Driver GUI design	12
5 Version history.....	13
5.1 Driver version	13
5.2 Document version	14

1 Overview

1.1 Description

This BNCS driver, EVERTZ_CTL.EXE, will control all Evertz / Quartz routers and their system controllers that support the multilevel protocol. This driver has definable options to control the latest Evertz EQX, EMR routers directly or via the new SC2000 controller range. There is continued support for the older SC1000 controllers and original Quartz routers.

The EVERTZ_CTL, this is now the recommended version of Evertz and Quartz Controller driver because of this version's ability to use either serial or IP comms to control these routers, including the ability to use an infodriver for destination locking, driver resilience and overall flexibility.

1.2 BNCS configuration

One of the features of this driver is the choice of either Serial or IP connectivity. Serial Comms is default, for backward compatibility, though all the latest Evertz routers use TCP/IP connections now. There are a number of key settings to indicate the type of router or Controller being controlled via the driver – please read through section 2.2 *Configuration* for the required settings, especially if using IP.

1.3 CSI version

This version of the driver works with V3 CSI, V4 CSI and CSI32.exe. The latter is now the recommended version of CSI to use.

2 Driver setup

2.1 Introduction

When EVERTZ_CTL.EXE starts up the driver looks for either a CTL_QTZ.INI file or, if passed a numeric parameter on start-up, a DEV_XXX.INI file. If started for the very first time default entries for all configuration parameters will be entered into the appropriate INI file. If no routers are defined within the configuration the driver displays a message to that effect and then terminates.

The driver determines the BNCS environment it is running within – either a V4/4.5 configuration with a dedicated directory structure or a classic BNCS configuration with files stored in the Windows directory or specified directories as defined within the C:\BnCS_Config.ini file.

The entries within the ini file also include the mapping of Quartz level letters onto BNCS driver numbers. EVERTZ_CTL then searches for the DEV_XXX.INI files for each of these router drivers and extracts details of router size, together with all the database names. From BNCS all the routers appear as their respective driver numbers and as such the operation from BNCS panels is identical to using a GRD.

Setting the Simulation parameter to 1 in the ini file provides an effective router simulator for all defined routers. When not in simulate the driver communicates continuously with the routers / controllers requesting tally status for all of the configured levels. Any updates are sent to the network.

To facilitate testing in “simulation”, the driver will run up in this mode if “-sim” is added as a command line parameter after the device ini file number, e.g. *Evertz_ctl.exe 520 -sim*

The ini file has 4 sections :

[Controller] --- Driver configuration defining communication settings, lock and database options

[Routers] --- Individual router device assignments, sizes and offsets within Evertz hardware

Optional sections headers:

[Locked_Dests_Router_a] --- Predefined locked destinations for each level of router.

[MultiRouting] --- functionality to group destinations and sources for user defined salvos – allows for multiple routes to be executed together.

The following sections cover the ini file settings in detail.

2.2 [Controller] Section

This section covers all the communication parameters, locking and database update options available for this driver.

Item	Value	Comment
Name	Default: Evertz Controller	Helpful name to describe driver function. The entry is displayed on the title of the application window.
ParkSource	Default: 0	The real source that will be routed to a destination when a client sends a Router-Crosspoint (RC) command with a value of zero for the source.
ParkName	Default: --	The name associated with the park source in source list box
ForwardBufferDelay	Default: 0 ms	The delay in milliseconds between forwarding groups of commands to the controller. This prevents the controller receiving bursts of commands and potentially ignoring or losing them.
CrossPointsPerBufferDelay	Default: 1	This is the maximum number of commands buffered in the driver that should be sent to Quartz controller at any one time.
MaxUnACKedRoutesBeforeCommsFail	Default: 10	There is a delay between the controller receiving multiple commands, processing them and issuing acknowledgements. This setting allows the maximum number of outstanding unanswered commands before flagging an error.
SaveDelay	Default: 10 minutes	Interval of time between saves to disc of all the current router tallies. Router tallies are also stored to disc when the driver closes down.
TallyDelay	Default: 500 ms	Time interval for cyclic polling of all destinations to verify the driver is up to date and maintain active comms to hardware.
UseRCrosspointQueue	Default: 1 0 = Off 1 = On	When enabled multiple incoming BNCS RC commands (e.g. several RC commands within the same BNCS message packet) are aggregated into a single ".M" quartz protocol command and sent to the hardware – results in faster processing of routes by the hardware. Option menu item "Use RC command queue" can be used to turn this on or off.
DebugMode	Default: 0 0 = Off 1 = On	If set to 1, more debug messages be seen via DBWIN32.exe. Default is off. Can be changed by an option on the menu bar.
SourceToDest	Default: 1 0 = no 1 = yes	Sources routed to destinations. Leave as 1 for normal routing requirements.
DestToSource	Default: 0 0 = no 1 = yes	Leave as 0 for normal routing requirements.
NameLength	Default : 8	Defines the length of the source / destination names listed in gui listboxes.
Simulation	Default: 0 0 = Off 1 = On	To run this GRD driver without the need to control any actual hardware. Can use "-sim" as part of command line parameters too. (saves on having to edit the dev ini file)
Connection_Mode	Default: SERIAL 1=SERIAL , 2=IP	Defines the method of communication to hardware. Alphabetic (in capitals) or numeric entries are valid.
IP_Address_A	Default: 0.0.0.0	IP Comms parameters. Up to 2 SC1000/SC2000 Controllers (A & B) each with up to 2 IP connections. Any address other than 0.0.0.0 is treated as valid and the driver will attempt to connect to.
IP_Address_A2	Default: 0.0.0.0	
IP_Address_B	Default: 0.0.0.0	
IP_Address_B2	Default: 0.0.0.0	

Item	Value	Comment
IP_Port IP_Port_A2 all 4 addresses can IP_Port_B now use their own IP_Port_B2 port numbers	All ports Default to port 4000	Connection states are reported in driver gui. IP_Port – port number to connect to – see notes in section 2.6.
Server_Type	Default: SC1000 SC1000 EQX EMR XENON	"SC1000" – for all older "Quartz" routers and SC1000 controllers. When controlling any of the latest Evertz routers then set it to the router type, either "EQX", "EMR" "XENON". The "SC2000" setting is no longer explicitly required – and will be changed to "EQX".
Com	Default: 1	SERIAL Comms Parameters
Speed	Default: 38400	
DataBits	Default: 8	
StopBits	Default: 1	
Parity	Default: N	
PollForSC1000LockStatus	Default: 0	Set to 1 to poll the HARDWARE for destination lock status. This status can be reported via external infodriver if next parameter is assigned
LockStatusInfodriverID	Default: -1	Any valid BNCS device number 1..999, except that of the current router driver. See section 2.9 for further discussion on the different uses for the Lock Infodriver.
UpdateSC1000DatabaseNames	Default: 0 0 = no 1 = yes	Enable Router Modify (RM) commands from BNCS to update source or destination names within the Quartz/Evertz domain.
UpdateSC1000FromBNCS Databases	Default: 0,1	Updates of db names to SC1000, (if enabled) will only occur for BNCS databases 0=sources or 1=destinations. These are the databases that will be used to update source and destination names in the Evertz domain from BNCS if initiated from new Menu option too.
Access	ALL	Leave value as default setting

2.3 [Routers] Section

The ini file contains a list for mapping the Quartz/Evertz routers onto BNCS driver numbers. The list has entries from A to Z. The actual letters are merely a placeholder for each entry to make it easier to select the different routers from the driver's dialog box. Each entry consists of a comma-delimited list of three numbers. The first number is the BNCS driver Id that you wish to use for this router. The second is the first destination to poll for tallies and the third is the last destination to poll. This allows the polling range to be confined to a subset of destinations actually in use and overrides the settings in the DEV_xyz.INI file.

The list of entries for levels A to Z are:

Quartz Level 1 is Router_V,

Quartz Level 2 is Router_A,

Quartz Level 3 is Router_B, level 4 is D and so on through the alphabet

Entries for each Router_level in the ini file are:

Router_<level>=<Device Number>,<Min_Dest>,<Max_Dest>,<Offset -Sources>,<Offsets -Destinations>

In the following example the BNCS driver Id 101 polls the quartz video level between destinations 1 and 256, where any source and destination offsets are 0. These offset values by default are 0, and for backward compatibility, default values will be added to older ini files :

For an example using offsets:

Router_V=101,1,256,0,0
Router_A=102,1,256,256,256
Router_B=103,1,256,512,512

The Offset values in this example show the cumulative effect of offsets over several levels. To the hardware there is one large router 768 square, but from Colledia it is treated as 3 routers of 256.

2.4 [Locked_Dests_Router_a] Sections

The driver on first starting up can set specified destinations to be LOCKED. By default all destinations are unlocked when the driver starts up.

These lists in the DEV_xxx ini file or CTL_QTZ.ini file take the form of a section heading in the form

[Locked_Dests_Router_V]
Locked_01=1,2,3,4,5
Locked_02=101,102,103....
Locked_xx...

There should be a separate entry for each level of the router(s) defined in the [Routers] section.

Lists of destinations are used to keep the entries in the ini file compact, with each line denoted by an incrementing number as shown in the example, and can list as many or as few destinations as required.

2.5 [MultiRouting] Section

Destinations or sources may be grouped together so that multiple routes can be triggered from one single Router Crosspoint command. The data detailing any groupings uses BNCS database files – to enable RM commands to allow runtime changes to be reflected / picked up by an executing driver.

There should be a separate entry for each level of the router(s) for those requiring the multiple routing options:

[MultiRouting]
Router_V=126,2,3,0

The format of the entries are :

device index, source database, destination database and offset value

The device index needs to be that of the router number already defined for that level in the "Routers" section, along with two specified database files that will hold the grouping data. Generally the offset value will be 0. In an aid to cover the situation where the database files

for a given router are already in use for other purposes then this definition can specify an offset value that could be used e.g : if a router is 1024 square, and all the database files are used then it an offset of, say in this example, of 2000 works – so that db entries from 2001 to 3024 could be used for the two specified dbs.

Database file entries for defined groups consist of a comma delimited list of indices.
e.g 0001=1,2,3,4 will mean source or destination 1 is the master of a group containing index 1 and 2 and 3 and 4. To clear a grouping the entry can be changed to 0001=0001.

Database entry 0001=2,3,4 means the same as 0001=1,2,3,4 by the way – because the index in the db file is treated as the “master” src or dest for the defined group.

There is a maximum of 32 entries for any defined group.

The following rules are employed with respect to multiple routing:

Working with the example source group: 0011=11,13,15,17 dest group: 0001=1,2,3,4

1. If an RC command specifies the “master” destination as its destination then all the destinations defined in the group will be routed to and reverts generated.
 - (a) If the specified source is also the master of a group then each member of the source group will be routed to the equivalent destination member. So using our example : source 11 will be routed to dest 1, src 13 to dest 2, 15 to 3 and 17 to 4.
 - (b) If the specified source is a not a master of a group then this same source will be routed to all members of the destination group.
e.g. RC 123 12 1 will result in source 12 being routed to dests 1,2,3 and 4
2. If there are fewer sources defined in a group than destinations then the “last” source in its group will be routed to all remaining destinations:
e.g. if source group 0011=11,13 then 11 routed to dest 1, and source 13 routed to all other destinations in its group ie 2,3 and 4, using our example.
3. Routing to a destination that is not the master one, but is defined within a group is treated as a standard route and other group members are not changed.
4. Changing a grouping definition via a RM command will not impact any routes already made. The grouping is only active at the point of an appropriate RC command.

If multi routing is enabled, sources / destinations in the driver GUI list boxes will be marked with (m) to indicate the “master” index of a known group.

Ver 6.3.2 – When routing a “single” source to a defined multi group destination. The source is routed to all destinations within the group. Setting a value in the mask part of a RC command however will mean the single source is routed to the first (master) destination in the group and the source in the mask parameter will be routed to the other destinations of the group.

E.g. RC 552 411 410 – where dest 410 is a multi group of 410, 411, 412 and 413. In this command source 411 is routed to all destinations in the group.

RC 552 411 410 ‘12’ – where source 12 is added in the mask parameter will result in 411 to destination 410 and source 12 to 411,412 and 413.

2.6 Mapped Sources and Destinations

The driver can read in user defined source and destination mappings for routers where a simple offset cannot be used and requires a more complex means of mapping BNCS sources / dests to actual hardware indices.

The driver for each defined router, during start-up, will look for an load specific mapping files. These should live in the relevant system path for the same device ini files.

Sources file : Dev_xxx.sourcemapping, Destinations file : Dev_xxx.destinationmapping

Where xxx is the router number.

These files take the form of <bncs index> = <hardware index [, optional comment]

[Mapping]

0001=24, optional comment about index

0002=23

0003=22

0004=21 etc - for all router indices.

The Driver GUI will indicate if Mapped data has been found, and when selecting a destination from the listbox will show both the BNCS value and defined hardware value for destinations and its current routed source.

Note – If such a mapping is used – then ALL sources and destinations need an entry defined in the two files for the relevant router.

Note – Mappings cannot be used if a simple offset has already been defined in the Routers section.

2.7 Server Type

This driver works with all existing Quartz and the latest Evertz EQX router range. For older routers and SC1000 controllers set the server type to the default "SC1000" setting, allocating the routers to the relevant router level in the [routers] section.

Current Evertz routers should be set to "EQX". If using a SC2000 then set this entry in the ini file to "SC2000".

2.8 IP Connectivity

If using Ethernet to connect to the hardware, set the Connection_Mode setting to "IP", and enter at least one valid IP address.

The IP_Port number is not recommended to be 23/25 (telnet connections). It is advised to use another port number eg 4000 or 4001 – this can be configured in the Quartz WinSetup software.

It has been found that using port 4000, say, that the SC-1000 will only let a valid connection be made to the main controller.

When multiple IP addresses are assigned the driver has been programmed to determine which controller is the main or live one.

This driver will work with SC1000s configured to use *virtual IP* Addressing. Simply set the entry of **IP_Address_A** as the IP address in the dev ini or ctl_qtz.ini file. **Note:** When the SC1000 hardware swaps between controllers in this virtual mode – the driver will lose connection for about **30 secs** as the hardware reboots both controllers and then when the hardware settles down the driver connection will be re-established.

2.9 Park Source

This driver shows the first source for each router as being a park source (- -). The value of the park source is configurable within the ini file, it's default is 0.

2.10 Update SC1000 / Quartz Source or Destination names

Setting the ini file option "UpdateSC1000DatabaseNames" to 1, enables the driver to send router source or destination name updates from BNCS Router modify commands to the SC1000. This will then change the name in the Quartz / Q-link domain.

Updates to the SC1000 will only be forwarded to the SC1000 for RM commands related to BNCS database 0 (sources) and database 1 (destinations) – or the two databases defined by the value assigned to the ini file parameter "UpdateSC1000FromBNCS Databases".

2.11 Use of BNCS Infodriver for destination Lock Status

LockStatusInfodriverID can be assigned a BNCS infodriver device number for the driver to hook into. This infodriver must be running on the same workstation as this driver and have been started before the Evertz_Ctl executable is run up. Each slot of the infodriver maps to a destination according to the Router Level Offsets discussed above – if a router has 576 destinations with an offset of 0, then the first 576 slots will report the BNCS lock status of each destination.

When the ini file parameter PollForSC1000LockStatus is set to 0 then the lock infodriver is used solely for BNCS locking. Setting this option to 1, enables the driver to poll the Evertz routers for their destination lock status. This poll is carried out during the background poll for the destination tally.

A zero in a slot means the destination is UNLOCKED. A value between 1 and 255 indicates it is LOCKED.

2.12 Driver Menu bar – Options tab

There is a list of options that can be "actioned" or enabled / disabled from the Options tab on the driver GUI. Depending on the Evertz router type being controlled some options may be disabled as they are not applicable.

A new option is the "Use RC command queue". This toggles the on/off state for the use of the Router Crosspoint command queue. When this option is enabled then multiple incoming BNCS RC commands (e.g. several RC commands within the same BNCS message packet) are aggregated into a single ".M" quartz protocol command which is then sent to the hardware making for faster processing of routes by the hardware.

When enabled the "RCQue" box on the driver GUI will show the value "15" ms. This slight delay allows for the incoming BNCS commands to be queued and aggregated.

2.13 Sample Ini

[Controller]

```
Name=Quartz Controller
ParkSource=0
ParkName=- -
ForwardBufferDelay=0
CrossPointsPerBufferDelay=1
MaxUnACKedRoutesBeforeCommsFail=10
SaveDelay=10
TallyDelay=500
```

```

DebugMode=0
SourceToDest=1
DestToSource=0
NameLength=8
Simulation=0
Connection_Mode=IP
IP_Address_A=1.2.3.4
IP_Address_A2=0.0.0.0
IP_Address_B=1.2.3.5
IP_Address_B2=0.0.0.0
IP_Port=4000
IP_Port_A2=4000
IP_Port_B=4000
IP_Port_B2=4000
Server_Type=EQX
Com=2
Speed=38400
DataBits=8
StopBits=1
Parity=N
PollForSC1000LockStatus=0
LockStatusInfodriverID=-1
UpdateSC1000DatabaseNames=0
UpdateSC1000FromBNCS Databases=0,1
UseRCrosspointQueue=1
Access=ALL

```

[Routers]

```

Router_A=Router,Min_Dest,Max_Dest,Offset_Src,Offset_Dest
Router_B=Router,Min_Dest,Max_Dest,Offset_Src,Offset_Dest
...
Router_U=Router,Min_Dest,Max_Dest,Offset_Src,Offset_Dest
Router_V=121,1,576,0,0
...
Router_Z=Router,Min_Dest,Max_Dest,Offset_Src,Offset_Dest

```

[Locked_Dests_Router_A]

```

Locked_01=
[Locked_Dests_Router_B]
Locked_01=
...
[Locked_Dests_Router_V]
Locked_01=1,15,21,22,23 - example of destinations to be locked on start-up
...
[Locked_Dests_Router_Z]
Locked_01=

```

[MultiRouting]

```

Router_V=121,2,3,0

```

3 Resilience and redundancy

This driver will run in dual driver redundancy mode – where the first driver to run up will become the main driver in TX/RX mode and any other instances will run in RX-only mode.

Should the main driver shutdown, a reserve one will then become TX RX and run as the main driver. Please note – that the driver will revert to RX-Only if comms is lost to the hardware. The driver will attempt to return to TX-RX when comms is later restored.

In serial or IP comms mode – if the driver loses connection to the hardware, the driver will drop into RX-Only mode, and revert back to TX-RX when valid comms have been restored.

4 Driver GUI design

Driver CSI Mode :
TXRX = main driver
RXONLY = reserve

ROUTER type being controlled,

Driver comms mode :
IP or Serial connectivity

Serial Comms status

IP Addresses and port numbers

IP Connection State

BNCS Lock infodriver number and Status if defined in ini file

Current source (ANS)
Current Evertz lock state (LOCK)
EMR Redundancy state
(EMR routers only)

Destination being polled and tally information received from hardware relating to

Router information as read in from ini file. In this example one router on the V level 576 sources and 576 destinations

Driver Critical messages displayed here

Source and Destination listboxes enabling users to see what is routed to a chosen destination, make routes and change the lock status of any destination.

There is also now the option to enable/disable debug messages from the menu bar – any messages can then be seen by using DBWIN32 (or any Windows Debug message viewer).

5 Version history

5.1 Driver version

The version history still holds that of the preceding V4CtlQtz driver, as this new Evertz_Ctl is derived directly from the SC1000 code.

Version	Date	Notes	Modified By
4.4.0.1	12/02/08	Initial Release	Paul Wilkins
4.4.0.6	17/12/08	Updated to use CTL_QTZ.ini – in line with, now old V3CtlQtz driver.	Paul Wilkins
4.4.4.50	23/02/09	Corrections for Router Modify Command responses. Corrections for TX-RX resilience issues. Corrections for config file default settings.	Paul Wilkins
4.4.9.24	19/05/2009	Approved released version for Local Radio / BBC use	Paul Wilkins
5.1.0.15	16/03/2009	Fix to Windows File Version properties. Serial comms loss no longer disconnects from CSI. Addition of Updating SC1000 source / destination names. Addition of polling SC1000 regarding destination lock state.	Paul Wilkins
6.0.0.10	01/05/2010	Evertz_Ctl created from SC1000 driver, to handle all Evertz and Quartz routers and controllers including new EQX range.	Paul Wilkins
6.1.0.24	01/12/2010	Revised and enhanced driver to work with SC2000 controllers. Using up to 4 IP connections for dual controllers. Improved driver redundancy and TXRX failover	Paul Wilkins
6.1.5.1	01/01/2011	Added Menu option to load all source and dest names to Evertz	Paul Wilkins
6.1.7.1	11/01/2011	Addition of EMR and XENON router type and hardware and destination redundancy handling.	Paul Wilkins
6.1.7.777	25/02/2011	Updated comms processing for multiple answers from MV commands	Paul Wilkins
6.1.7.990	12/09/2011	fix for db name update to dest list box	Paul Wilkins
6.1.7.1000	18/11/2011	Accepts 'out of range' Poll and Query commands. The upper poll index returned is limited by DatabaseSize_1.	Paul Wilkins
6.1.8.102	27/02/2012	Additions to driver GUI	Paul Wilkins
6.2.0.49	20/11/2014	Addition of the Multi-routing option for grouping sources and destinations	Paul Wilkins
6.2.1.4	03/02/2015	Ability to define different port numbers for each IP address	Paul Wilkins
6.2.2.18	15/02/2015	Added the functionality to use user defined source and destination mappings.	Paul Wilkins
6.3.0.29	17/07/2015	Added option to group incoming BNCS RC messages into one data message to hardware using protocol ".M" command. Revised driver GUI too.	Paul Wilkins
6.3.1.1	20/08/2015	Added command line option "-sim" in order to run driver in simulation mode, quicker than having to edit device ini file.	Paul Wilkins
6.3.2.3	24/03/2017	Mask parameter can be used to alter routing of single sources to a multi group defined destination.	P.W.

5.2 Document version

Version	Date	Details	Name
1.01	12/09/08	New Template Added	Simon Armstrong
1.02	Feb 09	Updated template	Amanda Atkin
1.03	Feb 09	Updated for configuration settings	Paul Wilkins
1.04	Mar 09	Updated Gui for latest driver version	Paul Wilkins
1.05	Jul 09	Updated for Website release	Paul Wilkins
1.06	March 2010	Updated for version 5.1	Paul Wilkins
1.07	May 2010	Revised for enhanced Evertz EQX routers	Paul Wilkins
1.08	January 2011	Updated for EMR routers and SC2000 controllers	Paul Wilkins
1.09	November 2011	Version info updates	Steve Lowe
1.10	May 2012	Version info update	Paul Wilkins
1.12	Feb 2015	Updated docs for multi routing and IP port numbers	Paul Wilkins
1.13	Feb 2015	Updated docs for added mapping functionality	Paul Wilkins
1.14	July 2015	Revised documentation to reflect driver changes for aggregating RC commands and GUI look revision.	Paul Wilkins
1.15	Aug 2015	Updated docs for "-sim" command line option	Paul Wilkins
1.16	Mar 2017	Additional use of mask in RC commands for multi group	P.W.