Template:FMX125 RS232/485

□

Contents

RS-232 / RS-485 modes

Available modes for RS-232 and RS-485 interfaces:

Interface	Mode															
	Log Mode	NMEA	LLS	LCD	RFID HID	RFID MF7	Garmin FMI	TCP/UDP Ascii	TCP/UDP Binary	TCP/UDP Ascii Buffered	TCP/UDP Binary Buffered	Mercury C4	ULF202 Fuel Sensor	DualCam	ADAS	Delimiter
RS-232	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□
RS-485	□	□	□					□	□	□[1]	□[1]					

[1] – available from FW version **03.28.XX**.

Do not under any circumstances connect RS485 A and B lines (or RS232 Tx and Rx) to power source.

Swapping RS485 A and B lines (or RS232 Tx and Rx) of the external device and also not connecting common ground first - May cause irreparable damage.

Log mode

In this mode via RS-232 or RS-485 interface prints the FMX125 device log and does not respond to commands.

GNSS NMEA mode

In NMEA mode via RS-232 or RS-485 interface FMX125 prints GNSS NMEA log and does not respond to commands.

LLS mode

In LLS mode RS-232 supports one and RS-485 supports up to five LLS sensors - each of which has a receiver ID.
Up to 16 LLS sensors are supported with **03.28.XX** firmware version. Up to 8 passenger counter sensors "PP-01" can be connected each requiring 2 LLS addresses. The passenger counter sensor "PP-01" provides the possibility to count passengers getting either on or off public transport (bus, trolleybus, etc.).

TCP (ASCII/Binary) modes

In *TCP Ascii/Binary* mode all data received from the external device is sent directly to the server. Data is encapsulated in codec 12 format. *TCP Binary* Mode has a delay of 30 ms, if no data is received for 30 ms, data is sent to the server. *TCP Ascii* mode waits for the End of Line (EOL) character (0x0D0A, \r\n, <CR><LF>) to pack data and send it to the server.

TCP Ascii Buffered and *TCP Binary Buffered* modes are used to collect data from RS232 and save it in the buffer if there is no link with the server and data cannot be sent immediately. When the link is established and there is data to transmit, then RS232 data from the buffer is transmitted after all records are sent. Data is sent in codec 13 protocol. Note. That in *TCP Ascii* and *TCP Binary* modes device sends data from the external device only to the main server. In Buffered modes – to both main and backup/duplicate servers.

	TCP Binary/TCP ASCII mode	**TCP Binary/TCP ASCII Buffered mode**
Data is saved in buffer	No	Yes
Data sending protocol	Codec 12	Codec 13*
Timestamp	Not using	Is using*
To which server is sent	Main	Main and backup

**If the Timestamp parameter is enabled, then Codec 13 is used for data sending. Otherwise, Codec 12 is used.*

Short video explaining TCP binary mode:

https://wiki.teltonika-gps.com/view/File:TCP_Binary_vid.mp4

Short video explaining TCP Ascii mode:

https://wiki.teltonika-gps.com/view/File:TCP_ASCII_vid.mp4

TCP Binary settings

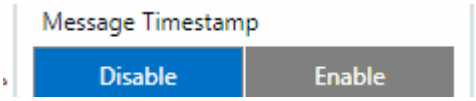
TCP Binary has a setting Prefix. It is possible to set Prefix 1, Prefix 2, or Prefix 3. These prefixes can be used separately or in unison. To configure this setting a value from 0 to 255 in decimal has to be

entered. The device will convert this value to HEX and compare the 1st, 2nd, or 3rd byte from incoming data. If the values do not match, the device will not accept incoming data.

Video explaining TCP Binary settings:

https://wiki.teltonika-gps.com/view/File:TCP_Binary_prefixes.mp4

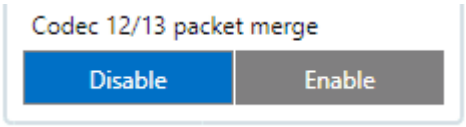
Message Timestamp



Message Timestamp parameter is used to determine if it is necessary to include timestamp in RS-232 TCP packet when sending to server. If parameter is Enabled, then Codec 13 is used for data sending. Otherwise, Codec 12 is used.

Codec 12/13 Packet Merge

This additional option configures the device to merge RS-232 records into a single packet instead of sending many separate packets.



This functionality only affects **TCP Binary Buffered** or **TCP Ascii Buffered** RS232 modes.

When enabled, the device will merge saved RS-232 records together into a single data packet until it is able to send it to the server. This optimizes RS-232 data sending by not having to send each RS-232 record separately. If the amount of saved RS-232 data exceeds the maximum 4 bytes of data in one Codec 12 packet, the device will start saving incoming RS-232 data into a new record.

RS-232 / RS-485 CMD ID

This parameter is used when is sending RS232/RS485 packet to a server, it overrides command type value in Codec12/Codec13 with user defined CMD ID value (1 - 14). behavior when it receives different CMD ID (Type) values in GPRS packet from server.

When receives these CMD ID values in the GPRS packet it behaves accordingly:

CMD ID	Functionality
5	Parse Codec12/Codec14 packet from the server
7	Forward packet meant for Garmin system
14	Forward packet to external device via uart using RS232/RS485
16	Forward packet to paired Bluetooth device

RS-232 Modes

RS-232 LCD mode

In this mode, the user is able to communicate with the server through the terminal. A link between the FMB device and the server has to be established for this mode to function properly.

To communicate from terminal to server - in terminal send command - `WT^W <text>`

To communicate with the server to the terminal send command - `#DO DAT= <text>` packet in Codec12 protocol

RS-232 RFID HID/RFID MF7 mode

The difference between RFID HID Mode and RFID MF7 Mode is that in RFID MF7 Mode understands RFID messages that are in hexadecimal text format and RFID HID Mode interprets messages that are in binary format. The type of RFID message sent to depends on the RFID reader. For example, the RFID MF7 mode message looks like `$aa$02$03$04$17$89$00$01` while HID mode message is of the following format: `1213141519`

The selected mode has to correspond to the RFID reader's mode. Please contact your local sales representative for more information about RFID IDs and devices.

RS-232 Garmin mode

Garmin provides a Fleet Management Interface Tool Kit, once is connected to the navigator it enables the driver to have a "screen" in their vehicle for real-time navigation and messaging and offers job dispatch capabilities to help them be more efficient.
and Garmin operational diagram is shown on the figure below.

RS-232 Delimiter mode

- This feature works from **Ver.03.28.05.Rev.00+** with Codec 8 Extended enabled.

Delimiter mode have two configurable delimiters to mark start and end of packet received from external device. Delimiter is configured as HEX string, max length 3 Bytes.

Examples: START delimiter param ‘3C’ and END delimiter param: ‘3E’ (3C = ASCII ‘<’ and 3E = ASCII ‘>’), < payload > (in ASCII) where ‘<’ and ‘>’ are the start and end delimiter START delimiter

param: ‘02’ and END delimiter param: ‘0A 0D 03’, **0x02** 0xFF 0xFF 0xFF 0xFF **0x0A 0x0D 0x03** (hexadecimal) where ‘0x02’ and ‘0x0A 0x0D 0x03’ is a delimiter

Note: If payload is empty. Record will be generated with empty payload.

Name	ID	Min symbols	Max symbols	Description
Start delimiter	167	0 (empty)	6 (3 bytes)	Dynamic HEX string value from 0 symbols (not configured) to 6 symbols (delimiter 3 bytes). Configured value have to be even number of symbols
End delimiter	168			

Table 1. START and END delimiters are configurable via SMS/GPRS Configurator

Start delimiter	End delimiter	Description
Empty	Empty	Packet received from RS232 will be saved to record. If payload is bigger than 254 bytes it will be separated to 2 or more records. Each record will be saved with incremented index starting from 0 index. 1 second of silence in the line is counted as the end of the packet. After end detected FMB will generated record with collected data.
Empty	Configured	Packet received from RS232 will be saved to record. If payload is bigger than 254 bytes it will be separated to 2 or more records. Each record will be saved with incremented index starting from 0 index. After end detected FMB will generated record with collected data. After record was generated by detected end delimiter, FMB will start collect data for next record right away.
Configured	Empty	Packet received from RS232 will be saved to record. If payload is bigger than 254 bytes it will be separated to 2 or more records. Each record will be saved with incremented index starting from 0 index. Data before Start delimiter will be discarded. 1 second of silence in the line is counted as the end of the packet. After end detected FMB will generated record with collected data.
Configured	Configured	Packet received from RS232 will be saved to record. If payload is bigger than 254 bytes it will be separated to 2 or more records. Each record will be saved with incremented index starting from 0 index. Data before Start delimiter will be discarded. After record was generated by detected end delimiter, FMB will start searching for next payload in received packet right away.

Table 2. Functionality description by delimiter configuration

Baudrate and parity configurable in base functionality

- New IO element **Serial Packet** ID:109:

When **Delimiter mode** is selected and packet is received via RS232 FMB will save record with added **Serial Packet** IO element with HIGH priority. If any other mode than

Delimiter is selected “Serial Packet” IO element will not be saved in records;

Serial Packet length will be variable, max length 257 bytes;

While no **Serial Packet** is received from external device FMB will save other periodic or eventual records without **Serial Packet** IO element.

Record with Serial Packet										
Preamble 4 bytes	AVL Data length	Timestamp	GPS element	Event ID,	IOs,	IO[109],	Index, 1 Byte	Data Length, 1 Byte	Payload, 255 Bytes	CRC
	4 Bytes	8 Bytes	15 Bytes	2 Bytes	X Bytes	2 Byte				4 Bytes

Table 3. Example of serial packet with Codec8 Extended Protocol

Index: Indicates the number of generated payload. Starting from 0 index.

Data Length: Length of payload.

Payload: Data parsed by delimiter functionality.

Note: Delimiter IO element is string type. Each type of IO elements has identifier byte that identify how many IOs of that type stored in record.

IO Types:

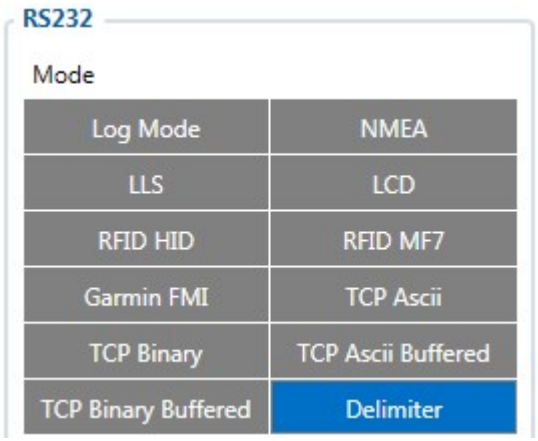
- 1byte IO elements
- 2 bytes IO elements
- 4 bytes IO elements
- 8 bytes IO elements
- String IO elements

Event ID	109 (delimiter)
Total IO count	All configured + 1 (delimiter)
1 byte IO’s count	All configured 1 byte IO’s count
1 byte IOs	All configured 1 byte IOs
2 bytes IO’s count	All configured 2 bytes IO’s count
2 bytes IOs	All configured 2 bytes IOs
4 bytes IO’s count	All configured 4 bytes IO’s count
4 bytes IOs	All configured 4 bytes IOs
8 bytes IO’s count	All configured 8 bytes IO’s count
8 bytes IOs	All configured 8 bytes IOs
String type IO’s count	All string type IO’s count
String type IOs	ID, index, length, serial packet
…	…

Table 4. Record structure

String type IO count byte will be added to the record only if record was generated with event ID equal to serial packet. Regular records that were generated not by Serial Packet will NOT have string type IO count identifier byte.

Configurable parameters



*New RS232 mode **Delimiter***

*Start and Stop Delimeter for new RS232 mode **Delimiter** value in HEX*

Garmin protocols

The following is a list of protocols supported and the corresponding list of features/benefits. FMB125 can fully support Fleet Management Interface (FMI) versions up to 2.1. Other or higher versions may be supported, but Teltonika is not responsible for the changes made by Garmin, which may affect the function of FMB125 and Garmin products. For more information about Garmin products and FMI versions, please refer to https://www.garmin.com/en-US/fleet-ready-navigators/. Notice that some Garmin products use different connection cables than others.

Standard protocols

Text Message Protocol:

- Allows text messages sent to the device to be displayed in "inbox" on navigation unit;
- Garmin can provide a confirmation that message was read;
- Garmin can also provide a yes/no box below the text of the message to enable a quicker response;
- Messages can be up to 199 characters long;
- Messages can also be generated from the device and sent to dispatcher/office;
- Messages received will be notified to the driver through a pop-up alert on the Garmin screen;
- Garmin provides a "virtual keyboard" for text communication.

Stop (Destination) Protocol:

- Garmin can display a list of Stops/Jobs reported to the device in a separate category called "My Stops";
- Driver has the ability to navigate directly to Stop from the list;
- Garmin can provide status of a current Stop in progress;
- Garmin can indicate whether the driver has stopped at the location;
- Garmin can inform how far the driver has progressed through the list of Stops;
- Garmin can provide confirmation that the driver has received a particular Stop, familiarized himself/herself with its details, or removed it from the list;
- Can provide confirmation that a Stop has been reached.

Estimated Time of Arrival Protocol:

- Dispatcher/office can request the ETA of the current Stop/job in progress;
- Garmin can notify about the actual time of arrival as well as the distance remaining to a Stop.

Auto-Arrival at Stop Protocol:

- This feature is used to tell Garmin PND to automatically detect that it has arrived at a Stop and then to prompt the driver if he/she would like to mark the Stop as done and begin navigating to a next Stop on the list;
- Auto-arrival can be determined by how long the unit is stopped close to the destination (in the case driver has to park and walk) or by how close the unit needs to be to the destination before the Auto-arrival feature is activated.

Data Deletion Protocol:

- Dispatcher/office has the ability to wipe clean the data on Garmin PND;
- It allows to clean messages in the inbox and remove stops.

Enhanced protocols

Canned Responses/Messages:

- Fleet managers can communicate by sending up to 200 "canned" responses from the server to be stored directly on Garmin devices;
- Up to 50 of these canned responses can be utilized for any given scenario;
- Drivers can store up to 120 canned messages, eliminating the need to type while driving.

Status Protocol:

- Up-to-the-minute communications that allow drivers to automatically send status updates;
- Driver's units can store up to 16 status indicators such as start/stop shift, on/off break, etc.

Supported features on TAVL client application

TAVL client application lets users use the following features of Garmin FMI:

- Text messaging;
- Destination message;
- ETA request.

Text messaging

The text messaging feature lets the user communicate with the driver (the user that operates the Garmin device) by sending text messages via GPRS.

Destination message

Destination message is used to inform the driver of a new destination. When the Garmin device receives a destination message from the server it displays it as a Stop to the driver and also gives the driver the ability to start navigating to the Stop location. A new destination in the TAVL client is represented as a Geozone so a new Geozone (as destination) has to be created first.

ETA request message

Estimated Time of Arrival request message is used when the user wants to know an expected arrival time to the currently active destination and the distance (in meters) from the current object location to the destination.