Programming Orbis encoders
with Asynchronous serial interface

Programming the Orbis encoder via the UART interface allows to:

- set position offset,
- set multiturn counter (if available),
- set new baud rate value,
- configure continuous response feature,
- start/stop continuous response feature,
- run the self-calibration function,
- store current Orbis configuration parameters to a non-volatile memory,
- reset Orbis configuration parameters to factory defaults

Related product

Orbis true absolute rotary encoder

Unlocking sequence

Programming is executed by sending separate bytes to the encoder.

Each of the programming options must start with the special unlocking sequence.
This prevents unintentional changes on configuration parameters of the encoder.
The procedure of each programming option is described below.

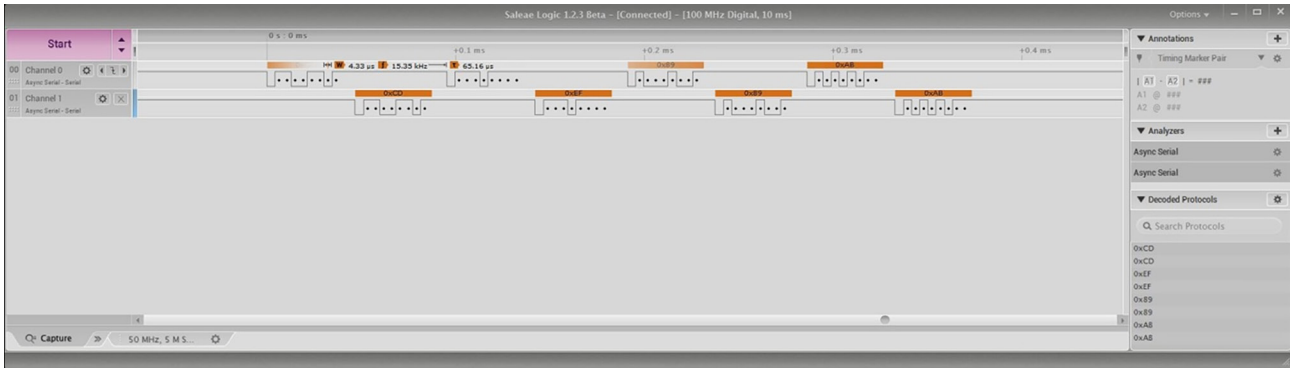
> The delay between each byte sent during programming must be a least 1 ms.

To unlock the encoder and enable programming, 4 exact bytes must be sent in specified order as listed in the following table.

Byte in sequence	Byte value (HEX)
B1	0xCD
B2	0xEF
B3	0x89
B4	0xAB

Whenever the sequence is interrupted with wrong byte, the unlock stage is reset and waits for the new sequence.

Screen capture below shows the unlocking sequence captured by the logic analyzer.



When the encoder is successfully unlocked, the fifth byte in a complete sequence must be one of the valid programming command bytes described in the following table.
If the fifth byte is none of these, the encoder will be »locked« again.

Programming command bytes

Programming command bytes vary depending on the desired programming function. The valid programming bytes are listed in the following table. Some of them perform the desired task immediately (e.g., resetting factory defaults), while others require additional data bytes (e.g., setting a new baud rate). The specific tasks are described in the following sections. When the command is executed, the encoder always returns to the locked state.

Programming feature	Programming command byte (ASCII)		Additional data bytes required
Position offset setting	'Z'	(0x5A)	4
Multiturn counter setting	'M'	(0x4D)	4
Baud rate setting	'B'	(0x42)	4
Continuous-response setting	'T'	(0x54)	4
Continuous-response start	'S'	(0x53)	0
Continuous-response stop	'P'	(0x50)	0
Configuration parameters save	'c'	(0x63)	0
Configuration parameters reset	'r'	(0x72)	0

Programming execution

Position offset setting

After sending a programming command byte 'Z', which allows the position offset to be set, four additional data bytes are required. These bytes contain the actual position offset data in encoder count values. When all four bytes are received, the new position offset value is set. The following table shows how these four bytes are converted into a 32-bit value representing a position offset.

Byte in sequence	Bits in 32-bit position offset
B1	b31 – b24
B2	b23 – b16
B3	b15 – b8
B4	b7 – b0

If the applied position offset is larger than the actual encoder resolution, the programming is discarded. The position offset is not changed.

After programming the position offset, the value is stored in RAM and used immediately. For permanent storage in non-volatile memory, another programming operation must be performed to save configuration parameters (refer to **Saving configuration parameters**).

By default (factory setting), the position offset is set to 0.

Multiturn counter setting

After sending a programming command byte 'M' that allows the multiturn counter to be set, four more additional data bytes are required. These bytes contain the desired multiturn counter. When all four bytes are received, the new multiturn counter is set. The following table shows how these four bytes are converted into a 32-bit value that represents a multiturn counter.

Byte in sequence	Bits in 32-bit multiturn counter
B1	b31 – b24
B2	b23 – b16
B3	b15 – b8
B4	b7 – b0

Only lower 16 bits are used to preset multiturn counter. B1 and B2 must be zero.

Baud rate setting

After sending a programming command byte 'B', which allows the baud rate to be set, four more data bytes are required. These bytes contain the new required baud rate in bits per second. When all four bytes are received, the new baud rate is set. The following table shows how these four bytes are converted into a 32-bit value representing a new baud rate.

Byte in sequence	Bits in 32-bit baud rate value
B1	b31 – b24
B2	b23 – b16
B3	b15 – b8
B4	b7 – b0

After the sequence to programme the baud rate is completed, the value is stored in RAM and used immediately. Further communication with the encoder is not possible with the previous baud rate. The master must reconfigure itself to the new baud rate and test the communication with the encoder. In case of exact communication, the new baud rate must be stored in a non-volatile memory by programming procedure for saving configuration parameters (refer to **Saving configuration parameters**). In that case, the new baud rate will be applied after each subsequent cycle of the power supply. If the new configuration is not stored in non-volatile memory, the previous setting will be restored on a power cycle.

Baud rate programming allows the programming of any baud rate in increments of 1 BAUD.

Continuous-response setting

After sending a programming command byte 'T', which enables continuous-response setting, four additional data bytes are required. These bytes contain:

- required period of continuous responds in microseconds,
- command that should be continuously responded,
- setting for automatic start of responding after power-on of the encoder.

Table below shows how these four bytes are transformed into separate configuration parameters.

Byte		Meaning
B1	b7 – b1	Not used
	b0	1 to enable automatic start after power-on of the encoder 0 to disable automatic start after power-on of the encoder
B2	b7 – b0	ASCII command for continuous responding (one from the basic command set, refer to BRD01 Orbis data sheet available at **RLS Media center**.)
B3	b7 – b0	b15 – b8 of the 16-bit period value
B4	b7 – b0	b7 – b0 of the 16-bit period value

The automatic start of the continuous-responding after power-on of the encoder is disabled by default.

The default command for continuous response is ASCII '3' (short response). Any other command from the "Command (ASCII)" table can be programmed. If none of the valid commands are programmed, it will be discarded and reset to the default (ASCII '3').

After programming the continuous-response, the parameters are stored in RAM. For permanent storage in a non-volatile memory, another programming operation must be performed to save the configuration parameters (refer to **Saving configuration parameters**). After this, the programmed values become valid after the first subsequent power cycle of the encoder.

The resolution of the period for the continuous-response is 1 µs. The maximum programmed period can be 65535 µs. The shortest possible period depends on the baud rate used and the command selected. If the programmed period is too short and the previous frame has not yet been transmitted, the transmission of the next frame will be delayed by the programmed period. If the user prefers the shortest possible period regardless of the baud rate and command settings, a period setting of 1 µs can be set.

Refer to the Continuous-response setting in the **Programming example**, which shows the appropriate continuous response programming sequence. When continuous response is active, any command sent to the encoder may result in an echo byte being inserted into the position data packet. It is recommended that you stop the continuous transmission with the 'P' command before sending any other command to the encoder.

Continuous-response start

After sending a programming command byte 'S', the continuous-response feature is activated immediately. The encoder starts transmitting data according to the selected continuous-response period and command.

Continuous-response stop

After sending a programming command byte 'P', the continuous-response feature is immediately deactivated.

Saving configuration parameters

After sending a programming command byte 'c', configuration parameters of the encoder are immediately stored in non-volatile memory. These parameters include:

- current baud rate,
- position offset,
- continuous-response settings (period, command, autostart enable command).

Resetting the encoder to factory settings

After sending a programming command byte 'r', the configuration parameters of the encoder are immediately reset to the factory values.

Self-calibration

Self-calibration of Orbis encoders is suitable after mounting the readhead. It improves the accuracy of the encoder, which depends on the installation precision. The user must first unlock the programming option with the sequence 0xCD 0xEF 0x89 0xAB following the SelfCalStart command 0x41 to start the self-calibration. During the process, communication via the UART interface is not possible; the encoder does not respond to incoming commands. The first command received during this period is queued and processed at the end of the self-calibration cycle.

The end of the process is indicated by LED flashing rapidly for 3 seconds. LED flashes green if the self-calibration was successful, otherwise it flashes red. After that, the UART interface is active again. The status byte of the self-calibration can be read with the command 0x69. An echo byte is returned first, followed by a status byte containing a two-bit counter and two status bits. The counter is incremented at the end of each self-calibration process. The error bits indicate the success or reasons for the error.

The status should be read before self-calibration. The 0x69 command returns 2 bytes, the first is the echo byte and the next is the status byte. The controller must remember the current self-calibration counter (bits 1:0). After the self-calibration command is sent, LED must be observed for completion. If LED is not visible, the readhead should be polled via the UART interface until communication with the readhead is re-established. Or wait 10 seconds, which is the longest possible time for completion. The self-calibration status register should then be read again. When the self-calibration counter has increased by 1 (compared to the previously read value), the self-calibration function has been completed. If the self-calibration was successful, both status bits (b3, b2) are zero.

The rotation speed during self-calibration can be up to 600 RPM and can be uneven, the direction is not important.
It is recommended to perform at least one complete rotation of the shaft within 10 s after sending the command.
Self-calibration must be started if there is no error (green LED).
If an error occurs during self-calibration, the result is invalid and the procedure must be repeated.

Self-calibration status byte values:

Bit	Meaning
b7 - b4	Reserved
b6	Calibration was already performed (error map was changed)
b3	Calculated parameters out of range. Mechanical installation is not inside tolerances.
b2	Timeout. Encoder ring did not make a complete turn during 10 seconds.
b1	Counter bit 1
b0	Counter bit 0

Unlock sequence: 0xCD 0xEF 0x89 0xAB
Self-calibration Start command: 0x41
Self-calibration Status request: 0x69

Programming examples

Encoder position offset setting to 5144 (0x1418) counts

Bytes listed in the table should be sent to the encoder in exact order with at least 1 ms of delay between bytes.

	Value			
Byte in sequence	**DEC**	**HEX**	**ASCII**	**Purpose**
B1	205	0xCD	/	Unlock sequence
B2	239	0xEF	/	
B3	137	0x89	/	
B4	171	0xAB	/	
B5	90	0x5A	'Z'	Offset position setting enable
B6	0	0x00	/	Additional data bytes
B7	0	0x00	/	
B8	20	0x14	/	
B9	24	0x18	/	

To store offset position in a non-volatile memory, encoder's configuration parameters must be saved in accordance to **Saving configuration parameters in a non-volatile memory**.

Continuous-response setting

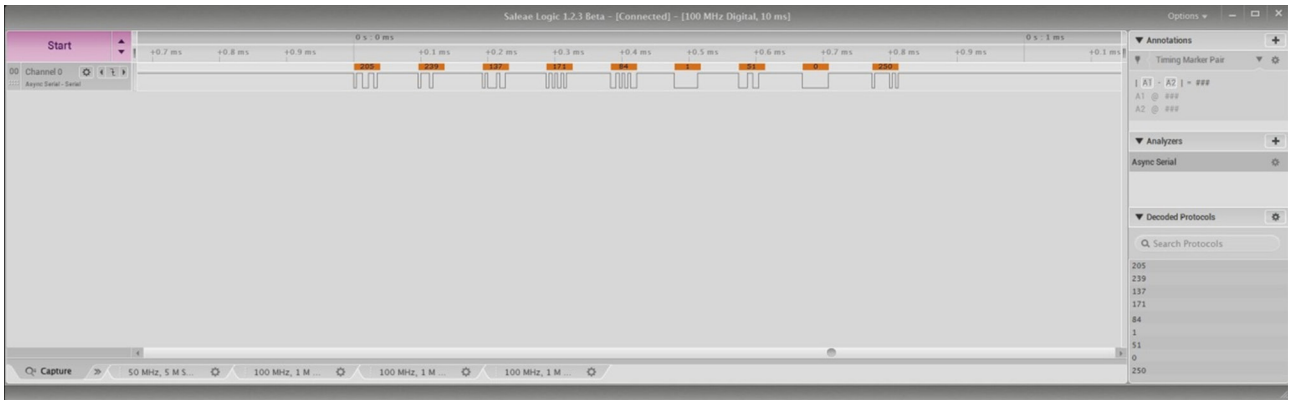
Continuous-response configuration parameters:

- Period: 250 µs (4 kHz)
- Command: '3' (shortest possible, 3 bytes of position + E/W)
- Auto-start after power-on: Enabled

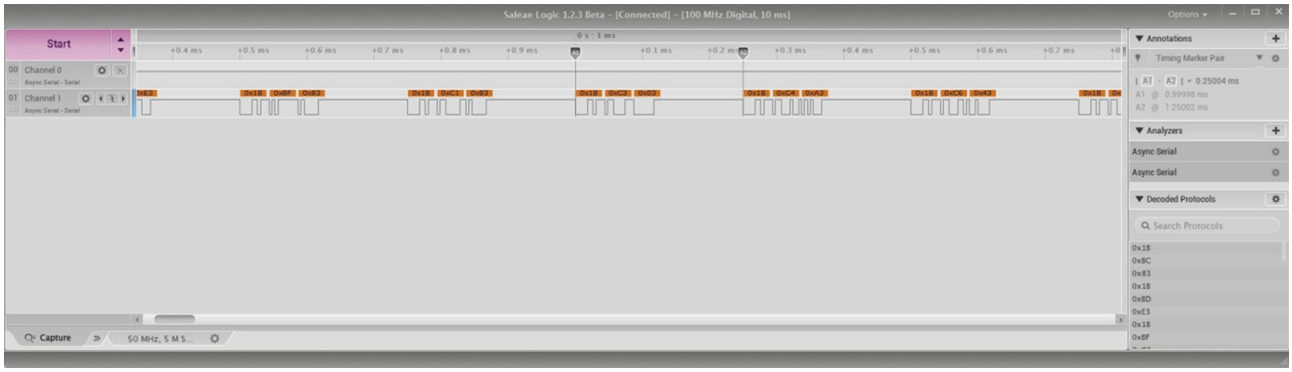
Bytes listed in table below should be sent to the encoder in exact order.

	Value			
Byte in sequence	**DEC**	**HEX**	**ASCII**	**Purpose**
B1	205	0xCD	/	Unlock sequence
B2	239	0xEF	/	
B3	137	0x89	/	
B4	171	0xAB	/	
B5	84	0x54	'T'	Continuous-response setting enable
B6	1	0x01	/	Auto-start enable
B7	51	0x33	'3'	Command for short response
B8	0	0x00	/	Period (250 µs)
B9	250	0xFA	/	

Screen capture below shows the described continuous-response programming sequence captured by the logic analyzer.



To store continuous-response settings in a non-volatile memory, encoder's configuration parameters must be saved in accordance to **Saving configuration parameters in a non-volatile memory** described below. After next cycle of the power supply, the encoder will start to transfer data automatically with 250 µs period as shown on the following screen capture.



Saving configuration parameters in a non-volatile memory

Bytes listed in table below should be sent to the encoder in exact order.

	Value			
Byte in sequence	**DEC**	**HEX**	**ASCII**	**Purpose**
B1	205	0xCD	/	Unlock sequence
B2	239	0xEF	/	
B3	137	0x89	/	
B4	171	0xAB	/	
B5	99	0x63	'c'	Save configuration parameters

Reset of configuration parameters to the factory settings

Bytes listed in table below should be sent to the encoder in exact order.

	Value			
Byte in sequence	**DEC**	**HEX**	**ASCII**	**Purpose**
B1	205	0xCD	/	Unlock sequence
B2	239	0xEF	/	
B3	137	0x89	/	
B4	171	0xAB	/	
B5	114	0x72	'r'	Reset configuration parameters

Head office

RLS Merilna tehnika d.o.o.

Poslovna cona Žeje pri Komendi
Pod vrbami 2
SI-1218 Komenda
Slovenia

T +386 1 5272100
F +386 1 5272129
E mail@rls.si

www.rls.si

Global support

Visit our **website** to contact your nearest sales representative.

Document issues

Date	Issue	Page	Description
3. 1. 2022	**2**	5	Self calibration description amended
		-	New design