Extended (8K) PL Integration Manual

Proprietary and Confidential

Version 1.0

Revised and Updated: October 10, 2013

Pointer Telocation Ltd. 14 Hamelacha St., Rosh Ha'ayin 48091, Israel • Tel: 972-3-5723111 • Fax: 972-3-5723100 • www.pointer.com

Legal Notices

IMPORTANT

1. All legal terms and safety and operating instructions should be read thoroughly before the product accompanying this document is installed and operated.
2. This document should be retained for future reference.
3. Attachments, accessories or peripheral devices not supplied or recommended in writing by Pointer Telocation Ltd. may be hazardous and/or may cause damage to the product and should not, in any circumstances, be used or combined with the product.

General

The product accompanying this document is not designated for and should not be used in life support appliances, devices, machines or other systems of any sort where any malfunction of the product can reasonably be expected to result in injury or death. Customers of Pointer Telocation Ltd. using, integrating, and/or selling the product for use in such applications do so at their own risk and agree to fully indemnify Pointer Telocation Ltd. for any resulting loss or damages.

Warranty Exceptions and Disclaimers

Pointer Telocation Ltd. shall bear no responsibility and shall have no obligation under the foregoing limited warranty for any damages resulting from normal wear and tear, the cost of obtaining substitute products, or any defect that is (i) discovered by purchaser during the warranty period but purchaser does not notify Pointer Telocation Ltd. until after the end of the warranty period, (ii) caused by any accident, force majeure, misuse, abuse, handling or testing, improper installation or unauthorized repair or modification of the product, (iii) caused by use of any software not supplied by Pointer Telocation Ltd., or by use of the product other than in accordance with its documentation, or (iv) the result of electrostatic discharge, electrical surge, fire, flood or similar causes. Unless otherwise provided in a written agreement between the purchaser and Pointer Telocation Ltd., the purchaser shall be solely responsible for the proper configuration, testing and verification of the product prior to deployment in the field.

POINTER TELOCATION LTD.'S SOLE RESPONSIBILITY AND PURCHASER'S SOLE REMEDY UNDER THIS LIMITED WARRANTY SHALL BE TO REPAIR OR REPLACE THE PRODUCT HARDWARE, SOFTWARE OR SOFTWARE MEDIA (OR IF REPAIR OR REPLACEMENT IS NOT POSSIBLE, OBTAIN A REFUND OF THE PURCHASE PRICE) AS PROVIDED ABOVE. POINTER TELOCATION LTD. EXPRESSLY DISCLAIMS ALL OTHER WARRANTIES OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, SATISFACTORY PERFORMANCE AND FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL POINTER TELOCATION LTD. BE LIABLE FOR ANY INDIRECT, SPECIAL, EXEMPLARY, INCIDENTAL OR CONSEQUENTIAL DAMAGES (INCLUDING WITHOUT LIMITATION LOSS OR INTERRUPTION OF USE, DATA, REVENUES OR PROFITS) RESULTING FROM A BREACH OF THIS WARRANTY OR BASED ON ANY OTHER LEGAL THEORY, EVEN IF POINTER TELOCATION LTD. HAS BEEN ADVISED OF THE POSSIBILITY OR LIKELIHOOD OF SUCH DAMAGES.

Intellectual Property

Table of Contents

1 Introduction

The purpose of this document is to describe the new Cello family devices support for extended configuration memory of 8 Kbytes.

The new configuration space is expected to support new Cello variants requiring more configuration data. The change includes remapping of configuration areas, protocol changes and tool changes. This document provides information about the memory remapping, and the resulted server side modifications.

1.1 Definitions, Acronyms and Abbreviations

Name	Description

1.2 References and Bibliography

No.	Document Name	Number	Version	Date	Location
1					
2					

1.3 Revision History

Version number	Date	Description
1.0	October 10, 2013	First draft

1.4 Compatibility Table

Variant	Firmware Version
Cello-IQ	32f
Cello-CANiQ	33a

2 PL Programming and Uploading for the Extended Configuration Memory

2.1 Overview

The introduction of 8 Kbytes configuration memory required support for a new OTA protocol, as the existing one had a limited address range. A new protocol type (11) has been added to the fleet protocol as an infrastructure for advanced modular application level message interchange. The protocol is modular as it is designed to carry predefined application layer data units (or modules) between an application running inside the Cello device and a server side application. The new Type 11 protocol is described in the *Cellocator Wireless Communication protocol* document.

The programming scenario has been changed as well

2.2 General Type 11 message format

The Type 11 message includes header, modules and check sum as presented in the Figre below.

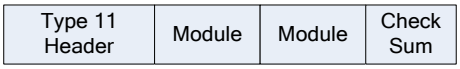

Figure 1: Type 11 Message

2.3 General Outbound Type 11 message format

The table below describes the information elements of an outbound message (that is, a message sent by the Cello unit). The header is a legacy MCGP header like the legacy fleet. The message length is held in 16 bits, in theory enabling a message size of 65536; in practice the message length is constrained by the Cello unit hardware. The numerator is used as an indication to associate between the message request and its response. The "Packet Control Field" identifies the message direction.

1	System Code, byte 1 – ASCII "M"
2	System Code, byte 2 – ASCII "C"
3	System Code, byte 3 – ASCII "G"
4	System Code, byte 4 – ASCII "P"
5	Message Type byte (11)
6	Length
	Unit's ID (total 32 bits)

	Numerator
	Spare (sent as 0)
	Packet Control Field
	CSA FW ID Module : Mandatory Module, see Packet Control Field
…	
	Other modules
	Check Sum

2.4 General Inbound Type 11 message format

The table below describes a Type 11 inbound message (a message sent to the Cello unit). Unlike an outbound message, the "Packet Control" field will reflect the packet direction, as described in Packet Control Field.

1	System Code, byte 1 – ASCII "M"
2	System Code, byte 2 – ASCII "C"
3	System Code, byte 3 – ASCII "G"
4	System Code, byte 4 – ASCII "P"
5	Message Type byte (11)
6	Destination Unit's ID (total 32 bits)
7	
8	
9	
10	Command Numerator
11	Authentication
12	
13	
14	

15	Packet Control Field
16-17	Length (of the modules section - not including the checksum)
18	Spare (sent as 0)
19	
20	
21	
20	Modules
	Check Sum

2.5 Packet Control Field description

Bit 7	Bit 6	Bits 5-0
Direction	Out of space indication	unused

Direction

0 – Data from the unit (Outbound)

1 – Request (Inbound)

Out of Space Indication

0 – All the requested data is present in the message

1 – Some sub-data was not returned due to data size

2.6 Programming related Type 11 modules

This section describes Type 11 modules associated with Cello configuration memory programming and configuration uploading.

The following sections describe three Type 11 message examples for **configuration programming**, **configuration block request by the server** and **configuration memory response to a request command**. Please note that although module number 10 can carry multiple instances of programming commands, currently we only support a single instance.

2.6.1 *Configuration Memory Block Programming*

The table below describes a Type 11 message example carrying a configuration block programming command.

Byte Number	Byte level Description	Field associations	
1	System Code, byte 1 – ASCII "M"	Type 11 Header	
2	System Code, byte 2 – ASCII "C"		
3	System Code, byte 3 – ASCII "G"		
4	System Code, byte 4 – ASCII "P"		
5	Message Type byte (11)		
6	Length		
7			
8	Destination Unit's ID (total 32 bits)		
9			
10			
11			
12	Command Numerator		
13	Spare (sent as 0)		
14			
15			
16			
17	Packet Control Field		
18	Module Name = 10	Module number 10 header with one programming instance	Programming Module number 10 with 1 programming instance
19	Module Length		
20			
21	Numerator		
22			
23	Number of memory instances attached: Set to 1		
24	Memory space: Spare: Set to 0	Programming instance carried by module number 10	
25	Memory entry type : 0-Bit, 1-Byte, 2-Word (16bits), 3-Long(32 bit) Must be set to 1 to support byte access.		

Byte Number	Byte level Description	Field associations	
26	Configuration memory address 0-8 Kbytes		
27			
28			
29			
30	Configuration memory blocks size.		
31			
32			
33			
	Payload		
	….		
	Check Sum	Check Sum	

2.6.2 *Configuration Memory Block Programming Ack*

Byte Number	Byte level Description	Field associations
1	System Code, byte 1 – ASCII "M"	Type 11 Header
2	System Code, byte 2 – ASCII "C"	
3	System Code, byte 3 – ASCII "G"	
4	System Code, byte 4 – ASCII "P"	
5	Message Type byte (11)	
6	Length	
7		
8	Destination Unit's ID (total 32 bits)	
9		
10		
11		

Byte Number	Byte level Description	Field associations	
12	Command Numerator		
13	Spare (sent as 0)		
14			
15			
16			
17	Packet Control Field		
18	Module Name = 10	Module number 10 header with one Ack. instance	Module 10 Programming Ack.
19	Module Length		
20			
21	Numerator		
22			
23	Number of Instances: Set to 1		
24	Instance action status 0 - OK 1 - Write Error	One instance of Ack status	
25	CheckSum		

2.6.3 *Configuration Memory Block Request Command*

The table below describes a Type 11 message example carrying a configuration block request command.

Byte Number	Byte level Description	Field associations
1	System Code, byte 1 – ASCII "M"	Type 11 Header
2	System Code, byte 2 – ASCII "C"	
3	System Code, byte 3 – ASCII "G"	
4	System Code, byte 4 – ASCII "P"	
5	Message Type byte (11)	
6	Length	
7		

Byte Number	Byte level Description	Field associations	
8	Destination Unit's ID (total 32 bits)		
9			
10			
11			
12	Command Numerator		
13	Spare (sent as 0)		
14			
15			
16			
17	Packet Control Field		
18	Module Name = 11	Module number 11 header with one instance of configuration Request	Module Number 11: Configuration block request
19	Module Length		
20			
21	Numerator		
22			
23	Number of Instances: Set to 1		
24	Memory space: Spare Set to 0	Configuration request instance carried by module number 11.	
25	Memory entry type : 0-Bit, 1-Byte, 2-Word (16bits), 3-Long(32 bit) Must be set to 1 to support byte access.		
26	Configuration memory address 0-8 Kbytes		
27			
28			
29			
30	Configuration memory blocks size.		
31			
32			
33			

Byte Number	Byte level Description	Field associations
32	Check Sum	

2.6.4 *Configuration Memory Block Response*

The table below describes a Type 11 message example carrying a configuration block Response message. This message is a response to the Configuration Memory Block Request Command. The message carries the content of configuration data memory; the type 11 message in this example carries 1 module.

Byte Number	Byte level Description	Field associations	
1	System Code, byte 1 – ASCII "M"	Type 11 Header	
2	System Code, byte 2 – ASCII "C"		
3	System Code, byte 3 – ASCII "G"		
4	System Code, byte 4 – ASCII "P"		
5	Message Type byte (11)		
6	Length		
7			
8	Destination Unit's ID (total 32 bits)		
9			
10			
11			
12	Command Numerator		
13	Spare (sent as 0)		
14			
15			
16			
17	Packet Control Field		
18	Module Name: FW ID: 8	Firmware ID Module header	Firmware ID module
19	Length of module 2 bytes		
20			
21	Spare	Firmware ID	

Byte Number	Byte level Description	Field associations	
22	Protocol version 1		
23	FW ID		
24			
25			
26			
18	Module Name = 11	Module number 11 header with one instance of memory data	Module number 11 carrying returned programming data
19	Module Length		
20			
21	Numerator		
22			
23	Number of Instances: Set to 1		
24	Memory space: Spare	One instance of Programming memory returned	
25	Memory entry type : 0-Bit, 1-Byte, 2-Word (16bits), 3-Long(32 bit) Must be set to 1 to support byte access.		
26	Configuration memory address 0-8 Kbytes		
27			
28			
29			
30	Configuration memory blocks size.		
31			
32			
33			
3	Payload		
	….		
	CheckSum		

2.7 Programming Scenario

Programming scenarios are initiated by the server side by sending a Configuration Memory Block Programming command towards the selected Cello unit.

The Cello unit will acknowledge each programming command with a Configuration Memory Block Programming Ack. When the server has no more configuration data to send, it is advisable to perform verification by requesting the unit to upload its configuration memory contents using the Configuration Memory Block Request Command and compare the configuration data returned via the Configuration Memory Block Response with the server's known configuration block.

If the verification phase shows a full correlation between the original configuration block sent and the configuration data uploaded the server will perform an OTA reset command to restart the Cello unit with its new configuration.