CS710S Handheld Sled RFID Reader Bluetooth and USB Byte Stream API Specifications

Version 1.0

2023 10 05

Copyright Convergence Systems Limited 2023 All Rights Reserved

Chapter 1: Release Notes

Dates	Release	Description
2023 2 16	0.97	Pre-Release
2023 5 30	0.101	Miscellaneous changes
2023 07 31	0.102	-9 channels
2023 10 05	1.0	First Release

Copyright Convergence Systems Limited 2023 All Rights Reserved

Chapter 2: Content

Chapter 1: Release Notes 2
Chapter 2: Content 3
Chapter 3: Introduction 5
Chapter 4: Interfaces to CS710S 6
4.1 Bluetooth Connection Details 7
4.1.1 CC2652 Family Bluetooth IC DLE Configuration 21
4.2 USB Connection Details 25
Chapter 5: LED definitions 26
Chapter 6: Push buttons Definition 27
Chapter 7: API Set Definition 28
5 API Format 28
7.2: Header of RFID Reader Commands 30
7.3: Header of Barcode Reader Commands 31
7.4: Header of Special Notifications 32
7.5: Header of ATMEL IC Commands 33
7.6: Header of Bluetooth IC Commands 34
Chapter 8: Payload – RFID Reader Commands 35
8.1 Downlink Payload 35
8.2 Uplink Payload 36
Chapter 9: Payload – Barcode Reader Commands 37
9.1 Downlink Payload 38
9.2 Uplink Payload 39
9.2.1 Uplink Barcode Payload Format 39
Chapter 10: Payload – Special Notifications 40
10.1 Downlink Payload 41
10.2 Uplink Payload 42
Chapter 11: Payload – ATMEL IC Commands 43
11.1 Downlink Payload 43
11.2 Uplink Payload 44
Chapter 12: Payload – Bluetooth IC Commands 45
12.1 Downlink Payload 45
12.2 Uplink Payload 46
Chapter 13: Factory Default Settings 47

Copyright Convergence Systems Limited 2023 All Rights Reserved

Appendix A – CSL Ex10 Based RFID Command Specification 48
A.1 Introduction 48
A.1.1 Communication Types Introduction 48
A.1.2 Program Flow Introduction 51
A.1.3 Packet Format Introduction 52
A.1.4 Tag Caching and Duplicate Elimination Rolling Window 53
A.2 CSL Commands 54
A.2.1 Write Register Command 55
A.2.2 Read Register Command 57
A.2.3 Operation Commands 59
A.2.4 Administration Commands 63
A.3 CSL Registers 66
A.3.1 CSL RFID Registers Summary Table 66
A.3.2 CSL RFID Registers Details 70
A.3.3 EPC Commands Refresh 72
A.3.4 CSL Ex10 Country Enum Table 75
A.3.5 Reader Mode Table 83
A.3.6 GPIO that can be used .... **Error! Bookmark not defined.**
A.4 CSL RFID Uplink Packets 84
A.4.1 CSL RFID Uplink Packets Summary Table 85
A.4.2 CSL RFID Uplink Packets Details 87
Appendix B – Error Codes 96
Appendix C: Barcode Reader Command Sequence Examples 99
C.1 Pre-setup of Barcode Reader 99
C.2 Normal Operation of Barcode Reader 100
Appendix D – How to Choose Reader Modes 101
Appendix E – Session 104
Appendix F – Tag Population and Q 105
Appendix G – Query Algorithm 106
Appendix H – Target 107
Appendix I – Security 108
Appendix J – Tag Focus 110
Appendix K – Models & Regulatory Regions 111
Appendix L – CRC Table/Compute Codes 112

Copyright Convergence Systems Limited 2023 All Rights Reserved

Chapter 3: Introduction

CSL E710 IC Based CS710S, is a Bluetooth or USB controlled RFID handheld reader sled. The sled allows the mounting of popular smart phones such as Android phone and iPhone, or iPod, onto the top; or connecting via USB from a PC or embedded system. One can also connect via Bluetooth from a Windows 10 laptop to it. An application inside the smart phones, downloadable free of charge from corresponding stores (Android Market or Apple App Store) will send commands via Bluetooth to the reader sled and operate the RFID reader module, the barcode module, the LEDs inside. Likewise, an application in the PC or embedded system can send commands down via USB to the CS710S. The App in the smart phone or the PC will also handle the display of data for interactive use by the operator, plus transferring of data out to the Internet Cloud.

The data collected by the smart phone App will contain the following:

1. RDID tag data
2. Barcode data

The user can further add functionalities to the application by developing their own applications based on the source codes of the smart phone App or the PC App, which is freely available from Convergence Systems Website.

This document describes the Bluetooth and USB Byte Stream API set. The CS710S can be controlled via a USB cable by a PC or any embedded system, sending byte stream down to CS710S, and with CS710S responding back via the USB cable in a byte stream manner. The CS710S can also be controlled via Bluetooth by iPhone, iPAD, Android phone, Windows 10 PC with Bluetooth BLE 5.0 connectivity, sending byte stream down to CS710S, and with CS710S responding back via Bluetooth in a byte stream manner. The following chapters define this byte stream interface. On the PC or embedded system platform, the user can employ any language: C#, Objective C, Swift, Java, Python, machine language, etc.

Copyright Convergence Systems Limited 2023 All Rights Reserved

Chapter 4: Interfaces to CS710S

For CS710S, the commands can be fed in via 2 physical interfaces:

1) Bluetooth BLE 5.0/5.1/5.2 (BLE stands for Bluetooth Low Energy)
2) USB

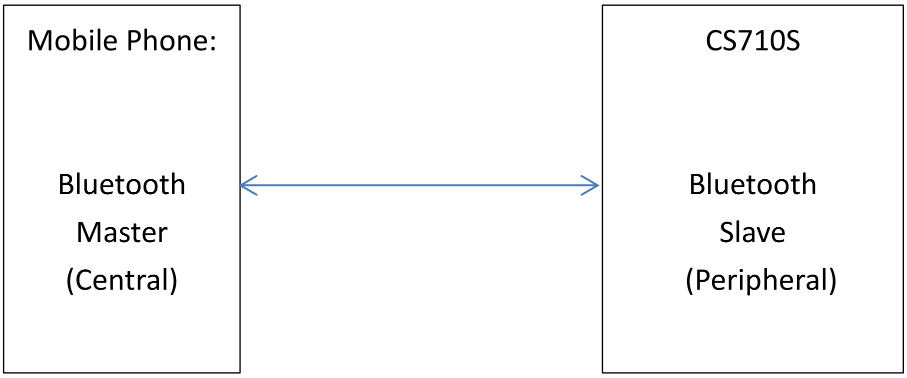
These are all byte stream commands.

There are 5 different groups of commands and replies/notifications:

1) RFID reader commands
2) Barcode reader commands
3) Special Notifications
4) Silicon Lab IC commands
5) Bluetooth IC commands

Copyright Convergence Systems Limited 2023 All Rights Reserved

4.1 Bluetooth Connection Details



For Bluetooth 5.0/5.1 connection, i.e. Bluetooth Low Energy, GATT protocol is used for search and connection.

UUID is *discoverable* by the smart phone whenever the CS710S is flashing the Bluetooth LED.

When the mobile phone application "scan" for Bluetooth devices, the iOS OS will provide the application with the Reader's Name, Device UUID (**9802**), Service UUID, etc.

CS710S's specific Service UUID is 9900 and 9901. 9900 is for downlink service from smart phone to CS710S (write characteristic), 9901 is for uplink service from CS710S to smart phone (send notification)

iOS will use these 2 numbers (9900 and 9901) to generate "characteristic UUIDs" for the application to use.

Write = TBD (for downlink command sending)
Notification = TBD (for uplink notification)

In other words, CS710S reader uses **Bluetooth Notification** to send data back to smart phone. CS710S/CS463 reader allows smart phone to use **Bluetooth Write Charateristic** to send data to the reader.

In theory, Bluetooth 5.0 has higher throughput than the previous Bluetooth 4.0 provided both the mobile phone and CS710S have implemented the proper configuration of Bluetooth 5.0.

However, the increase in data rate is not automatic and one needs to configure certain parameters to

Copyright Convergence Systems Limited 2023 All Rights Reserved

achieve highest rate.

If possible, the 2 Mbps (LE 2M PHY) should be configured on both sides. This will enable a raw data rate of 2 Mbps.

[Bluetooth 5 & BLE: Achieving maximum throughput and speed | Novel Bits](#)
https://interrupt.memfault.com/blog/ble-throughput-primer

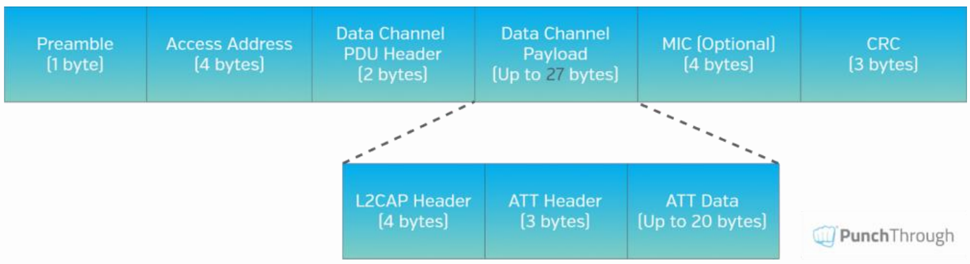
Copyright Convergence Systems Limited 2023 All Rights Reserved

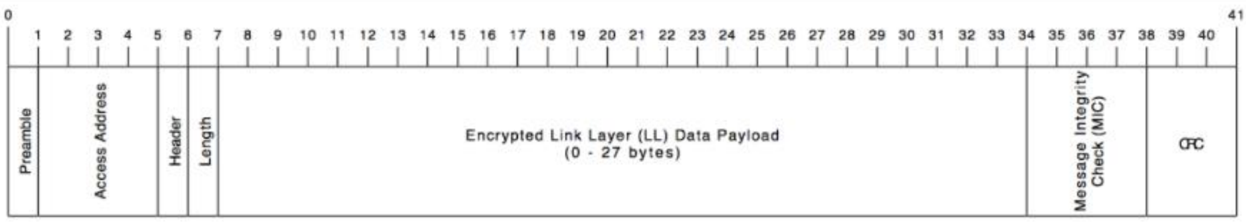
The following is definition of connection event and connection interval:

- **Connection Event** – For BLE, *exactly* two devices are talking with each other in one *connection*. Even if data is not being exchanged, the devices must exchange a packet periodically to ensure the *connection* is still alive. The start of every period in which two devices can exchange information with one another is known as a *Connection Event*. A connection itself is thus a sequence of Connection Events.
- **Connection Interval** – The time between each *Connection Event*. The *Connection Interval* can be negotiated once the two devices are connected. Longer connection intervals save power at the cost of latency. *Connection Intervals* can range from *7.5ms* to *4s*.

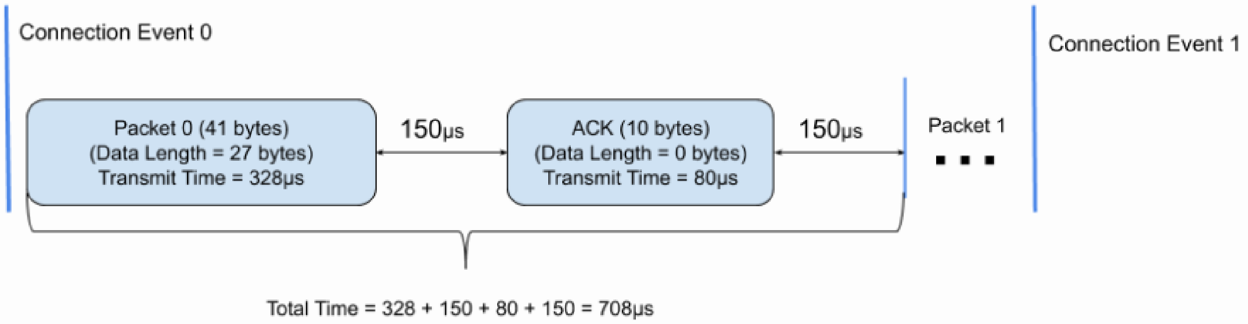
Note that in BLE, the two devices are always talking back and forth. Even if only 1 side is mainly transmitting, the other side must send an ACK. Also, Inter Frame Space (IFS) of at least 150 µsec is needed. See below for a generic Bluetooth 4.0 connection event. Note that data length is at most 27 bytes in Bluetooth 4.0 packet. This is called the MTU (Maximum Transmission Unit). Of that **MTU of 27 bytes** only 20 bytes are for real payload. Data rate is 1 Mbit/sec. Therefore, each packet of 41 bytes requires 41 x 8 = 328 µsec to send.

The following is a Bluetooth 4.0 Link Layer packet:





Copyright Convergence Systems Limited 2023 All Rights Reserved



The time it takes to transmit one packet can be computed as:

```
328µs data packet + 150µs T_IFS + 80µs ACK + 150µs T_IFS = 708µs
```

During this time period, **27** bytes of actual data can be transmitted which takes **216µs**.

This yields a raw data throughput of:

```
(216µs / 708µs) * 1Mbps = 305,084 bits/second = ~0.381 Mbps
```

So how good is 305 Kbit/sec in terms of sending RFID tag data back to host processor?

Assuming each tag inventory packet is 42 bytes in length, i.e., 336 bits, then one can send 1,347 tag inventory data back to host processor.

In reality, this is not achievable.

The reason is that there is a limit on the number of packets you can send during each connection interval. For example, typically a 15 msec connection interval is allowed in mobile phone, and then **within each connection interval, in Bluetooth 4.0, only 4 link layer packets are allowed with mobile phones in the market.** Moreover, the 27 bytes are not all dedicated to payload. 7 bytes are used as header. So the throughput is really only about 20 bytes per packet.

After calculating that, the resultant throughput is 5333 bytes per second. Assuming 42 bytes per tag, then total tag per second is 126. In a laboratory environment and in an anechoic chamber, one may be able to reach 120 tags per second. In real world environment (from our experience of using that), due to Bluetooth jamming from nearby users, we can often only be able to send 30 tags per second. That is about one quarter of the time. In other words, the efficiency due to a real world noisy environment is only about 25%.

Copyright Convergence Systems Limited 2023 All Rights Reserved

In Bluetooth 5.0, up to 16 packets are allowed per connection interval. Each packet can contain up to 251 bytes of payload, i.e., the MTU is 251 bytes. Of that Bluetooth 5.0 can send true payload of 244 bytes. Within that since CSL Bluetooth protocol needs to send a header of 8 byte, so the final tag related data (tag payload) that can be sent is 236 byte. As shown below, total bytes in a packet can be 265 bytes for 1M PHY and 266 bytes for 2M PHY. 265 bytes for 1M PHY takes up 2120 μsec, i.e., 2.12 msec. 266 bytes for 2M PHY takes up 1064 μsec, i.e. 1.06 msec.

Assuming connection interval is 15 msec, then maximum packets for 1M PHY can only be about 7, maximum packets for 2M PHY can be about 14. This number, however, needs to be configured in the firmware of the Bluetooth IC, and on the mobile phone side, if possible, by the driver.

Based on 2M PHY, then the total bytes transferred per second is $14 \times 238 \times (1 / 15 \times 10^{-3}) = 222{,}133$. Then the total number of tags is that number divided by 42, approximately 5,288 tags per second. Assuming there is a jamming resulting in only 25% successful transfer, then it is 1,322 tags per second.

Based on 1M PHY, then the total bytes transferred per second is $7 \times 238 \times (1 / 15 \times 10^{-3}) = 111{,}066$. Total number of tags per second is 2,644. 25% jamming induced efficiency means 661 tags per second.

Packet format for Uncoded LE data packets

<table>
<tr><th>Preamble</th><th>Access Address</th><th colspan="6">PDU (2-257 bytes)</th><th>CRC</th></tr>
<tr><td rowspan="5">1 byte
(1M PHY)
2 bytes
(2M PHY)</td><td rowspan="5">4 bytes</td><th>LL Header</th><th colspan="4">Payload (0-251 bytes)</th><th rowspan="2">MIC
(Optional)</th><td rowspan="5">3 bytes</td></tr>
<tr><td rowspan="4">2 bytes</td><th>L2CAP
Header</th><th colspan="3">ATT Data (0-247 bytes)</th></tr>
<tr><td rowspan="3">4 bytes</td><th colspan="2">ATT Header</th><th>ATT Payload</th><td rowspan="3">4 bytes</td></tr>
<tr><th>Op Code</th><th>Attribute Handle</th><td rowspan="2">Up to 244 bytes</td></tr>
<tr><td>1 byte</td><td>2 bytes</td></tr>
</table>

The above calculation assumes the mobile phone would dedicate all the time within a connection interval to actual LE data packets. **This is HIGHLY UNLIKELY.** The mobile phone needs to use the BLE time to handle other devices, check other broadcast packets nearby, etc. These maintenance stuffs will take up time as well. Hence it is highly unlikely a mobile phone would allocate all the time available for LE data packets. Some comment on the Internet shows developer trying to set up long LE data packets but was only able to get 1 to 2 of these long packets per 20 msec of connection interval. Some configurations are needed to get more packets.

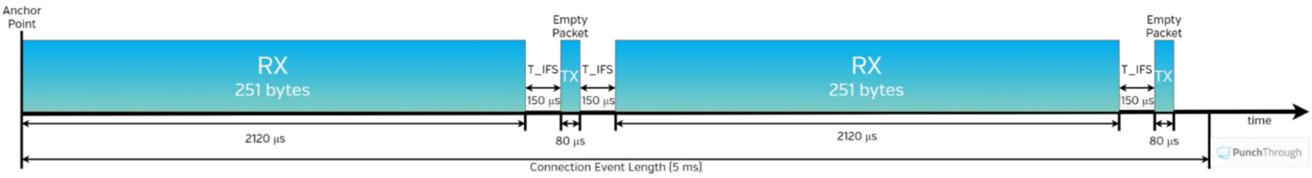
Here is an example with Nordic Bluetooth device how to maximize the number of packets per connection interval:

Copyright Convergence Systems Limited 2023 All Rights Reserved

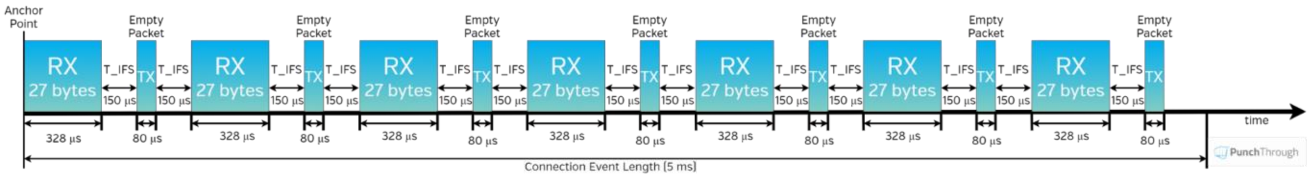
[How to determine the number of packets per connection interval – Nordic Q&A – Nordic DevZone – Nordic DevZone (nordicsemi.com)]

Data Length Extension (DLE):

An example of a Connection event with Data Length Extension enabled:

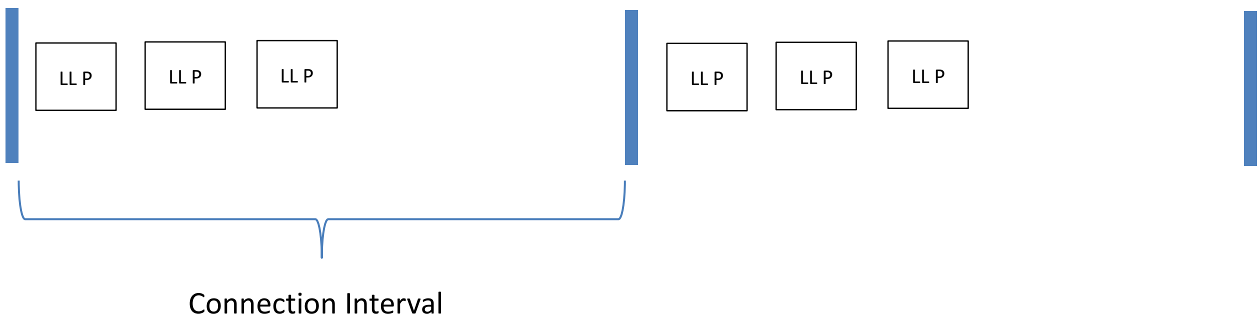


An example of a Connection Event **without** Data Length Extension:



From the above 2 examples, it is clear that the efficiency is much higher when DLE is enabled. Less time is wasted in the acknowledgement (which is an empty packet), and less time is wasted in the Inter Frame Space (IFS), and less time is wasted in the overhead in each packet.

There is one parameter that is difficult to control: how much time within each Connection Interval would the Master allow Link Layer packets interchange to take place? Since this is decided in the Master, which in our case is the Mobile Phone, it is hard to control:



Copyright Convergence Systems Limited 2023 All Rights Reserved

The connection interval effectively determines how many packets can be sent during one connection event. The higher the value, the more packets can be sent in one connection event (up to a certain limit for some devices).

However, the number of packets per connection event depends on the device and BLE stack so it is limited and differs between devices and stack versions on a specific device. This value also depends on the operation of the device, so the radio may have to attend to other events and the number of packets sent per connection events may not reach the max allowed by the stack. For example, the number differs between iOS and Android and also changes depending on the version of the OS running on the device.

Copyright Convergence Systems Limited 2023 All Rights Reserved

To achieve maximum possible transfer rate, one has to:

1) **Enable DLE,**
2) **Configure highest possible MTU value = 251 (maximum possible is 251)**
3) **Choose LE 2M PHY,**
4) **Configure master to allow maximum number of packets per connection interval (maximum communication time) – depends on specific mobile phone and its OS.**

Here are the details:

2 Enable DLE (Data Length Extension)

How to Enable DLE

How you'll actually enable DLE in your project or application will depend on the chip, stack and SDK you're running on, but at a lower level, the following needs to happen:

1. Both the master and the slave must support the Bluetooth 4.2 specification or newer (see the LL_VERSION_IND exchange)
2. Both the master and the slave must support the LE Data Packet Length Extension Link Layer feature (see the LL_FEATURE_REQ and LL_FEATURE_RSP exchange)
3. Either master or slave must initiate the actual Data Length Update Procedure by sending an **LL_LENGTH_REQ** command, which must be responded to with an **LL_LENGTH_RSP** command to complete the procedure. The following four values are negotiated between the master and slave during the Data Length Update Procedure:
4. connMaxTxOctets – the max number of payload bytes/octets that the stack can send in a single link layer data packet
5. connMaxTxTime – the length of time in microseconds that the stack can be actively transmitting a single link layer data packet
6. connMaxRxOctets- the max number of payload bytes/octets that the stack can receive in a single link layer data packet
7. connMaxRxTime- the length of time in microseconds that the stack can be actively receiving a single link layer data packet

Either the Master or the Peripheral can initiate this DLE configuration. From the slave, which is the Texas Instruments Bluetooth IC, the TI API need to be used.

Note that the default value of CC2652 BT 5.0 stack is such the transmit direction and the receive direction are different. So that has to be carefully handled.

Copyright Convergence Systems Limited 2023 All Rights Reserved

In the Peripheral, i.e., the Texas Instrument Bluetooth IC,

Take the following steps to configure the stack to support larger MTU values.

1. Use **LL_LENGTH_REQ** to set the `MAX_PDU_SIZE` preprocessor symbol in the application project to the desired value to the maximum desired size of the L2CAP PDU size. The maximum is set by the following equation `ATT_MTU=MAX_PDU_SIZE-L2CAP_HDR_SIZE`
2. Call GATT_ExchangeMTU() after a connection is formed (GATT client only). The MTU parameter passed into this function must be less than or equal to the definition from step 1.
3. Receive the `ATT_MTU_UPDATED_EVENT` in the calling task to verify that the MTU was successfully updated. This update requires the calling task to have registered for GATT messages. See Registering to Receive Additional GATT Events in the Application for more information.

Though the stack can be configured to support a `MAX_PDU_SIZE` up to 255 bytes, each Bluetooth Low Energy connection initially uses the default 27 bytes (`ATT_MTU` = 23 bytes) value until the exchange MTU procedure results in a larger MTU size. The exchange MTU procedure must be performed on each Bluetooth Low Energy connection and must be initiated by the client.

In the master, which may be an Android phone, an iPhone, etc., appropriate APIs need to be invoked to configure the above.

Copyright Convergence Systems Limited 2023 All Rights Reserved

2) Configure MTU value = 251 (maximum possible is 251)

MTU value needs to be configured.

For mobile phone, to configure MTU:

For example, the following is the Android API that requests an MTU size for a given connection:

requestMtu

Added in API level 21

```
public boolean requestMtu (int mtu)
```

Request an MTU size used for a given connection.

When performing a write request operation (write without response), the data sent is truncated to the MTU size. This function may be used to request a larger MTU size to be able to send more data at once.

A `BluetoothGattCallback#onMtuChanged` callback will indicate whether this operation was successful.

Requires `Manifest.permission.BLUETOOTH` permission.

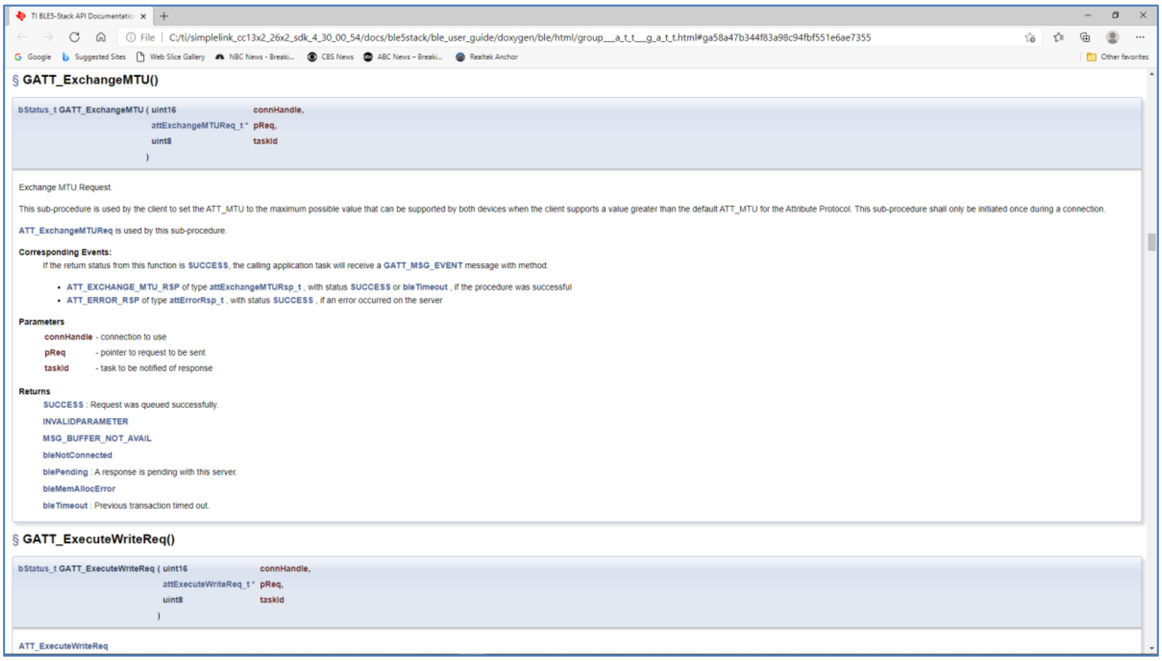
Parameters	
mtu	int

Returns	
boolean	true, if the new MTU value has been requested successfully

Copyright Convergence Systems Limited 2023 All Rights Reserved

For CS710S side, for CC2652, to configure MTU:

One can initiate a GATT_ExchangeMTU() request. This will automatically try to achieve a biggest possible MTU. Or one can do a GATT_UpdateMTU() to try increase its size.



TI BLE5-Stack API Documentatio

C:/ti/simplelink_cc13x2_26x2_sdk_4_30_00_54/docs/ble5stack/ble_user_guide/doxygen/ble/html/group__a_t_t__g_a_t_t.html#ga58a47b344f83a98c94fbf551e6ae7355

§ GATT_ExchangeMTU()

```
bStatus_t GATT_ExchangeMTU ( uint16                 connHandle,
                             attExchangeMTUReq_t *  pReq,
                             uint8                  taskId
                           )
```

Exchange MTU Request.

This sub-procedure is used by the client to set the ATT_MTU to the maximum possible value that can be supported by both devices when the client supports a value greater than the default ATT_MTU for the Attribute Protocol. This sub-procedure shall only be initiated once during a connection.

ATT_ExchangeMTUReq is used by this sub-procedure.

Corresponding Events:

If the return status from this function is SUCCESS, the calling application task will receive a GATT_MSG_EVENT message with method:

- ATT_EXCHANGE_MTU_RSP of type attExchangeMTURsp_t , with status SUCCESS or bleTimeout , if the procedure was successful
- ATT_ERROR_RSP of type attErrorRsp_t , with status SUCCESS , if an error occurred on the server

Parameters

connHandle - connection to use
pReq - pointer to request to be sent
taskId - task to be notified of response

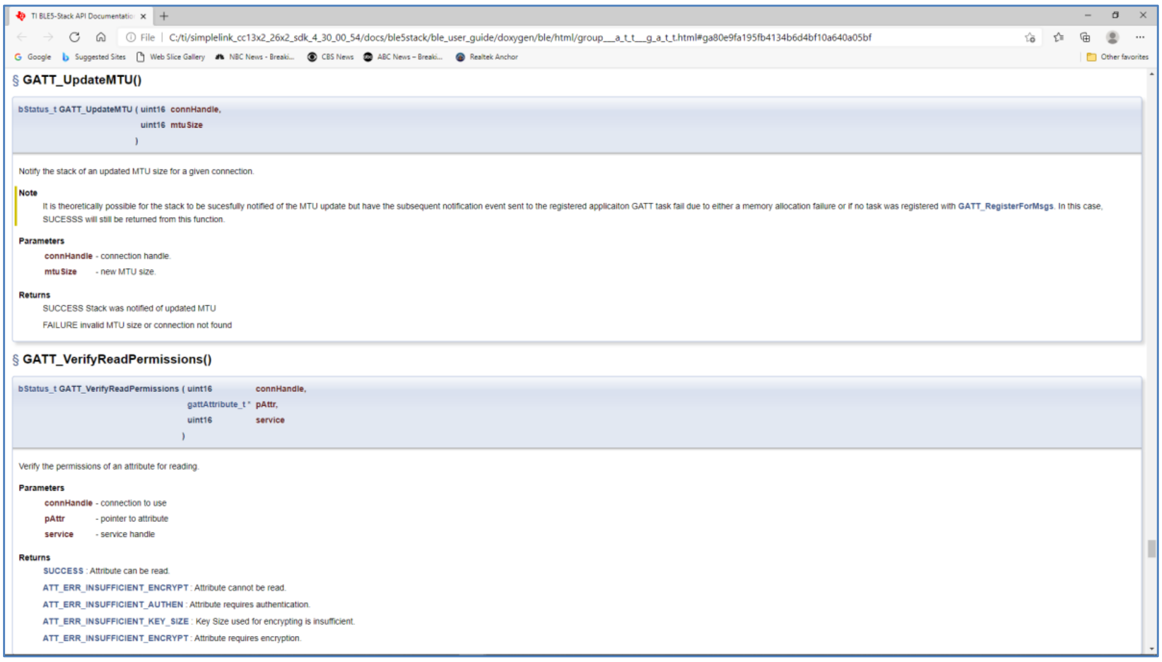
Returns

SUCCESS : Request was queued successfully.
INVALIDPARAMETER
MSG_BUFFER_NOT_AVAIL
bleNotConnected
blePending : A response is pending with this server.
bleMemAllocError
bleTimeout : Previous transaction timed out.

§ GATT_ExecuteWriteReq()

```
bStatus_t GATT_ExecuteWriteReq ( uint16                  connHandle,
                                 attExecuteWriteReq_t *  pReq,
                                 uint8                   taskId
                               )
```

ATT_ExecuteWriteReq



TI BLE5-Stack API Documentatio

C:/ti/simplelink_cc13x2_26x2_sdk_4_30_00_54/docs/ble5stack/ble_user_guide/doxygen/ble/html/group__a_t_t__g_a_t_t.html#ga80e9fa195fb4134b6d4bf10a640a05bf

§ GATT_UpdateMTU()

```
bStatus_t GATT_UpdateMTU ( uint16  connHandle,
                           uint16  mtuSize
                         )
```

Notify the stack of an updated MTU size for a given connection.

Note

It is theoretically possible for the stack to be sucesfully notified of the MTU update but have the subsequent notification event sent to the registered applicaiton GATT task fail due to either a memory allocation failure or if no task was registered with GATT_RegisterForMsgs. In this case, SUCESSS will still be returned from this function.

Parameters

connHandle - connection handle.
mtuSize - new MTU size.

Returns

SUCCESS Stack was notified of updated MTU
FAILURE invalid MTU size or connection not found

§ GATT_VerifyReadPermissions()

```
bStatus_t GATT_VerifyReadPermissions ( uint16             connHandle,
                                       gattAttribute_t *  pAttr,
                                       uint16             service
                                     )
```

Verify the permissions of an attribute for reading.

Parameters

connHandle - connection to use
pAttr - pointer to attribute
service - service handle

Returns

SUCCESS : Attribute can be read.
ATT_ERR_INSUFFICIENT_ENCRYPT : Attribute cannot be read.
ATT_ERR_INSUFFICIENT_AUTHEN : Attribute requires authentication.
ATT_ERR_INSUFFICIENT_KEY_SIZE : Key Size used for encrypting is insufficient.
ATT_ERR_INSUFFICIENT_ENCRYPT : Attribute requires encryption.

Copyright Convergence Systems Limited 2023 All Rights Reserved

3 Choose LE 2M PHY,

With 2M PHY, the time for each LL packet becomes shorter by one half for the same number of bytes. This is useful both in overall transfer throughput and also better jamming avoidance.

In mobile phones:

In Android, you can invoke the following to set it to LE 2M PHY:

PHY_LE_2M

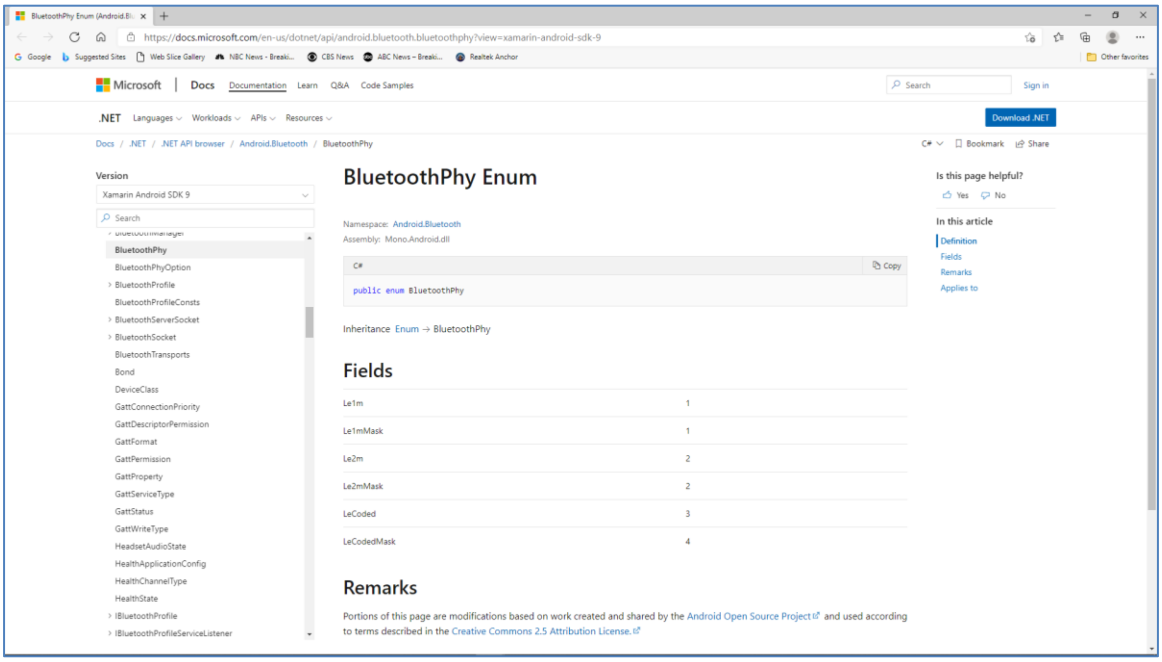
Added in API level 26

```
public static final int PHY_LE_2M
```

Bluetooth LE 2M PHY. Used to refer to LE 2M Physical Channel for advertising, scanning or connection.

Constant Value: 2 (0x00000002)

In Xamarin, you can invoke the following to set it to LE 2M PHY:



Copyright Convergence Systems Limited 2023 All Rights Reserved

On the CS710S side, in CC2652,

You can set the default PHY:

§ HCI_LE_SetDefaultPhyCmd()

```
hciStatus_t HCI_LE_SetDefaultPhyCmd ( uint8  allPhys,
                                      uint8  txPhy,
                                      uint8  rxPhy
                                    )
```

Set Default PHY

Allows the Host to specify its preferred values for the transmitter and receiver PHY to be used for all subsequent connections.

Corresponding Events

hciEvt_CmdComplete_t with cmdOpcode HCI_LE_SET_DEFAULT_PHY

Parameters

allPhys	Host preference on how to handle txPhy and rxPhy.
txPhy	Bit field of Host preferred Tx PHY. See 2 Mbps & Coded PHY
rxPhy	Bit field of Host preferred Rx PHY. See 2 Mbps & Coded PHY

Returns

HCI_SUCCESS

Copyright Convergence Systems Limited 2023 All Rights Reserved

4 Configure master to allow maximum number of packets per connection interval (maximum communication time).

This is the most difficult part.

Copyright Convergence Systems Limited 2023 All Rights Reserved

4.1.1 CC2652 Family Bluetooth IC DLE Configuration

The data length extension feature allows the LE controller to send data channel packet data units (PDUs) with payloads of up to 251 bytes of application data, while in the connected state. Furthermore, a new PDU size can be negotiated by either side at any time during a connection.

Previously, the controller's largest data channel payload was 27 bytes. This Feature increases the data rate by around 250% when compared to Bluetooth Core Specification Versions 4.0 and 4.1 devices (if both devices support extended packet length and are configured properly).

The CC13x2 or CC26x2 has Data Length Extension enabled by default – allowing peer devices to utilize this feature with no application overhead.

DLE Update Procedure and Definitions
Default Application DLE Behavior
Utilizing DLE in the Application
Disabling DLE at Runtime
Interoperability with Legacy Peers
RAM Considerations when using DLE
DLE Update Procedure and Definitions
This section describes what is done from a controller perspective during a connection as well as terminology.

Once a connection is formed, the controller will behave in one of two possible ways:

If prior to the connection, the suggested PDU size and time are set to the defaults for both TX and RX (27B, 328 us) then the CC13x2 or CC26x2 will not initiate a data length exchange (i.e. a LL_LENGTH_REQ will not be sent).

If the peer device sends a LL_LENGTH_REQ then the controller of the device will send a LL_LENGTH_RSP corresponding to the default sizes of 4.0 devices autonomously.

Note

See Disabling DLE at Runtime for information on how to modify this behavior.

If prior to the connection, the PDU size or the maximum time for RX or TX are not default, then the LE controller of the device will use the LL_LENGTH_REQ and LL_LENGTH_RSP control PDUs to

Copyright Convergence Systems Limited 2023 All Rights Reserved

negotiate a larger payload size for data channel PDUs.

A data length update may be initiated by the host or performed autonomously by the controller. Either the master or the slave can initiate the procedure.

After the data length update procedure is complete, both controllers select a new data length based on two parameters: PDU size and time. The largest size supported by both local and remote controller is selected; time is taken into account to support different data rates. These parameters are defined below:

PDU size
The largest application data payload size supported by the controller. This size does not include packet overhead, such as access address or preamble.

Time
The maximum number of microseconds that the device takes to transmit or receive a PDU at the PHY rate. This parameter uses units of microseconds (us).

Each direction has a PDU size and time; in other words there is a Receive PDU size/time and a separate Transmit PDU size/time. A device can only influence a peer's Receive PDU size/time by adjusting it's own Transmit PDU size/time via the DLE Update Procedure.

Reference ([Vol 6], Part B, Section 5.1.9) of the Bluetooth Core Specification Version 5.1 for more information about the data length update procedure.

Reference ([Vol 6], Part B, Section 4.5.10) of the Bluetooth Core Specification Version 5.1 for information on the valid ranges for data PDU length and timing parameters.

Default Application DLE Behavior
This section describes the default behavior of the CC13x2 or CC26x2 due to the feature being enabled by default.

The controller defaults to using TX PDU sizes compatible with 4.0 and 4.1 devices. It uses 27 bytes as its initial maximum PDU size, and 328 us as the maximum PDU transmit time.

On the RX PDU size and time, the controller defaults to the maximum PDU size and the maximum PDU transit time for a LE Data Packet Length Extension enabled device. In other words, the RX PDU size will be 251, and the RX PDU transmit time will be 2120 us.

Copyright Convergence Systems Limited 2023 All Rights Reserved

Note

As mentioned in DLE Update Procedure and Definitions, by default a LL_LENGTH_REQ control packet will be sent due to the RX max PDU size and max PDU transmit time not being default 4.0 PDU sizes and timings.

Utilizing DLE in the Application
This section describes how the application can influence the controller to use DLE for transmission of data at runtime.

The application can update the data length in two ways.

The application can set the connection initial TX PDU size or time to cause the controller to request the peer's RX PDU size and time to change for every connection.
The controller can initialize the connection with the default values of 27 octets and 328 us, then dynamically negotiate the data length at a later time in the connection using HCI commands.
For maximum throughput, high layer protocols such as the BLE host should also use a larger PDU size (see Maximum Transmission Unit (MTU)). See Link Layer Buffers for more information how the link layer manages buffers and PDUs.

The following HCI commands can be used to interact with the controller related to the data length extension feature:

LE Read Suggested Default Data Length Command (HCI_LE_ReadSuggestedDefaultDataLenCmd())
LE Write Suggested Default Data Length Command (HCI_LE_WriteSuggestedDefaultDataLenCmd())
LE Read Maximum Data Length Command (HCI_LE_ReadMaxDataLenCmd())
LE Set Data Length Command (HCI_LE_SetDataLenCmd())
The above commands may generate:

LE Data Length Change Event
For example, to dynamically change the TX PDU size and timing, the command HCI_LE_SetDataLenCmd() during a connection. This will cause the LE controller to negotiate with the peer's LE controller to adjust it's RX PDU size and timing as described in DLE Update Procedure and Definitions.

Uint16_t cxnHandle; //Request max supported size
uint16_t requestedPDUSize = 251;

Copyright Convergence Systems Limited 2023 All Rights Reserved

```
uint16_t requestedTxTime = 2120;

GAPRole_GetParameter(GAPROLE_CONNHANDLE, &cxnHandle); //This API is documented in hci.h

HCI_LE_SetDataLenCmd(cxnHandle, requestedPDUSize, requestedTxTime);
```

Note

For more information about these HCI commands and their fields, see the LE Controller Commands and Events sections ([Vol 2], Part E, Section 7.7-7.8) of the Bluetooth Core Specification Version 5.1. Additionally, the APIs for these commands are documented under BLE Stack API Reference.

Copyright Convergence Systems Limited 2023 All Rights Reserved

4.2 USB Connection Details

USB HID library on the host platform is used. The protocol using USB connection as described in this document is only for CS710S.

Copyright Convergence Systems Limited 2023 All Rights Reserved

Chapter 5: LED definitions

There are 7 LEDs on the CS710S device:

1) Power on LED: Turn on whenever power switch turns on power
2) Charging LED: Turn on whenever battery is being charged. Brighter when battery is more drained and charging current is higher.
3) External Power LED: Turn on whenever external power is available and used.
4) RFID LED: turned on continuous when RFID power is on. Flashes when tags are being read – (inventory started).
5) Barcode LED: turned on continuous when BARCODE power is on. Flashes when barcodes are being read – (barcode reading started).
6) Status LED: status of Atmel IC
7) Bluetooth LED: status of Bluetooth – on (waiting for pairing), off, pairing, paired
    1. When Bluetooth push button is pressed for 3 seconds, Bluetooth LED starts slow flashing, 1 second on, 1 second off. Unit is now ready for discovery and pairing by smart phone
    2. During pairing, fast flash, 0.5 second on, 0.5 second off
    3. When successful paired, Bluetooth LED changes to continuous on
    4. Anytime when Bluetooth is on (Bluetooth LED either slow flashing or fast flashing or continuous), by long pressing Bluetooth push button for 3 seconds Bluetooth LED will be turned off. Bluetooth will be turned off.

Copyright Convergence Systems Limited 2023 All Rights Reserved

Chapter 6: Push buttons Definition

There are 2 buttons on the CS710S device:

1) Power button: press power button continuously for 3 seconds, release button, after that power will turn on with power LED light up. Press power button continuously for 3 seconds, release button, then it will power off with power LED turned off.

2) Bluetooth button: this button controls Bluetooth on, off and pairing.

    1. When Bluetooth push button is pressed for 3 seconds, Bluetooth LED starts slow flashing, 1 second on, 1 second off. Unit is now ready for pairing by smart phone
    2. Anytime when Bluetooth is on (Bluetooth LED either slow flashing or fast flashing or continuous), by long pressing Bluetooth push button for 3 seconds Bluetooth LED will be turned off. Bluetooth will be turned off.

Copyright Convergence Systems Limited 2023 All Rights Reserved

Chapter 7: API Set Definition

There are 5 types of commands/responses/notifications API. All of them are byte streams:

1) RFID Reader
2) Barcode Reader
3) Special Notifications
4) Atmel IC
5) Bluetooth IC

5 API Format

API consists of an 8-byte header and a payload.

Header (8 bytes)	Payload (maximum 240 bytes)

The header contains information of whether it is a command/reply/notification, downlink or uplink, the length of the payload inside, CRC of packet.

Header Format:

Prefix (1 byte)	Connection (1 byte)	Payload Length (1 byte)	Destination/ Source (1 byte)	Reserve (1 byte)	Direction (1 byte)	CRC of packet (2 bytes)

Header Field:

Name	Length (byte)	Value – Description.
Prefix	1	0xA7
Connection	1	0xE6 – USB 0xB3 – Bluetooth
Payload Length	1	1 to 240 – Payload length
Destination/Source	1	0xC2 – RFID 0x6A – Barcode 0xD9 – Notification 0xE8 – ATMEL IC

Copyright Convergence Systems Limited 2023 All Rights Reserved

		0x5F – Bluetooth IC
Reserve	1	0x82 in most cases, **but for RFID Uplink (byte 3=C2, byte 5=9E), byte , this is a sequence number incrementing from 0 to 255 and repeat.** Application needs to store this sequence number. It should take in the first number and then track it as it increments.
Direction	1	0x37 – Downlink 0x9E – Uplink
CRC	2	CRC of packet

Notes:

1. CRC is not used when value is zero. For downlink, no need to use.
2. Payload length cannot be greater than 240.
3. Downlink means from smart phone/PC toward RFID reader. Uplink means RFID reader toward smart phone/PC

In summary, there are 5 types of commands (each can have downlink or uplink direction):

1	RFID Reader Command Header	RFID Reader Command Payload
2	Barcode Reader Command Header	RFID Reader Command Payload
3	Special Notification Command Header	Special Notification Command Payload
4	ATMEL IC Command Header	ATMEL IC Command Payload
5	Bluetooth IC Command Header	Bluetooth IC Command Payload

Copyright Convergence Systems Limited 2023 All Rights Reserved

7.2: Header of RFID Reader Commands

Downlink commands

Header is 8 byte long:

A7	B3/E6	Payload Length (1 byte)	C2	82	37	CRC of packet byte 1	CRC of packet byte 2

Payload is RFID reader commands. Please refer to Chapter 8 and Appendix A for definitions of RFID reader commands payload.

Uplink replies/notifications

(Notifications include tag returns)

Header is 8 byte long:

A7	B3/E6	Payload Length (1 byte)	C2	82	9E	CRC of packet byte 1	CRC of packet byte 2

Payload is RFID can be reader command replies and tag data. Please refer to Chapter 8 and Appendix A for definitions of payload.

Copyright Convergence Systems Limited 2023 All Rights Reserved

7.3: Header of Barcode Reader Commands

Downlink commands

Header is:

A7	B3/E6	Payload Length (1 byte)	6A	82	37	CRC of packet byte 1	CRC of packet byte 2

Payload is barcode reader commands. Please refer to Chapter 9 for definitions of payload.

Uplink replies/notifications

(Notifications include barcode returns)

Header is:

A7	B3/E6	Payload Length (1 byte)	6A	82	9E	CRC of packet byte 1	CRC of packet byte 2

Payload is barcode command replies and barcodes. Please refer to Chapter 9 for definitions of payload.

Copyright Convergence Systems Limited 2023 All Rights Reserved

7.4: Header of Special Notifications

Downlink commands

Header is:

A7	B3/E6	Payload Length (1 byte)	D9	82	37	CRC of packet byte 1	CRC of packet byte 2

Payload is special notification. Please refer to Chapter 10 for definitions of payload.

Uplink replies/notifications

(Notifications include special notifications)

Header is:

A7	B3/E6	Payload Length (1 byte)	D9	82	9E	CRC of packet byte 1	CRC of packet byte 2

Payload is special notifications. Please refer to Chapter 10 for definitions of payload.

Copyright Convergence Systems Limited 2023 All Rights Reserved

7.5: Header of ATMEL IC Commands

Downlink commands

Header is:

A7	B3/E6	Payload Length (1 byte)	E8	82	37	CRC of packet byte 1	CRC of packet byte 2

Payload is Atmel IC commands. Please refer to Chapter 11 for definitions of payload.

Uplink replies/notifications

Header is:

A7	B3/E6	Payload Length (1 byte)	E8	82	9E	CRC of packet byte 1	CRC of packet byte 2

Payload is Atmel Ics notifications. Please refer to Chapter 11 for definitions of payload.

Copyright Convergence Systems Limited 2023 All Rights Reserved

7.6: Header of Bluetooth IC Commands

Downlink commands

Header is:

A7	B3/E6	Payload Length (1 byte)	5F	82	37	CRC of packet byte 1	CRC of packet byte 2

Payload is Bluetooth IC commands. Please refer to Chapter 12 for definitions of payload.

Uplink replies/notifications

Header is:

A7	B3/E6	Payload Length (1 byte)	5F	82	9E	CRC of packet byte 1	CRC of packet byte 2

Payload is Bluetooth IC replies and notifications. Please refer to Chapter 12 for definitions of payload.

Copyright Convergence Systems Limited 2023 All Rights Reserved

Chapter 8: Payload – RFID Reader Commands

This chapter describes the payload for RFID reader operations. There are various downlink commands and uplink replies, plus uplink tag data and reader status notifications.

Payload Format:

Event code (2 bytes)	Data

Reply Format:

Event code (2 bytes)	Status

8.1 Downlink Payload

Payload Field

Event code	Data length in byte(s)	Event/Data Description.
0x8000	0	RFID reader power on.
0x8001	0	RFID reader power off.
0x8002	Depends on payload length	RFID firmware command data. See Appendix A
0x8003	238	Image raw data. Auto update when subpart = total number of subparts. Byte 0-1: total number of subparts Byte 2-3: subpart index starting from 1 Byte 4-237: Image raw data
0x8004	238	OEM raw data. Restore OEM when subpart = total number of subparts. Byte 0-1: total number of subparts Byte 2-3: subpart index starting from 1 Byte 4-237: OEM raw data

Copyright Convergence Systems Limited 2023 All Rights Reserved

Reply Payload Field

Event code	Status length in byte(s)	Status Description.
0x8000	1	Status: 0x00 – Power on success 0xFF – Failure with unknown reason
0x8001	1	Status: 0x00 – Power off success 0xFF – Failure with unknown reason
0x8002	1	Status: 0x00 – Success 0xFF – Failure with unknown reason
0x8003	1	0x00 – success 0x01 – failure 0x02 – full image received successfully
0x8004	1	0x00 – success 0x01 – failure 0x02 – full OEM data received successfully

8.2 Uplink Payload

Event code	Data length in byte(s)	Event/Data Description.
0x8100	Depends on payload length	RFID firmware response data. See Appendix A

Note that uplink payload can be:

1) Command reply
2) Tag data - when it is tag data, the header reserve byte contains the sequence number

Copyright Convergence Systems Limited 2023 All Rights Reserved

Chapter 9: Payload – Barcode Reader Commands

This chapter describes the payload for Barcode reader operation. The only downlink Barcode commands are barcode power on and power off. Once the barcode reader is power on, any press of the push button at the gun handle will start barcode reading. Any successful barcode capture will then be sent back to the originating source of the commands – either Bluetooth or USB.

Payload Format:

Event code (2 bytes)	Data

Reply Format:

Event code (2 bytes)	Status

Copyright Convergence Systems Limited 2023 All Rights Reserved

9.1 Downlink Payload

Payload Field

Event code	Data length in byte(s)	Event/Data Description.
0x9000	0	Barcode reader power on.
0x9001	0	Barcode reader power off.
0x9002	0	Trigger to start a scan
0x9003	1-50	Command data* sent to barcode device
0x9004	3	Vibrator on. Byte 0: Mode: 0 = Normal; 1 = Inventory on; 2 = Barcode good read on. Byte 1-2: On time in ms (all 0's mean forever)
0x9005	0	Vibrator off. Reset all modes in 0x9004

*for command data of barcode reader, please reference "Serial_Programming_Command_Manual_V1.3.1" from Newland (barcode module manufacturer)

Reply Payload Field

Event code	Status length in byte(s)	Status Description.
0x9000	1	Status: 0x00 – Power on success 0xFF – Failure with unknown reason
0x9001	1	Status: 0x00 – Power off success 0xFF – Failure with unknown reason
0x9002	1	Status: 0x00 – Trigger success 0x01 – Failure with barcode not powered on 0x02 – Failure with previous scan not ended 0xFF – Failure with unknown reason
0x9004	1	Status: 0x00 – Vibrator on success 0xFF – Failure with unknown reason
0x9005	1	Status: 0x00 –Vibrator off success 0xFF – Failure with unknown reason

Copyright Convergence Systems Limited 2023 All Rights Reserved

9.2 Uplink Payload

Event code	Data length in byte(s)	Event/Data Description.
0x9100	Depends on payload length	Barcode read/reply data.
0x9101	0	Good read

9.2.1 Uplink Barcode Payload Format

In the CS710S barcode scanner, the format of uplink "barcode successfully scanned" payload is as follows:

Prefix, Barcode, Suffix

Prefix = Self-prefix + Code ID + AIM ID
Suffix = Self-suffix

Self-prefix = STX, Null, BEL, BS, SI, VT (encoded by 39sci table as 020007101713)
Self-suffix = ENQ, SOH, TAB, SO, ETX, EOT (encoded by 39sci table as 050111160304)
(you can find the corresponding encoding from 39sci table)

Code ID, 1 byte, is a commonly used code that defines the type of barcode.
AIM ID, 3 bytes, is another commonly used code that defines the type of barcode.

Some users like to use Code ID, some users like to use AIM ID, so both of them are included here in the barcode scanner return.

Copyright Convergence Systems Limited 2023 All Rights Reserved

Chapter 10: Payload – Special Notifications

This chapter describes the payload for special notifications.

Downlink Payload Format:

Event code (2 bytes)	Data (variable bytes)

Reply Payload Format:

Event code (2 bytes)	Data (variable bytes)

Copyright Convergence Systems Limited 2023 All Rights Reserved

10.1 Downlink Payload

Payload Field

Event code	Data length in byte(s)	Event/Data Description.
0xA000	0	Get current battery voltage.
0xA001	0	Get current trigger state
0xA002	0	Start battery 5 seconds auto reporting (for BT connection only)
0xA003	0	Stop battery auto reporting
0xA004	1	1=invoke RFID Abort (default); 0= Not invoke Set Trigger Release Not to invoke Abort RFID (default is invoke RFID)
0xA005	0	Get Trigger Release Setting (1 = invoke RFID Abort, 0 = Not invoke RFID Abort)
0xA006	1	Set “Fast Trigger Button Barcode Scanning” mode, once in this mode, just press trigger button the reader will itself send 0x1b, 0x33 command to barcode engine, and release trigger will send 0x1b, 0x30 command to barcode engine 0 = off (default); 1=on
0xA007	0	Get “Fast Trigger Button Barcode Scanning” mode setting
0xA008	1	Start trigger state auto reporting (for BT connection only) 1 byte value = interval in second
0xA009	0	Stop trigger state auto reporting

Reply Payload Field

Event code	Data length in byte(s)	Data Description.
0xA000	2	Current battery voltage in mV Value 0xFFFF = battery fault
0xA001	1	0 = Released; 1 = Pushed
0xA002	1	0
0xA003	1	0
0xA004	1	0=success; 1=failure
0xA005	1	1 = invoke RFID Abort, 0 = Not invoke RFID Abort

Copyright Convergence Systems Limited 2023 All Rights Reserved

0xA006	1	0=success; 1=failure
0xA007	1	0 = off; 1=on

10.2 Uplink Payload

Payload Field:

Event code	Data length in byte(s)	Event/Data Description.
0xA100	0	Reserve
0xA101	2	Error code 0x0000 – Wrong header prefix 0x0001 – Payload length too large 0x0002 – Unknown target 0x0003 – Unknown event
0xA102	0	Trigger is pushed
0xA103	0	Trigger is released

Copyright Convergence Systems Limited 2023 All Rights Reserved

Chapter 11: Payload – ATMEL IC Commands

This chapter describes the payload for Atmel IC command.

Payload Format:

Event code (2 bytes)	Data

Reply Payload Format:

Event code (2 bytes)	Data

11.1 Downlink Payload

Payload Field

Event code	Data length in byte(s)	Event/Data Description.
0xB000	0	Get Atmel IC firmware version.
0xB001	238	Image raw data. Auto update when subpart = total number of subparts. Byte 0-1: total number of subparts Byte 2-3: subpart index starting from 1 Byte 4-237: Image raw data
0xB004	0	Get 32-byte serial number. (Null terminated)
0xB005	32	Set 32-byte serial number. (Null terminated)
0xB006	0	Get 32-byte model. (Null terminated)
0xB007	32	Set 32-byte model. (Null terminated)
0xB00C	0	Reset Atmel

Reply Payload Field

Event code	Data length in byte(s)	Data Description.
0xB000	3	Byte 0: Major version Byte 1: Minor version

Copyright Convergence Systems Limited 2023 All Rights Reserved

		Byte 2: Build version
0xB001	1	0x00 – success 0x01 – failure 0x02 – full image received successfully
0xB004	32	32-byte serial number (Null terminated)
0xB005	1	0x00 – success 0x01 – failure
0xB006	32	32-byte model (Null terminated)
0xB007	1	0x00 – success 0x01 – failure
0xB00C	1	0x00 – success 0x01 – failure

11.2 Uplink Payload

There is no uplink payload yet.

Copyright Convergence Systems Limited 2023 All Rights Reserved

Chapter 12: Payload – Bluetooth IC Commands

This chapter describes the payload for special notifications.

Payload Format:

Event code (2 bytes)	Data

Reply Format:

Event code (2 bytes)	Data

12.1 Downlink Payload

Payload Field

Event code	Data length in byte(s)	Event/Data Description.
0xC000	0	Get Bluetooth IC firmware version.
0xC001	238	Image raw data. Auto update when subpart = total number of subparts. Byte 0-1: total number of subparts Byte 2-3: subpart index starting from 1 Byte 4-237: Image raw data
0xC003	21	Set device name. Fixed length 21 including null-terminate char.
0xC004	0	Get device name
0xC005	0	Force BT disconnection

Reply Payload Field

Event code	Data length in byte(s)	Data Description.
0xC000	3	Byte 0: Major version (for CS463, 3) Byte 1: Minor version (for CS463, 0) Byte 2: Build version (for CS463, 0)

Copyright Convergence Systems Limited 2023 All Rights Reserved

0xC001	1	0x00 – success (for CS463, 0x00) 0x01 – failure 0x02 – full image received successfully
0xC003	1	0 = Success; 1= Fail (for CS463, 0x00)
0xC004	21	Device name
0xC005	1	0 = Success; 1= Fail (for CS463, 0x00)

12.2 Uplink Payload

There is no uplink payload yet.

Copyright Convergence Systems Limited 2023 All Rights Reserved

Chapter 13: Factory Default Settings

Factory default settings of CS710 includes:

1) Device Name: CS710S Reader

Copyright Convergence Systems Limited 2023 All Rights Reserved

Appendix A – CSL Ex10 Based RFID Command Specification

A.1 Introduction

This Appendix describes the basic CSL Ex10 RFID commands, registers, and events, and how they are activated and managed in the communication between host and CSL Ex10 RFID reader.

A.1.1 Communication Types Introduction

There are 2 main types of communications: commands and uplink packets.

Commands are issued from host processor downlink to the CSL Reader, and the CSL Reader will always give a response to such a command.

Events are generated from CSL Reader going uplink to the host processor. Events do NOT require the host processor to acknowledge back.

There are 3 types of commands and their corresponding responses:

1) CSL RFID Register Read/Write Commands(downlink) and Responses(uplink)

2) CSL RFID Operation Commands(downlink) and Responses(uplink)

3) CSL RFID Administration Commands(downlink) and Responses(uplink)

There is only 1 type of uplink packets: CSL RFID Uplink Packets (Uplink)

All the commands have unique command codes defining each command.

All the command replies echo the downlink command code it corresponds to.

Copyright Convergence Systems Limited 2023 All Rights Reserved

All the RFID uplink packets (uplink) have unique packet code defining each packet.

All command codes and uplink packet codes are 2 byte (4 hex numbers) long.

All packets have header and terminator. Both header and terminator are 2 bytes (4 hex numbers) long.

All packets have sequence number, consisting of 1 byte of cyclic number: 0 to 255, then back to 0 and so on and so forth.

All packets have CRC checksum protection. All CRC are 2 bytes (4 hex numbers) long.

1) CSL RFID Register Read/Write Commands consist of just 2 commands, 1 is read, 1 is write. Each of them can be configured to handle one or multiple registers. Command code for read is **0x1471**. Command code for write is **0x9A06**.

2) CSL RFID Operation Commands consist of short and long commands, and with corresponding command responses

    1. Short commands: short and efficient, but need multiple MAC registers write operations to prepare for it (precede it). Once registers written and verified (all register writes will have response), the short commands can execute operations quickly. Start with “SCSL”.

    2. Long commands: complex instructions, one long command containing all necessary information. Dramatically simplify host application development. Start with “LCSL”

    3. Command Replies: responses to each and every command

3) CSL RFID Administration Commands handle administration matters, e.g. image upload, and with corresponding command responses.

4) CSL RFID Uplink Packets consists of uplink data such as inventory tag data.

Copyright Convergence Systems Limited 2023 All Rights Reserved

Copyright Convergence Systems Limited 2023 All Rights Reserved

A.1.2 Program Flow Introduction

An example of the basic program flow is:

1. The host writes the desired values into the relevant CSL RFID registers.
2. The host receives a write CSL RFID registers response.
3. If needed, the host can also send a read CSL RFID register command to look at certain CSL RFID registers to double confirm its value has been correctly configured as desired.
4. The host receives a read CSL RFID registers response.
5. The host initiates the start of an operation by sending an operation command X.
6. The host receives an operation command X response.
7. The host receives RFID uplink packets.
8. The host waits for, at the end of the last RFID uplink packets, an RFID notification saying the operation command X has completed.

Copyright Convergence Systems Limited 2023 All Rights Reserved

A.1.3 Packet Format Introduction

All **commands** (read, write, operation, administration commands) are encoded in the following manner:

2. **Header** (2 bytes): **80B3**
3. **Command Code** (2 bytes)
4. **Downlink Sequence Number** (1 byte) (use a cyclic sequenced number)
5. **Length of Payload** (2 bytes): total length of payload
6. **Payload** (# of bytes according to #4 above)**:**

All **command responses** (read, write, operation, administration commands) are encoded in the following manner:

1. **Header** (2 bytes): **51E2**
2. **Echo Command Code** (2 bytes)
3. **Echo Downlink Sequence Number** (1 byte) (echo back the corresponding Downlink Sequence Number of the original downlink command)
4. **Length of Payload** (2 bytes): total length of payload
5. **Payload** (# of bytes according to #4 above)**:**

All **Uplink Packets** are encoded in the following manner (the sequence number here is independent of whatever previous downlink commands):

1. **Header** (2 bytes): **49DC**
2. **Event Code** (2 bytes)
3. **Uplink Sequence Number** (1 byte – cyclic sequenced number)
4. **Length of Payload** (2 bytes): total length of payload
5. **Payload** (# of bytes according to #4)**:**

All downlink commands have a sequence number. This sequence number is controlled by the host application to be sequential and recycling.
Uplink command replies echo this downlink sequence number.
Uplink command replies also echo the downlink command.
Uplink Packets have an independent cyclic sequence number controlled by the reader side host processor.

Copyright Convergence Systems Limited 2023 All Rights Reserved

A.1.4 Tag Caching and Duplicate Elimination Rolling Window

Tag caching is needed because CS710S can capture more than 1000 tags per second. Even for simple inventory where the uplink data only contains the PC, EPC and CRC data (12 bytes typically), each tag record requires 36 bytes to carry the information (including the header of the overall packet), then total data rate is 36 Kbyte per second.

A special scheme of tag caching is employed in CS710S.

A tag cache table of index versus PC/EPC/CRC versus tag read time (absolute time) is stored in the ATMEL IC. When a tag is read and found to be in the tag table already, then the uplink packet EPC ID section only contains the tag index (2 bytes only) instead of the whole EPC ID.

The host device (e.g. a mobile phone) will also contain a synchronized tag table and will then update the read for that table entry.

In addition to this scheme of tag data minimization, another mechanism of duplicate elimination rolling window is used in CS710S. User can configure a duplicate elimination time window, during which a tag, if repeatedly read, is only uploaded once to the host processor. Once the duplicate elimination time is exceeded, then the tag data is again uploaded to the host processor.

The combination of tag data buffering using the tag cache table and the tag data minimization using the duplicate elimination rolling window scheme will help reduce overall traffic at the Bluetooth connection.

ATMEL has 384 Kbyte internal SRAM. Circuit has additional external SDRAM of 2 Mbyte. Therefore the cache of tag data can be calculated as follows: assume each tag record takes up 36 bytes, then 2 Mbyte can hold 54,000 tag data. Therefore a conservative estimation of tag cache size it can handle is 12,000 tags. Actually the tag cache cannot be too large otherwise the duplicate elimination checking will be very slow. Also, to speed up duplicate elimination check, a binary tree sorting would be better.

Copyright Convergence Systems Limited 2023 All Rights Reserved

A.2 CSL Commands

There are 4 types of CSL Commands:

1) Write Register command,
2) Read Register command,
3) Operation command,
4) Administration command.

Each command has a corresponding command reply.

Copyright Convergence Systems Limited 2023 All Rights Reserved

A.2.1 Write Register Command

Write command writes data into CSL RFID register. Each CSL RFID register consists of an Address and a Length. Each Register Read and Write commands can target 1 or more registers.

The following is the packet format of the **Write CSL RFID Register** command and its respective command replies.

Write Register Command: SCSLRFIDWriteReg

Byte Offset(s)	Name	Description
1:0	Packet Header – 2 byte fixed Header	**0x80B3**
3:2	Command Code	**0x9A06**
4	Sequence Number	1 byte cyclic sequenced number
6:5	Length of Payload	Total length of payload (not including CRC)
Payload		
7	Number_of_registers to write	Number of registers to write: up to 255 registers (more than the total number of registers of E710)
9:8	reg_addr 1	16bit address of the register to be written.
10	reg_addr_1_length	Length of register 1
11+reg_addr_1_length-1 : 11	reg_addr_1_data	this is the data to be written.
…….	……	Repeat 8:11+reg_addr_1_length-1, total Number_of_registers -1 times

For multiple byte data, Big Endian is used.

Copyright Convergence Systems Limited 2023 All Rights Reserved

Write Register Command Reply:

Byte Offset(s)	**Name**	**Description**
1:0	2 byte fixed Header	**0x51E2**
3:2	Echo Command Code	**0x9A06 (echo the downlink write register command)**
4	Echo Sequence Number	Echo of the downlink command 1 byte cyclic sequenced number
6:5	Payload Length	Length of payload (not including CRC)
Payload		
7	Write Status	Write status return

Copyright Convergence Systems Limited 2023 All Rights Reserved

A.2.2 Read Register Command

Each CSL RFID register consists of an Address and a Length. Each Register Read and Write commands can target 1 or more registers.

The following is the packet format of the Read CSL RFID register command and its respective command reply.

Read Register Command: SCSLRFIDReadReg

Byte Offset(s)	Name	Description
1:0	Beginning of Packet – 2 byte fixed Header	**0x80B3**
3:2	Command Code	**0x1471**
4	Sequence Number	1 byte cyclic sequenced number
6:5	Length of Payload	Total length of payload (not including CRC)
Payload		
7	Number_of_registers to read	Number of registers to read: up to 255 registers (more than the total number of registers of E710)
9:8	reg_addr 1	16bit address of the register to read
10	reg_addr_1_length	1 byte register length
12 : 11	reg_addr_2	16bit address of the register to read
13	reg_addr_2_length	1 byte register length
…….	……	

Copyright Convergence Systems Limited 2023 All Rights Reserved

Read Register Command Reply:

Byte Offset(s)	Name	Description
1:0	2 byte fixed Header	**0x51E2**
3:2	Echo Command Code	**0x1471 (echo the downlink read register command)**
4	Echo Sequence Number	Echo of the downlink command 1 byte cyclic sequenced number
6:5	Payload Length	Length of payload (not including CRC)
Payload		
7+reg_addr_1_length-1 : 7	reg_addr_1_data	this is the data read.
…….	……	The other registers to be read

For multiple byte data, Big Endian is used.

Copyright Convergence Systems Limited 2023 All Rights Reserved

A.2.3 Operation Commands

Here is a table of CSL RFID Operation Commands – **Short Commands**

Short commands are very simple, mainly consisting of the command code. It assumes all the relevant MAC registers have been written to before.

Short Operation Command Packet Format

Byte Offset(s)	**Name**	**Description**
1:0	Packet Header – 2 byte fixed Header	**0x80B3**
3:2	Command Code	**See operation short command code table below.**
4	Sequence Number	1 byte cyclic sequenced number
6:5	Length of Payload	Total length of payload (not including CRC), for short command, it is always 0

Short Operation Command Reply Packet Format

Byte Offset(s)	**Name**	**Description**
1:0	2 byte fixed Header	**0x51E2**
3:2	Echo Command Code	**echo the downlink Short Operation command**
4	Echo Sequence Number	Echo of the downlink command 1 byte cyclic sequenced number
6:5	Length of Payload	Total length of payload (not including CRC), for short command, it is always 0

Copyright Convergence Systems Limited 2023 All Rights Reserved

Short Operation Commands Table

Command	Command Code	Length	Description
SCSLRFIDStartSimpleInventory	0x10A1	Fixed	Start Simple Inventory of EPC – assuming registers are properly configured. No Select. No Multiple Banks.
SCSLRFIDStartCompactInventory	0x10A2	Fixed	Start compact inventory with only PC, EPC, RSSI coming up, each uplink packet contains many tags
SCSLRFIDStartSelectInventory	0x10A3	Fixed	Start inventory of EPC with 1 to 7 select according to the Select configuration registers.
SCSLRFIDStartMBInventory	0x10A4	Fixed	Start multibank inventory of EPC according to the multibank configuration.
SCSLRFIDStartSelectMBInventory	0x10A5	Fixed	Start multibank inventory of EPC with 1 to 7 Select according to the Select configuration registers.
SCSLRFIDStartSelectCompactInventory	0x10A6	Fixed	Start compact inventory with 1 to 7 select according to the Select configuration registers.
SCSLRFIDStopOperation	0x10AE	Fixed	Stop Inventory
SCSLRFIDReadMB	0x10B1	Fixed	Read 1 tag based on 1 or more Select and multiple banks (at least 1 Select containing EPC)
SCSLRFIDWriteMB	0x10B2	Fixed	Write 1 tag based on 1 or more Select and multiple banks
SCSLRFIDWriteMBAny	0x10B3	Fixed	Write any tag based on 1 or more Select and multiple banks
SCSLRFIDBlockWriteMB	0x10B5	Fixed	Block write 1 tag based on 1 or more Select and multiple banks
SCSLRFIDBlockWriteMBAny	0x10B6	Fixed	Block write any tag based on 1 or more Select and multiple banks

Copyright Convergence Systems Limited 2023 All Rights Reserved

SCSLRFIDLock	0x10B7	Fixed	Lock 1 tag based on 1 or more Select
SCSLRFIDKill	0x10B8	Fixed	Kill 1 tag based on 1 or more Select and Kill password
SCSLRFIDAuthenticate	0x10B9	Fixed	Authenticate 1 tag based on 1 or more Select
SCSLClearTagCacheTable	0x10D3	Fixed	Clear tag cache table
SCSLUploadTagCacheTableToHost	0x10D4	Fixed	Upload tag cache table to host using RFID Uplink Packets
SCSLRFIDRegisterReset	0x10D5	Fixed	Reset all MAC registers to factory default
SCSLEx10Reset	0x10D6	Fixed	Soft Reset E710 IC
SCSLRFIDCircuitReset	0x10D7	Fixed	Reset Complete RFID Circuit (power off and power on whole RFID circuit)

Copyright Convergence Systems Limited 2023 All Rights Reserved

Long Commands (TBD)

Command	Command Code	Length	Description

Copyright Convergence Systems Limited 2023 All Rights Reserved

A.2.4 Administration Commands

Here is a table of CSL RFID Administration Commands

Administration commands handle administration matters, e.g., image upload (upgrade or downgrade)

Administration Command Packet Format

Byte Offset(s)	Name	Description
1:0	Packet Header – 2 byte fixed Header	**0x80B3**
3:2	Command Code	**See Administration command codes table below.**
4	Sequence Number	1 byte cyclic sequenced number
6:5	Length of Payload	Total length of payload (not including CRC), for short command, it is always 0
N:7	Payload	Payload

Copyright Convergence Systems Limited 2023 All Rights Reserved

Administration Command Reply Packet Format

Byte Offset(s)	Name	Description
1:0	2 byte fixed Header	**0x51E2**
3:2	Echo Command Code	**echo the downlink Administration command**
4	Echo Sequence Number	Echo of the downlink command 1 byte cyclic sequenced number
6:5	Length of Payload	Total length of payload (not including CRC)
N:7	Payload	Payload

Copyright Convergence Systems Limited 2023 All Rights Reserved

Command	Command Code	Payload	Length	Description
CSLSyncTime	0x8840	Current UTC timestamp	4	Send the UTC timestamp information to the reader for time synchronization

Copyright Convergence Systems Limited 2023 All Rights Reserved

A.3 CSL Registers

The following are definitions of the relevant registers that can be read or written to according to A.1. First a registers summary table is provided. Then the details of each register are described.

A.3.1 CSL RFID Registers Summary Table

Register	Address	Length	Read Write	Description
VersionString	0x0008	32	read-only	The version string of the application firmware
BuildNumber	0x0028	4	read-only	The build number of the application firmware
LoopStyle	0x3000	4	read-write	Time of loop execution of inventory: 0: infinite (default – infinite rounds) N: msec
HopStyle	0x3008	1	read-write	Method of Hop: 0: hop frequency from inventory round to inventory round 1: only hop frequency when RegulatoryDwellTime is reached (default)
RegulatoryNoEmissionTime	0x3010	2	Read-Write	Byte 0: Disable = 0 (default), Enable = 1 Byte 1: Custom Regulatory No Emission Time in msec. This is usually called the regulatory off time.
CountryEnum	0x3014	2	Read-write	Current Country Enum based on **Country Enum Table in section** A.3.4
FrequencyChannelIndex	0x3018	1	Read-write	For hopping channel, set to 0. For fixed channel, need to select the channel to use. CountryEnum table (hardcoded contains all frequency channel information
AntennaPortConfig	0x3030 – 0x3130	256	Read-write	Antenna Port #1 to #16: For each port:

Copyright Convergence Systems Limited 2023 All Rights Reserved

Up to Array of 16 ports 16 bytes per port				Byte 0: Disable = 0, Enable = 1 Byte 2:1 = Dwell Time in msec = 0 (default infinite) Byte 4:3 = Power (0.01 dBm step, 0 to 3000) Byte 8:5 = InventoryRoundControl (see A.3.2) Byte 12:9 = InventoryRoundControl_2 (see A.3.2) Byte 13 = Target Toggle (0 = No, 1 = Yes) Byte 14:15 = RfMode (see A.3.5) Default: Antenna Port 1 is enabled, Other antenna ports are disabled; Dwell time is 0 msec, power is 3000, toggle is on, RfMode is 302.
SelectConfiguration Array of 7 Sequentially Actioned Selects: 1 > 2 > 3 > 4 > 5 > 6 > 7	0x3140–0x3266	294	Read-write (42 bytes per Select configuration)	7 sequential sets: Enable/Disable(1 Byte) Membank(1 Byte) Pointer (4 bytes offset pointer) Length (1 byte) - # of bits of mask, maximum length is 255 bits Mask (32 bytes) – 255 bits max mask Target (1 byte) Action (1 byte) Post Configuration Delay in ms (1 byte) – default is 0
MultibankReadConfig	0x3270 – 3285	21	Read-write	3 sets (7 bytes per set) Enable/Disable(1 byte) Bank #(1 byte) Address (4 bytes offset pointer) Length (1 byte) - # of words to read

Copyright Convergence Systems Limited 2023 All Rights Reserved

MultibankWriteConfig	0x3290 – 38A5	1557	Read-Write	3 sets (519 bytes per set) Enable/Disable(1 byte) Bank #(1 byte) Address (4 bytes offset pointer) Length (1 byte) - # of words to write. Max 32. Data (512 bytes) – data to write
AccessPassword	0x38A6	4	Read-write	Access password
KillPassword	0x38AA	4	Read-write	Kill password
LockMask	0x38AE	2	Read-write	Lock action (see A.3.3)
LockAction	0x38B0	2	Read-write	Lock mask (see A.3.3)
DuplicateEliminationRollingWindow	0x3900	1	Read-Write	Duplicate elimination rolling window in seconds. Default = 0 (means no duplicate elimination).
TagCacheTableCurrentSize	0x3902	2	Read-Write	Number of tags in the tag cache table
TagCacheStatus	0x3904	1	Read-Write	Enable or disable
EventPacketUplinkEnable	0x3906	2	Read-Write	16 possible events to be enabled or disabled Bit 0: keep alive (default on) Bit 1: inventory round end (default off) Bit 2: CRC error (default off) Bit 3: Tag rate data (default off)
IntraPacketDelay	0x3908	1	Read-Write	Default 4 msec,, to control and minimize Bluetooth path packet loss
RssiFilteringConfig	0x390A	1	Read-Write	0 = disable (default) 1 = RSSI less than or equal to threshold 2 = RSSI greater than or equal to threshold
RssiThreshold	0x390C	2	Read-Write	unit in dBm x 100
AuthenticateConfig	0x390E	3	Read-Write	Bit 0: SenRep Bit 1: IncRepLen Bit 9:2: CSI Bit 21:10: Length of message in bits
AuthenticateMessage	0x3912	32	Read-Write	Authenticate message
AuthenticateResponseLen	0x3944	2	Read-Write	Expected response length in bits

Copyright Convergence Systems Limited 2023 All Rights Reserved

ReversePowerADC	0x3946	2	Read-Write	Last reverse power adc value
CurrentPort	0x3948	1	Read-Write	Current antenna port (0 – 15)
Model Name	0x5000	32	Read Only	Model code string
Serial Number	0x5020	32	Read Only	Serial no. string
Country Enum	0x5040	2	Read-Write	Default Country Enum based on **Country Enum Table in section A.3.4**

Copyright Convergence Systems Limited 2023 All Rights Reserved

A.3.2 CSL RFID Registers Details

The following are details of the CSL Customer MAC Registers:

InventoryRoundControl: Configuration for Inventory Round

<table>
<tr><th>Bit</th><th>Name</th><th>Description</th></tr>
<tr><td>3:0</td><td>InitialQ</td><td>The initial Q value to use to start the round.</td></tr>
<tr><td>7:4</td><td>MaxQ</td><td>The maximum allowed Q value</td></tr>
<tr><td>11:8</td><td>MinQ</td><td>The minimum allowed Q value</td></tr>
<tr><td>15:12</td><td>NumMinQCycles</td><td>The number of Empty Minimum Q querys or no valid EPCs required to end the Inventory Round</td></tr>
<tr><td>16</td><td>FixedQMode</td><td>Operate the Inventory round as a single pass through the slots No Q adjustment</td></tr>
<tr><td>17</td><td>QIncreaseUseQuery</td><td>If this is true the Q algorithm will send a Full Query instead of a QueryAdj command when increasing the Q.</td></tr>
<tr><td>18</td><td>QDecreaseUseQuery</td><td>If this is true the Q algorithm will send a Full Query instead of a QueryAdj command when decreasing Q</td></tr>
<tr><td>20:19</td><td>Session</td><td>The Gen2 session to use in the Query and other Query like Gen2 packets. This encoding matches the encoding specified in the Gen2 specification. This defaults to 1 if TagFocus is enabled.
<table>
<tr><th>Value</th><th>Name</th><th>Persistence Time</th></tr>
<tr><td>0x0</td><td>S0</td><td>0</td></tr>
<tr><td>0x1</td><td>S1</td><td>0.5 – 5 seconds</td></tr>
<tr><td>0x2</td><td>S2</td><td>2 – 5 seconds</td></tr>
<tr><td>0x3</td><td>S3</td><td>2 – 5 seconds</td></tr>
</table></td></tr>
<tr><td>22:21</td><td>Select</td><td>The Sel field in the Query command. This encoding matches the encoding specified in the Gen2 specification. This defaults to 1 if TagFocus or FastId are enabled.</td></tr>
<tr><td>23</td><td>Target</td><td>Indicates A or B target for session flag values. This encoding matches the encoding specified in the Gen2 specification.</td></tr>
<tr><td>24</td><td>HaltOnAllTags</td><td>When set, this will cause the modem to issue a ReqRn to open every tag that it reads and then allow the host to perform Gen2 access commands on that tag.</td></tr>
<tr><td>25</td><td>FastIdEnable</td><td>When set, this will cause the modem to automatically perform the extra operations required for FastID operation. This forces the select flag to 1.</td></tr>
</table>

Copyright Convergence Systems Limited 2023 All Rights Reserved

26	TagFocusEnable	Controls whether or not to enable tag focus at the beginning of every inventory round. This forces the select and session flags to 1.
27	AutoAccess	Enables the Auto Access feature which will automatically do all the access commands enabled in the Gen2AutoAccessEnable register and then continue to the next slot without halting (unless halt_on_all_tags is enabled). Note that the Auto Access feature does not depend on the control bits in the HaltedControl register. (Must be 0).
28	AbortOnFail	If this field is true it will cause the LMAC to immediately abort an auto access sequence if one of the transactions fails CRC. It will report all the successful transactions to the host along with the failing transaction. (Must be 0).
29	HaltOnFail	This field will cause the LMAC to enter the halted state and wait for host commands if one of the auto access sequence commands fails CRC. (Must be 0).
30	AlwaysAck	(Must be 0).

InventoryRoundControl_2: Configuration for Inventory Round

Bit	Name	Description
7:0	MaxQueriesSinceValidEpc	This is a control for the dynamic Q algorithm which specifies the number of Query or QueryAdj commands allowed without receiving a valid EPC. If this value is reached the modem will immediately end the inventory round. (default 8).
23:16	StartingMinQCount	This is a control for the dynamic Q algorithm which specifies what value to start the min_q count with. It allows the host to preload the counter value from the previous round if it was not finished. A value of 0 will give normal operation. (default 0).
31:24	StartingMaxQueriesSinceValidEpcCount	This is a control for the dynamic Q algorithm which specifies what value to start the num_queries_since_valid_epc count with. This allows the host to preload the counter value from the previous round if it was not finished. A value of 0 will give the normal operation. (default 0).

Copyright Convergence Systems Limited 2023 All Rights Reserved

A.3.3 EPC Commands Refresh

This section is purely for referencing the EPC commands. The following are some refresh of EPC commands that correspond to the variables used in the Registers above.

Query Command:

Table 6.32: *Query* command

	Command	DR	M	TRext	Sel	Session	Target	Q	CRC
# of bits	4	1	2	1	2	2	1	4	5
description	1000	0: DR=8 1: DR=64/3	00: M=1 01: M=2 10: M=4 11: M=8	0: No pilot tone 1: Use pilot tone	00: All 01: All 10: ~**SL** 11: **SL**	00: S0 01: S1 10: S2 11: S3	0: *A* 1: *B*	0–15	CRC-5

Select Command:

Table 6.29: *Select* command

	Command	Target	Action	MemBank	Pointer	Length	Mask	Truncate	CRC
# of bits	4	3	3	2	EBV	8	Variable	1	16
description	1010	000: **Inventoried** (S0) 001: **Inventoried** (S1) 010: **Inventoried** (S2) 011: **Inventoried** (S3) 100: **SL** 101: RFU 110: RFU 111: RFU	See Table 6.30	00: FileType 01: EPC 10: TID 11: File_0	Starting Mask address	Mask length (bits)	Mask value	0: Disable truncation 1: Enable truncation	CRC-16

Table 6.30: Tag response to Action parameter

Action	Tag Matching	Tag Not-Matching
000	assert **SL** or **inventoried** → *A*	deassert **SL** or **inventoried** → *B*
001	assert **SL** or **inventoried** → *A*	do nothing
010	do nothing	deassert **SL** or **inventoried** → *B*
011	negate **SL** or (*A* → *B*, *B* → *A*)	do nothing
100	deassert **SL** or **inventoried** → *B*	assert **SL** or **inventoried** → *A*
101	deassert **SL** or **inventoried** → *B*	do nothing
110	do nothing	assert **SL** or **inventoried** → *A*
111	do nothing	negate **SL** or (*A* → *B*, *B* → *A*)

Copyright Convergence Systems Limited 2023 All Rights Reserved

Lock Command:

Table 6.49: *Lock* command

	Command	Payload	RN	CRC
# of bits	8	20	16	16
description	11000101	Mask and Action Fields	handle	CRC-16

Lock-Command Payload

19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Kill Mask		Access Mask		EPC Mask		TID Mask		File_0 Mask		Kill Action		Access Action		EPC Action		TID Action		File_0 Action	

Masks and Associated Action Fields

	Kill pwd		Access pwd		EPC memory		TID memory		File_0 memory	
	19	18	17	16	15	14	13	12	11	10
Mask	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write
	9	8	7	6	5	4	3	2	1	0
Action	pwd read/ write	perma lock	pwd read/ write	perma lock	pwd write	perma lock	pwd write	perma lock	pwd write	perma lock

Figure 6.25: *Lock* payload and usage

Table 6.50: *Lock* Action-field functionality

pwd-write	permalock	Description
0	0	Associated memory bank/file is writeable from either the **open** or **secured** states.
0	1	Associated memory bank/file is permanently writeable from either the **open** or **secured** states and may never be locked.
1	0	Associated memory bank/file is writeable from the **secured** state but not from the **open** state.
1	1	Associated memory bank/file is not writeable from any state.
pwd-read/write	**permalock**	**Description**
0	0	Associated password location is readable and writeable from either the **open** or **secured** states.
0	1	Associated password location is permanently readable and writeable from either the **open** or **secured** states and may never be locked.
1	0	Associated password location is readable and writeable from the **secured** state but not from the **open** state.
1	1	Associated password location is not readable or writeable from any state.

Copyright Convergence Systems Limited 2023 All Rights Reserved

Authenticate Command:

Table 6.58: *Authenticate* command

	Command	RFU	SenRep	IncRepLen	CSI	Length	Message	RN	CRC
# of bits	8	2	1	1	8	12	Variable	16	16
description	11010101	00	0: store 1: send	0: Omit length from reply 1: Include length in reply	CSI	length of message	message (depends on CSI)	handle	CRC-16

Copyright Convergence Systems Limited 2023 All Rights Reserved

A.3.4 CSL Ex10 Country Enum Table

Country	CSL E710 Country Enum	CSL E710 Country Name	CSL Reader Model Code (Region Code)	Frequency Channel #	Fixed or Hop	Hop Time or On Time	Off Time	Channel separation	First Channel	Last Channel	Note
Albania	1	Albania1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	2	Albania2	-2 RW	23	Hop	400 msec		250 KHz	915.25 MHz	920.75 MHz	915-921 MHz
Algeria	3	Algeria1	-1	4	Fixed	3900 msec	100 msec	600 KHz	871.6 MHz	873.4 MHz	870-876 MHz
	4	Algeria2	-1	4	Fixed	3900 msec	100 msec	600 KHz	881.6 MHz	883.4 MHz	880-885 MHz
	5	Algeria3	**-9**	**4**	**Fixed**	**3900 msec**		**1200 KHz**	**916.3 MHz**	**919.9 MHz**	**915-921 MHz**
	6	Algeria4	-7	2	Fixed	3900 msec	100 msec	500 KHz	925.25 MHz	925.75 MHz	925-926 MHz
Argentina	7	Argentina	-2 RW	50	Hop	400 msec		500 KHz	902.75 MHz	927.25 MHz	
Armenia	8	Armenia	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Australia	9	Australia1	-2 AS	10	Hop	400 msec		500 KHz	920.75 MHz	925.25 MHz	
	10	Australia2	-2 AS	14	Hop	400 msec		500 KHz	918.75 MHz	925.25 MHz	
Austria	11	Austria1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	12	Austria2	**-9**	**4**	**Fixed**	**3900 msec**		**1200 KHz**	**916.3 MHz**	**919.9 MHz**	**915-921 MHz**
Azerbaijan	13	Azerbaijan	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Bahrain	14	Bahrain	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Bangladesh	15	Bangladesh	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Belarus	16	Belarus	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	

Copyright Convergence Systems Limited 2023 All Rights Reserved

Belgium	17	Belgium1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	18	Belgium2	**-9**	**4**	**Fixed**	**3900 msec**		**1200 KHz**	**916.3 MHz**	**919.9 MHz**	**915-921 MHz**
Bolivia	19	Bolivia	-2	50	Hop	400 msec		500 KHz	902.75 MHz	927.25 MHz	
Bosnia and Herzegovina	20	Bosnia	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Botswana	21	Botswana	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Brazil	22	Brazil1	-2 RW	9	Fixed	3900 msec	100 msec	500 KHz	902.75 MHz	906.75 MHz	
	23	Brazil2	-2 RW	24	Fixed	3900 msec	100 msec	500 KHz	915.75 MHz	927.25 MHz	
Brunei Darussalam	24	Brunei1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	25	Brunei2	-7	7	Fixed	3900 msec	100 msec	250 KHz	923.25 MHz	924.75 MHz	923 - 925 MHz
Bulgaria	26	Bulgaria1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	27	Bulgaria2	**-9**	**4**	**Fixed**	**3900 msec**		**1200 KHz**	**916.3 MHz**	**919.9 MHz**	**915-921 MHz**
Cambodia	28	Cambodia	-7	16	Hop	400 msec		250 KHz	920.625 MHz	924.375 MHz	
Cameroon	29	Cameroon	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Canada	30	Canada	-2	50	Hop	400 msec		500 KHz	902.75 MHz	927.25 MHz	
Chile	31	Chile1	-2 RW	3	Fixed	3900 msec	100 msec	1200 KHz	916.3 MHz	918.7 MHz	
	32	Chile2	-2 RW	24	Hop	400 msec		500 KHz	915.75 MHz	927.25 MHz	
	33	Chile3	-2 RW	4	Hop	400 msec		500 KHz	925.75 MHz	927.25 MHz	
China	34	China	-7	16	Hop	400 msec		250 KHz	920.625 MHz	924.375 MHz	
Colombia	35	Colombia	-2 RW	50	Hop	400 msec		500 KHz	902.75 MHz	927.25 MHz	
Congo, Rep.	36	Congo	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Costa Rica	37	CostaRica	-2 RW	50	Hop	400 msec		500 KHz	902.75 MHz	927.25 MHz	

Copyright Convergence Systems Limited 2023 All Rights Reserved

Côte d'Ivoire	38	CotedIvoire	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Croatia	39	Croatia	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Cuba	40	Cuba	-2 RW	50	Hop	400 msec		500 KHz	902.75 MHz	927.25 MHz	
Cyprus	41	Cyprus1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	42	Cyprus2	**-9**	**4**	**Fixed**	**3900 msec**		**1200 KHz**	**916.3 MHz**	**919.9 MHz**	**915-921 MHz**
Czech Republic	43	Czech1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	44	Czech2	**-9**	**4**	**Fixed**	**3900 msec**		**1200 KHz**	**916.3 MHz**	**919.9 MHz**	**915-921 MHz**
Denmark	45	Denmark1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	46	Denmark2	**-9**	**4**	**Fixed**	**3900 msec**		**1200 KHz**	**916.3 MHz**	**919.9 MHz**	**915-921 MHz**
Dominican Republic	47	Dominican	-2 RW	50	Hop	400 msec		500 KHz	902.75 MHz	927.25 MHz	
Ecuador	48	Ecuador	-2 RW	50	Hop	400 msec		500 KHz	902.75 MHz	927.25 MHz	
Egypt, Arab Rep.	49	Egypt	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
El Salvador	50	ElSalvador	-2 RW	50	Hop	400 msec		500 KHz	902.75 MHz	927.25 MHz	
Estonia	51	Estonia	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Finland	52	Finland1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	53	Finland2	**-9**	**4**	**Fixed**	**3900 msec**		**1200 KHz**	**916.3 MHz**	**919.9 MHz**	**915-921 MHz**
France	54	France	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Georgia	55	Georgia	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Germany	56	Germany	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Ghana	57	Ghana	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Greece	58	Greece	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Guatemala	59	Guatemala	-2 RW	50	Hop	400 msec		500 KHz	902.75 MHz	927.25 MHz	

Copyright Convergence Systems Limited 2023 All Rights Reserved

Hong Kong, China	60	HongKong1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	61	HongKong2	-2 OFCA	50	Hop	400 msec		50 KHz	921.25 MHz	923.7 MHz	
Hungary	62	Hungary1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	63	Hungary2	**-9**	**4**	**Fixed**	**3900 msec**		**1200 KHz**	**916.3 MHz**	**919.9 MHz**	**915-921 MHz**
Iceland	64	Iceland	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
India	65	India	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Indonesia	66	Indonesia	-7	4	Hop	400 msec		500 KHz	923.75 MHz	924.25 MHz	
Iran, Islamic Rep.	67	Iran	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Ireland	68	Ireland1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	69	Ireland2	**-9**	**4**	**Fixed**	**3900 msec**		**1200 KHz**	**916.3 MHz**	**919.9 MHz**	**915-921 MHz**
Israel	70	Israel	-9	3	Fixed	3900 msec		500 KHz	915.5 MHz	916.5 MHz	
Italy	71	Italy	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Jamaica	72	Jamaica	-2 RW	50	Hop	400 msec		500 KHz	902.75 MHz	927.25 MHz	
Japan	73	Japan4	-8 JP4	4	Fixed	3900 msec		1200 KHz	916.8 MHz	920.4 MHz	
	74	Japan6	-8 JP6	6	Fixed	3900 msec	100 msec	1200 KHz, 200 KHz Last 2	916.8 MHz	920.8 MHz	LBT carrier sense with Transmission Time Control
Jordan	75	Jordan	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Kazakhstan	76	Kazakhstan	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Kenya	77	Kenya	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Korea, Rep.	78	Korea	-6	6	Hop	400 msec		600 KHz	917.3 MHz	920.3 MHz	
Korea (DPR)	79	KoreaDPR	-7	16	Hop	400 msec		250 KHz	920.625 MHz	924.375 MHz	
Kuwait	80	Kuwait	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	

Copyright Convergence Systems Limited 2023 All Rights Reserved

Kyrgyz Republic	81	Kyrgyz	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Latvia	82	Latvia	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Lebanon	83	Lebanon	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Libya	84	Libya	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Liechtenstein	85	Liechtenstein1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	86	Liechtenstein2	**-9**	**4**	**Fixed**	**3900 msec**		**1200 KHz**	**916.3 MHz**	**919.9 MHz**	**915-921 MHz**
Lithuania	87	Lithuania1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	88	Lithuania2	**-9**	**4**	**Fixed**	**3900 msec**		**1200 KHz**	**916.3 MHz**	**919.9 MHz**	**915-921 MHz**
Luxembourg	89	Luxembourg1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	90	Luxembourg2	**-9**	**4**	**Fixed**	**3900 msec**		**1200 KHz**	**916.3 MHz**	**919.9 MHz**	**915-921 MHz**
Macao, China	91	Macao	-7	16	Hop	400 msec		250 KHz	920.625 MHz	924.375 MHz	
Macedonia, FYR	92	Macedonia	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Malaysia	93	Malaysia	-7	6	Hop	400 msec		500 KHz	919.75 MHz	922.25 MHz	
Malta	94	Malta1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	95	Malta2	**-9**	**4**	**Fixed**	**3900 msec**		**1200 KHz**	**916.3 MHz**	**919.9 MHz**	**915-921 MHz**
Mauritius	96	Mauritius	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Mexico	97	Mexico	-2	50	Hop	400 msec		500 KHz	902.75 MHz	927.25 MHz	
Moldova	98	Moldova1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	99	Moldova2	**-9**	**4**	**Fixed**	**3900 msec**		**1200 KHz**	**916.3 MHz**	**919.9 MHz**	**915-921 MHz**
Mongolia	100	Mongolia	-7	16	Hop	400 msec		250 KHz	920.625 MHz	924.375 MHz	
Montenegro	101	Montenegro	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Morocco	102	Morocco	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Netherlands	103	Netherlands	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
New Zealand	104	NewZealand1	-1	4	Hop	400 msec		500 KHz	864.75 MHz	867.25 MHz	

Copyright Convergence Systems Limited 2023 All Rights Reserved

	105	NewZealand2	-2 NZ	14	Hop	400 msec		500 KHz	920.75 MHz	927.25 MHz	
Nicaragua	106	Nicaragua	-2 RW	50	Hop	400 msec		500 KHz	902.75 MHz	927.25 MHz	
Nigeria	107	Nigeria	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Norway	108	Norway1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	109	Norway2	**-9**	**4**	**Fixed**	**3900 msec**		**1200 KHz**	**916.3 MHz**	**919.9 MHz**	**915-921 MHz**
Oman	110	Oman	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Pakistan	111	Pakistan	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Panama	112	Panama	-2 RW	50	Hop	400 msec		500 KHz	902.75 MHz	927.25 MHz	
Paraguay	113	Paraguay	-2 RW	50	Hop	400 msec		500 KHz	902.75 MHz	927.25 MHz	
Peru	114	Peru	-2 RW	24	Hop	400 msec		500 KHz	915.75 MHz	927.25 MHz	
Philippines	115	Philippines	-2 RW	8	Hop	400 msec		250 KHz	918.125 MHz	919.875 MHz	
Poland	116	Poland	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Portugal	117	Portugal	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Romania	118	Romania	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Russian Federation	119	Russia1	-1	4	Fixed	3900 msec	100 msec	600 KHz	866.3 MHz	867.5 MHz	2 W ERP
	120	Russia3	**-9**	**4**	**Fixed**	**3900 msec**		**1200 KHz**	**916.3 MHz**	**919.9 MHz**	**1 W ERP**
Senegal	121	Senegal	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Serbia	122	Serbia	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Singapore	123	Singapore1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	124	Singapore2	-2 SG	8	Hop	400 msec		500 KHz	920.75 MHz	924.25 MHz	
Slovak Republic	125	Slovak1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	126	Slovak2	**-9**	**4**	**Fixed**	**3900 msec**		**1200 KHz**	**916.3 MHz**	**919.9 MHz**	**915-921 MHz**
Slovenia	127	Slovenia1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	128	Solvenia2	**-9**	**4**	**Fixed**	**3900 msec**		**1200 KHz**	**916.3 MHz**	**919.9 MHz**	**915-921 MHz**

Copyright Convergence Systems Limited 2023 All Rights Reserved

South Africa	129	SAfrica1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	130	SAfrica2	**-9**	**4**	**Fixed**	**3900 msec**		**1200 KHz**	**916.3 MHz**	**919.9 MHz**	**915-921 MHz**
Spain	131	Spain	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Sri Lanka	132	SriLanka	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Sudan	133	Sudan	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Sweden	134	Sweden1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	135	Sweden2	**-9**	**4**	**Fixed**	**3900 msec**		**1200 KHz**	**916.3 MHz**	**919.9 MHz**	**915-921 MHz**
Switzerland	136	Switzerland1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	137	Switzerland2	**-9**	**4**	**Fixed**	**3900 msec**		**1200 KHz**	**916.3 MHz**	**919.9 MHz**	**915-921 MHz**
Syrian Arab Rep.	138	Syria	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Taiwan	139	Taiwan1	-4	12	Hop	400 msec		375 KHz	922.875 MHz	927.000 MHz	1 Watt ERP for Indoor
	140	Taiwan2	-4	12	Hop	400 msec		375 KHz	922.875 MHz	927.000 MHz	0.5 Watt ERP for Outdoor
Tajikistan	141	Tajikistan	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Tanzania	142	Tanzania	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Thailand	143	Thailand	-2 RW	8	Hop	400 msec		500 KHz	920.75 MHz	924.25 MHz	
Trinidad and Tobago	144	Trinidad	-2 RW	50	Hop	400 msec		500 KHz	902.75 MHz	927.25 MHz	
Tunisia	145	Tunisia	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Turkey	146	Turkey	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Turkmenistan	147	Turkmenistan	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Uganda	148	Uganda	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Ukraine	149	Ukraine	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
United Arab Emirates	150	UAE	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	

Copyright Convergence Systems Limited 2023 All Rights Reserved

United Kingdom	151	UK1	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
	152	UK2	**-9**	**4**	**Fixed**	**3900 msec**		**1200 KHz**	**916.3 MHz**	**919.9 MHz**	**915-921 MHz**
United States	153	USA	-2	50	Hop	400 msec		500 KHz	902.75 MHz	927.25 MHz	
Uruguay	154	Uruguay	-2 RW	50	Hop	400 msec		500 KHz	902.75 MHz	927.25 MHz	
Venezuela, RB	155	Venezuela	-2 RW	50	Hop	400 msec		500 KHz	902.75 MHz	927.25 MHz	
Vietnam	156	Vietnam1	-1	3	Fixed	3900 msec	100 msec	600 KHz	866.3 MHz	867.5 MHz	866-868 MHz
	157	Vietnam2	-7	8	Hop	400 msec		500 KHz	918.75 MHz	922.25 MHz	
Yemen, Rep.	158	Yemen	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Zimbabwe	159	Zimbabwe	-1	4	Fixed	3900 msec	100 msec	600 KHz	865.7 MHz	867.5 MHz	
Vietnam	160	Vietnam3	-7	4	Hop	400 msec		500 KHz	920.75 MHz	922.25 MHz	

Remark: Enum 160 (Vietnam3) is added to Vietnam country.

Copyright Convergence Systems Limited 2023 All Rights Reserved

A.3.5 Reader Mode Table

Mode ID	**Mode Optimization**	**Intended Region**	**Forward Link Modulation**	**Tari (µs)**	**BLF (kHz)**	**Reverse Link Modulation**	**Chip RX Sensitivity (dBm)**	**Maximum Read Rate (tags/s)**
103	Read Rate	FCC	DSB	6.25	640	Miller M=1	**-78**	**1000**
120	Read Rate	FCC	DSB	6.25	640	Miller M=2	**-81**	**700**
345	Read Rate	FCC, ETSI UB	PR_ASK	7.5	640	Miller M=4	**-84**	**400**
302	Read Rate	All	PR_ASK	7.5	640	Miller M=1	**-78**	**800**
323	Read Rate	All	PR_ASK	7.5	640	Miller M=2	**-81**	**550**
344	ETSI UB DRM	All	PR_ASK	7.5	640	Miller M=4	**-84**	**400**
223	ETSI LB	ETSI	PR_ASK	15	320	Miller M=2	**-84**	**300**
222	ETSI LB	ETSI	PR_ASK	20	320	Miller M=2	**-84**	**250**
241	ETSI LB DRM	ETSI	PR_ASK	20	320	Miller M=4	**-87**	**200**
244	FCC DRM	FCC	PR_ASK	20	250	Miller M=4	**-88**	**150**
285	Sensitivity	All	PR_ASK	20	160	Miller M=8	**-93**	**50**

Copyright Convergence Systems Limited 2023 All Rights Reserved

A.4 CSL RFID Uplink Packets

RFID Uplink Packets come back from RFID reader module to the host processor to report important information, for example, tag inventory data (tag_read).

Both CSL packets and Impinj packets are sent back to host processor.

Byte Offset(s)	**Name**	**Description**
1:0	Beginning of Packet – 2 byte fixed Header	**0x49DC**
3:2	Packet Code	Packet codes
4	Sequence Number	1 byte cyclic sequenced number
6:5	Length of Packet Payload	Total length of payload (not including CRC)
N:7 (N=7+Payload Length-1)	Packet Payload	See below packet table and packet details

Copyright Convergence Systems Limited 2023 All Rights Reserved

A.4.1 CSL RFID Uplink Packets Summary Table

The following is table of CSL RFID Uplink Packets:

RFID Packet Code	Name	Descriptions
0x3001	csl_tag_read_epc_only_new	Simple inventory return containing only PC/EPC/CRC of EPC Bank, antenna port, phase
0x3002	csl_tag_read_epc_only_recurrent	Index only replacing the full PC/EPC/CRC data
0x3003	csl_tag_read_multibank_new	Multibank inventory return containing Bank 1 PC and EPC plus extra N sets of data from other banks.
0x3004	csl_tag_read_multibank_recurrent_index_only	Index only replacing the full PC/EPC/CRC data
0x3006	csl_tag_read_compact	Compact tag return, multiple tags per event, only PC, EPC and RSSI returned.
0x3007	csl_miscellaneous_event	Contains various events, for example, antenna cycle end, keep alive, CRC error, etc.

Copyright Convergence Systems Limited 2023 All Rights Reserved

0x3008	csl_operation_complete	Generated at the completion of any operation
0x3009	csl_access_complete	Generated at the completion of any access (read, write, lock, kill, etc…) command

Copyright Convergence Systems Limited 2023 All Rights Reserved

A.4.2 CSL RFID Uplink Packets Details

csl_tag_read_epc_only_new

Packet Code: 0x3001

Packet Payload:

Offset	**Name**	**Size**	**Description**
0	UTC Time Stamp	4 bytes	32 bit
4	RSSI	2 bytes	RSSI of the tag response
6	rf_phase_begin	2 bytes	The rf phase of the tag response at the beginning of the packet
8	rf_phase_end	2 bytes	The rf phase of the tag response at the end of the packet
10	Port Number	1 byte	Antenna Port Number
11	RESERVE	2 bytes	Reserve
13	Tag_Index	2 bytes	New assigned Tag index
15	Tag Data	Variable, Depends on PC bits	The PC, EPC of the tag. The host must examine the PC bits to determine the number of bytes to include in the EPC. Typical 96 bit EPC implies PC (2 byte) + EPC (12 byte) = 14 bytes (Some tags contain XPC as well) Note that CRC check is done before sending up to host, so that tag with incorrect CRC are removed If FastID is enabled, TID data will follow the EPC.

From above table, a typical simple 96 bit EPC tag packet would be 29 bytes long.

Copyright Convergence Systems Limited 2023 All Rights Reserved

csl_tag_read_epc_only_recurrent

Packet Code: 0x3002

Packet Payload:

Offset	Name	Size	Description
0	UTC Time Stamp	4 bytes	32 bit
4	RSSI	2 bytes	RSSI of the tag response
6	rf_phase_begin	2 bytes	The rf phase of the tag response at the beginning of the packet
8	rf_phase_end	2 bytes	The rf phase of the tag response at the end of the packet
10	Port Number	1 byte	Antenna Port Number
11	Reserve	2 bytes	Reserve
13	Tag_Index	2 bytes	Tag index

From above table, a EPC tag packet would be 15 bytes long.

Copyright Convergence Systems Limited 2023 All Rights Reserved

csl_tag_read_multibank_new

Packet Code: 0x3003

Packet Payload:

Offset	Name	Size	Description
0	UTC Time Stamp	4 bytes	32 bit
4	RSSI	2 bytes	RSSI of the tag reponse
6	rf_phase_begin	2 bytes	The rf phase of the tag response at the beginning of the packet
8	rf_phase_end	2 bytes	The rf phase of the tag response at the end of the packet
10	Port Number	1 byte	Antenna Port number
11	Reserve	2 bytes	Reserve
13	Tag_Index	2 bytes	New assigned Tag index
15	EPC Bank Data	Variable, depends on PC bits	The PC, EPC of the tag. The host must examine the PC bits to determine the number of bytes to include in the EPC.
	Number of Extra Banks	1 byte	N
	Extra Bank Index 1 Data	M bytes	
	…..		
	Extra Bank Index N Data	K	

Copyright Convergence Systems Limited 2023 All Rights Reserved

csl_tag_read_multibank_recurrent_index_only

Packet Code: 0x3004

Packet Payload:

Offset	Name	Size	Description
0	UTC Time Stamp	4 bytes	32 bit
4	RSSI	2 bytes	RSSI of the tag reponse
6	rf_phase_begin	2 bytes	The rf phase of the tag response at the beginning of the packet
8	rf_phase_end	2 bytes	The rf phase of the tag response at the end of the packet
10	Port Number	1 byte	Antenna Port number
11	Reserve	2 bytes	Reserved For Future Use
13	Tag_Index	2 bytes	Tag index

Copyright Convergence Systems Limited 2023 All Rights Reserved

csl_tag_read_compact

Packet Code: 0x3006

Packet Payload:

Offset	Name	Size	Description
0	UTC Time Stamp	4 bytes	32 bit
4	Reserve	2 bytes	Reserve
6	Tag Data	N bytes	Tag Data: PC, EPC, RSSI (2 bytes)

Where N <= 225

Copyright Convergence Systems Limited 2023 All Rights Reserved

csl_miscellaneous_event

Packet Code: 0x3007

Packet Payload:

Offset	Name	Size	Description
0	UTC Time Stamp	4 bytes	32 bit
4	Event Code	2 bytes	Code of the miscellaneous events: 0x0001 = keep alive 0x0002 = inventory round end 0x0003 = CRC error rate (2 bytes Data) 0x0004 = tag rate value (2 bytes Data)
6	Data	N Bytes	

Copyright Convergence Systems Limited 2023 All Rights Reserved

csl_operation_complete

Packet Code: 0x3008

Packet Payload:

Offset	Name	Size	Description
0	UTC Time Stamp	4 bytes	32 bit
4	Command code	2 bytes	Command code
6	Status	2 bytes	0x0000 = Success Error codes references to Appendix B.

Copyright Convergence Systems Limited 2023 All Rights Reserved

csl_access_complete

Packet Code: 0x3009

Packet Payload:

Offset	Name	Size	Description
0	UTC Time Stamp	4 bytes	32 bit
4	Access command	2 bytes	0xC2 = Read 0xC3 = Write 0xC4 = Kill 0xC5 = Lock 0xC6 = Access 0xC7 = Block Write 0xC9 = Block Permalock 0xD5 = Authenticate
6	Tag error code	1 byte	0x00 = Other error 0x01 = Not supported 0x02 = Insufficient privileges 0x03 = Memory overrun 0x04 = Memory locked 0x05 = Crypto suite error 0x06 = Command not encapsulated 0x07 = ResponseBuffer overflow 0x08 = Security timeout 0x0B = Insufficient power 0x0F = Non-specific error 0x10 = No error
7	Mac error code	1 bytes	0x00 = No error 0x01 = No tag reply 0x02 = Invalid password 0x03 = Failed to send command 0x04 = No access reply
8	Write word count	2 bytes	The number of words successfully written.
10	Reserve	2 bytes	Reserve

Copyright Convergence Systems Limited 2023 All Rights Reserved

12	Data	Variable	The tag response data if present

Copyright Convergence Systems Limited 2023 All Rights Reserved

Appendix B – Error Codes

The “status” field of the csl_operation_complete packet comes back with error code field.

Code (hex number)	Description
0x0001	Tag cache table buffer is overflowed.
0x0002	Wrong register address
0x0003	Register length too large
0x0004	E710 not powered up
0x0005	Invalid parameter
0x0006	Event fifo full
0x0007	TX not ramped up
0x0008	Register read only
0x0009	Failed to halt
0x000A	PLL not locked
0x000B	Power control target failed
0x000C	Radio power not enabled
0x000D	E710 command error
0x000E	E710 Op timeout
0x000F	E710 Aggregate error
0x0FFF	Other error

Copyright Convergence Systems Limited 2023 All Rights Reserved

Copyright Convergence Systems Limited 2023 All Rights Reserved

Copyright Convergence Systems Limited 2023 All Rights Reserved

Appendix C: Barcode Reader Command Sequence Examples

C.1 Pre-setup of Barcode Reader

Barcode reader requires presetting it to Trigger Mode once (saving settings in non-volatile memory inside the barcode reader). This is actually done ex-factory and normally not needed to do, unless the barcode reader has been set to other modes by using early version of Apps.

This step is normally NOT required because CS710S has its barcode reader preset to Trigger mode ex-factory.

Step 1: Power on barcode reader using 0x9000 command

Step 2: Use 0x9003 barcode downlink command data API to send the following to CS710S:

nls0006010;

nls0313000=30000;

nls0302000;

nls0006000;

Since 0x9003 can handle 50 bytes of command payload, so the above 4 commands, 50 characters and hence 50 bytes of ascii, can be concatenated and sent down in one shot!!!!

This command will set the barcode reader properly and save that to non-volatile memory. Next time on power up the barcode reader will behave properly - no need to send this again.

Step 3: Power off barcode reader using 0x9001 command

Copyright Convergence Systems Limited 2023 All Rights Reserved

C.2 Normal Operation of Barcode Reader

This is the normal operation step

Step 1: On initial successful connection to CS710S, power on barcode reader using 0x9000 command and leave it on.

Step 2: In barcode reading mode, when the "Start" button or the Trigger button is pressed, use 0x9003 command to send **0x1b, 0x33** down to barcode (only need to send 1 time BT downlink, 2 bytes of payload)

Step 3: In barcode reading mode, when the "Stop" button or the Trigger button is released, use 0x9003 command to send **0x1b, 0x30** down to barcode

Step 4: On final closing of App, then power off the barcode reader using 0x9001 command.

Copyright Convergence Systems Limited 2023 All Rights Reserved

Appendix D – How to Choose Reader Modes

There are 11 reader modes in CS710S and are enumerated as in following table. Only 1 profile is active at any time in CS710S. The purpose of each reader mode is explained below. These purposes correspond to different business case and physical scenarios. The user should try out each profile to see which one gives best performance.

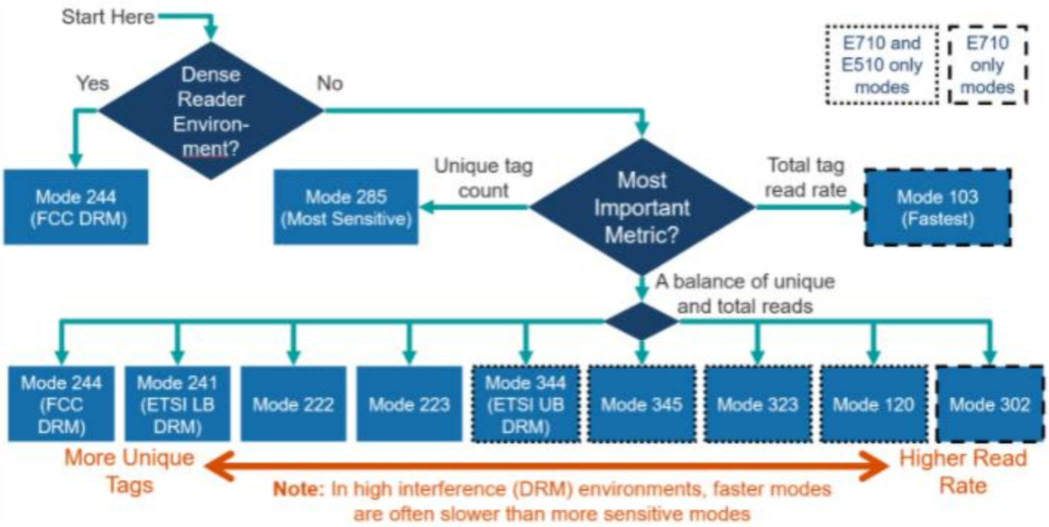
Mode ID	Mode Optimization	Intended Region	Forward Link Modulation	Tari (µs)	BLF (kHz)	Reverse Link Modulation	Chip RX Sensitivity (dBm)	Maximum Read Rate (tags/s)
103	Read Rate	FCC	DSB	6.25	640	Miller M=1	**-78**	**1000**
120	Read Rate	FCC	DSB	6.25	640	Miller M=2	**-81**	**700**
345	Read Rate	FCC, ETSI UB	PR_ASK	7.5	640	Miller M=4	**-84**	**400**
302	Read Rate	All	PR_ASK	7.5	640	Miller M=1	**-78**	**800**
323	Read Rate	All	PR_ASK	7.5	640	Miller M=2	**-81**	**550**
344	ETSI UB DRM	All	PR_ASK	7.5	640	Miller M=4	**-84**	**400**
223	ETSI LB	ETSI	PR_ASK	15	320	Miller M=2	**-84**	**300**
222	ETSI LB	ETSI	PR_ASK	20	320	Miller M=2	**-84**	**250**
241	ETSI LB DRM	ETSI	PR_ASK	20	320	Miller M=4	**-87**	**200**
244	FCC DRM	FCC	PR_ASK	20	250	Miller M=4	**-88**	**150**
285	Sensitivity	All	PR_ASK	20	160	Miller M=8	**-93**	**50**

The following are guidelines for choosing which Reader Mode in the various geographic regions of the world:

Copyright Convergence Systems Limited 2023 All Rights Reserved

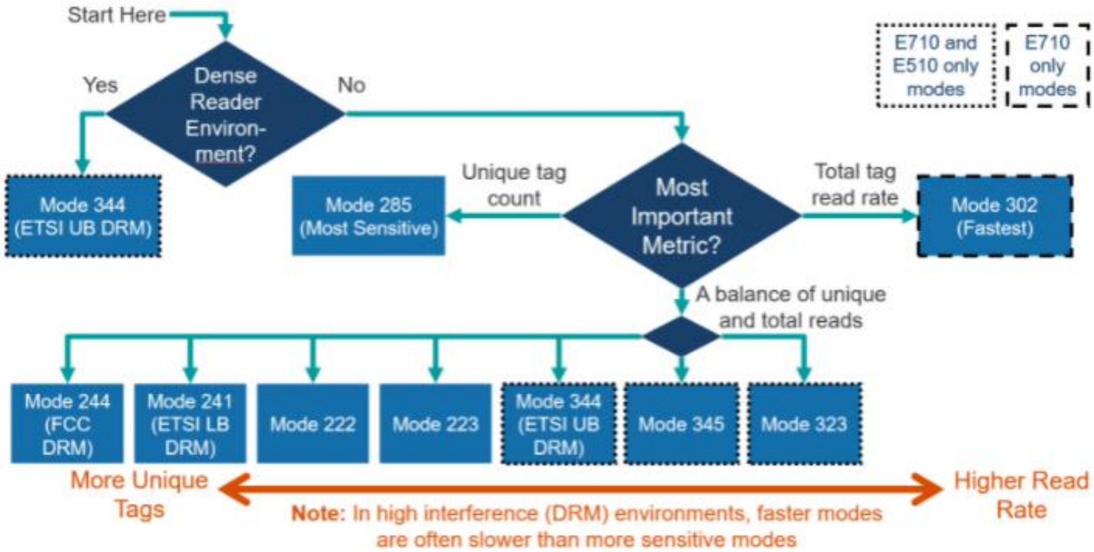
FCC

Figure 1 - Selecting a FW v1.1 Reader Mode in FCC



ETSI Upper Band

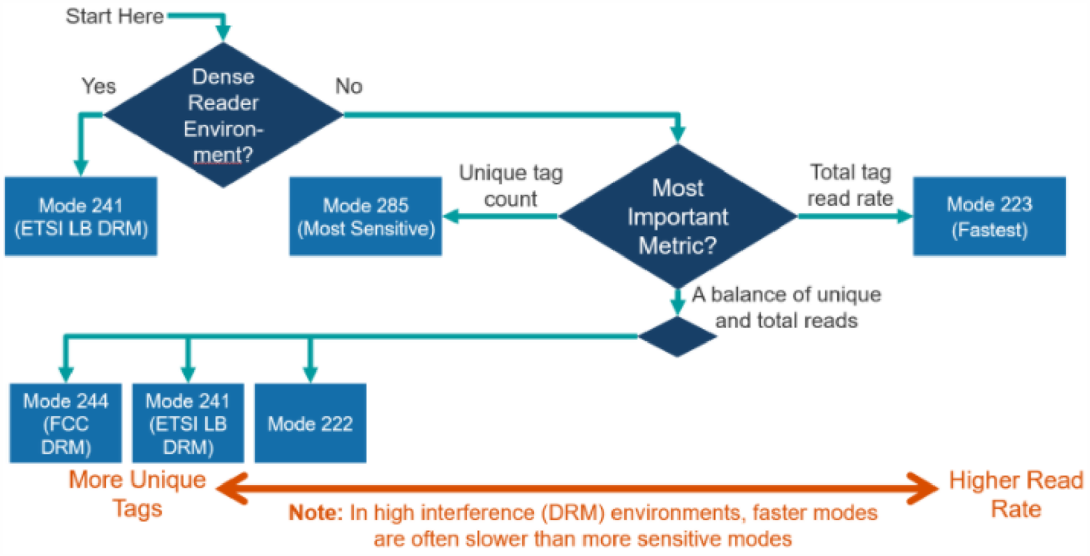
Figure 2 - Selecting a FW v1.1 Reader Mode in ETSI Upper Band (EU2)



Copyright Convergence Systems Limited 2023 All Rights Reserved

ETSI Lower Band, China, Japan, or Korea

Figure 3 - Selecting a FW v1.1 Reader Mode in ETSI Lower Band (EU1) or China, Japan, Korea



Copyright Convergence Systems Limited 2023 All Rights Reserved

Appendix E – Session

Session is a concept of EPC to allow a tag to respond to multiple readers inventorying it at the same time, each using a different session number.

There are 4 possible sessions: S0, S1, S2, S3.

The user however has to be careful because these 4 sessions have different behavior, notably how the tag flag "persist" in time. A tag, before inventory or when just after power on, has a flag of State A. When it is inventoried, the flag will go to State B. The tag flag will stay in State B until the tag powers off or the persistence time is up.

A reader can declare it only wants to inventory flag A, so that after a tag is inventoried and its flag gone to State B, it will no longer respond to further inventory rounds – until the end of the persistence time.

Now for S0, S1, S2 and S3, the persistence times are DIFFERENT! Because of that, one has to be very careful in choosing which session to use.

Session	**Tag Flags Persistence Time**
S0	Tag Energized: indefinite Tag Not Energized: none
S1	Tag Energized: 0.5 second < Persistence Time < 5 seconds Tag Not Energized: 0.5 second < Persistence Time < 5 seconds
S2	Tag Energized: indefinite Tag Not Energized: 2 seconds < Persistence Time
S3	Tag Energized: indefinite Tag Not Energized: 2 seconds < Persistence Time

Copyright Convergence Systems Limited 2023 All Rights Reserved

Appendix F – Tag Population and Q

Tag Population is the RFID tag population that is to be inventoried. To be more precise, it is the population of tags that can be "seen" by the RFID reader.

Q is an EPC concept related to the way a group of tags is inventoried. When a reader broadcast its desire to inventory tags, it sends out a Q value. The tag will, based on that Q, calculate a certain number and define that as the number of repeated inventories the reader will do. Basically, the relationship of Inventory Repeats and Q is:

Inventory Repeats = 2^Q

The tag will then choose by random a certain number less than this Inventory Repeats. When the reader starts doing inventory, the tag will then respond at that repeat number.

In other words, the Inventory Repeats should correspond to Tag Population:

Tag Population = Inventory Repeats = 2^Q

For example, if there are 8 tags, then in theory the Q can be 3, and if each tag chooses a number different from that of the other 7 (miraculously, of course), then the 8 tags will be inventoried in an orderly manner in turn.

Of course this will never happen, as the tags will easily choose a number the same as that of another one, and a collision will happen.

Therefore, it is a normal practice to have a bigger Q, such as 4 in this case, so that the 8 tags would have a lower chance of choosing the same number.

Therefore, reversing the equation, ideally, we can have:

Q = INTEGER(LOG_2(Tag Population))

But in reality, we need some headroom, so that:

Q = INTEGER(LOG_2(Tag Population x 2)+1)

Copyright Convergence Systems Limited 2023 All Rights Reserved

Appendix G – Query Algorithm

There are 2 types of Query Algorithm: Fixed Q and Dynamic Q.

For Fixed Q, the Q value does not change. In other words, the expected Tag Population does not change.

For Dynamic Q, the Q value changes adaptively: when there are a lot of inventory repeats where no tags respond, the reader will interpret that there are not that many RFID tags in the front, and hence it is more efficient to change the Q to a smaller value. When there are a lot of inventory repeats where the reader receive data but they do not satisfy checksum, meaning there is heavy collision, then the reader will interpret that there are too many RFID tags in the front of the reader, and hence it is better to increase the value of Q. Dynamic Q algorithm is a way to allow the RFID reader to adapt to different amount of RFID tags being seen by the reader. The idea is that if there are not so many tags, then the Q can be reduced and the reader can collect all the tag data faster.

Copyright Convergence Systems Limited 2023 All Rights Reserved

Appendix H – Target

Target here actually refers to the target flag that the reader wants to inventory. There are 2 possible flags of an RFID tag: State A and State B.

When an RFID tag is first powered up, it has a flag of State A. After it is inventoried, the state of the flag becomes State B.

The tag will only go back to State A if either it is powered off and powered on again, or if its persistence time has run up.

For each round of inventory, the reader sends out notification to the world which tag flag state it wants to inventory. It can keep on inventory State A, or it can inventory State A and State B alternatively from one round of inventory to the next round of inventory.

In theory, it is a good thing to inventory only State A. The reason being that those tags that have been inventoried should not respond again, and will hence quickly reduce the amount of collision between tags. So in general if you set inventory to State A only, the inventory of large amount of tags can be very fast.

The only catch is that when a tag responds to the reader, it does not know another tag is colliding with it. It sends out the response and thinks it has done the job, hence transitioning to flag State B. So in such case, the tag will not respond to further inventory, even though its response has been lost due to collision. Because of that, sometimes the user will set the inventory to target State A in one inventory round, and then State B in the next round, and vice versa, and so on. This is called A/B Toggle or A & B Dual Target or simply Dual Target.

Copyright Convergence Systems Limited 2023 All Rights Reserved

Appendix I – Security

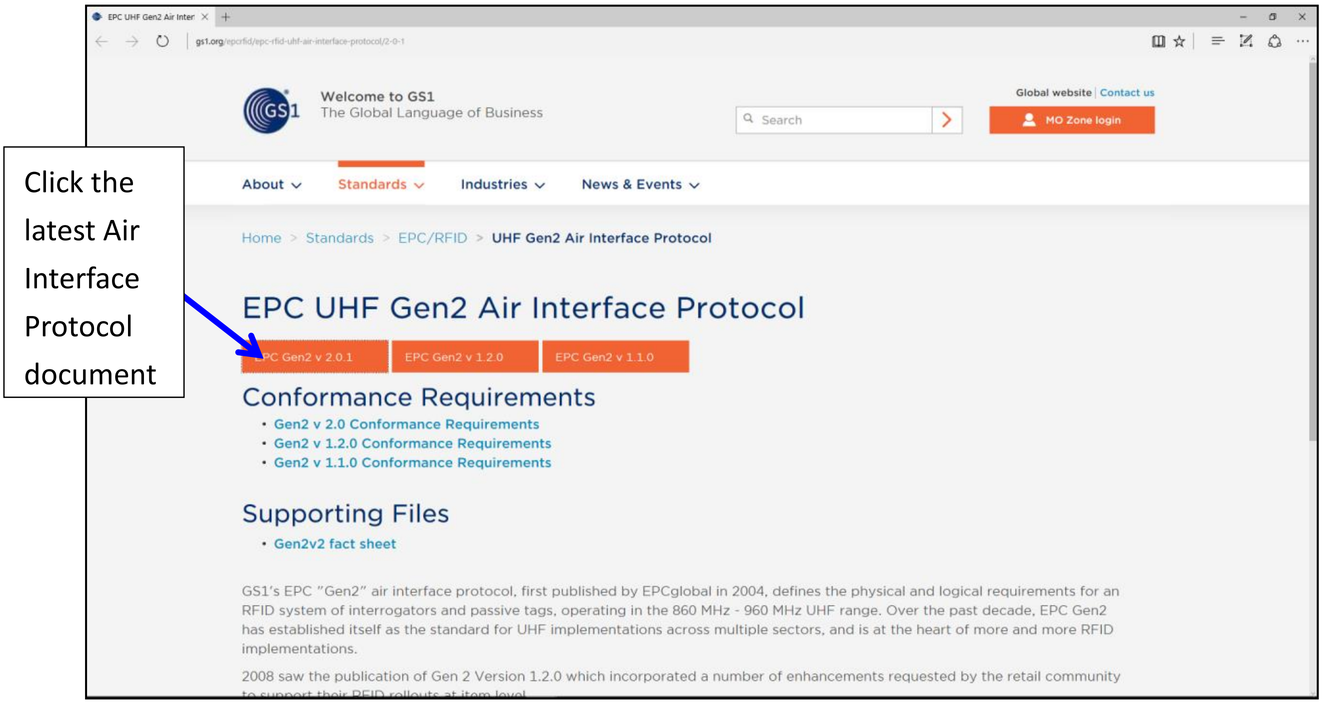
There are 4 actions you can apply on the memory inside an RFID tag:

1) Lock
2) Unlock
3) Permanent Lock
4) Permanent Unlock

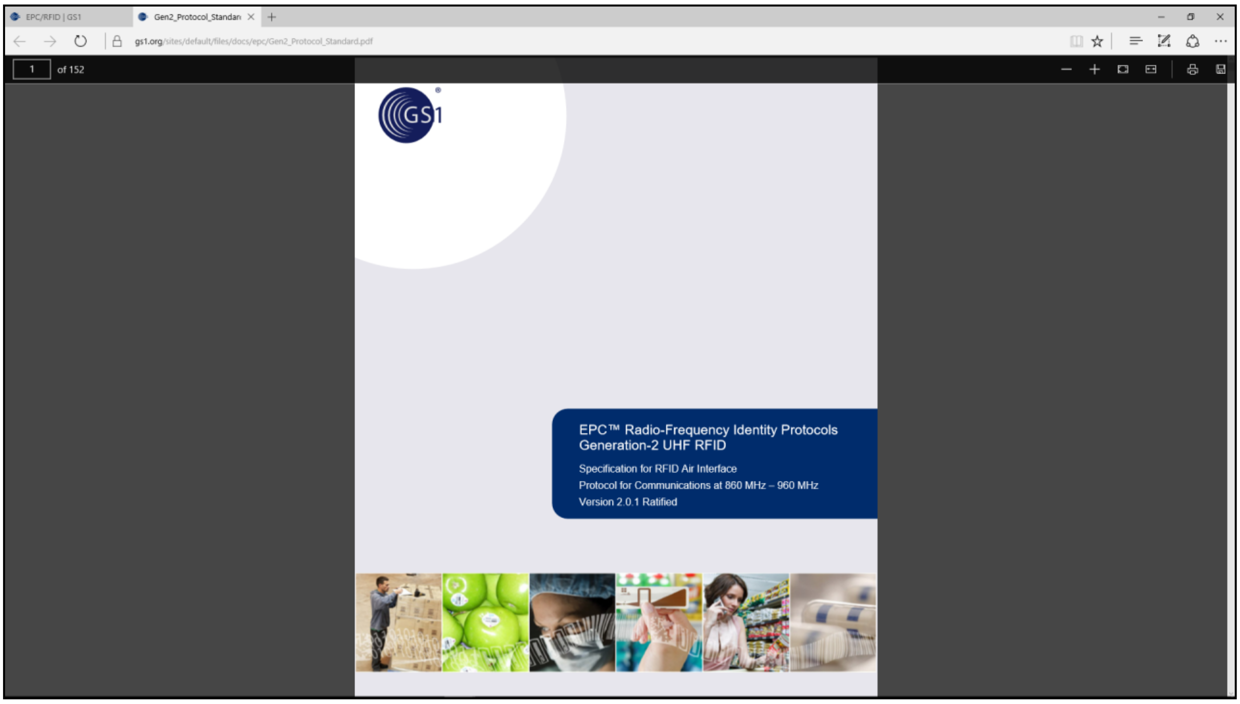
You can obtain an EPC Global document which can be downloaded from the EPC GLobal website that explains this:

.

Once there, press the button showing the latest air interface protocol document and click on it to get the pdf file.



Copyright Convergence Systems Limited 2023 All Rights Reserved



For the Access Password and Kill Password the security locking affects both reading and writing.

For the EPC memory bank and the User memory bank, the security locking affects only writing.

For the TID memory bank, since we are the user and not the manufacturing vendor, there is no security action that can be applied. It has been permanently unlocked in the factory and it cannot be changed.

Copyright Convergence Systems Limited 2023 All Rights Reserved

Appendix J – Tag Focus

Tag Focus is a special feature of Impinj tag IC. When enabled, and when the reader is using Session S1 and Target A to query the tag, the tag will, once inventoried, remain in Flag B until the inventory is completed.

This is in contrast to the normal EPC query using S1 and Target A, where the tag will only remain in Flag B for 2 to 5 seconds – this time being defined as persistence time.

The original purpose of EPC S1 and Target A is so that those tags that have been inventoried before would not come back and be inventoried again so quickly, so that more time slots are available for other tags that have not been inventoried yet. However, the time of 2 to 5 seconds are simply too short if there are many tags in the environment being illuminated by the reader. This is particularly true if the reader is "seeing" a whole bunch of tagged items in a warehouse, where there may be more than 1000 tags the reader can "see" at any moment. The end result is before all the tags have been inventoried, the early inventoried tags, having passed 5 seconds after it was first inventoried, join back to be inventoried!!

In the early days of EPC standard, a few hundred tags being inventoried is already a rather unthinkable matter, and so 2 to 5 seconds of persistence time is enough. Nowadays, with the ever improving sensitivity of tags, 5 seconds is not enough.

With Impinj Tag Focus enabled, the tag simply would not respond again until the inventory is completely over.

Copyright Convergence Systems Limited 2023 All Rights Reserved

Appendix K – Models & Regulatory Regions

There are various models, denoted by the alphanumeric key to the right of the dash after the "CS710S-", here denoted by "**N**". The applicable regulatory regions for each model are described below:

N=1:	865-868 MHz for Europe ETSI, Russia, Middle East countries, 865-867 MHz for India
N=2:	902-928 MHz, FCC, for USA, Canada and Mexico. Hopping frequencies locked
N=2 AS:	920-926 MHz, Australia. Hopping frequencies locked
N=2 NZ:	921.5-928 MHz, New Zealand. Hopping frequencies locked
N=2 OFCA:	920-925 MHz, Hong Kong. Hopping frequencies locked
N=2 RW:	920-928 MHz, Rest of the World, e.g. Philippines, Brazil, Peru, Uruguay, etc.
N=4:	922-928 MHz, Taiwan
N=6:	917-920.8 South Korea
N=7:	920-925 MHz, China
N=8:	916.7-920.9 MHz, Japan
N=9:	915-921 MHz, Europe Upper Band

Copyright Convergence Systems Limited 2023 All Rights Reserved

Appendix L – CRC Table/Compute Codes

The following is the CRC lookup table and compute code.

```
const unsigned int xdata crc_lookup_table[256] =
{
  0x0000,0x1189,0x2312,0x329b,0x4624,0x57ad,0x6536,0x74bf,
  0x8c48,0x9dc1,0xaf5a,0xbed3,0xca6c,0xdbe5,0xe97e,0xf8f7,
  0x1081,0x0108,0x3393,0x221a,0x56a5,0x472c,0x75b7,0x643e,
  0x9cc9,0x8d40,0xbfdb,0xae52,0xdaed,0xcb64,0xf9ff,0xe876,
  0x2102,0x308b,0x0210,0x1399,0x6726,0x76af,0x4434,0x55bd,
  0xad4a,0xbcc3,0x8e58,0x9fd1,0xeb6e,0xfae7,0xc87c,0xd9f5,
  0x3183,0x200a,0x1291,0x0318,0x77a7,0x662e,0x54b5,0x453c,
  0xbdcb,0xac42,0x9ed9,0x8f50,0xfbef,0xea66,0xd8fd,0xc974,
  0x4204,0x538d,0x6116,0x709f,0x0420,0x15a9,0x2732,0x36bb,
  0xce4c,0xdfc5,0xed5e,0xfcd7,0x8868,0x99e1,0xab7a,0xbaf3,
  0x5285,0x430c,0x7197,0x601e,0x14a1,0x0528,0x37b3,0x263a,
  0xdecd,0xcf44,0xfddf,0xec56,0x98e9,0x8960,0xbbfb,0xaa72,
  0x6306,0x728f,0x4014,0x519d,0x2522,0x34ab,0x0630,0x17b9,
  0xef4e,0xfec7,0xcc5c,0xddd5,0xa96a,0xb8e3,0x8a78,0x9bf1,
  0x7387,0x620e,0x5095,0x411c,0x35a3,0x242a,0x16b1,0x0738,
  0xffcf,0xee46,0xdcdd,0xcd54,0xb9eb,0xa862,0x9af9,0x8b70,
  0x8408,0x9581,0xa71a,0xb693,0xc22c,0xd3a5,0xe13e,0xf0b7,
  0x0840,0x19c9,0x2b52,0x3adb,0x4e64,0x5fed,0x6d76,0x7cff,
  0x9489,0x8500,0xb79b,0xa612,0xd2ad,0xc324,0xf1bf,0xe036,
  0x18c1,0x0948,0x3bd3,0x2a5a,0x5ee5,0x4f6c,0x7df7,0x6c7e,
  0xa50a,0xb483,0x8618,0x9791,0xe32e,0xf2a7,0xc03c,0xd1b5,
  0x2942,0x38cb,0x0a50,0x1bd9,0x6f66,0x7eef,0x4c74,0x5dfd,
  0xb58b,0xa402,0x9699,0x8710,0xf3af,0xe226,0xd0bd,0xc134,
  0x39c3,0x284a,0x1ad1,0x0b58,0x7fe7,0x6e6e,0x5cf5,0x4d7c,
  0xc60c,0xd785,0xe51e,0xf497,0x8028,0x91a1,0xa33a,0xb2b3,
  0x4a44,0x5bcd,0x6956,0x78df,0x0c60,0x1de9,0x2f72,0x3efb,
  0xd68d,0xc704,0xf59f,0xe416,0x90a9,0x8120,0xb3bb,0xa232,
  0x5ac5,0x4b4c,0x79d7,0x685e,0x1ce1,0x0d68,0x3ff3,0x2e7a,
  0xe70e,0xf687,0xc41c,0xd595,0xa12a,0xb0a3,0x8238,0x93b1,
  0x6b46,0x7acf,0x4854,0x59dd,0x2d62,0x3ceb,0x0e70,0x1ff9,
```

Copyright Convergence Systems Limited 2023 All Rights Reserved

```
  0xf78f,0xe606,0xd49d,0xc514,0xb1ab,0xa022,0x92b9,0x8330,
  0x7bc7,0x6a4e,0x58d5,0x495c,0x3de3,0x2c6a,0x1ef1,0x0f78
};

// -----------------------------------------------------------------------
// ComputeCRC
//
// This function computes the CRC returns the 16-bit CRC
// -----------------------------------------------------------------------
unsigned int ComputeCRC(unsigned char* input, unsigned int length, unsigned int init)
{
    unsigned int CRC;
    unsigned int i;

    CRC = init;
    for (i = 0; i < length; i++)
    {
      CRC = UpdateCRC (CRC, input[i]);
    }
    return CRC;
}

// -----------------------------------------------------------------------
// UpdateCRC
//
// This function accepts a CRC argument and a <newbyte> and returns an
// updated CRC value; Uses the CRC Lookup Table
// -----------------------------------------------------------------------
unsigned int UpdateCRC (unsigned int crc, unsigned char newbyte)
{
    unsigned short retval;
    unsigned short index;
    unsigned short table_value;
    index = (crc ^ newbyte) & 0xff ;
    table_value = crc_lookup_table[index];
    retval = (crc >> 8) ^ table_value;
    return retval;
}
```

Copyright Convergence Systems Limited 2023 All Rights Reserved