PMD17059-AC

DRAFT - INTERNAL

Smart Readers with Orange Pack ID

Configuration and Software Integration

www.springcard.com

Document Identification

Category	Development guide		
Family/Customer	Smart Readers with Orange Pack ID		
Reference	PMD17059	Version	AC
Status	Draft	Classification	Internal
Keywords			
Abstract			

File name	V:\Dossiers\SpringCard\A-Notices\RFID scanners et lecteurs\SpringBlue\[PMD17059-AC] RDR with Orange PackID-Configuration and Software Integration.odt		
Date saved	05/06/19	Date printed	07/03/17

REVISION HISTORY

Ver.	Date	Author	Valid. by		Approv. by	Details
			Tech.	Qual.		
AA	07/03/17	CRA			JDA	Creation
AB	07/06/17	JDA				Now implementing secure version of the protocol
AC	05/06/19	JDA				Detailed configuration described in chapter 5

Contents

Illustrations

Tables

1. Introduction

1.1. Abstract

Orange Pack ID is an application framework designed to create a smartphone-based, hand-free tag for identification, access control or e-wallet applications. **Orange Pack ID** uses Bluetooth 4.0 Smart *aka* Bluetooth Low Energy (BLE) for communication between the smartphone and a compliant Reader.

SpringCard has developed the **Twist'N'Blue**, a versatile board with a BLE interface to experiment and evaluate the technology. A firmware has been written to implement in the **Twist'N'Blue** the Reader side of the **Orange Pack ID** system.

This document shows how to configure the **Twist'N'Blue with Orange Pack ID** firmware to test and demonstrate the concept.

1.2. Related products

At the date of writing, the **Twist'N'Blue** doesn't target mainstream deployment and is therefore listed under "Products in development".

SpringCard is also able to port the **Orange Pack ID** system in all its products featuring a BLE interface, provided that there are actual market opportunities for it.

For instance, the system has been ported and tested in next-generation **FunkyGate** (wall-mount access control reader) which provides both NFC and BLE interfaces in a single shell.

1.2.1. Products in development

Firmware	Product	Order code	Description
K663/BLE	Twist'N'Blue TTL *ORANGE* Twist'N'Blue 232 *ORANGE* Twist'N'Blue 485 *ORANGE*	SC17051 SC17052 SC17053	OEM BLE-to-PC interface board with Orange Firmware
S663/RDR+BLE	FunkyGate DW NFC+BLE *ORANGE*		Contactless/RFID/NFC/BLE DataClock, Wiegand and RS-485 wall-mounted Smart Reader with Orange Firmware

Table 1: List of products currently in development, supporting Orange Pack ID

The fact that a product is in development doesn't guarantee that it will ever be released!

1.3. Related documents

EDITOR	DOCUMENT #	TITLE / VERSION
SpringCard	PNA17045	Twist'N'Blue Harware Integration Guide
SpringCard	PMA13205	Smart Readers and RFID Scanners Templates

Table 2: List of related documents

1.4. Audience

This manual is designed for use by application developers and system integrators. It assumes that the reader has a good knowledge of computer development.

1.5. Disclaimers

- **SpringCard Products featuring the Orange Pack ID technology have not been certified by Bluetooth SIG, and have not endorsed CE nor FCC certification. They shall be used for evaluation purposes only.**
- As evaluation-only products, the purchase of these products carries with it no warranties, either expressed or implied.
- While every care has been taken to provide quality products, we cannot guarantee that these products will function correctly together with all Bluetooth Low Energy devices; the products may not operate or may operate improperly with some Bluetooth Low Energy devices.
- **SpringCard** does not take any responsibility for leakage of information during Bluetooth Low Energy communication.

1.6. Support and updates

Useful related materials (product datasheets, application notes, sample software, HOWTOs and FAQs…) are available at SpringCard's web site:

www.springcard.com

Updated versions of this document and others are posted on this web site as soon as they are available.

For technical support enquiries, please refer to SpringCard support page, on the web at

www.springcard.com/support

2. Concepts and transaction principles

2.1. Basics

The aim of the **Orange Pack ID** protocol is to address to the use case of user identification using the smartphone and a BLE Reader:

1. A user has a smartphone with the application “Pack ID BLE” installed.
2. An administrator assigns this user an **ID**, associated to access rights on an area controlled by an access control system.
3. The user install his **ID** in his phone.
4. The user comes near to a BLE Reader.
5. The phone detects the Reader and opens a BLE communication with it.
6. The Reader fetches the ID and transmit it to the access control system (typically through a serial port). On version 1.75 and onward, the transmission is authenticated by AES keys.
7. The access control system decides whether the access will be granted or not, and provides this information back to the Reader.
8. The Reader in turns forward the information to the phone.
9. The phone shows the information to the user.

2.2. Protocol Description – Common part

The smartphone is the BLE Central and the Reader is the BLE peripheral.

Once the connection is established, the smartphone is the GATT client and the Reader is the GATT server.

As a BLE peripheral the reader broadcasts advertisement frames continuously. Each advertisement contains the UUID of the **Orange Pack ID BLE Service**, the **Company ID** of the access control system, the **Contactless ID** (or Service ID) of the customer’s facilities where the system is installed, and the **Zone ID** (or Access ID) of the area or subdivision controller the Reader.

- The **Company ID** is a 2-byte array
- The **Contactless ID** is a 4-byte array
- the **Zone ID** is a 6-byte array.

As the GATT server the Reader exposes the **Orange Pack ID BLE Service** (either secure or unsecure) in its GATT.

2.3. Protocol description – Unsecure version

This **Orange Pack ID BLE Service (Unsecure)** has 2 characteristics.

The characteristic **Pack ID identifier** (write-only) is used by the client -the smartphone- to transmit its ID to the Reader.

The characteristic **Confirmation message** (read-only) is used by the Reader to transmit a text message to be displayed by the smartphone.

As a BLE central, the smartphone waits for advertisements frames.

When an advertisement frame is received, the **Orange Pack ID** application running in the smartphone immediately checks that it belongs to a compliant Reader thanks to the advertised **Pack ID BLE Service**. It also checks whether the advertised **Company ID** and **Access ID** are compatible with the user's data stored in the smartphone.

If the **Company ID** and **Access ID** match, the smartphone connects to the Reader, and explore its GATT to retrieve the list of services and characteristics.

The smartphone then write the user's **Pack ID** identifier into the corresponding characteristic.

The Reader receive this **Pack ID**, and forward it to the access control unit for validation of the access rights.

After that, the phone starts reading the **Confirmation message** characteristic. This text information is displayed to the user.

The demonstration setup with the **Twist'N'Blue** doesn't involve an access control unit. Therefore, the **Confirmation message** will always read "Tag read".

Care must been taken when writing the smartphone application that an actual access control system may take up to a few seconds before providing an answer.

At the end, the smartphone close the BLE connection and wait a few seconds before scanning again.

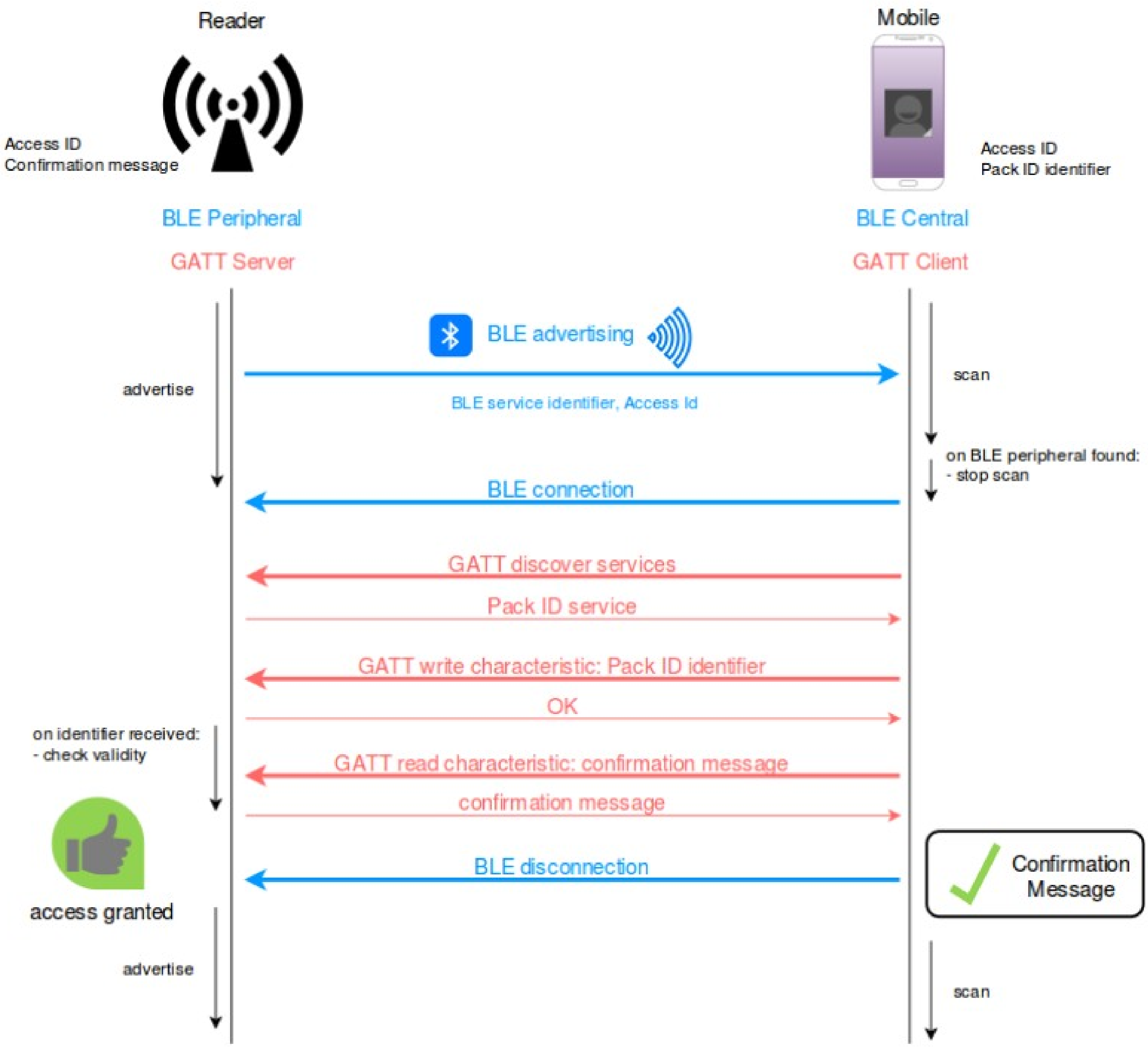

Illustration 1: Workflow diagramm (source Orange)

3. BLE implementation

3.1. BLE roles and connection principles

When using the Pack ID BLE application the reader is configured as a BLE Peripheral and as the GATT server. It will advertise until a smartphone connect to him a begin the exchange, and it will restart the advertising at the end of the dialog.

On the other side the phone is the BLE central and the GATT client. The smartphone connects to the reader and is able to read or write the characteristics exposed through its GATT. The communication takes place in unpaired, not-bound mode. The security is implemented at application level, not at communication level.

3.2. Advertising data of the Orange Pack ID BLE Reader

3.2.1. Advertisement frame

Descriptor #1		
Len	Type	Data
$_{h}$02	$_{h}$01	$_{h}$05
Flags		- LE Limited Discoverable Mode - No BR/EDR (BLE only)

Descriptor #2		
Len	Type	Data
$_{h}$03	$_{h}$03	$_{H}$FA 18
16-bit Service UUIDs		Service Pack ID

Descriptor #3		
Len	Type	Data
$_{h}$09	$_{h}$FF	$_{H}$FF FF 00 00 00 EE EE EE
Manufacturer specific data		- **Company ID** : $_{h}$FFFF (generic values for testing) - Additional manufacturer specific data : **Access ID** (6 bytes containing the area controlled by the BLE reader).

Descriptor #4		
Len	Type	Data
${}_h$02	${}_h$0A	${}_H$F9
TX Power Level		Example: F9 = -7 dBm (${}_h$100 – ${}_h$F9 = ${}_h$07)

3.2.2. Scan response data

Descriptor #1		
Len	Type	Data
${}_h$16	${}_h$09	${}_H$4F 72 61 6E 67 65 20 50 61 63 6B 20 49 44 20 58 58 58 58 58 58
Complete Local Name		Local Name: “Orange Pack ID XXXXXX” (XXXXXX are the last 3 bytes of the MAC address of the reader)

3.3. GATT profile of the Orange Pack ID BLE Reader

UUID	Mnemonic	Access	Description
Generic Attribute			
1801	org.bluetooth.service.generic_attribute		
2A05	org.bluetooth.characteristic.gatt.service_changed	Read Indicate	Informative means that the text may provide background or context to the authoritative text contained in the adopted Bluetooth specification. Informative text is not considered when determining compliance to the Bluetooth specification. Please refer to the adopted Bluetooth specification for the normative (i.e. authoritative) text used to establish compliance. Compliance issues due to errors on this web page are not the responsibility of the Bluetooth SIG and rest solely with the member.

UUID	Mnemonic	Access	Description
Generic Access Profile			
1800	org.bluetooth.service.generic_access		
2A00	org.bluetooth.characteristic.gap.device_name	Read	“Orange Pack ID XXXXXX” *(XXXXXX are the last 3 byte of the mac address)*
2A01	org.bluetooth.characteristic.gap.appearance	Read	

UUID	Mnemonic	Access	Description
Device Information			
180A	org.bluetooth.service.device_information		
2A29	org.bluetooth.characteristic.manufacturer_name_string	Read	“SpringCard” *(not changeable)*
2A24	org.bluetooth.characteristic.model_number_string	Read	“Twist’N’Blue” *(not changeable)*
2A25	org.bluetooth.characteristic.serial_number_string	Read	“xxxxxxxxxxxx” *(BT_ADDR in hex)*
2A26	org.bluetooth.characteristic.firmware_revision_string	Read	The version of the BlueGecko stack in the BGG113/BGM11 “xxxx yyyy zzzz” (major version, minor version, patch level)
2A28	org.bluetooth.characteristic.software_revision_string	Read	“MM.mm”
2A50	org.bluetooth.characteristic.pnp_id	Read	Vendor ID Source = ${}_h$02 Vendor ID = ${}_h$1C34 Product ID = ${}_h$ABD0 Product Version = ${}_h$*MMmm*

UUID	Mnemonic	Access	Description
Tx Power			
1804	org.bluetooth.service.tx_power		
2A07	org.bluetooth.characteristic.tx_power_level	Read	The Transmit Power Level characteristic represents the current transmit power level in dBm, and the level ranges from -100 dBm to +20 dBm to a resolution of 1 dBm.

UUID	Name	Access	Description
Service Pack ID			
000018FA0000-1000-8000-00805f9b34f	*Service Pack ID*		
00002AFF-0000-1000-8000-00805f9b34f	Pack ID identifier	Write	This characteristic is used to received the Pack ID identifier coming from the phone. When the reader receive this characteristic (when a write is achieved) the feature is activated (open a door for example)
00002AFE-0000-1000-800000805f9b34fb	Confirmation message	Read	Characteristic by the reader to write a message that will be transmit to the phone after that he sent the ID. The format must be utf-8.

4. Setting up a Twist'N'Blue to evaluate Orange Pack ID

4.1. The Twist'N'Blue board

SpringCard Twist'N'Blue is an OEM BLE-to-PC interface board. It has been created to prototype, experiment and benchmark Bluetooth Low Energy (BLE) applications.

Please refer document PMD17049 to read how to use the **Twist'N'Blue** board and how to connect it to a computer. The document PNA17045 – SpringCard Twist'N'Blue Hardware Integration guide might be useful as well.

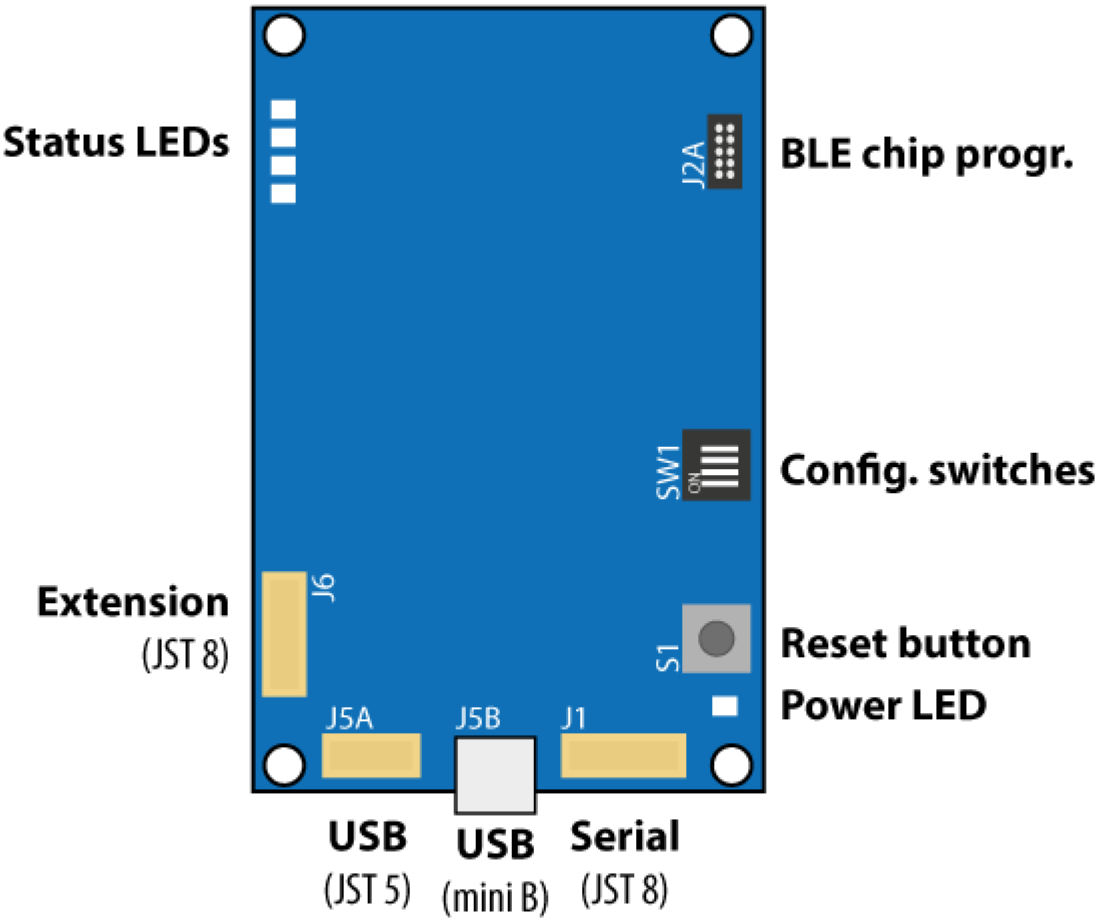

Illustration 2: The Twist'N'Blue PCB (top view)

4.2. CONNECTING THE TWIST'N'BLUE TO THE COMPUTER

There are 2 options to connect the **Twist'N'Blue** to a computer:

- Through USB (J5A or J5B)
- Through a Serial port (J1).

4.2.1. Using the USB port

The **Twist'N'Blue** features 2 USB connectors, that could be used indifferently to connect the product to a host computer. J5B is a standard USB mini B plug, J5A is a 5-pin JST connector adapted to industrial systems.

> *DO NOT connect both USB connectors (J5A and J5B) at the same time.*
> *DO NOT connect anything to the Serial interface (J1) when the USB interface (J5A or J5B) is in use.*

The **Twist'N'Blue**'s USB interface is implemented through a USB-to-Serial bridge, part number **FTDI FT232RQ**.

Please go to http://www.ftdichip.com/Drivers/VCP.htm to download the latest driver for your operating system.

After installing the FTDI driver, the **Twist'N'Blue** is seen by the computer as a serial port (*COMxx* on Windows, */dev/ttyUSBX* on Unix).

4.2.2. Using the Serial line

Depending on the components mounted on the PCB, the J1 Serial communication port could provide and accommodate either

- TTL (0/5V) or CMOS (0/3.3V) RX/TX signals for products with "TTL" in the name
- RS-232 (-6V/+6V) RX/TX signals for products with "232" in the name
- RS-485 Bus A/Bus B signals for products with "485" in the name

Only the RS-232 version may be connected "directly" to a computer's standard RS-232 port (DB9 plug).

> *There's no visible difference between a RS-TTL, RS-232 and RS-485 version. Please verify carefully the product's label to identify which of the 3 Serial versions is actually mounted.*

> *DO NOT connect anything to the Serial interface (J1) when the USB interface (J5A or J5B) is in use.*

4.2.3. Communication parameters

The same communication parameters apply to both USB and Serial:

- baudrate = 38400bps,
- 8 data bits,
- 1 stop bit,
- no parity,
- no flow control.

The baudrate could be changed by changing the Configuration Register SER ($_h$67). The other parameters are fixed.

4.2.4. Validating your hardware integration

The easiest way to test your installation is to use a **terminal emulation software** running on a desktop or laptop computer.

Popular terminal emulation software are **HyperTerminal** and **HTerm** on Microsoft Windows, and **minicom** on Linux.

Open your terminal emulation software, set the communication parameter as specified above, and press the **Twist'N'Blue**'s reset button.

You must see the Reader's startup string "**SpringCard K663/BLE Orange**" (§ 4.5.2.a).

Enter the string "info" (without the quotes), and hit the ENTER key. The Reader sends its information data in response.

If one of those two tests fails, please double-check your hardware (wiring, power supply...) and the port number you've selected on the computer.

4.2.5. Used ASCII characters

The constants used in the following paragraphs are specified in the ASCII standard:

ASCII constant	Hex value	Description
`[STX]`	$_h$01	Start of header
`[STX]`	$_h$02	Start of text
`[ETX]`	$_h$03	End of text
`[ACK]`	$_h$06	Positive acknowledge
`[BEL]`	$_h$07	Bell or ring
`[TAB]`	$_h$09	Horizontal tabulation
`[LF]`	$_h$0A	Line feed
`[CR]`	$_h$0D	Carriage return
`[NAK]`	$_h$15	Negative acknowledge
`[ESC]`	$_h$1B	Escape

Tablea3: List of ASCII constants used for Serial communication

4.3. The Console

The Reader features a “human” command processor (shell or console). This feature is primarily made for testing and demonstration purpose. Only the few commands depicted in this chapter could safely be used for configuration and diagnostic.

4.3.1. Sending a Console command to the Reader

Enter “info” to verify that your communications parameters are correct. If not, go back to § 4.2.4.

If the Reader answers, you're now ready to communicate with it using the commands listed below.

> *Note that the Reader does not echo the entered characters; you should activate the local echo in your terminal-emulation software to see what you are typing.*
> *The Reader accepts any end-of-line marker: [CR] alone, [LF] alone as well as [CR][LF] are valid.*

4.3.2. List of Console commands

Command	Meaning
`version`	Show the firmware version
`info`	Show the firmware information data
`show`	Show the current configuration
`cfg`	Dump all Configuration Registers written into persistent memory
`cfgXX=YY...YY`	Write value $_h$YY...YY to Configuration Register $_h$XX
`cfgXX=!!`	Erase Configuration Register $_h$XX
`cfgXX`	Read Configuration Register $_h$XX
`exit`	Leave the Console (send ESCAPE twice to before next Console commands)
`echo on`	Turn echo ON
`echo off`	Turn echo OFF

Table 4: List of commands supported by the Console

4.3.3. Using the console to edit Reader's configuration

The Reader's configuration is stored in a set of non-volatile Configuration Registers. There are three groups of Registers:

- The Global Registers control the global behaviour of the Reader. They are stored on addresses greater than $_h$60 and are specific to every Reader. The **Twist'N'Blue** running the **K663/BLE** firmware shares the same Global Registers as **K663/RDR** firmware. Only the SER register (§ 4.4) is used.
- The Template Registers that control how the Reader fetches data from contactless cards. They are stored on addresses $_h$10 to $_h$5F. Since the **Twist'N'Blue** has no contactless interface, these registers are not used only for the card output format.
- The **Orange Pack ID** Registers that control how the Reader fetches data from an **Orange Pack ID**-compliant BLE smartphone. They are stored on addresses $_h$01 to $_h$0F and are documented in chapter 5.
- The BLE registers are used to configure the tx power level, the advertisement and connection parameters... They are also stored on addresses $_h$01 to $_h$0F and are documented in chapter 6.

a. Reading Configuration Registers

Enter "`cfg`" to list all Configuration registers currently defined (registers that are not explicitely defined keep their default value).

Enter "`cfg`XX" to read the value of the Configuration register $_h$XX, from $_h$01 to $_h$FE.

Note that Configuration registers that hold sensitive data are masked by 'X' characters when read-back.

b. Writing Configuration Registers

Enter "`cfg`XX=YYYY" to update Configuration Register $_h$XX with value $_h$YYYY. YYYY can have any length between 1 and 32 bytes.

Enter "`cfg`XX=`!!`" to erase Configuration Register $_h$XX.

4.4. Configuring the Serial line through the SER register

For the moment the only available baudrate is 38400bps, in the future this will change.

Name	Tag	Description	Size
SER	$_h67$	Serial configuration bits. See table below	1

Serial configuration bits

Bits	Value	Meaning
7	0	No STX / ETX frame markers
	1	Use STX and ETX as frame markers
6 - 5	00	No BEL / TAB / CR/LF frame markers
	01	Use CR/LF only
	10	Use BEL and CR/LF as frame markers
	11	Use TAB and CR/LF as frame markers
		Serial Repeat
4 - 3	00	No repeat
	01	Repeat 4 times with timeout of 100ms (Host must send an ACK to cancel)
	10	Repeat 4 times with timeout of 250ms (Host must send an ACK to cancel)
	11	Repeat 9 times with timeout of 250ms (Host must send an ACK to cancel)
		Baudrate
2 - 0	000	1200bps
	001	2400bps
	010	4800bps
	011	9600bps
	100	19200bps
	101	38400bps
	110	RFU
	111	115200bps

Default value: $_b11000101$

4.5. THE MK1 READER PROTOCOL

4.5.1. Overview

The MK1 Reader Protocol is the protocol used by the **Twist'N'Blue** to notify the host computer that a ***Pack ID identifier*** has been read. The protocol also allows the host to drive the **Twist'N'Blue**'s LEDs and buzzer.

The MK1 Reader Protocol is very simple and "human-readable" since it relies on ASCII-printable chars. This protocol is made for 1-to-1, peer-to-peer communication. It doesn't provide any kind of collision avoidance or collision detection feature, and therefore its reliability on a RS485 link is poor.

4.5.2. Reader → Host notifications

a. *Startup string*

When configured to use the MK1 Reader Protocol, upon startup, the Reader sends its name and version, for instance

```
SpringCard K663/BLE Orange
```

The startup string is always terminated by `[CR][LF]` (carriage-return, line-feed).

The host application may ignore the content of the startup string, but shall wait for the [CR] [LF] sequence that terminates it before sending any command to the reader. Any character coming over the serial line before the end of the startup string will be discarded.

b. *Notification when a Pack ID identifier is read*

When a credential is read, it is formatted according to the settings of the TOF register (§ 5.3).

The formatted credential is then transmitted by the Reader over the serial link as follow:

```
<BEGIN SEQUENCE><PREFIX><CREDENTIAL><END SEQUENCE>
```

Where

- PREFIX is the content of the PFX register (§ 5.4) if not empty,
- BEGIN SEQUENCE and END SEQUENCE fields are configured by bits 7-5 of Register SER (§ 4.4), according to the following table:

Bits 7-6-5 in SER	BEGIN SEQUENCE	END SEQUENCE	Example with ***Pack ID identifier*** = “0123456789ABCDEF”
000	*(empty)*	*(empty)*	0123456789ABCDEF
001	*(empty)*	[CR][LF]	0123456789ABCDEF[CR][LF]
010	[BEL]	[CR][LF]	[BEL]0123456789ABCDEF[CR][LF]
011	[TAB]	[CR][LF]	[TAB]0123456789ABCDEF[CR][LF]
100	[STX]	[ETX]	[STX]0123456789ABCDEF[ETX]
101	[STX]	[ETX][CR][LF]	[STX]0123456789ABCDEF [ETX][CR][LF]
110	[BEL][STX]	[ETX][CR][LF]	[BEL][STX]0123456789ABCDEF[ETX][CR][LF]
111	[TAB][STX]	[ETX][CR][LF]	[TAB][STX]0123456789ABCDEF[ETX][CR][LF]

Table 5: Impact of SER bits 7-6-5 on the *Pack ID identifier* output frame

c. Host's ACK

It is recommended that the Host sends the ASCII “Acknowledge” character ($_h$06) after receiving a valid credential output frame.

```
[ACK]
```

If the Reader is configured to repeat it output until the host has acknowledged it (bits 4-3 of SER register), the host’s ACK stops the repetition.

4.5.3. Host → Reader commands (and Reader's ACK)

a. Command/response sequences

The Host may send any of the commands listed in § 4.5.3.c. The command frame has no prefix, and is terminated by [CR][LF]:

```
<COMMAND>[CR][LF]
```

b. Reader's ACK

When a valid command is received from the Host, the Reader sends the ASCII "Acknowledge" character ($_h$06) within 50ms.

```
[ACK]
```

When an invalid command is received or a communication error occurs, the Reader sends the ASCII "Not Acknowledge" character ($_h$15) within 100ms after having detected the error.

```
[NAK]
```

c. List of commands

Command	Meaning
A0	Stop Reader
A1	Start Reader
R0	Red LED is switched OFF
R1	Red LED is switched ON
R2	Red LED blinks slowly
R3	Red LED blinks quickly
G0	Green LED is switched OFF
G1	Green LED is switched ON
G2	Green LED blinks slowly
G3	Green LED blinks quickly
Z0	Buzzer stops
Z1	Buzzer starts
Z2	Short buzzer sound
Z3	Long buzzer sound
C	Clear LED / buzzer (same as sending R0, G0, Z0)

Table 6: List of commands supported by the MK1 Reader Protocol

5. Configuration of an Orange Pack ID BLE Reader

This chapter details the configuration registers used by the Reader to support **Orange Pack ID** over BLE.

5.1. Company Identifier, Contactless Identifier and Zone Identifier

The Pack ID BLE service has 3 main configuration parameters:

- A **Company ID** on 2 bytes,
- A **Contactless ID** (or Service ID) on 4 bytes,
- A **Zone ID** (or Access ID) on 6 bytes.

The register $_h$03 (LOC) is used to store these 3 parameters, on a single 12-B value.

Pack ID Company ID, Contactless ID and Zone ID – Address: $_h$03, size: 12 bytes

Byte	Meaning	Notes	Default value
Byte 0-2			
0	**Company ID**	MSB	`43 05`
1		LSB	
Byte 2-5			
2	**Contactless ID**	MSB	`00 00 00 01`
3			
4			
5		LSB	
Bytes 6-11			
6	**Zone ID**	MSB	`0F 0F 0F 0F 0F 0F`
7			
8			
9			
10			
11		LSB	

5.2. Master key

In secure mode, the Pack ID BLE service is protected by an AES-128 master key.

This key is provided by register $_h$05 (AUT).

Pack ID Master Key – Address: $_{h}$05, size: 16 bytes

Byte	Meaning	Notes	Default value

0	AES-128 master key		`11 22 33 44 55 66 77 88`
			`99 00 AA BB CC DD EE FF`

...			
15			

5.3. Orange Pack ID Size and format of output (TOF register)

This register controls the output format, see document PMA13205 Smart Readers and RFID Scanners Template System for more informations.

5.4. Orange Pack ID Prefix (PFX register)

The PFX register stores a constant value that the reader will use to prefix the data.

PFX for Orange Pack ID – Address: $_{h}$02, size: 0 to 8 bytes

Uses the PFX register to transmit an arbitrary (constant) string before the data returned by this Template.

6. Advanced BLE configuration

6.1. Tx Power Level

The register $_h$0A can be used to set the TX power. Once configured the value will be used on the advertisement data and the GATT value will be updated. TX power in 0.1dBm steps, for example the value of 10 is 1 dBm and 60 is 6 dBm. The value is on 16 bits and the maximum is 140 (14 dBm), and the default value is 100 (10 dBm).

Tx Power Level – Address: $_h$0A, size: 2 bytes

Byte	Meaning	Notes / Valid range
Byte 0-2		
0	**Tx Power Level**	MSB
1		LSB

Default value: $_h$64

6.2. Minimum RSSI

You can ignore some devices too far by setting the minimum RSSI value in the register $_h$09. When a device try to connect to the reader the reader check the RSSI and if it's below the minimum, the reader automatically disconnect the device. The value is a signed byte (for example $_h$C4 is -60), the default value is -50.

Minimum RSSI – Address: $_h$09, size: 1 bytes

Byte	Meaning	Notes / Valid range
Byte 0		
0	**Minimum RSSI**	Signed Byte

Default value: $_h$CE

6.3. Advertisement Parameters

The register $_h$0B is used to set the advertisement parameters. The field is composed of 5 bytes:

the minimum and maximum connection interval on 16 bits and the channel map on only 8 bits.

The format of this field is AAAABBBBCC (A for the minimum interval, B for the maximum and C for the channel map).

Advertisement Parameters – Address: $_{h}$0B, size: 5 bytes

Byte	Meaning	Notes / Valid range
Byte 0-2		
0	**Minimum connection interval**	MSB
1		LSB
Byte 2-4		
2	**Maximum connection interval**	MSB
3		LSB
Byte 4		
4	**Channel map**	

Default value: $_{h}$0032006407

Minimum connection interval.

Value in units of 0.625 ms

- Range: $_{h}$0020 to $_{h}$4000 (connectable advertising)
- Time range: 20 ms to 10.24 s (connectable advertising)
- Range: $_{h}$00a0 to $_{h}$4000 (non-connectable advertising)
- Time range: 100 ms to 10.24 s (non-connectable advertising)

Maximum connection interval.

Value in units of 0.625 ms

- Range: $_{h}$0020 to $_{h}$4000
- Time range: 20 ms to 10.24 s
- Note: interval_max must be at least equal to or bigger than interval_min

Advertisement channel map

Determines which of the three channels will be used for advertising. This value is given as a bit-mask.

Values:

- 1: Advertise on CH37
- 2: Advertise on CH38
- 3: Advertise on CH37 and CH38
- 4: Advertise on CH39
- 5: Advertise on CH37 and CH39

- 6: Advertise on CH38 and CH39
- 7: Advertise on all channels
- Recommended value: 7

Default values are 50 (31,25 ms) for the minimum interval, 100 (62,5 ms) for the maximum and 7 for the channel map.

6.4. Connection Parameters

The register $_h$0C is used for the connection parameters. It's composed of 4 values of 16 bits:

the minimum and the maximum value for the connection event interval, the slave latency and the supervision timeout. The format of this field is AAAABBBBCCCCDDDD (A for the minimum interval, B for the maximum and C for the slave latency and D for the timeout).

Connection Parameters – Address: $_h$0C, size: 8 bytes

Byte	Meaning	Notes / Valid range
Byte 0-2		
0	**Minimum connection interval**	MSB
1		LSB
Byte 2-4		
2	**Maximum connection interval**	MSB
3		LSB
Byte 4-6		
4	**Slave latency**	MSB
5		LSB
Byte 6-8		
6	**Supervision timeout**	MSB
7		LSB

Default value: $_h$00060018000100E

Minimum value for the connection event interval.

This must be set be less than or equal to max_interval.

- Time = Value x 1.25 ms
- Range: $_h$0006 to $_h$0c80
- Time Range: 7.5 ms to 4 s

Maximum value for the connection event interval.

This must be set greater than or equal to min_interval.

- Time = Value x 1.25 ms
- Range: $_h$0006 to $_h$0c80
- Time Range: 7.5 ms to 4 s

Slave latency.

This parameter defines how many connection intervals the slave can skip if it has no data to send

- Range: $_h$0000 to $_h$01f4

Supervision timeout.

The supervision timeout defines for how long the connection is maintained despite the devices being unable to communicate at the currently configured connection intervals.

- Range: $_h$000a to $_h$0c80
- Time = Value x 10 ms
- Time Range: 100 ms to 32 s
- The value in milliseconds must be larger than (1 + latency) * max_interval * 2, where max_interval is given in milliseconds It is recommended that the supervision timeout is set at a value which allows communication attempts over at least a few connection intervals.

The default value for the minimum connection interval is 6, 24 for the maximum, 1 for the latency and 30 for the supervision timeout.

Disclaimer

This document is provided for informational purposes only and shall not be construed as a commercial offer, a license, an advisory, fiduciary or professional relationship between SPRINGCARD and you. No information provided in this document shall be considered a substitute for your independent investigation.

The information provided in document may be related to products or services that are not available in your country.

This document is provided "as is" and without warranty of any kind to the extent allowed by the applicable law. While SPRINGCARD will use reasonable efforts to provide reliable information, we don't warrant that this document is free of inaccuracies, errors and/or omissions, or that its content is appropriate for your particular use or up to date. SPRINGCARD reserves the right to change the information at any time without notice.

SPRINGCARD doesn't warrant any results derived from the use of the products described in this document. SPRINGCARD will not be liable for any indirect, consequential or incidental damages, including but not limited to lost profits or revenues, business interruption, loss of data arising out of or in connection with the use, inability to use or reliance on any product (either hardware or software) described in this document.

These products are not designed for use in life support appliances, devices, or systems where malfunction of these product may result in personal injury. SPRINGCARD customers using or selling these products for use in such applications do so on their own risk and agree to fully indemnify SPRINGCARD for any damages resulting from such improper use or sale.

Copyright notice

Editor's information

SPRINGCARD SAS company with a capital of 227 000 €

RCS EVRY B 429 665 482

Parc Gutenberg, 2 voie La Cardon

91120 Palaiseau – FRANCE

Contact information

For more information and to locate our sales office or distributor in your country or area, please visit

www.springcard.com