



PMD17049-BA
DRAFT - INTERNAL

SMART READERS WITH SPRINGBLUE

Configuration and Software Integration

DOCUMENT IDENTIFICATION

Category	Development Manual		
Family/Customer	Smart Readers with BLE		
Reference	PMD17049	Version	BA
Status	Draft	Classification	Internal
Keywords			
Abstract			

File name	V:\Dossiers\SpringCard\A-Notices\RFID scanners et lecteurs\SpringBlue\[PMD17049-BA] RDR with SpringBlue - Configuration and Software Integration.odt		
Date saved	19/06/17	Date printed	02/03/17

REVISION HISTORY

Ver.	Date	Author	Valid. by		Approv. by	Details
			Tech.	Qual.		
AA	01/03/17	JDA				Creation
BA	19/06/17	JDA				New implementation with secure communication

CONTENTS

1. INTRODUCTION.....	6	5. THE SPRINGBLUE DEMO APPLICATION ON ANDROID & IOS.....	31
1.1. ABSTRACT.....	6	5.1. INSTALLING THE APPLICATION ON ANDROID.....	31
1.2. RELATED PRODUCTS.....	7	5.2. INSTALLING THE APPLICATION ON IOS.....	31
1.2.1. Available products.....	7	5.3. INSERTING A CREDENTIAL.....	31
1.2.2. Products in development.....	7	5.4. USING THE APPLICATION WITH A SPRINGBLUE BLE READER.....	33
1.3. RELATED DOCUMENTS.....	8	5.5. USING THE APPLICATION WITH A SPRINGBLUE NFC READER.....	34
1.4. AUDIENCE.....	8	5.6. DEMO DATA.....	34
1.5. DISCLAIMERS.....	8	6. CONFIGURATION OF A SPRINGBLUE BLE READER.....	35
1.6. SUPPORT AND UPDATES.....	8	6.1. SPRINGBLUE BLE INSTALLATION ID (LOC REGISTER).....	35
2. CONCEPTS AND TRANSACTION PRINCIPLES.....	9	6.2. SPRINGBLUE BLE AUTHENTICATION KEY (AUT REGISTER).....	35
2.1. OVERVIEW.....	9	6.3. SPRINGBLUE BLE SIZE AND FORMAT OF OUTPUT (TOF REGISTER). 36	
2.2. SMARTPHONE APPLICATION, CREDENTIAL STORED IN THE SMARTPHONE. 9		6.4. SPRINGBLUE BLE PREFIX (PFX REGISTER).....	37
2.2.1. NFC.....	9	6.5. DEMO DATA.....	37
2.2.2. BLE.....	9	7. ADVANCED BLE CONFIGURATION.....	39
2.3. CONFIGURATION OF THE READER.....	10	7.1. TX POWER LEVEL.....	39
2.4. THE SPRINGBLUE TRANSACTION.....	10	7.2. MINIMUM RSSI.....	39
2.5. DETAIL OF THE APDUS.....	11	7.3. ADVERTISEMENT PARAMETERS.....	39
2.5.1. Select Application.....	11	7.4. CONNECTION PARAMETERS.....	41
2.5.2. Security Activation (Internal Authenticate).....	12		
2.5.3. The reader ask for the <i>UserID</i> associated with the <i>InstallationId</i>	12		
2.5.4. Test vectors example.....	13		
2.5.5. General error codes.....	14		
2.6. CRYPTOGRAPHIC ASPECTS.....	14		
3. BLE IMPLEMENTATION.....	15		
3.1. BLE ROLES AND CONNECTION PRINCIPLES.....	15		
3.2. ADVERTISEMENT DATA OF THE SPRINGBLUE BLE READER.....	16		
3.2.1. Advertisement frame.....	16		
3.3. GATT PROFILE OF THE SPRINGBLUE BLE READER.....	17		
3.3.1. Standard services and characteristics.....	18		
3.3.2. SpringBlue service and characteristic.....	20		
4. SETTING UP A TWIST'N'BLUE TO EVALUATE SPRINGBLUE. 21			
4.1. THE TWIST'N'BLUE BOARD.....	21		
4.2. CONNECTING THE TWIST'N'BLUE TO THE COMPUTER.....	22		
4.2.1. Using the USB port.....	22		
4.2.2. Using the Serial line.....	22		
4.2.3. Communication parameters.....	23		
4.2.4. Validating your hardware integration.....	23		
4.2.5. Used ASCII characters.....	24		
4.3. THE CONSOLE.....	25		
4.3.1. Sending a Console command to the Reader.....	25		
4.3.2. List of Console commands.....	25		
4.3.3. Using the console to edit Reader's configuration.....	25		
4.4. CONFIGURING THE SERIAL LINE THROUGH THE SER REGISTER.....	27		
4.5. THE MK1 READER PROTOCOL.....	28		
4.5.1. Overview.....	28		
4.5.2. Reader → Host notifications.....	28		
4.5.3. Host → Reader commands (and Reader's ACK).....	29		

ILLUSTRATIONS

Illustration 1 : Workflow of SpringBlue transaction.....	11
Illustration 2: Full workflow of SpringBlue BLE transaction.....	16
Illustration 3: The Twist'N'Blue PCB (top view).....	21

TABLES

Table 1: List of products supporting SpringBlue.....	7
Table 2: List of products currently in development, supporting SpringBlue.....	7
Table 3: List of related documents.....	8
Tablea4: List of ASCII constants used for Serial communication.....	24
Table 5: List of commands supported by the Console.....	25
Tableau 6: Impact of SER bits 7-6-5 on the UserID output frame.....	29
Table 7: List of commands supported by the MK1 Reader Protocol.....	30

1. INTRODUCTION

1.1. ABSTRACT

SpringBlue is an innovative identification scheme, targeting smartphone applications, where the user's ID is pushed securely to a **SpringBlue**-enabled Reader, either through NFC Host Card Emulation (HCE) or through Bluetooth Smart, *aka* BLE (Bluetooth Low Energy).

SpringBlue has been designed not only with security but also with privacy in mind: only the system owning an ID is able to read it. **SpringBlue** opens numerous use cases related to user identification: physical access control, loyalty programs, car park, car sharing or bike sharing schemes, and more.

The first part of this document introduces **SpringBlue** architecture and details the exchanges between the smartphone and the Reader through either BLE or HCE.

The second part covers using SpringCard **Twist'N'Blue** board and the **SpringBlue Demo Application** (Android / iOS) to start evaluating **SpringBlue**.

The last part focuses on how to configure the Reader to operate **SpringBlue** over BLE with custom parameters and security keys – **SpringBlue** over NFC being covered by the Template system of any SpringCard Smart Reader.

1.2. RELATED PRODUCTS

1.2.1. Available products

At the date of writing, **SpringBlue** is implemented in the following SpringCard products:

Firmware	Product	Order code	Description	NFC	BLE
E663/RDR	FunkyGate IP NFC	SC14002	Contactless/RFID/NFC over Ethernet wall-mounted Smart Reader	✓	
	FunkyGate IP+POE NFC	SC14003	Contactless/RFID/NFC over Ethernet wall-mounted Smart reader (powered by the network)	✓	
	TwistyWriter IP RDR	SC16091	OEM contactless/RFID/NFC over Ethernet Smart reader (remote antenna)	✓	
S663/RDR	FunkyGate DW NFC	SC14004	Contactless/RFID/NFC DataClock, Wiegand and RS-485 wall-mounted Smart Reader	✓	
K663/RDR	K663/RDR	SC13116	OEM contactless/RFID/NFC Smart Reader serial module (without antenna)	✓	
	K663-TTL/RDR K663-232/RDR K663-485/RDR	SC13118 SC13119 SC14181	Ready-to-use OEM contactless/RFID/NFC Smart Reader with serial interface	✓	
H663/RDR	Prox'N'Roll HSP RFID Scanner	SC15132	Desktop USB Smart Reader for contactless/RFID/NFC cards	✓	

Table 1: List of products supporting SpringBlue

1.2.2. Products in development

Firmware	Product	Order code	Description	NFC	BLE
S663/RDR+BLE	FunkyGate DW NFC+BLE <i>BLUE</i>		Contactless/RFID/NFC/BLE DataClock, Wiegand and RS-485 wall-mounted Smart Reader with SpringBlue Firmware	✓	✓
K663/BLE	Twist'N'Blue TTL <i>BLUE</i> Twist'N'Blue 232 <i>BLUE</i> Twist'N'Blue 485 <i>BLUE</i>	SC16236 SC16237 SC17050	OEM BLE-to-PC interface board with SpringBlue Firmware		✓

Table 2: List of products currently in development, supporting SpringBlue

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 Integration Guide
SpringCard	PMD17128	SpringBlue ID BLE/NFC identification solution Developer's Implementation Manual
SpringCard	PMA13205	Smart Readers and RFID Scanners Templates

Table 3: 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

- **Products featuring the SpringBlue 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. OVERVIEW

Merchants and service providers are identified by a unique **Siteld** (on 4 bytes). Every user “belonging” to this merchant or service provider is identified by a unique **Userld** (on 8 bytes).

The **SpringBlue ID** Reader is configured to read only the **Userld** belonging to a given **Siteld**.

A security scheme (based on AES block cipher and using two secret keys) allows to verify that the **Userld** credential comes from a trusted source. It prevents both cloning and replay attacks.

2.2. SMARTPHONE APPLICATION, CREDENTIAL STORED IN THE SMARTPHONE

A user's smartphone may hold any number of tuples (**Siteld**, **Userld**, **secret keys**). They are typically stored in an embedded database such as SQLite. The keys especially shall be protected against unauthorised readings by software countermeasures.

The smartphone itself is identified by a pseudo-unique **ObjectID**, which is required to secure the transaction, but that is not returned by the Reader to the downstream system.

2.2.1. NFC

This mode is available only on smartphones featuring an NFC interface and supporting HCE (Host Card Emulation).

The **SpringBlue ID** NFC-enabled application is registered as HCE application provider and associated to the **SpringBlue ID** AID (Application Identifier) thanks to the application's manifest.

When the smartphone is waved over an NFC Reader that looks for this very AID, the smartphone's operating system wakes up the application, and the transaction takes place: the **SpringBlue ID** NFC Reader sends Command (C-APDU) through the contactless (NFC) interface, and retrieves the Responses (R-APDU) from the smartphone application through the same channel.

2.2.2. BLE

Running in a smartphone supporting BLE, the **SpringBlue ID** BLE-enabled application looks for a compliant **SpringBlue ID** BLE Reader in the nearby, either periodically or following an action from the user.

Once such a Reader has been found, the **SpringBlue ID** BLE-enabled application opens a BLE connection onto the Reader, and the transaction takes place.

The transaction uses a bi-directional GATT characteristic which mimics the behaviour of a smart card channel:

- At first, the smartphone application writes a so-called ATR to tell the **SpringBlue ID** BLE Reader that it is something worth communicating then,
- The **SpringBlue ID** BLE Reader puts its Command (C-APDU) in the characteristic, and uses BLE Indication to notify the smartphone that there's something to read,
- The smartphone application Reads the characteristic to retrieve the Command, and executes it,
- The smartphone application Writes its Response (R-APDU) back into the same characteristic.
- The dialog continue with the **SpringBlue ID** BLE Reader notifying the availability of a new Command, or requesting to close the connection.

2.3. CONFIGURATION OF THE READER

The **SpringBlue ID** Reader has 3 configuration data:

- **SiteId**,
- Site's secret key to decode the **ObjectId** transmitted by the smartphone (**OSUK**),
- Site's master key to decode the **UserId** transmitted by the smartphone (**MSUK**). The **MSUK** and the **ObjectId** are combined by the Reader to retrieve the smartphone's **SOIK**, which is the key actually used to protect the **UserId**.

Handling the **UserId** is eventually the responsibility of the host system. The host system may be a computer, a PLC or an access-control unit; its role is to know what to do with any given **UserId**: point to a record in a database, log a notification, open a door – the Reader just doesn't care.

3. SETTING UP A TWIST'N'BLUE TO EVALUATE SPRINGBLUE ID

3.1. 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 to document PMD17049 to read how to use the **Twist'N'Blue** board and how to connect it to a computer.

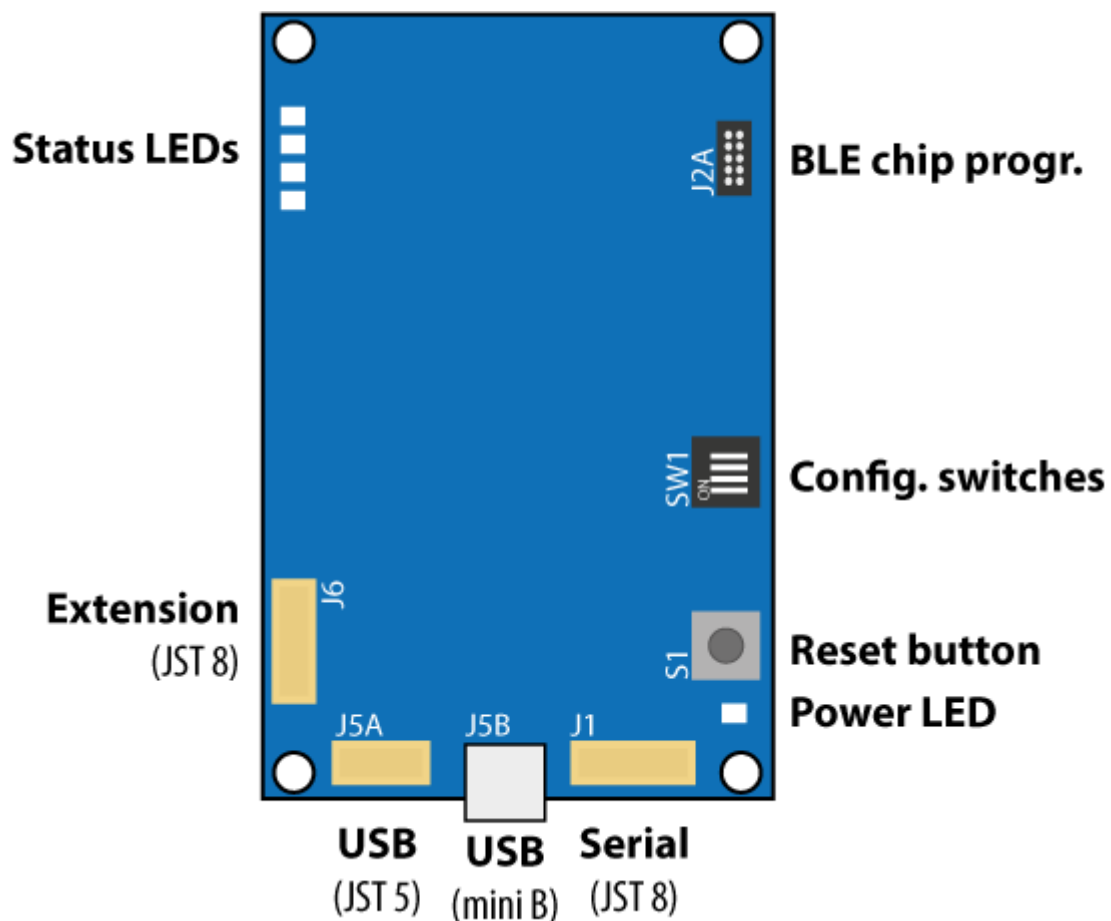


Illustration 1: The Twist'N'Blue PCB (top view)

3.2. 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).

3.2.1. 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).

3.2.2. 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.

3.2.3. 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 (_h67). The other parameters are fixed.

3.2.4. 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 Twistnblue 1.74**" (§ 3.5.2.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.

3.2.5. Used ASCII characters

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

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

Tablea4: List of ASCII constants used for Serial communication

3.3. 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.

3.3.1. Sending a Console command to the Reader

Enter “info” to verify that your communications parameters are correct. If not, go back to § 3.2.4.

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.

3.3.2. 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 5: List of commands supported by the Console

3.3.3. 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 (§ 3.4) 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.
- The **SpringBlue** BLE Registers that control how the Reader fetches data from a SpringBlue-compliant BLE smartphone. They are stored on addresses $_{h}01$ to $_{h}0F$ and are documented in chapter 5.

a. Reading Configuration Registers

Enter “cfg” to list all Configuration registers currently defined (registers that are not explicitly defined keep their default value).

Enter “cfgXX” 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 “cfgXX=YYYY” to update Configuration Register $_{h}XX$ with value $_{h}YYYY$. YYYY can have any length between 1 and 32 bytes.

Enter “cfgXX=! ” to erase Configuration Register $_{h}XX$.

3.4. 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	_h 67	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
4 - 3		Serial Repeat
	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)
2 - 0		Baudrate
	000	1200bps
	001	2400bps
	010	4800bps
	011	9600bps
	100	19200bps
	101	38400bps
	110	RFU
	111	115200bps

Default value: _b11000101

3.5. THE MK1 READER PROTOCOL

3.5.1. Overview

The MK1 Reader Protocol is the protocol used by the **Twist'N'Blue** to notify the host computer that a **UserID** 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.

3.5.2. 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 SpringBlue 1.70

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 UserID is read

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

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

<BEGIN SEQUENCE><PREFIX><USER ID><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 (§ 3.4), according to the following table:

Bits 7-6-5 in SER	BEGIN SEQUENCE	END SEQUENCE	Example with <i>UserID</i> = "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]

Tableau 6: Impact of SER bits 7-6-5 on the *UserID* output frame

c. Host's ACK

It is recommended that the Host sends the ASCII "Acknowledge" character ($_{h06}$) after receiving a valid *UserID* 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.

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

a. Command/response sequences

The Host may send any of the commands listed in § 3.5.3.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 ($_{h06}$) within 50ms.

[ACK]

When an invalid command is received or a communication error occurs, the Reader sends the ASCII "Not Acknowledge" character ($_{h15}$) 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 7: List of commands supported by the MK1 Reader Protocol

4. THE SPRINGBLUE DEMO APPLICATION ON ANDROID & IOS

4.1. INSTALLING THE APPLICATION ON ANDROID

To install the SprinBlue application on a Android phone or tablet, you can go on the Google play and download the application at this address:

<https://play.google.com/store/apps/details?id=com.springcard.SpringBlue>

4.2. INSTALLING THE APPLICATION ON IOS

To install the SprinBlue application on a Apple phone or tablet, you can go on the Apple store and download the application at this address:

<https://itunes.apple.com/us/app/springblue/id1022837202?mt=8>

4.3. INSERTING A CREDENTIAL

When you open the SpringBlue application for the first time you have no credentials installed (**UserId**, **InstallationId** and **InstallationAuthKey**). To do so, you need to open the menu (the 3 horizontal lines at the top left of the screen), then go to the section “My Tags”.

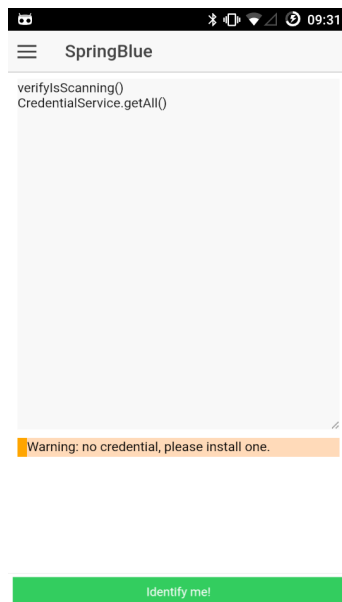


Illustration 2: Main screen
without credentials

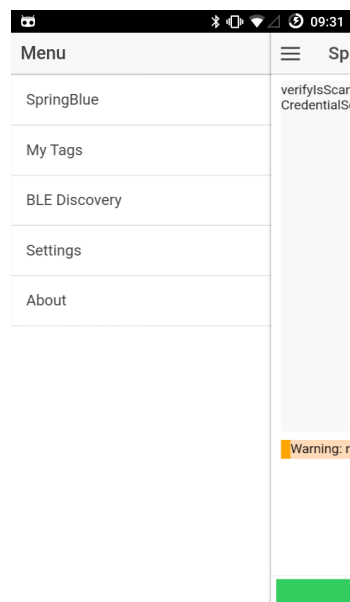


Illustration 3: Menu



Illustration 4: My tags screen

Once in "My tags" section, click on the "Add" button.

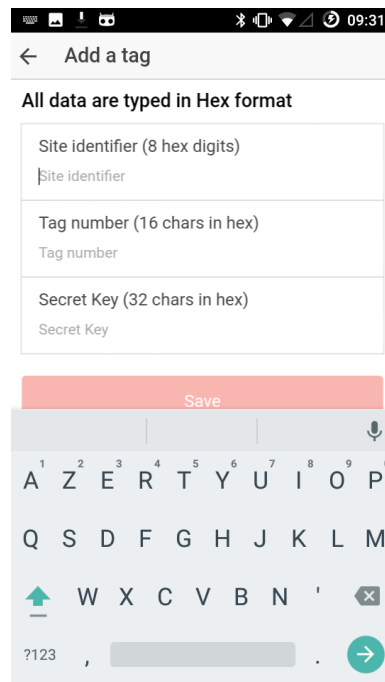


Illustration 5: Adding a new tag

It will open a new window that will ask you to fill the fields. The Site Identifier correspond to the **InstallationId**, Tag Number to the **UserId** and Secret key to **InstallationAuthKey**.

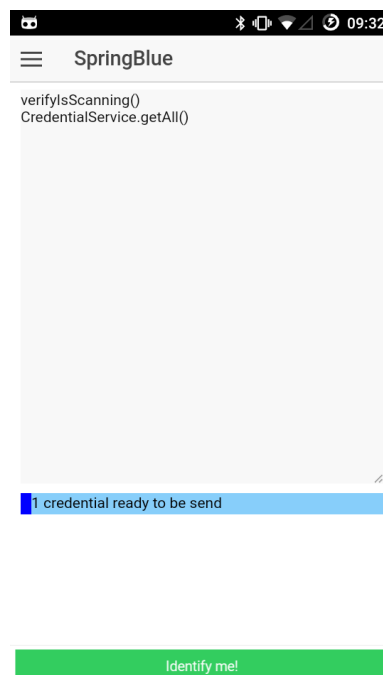


Illustration 6: Main screen with one credential ready

Once you go back to the main screen, It will indicate you that a credential is ready to be sent.

4.4. USING THE APPLICATION WITH A SPRINGBLUE BLE READER

Turn on Bluetooth and Localization on your phone then open the SpringBlue application.

If a credential is ready to be send and if you reader is configured (see chapter § 5) then you just have to click “identify me” and the application will search for the nearest SpringBlue reader and communicate with it.

If you connect to the console you will see this output:

```

Received Data
1 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 8
gecko_evt_le_connection_opened_id: 0x0C, 0xFF from 70:38:08:6A:6A:97
No security
Unmanaged: 000900a0
Indication request received: 0x19
2 Rcv = 25 123B8E01805C537072696E67426C756530315D
values received = BLE: length = 18
BLE: received ATR=3B8E01805C537072696E67426C756530315D
BLE: constant ATR=3B8E01805C537072696E67426C756530315D
BLE: ATR correct
BLE: data send = 00A4040010A000000614537072696E67426C756530
BGM_SendData result = 1
4 Rcv = 25 029000
values received = BLE:Response=9000
BLE: data send = 00880000107487C2958669E0F8B252DB5B3138905D
BGM_SendData result = 1
6 Rcv = 25 029000
values received = BLE:Response=9000
BLE: data send = 00A4010004000000001
BGM_SendData result = 1
8 Rcv = 25 196F15A513BF0C106A31D0C4FF6417E9DD87F9E2387462349000
values received = Security check
key=A0A1A2A3A4A5A6A7A8A9AAABACADAEAF
challenge=7487C2958669E0F8B252DB5B3138905D
Valid ID: 0102030405060708
000102030405060708
gecko_evt_le_connection_closed_id: 0x0C
Advertising started
BGM_sendDisconnectEvent
Advertising started

```

Illustration 7: Output of the condole with log trace

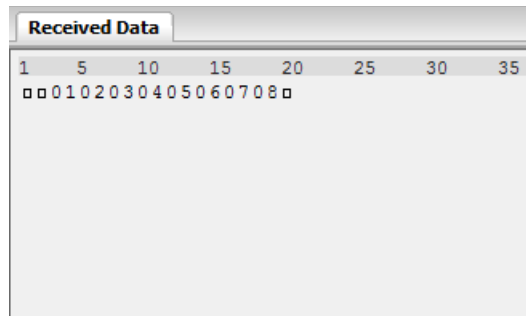


Illustration 8: Output of the console without log trace

4.5. USING THE APPLICATION WITH A SPRINGBLUE NFC READER

Turn on the NFC in the settings of your phone then open the SpringBlue application. If a credential is already installed and the phone is configured to detect a SpringBlue phone via the Template system. Then it will send the **UserId** to the reader when you put the phone in front of the reader.

4.6. DEMO DATA

To use the demo credentials you have to go to the “My Tags” screen and click on “Insert demo Tag” it will add a credential with the default values.

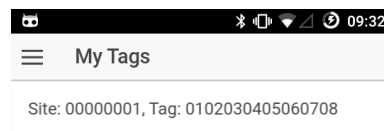


Illustration 9: Demo tag added

5. CONFIGURATION OF A SPRINGBLUE ID BLE READER

This chapter details the configuration registers used by the Reader to support **SpringBlue ID** over BLE. The configuration of **SpringBlue ID** over NFC (HCE) is documented in [PMA13205](#), chapter “SpringBlue ID NFC”.

The minimal configuration of a **SpringBlue ID** BLE Reader contains:

- The **SiteID** (§ 5.1). If the **SiteID** is left empty, the Reader uses the default **SiteID** as defined in [PMA17128](#), chapter 8 “test vectors” (h00000001).
- The **SOIK** and the **MSUK** (§ 5.2). If these keys are not configured, the Reader uses the default keys defined in [PMA17128](#), chapter 8 “test vectors”:
 - SOIK=hA0A1A2A3A4A5A6A7A8A9AAABACADAEAF
 - MSUK=hB0B1B2B3B4B5B6B7B8B9BABBBCBDBEBF

5.1. SiteID (LOC REGISTER)

The LOC register contains the **SiteID**.

LOC for SpringBlue ID BLE – Address: h03, size: 4 bytes

Byte	Meaning	Notes / Valid range
Byte 0-4		
0	SiteID	MSB
1		
2		LSB
3		

Default value: h00000001

5.2. SOIK AND MSUK (AUT REGISTER)

The AUT register contains both site’s secret keys. Each key is 16B-long (AES), so the total length of the register is 32B.

AUT for SpringBlue ID BLE – Address: $\text{h}05$, size: 32 bytes

Bits	Value	Meaning
Bytes 0 to 15		
SOIK (Site's ObjectID key) (16 bytes)		
Bytes 16 to 31		
MSUK (Master Key to protect Site's UserIDs) (16 bytes)		

Default value:

$\text{h}A0A1A2A3A4A5A6A7A8A9AAABACADAEAFB0B1B2B3B4B5B6B7B8B9BABBBCBDBEBF$

5.3. SPRINGBLUE ID BLE SIZE AND FORMAT OF OUTPUT (TOF REGISTER)

The **UserID** that has been read may be transmitted “as is” (RAW mode, hexadecimal output on ASCII serial line), or may be translated into Decimal (BCD). The TOF register controls the format and the size of the output.

TOF for SpringBlue ID BLE – Address: $\text{h}01$, size: 1 byte

Bits	Value	Meaning
RFU bits		
7	$\text{b}0$	RFU → must be $\text{b}0$
6	$\text{b}0$	RFU → must be $\text{b}0$
5	$\text{b}0$	RFU → must be $\text{b}0$
4	$\text{b}0$	RFU → must be $\text{b}0$
Size and format of output		
3-0	$\text{b}0001$	RAW (hexadecimal) mode
	$\text{b}0002$	Fixed length, 4 bytes
	$\text{b}0011$	Fixed length, 8 bytes
	$\text{b}0100$	Fixed length, 5 bytes
	$\text{b}0101$	Fixed length, 10 bytes
	$\text{b}0110$	Fixed length, 7 bytes
	$\text{b}1000$	Fixed length, 11 bytes
	$\text{b}1001$	Fixed length, 16 bytes
	$\text{b}1010$	Fixed length, 20 bytes
	$\text{b}1011$	Fixed length, 24 bytes
	$\text{b}1111$	Fixed length, 32 bytes
		Variable length (the UserID sent by the application is transmitted “as is”)
	$\text{b}0000$	Decimal mode
	$\text{b}1100$	10 digits (truncation to 4 bytes, fixed-length BCD output)
	$\text{b}1101$	12 digits (truncation to 5 bytes, fixed-length BCD output)
	$\text{b}1110$	13 digits (truncation to 5 bytes, fixed-length BCD output)
		Variable length (no truncation, BCD output)

Default value: $\text{h}0F$ ($\text{b}00001111$): RAW mode, variable length (i.e. send the 8 bytes of the **UserID**)

5.4. **SPRINGBLUE ID BLE PREFIX (PFX REGISTER)**

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

This is typically used to discriminate among templates, in order to be able to know where the data comes from. For instance, set PFX to 'B' for **SpringBlue ID BLE**, and to 'N' for **SpringBlue ID NFC** if you want to discriminate among the communication channel inside the upper-level applications.

PFX for SpringBlue ID BLE – Address: $\text{h}02$, 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 $_{h0A}$ 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: $_{h0A}$, size: 2 bytes

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

Default value: $_{h0064}$

6.2. MINIMUM RSSI

You can ignore some devices too far by setting the minimum RSSI value in the register $_{h09}$. 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 $_{hC4}$ is -60), the default value is -50.

Minimum RSSI – Address: $_{h09}$, size: 1 bytes

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

Default value: $_{hCE}$

6.3. ADVERTISEMENT PARAMETERS

The register $_{h0B}$ 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: $\text{h}0\text{B}$, 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: $\text{h}0032006407$

Minimum connection interval.

Value in units of 0.625 ms

- Range: $\text{h}0020$ to $\text{h}4000$ (connectable advertising)
- Time range: 20 ms to 10.24 s (connectable advertising)
- Range: $\text{h}00\text{a}0$ to $\text{h}4000$ (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: $\text{h}0020$ to $\text{h}4000$
- 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 `_h0C` 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: `_h0C`, 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: `_h00060018000100E`

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: `_h0006` to `_h0c80`
- 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 $(1 + \text{latency}) * \text{max_interval} * 2$, 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

All information in this document is either public information or is the intellectual property of SPRINGCARD and/or its suppliers or partners.

You are free to view and print this document for your own use only. Those rights granted to you constitute a license and not a transfer of title : you may not remove this copyright notice nor the proprietary notices contained in this documents, and you are not allowed to publish or reproduce this document, either on the web or by any mean, without written permission of SPRINGCARD.

Copyright © SPRINGCARD SAS 2017, all rights reserved.

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