PMA3046-BA

DRAFT - CONFIDENTIAL

SpringCard SpringProx Readers

Hidden registers and advanced configuration options

Document Identification

Category	Administration manual		
Family/Customer	SpringProx FW		
Reference	PMA3046	Version	BA
Status	draft	Classification	Confidential
Keywords	H663, CrazyWriter-HSP, CSB-HSP, H512, K663, Prox'N'Drive, CSB4		
Abstract			

File name	V:\Dossiers\SpringCard\A-Notices\Commun\Configuration avancee\[PMA3046-BB] SpringProx Hidden registers and advanced configuration options.odt		
Date saved	22/01/16	Date printed	05/12/12

Revision History

Ver.	Date	Author	Valid. by		Approv. by	Details
			Tech.	Qual.		
AA	06/03/13	JDA				Created
AB	21/05/14	JDA				Updated for CW-HSP and CSB-HSP
AC	12/06/14	JDA				Mode2 removed in 3.7.1
BA	18/08/14	JDA				Now covering all SpringProx products (K663 family added)
BB	02/09/14	JDA				Added timings for CCID mode

Contents

1. Introduction

2. How to access to the configuration registers

2.1.1. Abstract

The non-volatile memory has a limited write/erase endurance.
Writing any configuration register more than 100 times may permanently damage the product.

2.1.2. Using the product's console

2.1.3. Using the product's Legacy interface through SpringProx API

2.1.4. Using the product's PC/SC interface through SCardTransmit

Use the **READER CONTROL** command APDU (FF F0 00 00 …) as follow:

2.1.5. Using the product's PC/SC interface through SCardControl

a. Reading reader's registers

To read the value of the configuration register at <index>, send the sequence:

```
    58 0E <index>
```

Remember that the returned value (if some) is prefixed by the status code ($_h$00 on success, $_h$16 if the value is not defined in the non-volatile memory).

b. Writing reader's registers

To define the value of the configuration register at <index>, send the sequence:

```
    58 0D <index> <…data…>
```

Send an empty <data> (zero-length) to erase the current value. In this case, default value will be used.

The value of the configuration registers is loaded by the SpringProx's firmware upon reset only. To apply the new configuration, you must reset the SpringProx (or cycle power).

Alternatively, you may load temporary configuration settings as explained in the next paragraph.

2.1.6. Pushing a new temporary configuration

To overrule temporarily the value of the configuration register at <index>, send the sequence:

```
    58 8D <index> <…data…>
```

Send an empty <data> (zero-length) to reload the default value.

This value will be applied immediately, but on next reset the SpringProx will reload its configuration registers from the non-volatile memory.

3. Hidden configuration registers

3.1. List of the configuration registers

3.1.1. Publicly-documented configuration registers

Address	Section	Name	Remark
$_h$B0	Contactless	Enabled HF protocols	
$_h$B2	PC/SC	CLA of the APDU interpreter	
$_h$B3	PC/SC	RF behaviour in PC/SC mode	
$_h$B4	Contactless	Parameters for polling	
$_h$B8	Contactless	Enabled LF protocols	Prox'N'Drive HF/LF
$_h$BB	7816	Allowed baudrates for the smartcard slots	
$_h$C3	7816	Options for the smartcard slots	
$_h$C4	Contactless	Allowed baudrates in T=CL	
$_h$C5	Contactless	Options for T=CL	
$_h$C8	Contactless	Number of antennas	
$_h$C9	Contactless	Options for polling	
$_h$CA	Core	Configuration of the LEDs	
$_h$CB	Core	Options for the LEDs and GPIOs	
$_h$CC	Core	Behaviour of the LEDs and buzzer	
$_h$CF	Felica	Service Codes for Felica read/write	
$_h$E1	NFC P2P	Global Bytes bytes in ATR_REQ	

All those registers are documented in doc. PMD2271.

3.1.2. Hidden configuration registers

Address	Section	Name	See §
$_h$B1	PC/SC	Slot mapping and naming	3.3.1
$_h$B5	PC/SC	Timings for CCID state machine	
$_h$B6	Contactless	RC663 bug (?) workaround	3.3.3
$_h$BA		Card presence edge / fool hardware detect	3.7.1
$_h$BD		CCID debug level	3.3.4
$_h$C0	Global	Firmware operation mode	3.3.5
$_h$C1	Contactless	RF levels	
$_h$C2	7816	Tweak timeouts	
$_h$C6	Contactless	RF overshoot compensation	
$_h$C7		LPCD settings	3.6.1
$_h$CD		Global debug level	3.3.6
$_h$CE		Field shutdown	3.3.7
$_h$F0		Lock configuration registers	3.2.1
$_h$F1		Serial number	3.2.2
$_h$F2		Product name	3.2.3
$_h$FA		ABC7816 licence and smartcard CCID options	3.7.2

3.2. Product UID, name, and protection against changes

3.2.1. Lock configuration registers

This register prevent overwriting the value of some specific configuration register.

Address: $_h$F0 – Size: 1 byte

	Bit	Action if set	Note
msb	7	Lock registers $_h$F1 to $_h$FE	
	6	Lock register $_h$F0 itself	
	5	Lock register $_h$C0	
	4	RFU	
	3	RFU	
	2	RFU	
	1	RFU	
lsb	0	RFU	

Default value: $_h$00 (all registers are writeable)

3.2.2. Serial number

This register defines the serial number announced by the product (in its USB descriptor, among others).

Address: $_h$F1 – Size: 4 bytes

Default value: empty (no serial number)

3.2.3. Product name

This register defines the name announced by the product (in its USB descriptor, among others).

Address: $_h$F2 – Size: variable

Default value: empty (use default product name defined in the firmware: “H512” or “H663”).

3.3. Common to most products

3.3.1. Slot mapping and naming, slot behaviour on startup

Warning: this register has changed between version 1.xx and 2.xx
The first byte is present in public documentations, the second byte is documented only here

Address: $_{h}$B1 – Size: 1 byte

	Bit	Action if set	Note
Byte 0			
msb	7	Force a letter in the name of the SAM slots	Even if there's only one slot, it will be named “SAM A”
	6	Force a letter in the name of the ID-1 slots	Even if there's only one slot, it will be named “Contact A”
	5	RFU	
	4	Prefix the slot name using the product's serial number (in hex)	This is useful for computers with numerous products attached
	3	Start with SAM slot(s) OFF	All the SAM slot(s) will not run until resumed by a Control command
	2	Start with Contact slot OFF	The Contact slot will not run until resumed by a Control command
	1	Start with Contactless slot OFF	The Contactless slot will not run until resumed by a Control command
msb	0	No contactless slot	The Contactless slot will not be enumerated (and will never run)
Byte 1			
msb	7	RFU	
	6	RFU	
	5	RFU	
	4	RFU	
	3	RFU	
lsb	2 - 0	Slot enumeration order: $_{b}$000 : Contactless → Contact → SAM(s) $_{b}$001 : Contactless → SAM(s) → Contact $_{b}$010 : Contact → Contactless → SAM(s) $_{b}$011 : Contact → SAM(s) → Contactless $_{b}$100 : SAM(s) → Contactless → Contact $_{b}$101 : SAM(s) → Contact → Contactless Other values are RFU	

Default value: $_{h}$00 00

3.3.2. Timings for the CCID state machine

Warning: this register's default values have changed between version 1.xx and 2.xx

Address: $_h$B5 – Size: 8 bytes

Byte	Role	Default
0	CCID startup delay (s)	2
1	Timeout on the USB link (s)	2
2	Min. delay between 2 USB interrupts (ms)	150
3	Min. delay between an interrupt and a bulk-out (ms)	30
4	Delay the card tracking after power ON (ms)	25
5	Delay the card tracking after power OFF (ms)	30
6	Interval between ISO 7816 GetStatus (ms)	25
7	Debounce start/stop commands (ms)	150

3.3.3. RC663 bug (?) workaround

3.3.4. CCID debug level

Address: $_h$BD – Size: 1 byte

Bit	Action if set	Note
msb 7	Trace specially the smartcard slots power sequences	
6	Trace *PC_To_RDR_Escape (SCardControl)*	
5	Trace the embedded APDU interpreter	
4	Trace *PC_To_RDR_Transmit (SCardTransmit)*	
3	Trace the dialog with the PC (Bulk In/Bulk Out frames)	
2	Trace the function calls	
1	Trace the status of the CCID slots	
lsb 0	Trace errors	

Default value: $_h$00 (CCID debug disabled)

3.3.5. Firmware operating mode

This register defines how the product's firmware will be seen by the computer. It can be either PC/SC or Legacy.

Address: $_h$C0 – Size: 1 byte

Value	Operating mode
$_h$01	Legacy mode
$_h$02	PC/SC mode
$_h$03	HID (keyboard) mode
$_h$81	Legacy mode without serial number in USB descriptor
$_h$82	PC/SC mode without serial number in USB descriptor
$_h$83	HID (keyboard) mode without serial number in USB descriptor

Default value: $_h$02 (PC/SC with serial number)

All other values are RFU and shall not be used.

3.3.6. Global debug level

Address: $_h$CD – Size: 2 bytes

Bit	Action if set	Note
msb 15	RFU	
14	RFU	
13	RFU	
12	RFU	
11	RFU	
10	RFU	
9	RFU	
8	RFU	
7	RFU	
6	RFU	
5	RFU	
4	RFU	
3	RFU	
2	RFU	
1	RFU	
lsb 0	RFU	

Default value: $_h$0000 (Global debug disabled)

3.3.7. Field shutdown

Address: $_h$CE – Size: 1 byte

	Bit	Action if set	Note
msb	7	RFU	
	6	RFU	
	5	RFU	
	4	RFU	
	3	RFU	
	2	RFU	
	1	RFU	
lsb	0	RF field is switched OFF	

Default value: $_h$00 (RF field is active)

3.4. Common to all products based on the AVR32 core (H512 & H663)

3.5. Specific to H512

3.6. Specific to H663

3.6.1. LPCD settings

This register defines the parameters used by the **H663** when LPCD is active.

Address: $_h$C7 – Size: 5 bytes

Byte	Data	Default value	Remark
0	Threshold	$_h$00	
1 - 2	Interval (ms)	$_h$0000	
3 - 4	Probe time (ns)	$_h$0000	

3.7. Specific to ABC7816 library

3.7.1. Card presence edge / fool hardware detect

This register defines how the **H663** detects and handles the smartcard slots.

Address: ${}_{h}$BA – Size: 2 bytes

	Bit	Action if set	Note
Logic of the card presence switches			
msb	15	Stop contactless slot when there's a contact card in slot 0	
	14	RFU	
	13	RFU	
	12	Invert logic for slot 4 card pres. switch	
	11	Invert logic for slot 3 card pres. switch	
	10	Invert logic for slot 2 card pres. switch	
	9	Invert logic for slot 1 card pres. switch	
	8	Invert logic for slot 0 card pres. switch	
Fool hardware detect (multi-slot mode only)			
	7	RFU	
	6	RFU	
	5	Invert logic to detect slot 1	
	4	RFU	
	3	RFU	
	2	RFU	
	1	Invert logic to detect slots 2-3-4	
lsb	0	Invert logic to detect slot 0	

Default value: ${}_{h}$0000 (Trust the hardware)

3.7.2. ABC7816 licence and smartcard CCID options

This register activates and configure the ISO 7816 library provided by out partner ABC Smartcard.

Address: $_h$FA – Size: 1 byte

	Bit	Action if set	Note
msb	7	RFU	
	6	RFU	
	5	RFU	
	4	RFU	
	3	RFU	
	2	RFU	
lsb	0-1	Support level: 00 : ABC7816 disabled 01 : ABC7816 enabled, single slot mode only 02 : *RFU* 03 : ABC7816 enabled, multi-slot mode	

Default value: $_h$00 (ABC7816 disabled)

4. Exploring the product's hardware

4.1.1. Abstract

4.1.2. Using the product's console

4.1.3. Using the product's Legacy interface through SpringProx API

4.1.4. Using the product's PC/SC interface through SCardTransmit

Use the **READER CONTROL** command APDU (FF F0 00 00 …) as follow:

4.1.5. Using the product's PC/SC interface through ScardControl

5. Access to the product's I2C bus

5.1.1. Abstract

5.1.2. Using the product's console

5.1.3. Using the product's Legacy interface through SpringProx API

To read a value through i2C, you will need to send the command :

Sprox Control	Read I2C	Adress	Byte number to read
0x58	0x12	@	N

To write a value through i2C, you will need to send the command :

Sprox Control	Write I2C	Adress	Table to send
0x58	0x13	@	&tab

5.1.4. Using the product's PC/SC interface through SCardTransmit

Use the **READER CONTROL** command APDU (FF F0 00 00 …) as follow:

5.1.5. Using the product's PC/SC interface through ScardControl

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 PRO ACTIVE 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 PRO ACTIVE 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. PRO ACTIVE reserves the right to change the information at any time without notice.

PRO ACTIVE doesn't warrant any results derived from the use of the products described in this document. PRO ACTIVE 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. PRO ACTIVE customers using or selling these products for use in such applications do so on their own risk and agree to fully indemnify PRO ACTIVE for any damages resulting from such improper use or sale.

Copyright notice

Editor’s information

PRO ACTIVE SAS company with a capital of 227 000 €

RCS EVRY B 429 665 482

Parc Gutenberg, 13 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