RFID B1 Module User Manual

V2.3
21/05/2018

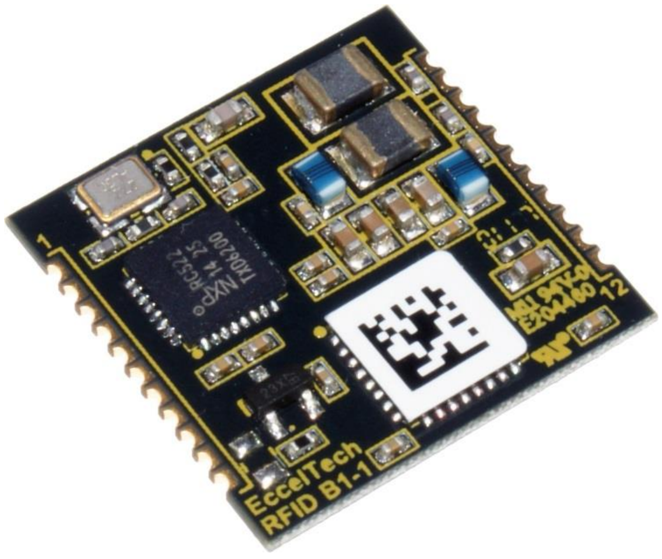
Table of Contents

1 Introduction

1.1 Device Overview

Features

- Low cost RFID Reader with MIFARE Classic®, MIFARE Ultralight® and NTAG2® support
- Polling functionality (Stand alone mode)
- Command interface via UART with optional AES-128 encryption
- Multiple Tag support
- UART baud rate up to 921600 bps
- Compact form factor
- Castellated SMT pads for easy and reliable PCB mounting
- High transponder read and write speed
- Low power design
- Single operating voltage: 2.5V to 3.6V
- -25°C to 85°C operating range
- 4 configurable GPIOs with interrupts
- 3 configurable PWMs
- Comparator
- ADC
- Current Output DAC
- AES-128 encryption engine
- Multiple internal reference voltages
- RoHS compliant



Description

The RFID B1 module is the second in an evolving family of 13.56MHz sub assemblies from Eccel Technology Ltd (IB Technology). The product is designed with both embedded applications and computing / PLC platforms in mind. This product is an ideal design choice if the user wishes to add RFID capability to their design quickly and without requiring extensive RFID and embedded software expertise and time. An on board low power ARM microcontroller handles the RFID configuration setup and provides the user with a powerful yet simple command interface to facilitate fast and easy read/write access to the memory and features of the various transponders supported by this module.

A polling option with configurable IOs, PWM and Packet behavior that is dependent upon the UID of the tag detected, makes it simple to build a stand-alone device that can control door locks, etc. and additionally send the detected tag UIDs to a host that is connected to the module serial port (UART).

The module simply requires a single power and GND connection from the user PCB, along with two connections to an antenna.

Applications

- Access control
- Monitoring goods
- Approval and monitoring consumables
- Pre-payment systems
- Managing resources
- Connection-less data storage systems

1.2 Pinout

Pin Number	Symbol	Type	Description
1	GND	Ground	
2	nRESET	Digital input with pull-up	Reset input signal (active low). This pin requires no external pull-up / down resistor unless the module is used in noisy environments, in which case connection of an external pull-up resistor combined with HF filter is recommended.
3	nSLEEP	Digital push-pull output	Output signal indicating the device is in Sleep Mode or Power Down Mode (active low).
4	nPWRDN	Digital input with no pull resistors	Power Down Request input signal (active low). This pin has no pull-up/down resistor and should NOT be left floating. For power optimization it is recommended that the user drives this pin with a push-pull GPIO or similar.
5	IO3	General purpose digital input-output	General purpose digital input-output pin 3.
6	IO2	General purpose digital input-output	General purpose digital input-output pin 2 / PWM Output 2.
7	IO1	General purpose digital input-output	General purpose digital input-output pin 1 / PWM Output 1 / Comparator Output.
8	IO0	General purpose digital input-output	General purpose digital input-output pin 0 / PWM Output 0.
9	UART TX	Digital push-pull output	UART transmitter signal line.
10	UART RX	Digital input with no pull resistors	UART receiver signal line. This pin has no pull-up/down resistor. It should not be left floating and it is recommended that the user connect this pin to the UART TX pin of their host controller (GND-Vdd voltage range only).
11	GND	Ground	
12	GND	Ground	
13	ADC IN	Analog input	Analog to Digital Converter input.
14	CMP IN	Analog input	Comparator positive input.
15	IDAC OUT	Analog output	Digital to Analog Converter with current-type output.
16	NOT USED	-	Leave floating.
17	TPI	Digital push-pull output	Tag Presence Indicator is set to low state if Tag presence is detected. The state is updated with every requested RFID command that enforces communication with the Tag
18	V_{DD}	Power supply pin	Power Supply pin. Low ESR capacitor with capacitance 10uF or higher should be connected close to this pin.
19	GND	Ground	
20	ANT2	Analog output	Antenna output.
21	ANT1	Analog output	Antenna output.
22	GND	Ground	

Table 1.2

1.3 Application

The RFID B1 module is specifically designed for embedded applications, where low pin count connection to a host microcontroller together with ease of implementation are the most important factors. The UART interface provides a command protocol which is flexible and easy to use. For the module, itself to be a fully functional RFID reader/writer it only requires connection to a power supply and an antenna. Eccel Technology Ltd manufactures a wide range of such antennas designed to give optimal performance with this module. Please contact us for further details of our range of antennas.

1.4 Typical application schematic

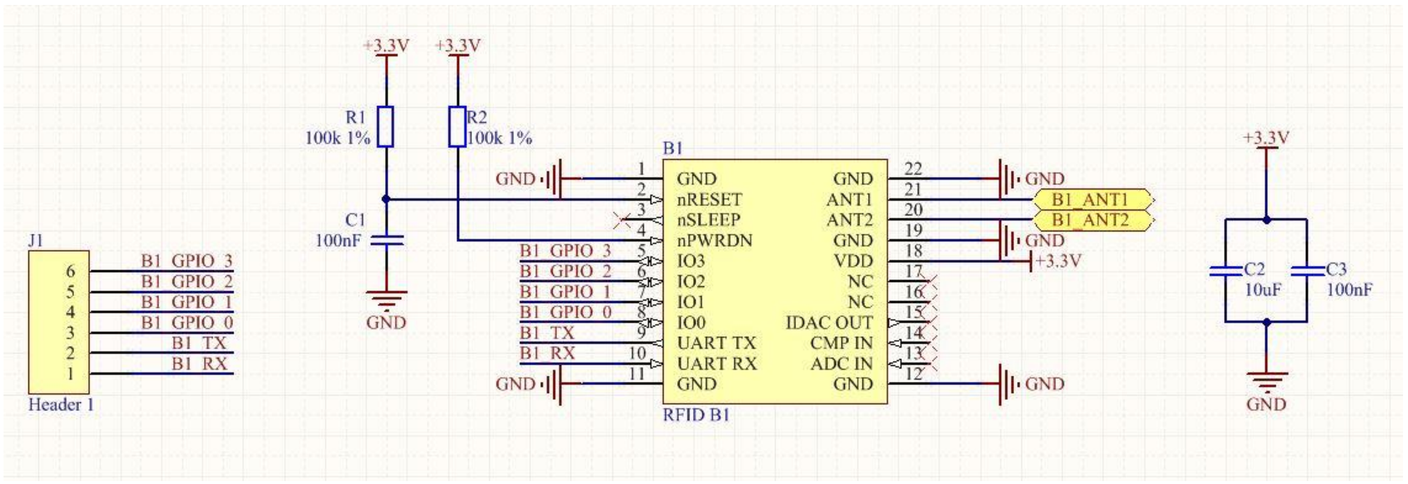


Figure 1.4-1

2 Electrical Characteristics

2.1 Test Conditions

Typical device parameters have been measured at ambient temperature 22°C ±3°C and using a power supply of 3.3V ±5%.

2.2 Absolute Maximum Ratings

Symbol	Parameter	Min	Max	Unit	Notes
T_S	Storage Temperature	-40	150	°C	Tested for 10'000 hours at 150°C.
V_{DDMAX}	Supply Voltage	0	3.8	V	
V_{IOMAX}	Input Pin Voltage	-0.3	V_{DD} + 0.3	V	
I_{IOMAX}	Output Pin Current	0	6	mA	
I_{ANT}	ANT1 and ANT2 Current	0	100	mA	Maximum continuous current. This depends upon the impedance of the circuit between ANT1 and ANT2 at 13.56MHz.

Table 2.2

2.3 Operating Conditions

Symbol	Parameter	Min	Max	Unit
T_O	Ambient Temperature	-25	85	°C
V_{DD}	Supply Voltage	2.5	3.6	V

Table 2.3

2.4 Current Consumption

Symbol	Parameter	Typ	Max	Unit	Comment
I_{IDDLE}	Iddle State Current	1.78	1.8	mA	T = 25°C, Vdd = 3.3V. ADC, DAC, ACMP and PWM turned off.
			1.85	mA	Full range of temperature and power supply voltage.
I_{SLEEP}	Sleep Mode Current	0.6	0.9	µA	T = 25°C, Vdd = 3.3V.
			1.8	µA	Full range of temperature and power supply voltage.
I_{PWRDN}	Power Down Current	20	40	nA	T = 25°C, Vdd = 3.3V.
			400	nA	Full range of temperature and power supply voltage.
I_{TX}	RFID Power Up Current	14	24	mA	T = 25°C, Vdd = 3.3V, 50Ω antenna connected between ANT1 and ANT2.
I_{MAX}	Module Maximum Current		120	mA	Maximum current consumed by the module in the worst conditions.
I_{PWMCH}	Single Channel PWM Current	180	200	uA	T = 25°C, Vdd = 3.3V. No load on output. Frequency 200kHz.
I_{CMP}	Comparator Current	46	50	uA	T = 25°C, Vdd = 3.3V. No load on output. Iddle State.
		150	400	nA	No load on output. Sleep Mode State.

Table 2.4

2.5 GPIO

Symbol	Parameter	Min	Typ	Max	Unit	Notes
V_{IOIL}	Input Low Voltage			$0.3V_{DD}$	V	
V_{IOIH}	Input High Voltage	$0.7V_{DD}$			V	
I_{IOMAX}	Output Pin Current			± 6	mA	
I_{IOLEAK}	Input Leakage Current		± 0.1	± 40	nA	High impedance IO connected to V_{DD} or GND.
R_{IOESD}	Internal ESD Series Resistor		200		Ω	
V_{IOHYST}	**IO Pin Histeresis**	**$0.1V_{DD}$**			**V**	

Table 2.5

2.6 Antenna Output

Symbol	Parameter	Min	Typ	Max	Unit	Notes
f_{ANT}	Antenna Signal Frequency		13.56		MHz	±30 ppm (-20°C - 70°C).
f_{ANTAG}	Antenna Signal Frequency Aging	0		3	ppm	At 25°C.
V_{ANTH}	Antenna High Level Output Voltage	V_{DD} - 0.64			V	V_{DD} = 2.5V, I_{ANT} = 80mA.
V_{ANTL}	Antenna Low Level Output Voltage			0.64	V	V_{DD} = 2.5V, I_{ANT} = 80mA.
I_{ANT}	ANT1 and ANT2 Current	0	60	100	mA	Maximum continuous current. This depends upon the impedance of the circuit between ANT1 and ANT2 at 13.56MHz.

Table 2.6

2.7 Flash

Symbol	Parameter	Min	Typ	Max	Unit	Notes
C_{FE}	Flash Erase Cycles Before Failure	20000			cycles	
T_{FDR}	Flash Data Retention Time	10			years	For ambient temperature < 85°C
		20			years	For ambient temperature < 70°C

Table 2.7

2.8 IDAC

2.8.1 Parameters

IDAC Parameters									
Precision				Source			Sink		
Range No	Range [µA]	Step Size [nA]	Nominal Current [µA]	Current drop at Vdd - 100 mV [%]	Temperature coefficient [nA/°C]	Voltage coefficient [nA/V]	Current drop at 200 mV [%]	Temperature coefficient [nA/°C]	Voltage coefficient [nA/V]
0	< 0 ; 1.6 >	50	0.85	0.79	0.3	11.7	0.3	0.2	12.5
1	(1.6 ; 4.7 >	100	3.2	0.75	0.7	38.4	0.32	0.7	40.9
2	(4.7 ; 16 >	500	8.5	1.22	2.8	96.6	0.62	2.8	94.4
3	(16 ; 64 >	2000	34	3.54	10.9	159.5	1.75	10.9	148.6

Table 2.8.1

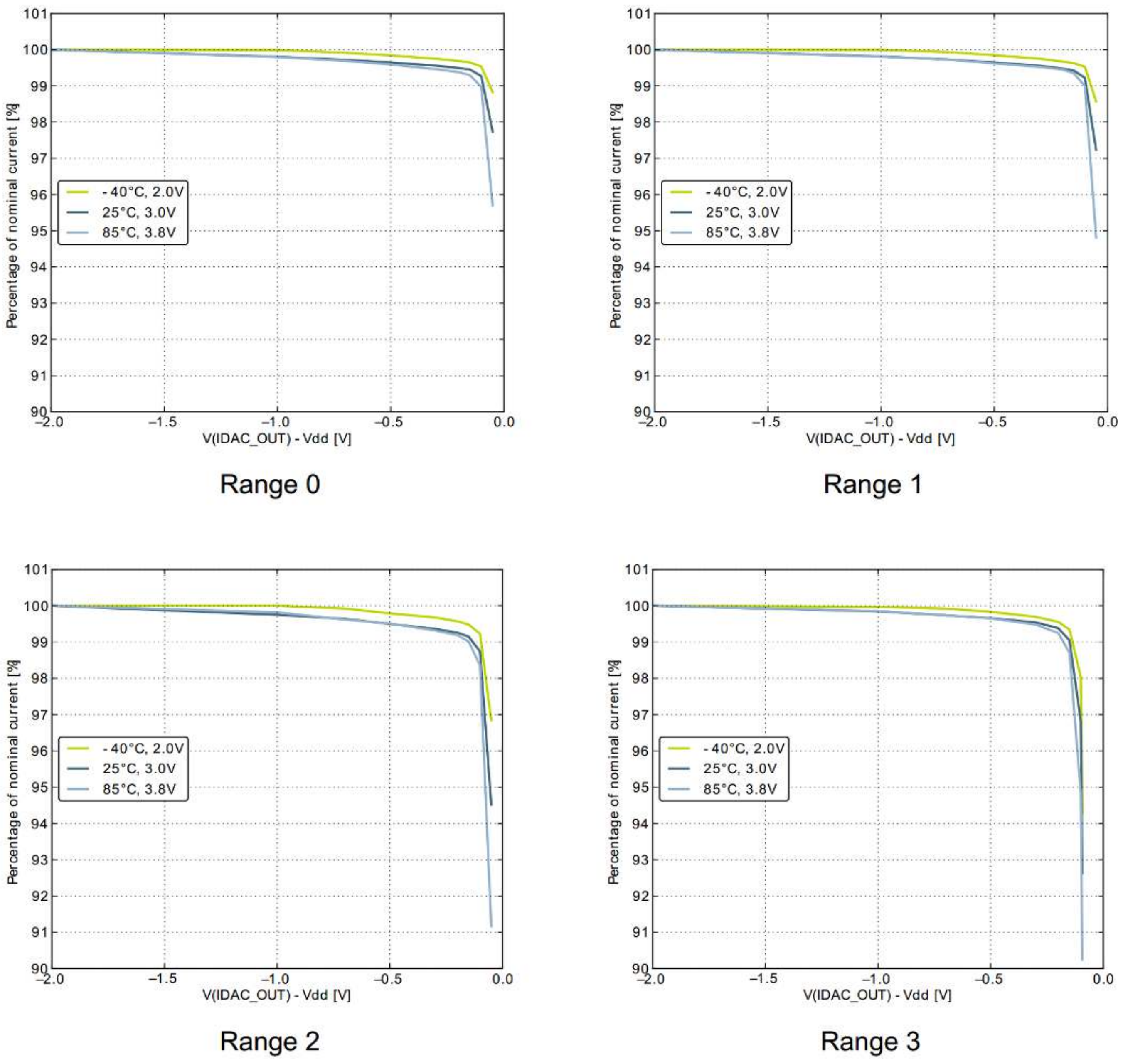


Figure 2.8.1.1 Source Current

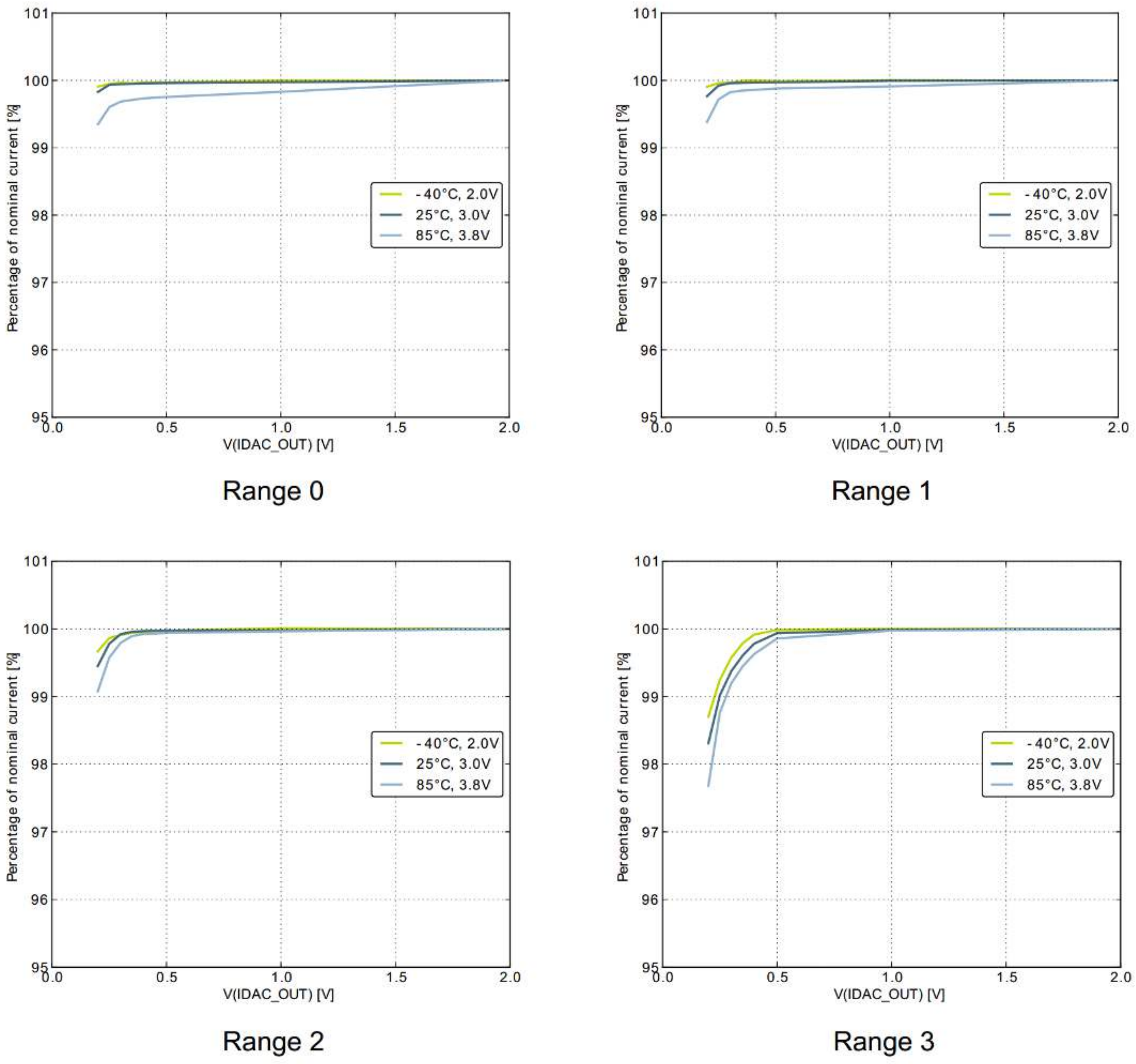


Figure 2.8.1.2 Sink Current

2.8.2 Example Measurement (Error and Offset)

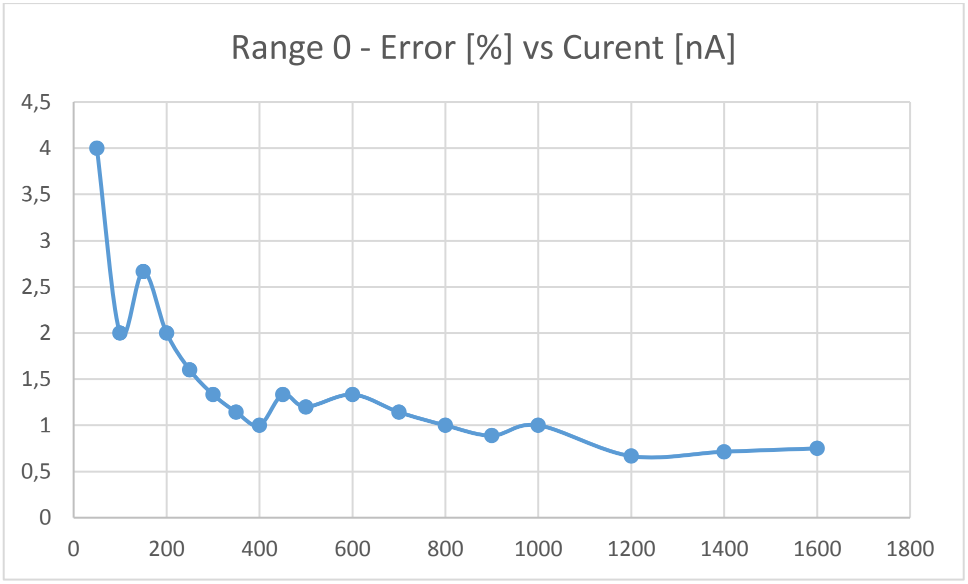


Figure 2.8.1.3 50kΩ Sourcing

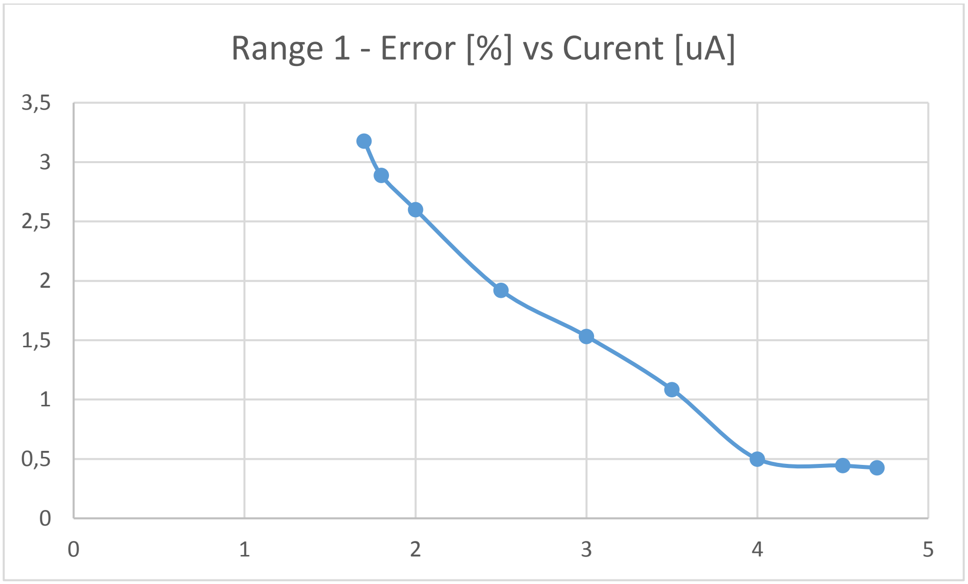


Figure 2.8.1.4 50kΩ Sourcing

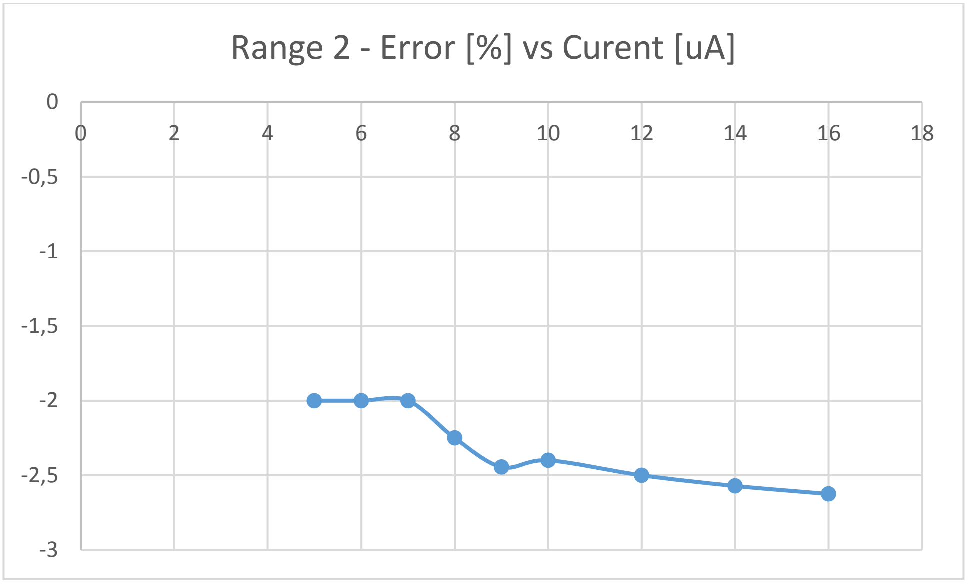


Figure 2.8.1.5 50kΩ Sourcing

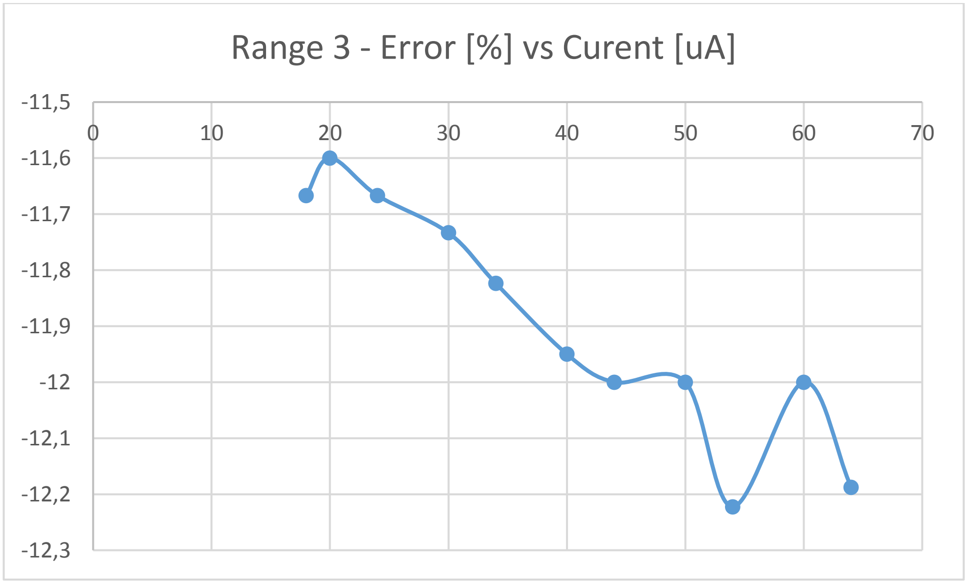


Figure 2.8.1.6 50kΩ Sourcing

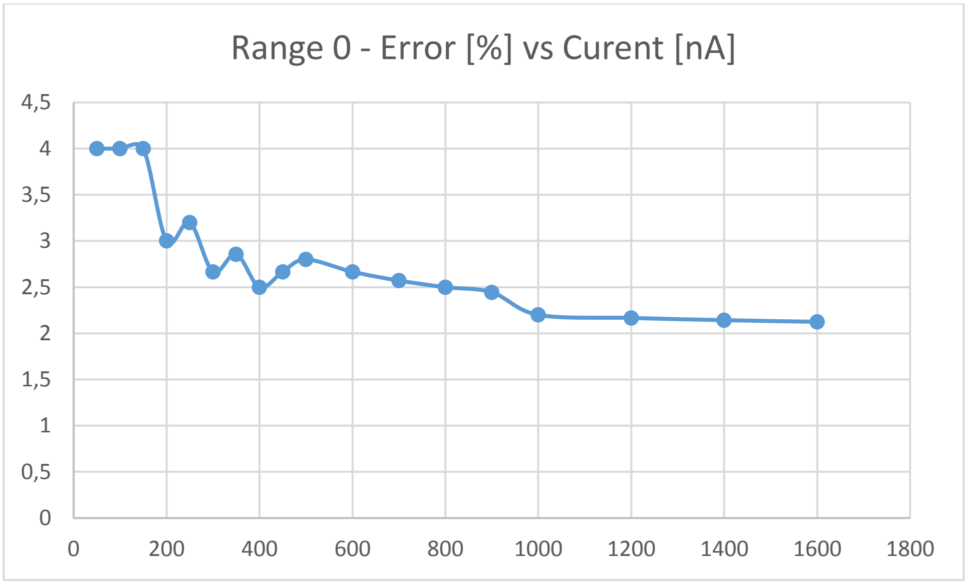


Figure 2.8.1.7 50kΩ Sinking

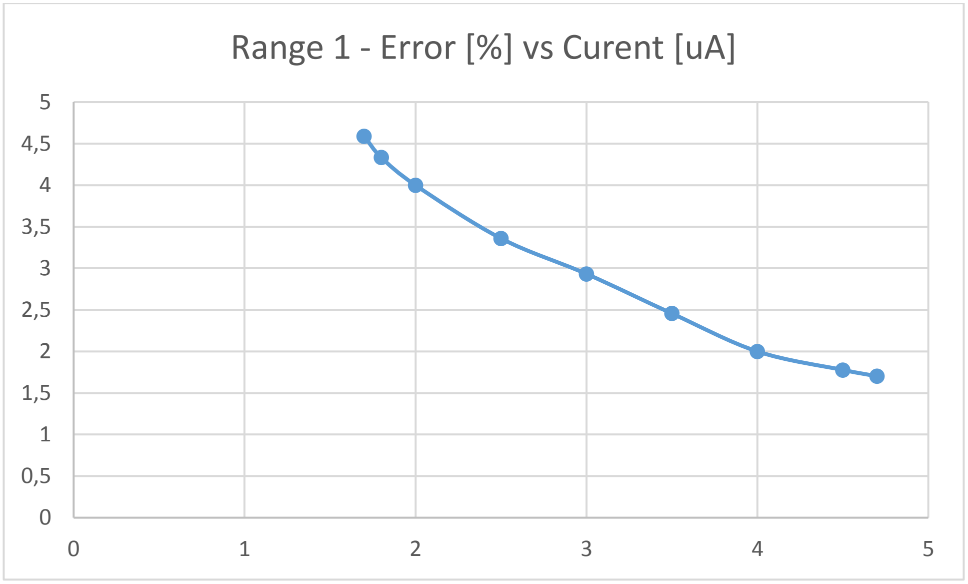


Figure 2.8-1.1.8 50kΩ Sinking

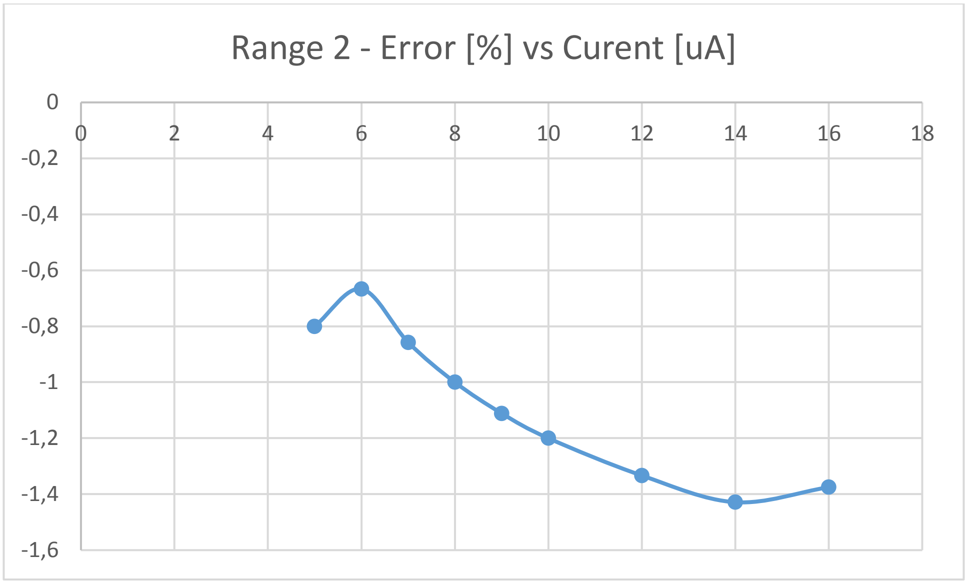


Figure 2.8.1.2.8-2 50kΩ Sinking

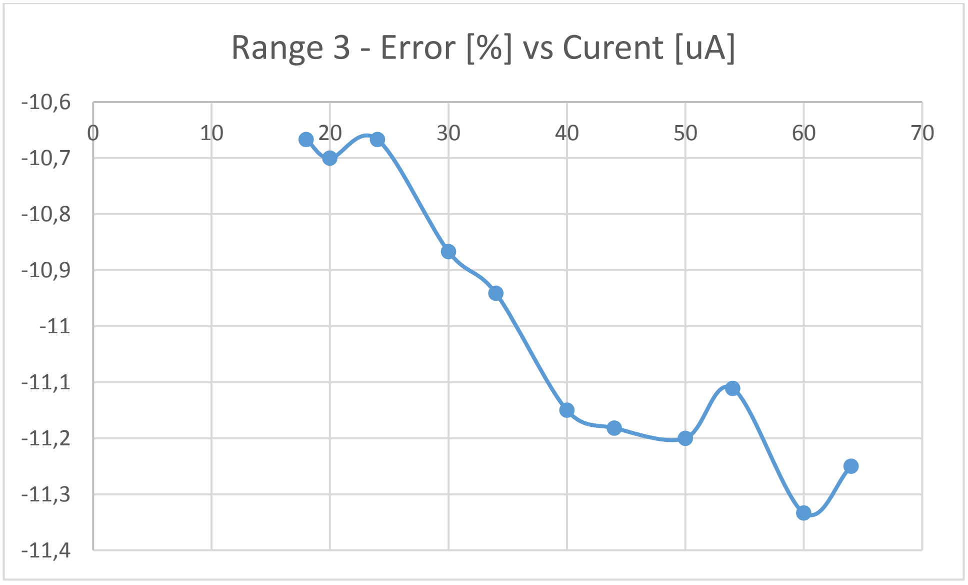


Figure 2.8.1.2.8-3 50kΩ Sinking

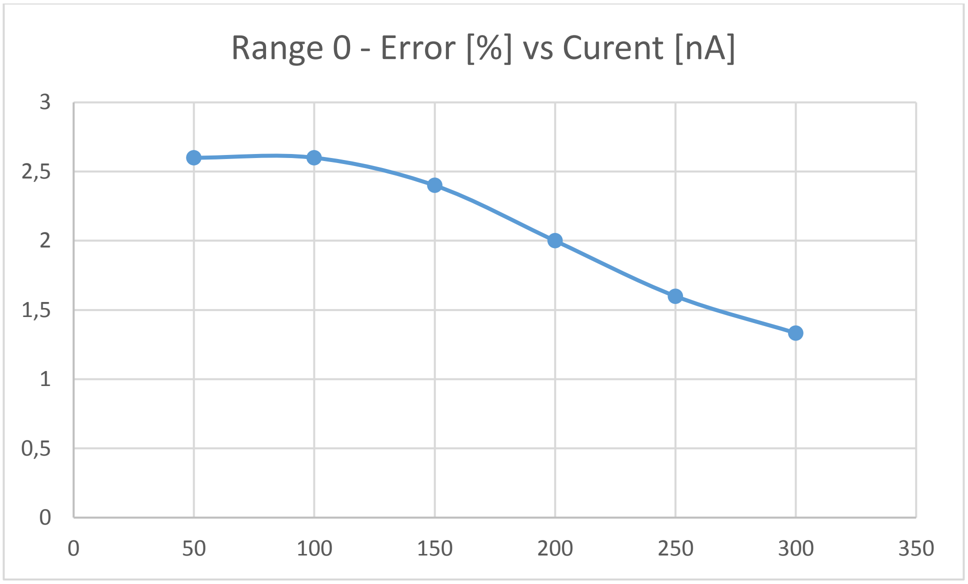


Figure 2.8.1.2.8-4 10MΩ Sourcing

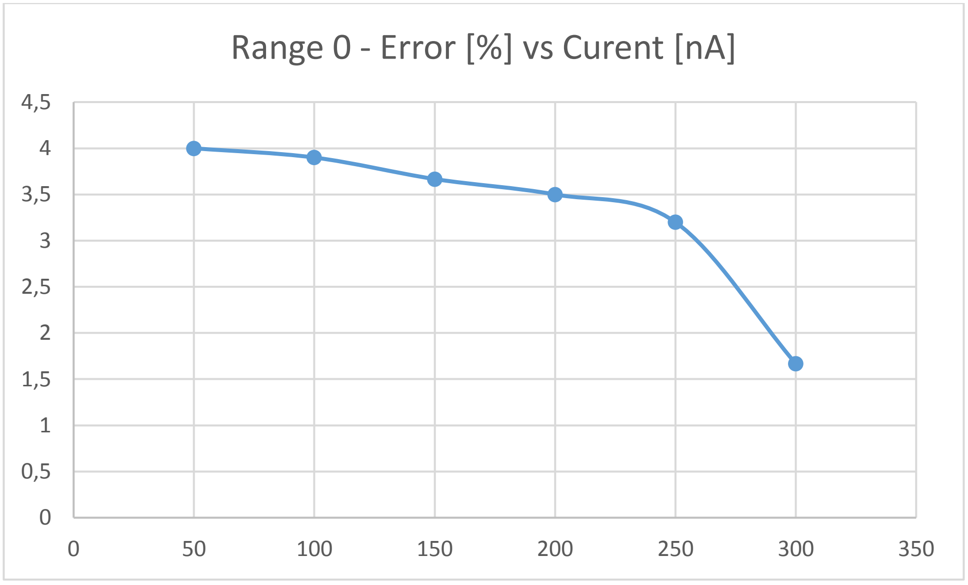


Figure 2.8.1.2.8-5 10MΩ Sinking

2.9 PWM

PWM Parameters				
Period		Frequency		
Minimum [µs]	Maximum [s]	Minimum [Hz]	Maximum [kHz]	Maximum Error [%]
4.81	3.19	0.313	207.9	3

Table 2.9

2.10 ADC

Symbol	Parameter	Min	Typ	Max	Unit	Notes
V_{ADCIN}	Input Voltage Range	0		2.5	V	Internal 2.5V reference voltage used.
I_{ADCIN}	Input Current			100	nA	
C_{ADCIN}	Input Capacitance			2	pF	
R_{ADCIN}	Input On Resistance	1			MΩ	

Table 2.10

2.11 Comparator

Symbol	Parameter	Min	Typ	Max	Unit	Notes
V_{CMPIN}	Input Voltage Range	0		Vdd	V	
$V_{CMPOFST}$	Offset Voltage	-12	0	12	mV	
$V_{CMPHYST}$	Hysteresis		50		mV	

Table 2.11

3 System

3.1 Overview

The general overview of system components is shown in Figure 3.1-1. The system internally consists of four main parts:

- CORE – the main processing part of the microcontroller firmware responsible for managing all system tasks, parsing and the execution of commands received from the user's master controller.
- RFID – dedicated RFID IC together with its firmware drivers responsible for communication with the RFID tag.
- POWER MANAGER – a subsystem responsible for managing power states and clocks in the module to minimise power consumption during operation.
- UART – communication module providing a command protocol over the UART interface.
- TEMPERATURE SENSOR – on-chip analog sensor measuring the microcontroller's die temperature.
- ADC + MUX – Analog to Digital Converter with a Multiplexer used to measure voltage on one of three sources: temperature sensor, power supply voltage divider and external input.
- Vdd DIVIDER – digitally controlled potentiometer used to divide the voltage in 1/63 steps.
- IDAC – Digital to Analog Converter with current-type output.
- COMP – configurable low power comparator with output available on one of the IOs.
- VOLTAGE REFERENCE – on-chip 1.25V and 2.5V precise voltage references.
- PWM GENERATOR – module providing configurable generation of a PWM signal with configurable duty cycle.
- IO SYSTEM – module managing the General-Purpose Input-Output pins of the module.

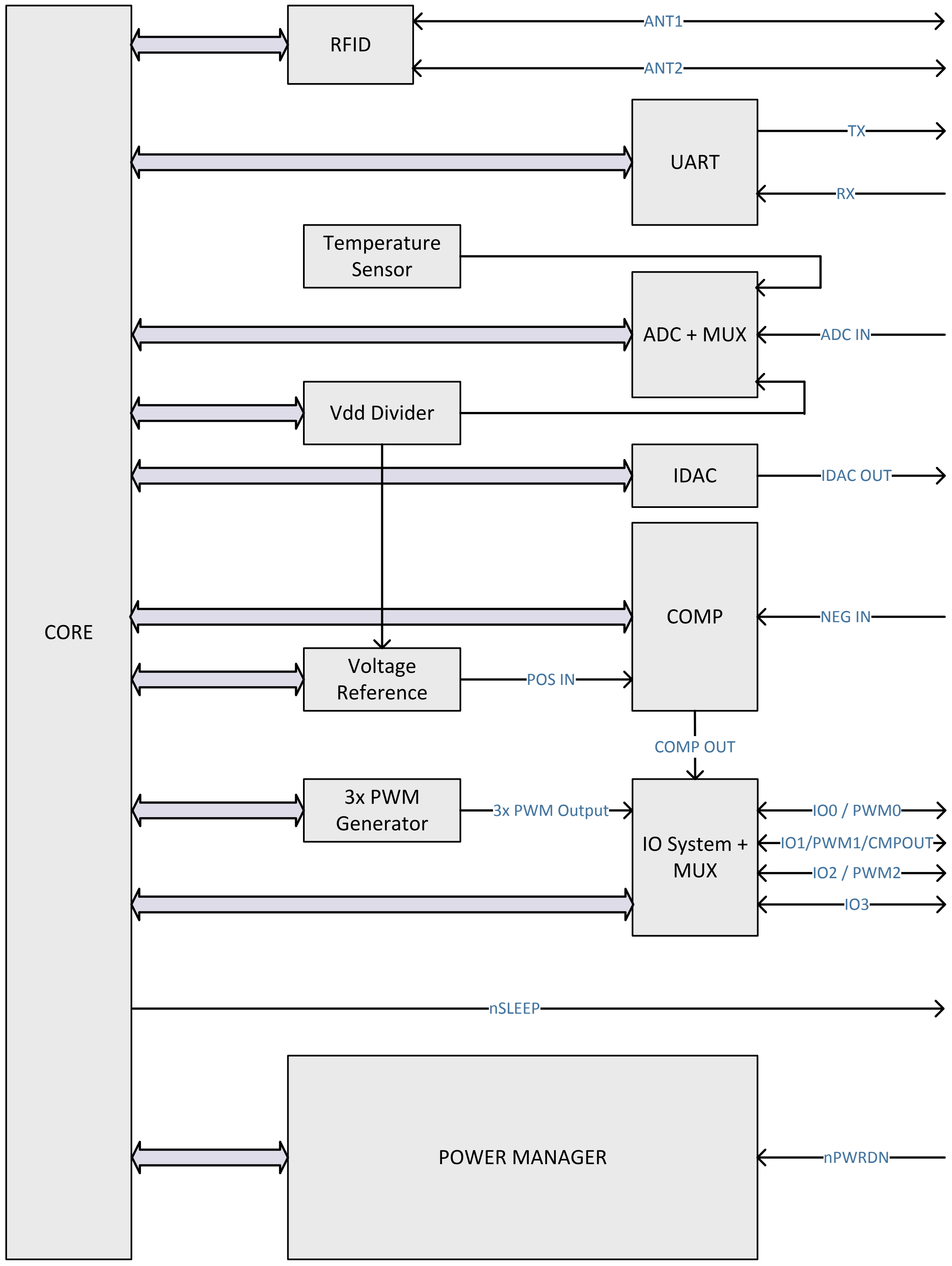

RFID
ANT1
ANT2
UART
TX
RX
Temperature Sensor
ADC + MUX
ADC IN
Vdd Divider
IDAC
IDAC OUT
CORE
COMP
NEG IN
Voltage Reference
POS IN
COMP OUT
3x PWM Generator
3x PWM Output
IO System + MUX
IO0 / PWM0
IO1/PWM1/CMPOUT
IO2 / PWM2
IO3
nSLEEP
POWER MANAGER
nPWRDN

Figure 3.1-1

3.2 Modules

3.2.1 Core

The core of the system shown above in Figure 3.1-1, is the part of the modules' hardware and firmware responsible for managing tasks, prioritizing them and allocating resources to other components in the system. When the user sends a command, the core is the part of the system which ensures that the command is executed properly, that the outcome is saved and that a response is sent back to the user's master controller.

3.2.2 RFID

The RFID component is the part of the hardware and module firmware responsible for the complete communication interaction with the transponder via RF. In the RFID B1 module this part currently provides communication capabilities with MIFARE Classic®, MIFARE Ultralight® and NTAG2xx® transponder types. The key features of this RFID section are both fastest speed of transmission and system responsiveness, and optimal low power consumption.

3.2.3 Power Manager

The Power Manager is the part of the firmware, together with hardware support, that manages the power states of the system and tries to minimize power consumption as much as possible. The user has the option to put the module into Power Down Mode by pulling low the nPWRDN pin.

When the system enters Power Down Mode, all clocks and submodules are disabled and no response to user communication commands will occur until the system is re-enabled by the nPWDN being taken high by the user's system.

3.2.4 UART

The UART component of the RFID B1 module is responsible for the entire communication with the user master controller, together with parsing commands for subsequent execution by the module core. The default baud rate is 9600. The user has the option to change this setting using only a single command. Communication via the UART interface is in a command-response format, where the user first sends commands and the module then replies to each command. There are also asynchronous packets sent to the user whenever an enabled interrupt is generated. Possible interrupt sources are IO state changes, comparator output state change and RFID command execution end.

3.2.5 Temperature Sensor

The Temperature Sensor of the RFID B1 is analog. The voltage generated by the sensor is proportional to the temperature of the microcontroller's die. The system can measure this voltage and therefore calculate the temperature of the die when requested.

3.2.6 ADC + MUX

The Analog to Digital converter together with the analog multiplexer in the module can measure the voltage of various sources. The source is determined by the user selected multiplexer configuration. There are three possible input configurations of the multiplexer: Temperature Sensor, External Input (ADC IN) and Power Supply divider. The multiplexer is configured automatically when using the ADC and no special actions from the user are required for this configuration.

3.2.7 Vdd Divider

The Vdd Divider module is based on a digital potentiometer and divides the power supply voltage by a user selected value. This divided value could be used to measure the power supply voltage using the ADC or by supplying a reference voltage to the positive input of the comparator. The Power Supply is divided per following formula:

$$V_{ref} = V_{dd} * \frac{Divider\ Value}{63}.$$

3.2.8 IDAC

The Digital to Analog converter generates the user selected current flow through the IDAC output pin. The IDAC can be configured as a current source or sink.

3.2.9 Comparator

The Comparator is a general purpose and low power and can be used for voltage level detection. In addition to its output pin, the comparator can generate system interrupts and send asynchronous UART packets if there is a change of its output state. When enabled the output pin is a push-pull type IO with no pull-up or pull-down resistors. When disable command is executed the output pin is disabled regardless of the previous settings.

3.2.10 Voltage Reference

The Voltage Reference is used to generate the user selected reference voltage for the positive input of the comparator.

3.2.11 PWM Generator

The PWM Generator if based on a timer and is used to generate PWM signals on the IO pins. These signals are typically for user interaction like flashing LEDs or driving sounder but also not standard use can be implemented.

3.2.12 IO System

The IO System is the front-end of the General-Purpose Input-Output pins and manages the usage and configuration of the IO pins. The IOs can be configured as input, output low, output high or disabled. They don't have any pull-up or pull-down resistors.

3.2.13 TPI – Tag Presence Indicator

The TPI output is automatically set to low state every time that presence of a tag is detected (after executing the commands: GetUIDandType, EnumerateTagsUID, EnumerateTagsUIDandType, Polling) The TPI output is set to the high state if there is no Tag in range or if any error in Tag communication occurs. The state is updated with every requested RFID command that undertakes communication with the Tag.

3.3 Memory Map

The device memory layout is shown below in Table 3.33.

In the RFID B1 module there are 728 bytes of user accessible memory. Each byte of the memory has a defined factory default value and these values can be recovered by using the 'Reset to Factory Defaults' command. The first 288 bytes are volatile memory which also have a default reset state that is the same as the factory default value. The other memory is buffered non-volatile memory and can be modified and stored using 'Unlock' and 'Lock' commands. The RFID B1 module is automatically returned to factory default state if after power up there is no valid configuration stored in non-volatile memory. The first 288 bytes of this user accessible memory are not stored in non-volatile memory and are always reset to factory defaults after a power-up sequence or after exit from the Power Down Mode.

Reading and writing to the registers can be done using the UART Read (0x02) and Write (0x01) commands. The rest of the memory is accessible only when the module is unlocked. The default state of the module after power up is locked.

Address [DEC]	Address [HEX]	Size [bytes]	Description	Access	Locked and Stored in non-volatile memory
0	0x0000	1	Result	Read Only	No
1	0x0001	1	Command	R / W	No
2	0x0002	18	Command Parameters	R / W	No
20	0x0014	10	Tag UID	Read Only	No
30	0x001E	1	Tag Type	Read Only	No
31	0x001F	1	Tag UID Size	Read Only	No
32	0x0020	256	Data Buffer	R / W	No
288	0x0120	8	Password	R / W when unlocked	Yes
296	0x0128	16	AES Initialization Vector 0	R / W when unlocked	Yes
312	0x0138	16	AES Initialization Vector 1	R / W when unlocked	Yes
328	0x0148	16	AES Key 0	R / W when unlocked	Yes
344	0x0158	16	AES Key 1	R / W when unlocked	Yes
360	0x0168	6	Authentication Key / Password 0	R / W when unlocked	Yes
366	0x016E	6	Authentication Key / Password 1	R / W when unlocked	Yes
372	0x0174	6	Authentication Key / Password 2	R / W when unlocked	Yes
378	0x017A	6	Authentication Key / Password 3	R / W when unlocked	Yes
384	0x0180	6	Authentication Key / Password 4	R / W when unlocked	Yes
390	0x0186	6	Authentication Key / Password 5	R / W when unlocked	Yes
396	0x018C	6	Authentication Key / Password 6	R / W when unlocked	Yes
402	0x0192	6	Authentication Key / Password 7	R / W when unlocked	Yes
408	0x0198	6	Authentication Key / Password 8	R / W when unlocked	Yes
414	0x019E	6	Authentication Key / Password 9	R / W when unlocked	Yes
420	0x01A4	6	Authentication Key / Password 10	R / W when unlocked	Yes
426	0x01AA	6	Authentication Key / Password 11	R / W when unlocked	Yes
432	0x01B0	6	Authentication Key / Password 12	R / W when unlocked	Yes
438	0x01B6	6	Authentication Key / Password 13	R / W when unlocked	Yes
444	0x01BC	6	Authentication Key / Password 14	R / W when unlocked	Yes
450	0x01C2	6	Authentication Key / Password 15	R / W when unlocked	Yes
456	0x01C8	6	Authentication Key / Password 16	R / W when unlocked	Yes
462	0x01CE	6	Authentication Key / Password 17	R / W when unlocked	Yes
468	0x01D4	6	Authentication Key / Password 18	R / W when unlocked	Yes
474	0x01DA	6	Authentication Key / Password 19	R / W when unlocked	Yes
480	0x01E0	6	Authentication Key / Password 20	R / W when unlocked	Yes
486	0x01E6	6	Authentication Key / Password 21	R / W when unlocked	Yes
492	0x01EC	6	Authentication Key / Password 22	R / W when unlocked	Yes
498	0x01F2	6	Authentication Key / Password 23	R / W when unlocked	Yes
504	0x01F8	6	Authentication Key / Password 24	R / W when unlocked	Yes
510	0x01FE	6	Authentication Key / Password 25	R / W when unlocked	Yes
516	0x0204	6	Authentication Key / Password 26	R / W when unlocked	Yes
522	0x020A	6	Authentication Key / Password 27	R / W when unlocked	Yes
528	0x0210	6	Authentication Key / Password 28	R / W when unlocked	Yes
534	0x0216	6	Authentication Key / Password 29	R / W when unlocked	Yes
540	0x021C	6	Authentication Key / Password 30	R / W when unlocked	Yes
546	0x0222	6	Authentication Key / Password 31	R / W when unlocked	Yes
552	0x0228	6	Authentication Key / Password 32	R / W when unlocked	Yes
558	0x022E	6	Authentication Key / Password 33	R / W when unlocked	Yes
564	0x0234	6	Authentication Key / Password 34	R / W when unlocked	Yes
570	0x023A	6	Authentication Key / Password 35	R / W when unlocked	Yes
576	0x0240	6	Authentication Key / Password 36	R / W when unlocked	Yes
582	0x0246	6	Authentication Key / Password 37	R / W when unlocked	Yes
588	0x024C	6	Authentication Key / Password 38	R / W when unlocked	Yes
594	0x0252	6	Authentication Key / Password 39	R / W when unlocked	Yes
600	0x0258	128	User Memory	R / W when unlocked	Yes

Table 3.3

3.3.1 Result Register

The Result Register is 1-byte long, located at address 0x0000, with both read only access. Writing to this register has no effect. The register contains the result (error code) of the last executed command. The list of all possible results is shown in Table 3.3.1.

Result Register Values		
Value	**Type**	**Description**
0x00	No Error	Command was executed successfully, and results were stored in the registers.
0x01	Invalid Command	Value written to command register is invalid.
0x02	Invalid Command Parameter	One of the parameters taken by the command is invalid.
0x03	Indexes Out Of Range	Indexes passed as command parameters exceed limit.
0x04	Error When Writing To Non Volatile Memory	There was an internal error during writing to the non-volatile memory.
0x05	System Error	Internal system error. Shall be considered as fatal.
0x06	Tag CRC Error	During communication with the tag a CRC was not correct.
0x07	Tag Collision	Module cannot handle anticollison procedure.
0x08	Tag is not present	There is no tag within range.
0x09	Tag Authentication Error	Authentication failed due to incorrect Authentication Key or Password.
0x0A	Tag Value Block Corrupted	At least one value block is corrupted in the tag memory.
0x0B	Module Overheated	An overheat was detected.
0x0C	Tag Not Supported	There is a tag in the field which is not supported.
0x0D	Tag Communication Error	There was an error during communication with the tag.
0x0E	Invalid Password	The Password used in the Unlock command string was invalid.
0x0F	Already Locked	You are trying to lock a module that is already locked.
0xFF	Module Busy	Your command was ignored because the module is busy. Retry later.

Table 3.3.1

3.3.2 Command Register

The Command Register is 1-byte long, located at address 0x0001, with both read and write access. Writing to this register is recognized by the module as a command execution request. Depending upon the command (value written to the register) the Command Parameters Register is parsed to extract arguments for the command. Whilst commands are executing the Result Register value is set to 0xFF and any write to the RFID Module memory is discarded. When command execution is complete the memory together with the Result Register is updated and an asynchronous packet is sent indicating that the module is ready to receive another command or generally that a write to the RFID Module memory can be performed.

3.3.3 Command Parameters Register

Register Name	Command Parameters																	
Register Address	0x0002																	
Byte Offset	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F	0x10	0x11
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Factory Default Value	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
Read Function	Parameters taken when executing commands. Each command has various number and order of parameters.																	
Write Function																		

Table 3.1.3

The Command Parameters Register is 18-bytes long, located at address 0x0002 to 0x0013, with both read and write access. This is the place from where the system parses the arguments necessary to perform the requested operation when a command is executed. Depending upon the command, this register is parsed and interpreted in different ways. The details of interpretation of the data stored in this register can be found in chapter 5.3. The module never changes the values inside this register except after power-up or exit from Power Down Mode.

3.3.4 Tag UID Register

Register Name	Tag UID									
Register Address	0x0014									
Byte Offset	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09
Access	R	R	R	R	R	R	R	R	R	R
Factory Default Value	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
Read Function	UID[0]	UID[1]	UID[2]	UID[3]	UID[4]	UID[5]	UID[6]	UID[7]	UID[8]	UID[9]

Table 3.3.2

Tag UID Register is 10-bytes long, located at address 0x0014 to 0x001D, with read only access. In most cases the tag UIDs are 4 bytes, 7 bytes or 10 bytes long, thus this register can cover the longest UID but not necessarily all bytes available in the register will be used. Bytes within the register are ordered from the least significant byte to the most significant byte. This register is updated whenever Get UID and Type commands are used. The tag UID Size Register contains information detailing how many bytes of the ten available represent the tag UID.

3.3.5 Tag Type Register

The Tag Type Register is 1-byte long memory space at address 0x001E with read only access. This register contains information about the type of the tag which was last seen in the field. Possible tag types are shown in Table 3.3.3.

Returned value	Tag type
0x00	No Tag
0x01	Incomplete Type
0x02	Ultralight
0x03	Ultralight EV1 80B
0x04	Ultralight EV1 164B
0x05	Classic Mini
0x06	Classic 1K
0x07	Classic 4K
0x08	NTAG203F
0x09	NTAG210
0x0A	NTAG212
0x0B	NTAG213F
0x0C	NTAG216F
0x0D	NTAG213
0x0E	NTAG215
0x0F	NTAG216
0x10	Unknown

Table 3.3.3

3.3.6 Tag UID Size Register

The Tag UID Size Register is 1-byte long, located at address 0x001F, with read only access. It contains the information of what the UID size in bytes was of the last tag in the field.

3.3.7 Data Buffer

The Data Buffer is a 256-byte long, located at address 0x0020 to 0x011F, with both read and write access. This buffer is used for data transfers between the tag and the user of the module.

3.3.8 Password Register

Register Name	Password							
Register Address	0x0120							
Byte Offset	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Factory Default Value	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
Read Function	PASS[0]	PASS[1]	PASS[2]	PASS[3]	PASS[4]	PASS[5]	PASS[6]	PASS[7]
Write Function								

Table 3.3.8

The Password Register is 8-bytes long, located at address 0x0120 to 0x0127, with both read and write access but only after first unlocking the device. This register is inaccessible when the module is locked. It contains an 8-byte long password which must be used with the Unlock Command to unlock protected memory. This password can be changed when the device is unlocked. The password change will only be updated and become valid after executing the Lock command.

3.3.9 AES Initialization Vector Register

Register Name	AES Initialization Vector															
Register Address	0x0128,0x0138															
Byte Offset	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Factory Default Value	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
Read Function	InVec[0]	InVec[1]	InVec[2]	InVec[3]	InVec[4]	InVec[5]	InVec[6]	InVec[7]	InVec[8]	InVec[9]	InVec[10]	InVec[11]	InVec[12]	InVec[13]	InVec[14]	InVec[15]
Write Function																

Table 3.3.9

The AES Initialization Vector Registers are two 16-bytes long registers, located at address 0x0128 to 0x0147, with both read and write access but only after first unlocking the device. These registers are inaccessible when the module is locked. They can be used as an initialization vector for the first encrypted data block. The byte order in the memory is from the least significant byte. Their role during data encryption and decryption is described in detail in chapter 5.2.8 and 5.2.9.

3.3.10 AES Key Register

Register Name	AES Key															
Register Address	0x0148, 0x0158															
Byte Offset	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Factory Default Value	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
Read Function / Write Function	Key[0]	Key[1]	Key[2]	Key[3]	Key[4]	Key[5]	Key[6]	Key[7]	Key[8]	Key[9]	Key[10]	Key[11]	Key[12]	Key[13]	Key[14]	Key[15]

Table 3.3.10

The AES Key Registers are two 16-bytes long registers, located at address 0x0148 to 0x0167, with both read and write access but only after first unlocking the device. These registers are inaccessible when the module is locked. Both registers contain an AES encryption key which can be used for encryption of the Data Buffer. Their role during encryption and decryption of the data in the buffer is described in detail in chapter 5.2.8 and 5.2.9.

3.3.11 Authentication Key / Password Register

Register Name	Authentication Key / Password					
Register Address	0x0168, 0x016E … 0x0252					
Byte Offset	0x00	0x01	0x02	0x03	0x04	0x05
Access	R/W	R/W	R/W	R/W	R/W	R/W
Factory Default Value	0x00	0x00	0x00	0x00	0x00	0x00
Read Function	KEY[0] / PASS[0]	KEY[1] / PASS[1]	KEY[2] / PASS[2]	KEY[3] / PASS[3]	KEY[4]	KEY[5]
Write Function	KEY[0] / PASS[0]	KEY[1] / PASS[1]	KEY[2] / PASS[2]	KEY[3] / PASS[3]	KEY[4]	KEY[5]

Table 3.3.11

The Authentication Key and Password Registers are forty 6-bytes long registers, located at address 0x0168 to 0x0257, with both read and write access but only after first unlocking the device. These registers are inaccessible when the module is locked. When working with MIFARE Classic® tags these registers contain the password keys used for block authentication in the tag. When working with Ultralight® and NTAG® transponders, these registers contain 4-byte passwords. There are forty Authentication Key and Password registers numbered from 0 to 39. The number of the key register to be used is passed as an argument in some commands.

3.3.12 User Memory

There are 128 bytes of memory available for the user as a protected memory space from address 0x0258 to 0x02D7. This memory is inaccessible when the device is locked. This data is stored into non-volatile memory when the Lock command is executed.

If the first byte in User Memory is 0x22, then the module tries to execute Polling command. See chapter 5.5 Polling Mode for details.

4 Communication Interface

4.1 Overview

The B1 module provides an asynchronous UART communication interface. The default baud rate is 9600 bps. The user has the option to change the transmission baud rate by using dedicated command. New settings are automatically stored in the non-volatile memory.

Communication is of a packetized command-response format which means that after every packet is sent from the master there will be a response packet sent back by the module. Apart of this type of user-master generated communication, the module also autonomously sends asynchronous packets when, for example an interrupt is triggered.

All data sent via the UART in both directions is and must be packetized. Each packet consists of a Packet Header and Packet Data. The B1 module provides several options for user selectable Packet Header and Packet Data configuration. Packet Data can be configured as Plain or Encrypted (using AES-128 encryption). The Packet Header has two configuration options: Type A and Type B. To best suit user requirements, both Packet Data and Packet Header are configurable independently.

4.2 Interface Signals

UART interface is available to the user as two communication lines – receive RX and transmit TX. On the RX line the information is transmitted from the master to the B1 module. The RX line is configured as an input on the B1 module side without any pull-up or pull-down resistors, thus it cannot be left floating. On the TX line the information is transmitted in the opposite direction. This line is driven by the module to ground or to Vcc.

The UART frame timing diagram is shown in Figure 4.2. During communication, the LSB is transmitted first and the MSB is transmitted last. The protocol is configured with one start bit, eight data bits, one stop bit and no parity bits.

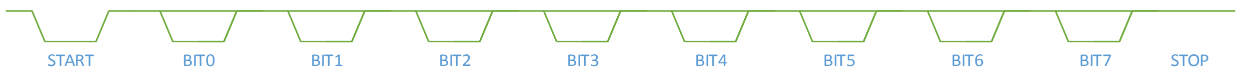


Figure 4.2

4.3 Communication Protocol

The set of UART commands provided by the module is shown in Table 4.3.13.2. The module replies to each command with an ACK once received. If any command received is not recognized, invalid parameter was sent or there is an error during communication, then the module replies with either the Invalid Command (0x01) or the Invalid Command Parameter (0x02) or the Protocol Error (0x03) packets. The module also sends asynchronous packets upon either start-up, or when an IO interrupt has been triggered, or when the Comparator Output Pin changes state, or when any RFID Command is executed. A full list of UART responses is shown in Table 4.3.1.

Response Packets			
Value	Type	Description	Additional Data
0x00	ACK (No Error)	Command was received and processed sucessfully.	Dependant upon the command
0x01	Invalid Command	Command byte is invalid.	-
0x02	Invalid Command Parameter	One of the parameters supplied within the command is invalid.	Parameter Number
0x03	Protocol Error	Packet received cannot be interpreted.	-
0x04	Memory Error	Memory allocation error when processing data.	-
0x05	System Error	Fatal error generated to indicate that the system is not working correctly	-
0x06	Module Timeout	Sent when the time delay between bytes in a packet is over 100milliiseconds.	-
0x07	Overflow	Input buffer overflow.	-
0x08	Asynchronous Packet	Sent when an asynchronous event is generated.	Data Byte with events flags.
0x09	Busy	Sent when the module is busy with the execution of an RFID command and the user has tried to write data to the RFID module memory or to put it into sleep mode.	-
0x0A	System Start	First packet send after power up.	-

Table 4.3.1

Command number	Command Description	Arguments Taken	Valid Command Respone	Possible Errors	Possible Async Packets
0x00	Dummy command	-	ACK	Protocol Error, Invalid Parameter	
0x01	Write to RFID Memory	Address, Data Size, Data	ACK, RFID Processing Finished	Protocol Error, Invalid Parameter, Busy	RFID Command End
0x02	Read from RFID Memory	Address, Data Size	ACK, Data	Protocol Error, Invalid Parameter	
0x03	Enter Sleep Mode	-	ACK	Protocol Error	
0x04	Reset	-	ACK, System Start	Protocol Error, System Error	
0x05	Set Baud Rate	Baud Rate Value	ACK, Real Baud Rate	Protocol Error, Invalid Parameter	
0x06	Set Data Type	Data Type	ACK	Protocol Error, Invalid Parameter	
0x07	Set Header Type	Header Type Configuration	ACK	Protocol Error, Invalid Parameter	
0x08	Set IO State	IO Number, IO State	ACK	Protocol Error, Invalid Parameter, System Error	
0x09	Read IO State	IO Number	ACK, IO State	Protocol Error, Invalid Parameter, System Error	
0x0A	Set IO Interrupt	IO Number, Interrupt Configuration	ACK	Protocol Error, Invalid Parameter, System Error	IO Edge
0x0B	Measure Voltage	Source, Value Format	ACK, Voltage Value	Protocol Error, Invalid Parameter, System Error	
0x0C	Measure Die Temperature	Value Format	ACK, Temperature Value	Protocol Error, Invalid Parameter, System Error	
0x0D	Set IDAC Current	Value Format, Current Value	ACK, Current Value	Protocol Error, Invalid Parameter, System Error	
0x0E	Enable Comparator	Reference Voltage, Output Pin Configuration, Async Packet Configuration, Power Supply Divider	ACK	Protocol Error, Invalid Parameter, System Error	Comparator Edge
0x0F	Disable Comparator	-	ACK	Protocol Error	
0x10	Enable PWM	IO Number, Duty Cycle, Value Format, Frequency / Period Value	ACK	Protocol Error, Invalid Parameter, System Error	
0x11	Set AES Init Vector	Initialization Vector	ACK	Protocol Error, System Error	
0x12	Set AES Key	128-bit Key	ACK	Protocol Error, System Error	
0x13	Read AES Init Vector	-	ACK, Initialization Vector	Protocol Error	
0x14	Read AES Key	-	ACK, AES Key	Protocol Error	

Table 4.3.1

4.4 CRC Calculation

Some communication and commands require CRC calculations to be performed by the B1 module. In all such cases the CRC used for calculation is a 16-bit CRC-CCITT with a polynomial equal to 0x1021. The initial value is set to 0xFFFF, the input data and the output CRC is not negated. In addition, no XOR is performed on the output value. The result of the CRC is always stored in memory with the least significant byte first. Example C code is shown below.

```
static const uint16_t CCITTCRCTable [256] = {
0x0000, 0x1021, 0x2042, 0x3063, 0x4084, 0x50a5,
0x60c6, 0x70e7, 0x8108, 0x9129, 0xa14a, 0xb16b,
0xc18c, 0xd1ad, 0xe1ce, 0xf1ef, 0x1231, 0x0210,
0x3273, 0x2252, 0x52b5, 0x4294, 0x72f7, 0x62d6,
0x9339, 0x8318, 0xb37b, 0xa35a, 0xd3bd, 0xc39c,
0xf3ff, 0xe3de, 0x2462, 0x3443, 0x0420, 0x1401,
0x64e6, 0x74c7, 0x44a4, 0x5485, 0xa56a, 0xb54b,
0x8528, 0x9509, 0xe5ee, 0xf5cf, 0xc5ac, 0xd58d,
0x3653, 0x2672, 0x1611, 0x0630, 0x76d7, 0x66f6,
0x5695, 0x46b4, 0xb75b, 0xa77a, 0x9719, 0x8738,
0xf7df, 0xe7fe, 0xd79d, 0xc7bc, 0x48c4, 0x58e5,
0x6886, 0x78a7, 0x0840, 0x1861, 0x2802, 0x3823,
0xc9cc, 0xd9ed, 0xe98e, 0xf9af, 0x8948, 0x9969,
0xa90a, 0xb92b, 0x5af5, 0x4ad4, 0x7ab7, 0x6a96,
0x1a71, 0x0a50, 0x3a33, 0x2a12, 0xdbfd, 0xcbdc,
0xfbbf, 0xeb9e, 0x9b79, 0x8b58, 0xbb3b, 0xab1a,
0x6ca6, 0x7c87, 0x4ce4, 0x5cc5, 0x2c22, 0x3c03,
0x0c60, 0x1c41, 0xedae, 0xfd8f, 0xcdec, 0xddcd,
0xad2a, 0xbd0b, 0x8d68, 0x9d49, 0x7e97, 0x6eb6,
0x5ed5, 0x4ef4, 0x3e13, 0x2e32, 0x1e51, 0x0e70,
0xff9f, 0xefbe, 0xdfdd, 0xcffc, 0xbf1b, 0xaf3a,
0x9f59, 0x8f78, 0x9188, 0x81a9, 0xb1ca, 0xa1eb,
0xd10c, 0xc12d, 0xf14e, 0xe16f, 0x1080, 0x00a1,
0x30c2, 0x20e3, 0x5004, 0x4025, 0x7046, 0x6067,
0x83b9, 0x9398, 0xa3fb, 0xb3da, 0xc33d, 0xd31c,
0xe37f, 0xf35e, 0x02b1, 0x1290, 0x22f3, 0x32d2,
0x4235, 0x5214, 0x6277, 0x7256, 0xb5ea, 0xa5cb,
0x95a8, 0x8589, 0xf56e, 0xe54f, 0xd52c, 0xc50d,
0x34e2, 0x24c3, 0x14a0, 0x0481, 0x7466, 0x6447,
0x5424, 0x4405, 0xa7db, 0xb7fa, 0x8799, 0x97b8,
0xe75f, 0xf77e, 0xc71d, 0xd73c, 0x26d3, 0x36f2,
0x0691, 0x16b0, 0x6657, 0x7676, 0x4615, 0x5634,
0xd94c, 0xc96d, 0xf90e, 0xe92f, 0x99c8, 0x89e9,
0xb98a, 0xa9ab, 0x5844, 0x4865, 0x7806, 0x6827,
0x18c0, 0x08e1, 0x3882, 0x28a3, 0xcb7d, 0xdb5c,
0xeb3f, 0xfb1e, 0x8bf9, 0x9bd8, 0xabbb, 0xbb9a,
0x4a75, 0x5a54, 0x6a37, 0x7a16, 0x0af1, 0x1ad0,
0x2ab3, 0x3a92, 0xfd2e, 0xed0f, 0xdd6c, 0xcd4d,
0xbdaa, 0xad8b, 0x9de8, 0x8dc9, 0x7c26, 0x6c07,
0x5c64, 0x4c45, 0x3ca2, 0x2c83, 0x1ce0, 0x0cc1,
0xef1f, 0xff3e, 0xcf5d, 0xdf7c, 0xaf9b, 0xbfba,
0x8fd9, 0x9ff8, 0x6e17, 0x7e36, 0x4e55, 0x5e74,
0x2e93, 0x3eb2, 0x0ed1, 0x1ef0 };

static uint16_t GetCCITTCRC(const uint8_t* Data, uint32_t Size) {
uint16_t CRC;
uint16_t Temp;
uint32_t Index;

if (Size == 0) {
return 0;
}

CRC = 0xFFFF;

for (Index = 0; Index < Size; Index++){
Temp = (uint16_t)( (CRC >> 8) ^ Data[Index] ) & 0x00FF;
CRC = CCITTCRCTable[Temp] ^ (CRC << 8);
}

return CRC;
}
```

4.5 Command / Response Packet Construction

4.5.1 Header Construction

4.5.1.1 Type A Header

The user has the option to configure the Packet Header as Type A. The entire packet construction of a Type A Header is shown in Table 4.5.1.15.1.1.

Packet with Type A Header								
Byte number	1	2	3	4	5	6	...	5 + k
Value	0x02	0x00 - 0xFF	0x00 - 0xFF	0x00 - 0xFF	0x00 - 0xFF	0x00 - 0xFF	...	0x00 - 0xFF
Type	Start Of Text	Data Size (k) LSB	Data Size (k) MSB	Header CRC LSByte	Header CRC MSByte	Data (Plain or Encrypted)		

Table 4.5.1.1

A Type A header always starts with 0x02 and is followed by two bytes representing the data size inside the packet (from byte number 6 onwards). Data size is encoded as an unsigned 16-bit number with least significant byte first. At the end of the header there is a 16-bit CRC of the header calculated as per the description in chapter 4.4. The data field follows the header and can be in plain or encrypted format. Type A Header is default.

4.5.1.2 Type B Header

The user also has the option to configure the Packet Header as Type B. The entire packet construction of a Type B Header is shown in Table 4.5.1.2.

Packet with Type B Header					
Byte number	1	2	...	n - 1	n
Value	0x02	0x00 - 0xFF / {0x02, 0x03}	...	0x00 - 0xFF / {0x02, 0x03}	0x03
Type	Start Of Text	DLE Modified Data			End Of Text

Table 4.5.1.2

This protocol using Packet Type B is a modified and simplified version of the Binary Synchronous Communication protocol. Each packet always starts with a 0x02 (STX – Start of Text) character and ends with a 0x03 (ETX – End of Text) character. The data of the packet is contained in between these two characters. The packet data is modified before sending such that there are no 0x02 and 0x03 characters sent in between the STX and ETX bytes. Whenever in the data stream there is either a 0x02, 0x03 or 0x10 value it is replaced by two characters, where the first character is 0x10 (DLE – Data Link Escape) and the second character is (0x10 + C) where C is the value of the character that has been replaced.

It means that 0x02 is replaced by [0x10, 0x12], 0x03 is replaced by [0x10, 0x13] and 0x10 is replaced by [0x10, 0x20]. This solution creates a packet where 0x02 always means beginning of transmission, 0x03 always means end of transmission and there are no confusing data bytes in between. The data field can be in plain or encrypted format.

4.5.2 Data Construction

4.5.2.1 Plain

Plain Data starts with a UART Command or Response byte and is followed by parameters bytes if required. Two bytes of CRC-16 are added at the end, calculated as per the description in chapter 4.4. The format of a Plain Command is shown in Table 4.2.2.1.1.

Plain Command						
Byte number	1	2	...	N + 1	N + 2	N + 3
Value	0x00 - 0x1E	0x00-0xFF	...	0x00 - 0xFF	0x00 - 0xFF	0x00 - 0xFF
Type	Command Byte	Parameters Byte 1	...	Parameters Byte N	CRC LSByte	CRC MSByte

Table 4.2.2.1.1

The format of a Plain Response is shown in Table 4.5.2.1.2.

Plain Response						
Byte number	1	2	...	N + 1	N + 2	N + 3
Value	0x00 - 0x02, 0xF0 - 0xF3	0x00-0xFF	...	0x00 - 0xFF	0x00 - 0xFF	0x00 - 0xFF
Type	Response Byte	Parameters Byte 1	...	Parameters Byte N	CRC LSByte	CRC MSByte

Table 4.5.2.1.2

<u>Plain Data is default.</u>

4.5.2.2 Encrypted

Encrypted Data can be used for secure communication between the module and the host. The encryption method used is AES-128. This method forces the whole packet data size (two-character string data size, a Command Byte, Parameters bytes, CRC and random fill bytes) to be an exact multiple of 16, thus the packet is filled with random data at the end and before the encryption to meet this requirement. The two characters at the beginning of the packet represent the number of meaningful bytes (characters) contained in the data packet (two Data Size bytes, one Command byte, two CRC bytes and Parameters bytes) and their minimum value is 0x05.

Encrypted Command before encryption (size has to be multiple of 16)										
Byte number	1	2	3	4	...	N + 3	N + 4	N + 5	...	16*k
Value	0x00 - 0xFF	0x00 - 0xFF	0x00 - 0x1E	0x00-0xFF	...	0x00 - 0xFF	0x00 - 0xFF	0x00 - 0xFF	...	0x00 - 0xFF
Type	Data Size (N + 1) LSB	Data Size (N + 1) MSB	Command Byte	Parameters Byte 1	...	Parameters Byte N	CRC LSByte	CRC MSByte	...	Random Fill

Table 4.5.2.2.1

Encrypted Response before encryption (size has to be multiple of 16)										
Byte number	1	2	3	4	...	N + 3	N + 4	N + 5	...	16*k
Value	0x00 - 0xFF	0x00 - 0xFF	0x00 - 0x02, 0xF0 - 0xF3	0x00-0xFF	...	0x00 - 0xFF	0x00 - 0xFF	0x00 - 0xFF	...	0x00 - 0xFF
Type	Data Size (N + 1) LSB	Data Size (N + 1) MSB	Response Byte	Parameters Byte 1	...	Parameters Byte N	CRC LSByte	CRC MSByte	...	Random Fill

Table 4.5.2.2.2

Table 4.5.2.2.1 and Table 4.5.2.2.2 show the Encrypted Command and Response packet formats before encryption is performed. The user should be aware that after encryption the size of the data stays the same, but the byte values no longer have any meaning.

4.6 Timeout

A timeout is implemented in the system to avoid the scenario that a lost byte during transmission could cause an infinite delay time for the module to respond. When the module starts receiving (which means it detects a STX character) it sets up a timer which is reset upon receipt of each successive character and turned off at the end of the transmission. When the timer reaches 100ms a timeout packet is sent, and the RX buffer is cleared. In practice this places a restriction upon the master controller that the time delay between any two bytes sent in a packet must be less than 100ms.

5 Functional Description

5.1 Overview

The RFID B1 module is designed to provide a simple interface to communicate and manage RFID tags from the MIFARE Classic®, MIFARE Ultralight® and NTAG2xx® families. The interface is provided via the UART peripheral. The user has available a set of commands to control and interact with the module. The module replies after each valid command received.

To work with RFID tags, the user must write to and read from the RFID Module memory using *Write to RFID Memory* and *Read from RFID Memory* commands. Depending upon the memory address, the role of registers located in the memory differ as per the Memory Map description given in chapter 3.3. By writing to and reading from these registers, the user can achieve the desired result.

In addition to Read and Write commands the UART interface provides an additional set of commands not related to the RFID subsystem. A full list of available UART commands is covered in chapter 5.3.

5.2 Flash Usage

Some UART commands or RFID commands store data in the flash memory. Those commands are:

- *Set Baud Rate* 0x05
- *Set Data Type* 0x06
- *Set Header Type* 0x07
- *Set AES Init Vector* 0x11
- *Set AES Key* 0x12
- RFID Command *Lock* 0x1C

The user should be aware that the flash memory has limited number of write cycles as described in chapter 2.7. The user also must consider the switching time between previous and new settings when using these commands thus after receiving ACK packet the user should wait with new packet at least 50 milliseconds.

5.3 UART Commands

5.3.1 Dummy Command (0x00)

The Dummy Command takes no arguments. It is used to check that the module alive. The module replies to this command with an ACK response and no parameters.

5.3.2 Write to RFID Memory (0x01)

Command Number	0x01		
Command Name	Write to RFID Memory		
Argument Offset [bytes]	0x00	0x02	0x04
Argument Name	Memory Address	Data Size	Data
Argument Size [bytes]	0x02	0x02	Equal to Data Size
Argument Description	Memory address in the RFID subsystem memory to where the data will be stored. This is an unsigned 16-bit number with least significant byte first.	Size in bytes of the data which will be stored. This is an unsigned 16-bit number with least significant byte first.	Data bytes to be stored in the RFID subsystem memory. Length of the data must be equal to the Data Size parameter.

Table 5.3.2

The Write to RFID Memory command takes as arguments Memory Address, Data Size and Data. The Memory Address parameter is a 16-bit unsigned number with least significant byte first, and points to where the data will be stored in the memory. Data Size parameter is a 16-bit unsigned number with least significant byte first, that informs the module how many bytes the user wants to write to the memory. Data is the variable length set of data to be written to the RFID subsystem memory.

After the B1 module has received a valid packet, it replies with an ACK response and no parameters. At the same time the packet processing starts together with any RFID command execution if a RFID command was sent. When any command execution is finished, the B1 module replies with an asynchronous packet containing the appropriate bits set.

5.3.3 Read from RFID Memory (0x02)

Command Number	0x02	
Command Name	Read from RFID Memory	
Argument Offset [bytes]	0x00	0x02
Argument Name	Memory Address	Data Size
Argument Size [bytes]	0x02	0x02
Argument Description	Memory address in the RFID subsystem memory from where the data will be read. This is an unsigned 16-bit number with least significant byte first.	Size in bytes of the data which will be read. This is an unsigned 16-bit number with least significant byte first.

Table 5.3.3

The Read from RFID Memory command takes as arguments Memory Address and Data Size. The Memory Address parameter is a 16-bit unsigned number with least significant byte first, pointing to the location in the memory from where the data will be read. The Data Size parameter is a 16-bit unsigned number with the least significant byte first that informs the module how many bytes the user wants to read from the memory.

When the B1 module receives a valid packet, it replies with an ACK response and the requested data.

5.3.4 Enter Sleep Mode (0x03)

The Enter Sleep Mode command takes no arguments. After receiving this command, the module replies with an ACK response and no parameters. Immediately afterwards the module enters Sleep Mode. To wake the module up from Sleep Mode the user must send 0x00 over the UART. After waking up the module sends ACK packet response to inform the user it is ready to receive new commands.

5.3.5 Reset (0x04)

The Reset command takes no arguments. After receiving this command, the module replies with an ACK response and no parameters. Immediately afterwards it resets. After reset the system starts the initialization procedure. At the end of this procedure it sends a System Start packet to inform the user that it is ready to receive further commands.

5.3.6 Set Baud Rate (0x05)

Command Number	0x05
Command Name	Set Baud Rate
Argument Offset [bytes]	0x00
Argument Name	Baud Rate Value
Argument Size [bytes]	0x04
Argument Description	Baud Rate Value is represented as unsigned 32-bit integer number with least significant byte first.

Table 5.1.6

The Set Baud Rate command takes as argument 4 bytes which are interpreted as a 32-bit unsigned integer. This number directly reflects the baud rate settings thus the module can work with non-standard baud rates but only within limits. When the module receives this command, it replies with an ACK response and parameters representing the real baud rate value which could be different from that requested due to limited precision of clock dividers. This response is sent at the old baud rate speed, and then the module immediately switches to the new settings. The user must consider the switching time between previous and new baud rate thus after receiving ACK packet the user should wait with new packet at least 50 milliseconds.

5.3.7 Set Data Type (0x06)

Command Number	0x06
Command Name	Set Data Type
Argument Offset [bytes]	0x00
Argument Name	Data Type Configuration
Argument Size [bytes]	0x01
Argument Description	Parameter configuring the data type: - Plain (0x00) - Encrypted (0x01)

Table 5.3.7

The Set Data Type command takes as an argument the Data Type Configuration byte. When the module receives this command, it replies with an ACK response and no parameters. The reply is formatted as per the old configuration. Immediately afterwards the module switches to the new configuration. The user must consider the switching time between previous and new settings thus after receiving ACK packet the user should wait with new packet at least 50 milliseconds.

5.3.8 Set Header Type (0x07)

Command Number	0x07
Command Name	Set Header Type
Argument Offset [bytes]	0x00
Argument Name	Header Type Configuration
Argument Size [bytes]	0x01
Argument Description	Parameter used to configure the header type. The following configurations are available: - Type A Header (0x00) - Type B Header (0x01)

Table 5.3.8

Set Header Type command takes as an argument the Header Type Configuration byte. When the module receives this command, it replies with an ACK response and no parameters. The reply is formatted as per the old configuration. Immediately afterwards the module switches to the new configuration. The user must consider the switching time between previous and new settings thus after receiving ACK packet the user should wait with new packet at least 50 milliseconds.

5.3.9 Set IO State (0x08)

Command Number	0x08	
Command Name	Set IO State	
Argument Offset [bytes]	0x00	0x01
Argument Name	IO Number	IO State
Argument Size [bytes]	0x01	0x01
Argument Description	The GPIO number to be configured. Numbering starts at 0 and depends upon the quantity of available GPIOs.	IO state to which the selected IO will be switched. The following states are available: - Disabled /Hi-Z (0x00) - Input (0x01) - Output Low (0x02) - Output High (0x03)

Table 5.3.9

The Set IO State command takes as arguments the IO Number and the user desired IO State. The IO Number is the number of the GPIO to which the state setting is to apply. Numbering starts from 0 and depends upon the number of available IOs in the system. IO State is the configuration of the IO pin. When the module receives this command, it sets up the IO state and replies with an ACK response and no parameters.

5.3.10 Read IO State (0x09)

Command Number	0x09
Command Name	Read IO State
Argument Offset [bytes]	0x00
Argument Name	IO Number
Argument Size [bytes]	0x01
Argument Description	The GPIO number whose state is to be read. Numbering starts from 0 and depends upon the quantity of available GPIOs.

Table 5.3.10

The Read IO State command takes as an argument an IO Number. This IO Number refers to the IO pin that the user wished to read the state of. Numbering starts from 0 and depends upon the number of available IOs in the system. When the module receives this command, it reads the desired IO state and replies with an ACK response and one parameter byte that represents the current IO State – 0x00 when the state is low and 0x01 when the state is high. Before reading the IO, the user must configure the IO as input.

5.3.11 Set IO Interrupt (0x0A)

Command Number	0x0A	
Command Name	Set IO Interrupt	
Argument Offset [bytes]	0x00	0x01
Argument Name	IO Number	Interrupt Configuration
Argument Size [bytes]	0x01	0x01
Argument Description	GPIO number to be configured. Numbering starts at 0 and depends upon the quantity of available GPIOs.	The interrupt configuration to be applied to the given IO. The following configurations are available: - Falling Edge (0x00) - Rising Edge (0x01) - Any Edge (0x02) - Disable (0x03)

Table 5.3.11

The Set IO Interrupt command takes as an argument the IO Number and the Interrupt Configuration. The IO Number is the number of the GPIO to which the interrupt setting is to be applied. Numbering starts from 0 and depends upon the number of available IOs in the system. The Interrupt Configuration byte refers to the configuration of the IO interrupt to be set. When the module receives this command, it configures the interrupt and replies with an ACK response and no parameters. Regardless from previous IO settings when interrupt is enabled the IO automatically changes configuration to input and when interrupt is disabled the IO is automatically disabled.

5.3.12 Measure Voltage (0x0B)

Command Number	0x0B	
Command Name	Measure Voltage	
Argument Offset [bytes]	0x00	0x01
Argument Name	Source	Value Format
Argument Size [bytes]	0x01	0x01
Argument Description	The source of the voltage measurement. The following sources are available: - ADC Pin (0x00) - Power Supply (0x01)	The format of the returned voltage value. The following formats are available: - 32-bit unsigned integer in mV (0x00) - 32-bit floating point in mV (0x01) - 32-bit floating point in V (0x02)

Table 5.3.12

The Measure Voltage command takes as an argument a Source and a Value Format. Source is a one-byte long parameter that configures the input source for the ADC. Value format is a one-byte long parameter that configures the response format (the value type and units). When the module receives this command, it measures the desired voltage and replies with an ACK response and 4 bytes of data representing the voltage in the requested format and units.

5.3.13 Measure Die Temperature (0x0C)

Command Number	0x0C
Command Name	Measure Die Temperature
Argument Offset [bytes]	0x00
Argument Name	Value Format
Argument Size [bytes]	0x01
Argument Description	The format of the returned temperature value. The following formats are available: - 32-bit signed integer in m°C (0x00) - 32-bit floating point in m°C (0x01) - 32-bit floating point in °C (0x02) - 32-bit signed integer in m°F (0x03) - 32-bit floating point in m°F (0x04) - 32-bit floating point in °F (0x05)

Table 5.3.13

The Measure Die Temperature command takes as an argument the Value Format, which is a one-byte long parameter that configures the response format (the value type and units). When the module receives this command, it measures the die temperature and replies with an ACK response and 4 bytes of data which represents the temperature in the requested format and units.

5.3.14 Set IDAC Current (0x0D)

Command Number	0x0D	
Command Name	Set IDAC Current	
Argument Offset [bytes]	0x00	0x01
Argument Name	Value Format	Current Value
Argument Size [bytes]	0x01	0x04
Argument Description	The desired number format and units of the following associated output current value. Following formats are available: - 32-bit signed integer in nA (0x00) - 32-bit floating point in nA (0x01) - 32-bit floating point in uA (0x02)	Current Value in relation to the Value Format.

Table 5.3.14

The Set IDAC Current command takes as an argument two parameters, Value Format and Current Value. Value Format is a one-byte long parameter that configures the value type and units of the following current value – Current Value. When the module receives this command, it sets up the IDAC current and replies with an ACK response and 4 bytes of data representing the real current settings in the configured format. Current values higher than 0 are setting the IDAC as a current source, values lower than 0 are setting the IDAC as a current sink. Current value 0 disables IDAC.

5.3.15 Enable Comparator (0x0E)

Command Number	0x0E			
Command Name	Enable Comparator			
Argument Offset [bytes]	0x00	0x01	0x02	0x03
Argument Name	Negative Input Conection	Output Pin Configuration	UART Asynchronous Packet Edge Sensitivity	Power Supply Divider Value
Argument Size [bytes]	0x01	0x01	0x01	0x01
Argument Description	The following options are available: - 1.25V Reference Voltage (0x00) - 2.5V Reference Voltage (0x01) - Divided Power Supply (0x02)	The following configurations are available: - Output Pin Disabled (0x00) - Output Pin Enabled (0x01) - Output Pin Enabled and Negated (0x02)	The configuration setting for the UART Asynchronous Packet Edge Sensitivity. The following options are available: - Async Packet Disabled (0x00) - Async Packet on Falling Edge (0x01) - Async Packet on Rising Edge (0x02) - Async Packet on Any Edge (0x03)	The value of the power supply divider. Available range is between 0x00 and 0x3F.

Table 5.3.15

The Enable Comparator command takes as arguments the Negative Input Connection, the Output Pin Configuration, the UART Asynchronous Packet Edge Sensitivity and the Power Supply Divider Value. Each of these parameters is one-byte long. When the module receives this command, it configures the comparator and replies with an ACK response and no parameters. The Power Supply is divided per following formula:

$$V_{ref} = V_{dd} * \frac{Divider\ Value}{63}.$$

5.3.16 Disable Comparator (0x0F)

The Disable Comparator command takes no arguments. This command disables the comparator and turns it off. When the module receives this command, it disables and turns off the comparator and replies with an ACK response and no parameters.

5.3.17 Enable PWM (0x10)

Command Number	0x10			
Command Name	Enable PWM			
Argument Offset [bytes]	0x00	0x01	0x02	0x03
Argument Name	IO Number	Duty Cycle	Value Format	Frequency / Period Value
Argument Size [bytes]	0x01	0x01	0x01	0x04
Argument Description	IO Number on which the PWM signal will be generated.	Duty Cycle of the PWM signal in %. Available range is from 1% to 99%.	The format of the following frequency/period value argument. The following format are available: - 32-bit unsigned integer in Hz (0x00) - 32-bit floating point in Hz (0x01) - 32-bit unsigned integer in us (0x02) - 32-bit floating point is s (0x03)	The value of the PWM period or frequency with regard to the value format configuration byte.

Table 5.3.17

The Enable PWM command takes as arguments the IO Number, Duty Cycle, Value Format and Frequency / Period Value. The IO Number is the number of the GPIO on which the PWM signal will be generated. Numbering starts from 0 and depends upon the number of available PWMs in the system. Duty cycle is a 1-byte long parameter representing the desired duty cycle as a percentage. Available values are between 1% and 99%. Value Format is a one-byte long parameter that configures the value type and units of the following Frequency / Period Value parameter. The last 4-byte long parameter represents the desired period or frequency of the PWM signal in accordance to Value Format. When the module receives this command, it enables and configures the PWM module and replies with an ACK response and 4 data bytes that represents the real frequency or period value which has been set up. To disable the PWM signal the user must use the Set IO State command.

5.3.18 Set AES Init Vector (0x11)

Command Number	0x11
Command Name	Set AES Init Vector
Argument Offset [bytes]	0x00
Argument Name	Initialization Vector
Argument Size [bytes]	0x10
Argument Description	A 16-byte long initialization vector used for encrypted RAW serial communication. The order is least significant byte first.

Table 5.3.18

Set AES Init Vector takes as an argument an Initialization Vector, which is a 16-byte long vector used to initialize the encryption. The AES-128 encryption used in this communication is the same as that used in RFID subsystem commands, Encrypt Data (0x08) and Decrypt Data (0x09). The encryption and decryption are done on packet data. It requires that the data size be a multiple of 16-bytes, thus the data is filled with random bytes at the end. When the module receives this command, it changes the initialization vector and replies with an ACK response and no parameters. The reply is encrypted using the new initialization vector if Encrypted Data is selected. The user must consider the switching time between previous and new settings thus after receiving ACK packet the user should wait with new packet at least 50 milliseconds.

5.3.19 Set AES Key (0x12)

Command Number	0x12
Command Name	Set AES Key
Argument Offset [bytes]	0x00
Argument Name	AES Key
Argument Size [bytes]	0x10
Argument Description	A 16-byte long encryption key that is used for encrypted raw serial communication. The order is least significant byte first.

Table 5.3.19

The Set AES Key command takes as an argument an AES Key which is a 16-byte long encryption key used for encrypting and decrypting data. The AES-128 encryption used in this communication is the same as that used in the RFID subsystem commands, Encrypt Data (0x08) and Decrypt Data (0x09). The encryption and decryption are done on packet data. It requires that the data size be a multiple of 16-bytes, thus the data is filled with random bytes at the end. When the module receives this command, it changes the AES key and replies with an ACK response and no parameters. The reply is encrypted with the new key if Encrypted Data is selected. The user must consider the switching time between previous and new settings thus after receiving ACK packet the user should wait with new packet at least 50 milliseconds.

5.3.20 Read AES Init Vector (0x13)

The Read AES Init Vector command takes no arguments. When the module receives a valid command, it replies with an ACK and the 16-byte AES Init Vector that is to be used for the encryption of data.

5.3.21 Read AES Key (0x14)

The Read AES Key command takes no arguments. When the module receives a valid command, it replies with an ACK and the bytes of the AES Key that is to be used for the encryption of data.

5.4 RFID Commands

The RFID B1 Module has a special register, the Command Register. Writing to this register is interpreted as a command execution request from the user. The command type corresponds to the value written into the register. Execution of a command starts when the communication is finished. When command execution is finished, and the result is available for the user, the module replies with an asynchronous response. The full RFID B1 command list is shown below in Table 5.4 .

Depending upon the command, different arguments are taken as parameters and various registers are modified. After each command execution, the Result Register is updated.

Value	Command Type	Arguments Taken	Memory Modified
0x00	No action	-	-
0x01	Get UID and Type	-	Result Register, Tag Type, Tag UID, Tag UID Size
0x02	Read Block	Block Address, Read Length, Authentication Key Number, Authentication Key, Buffer Offset	Result Register, Data Buffer
0x03	Write Block	Block Address, WriteLength, Authentication Key Number, Authentication Key, Buffer Offset	Result Register
0x04	Read Data Block	Block Address, Read Length, Authentication Key Number, Authentication Key, Buffer Offset	Result Register, Data Buffer
0x05	Write Data Block	Block Address, Read Length, Authentication Key Number, Authentication Key, Buffer Offset	Result Register
0x06	Read Page	Page Address, Read Length, Buffer Offset	Result Register, Data Buffer
0x07	Write Page	Page Address, Read Length, Buffer Offset	Result Register
0x08	Encrypt Data	Encryption Key Number, Initialization Vector Number, Buffer Offset, Data Length	Result Register, Data Buffer
0x09	Decrypt Data	Encryption Key Number, Initialization Vector Number, Buffer Offset, Data Length	Result Register, Data Buffer
0x0A	Read Value	Block Address, Authentication Key Number, Authentication Key, Buffer Offset	Result Register, Data Buffer
0x0B	Write Value	Block Address, Authentication Key Number, Authentication Key, 32-bit signed Value, Stored Block Address	Result Register
0x0C	Increment Value	Block Address, Authentication Key Number, Authentication Key, 32-bit signed Delta Value	Result Register
0x0D	Decrement Value	Block Address, Authentication Key Number, Authentication Key, 32-bit signed Delta Value	Result Register
0x0E	Restore Value	Block Address, Authentication Key Number, Authentication Key	Result Register
0x0F	Transfer Value	Block Address, Authentication Key Number, Authentication Key	Result Register
0x10	Recover Value	Block Address, Buffer Offset, Authentication Key Number, Authentication Key	Result Register
0x11	Get Version	-	Result Register, Data Buffer
0x12	Read Signature	-	Result Register, Data Buffer
0x13	Configure UID	Authentication Key Number, Authentication Key, UID Type	Result Register
0x14	Read Counter	Counter Number, Buffer Offset	Result Register, Data Buffer
0x15	Increment Counter	Counter Number, 24-bit signed Increment Value	Result Register
0x16	Check Tearing Event	Counter Number, Buffer Offset	Result Register, Data Buffer
0x17	Password Authentication	Buffer Offset, Password Number, Password	Result Register, Data Buffer
0x18	Halt	-	Result Register
0x19	Calculate CRC	Memory Address, Data Length, Buffer Offset	Data Buffer
0x1A	Copy Data	Destination Address, Source Address, Data Length	All
0x1B	Unlock	Password	Result Register
0x1C	Lock	-	Result Register, Non-volatile memory
0x1D	Get Module Version	-	Data Buffer
0x1E	Reset to Defaults	-	All
0x1F	Enumerate Tags UID	-	Result Register, Data Buffer
0x20	Enumerate Tags UID and Type	-	Result Register, Data Buffer
0x21	Select Tag	UID Size, UID	Result Register
0x22	Polling	Period, No. of def. Tags, Async packet/IO/PWM mode, Timeout	Result Register

Table 5.4

5.4.1 Get UID and Type (0x01)

The ‘Get UID and Type’ command takes no arguments. After receiving this command, the RFID B1 Module checks for any tag presence in the field. If there is no tag in the field, it returns ‘No Tag’ value in the ‘Tag Type Register’. If there is a tag in the field, it reads its UID and type and writes it to Tag UID and Tag Type registers. The ‘UID Size Register’ together with the ‘Result Register’ are updated also. This command must always be executed first before any other command to turn on and initialize the tag. Additionally, this command must be executed after any error when doing any operations on a tag to reset the tag and to enable it to be able to respond to further command requests. Additionally, via this command one can discover serial numbers from MIFARE Plus®, MIFARE® DESfire® and any transponder compatible with the ISO14443A standard.

5.4.2 Read Block (0x02)

Command Number	0x02				
Command Name	Read Block				
Valid Tag Types	Mifare Classic				
Argument Offset [bytes]	0x00	0x01	0x02	0x03	0x04
Argument Name	Absolute Block Number	Read Length	Data Buffer Offset	Authentication Key Number	Authentication Key
Argument Size [bytes]	0x01	0x01	0x01	0x01	0x06
Argument Description	Absolute order number of the first block in the memory that is to be read. It is counted from 0. It is not a byte address. For example, the block 0x00 in Mifare memory at sector 0x01 would have an Absolute Block Number of 0x04.	Number of blocks to read. Value has to be larger than 0.	Data Buffer offset in bytes of where the data from the tag is to be stored. The total read size and the offset must not exceed the Data Buffer length (number of bytes).	The 6 least significant bits define the Authentication Key Number from 0 to 39 that is to be used to decrypt the data read. If the most significant bit (bit 7) is set, then the key will be used as key B. If bit 7 is zero it will be used as key A. If bit 6 is set, then the key will be taken from the following argument list (next 6 bytes).	Optional parameter used when bit 6 in Authentication Key Number argument is set. This Key will be used as Key B if bit 7 in Authentication Key Number is set or as Key A if bit 7 is not set. The byte order is the least significant byte first.

Table 5.4.2

The Read Block command takes as arguments block number of the first block to read (Absolute Block Number), the number of blocks to read (Read Length), the byte offset in the Data Buffer (Data Buffer Offset), the Authentication Key Number and (optionally) the Authentication Key. If the Authentication Key is set-up, the command first tries to authenticate the tag memory sector which contains the first block. If authentication fails, the ‘Result Register’ is updated with the appropriate error value and command execution finishes. After a successful authentication, the first block is read and copied to the Data Buffer. The command continues to read and copy following blocks if the Read Length is higher than 1. If the next blocks fall into another tag memory sector, this sector is authenticated using the same key. If everything goes well all block bytes from all blocks are copied to the Data Buffer starting from Data Buffer Offset, and the Result Register is updated with the ‘No Error’ value. If the RFID Module is unable to read tag memory, an error value is stored in the Result Register.

5.4.3 Write Block (0x03)

Command Number	0x03				
Command Name	Write Block				
Valid Tag Types	Mifare Classic				
Argument Offset [bytes]	0x00	0x01	0x02	0x03	0x04
Argument Name	Absolute Block Number	Write Length	Data Buffer Offset	Authentication Key Number	Authentication Key
Argument Size [bytes]	0x01	0x01	0x01	0x01	0x06
Argument Description	Absolute order number of the first block in the memory that is to be written. It is counted from 0. It is not a byte address. For example, the block 0x00 in Mifare memory at sector 0x01 would have an Absolute Block Number of 0x04.	Number of blocks to write. Value has to be larger than 0.	Data Buffer offset in bytes from where the data is to be taken to write to the tag. The total write size and the offset must not exceed the Data Buffer length (number of bytes).	The 6 least significant bits define the Authentication Key Number from 0 to 39 that is to be used to encrypt the data that is to be written. If the most significant bit (bit 7) is set, then the key will be used as key B. If bit 7 is zero it will be used as key A. If bit 6 is set, then the key will be taken from the following argument list (next 6 bytes).	Optional parameter used when bit 6 in Authentication Key Number argument is set. This Key will be used as Key B if bit 7 in Authentication Key Number is set or as Key A if bit 7 is not set. The byte order is the least significant byte first.

Table 5.4.3

The Write Block command takes as arguments block number of the first block to write (Absolute Block Number), the number of blocks to write (Write Length), the byte offset in the Data Buffer (Data Buffer Offset), the Authentication Key Number and (optionally) the Authentication Key. If the Authentication Key is set-up, the command first tries to authenticate the tag memory sector which contains the first block. If authentication fails, the Result Register is updated with the appropriate error code and command execution finishes. After a successful authentication, the data is copied from the Data Buffer to the first block. The command continues to copy data from the Data Buffer to the following blocks if the Write Length is higher than 1. If subsequent blocks fall into another tag memory sector, this sector is authenticated using the same key. If everything goes well all block bytes are copied from the Data Buffer starting from Data Buffer Offset, and the Result Register is updated with the 'No Error' code. If the RFID Module is unable to write to the blocks an appropriate error code is stored in the Result Register.

5.4.4 Read Data Block (0x04)

Command Number	0x04				
Command Name	Read Data Block				
Valid Tag Types	Mifare Classic				
Argument Offset [bytes]	0x00	0x01	0x02	0x03	0x04
Argument Name	Absolute Block Number	Read Length	Data Buffer Offset	Authentication Key Number	Authentication Key
Argument Size [bytes]	0x01	0x01	0x01	0x01	0x06
Argument Description	Absolute order number of the first block in the memory that is to be read. It is counted from 0. It is not a byte address. For example, the block 0x00 in Mifare memory at sector 0x01 would have an Absolute Block Number of 0x04.	Number of blocks to read. Value has to be larger than 0.	Data Buffer offset in bytes of where the data from the tag is to be stored. The total read size and the offset must not exceed the Data Buffer length (number of bytes).	The 6 least significant bits define the Authentication Key Number from 0 to 39 that is to be used to decrypt the data read. If the most significant bit (bit 7) is set, then the key will be used as key B. If bit 7 is zero it will be used as key A. If bit 6 is set, then the key will be taken from the following argument list (next 6 bytes).	Optional parameter used when bit 6 in Authentication Key Number argument is set. This Key will be used as Key B if bit 7 in Authentication Key Number is set or as Key A if bit 7 is not set. The byte order is the least significant byte first.

Table 5.4.4

The Read Data Block command takes as arguments block number of the first block to read (Absolute Block Number), the number of blocks to read (Read Length), the byte offset in the Data Buffer (Data Buffer Offset), the Authentication Key Number and (optionally) the Authentication Key. If the Authentication Key is set-up, the command first tries to authenticate the tag memory sector which contains the first block. If authentication fails, the Result Register is updated with the appropriate error value and command execution finishes. After successful authentication, the first block is read and copied to the Data Buffer. The command continues to read and copy following blocks if the Read Length is higher than 1. The difference between the Read Data Block command and the Read Block command is that the first one omits blocks containing authentication keys and lock bits. It only reads the data blocks from tag memory. If the next blocks fall into another tag memory sector, this sector is authenticated using the same key. If everything goes well all block bytes from all blocks are copied to the Data Buffer starting from the Data Buffer Offset, and the Result Register is updated with the 'No Error' value. If the RFID Module is unable to read tag memory, an error value is stored in the Result Register.

5.4.5 Write Data Block (0x05)

Command Number	0x05				
Command Name	Write Data Block				
Valid Tag Types	Mifare Classic				
Argument Offset [bytes]	0x00	0x01	0x02	0x03	0x04
Argument Name	Absolute Block Number	Write Length	Data Buffer Offset	Authentication Key Number	Authentication Key
Argument Size [bytes]	0x01	0x01	0x01	0x01	0x06
Argument Description	Absolute order number of the first block in the memory that is to be written. It is counted from 0. It is not a byte address. For example, the block 0x00 in Mifare memory at sector 0x01 would have an Absolute Block Number of 0x04.	Number of blocks to write. Value has to be larger than 0.	Data Buffer offset in bytes from where the data is to be taken to write to the tag. The total write size and the offset must not exceed the Data Buffer length (number of bytes).	The 6 least significant bits define the Authentication Key Number from 0 to 39 that is to be used to encrypt the data that is to be written. If the most significant bit (bit 7) is set, then the key will be used as key B. If bit 7 is zero it will be used as key A. If bit 6 is set, then the key will be taken from the following argument list (next 6 bytes).	Optional parameter used when bit 6 in Authentication Key Number argument is set. This Key will be used as Key B if bit 7 in Authentication Key Number is set or as Key A if bit 7 is not set. The byte order is the least significant byte first.

Table 5.4.5

The Write Data Block command takes as arguments block number of the first block to write (Block Address), the number of blocks to write (Write Length), the byte offset in the Data Buffer (Data Buffer Offset), the Authentication Key Number and (optionally) the Authentication Key. If the Authentication Key is set-up, the command first tries to authenticate the tag memory sector which contains the first block. If authentication fails, the 'Result Register' is updated with the appropriate error value and command execution finishes. After a successful authentication, the data is copied from the Data Buffer to the first block. The command continues to copy data from the Data Buffer to the following blocks if the Write Length is higher than 1. The difference between Write Data Block command and Write Block command is that the first one omits blocks containing authentication keys and lock bits. It only writes to the data blocks in tag memory. If subsequent blocks fall into another tag memory sector, this sector is authenticated using the same key. If everything goes well all block bytes are copied from the Data Buffer starting from the Data Buffer Offset, and the Result Register is updated with a 'No Error' value. If the RFID Module is unable to write to the blocks, an appropriate error code is stored in the Result Register.

5.4.6 Read Page (0x06)

Command Number	0x06		
Command Name	Read Page		
Valid Tag Types	Mifare Ultralight, Mifare Ultralight EV1, NTAG		
Argument Offset [bytes]	0x00	0x01	0x02
Argument Name	Absolute Page Number	Read Length	Data Buffer Offset
Argument Size [bytes]	0x01	0x01	0x01
Argument Description	Absolute order number of the first page in the memory that is to be read. It is counted from 0. It is not a byte address.	Number of pages to read. Value has to be larger than 1.	Data Buffer offset in bytes of where in memory the data from the tag is to be stored. Total read size and the offset cannot exceed the Data Buffer length.

Table 5.4.6

The Read Page command takes as arguments page number of the first page to read (Absolute Page Number), the number of pages to read (Read Length) and the byte offset in the Data Buffer (Data Buffer Offset). If any of the pages to be read are password protected a Password Authentication is necessary before doing a read operation. If read fails, the Result Register is updated with an error value and command execution finishes. After successful communication, the first page is read and copied to the Data Buffer. Command continues to read and copy subsequent pages if the Read Length is higher than 1. If everything goes well all bytes from all pages are copied to the Data Buffer starting from Data Buffer Offset, and the Result Register is updated with a 'No Error' value.

5.4.7 Write Page (0x07)

Command Number	0x07		
Command Name	Write Page		
Valid Tag Types	Mifare Ultralight, Mifare Ultralight EV1, NTAG		
Argument Offset [bytes]	0x00	0x01	0x02
Argument Name	Absolute Page Number	Write Length	Data Buffer Offset
Argument Size [bytes]	0x01	0x01	0x01
Argument Description	Absolute order number of the first page in the memory to which the data is to be written. It is counted from 0. It is not a byte address.	Number of pages to write. Value has to be larger than 1.	Data Buffer offset in bytes from where the data to be written to the tag is to be taken. Total write size and the offset cannot exceed the Data Buffer length.

Table 5.4.7

The Write Page command takes as arguments page number of the first page to be written (Absolute Page Number), the number of pages to be written (Write Length) and the byte offset in the Data Buffer (Data Buffer Offset). If any of the pages to be written are password protected, then a Password Authentication is necessary before doing a write operation. If write fails, the Result Register is updated with an error value and the command execution finishes. After successful communication, the data is copied from the Data Buffer to the first page. This Command continues to copy to subsequent pages if the Write Length is higher than 1. If everything goes well all bytes are copied from the Data Buffer starting from the Data Buffer Offset, and the Result Register is updated with a 'No Error' value.

5.4.8 Encrypt Data (0x08)

Command Number	0x08			
Command Name	Encrypt Data			
Valid Tag Types	-			
Argument Offset [bytes]	0x00	0x01	0x02	0x03
Argument Name	Encryption Key Number	Initialization Vector Number	Data Buffer Offset	Data Length
Argument Size [bytes]	0x01	0x01	0x01	0x01
Argument Description	Number of the AES encryption key to be used for encryption. Valid numbers are 0x00 and 0x01.	Number of the Initialization Vector to be used for encryption. Valid numbers are 0x00 and 0x01.	Data Buffer offset in 16-byte blocks from where encryption will start. Total data size and the offset cannot exceed the Data Buffer length.	Number of 16 byte blocks of data to be encrypted. This value can be between 1 and 16.

Table 5.4.8

The Encrypt Data command takes as command arguments the Encryption Key Number (0x00 or 0x01), the Initialization Vector Number (0x00 or 0x01), the Data Buffer Offset (16-bytes blocks) and the Data Length (16-bytes blocks). This command encrypts the 'Data Length' number of 16-byte blocks using the AES128 CBC encryption methodology. It operates only within the Data Buffer. This encryption method is shown in Figure 5..

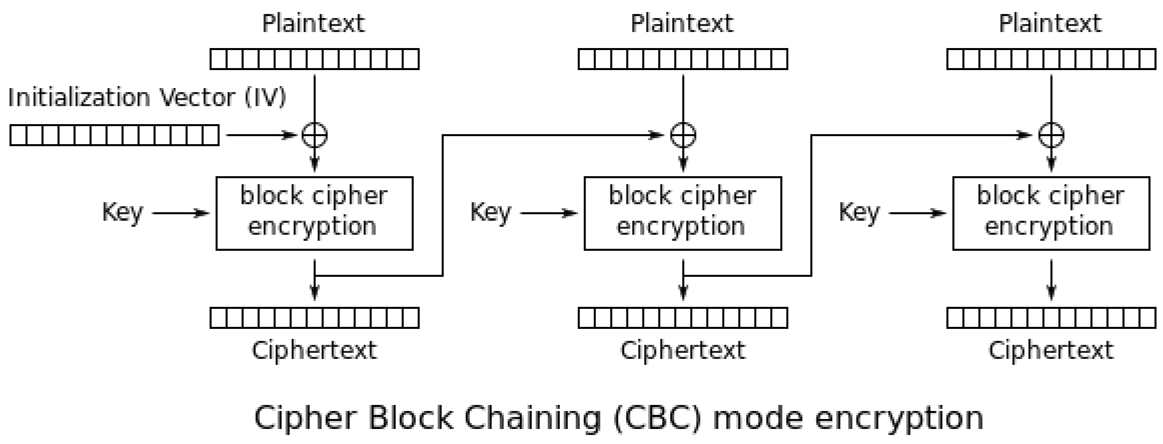


Figure 5.4.8

The Initialization Vector is used at the beginning to encrypt the first plaintext. For each subsequent 16-byte block instead of the Initialization Vector, the algorithm uses the ciphertext output of the previous 16-byte block encryption. The same AES Key is used for all blocks. When working with the Data Buffer during encryption the plaintext is substituted with the corresponding ciphertext.

5.4.9 Decrypt Data (0x09)

Command Number	0x09			
Command Name	Decrypt Data			
Valid Tag Types	-			
Argument Offset [bytes]	0x00	0x01	0x02	0x03
Argument Name	Decryption Key Number	Initialization Vector Number	Data Buffer Offset	Data Length
Argument Size [bytes]	0x01	0x01	0x01	0x01
Argument Description	The number of the AES Key to be used for decryption. Valid numbers are 0x00 and 0x01.	Number of the initialization Vector to be used for decryption. Valid numbers are 0x00 and 0x01.	Data Buffer offset in 16-byte blocks from where the decryption is to start. Total data size and the offset cannot exceed the Data Buffer length.	The number of 16-byte blocks to be decrypted. This value has to be between 1 and 16.

Table 5.4.9

The Decrypt Data command takes as arguments the Decryption Key Number (0x00 or 0x01), the Initialization Vector Number (0x00 or 0x01), the Data Buffer Offset (16-bytes) and the Data Length (16-bytes). This command decrypts the 'Data Length' number of 16-bytes blocks using the AES128 CBC decryption methodology. It operates only within the Data Buffer. This decryption method is shown in Figure 5.4.9.

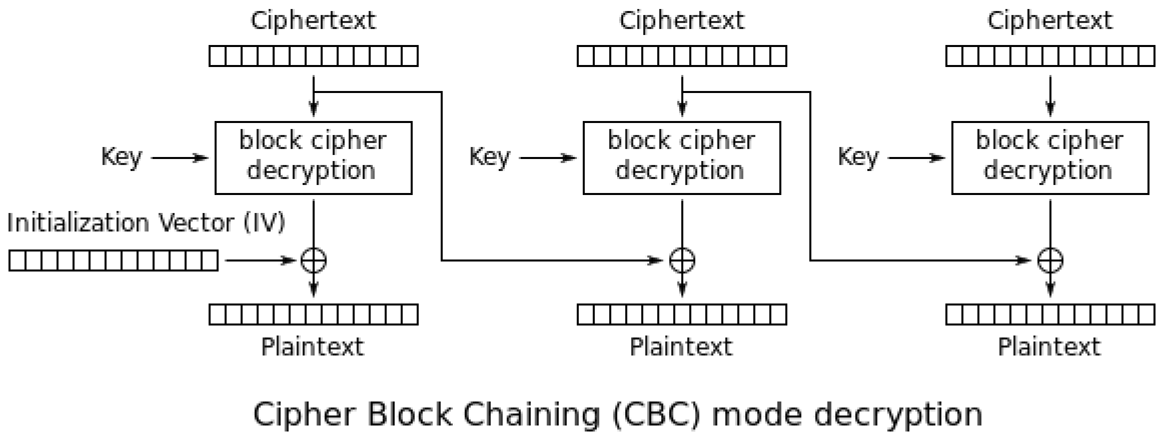


Figure 5.4.9

The Initialization Vector is used at the beginning to decrypt the first ciphertext. For each subsequent 16-byte block, instead of the Initialization Vector, the algorithm uses the ciphertext output of the previous 16-byte block decryption. The same AES Key is used for all blocks. When working with the Data Buffer during decryption the ciphertext is substituted with the corresponding plaintext.

5.4.10 Read Value (0x0A)

Command Number	0x0A			
Command Name	Read Value			
Valid Tag Types	Mifare Classic			
Argument Offset [bytes]	0x00	0x01	0x02	0x03
Argument Name	Absolute Block Number	Data Buffer Offset	Authentication Key Number	Authentication Key
Argument Size [bytes]	0x01	0x01	0x01	0x06
Argument Description	Absolute order number of the first block in the memory that is to be read. It is counted from 0. It is not a byte address. For example, the block 0x00 in Mifare memory at sector 0x01 would have an Absolute Block Number of 0x04. The block has to be formatted as a value type block before reading.	Data Buffer offset in bytes from where the data read from the tag will be stored. Total read size and the offset cannot exceed the Data Buffer length.	The 6 least significant bits define the Authentication Key from 0 to 39 that is to be used to read the data. If the most significant bit (bit 7) is set, then the key will be used as Key B. If bit 7 is zero, then it will be used as Key A. If bit 6 is set, then the key will be taken from the following argument list (next 6 bytes).	This optional parameter is only used when bit 6 of the immediately preceding Authentication Key Number argument is set. If used, this key will be used as Key A if bit 7 of the preceding Authentication number is set or as Key B if not set. The byte order is the least significant byte first.

Table 5.4.10

The Read Value command takes as arguments block number of the first block to read (Absolute Block Number), the offset of the Data Buffer where the value and read address will be stored (Data Buffer Offset), the Authentication Key Number and (optionally) the Authentication Key. This command has the same functionality as the Read Block command except it treats the block as a special Value type defined by the MIFARE® standard. It tries to parse the block content to this 4-byte signed value and to read a-byte address stored in last four bytes of the block. If read was successful, the value is stored in the Data Buffer at the pointed offset, with the address byte stored straight after the value. The value is stored as a signed 32-bit integer, with least significant byte first, and the address is stored as an unsigned 8-bit value.

5.4.11 Write Value (0x0B)

Command Number	0x0B				
Command Name	Write Value				
Valid Tag Types	Mifare Classic				
Argument Offset [bytes]	0x00	0x01	0x05	0x06	0x07
Argument Name	Absolute Block Number	Value	Stored Block Address	Authentication Key Number	Authentication Key
Argument Size [bytes]	0x01	0x04	0x01	0x01	0x06
Argument Description	Absolute order number of the first block in the memory that is to be written. It is counted from 0. It is not a byte address. For example, the block 0x00 in Mifare memory at sector 0x01 would have an Absolute Block Number of 0x04. The block will be formatted as a value type block after writing.	Signed 32-bit value that is to be written to the value block. The least significant byte is stored first.	8-bit unsigned value to be stored in the last 4 bytes of the block memory.	The 6 least significant bits (bits 0-5) define the Authentication Key Number from 0 to 39 to be used to encrypt the data. If the most significant bit is set (bit 7), then the key will be used as Key B, if clear then the key will be used as Key A. If bit 6 is set, then the key will be taken from the following argument list (next 6 bytes).	Optional parameter used when bit 6 in Authentication Key Number argument is set. This Key will be used as Key B if bit 7 in Authentication Key Number is set or as Key A if bit 7 is not set. The byte order is the least significant byte first.

Table 5.2.11

The Write Value command takes as arguments block number of the first block to write (Absolute Block Number), the signed 32-bit integer value to be stored (Value), the block address value to store in the memory block (Stored Block Address), the Authentication Key Number and (optionally) the Authentication Key. This command has the same functionality as the Write Block command, except that it treats the block as a special Value type as defined by the MIFARE® standard. It tries to format the block content to this 4-byte signed value together with the address in the last 4 bytes of the memory.

5.4.12 Increment Value (0x0C)

Command Number	0x0C			
Command Name	Increment Value			
Valid Tag Types	Mifare Classic			
Argument Offset [bytes]	0x00	0x01	0x05	0x06
Argument Name	Absolute Block Number	Delta Value	Authentication Key Number	Authentication Key
Argument Size [bytes]	0x01	0x04	0x01	0x06
Argument Description	Absolute order number of the first block in the memory from which the value is to be read and incremented. It is counted from 0. It is not a byte address. For example, the block 0x00 in Mifare memory at sector 0x01 would have an Absolute Block Number of 0x04. Block has to be formatted as a value type block before reading.	Signed 32-bit number to be added to the value read from the tag block.	The 6 least significant bits define the Authentication Key Number from 0 to 39 that is to be used to read the data. If the most significant bit (bit 7) is set, then the key will be used as Key B. If it is zero it will be used as Key A. If bit 6 is set, then the key will be taken from the argument list (next 6 bytes).	Optional parameter used when bit 6 in the preceding Authentication Key Number argument is set. This key will be used as Key B if bit 7 of the Authentication Key Number is set, or as Key A if it is zero. The byte order is the least significant byte first.

Table 5.4.12

The Increment Value command is an operation on a value-type block as defined by the MIFARE® standard. It takes as arguments block number of the block where the value is stored (Absolute Block Number), the signed 32-bit integer number which will be added to the value (Delta Value), the Authentication Key Number and (optionally) the Authentication Key. The command reads the value from the block to the volatile memory on the tag and increments it by the selected Delta Value. There is no further operation done. To store the value in the same or another block (copy to non-volatile memory on the tag), the user must execute the Transfer Value command.

5.4.13 Decrement Value (0x0D)

Command Number	0x0D			
Command Name	Decrement Value			
Valid Tag Types	Mifare Classic			
Argument Offset [bytes]	0x00	0x01	0x05	0x06
Argument Name	Absolute Block Number	Delta Value	Authentication Key Number	Authentication Key
Argument Size [bytes]	0x01	0x04	0x01	0x06
Argument Description	Absolute order number of the first block in the memory from which the value is to be read and decremented. It is counted from 0. It is not a byte address. For example, the block 0x00 in Mifare memory at sector 0x01 would have an Absolute Block Number of 0x04. Block has to be formatted as a value type block before reading.	Signed 32-bit number to be substracted from the value read from the tag block.	The 6 least significant bits define the Authentication Key Number from 0 to 39 that is to be used to read the data. If the most significant bit (bit 7) is set, then the key will be used as Key B. If it is zero it will be used as Key A. If bit 6 is set, then the key will be taken from the argument list (next 6 bytes).	Optional parameter used when bit 6 in the preceding Authentication Key Number argument is set. This key will be used as Key B if bit 7 of the Authentication Key Number is set, or as Key A if it is zero. The byte order is the least significant byte first.

Table 5.4.13

The Decrement Value command is an operation on a value-type block, as defined by the MIFARE® standard. It takes as arguments block number of the block where the value is stored (Absolute Block Number), the signed 32-bit integer number which will be subtracted from the value (Delta Value), the Authentication Key Number and (optionally) the Authentication Key. The command reads the value from the block into the volatile memory on the tag and decrements it by the delta value. There is no further operation done. To store the value in the same or another block (copy to non-volatile memory on the tag), the user must execute the Transfer Value command.

5.4.14 Restore Value (0x0E)

Command Number	0x0E		
Command Name	Restore Value		
Valid Tag Types	Mifare Classic		
Argument Offset [bytes]	0x00	0x01	0x02
Argument Name	Absolute Block Number	Authentication Key Number	Authentication Key
Argument Size [bytes]	0x01	0x01	0x06
Argument Description	Absolute order number of the first block in the memory from which the value is to be read and restored. It is counted from 0. It is not a byte address. For example, the block 0x00 in Mifare memory at sector 0x01 would have an Absolute Block Number of 0x04. The block has to be formatted as a value type block before reading.	The 6 least significant bits define the Authentication Key Number from 0 to 39 that is to be used to read the data. If the most significant bit (bit 7) is set, then the key will be used as Key B. If it is zero it will be used as Key A. If bit 6 is set, then the key will be taken from the argument list (next 6 bytes).	Optional parameter used when bit 6 in the preceding Authentication Key Number argument is set. This key will be used as Key B if bit 7 of the Authentication Key Number is set, or as Key A if it is zero. The byte order is the least significant byte first.

Table 5.4.14

The Restore Value command is an operation on a value-type block, as defined by the MIFARE® standard. It takes as arguments block number of the block where the value is stored (Absolute Block Number), the Authentication Key Number and (optionally) the Authentication Key. If the block is properly formatted, then the value from the block is copied to a volatile memory register on the tag. There is no further operation done. To store the value in the same or another block (copy to non-volatile memory on the tag), the user must execute the Transfer Value command.

5.4.15 Transfer Value (0x0F)

Command Number	0x0F		
Command Name	Transfer Value		
Valid Tag Types	Mifare Classic		
Argument Offset [bytes]	0x00	0x01	0x02
Argument Name	Absolute Block Number	Authentication Key Number	Authentication Key
Argument Size [bytes]	0x01	0x01	0x06
Argument Description	Absolute order number of the first block in the memory from which the value is to be transferred. It is counted from 0. It is not a byte address. For example, the block 0x00 in Mifare memory at sector 0x01 would have an Absolute Block Number of 0x04. The block is formatted as a value type block after execution of this command.	The 6 least significant bits define the Authentication Key Number from 0 to 39 that is to be used to read the data. If the most significant bit (bit 7) is set, then the key will be used as Key B. If it is zero it will be used as Key A. If bit 6 is set, then the key will be taken from the argument list (next 6 bytes).	Optional parameter used when bit 6 in the preceding Authentication Key Number argument is set. This key will be used as Key B if bit 7 of the Authentication Key Number is set, or as Key A if it is zero. The byte order is the least significant byte first.

Table 5.4.15

The Transfer Value command is an operation on a value-type block as defined by the MIFARE® standard. It takes as arguments block number of the block where the value is stored (Absolute Block Number), the Authentication Key Number and (optionally) the Authentication Key. The value from the volatile register on the tag is copied to the pointed block. The block is formatted as a value-type block during this operation.

5.4.16 Recover Value (0x10)

Command Number	0x10			
Command Name	Recover Value			
Valid Tag Types	Mifare Classic			
Argument Offset [bytes]	0x00	0x01	0x02	0x03
Argument Name	Absolute Block Number	Data Buffer Offset	Authentication Key Number	Authentication Key
Argument Size [bytes]	0x01	0x01	0x01	0x06
Argument Description	Absolute order number of the first block in the memory from which the value is to be read and recovered. It is counted from 0. It is not a byte address. For example, the block 0x00 in Mifare memory at sector 0x01 would have an Absolute Block Number of 0x04. The block must be formatted as a value type block before reading.	Data Buffer offset in bytes from where the data from the tag is to be stored. Total read size and the offset cannot exceed the Data Buffer length.	The 6 least significant bits define the Authentication Key Number from 0 to 39 that is to be used to read the data. If the most significant bit (bit 7) is set, then the key will be used as Key B. If it is zero it will be used as Key A. If bit 6 is set, then the key will be taken from the argument list (next 6 bytes).	Optional parameter used when bit 6 in the preceding Authentication Key Number argument is set. This key will be used as Key B if bit 7 of the Authentication Key Number is set, or as Key A if it is zero. The byte order is the least significant byte first.

Table 5.4.16

The Recover Value command takes as arguments block number of the block where the value is stored (Absolute Block Number), the offset of the Data Buffer where the value and read address will be stored (Data Buffer Offset), the Authentication Key Number and (optionally) the Authentication Key. This command has the same functionality as the Read Value command, except that it can be used on a block which is corrupted – it tries to recover data from a corrupted block. The format of a value-type block allows for some bits to be corrupted and it still be possible to read and recover the proper value. The 4-byte signed value followed by 1-byte address value is stored in the Data Buffer under the Data Buffer Offset index.

5.4.17 Get Version (0x11)

The Get Version command doesn't take any arguments. This command can be used with MIFARE Ultralight® EV1, NTAG213, NTAG215 and NTAG216 tags. After successful reading, the first 8 bytes in the data buffer are filled with data defined by the NXP standard. Example are shown in Table 5.4.17.1 (MIFARE Ultralight® EV1) and Table 5.4.17.2 (NTAG®).

Byte No	Description	MF0UL11/MF0ULH11	MF0UL21/MF0ULH21	Interpretation
0	Fixed header	0x00	0x00	
1	Vendor ID	0x04	0x04	NXP Semiconductors
2	Product type	0x03	0x03	MIFARE Ultralight
3	Product subtype	0x01/0x02	0x01/0x02	17 pF / 50 pF
4	Major product version	0x01	0x01	EV1
5	Minor product version	0x00	0x00	V0
6	Storage size	0x0B	0x0E	
7	Protocol type	0x03	0x03	ISO/IEC 14443-3 compliant

Table 5.4.17.1

Byte No	Description	NTAG213	NTAG215	NTAG216	Interpretation
0	Fixed header	0x00	0x00	0x00	
1	Vendor ID	0x04	0x04	0x04	NXP Semiconductors
2	Product type	0x04	0x04	0x04	NTAG
3	Product subtype	0x02	0x02	0x02	50 pF
4	Major product version	0x01	0x01	0x01	1
5	Minor product version	0x00	0x00	0x00	V0
6	Storage size	0x0F	0x11	0x13	
7	Protocol type	0x03	0x03	0x03	ISO/IEC 14443-3 compliant

Table 5.4.17.2

Please refer to the NXP documentation for more information.

5.4.18 Read Signature (0x12)

The Read Signature command doesn't take any arguments. This command can be used with MIFARE Ultralight® EV1, NTAG213, NTAG215 and NTAG216 tags. After successful reading, the first 32 bytes in the data buffer are filled with the ECC signature defined by the NXP standard. Please refer to the NXP documentation for more information.

5.4.19 Configure UID (0x13)

Command Number	0x13		
Command Name	Configure UID		
Valid Tag Types	Some Mifare Classic		
Argument Offset [bytes]	0x00	0x01	0x02
Argument Name	UID Type	Authentication Key Number	Authentication Key
Argument Size [bytes]	0x01	0x01	0x06
Argument Description	UID Type configuration byte.	The 6 least significant bits define the Authentication Key Number from 0 to 39 that is to be used to read the data. If the most significant bit (bit 7) is set, then the key will be used as Key B. If it is zero it will be used as Key A. If bit 6 is set, then the key will be taken from the argument list (next 6 bytes).	Optional parameter used when bit 6 in the preceding Authentication Key Number argument is set. This key will be used as Key B if bit 7 of the Authentication Key Number is set, or as Key A if it is zero. The byte order is the least significant byte first.

Table 5.4.19.1

The Configure UID command takes as arguments the UID configuration byte (UID Type), the Authentication Key Number and (optionally) the Authentication Key. This command changes the configuration of the UID on some MIFARE Classic® tags. The UID Type parameter is explained in Table 5.4.19.2.

UID Configuration Parameter Value	Description
0x00	Anti-collision and selection with the double size UID.
0x01	Anti-collision and selection with the double size UID and optional usage of a selection process shortcut.
0x02	Anti-collision and selection with a single size random ID.
0x03	Anti-collision and selection with a single size NUID where the NUID is calculated from the 7-byte UID.

Table 5.4.19.2

5.4.20 Read Counter (0x14)

Command Number	0x14	
Command Name	Read Counter	
Valid Tag Types	Ultralight EV1, NTAG213, NTAG215, NTAG216, NTAG213F, NTAG216F	
Argument Offset [bytes]	0x00	0x01
Argument Name	Counter Number	Data Buffer Offset
Argument Size [bytes]	0x01	0x01
Argument Description	Counter number cannot exceed number of counters in the tag.	Data Buffer offset in bytes from where the counter value is to be stored. Total counter value size and the offset cannot exceed the Data Buffer length.

Table 5.4.20

The Read Counter command takes as arguments the tag Counter Number and the Data Buffer Offset in bytes. This command reads the counter value of the counter pointed to by the Counter Number and stores it in the Data Buffer at the Data Buffer Offset index as an unsigned 24-bit integer with the least significant byte first.

5.4.21 Increment Counter (0x15)

Command Number	0x15	
Command Name	Increment Counter	
Valid Tag Types	Ultralight EV1	
Argument Offset [bytes]	0x00	0x01
Argument Name	Counter Number	Increment Value
Argument Size [bytes]	0x01	0x03
Argument Description	Counter number cannot exceed number of counters in the tag.	24-bit unsigned integer value which will be added to the counter value. Byte order is least significant byte first.

Table 5.4.21

The Increment Counter command takes as arguments the tag Counter Number and the Increment Value to be added to the counter value. This command increments the counter value of the counter pointed to by the Counter Number. The Increment Value is a 24-bit unsigned number with the least significant byte first.

5.4.22 Check Tearing Event (0x16)

Command Number	0x16	
Command Name	Check Tearing Event	
Valid Tag Types	Ultralight EV1	
Argument Offset [bytes]	0x00	0x01
Argument Name	Counter Number	Data Buffer Offset
Argument Size [bytes]	0x01	0x01
Argument Description	Counter number cannot exceed number of counters in the tag.	Data Buffer offset in bytes where the check result will be stored. Offset cannot exceed the Data Buffer length.

Table 5.4.22

The Check Tearing Event command takes as arguments the tag Counter Number and the Data Buffer Offset in bytes. This command checks whether there was a tearing event in the counter and stores the flag value in the Data Buffer at the Data Buffer Offset index. The value ‘0x00’ is stored if there has been no tearing event, and ‘0x01’ is stored if a tearing event occurred.

5.4.23 Password Authentication (0x17)

Command Number	0x17		
Command Name	Password Authentication		
Valid Tag Types	Ultralight EV1, NTAG210, NTAG212, NTAG213, NTAG215, NTAG216, NTAG213F, NTAG216F		
Argument Offset [bytes]	0x00	0x01	0x02
Argument Name	Data Buffer Offset	Password Number	Password
Argument Size [bytes]	0x01	0x01	0x04
Argument Description	Data Buffer offset in bytes where the PACK result will be stored. Offset cannot exceed the Data Buffer length.	The 6 least significant bytes define the Password Number for 0 to 39 that is to be used to authenticate. If the value is equal to 0x80 the password will be taken from next four bytes of the command parameters.	An Optional parameter to be used when the preceding password number is set to 0x80. The byte order is the least significant byte first.

Table 5.4.23

The Password Authentication command takes as arguments the Data Buffer Offset where the PACK result will be stored, the Password Number and (optionally) the Password. This command tries to authenticate the tag using the chosen (pointed to) password. The 2-byte PACK result is stored in the Data Buffer under Data Buffer Offset index.

5.4.24 Halt (0x18)

The Halt command takes no arguments. It halts the tag and turns off the RF field. It must be executed at the end of each operation on a tag to disable the antenna and reduce the power consumption.

5.4.25 Calculate CRC (0x19)

Command Number	0x19		
Command Name	Calculate CRC		
Valid Tag Types	-		
Argument Offset [bytes]	0x00	0x02	0x04
Argument Name	Memory Address	Data Length	Data Buffer Offset
Argument Size [bytes]	0x02	0x02	0x01
Argument Description	Byte address in the memory from which the CRC calculation is to start. This is an unsigned 16-bit value with least significant byte first.	Data length in bytes upon which the CRC is to be performed. This is an unsigned 16-bit value with least significant byte first.	Buffer offset in bytes where the CRC value is to be stored. The CRC value is 16-bit unsigned with least significant byte first.

Table 5.4.25

The Calculate CRC command takes as arguments the Memory Address, the Data Length in bytes and the Data Buffer Offset. The CRC calculation starts at the byte pointed at by the memory address and is done on the 'Data Length' number of bytes in the volatile memory. The result is stored in the Data Buffer at the Data Buffer Offset index. The result is a 16-bit unsigned value with least significant byte first. The CRC calculation code is shown below.

```
static const uint16_t CCITTCRCTable [256] = {
0x0000, 0x1021, 0x2042, 0x3063, 0x4084, 0x50a5,
0x60c6, 0x70e7, 0x8108, 0x9129, 0xa14a, 0xb16b,
0xc18c, 0xd1ad, 0xe1ce, 0xf1ef, 0x1231, 0x0210,
0x3273, 0x2252, 0x52b5, 0x4294, 0x72f7, 0x62d6,
0x9339, 0x8318, 0xb37b, 0xa35a, 0xd3bd, 0xc39c,
0xf3ff, 0xe3de, 0x2462, 0x3443, 0x0420, 0x1401,
0x64e6, 0x74c7, 0x44a4, 0x5485, 0xa56a, 0xb54b,
0x8528, 0x9509, 0xe5ee, 0xf5cf, 0xc5ac, 0xd58d,
0x3653, 0x2672, 0x1611, 0x0630, 0x76d7, 0x66f6,
0x5695, 0x46b4, 0xb75b, 0xa77a, 0x9719, 0x8738,
0xf7df, 0xe7fe, 0xd79d, 0xc7bc, 0x48c4, 0x58e5,
0x6886, 0x78a7, 0x0840, 0x1861, 0x2802, 0x3823,
0xc9cc, 0xd9ed, 0xe98e, 0xf9af, 0x8948, 0x9969,
0xa90a, 0xb92b, 0x5af5, 0x4ad4, 0x7ab7, 0x6a96,
0x1a71, 0x0a50, 0x3a33, 0x2a12, 0xdbfd, 0xcbdc,
0xfbbf, 0xeb9e, 0x9b79, 0x8b58, 0xbb3b, 0xab1a,
0x6ca6, 0x7c87, 0x4ce4, 0x5cc5, 0x2c22, 0x3c03,
0x0c60, 0x1c41, 0xedae, 0xfd8f, 0xcdec, 0xddcd,
0xad2a, 0xbd0b, 0x8d68, 0x9d49, 0x7e97, 0x6eb6,
0x5ed5, 0x4ef4, 0x3e13, 0x2e32, 0x1e51, 0x0e70,
0xff9f, 0xefbe, 0xdfdd, 0xcffc, 0xbf1b, 0xaf3a,
0x9f59, 0x8f78, 0x9188, 0x81a9, 0xb1ca, 0xa1eb,
0xd10c, 0xc12d, 0xf14e, 0xe16f, 0x1080, 0x00a1,
0x30c2, 0x20e3, 0x5004, 0x4025, 0x7046, 0x6067,
0x83b9, 0x9398, 0xa3fb, 0xb3da, 0xc33d, 0xd31c,
0xe37f, 0xf35e, 0x02b1, 0x1290, 0x22f3, 0x32d2,
0x4235, 0x5214, 0x6277, 0x7256, 0xb5ea, 0xa5cb,
0x95a8, 0x8589, 0xf56e, 0xe54f, 0xd52c, 0xc50d,
0x34e2, 0x24c3, 0x14a0, 0x0481, 0x7466, 0x6447,
0x5424, 0x4405, 0xa7db, 0xb7fa, 0x8799, 0x97b8,
0xe75f, 0xf77e, 0xc71d, 0xd73c, 0x26d3, 0x36f2,
0x0691, 0x16b0, 0x6657, 0x7676, 0x4615, 0x5634,
0xd94c, 0xc96d, 0xf90e, 0xe92f, 0x99c8, 0x89e9,
0xb98a, 0xa9ab, 0x5844, 0x4865, 0x7806, 0x6827,
0x18c0, 0x08e1, 0x3882, 0x28a3, 0xcb7d, 0xdb5c,
0xeb3f, 0xfb1e, 0x8bf9, 0x9bd8, 0xabbb, 0xbb9a,
0x4a75, 0x5a54, 0x6a37, 0x7a16, 0x0af1, 0x1ad0,
0x2ab3, 0x3a92, 0xfd2e, 0xed0f, 0xdd6c, 0xcd4d,
0xbdaa, 0xad8b, 0x9de8, 0x8dc9, 0x7c26, 0x6c07,
0x5c64, 0x4c45, 0x3ca2, 0x2c83, 0x1ce0, 0x0cc1,
0xef1f, 0xff3e, 0xcf5d, 0xdf7c, 0xaf9b, 0xbfba,
0x8fd9, 0x9ff8, 0x6e17, 0x7e36, 0x4e55, 0x5e74,
0x2e93, 0x3eb2, 0x0ed1, 0x1ef0 };

static uint16_t GetCCITTCRC(const uint8_t* Data, uint32_t Size) {
uint16_t CRC;
uint16_t Temp;
uint32_t Index;

if (Size == 0) {
return 0;
}

CRC = 0xFFFF;

for (Index = 0; Index < Size; Index++){
Temp = (uint16_t)( (CRC >> 8) ^ Data[Index] ) & 0x00FF;
CRC = CCITTCRCTable[Temp] ^ (CRC << 8);
}

return CRC;
}
```

5.4.26 Copy Data (0x1A)

Command Number	0x1A		
Command Name	Copy Data		
Valid Tag Types	-		
Argument Offset [bytes]	0x00	0x02	0x02
Argument Name	Destination Address	Source Address	Data Length
Argument Size [bytes]	0x02	0x02	0x02
Argument Description	The byte address in the memory to where the data is to be copied.	The byte address in the memory from where the data is to be copied.	Size of the copied data in bytes.

Table 5.4.26

The Copy Data command copies data around inside the RFID Module memory. This command takes as arguments the Destination Address, the Source Address and the Data Length in bytes.

5.4.27 Unlock (0x1B)

Command Number	0x1B
Command Name	Unlock
Valid Tag Types	-
Argument Offset [bytes]	0x00
Argument Name	Password
Argument Size [bytes]	0x08
Argument Description	8-byte long password with the least significant byte first.

Table 5.4.27

The Unlock command takes as an argument an 8-bytes long password. If the password matches the actual password in the module the memory which contains:

- AES Initialization Vectors
- AES Encryption Keys
- Authentication Keys and Passwords
- User Memory
- Password

will be unlocked and made accessible for read and write operations.

5.4.28 Lock (0x1C)

This Function takes no arguments. If the device is unlocked and the user executes the Lock command, protected memory which contains:

- AES Initialization Vectors
- AES Encryption Keys
- Authentication Keys and Passwords
- User Memory
- Password

will be saved to non-volatile memory and locked (direct access will be blocked).

5.4.29 Get Module Version (0x1D)

The Get Module Version command takes no arguments. After execution, the Data Buffer at index 0x00 is filled with a NULL-ended ASCII string which describes the hardware and firmware version of the module.

5.4.30 Reset to Default (0x1E)

The Reset to Default command doesn't take any arguments. It resets all memory including protected memory to factory default settings. All locked information is lost.

5.4.31 Enumerate Tags UID (0x1F)

The Enumerate Tags UID command doesn't take any arguments. This command scans for all tags within the range of the antenna and enumerates their UIDs in the Data Buffer. The first byte in the Data Buffer says how many tags were discovered. If there are no tags in the range, then the Result Register is updated with 0x08 (Tag is not present) value. After enumeration, if there is at least one tag in the field, the Result Register is updated with 0x00 (No Error) value. Information about all enumerated tags is stored in the Data Buffer, starting from the second byte of the buffer. There are as many records in the Data Buffer as there were tags found. Each record contains the Tag UID Size as the first byte and the Tag UID is stored in the rest of the bytes in the record. The record length is variable and equal to the UID size plus one byte.

5.4.32 Enumerate Tags UID and Type (0x20)

The Enumerate Tags UID and Type command doesn't take any arguments. This command scans for all tags within the range of the antenna and enumerates their UID and type in the Data Buffer. The first byte in the Data Buffer says how many tags were discovered. If there are no tags in the range, then the Result Register is updated with 0x08 (Tag is not present) value. After enumeration, if there is at least one tag in the field the Result Register is updated with 0x00 (No Error) value. Information about all enumerated tags is stored in the Data Buffer starting from the second byte of the buffer. There are as many records in the Data Buffer as there were tags found. Each record contains the Tag Type as the first byte, the Tag UID Size as the second byte and the Tag UID is stored in the rest of the bytes in the record. The record length is variable and equal to the UID size plus two bytes.

5.4.33 Select Tag (0x21)

Command Number	0x21	
Command Name	Select Tag	
Valid Tag Types	-	
Argument Offset [bytes]	0x00	0x01
Argument Name	UID Size	UID
Argument Size [bytes]	0x01	0x04, 0x07, 0x0A
Argument Description	Size in byte of the UID which follows this parameter.	UID of the tag to select. The size of the UID is determined by previous parameter and cannot be other than 4, 7 or 10 bytes.

Table 5.4.33

The Select Tag command takes as arguments the UID Size in bytes and the UID bytes themselves. If the command execution is successful, then the addressed tag is selected, and the Result Register is updated with 0x00 (No Error) value.

5.4.34 Polling (0x22)

Command Number	0x22								
Command Name	Polling								
Valid Tag Types	All								
Argument Offset [bytes]	0x00	0x01	0x02	0x03	0x04	0x05	0x06-0x08	0x09	…
Argument Name	Period [100ms]	Number of Defined Tags	Defined Tag: Asynchronous Packet Mode	Defined Tag: IO Config	Defined Tag: PWM	Defined Tag: PWM Duty	Defined Tag: PWM Period [uS]	Defined Tag: Tag Timeout [100ms]	
Argument Size [bytes]	0x01	0x01	0x01	0x01	0x01	0x01	0x03	0x01	
Argument Description	Pooling period = value x 100mS	Number of defined Tags in User Memory from Index 19. Syntax: <UIDSize><UID>…	0x00 - Do not sent any packet 0x01 - Send Async Packet if detected Tag is on list 0x02 - Send Binary serial packet. Syntax: <UIDSize><UID> 0x03 - Send String serial packet. Syntax: <UIDSize><UID><\r\n>	0b0000xxxx - do not use IO outputs 0b0001xxxS - Set IO0 to S State if Tag is on list 0b0010xxSx - Set IO1 to S State if Tag is on list 0b0100xSxx - Set IO2 to S State if Tag is on list 0b1000Sxxx - Set IO3 to S State if Tag is on list	0x00 - Use PWM on IO0 0x01 - Use PWM on IO1 0x02 - Use PWM on IO2 > 0x02 - Do not use PWM	0% < Duty < 100%	5uS < Period < 3120761uS	Delay of next Tag detection	

Table 5.4.34.1 Table continuation on next page.

Polling Period – Delay between polling attempts for detection of tag presence.
Number Of Defined Tags – States how many UIDs are stored in user memory.
Defined Tag: Asynchronous Packet Mode – Defines in which format UID will be sent via UART interface

- Async – Module sends Asynchronous Packet with UID information. See chapter 5.6 for details.
- Binary – Module sends <UID0Size><UID0>…<UIDnSize><UIDn> as binary values. If Tag is on defined list then only one UID is sent.
- String – Module sends UIDs in string format. If Tag is on defined list, then only one UID is sent.

Defined Tag: IO Config – Bits 4 to 7 defines which IO is used. Bits 0 to 3 defines state on defined Tag presence. See Table 5.4.32.3 for details.
Defined Tag: PWM - Defines which PWM channel will be used. Value > 2 means that PWM is not used.
Defined Tag: PWM Duty – Duty.
Defined Tag: PWM Period – PWM frequency = 1 / Period[uS].
Defined Tag: Tag Timeout – T = Value x 100mS. Delay between Tag detection and polling restart.

…	Command Number	0x22					
	Command Name	Polling					
	Valid Tag Types	All					
	Argument Offset [bytes]	0x0A	0x0B	0x0C	0x0D	0x0E-0x10	0x11
	Argument Name	Undefined Tag: Asynchronous Packet Mode	Undefined Tag: IO Config	Undefined Tag: PWM	Undefined Tag: PWM Duty	Undefined Tag: PWM Period [uS]	Undefined Tag: Tag Timeout [100ms]
	Argument Size [bytes]	0x01	0x01	0x01	0x01	0x03	0x01
	Argument Description	0x00 - Do not sent any packet 0x01 - Send Async Packet if detected Tag is not on list 0x02 - Send Binary serial packet. Syntax: <UIDSize><UID><UIDSize><UID>… 0x03 - Send String serial packet. Syntax: <UIDSize><UID><\r\n\><UIDSize><UID><\r\n>…	0b0000xxxx - do not use IO outputs 0b0001xxxS - Set IO0 to S State if Tag is on list 0b0010xxSx - Set IO1 to S State if Tag is on list 0b0100xSxx - Set IO2 to S State if Tag is on list 0b1000Sxxx - Set IO3 to S State if Tag is on list	0x00 - Use PWM on IO0 0x01 - Use PWM on IO1 0x02 - Use PWM on IO2 > 0x02 - Do not use PWM	0% < Duty < 100%	5uS < Period < 3120761uS	Delay of next Tag detection

Table 5.4.34.2

Undefined Tag: Asynchronous Packet Mode – Defines in which format the UID will be sent via the UART interface:

- Async – The module sends an Asynchronous Packet with UID information. See chapter 5.6 for details.
- Binary – The module sends <UID0Size><UID0>…<UIDnSize><UIDn> as binary values. If the Tag is on the defined list, then only one UID is sent.
- String – The module sends UIDs in string format. If the tag is on the defined list, then only one UID is sent.

Undefined Tag: IO Config – Bits 4 to 7 defines which IO is used. Bits 0 to 3 defines the state on defined Tag presence. See Table 5.4.32.3 for details.
Undefined Tag: PWM - Defines which PWM channel will be used. Value > 2 means that PWM is not used.
Undefined Tag: PWM Duty – Duty.
Undefined Tag: PWM Period – PWM frequency = 1 / Period[uS].
Undefined Tag: Tag Timeout – T = Value x 100mS. Delay between Tag detection and polling restart.

Parameter name	IO Config							
Parameter Size	0x01							
Bit Number	7	6	5	4	3	2	1	0
Name	IO3 enable	IO2 enable	IO1 enable	IO0 enable	IO3 Active state	IO2 Active state	IO1 Active state	IO0 Active state
Argument Description	If set to 1 then IO3 if set to output. State is defined by bit 3 (IO3 Active state). IF set to 0 then IO3 is not used.	If set to 1 then IO2 if set to output. State is defined by bit 2 (IO2 Active state). IF set to 0 then IO2 is not used.	If set to 1 then IO1 if set to output. State is defined by bit 1 (IO1 Active state). IF set to 0 then IO1 is not used.	If set to 1 then IO0 if set to output. State is defined by bit 0 (IO0 Active state). IF set to 0 then IO0 is not used.	If IO3 enable is set to 1 then output is set to value of this bit.	If IO2 enable is set to 1 then output is set to value of this bit.	If IO1 enable is set to 1 then output is set to value of this bit.	If IO0 enable is set to 1 then output is set to value of this bit.

Table 5.4.34.3

The Polling command takes arguments listed in the above tables. If the command execution is successful, then the Result Register is updated with 0x00 (No Error) value. For more information about the Polling command, please see chapter 5.5 Polling Mode.

5.5 Polling Mode

This command enables Tag UID polling. In this mode the module executes the continuous repeated Enumerate Tags UID command, writes the Command Polling value to the Command Register and parameters to the Parameter Registers to start polling. Parameters are described in 5.4.32. Polling mode can be stopped by writing any RFID command to the Command Register.

The SELECT TAG command must be executed to activate the detected Tag for the communication.

5.5.1 Polling internal procedure

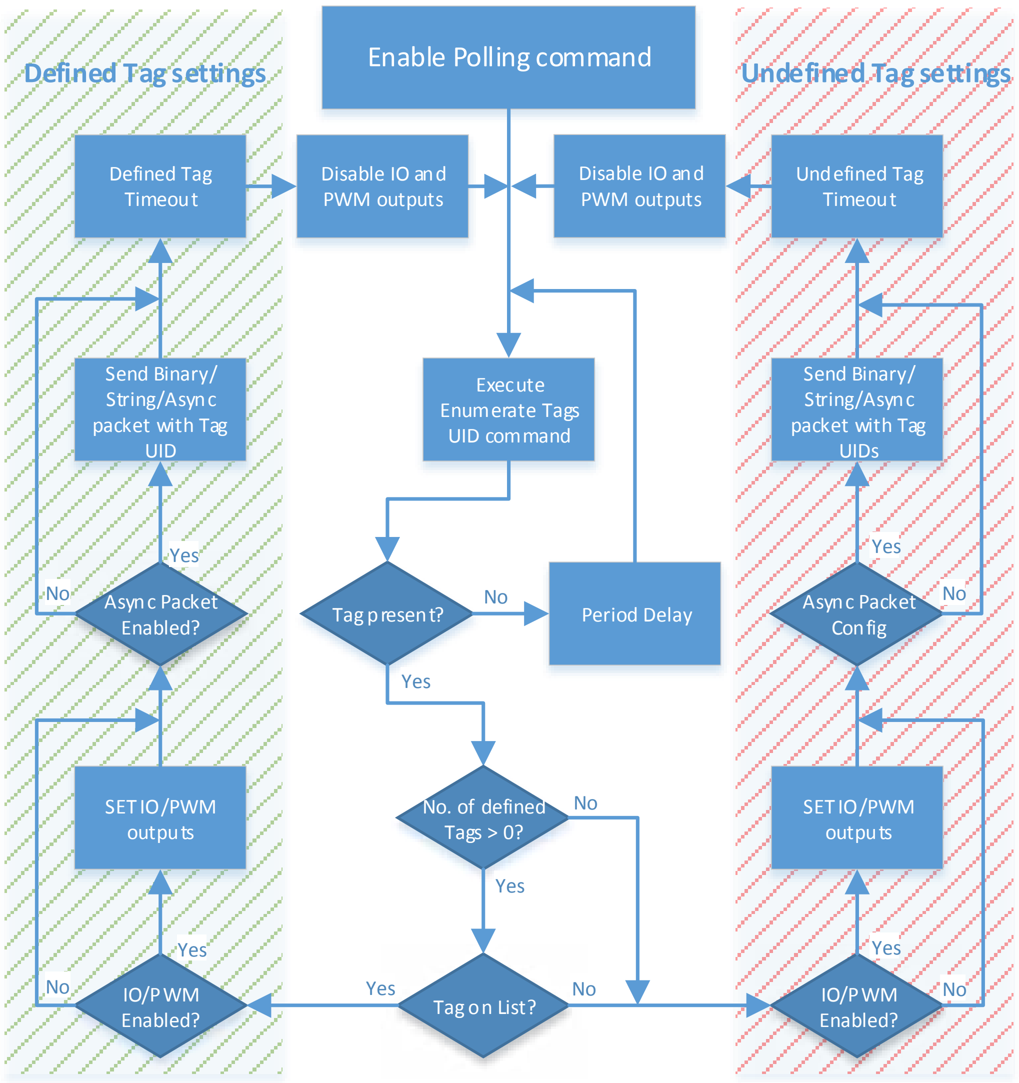


Figure 5.5.1

5.5.2 Polling auto start

Polling can be automatically started after power-up or reset. This functionality requires that a command and parameters be written to User Memory from index 0. If Polling mode is enabled at module start up, then a System Start packet is not sent. More information can be found in chapter 5.5.6. Polling mode can be stopped by writing any RFID command to the Command Register. Access to User Memory must be unlocked by the Unlock command before writing data. Data in the User Memory is saved after a Lock command.

5.5.3 Polling use simple example – Undefined Tags

This example of polling command use demonstrates how to configure simple signalling of any Tag presence. Command Polling followed by parameters:

Argument Offset [bytes]	0x00	0x01	0x02	0x03	0x04	0x05	0x06-0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E-0x10	0x11
Argument Name	Period [100ms]	Number of Defined Tags	Defined Tag: Asynchronous Packet Mode	Defined Tag: IO Config	Defined Tag: PWM	Defined Tag: PWM Duty	Defined Tag: PWM Period [uS]	Defined Tag: Tag Timeout [100ms]	Undefined Tag: Asynchronous Packet Mode	Undefined Tag: IO Config	Undefined Tag: PWM	Undefined Tag: PWM Duty	Undefined Tag: PWM Period [uS]	Undefined Tag: Tag Timeout [100ms]
Argument Size [bytes]	0x01	0x01	0x01	0x01	0x01	0x01	0x03	0x01	0x01	0x01	0x01	0x01	0x03	0x01
Argument Value	0x01	0x00	x	x	x	x	x	x	0x03	0x22	0x00	0x32	0x003E8	0x32
Argument Description	Polling period every 100mS	No Defined Tags in User Memory	Parameter ignored	Parameter ignored	Parameter ignored	Parameter ignored	Parameter ignored	Parameter ignored	Send Tag UID in string format	Set IO1 to High state	Enable PWM on IO0	50%	1ms - 1kHz	Wait for 5 seconds

Table 5.5.3

The module will check for Tag presence every 100ms. If a tag is detected, then IO1 is set to the High state, PWM on IO0 is set to 50%, 1kHz and detected Tag(s) UIDSize and UID are send to the UART in string format. After 5 seconds the PWM is disabled, IO1 is set to the Low state and the module checks for Tag presence once again.

5.5.4 Polling use simple example – Defined Tags

This example of a polling command use shows how to configure simple signalling of only defined Tag presence. Command Polling followed by parameters:

Argument Offset [bytes]	0x00	0x01	0x02	0x03	0x04	0x05	0x06-0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E-0x10	0x11
Argument Name	Period [100ms]	Number of Defined Tags	Defined Tag: Asynchronous Packet Mode	Defined Tag: IO Config	Defined Tag: PWM	Defined Tag: PWM Duty	Defined Tag: PWM Period [uS]	Defined Tag: Tag Timeout [100ms]	Undefined Tag: Asynchronous Packet Mode	Undefined Tag: IO Config	Undefined Tag: PWM	Undefined Tag: PWM Duty	Undefined Tag: PWM Period [uS]	Undefined Tag: Tag Timeout [100ms]
Argument Size [bytes]	0x01	0x01	0x01	0x01	0x01	0x01	0x03	0x01	0x01	0x01	0x01	0x01	0x03	0x01
Argument Value	0x01	0x05	0x03	0x40	0x01	0x32	0x0007D0	0x64	0x00	0x00	0x03	X	x	0x01
Argument Description	Polling period every 100mS	Five Defined Tags in User Memory	Send Tag UID in string format	Set IO3 to Low state	Enable PWM on IO1	50%	500Hz	Wait for 10 seconds	Disable	Do not use IO	Do not use PWM	Parameter ignored	Parameter ignored	Wait for 100mS

Table 5.5.4

The module will check for Tag presence every 100ms. If a tag is detected and this tag is on the defined list in user memory then IO3 is set to the Low state, PWM on IO1 is set to 50%, 500Hz and the detected Tag UIDSize and UID are send to the UART in string format. After 10 seconds the PWM is disabled, IO3 is set to the Low state and the module checks for Tag presence once again. The module does not react (only waits 100mS) if the detected Tag is not on list. Information of how to prepare a Defined Tag List can be found in chapter 5.5.6.

5.5.5 Polling use simple example – Undefined and Defined Tags

This example of polling command use shows how to configure simple signalling of defined and undefined Tag presence. Command Polling followed by parameters:

Argument Offset [bytes]	0x00	0x01	0x02	0x03	0x04	0x05	0x06-0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E-0x10	0x11
Argument Name	Period [100ms]	Number of Defined Tags	Defined Tag: Asynchronous Packet Mode	Defined Tag: IO Config	Defined Tag: PWM	Defined Tag: PWM Duty	Defined Tag: PWM Period [uS]	Defined Tag: Tag Timeout [100ms]	Undefined Tag: Asynchronous Packet Mode	Undefined Tag: IO Config	Undefined Tag: PWM	Undefined Tag: PWM Duty	Undefined Tag: PWM Period [uS]	Undefined Tag: Tag Timeout [100ms]
Argument Size [bytes]	0x01	0x01	0x01	0x01	0x01	0x01	0x03	0x01	0x01	0x01	0x01	0x01	0x03	0x01
Argument Value	0x01	0x0A	0x01	0xA1	0x01	0x32	0x0007D0	0x64	0x01	0x41	0x03	0x32	0x0001F4	0x1E
Argument Description	Polling period every 100mS	Ten Defined Tags in User Memory	Send Tag UID as B1 Async Packet	Set IO0 to High state and IO3 to Low state	Enable PWM on IO1	50%	500Hz	Wait for 10 seconds	Send Tag UID as B1 Async Packet	Set IO2 to High state.	Enable PWM on IO1	50%	2kHz	Wait for 3 seconds

Table 5.5.5

The module will test for Tag presence every 100ms. If a tag is detected and this tag is on the defined list in the user memory, then IO0 is set to the high state, IO3 is set to the low state, PWM on IO1 is set to 50%, 500Hz and detected Tag UIDSize and UID are send to the UART in Async Packet B1 format. After 10 seconds, the PWM is disabled, IO0 is set to the high state, IO3 is set to the low state and the module checks for tag presence once again. If the tag is not on the defined List, then IO2 is set to the high state, PWM on IO1 is set to 50%, 2kHz and the detected Tag(s) UIDSize and UID are send to the UART in Async Packet B1 format. After 3 seconds the PWM is disabled, IO2 is set to the low state and the module checks for Tag presence once again. Information of how to prepare a defined Tag List can be found in chapter 5.5.6.

5.5.6 Defined Tag List

The defined tag list is stored in User Memory from index 19. UIDSize followed by UID must be written for every listed tag.

Address hex	0x13	0x14	0x15	0x16	0x17	0x18	0x19	0x1A	0x1B	0x1C	0x1D	0x1E	0x1F	…
Address dec	19	20	21	22	23	24	25	26	27	28	29	30	30	
Tag number	0					1								…
Value type	**UIDSize**	UID[0]	UID[1]	UID[2]	UID[3]	**UIDSize**	UID[0]	UID[1]	UID[2]	UID[3]	UID[4]	UID[5]	UID[6]	…
Value	**0x04**	0x00	0x01	0x02	0x03	**0x07**	0xAA	0xBB	0xCC	0xDD	0xEE	0xFF	0x32	

Table 5.5.6

If any of UIDSize contains an incorrect value (allowed values are 4, 7 and 10) then the module stops comparing the detected UID to the list. The number of Tags on the list is defined by the 'Number of Defined Tags' polling command parameter. Access to User Memory must be unlocked by the Unlock command before writing data. Data in User Memory is saved after a Lock command.

5.5.7 Polling stop and communication with tag example.

To communicate with one of the detected tags user must send Select Tag command followed by UID.

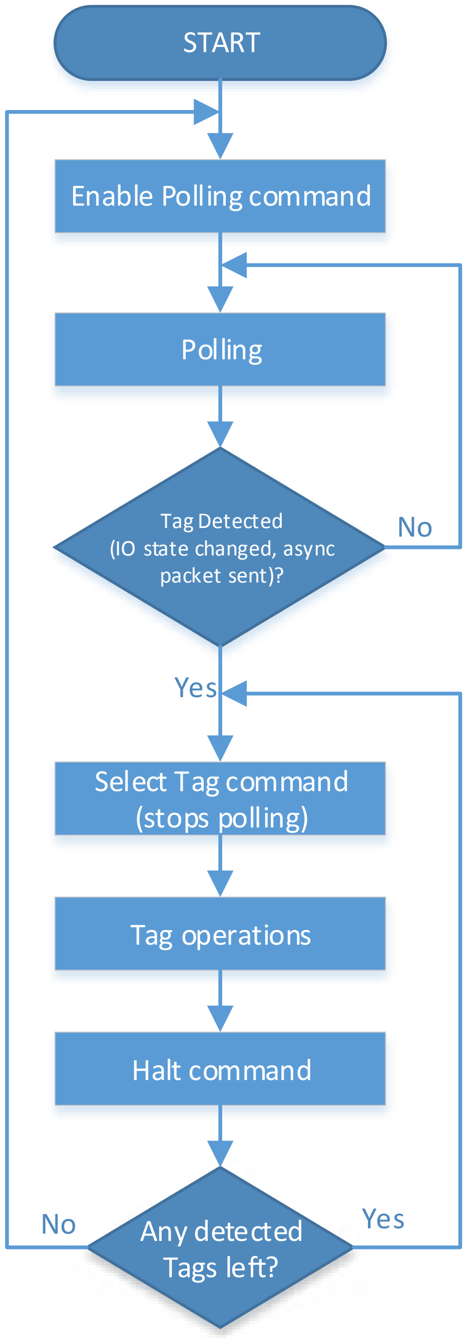


Figure 5.5.7 Polling stop and Tag r/w example.

5.6 Asynchronous Packets

The B1 Module can send asynchronous packets. These packets are sent when system interrupts are generated. The data consist of a response byte equal to 0x08 and one or more parameter bytes, which are defined below.

Byte Name	Asynchronous Packet Parameter at index 0x00								First detected tag UID size	First detected tag UID	…
Byte Offset in Packet Parameters	0x00								0x01	0x02 - 0x02+UIDSize	…
Bit Position	7	6	5	4	3	2	1	0	Sent only if Polling Event has been occurred.		
Bit Meaning	-	Polling event	RFID Command End	Comparator Output Pin State Change	IO3 Edge	IO2 Edge	IO1 Edge	IO0 Edge			

Table 5.6

Bits 0 to 7 in the parameter byte are set when a given interrupt is generated as per the configuration. Interrupts related with IO edges, polling event and comparator output pin state change are configurable and can be turned off. The only permanently enabled interrupt is that generated by an RFID command end. Polling interrupt can be configured in Polling command.

5.7 Sleep Mode

Sleep mode functionality is provided by a dedicated "Enter Sleep Mode" command (0x03). The module replies to this command with an ACK response and no parameters. The module can be woken from Sleep Mode by sending a 0x00 byte. After waking up the module sends ACK packet response to inform the user it is ready to receive new commands. In comparison to the Power Down Mode, the module does not need to restart after waking up from Sleep Mode. In Sleep Mode, the current consumption is reduced. The communication system and PWM are also turned off. If enabled, the IDAC and the comparator still operate (resulting in a higher current consumption). The user will still receive asynchronous packets from the comparator and IOs- the module will wake up, send the required packets and then re-enter Sleep Mode.

5.8 Power Down Mode

The Power Down functionality is provided via the nPWRDN pin. This pin is configured as an input without any pull-up or pull-down resistors, thus it must be driven by the user. For normal operation, this pin should be connected to VCC or driven high. After the user drives this pin low the system prepares for going into Power Down Mode and drives nSLEEP line low just before entering this state. To wake up the module the nPWRDN pin must be driven high. The module will restart and drive the nSLEEP line low to inform the user that it is starting the system again. When start-up is completed, the nSLEEP line is driven high and System Start packet is sent. During the restart, none of the volatile memory content is retained.

The reaction time of the system will vary depending upon the active configuration and the following:

- Communication – if the line is pulled low whilst there is ongoing communication, then the Power Manager will wait until communication is finished before going into the Power Down Mode.
- Command execution – if the line is pulled low whilst the system is executing a command, then the Power Manager will wait until command execution is finished before going into the Power Down Mode.

When the module is in the Power Down Mode, all systems are turned-off and all IOs are in a high impedance state. The module won't respond to any signals or communication coming from outside.

5.9 Memory Locking

The module has an option to lock its memory using an 8-byte long password. The default state of the module after power-up is locked and the default password is all bytes equal to 0x00. The user cannot see the memory content when the module is locked, and any write to the memory will be discarded. During the unlock procedure, the content of the non-volatile memory is copied to the volatile memory and is available for reading and writing. Changes made in volatile memory are updated in non-volatile memory during the lock procedure. Commands using data from this memory range use the volatile memory when the module is unlocked and non-volatile memory when the module is locked.

5.10 nSLEEP Pin

The B1 module gives the user an option to put it into one of two low power modes. These states are Sleep Mode (via a UART command) and Power Down Mode (via the nPWRDN input pin). The nSLEEP output pin is an active low line indicating that the module is in Sleep Mode or Power Down Mode.

5.11 Reset to Defaults

The B1 module gives the user an option to reset to defaults the entire memory in case of any problems with the AES Encryption blocking any communication with the module. If the user holds the nPWRDN pin low during a reset (not power up) of the module, it will reset the entire memory to its default value.

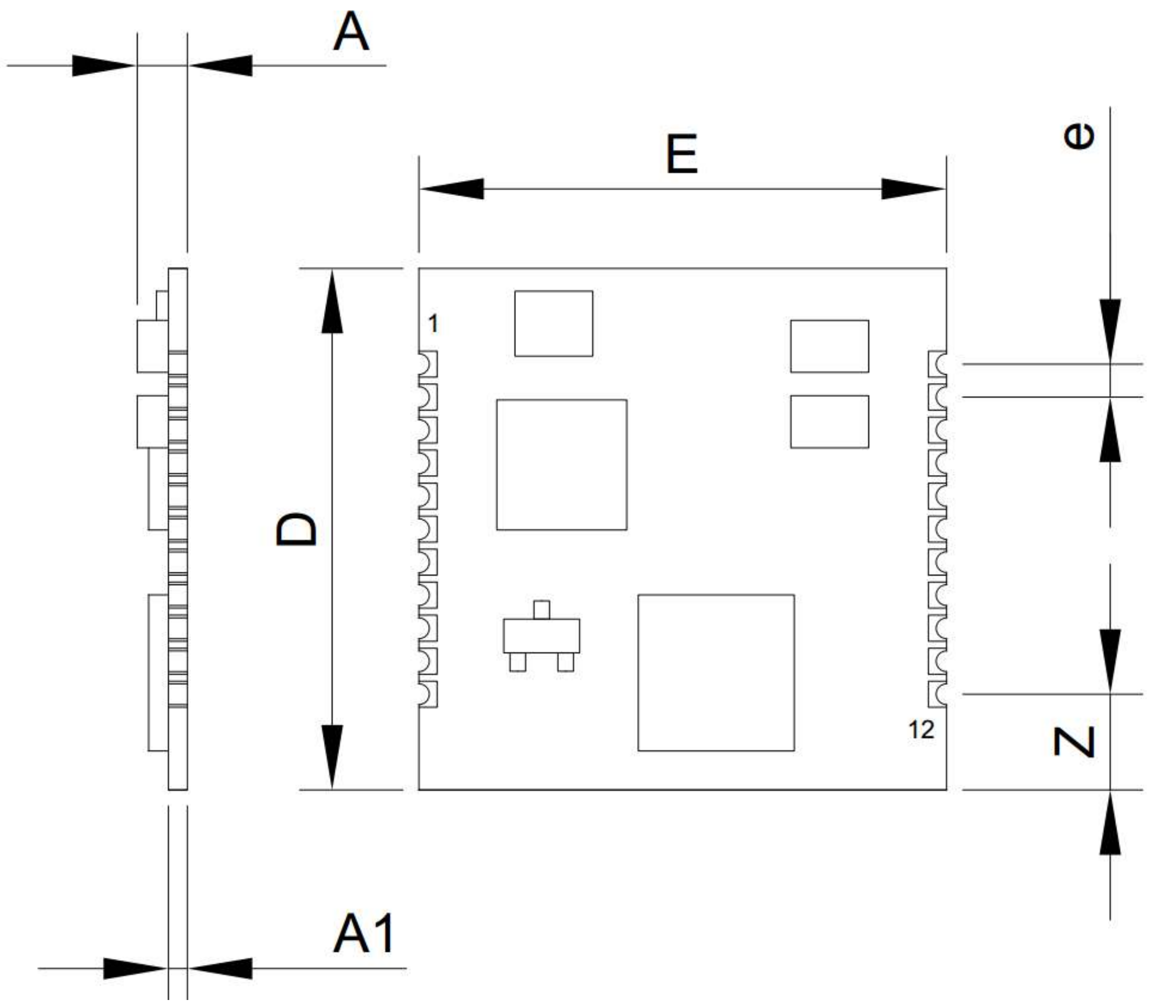
Reset to Defaults step by step procedure:

- Power up module (RST high, nPWRDN high)
- Wait for 50ms
- Set RST and nPWRDN to low
- Wait for 50ms
- Set RST to high

- Wait for 50ms
- Set nPWRDN to high

6 Mechanical

6.1 Dimensions

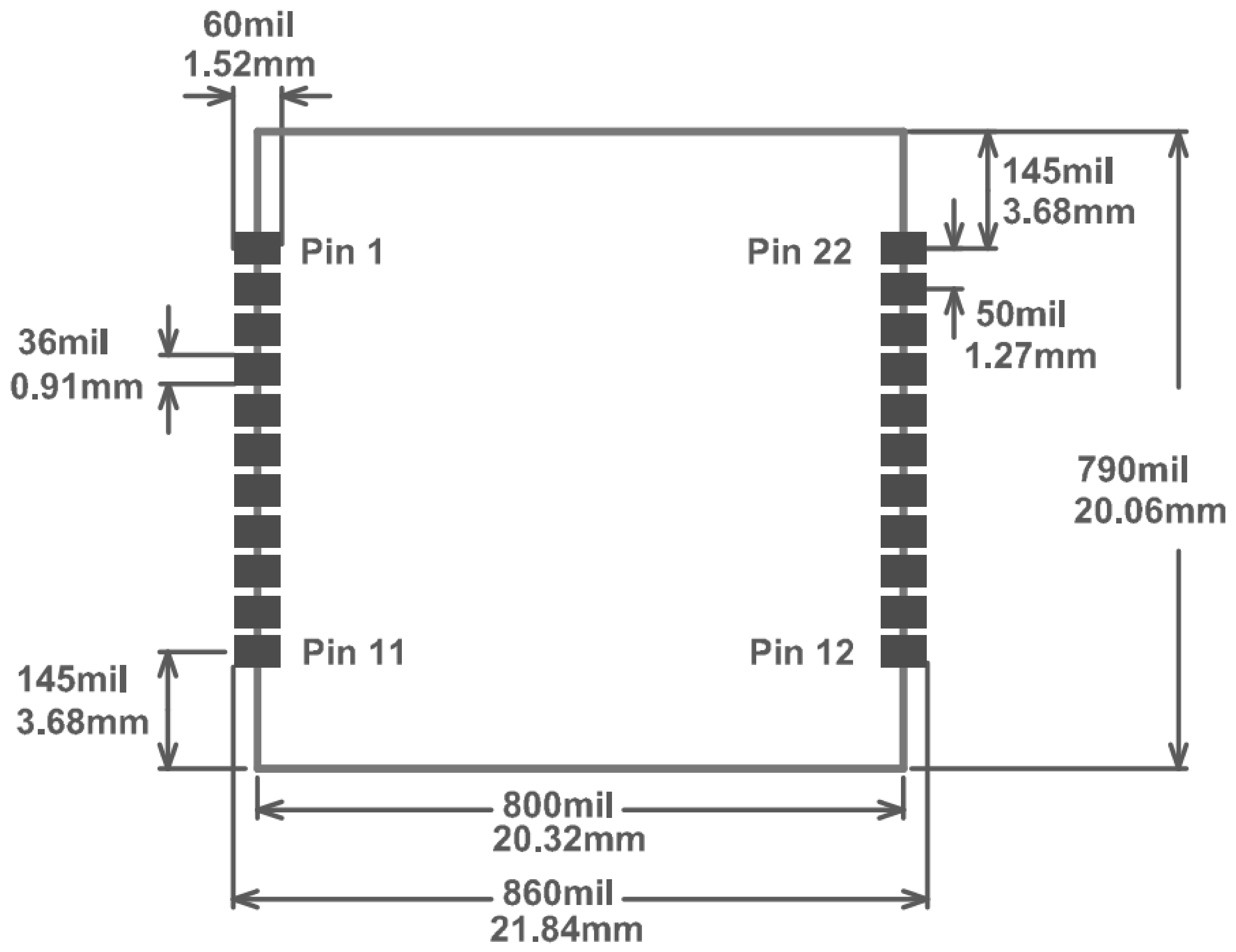


Drawing 6.1

UNIT	A	A1	D	E	e	Z
mm	2	0.7	20.06	20.32	1.27	3.68
inches	0.079	27.56	0.79	0.8	0.05	0.145

Table 6.1

6.2 Recommended Footprint



Drawing 6.1

7 Errata

7.1 IDAC

Due to an undefined shut-down state of the IDAC, powered devices that do not use the IDAC continuously might experience some degradation in the current output over the lifetime of the device. The degradation is very small when the device is used at room temperature, but the output current will fall well outside specs if the device is exposed to higher temperatures for longer periods of time.

If the IDAC output current stability is crucial to the application, the IDAC should never be completely disabled while the device is powered. Leaving the IDAC enabled in the lowest output code setting with duty-cycling enabled consumes 50 nA of extra current and eliminates the problem.

7.2 UART Packet Sizes

Due to limited processing power, maximum packet size which the module can process at once is limited above baud rate 230400. Maximum packet size which can be received and processed at those baud rates is 768 bytes in total. Any packet with larger size can cause buffer overflow.

No responsibility is taken for the method of integration or final use of the RFID B1 module

More information about the B1 module and other products can be found at the Internet site:

http://www.eccel.co.uk

or alternatively contact ECCEL Technology (IB Technology) by e-mail at:

sales@eccel.co.uk