USER GUIDE

AVR2051: SerialNet

Introduction

SerialNet is an embedded firmware that provides wireless mesh-network control for Atmel® 802.15.4 wireless platforms via ASCII-based AT commands that can be sent from a host controller through a serial interface.

Using SerialNet requires no embedded API-based programming and thus significantly simplifies product prototyping and development.

Features

- Simple AT-command control interface
- Full mesh-routing support
- Sleeping nodes support
- Large-scale network operation
- Secure communication
- Remote node control
- Auto-networking feature

Table of Contents

1 Overview

SerialNet is based on the AT command protocol, which is widely used in embedded networking systems due to its simplicity, textual parameter representation, and inherent flexibility. This chapter gives a brief introduction into the concept of the SerialNet protocol, lists hardware platforms SerialNet is available for, and describes conventions used throughout the document.

1.1 Supported Platforms

Hardware platforms supported by SerialNet are listed in Table 1-1.

Table 1-1. Supported Hardware Platforms

Name in this document	Platform (MCU + RF)	Modules / Dev boards
MEGARFR2-ZIGBIT®	ATmega256RFR2	ATZB-S1-256-3-0-C [1] ATZB-S1-256-3-0-U [2]
MEGARFR2-ZIGBIT-XPRO	ATmega256RFR2	ATZB-256RFR2-XPRO [3]
MEGARFR2-XPRO	ATmega256RFR2	ATmega256RFR2-XPRO [4]

Most of the SerialNet commands are hardware independent and can be executed on all supported platforms. However, a few commands either exhibit platform-specific behavior or are supported on particular hardware platforms only. For such cases, command descriptions given in the Chapter 5 Command Descriptions provide corresponding differences in the command functionality for various platforms. If no reference to a platform is given in a command description, then platform-independence is implied.

1.2 Firmware Images

SerialNet firmware images can be found in the `Firmware/` directory inside the SerialNet package. This directory is structured by platform family. Currently, only the ATmega256RF2 family is supported and its directory structure is described in Section 1.2.1 ATmega256RFR2 Firmware Images. The meaning of common naming extensions in firmware files is given in Table 1-2.

Table 1-2. Naming Extensions in SerialNet Firmware Images

Naming Convention in Firmware Images	Description
`_UART1 / _UART0`	Indicates what serial interface is configured in the image for use of AT commands and serial bootloader (if part of the image). Note that if a bootloader image and a SerialNet image are loaded separately they can have different configurations for supported serial interface.
`_NoSec /_StdSec`	Indicates what security type is used in the image. `_StdSec` stands for standard security, `_NoSec` stands for no security.

Naming Convention in Firmware Images	Description
`_defaultBL /_SerialBL / _NoBL`	Indicates what type of serial bootloader is compatible with the image file. `_defaultBL` means that the image works only with default bootloader already pre-programmed by Atmel during module manufacturing. Hence this option is only present for files in `.srec` format that should be loaded via the bootloader interface [12], [13]. `_SerialBL` means that the image works only with AVR2054 Serial bootloader that is either already included as part of the image file (for `.hex` and `.elf` formats) or can be programmed to the module by the user upfront via JTAG/ISP. `_NoBL` means that the image requires no bootloader present on the device. Hence this option is available only for for `.hex` and `.elf` formats that need to be programmed via JTAG/ISP.

1.2.1 ATmega256RFR2 Firmware Images

Firmware images for ATmega256RFR2 modules are located in `Firmware/Atmega256RFR2` directory as described in Table 1-3.

Note: ATZB-S1-256-3-0-C/U modules ([1], [2]) are by default shipped preprogrammed with an embedded bootloader FW that uses the same serial protocol as AVR2054 Bootloader [13]. However both bootloader versions have differences in internal organization and behavior. To ensure that storing and restoring of persistent network data works correctly, different SerialNet images should be used depending on the version of the bootloader present on the device. Corresponding firmware images are located in different folders as described in Table 1-3 and also differentiated by `_defaultBL /_SerialBL` suffix in the file name as described in Table 1-2.

Table 1-3. Sub-directories in Firmware/ATmega256RF2/

Directory Name	Description
`/AVR2054_Bootloader_v3.2.0/`	Contains FW images of the latest AVR2054 bootloader version that has been verified to work correctly with SerialNet (e.g. v3.2.0). Images shall be loaded via JTAG/ISP interface and require that fuse bytes are set as *Ext: 0xFE; High: 0x18; Low: 0x62.* The images have following settings for the serial interface: • *Baudrate: 38400; Data bits: 8; Parity: None; Stop bits: 1; Flow ctrl: None* (1)
`/SerialNet_RFR2_ZigBit_Legacy/`	Contains SerialNet FW images in `.srec` format intended for use on legacy ATZB-S1-256-3-0-C/U modules ([1], [2]) that are preprogrammed with default bootloader during module manufacturing. Images can be loaded only via serial bootloader interface ([12], [13]) over UART1 channel with following settings: • *Baudrate: 9600; Data bits: 8; Parity: None; Stop bits: 1; Flow ctrl: None;*

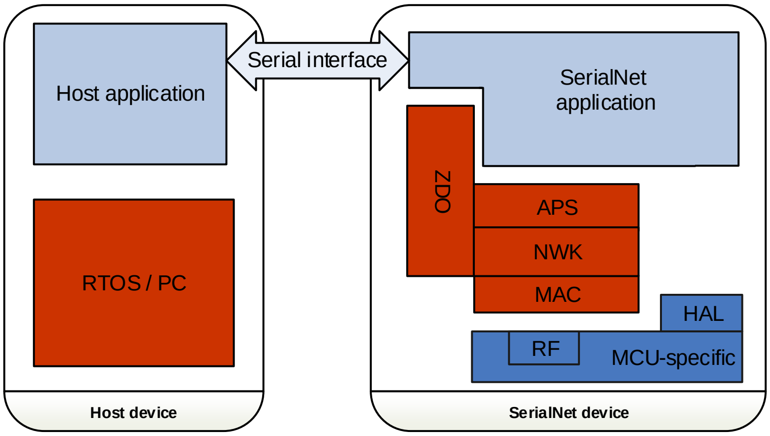
Directory Name	Description
/SerialNet_RFR2_Misc/ ../Uart0/ ../Uart1/	Contains SerialNet FW images that can be used without bootloader or with latest AVR2054 bootloader. `.srec` images require that AVR2054 bootloader [13] will be loaded first via JTAG/ISP and then actual SerialNet image should be programmed over the serial bootloader interface per its settings [1]. `.hex` images contain SerialNet application and may also contain AVR2054 bootloader code if indicated so by the name (see Table 1-2). They shall be loaded via JTAG/ISP and require following fuse bytes: • Files with _SerialBL: *Ext: 0xFE; High: 0x18; Low: 0x62* • Files with _NoBL: *Ext: 0xFE; High: 0x19; Low: 0x62* `.elf` images contain SerialNet, bootloader if indicated by name (see Table 1-2), as well as necessary fuse settings. Shall be programmed via JTAG/ISP.

Note: 1. Due to specifics in the board design when connecting to MEGARFR2-XPRO [4] the host side shall configure serial interface with Flow Control set to *Hardware (RTS/CTS)* even if embedded image on the board uses no flow control.

1.3 Architecture Overview

The SerialNet application is developed on top of the Atmel BitCloud® ZigBee® PRO-certified stack [10]. It provides an easy-to-use control over ZigBee PRO networking functionality that is accessible to the host device through a serial connection using an extensive set of AT commands in ASCII format. A SerialNet device executes received requests and responds to the host. Figure 1-1 illustrates the basic architecture.

Figure 1-1. SerialNet usage Scheme

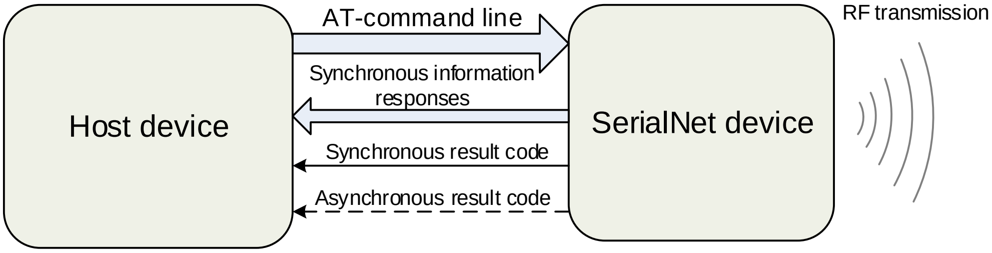


An important feature of SerialNet is the capability to request execution of a particular function over the air via the `ATR` command (see Section 5.9 Remote Management). It allows transferring AT commands to the remote node in the network, executing them there and redirecting the execution output to the originator. Thus, the remote node can be monitored and commissioned, and the corresponding parameters can be set.

1.4 Protocol Principles

SerialNet supports an extensive set of AT commands that provide full control over different functionality of the module. Read/write commands to S-registers can be used to access device and network parameters. In many cases, AT command functionality can be duplicated by certain S-registers to reduce overhead of the serial protocol. The basic principle of the SerialNet protocol is illustrated in Figure 1-2.

Figure 1-2. SerialNet Command Executions



The host device shall transmit a command line prefixed by the "AT" string and followed by the chained SerialNet commands to be executed consecutively. Upon successful execution of each command in the sequence, corresponding information responses is returned to the host device in an easily recognizable string format. The final result of the command line execution is indicated by the result code. In case any command is executed incorrectly, the command sequence is interrupted and the **ERROR** result code is returned. The **OK** result code is returned if all commands in the sequence were executed successfully.

Each command in a sequence may have a different syntax, depending on whether it is used to execute an action, read or write parameter(s), or test a valid parameter range. An example illustrating different command and response types is provided in Table 1-4.

Table 1-4. AT Command String Execution Example

	Command/response	Comment
Command string to device	ATE1V1+WTXPWR=-4+WLQI2+WRSSI2S22?	Turn echo on (E1), enable verbose response (V1), set TX power level to -4dBm (+WTXPWR=-4), request LQI and RSSI for link with node having short address 2 (+WLQI2+WRSSI2), request active channel (S22?)
Information responses	**+WLQI:254**	LQI value is 254
	+WRSSI:-80	RSSI is -80dBm
	B	Node is operating on channel 0x0B
Result code	**OK**	Execution is completed successfully

More complex examples are provided in Section 2.2 Command Examples.

In addition to synchronous result codes indicating command execution status, a SerialNet device can send asynchronous result codes to the host device upon specific events. The full list of both verbose and numeric forms of the result codes can be found in Section 3.3 SerialNet Result Codes.

Section 3.1 SerialNet AT Commands summarizes the basic specifications of AT commands grouped into functional categories, while detailed definition for each command is given in Chapter 5 Command Descriptions.

2 Getting Started

2.1 Connection with SerialNet Device

The supported platform (see Table 1-1) is first programmed via JTAG with the SerialNet firmware image built for the corresponding platform.

After that it is connected to a host device (a PC, MCU, etc.) using a USART interface via target USART port. To start communication, the host device shall open corresponding serial port with default settings given in Table 2-1.

Table 2-1. Default COM Port Settings for SerialNet

Baud rate	38400
Data bits	8
Parity	None
Stop bits	1
Flow control	None

Note: These parameters can be modified for a SerialNet device and saved in persistent memory using the corresponding commands described in Section 5.7.8 "+IPR" - Serial Port Communication Rate and Section 5.7.9 "+IFC" - Serial Port Flow Control.

If a PC is the host, the HyperTerminal application included with Microsoft® Windows® can be used to communicate with the SerialNet device. To check the connection, "`AT`" should be entered in the terminal window, followed by a CR (Enter button). If the board responds with an OK result code, the communication between the host and SerialNet devices has been established successfully.

Section 2.2 Command Examples provides examples showing how a SerialNet device can be configured for network operations; perform data exchange, remote control, and other commands.

2.2 Command Examples

The examples given below show the usage of AT commands to control SerialNet devices, and are valid for all supported platforms listed in Table 1-1.

2.2.1 Configure Nodes for Networking

The following examples require at least two nodes. The first step is to configure network parameters on each node. One of the nodes will function as a network coordinator, and the others can be routers or end devices. It is important that all nodes have different extended (MAC) and short (NWK) addresses. The coordinator node shall have short address 0, and all other nodes shall have non-zero addresses.

Note: Selection of particular addresses is application dependent. It should be done only for the first time during the manufacturing process of initial installation.

Table 2-2. Configuring Network Coordinator

Command/response	Comment
`ATX`	Set a node to transmit **EVENT** and **DATA** to a host.
OK	
`AT+GSN=1`	Set extended address for the node.
OK	
`AT+WPANID=1620`	Set node's extended PAN ID.
OK	
`AT+WCHMASK=100000`	Set node's channel mask (this one enables channel 0x14 only).
OK	
`AT+WROLE=0 +WSRC=0`	Set coordinator role and short address to 0x0000.
OK	
`AT+WJOIN`	Perform network start.
OK	Result code for successful network start.

If the node indicates **ERROR**, it means the embedded software does not support the coordinator function and cannot be configured in such a way. In this case, try checking the coordinator support on other nodes using the `AT+WROLE?` command, as described in Section 5.1.7 "+WROLE" - Set/Get Node Role (coordinator / router / end device).

Then configure another device to be a router node:

Table 2-3. Configuring Network Router

Command/response	Comment
`ATX`	Set a node to transmit **EVENT** and **DATA** to a host.
OK	
`AT+GSN=2`	Set extended address for the node.
OK	
`AT+WPANID=1620`	Set node's extended PAN ID.
OK	
`AT+WCHMASK=100000`	Set node's channel mask (this one enables channel 0x14, only).
OK	
`AT+WROLE=1 +WSRC=55`	Set router role, short address equal to 0x0055.
OK	
`AT+WJOIN`	Perform network join.
OK	Indication that node has successfully joined the network.

2.2.2 Checking Network Status and Data Transmission

Now we can easily verify networking status on both devices by sending the `AT+WNWK` command and perform data exchange between them. For example, on the coordinator:

Table 2-4. Verify Networking Status on the Coordinator

Command/response	Comment
`AT+WNWK`	Request networking status.
`OK`	Means the node is in the network.

`AT+WWAIT=3000`
`OK`
`ATD55`
`HELLO`
`OK`
	Set a three second timeout to wait for input, and send `HELLO` to the node with short address 55.

Simultaneously, `HELLO` will appear on the terminal connected to the router in the form of a **`DATA`** event:

Table 2-5. Verify Networking Status on Router Terminal

Command/response	Comment
`DATA 0000,0,5:HELLO`	Data (five bytes) came from the device with address 0000 by unicast request without acknowledgement required.

2.2.3 Remote Command Execution

The ATR command provides a mechanism for AT command execution on a remote node with command response redirection to the originator. Thus, it allows remote monitoring and configuration over the air.

The example below demonstrates how to execute AT commands on the router device remotely using the ATR command on the coordinator:

Table 2-6. Remote Execution of AT Commands

Command/response	Comment

`ATR55,0,+WROLE?+GSN?`
`+WROLE:1`
`+GSN:0000000000000002 OK`
	Get node role and extended address from the node with address 55

`ATR55,0,+GMI?`
`+GMI:ATMEL OK`
	Get model number from the node with address 55

`ATR55,0,+WAUTONET=1S30=1`
`OK`
	Set auto-network mode and command addressing mode to the node with address 55

2.2.4 End Device Power Control

This example demonstrates how to configure an end device node with a certain duty cycle, perform a network join, and deliver data to an end device:

Table 2-7. Configure End-device Node with Duty Cycle

Command/response	Comment

`ATX`
`OK`
`AT+GSN=3`
`OK`
`AT+WROLE=2 +WSRC=56`
`OK`
`AT+WPANID=1620+WCHMASK=100000`
`OK`
	Set a node to transmit `EVENT` and `DATA` to a host
	Set extended (MAC) address for the node
	Set the node as an end device with short address 0x0056
	Set extended PAN ID and channel mask (channel `0x14`) for this node

`AT+WPWR=100,100`
`OK`
`AT+WPWR?`
`+WPWR:100,100`
`OK`
	Set duty cycle to 10s sleep / 1s active
	Verify that the duty cycle is accepted successfully

`AT+WJOIN` **`OK`**	Perform a network join Result code indicating successful network join for the end device

Now, the data intended for the end device can be sent from the coordinator:

Table 2-8. Test Data from the Coordinator

Command/response	Comment

`ATD56,0,4`
`test`
`OK`
	Send `test` data from the coordinator to the node with address 56

In the active state, the end device periodically polls its parent for buffered data with an interval configured by the `+WSYNCPRD` parameter. In the example given, it retrieves the test frame:

Table 2-9. Polling of Buffered Data from Parent

Command/response	Comment
`DATA 0000,0,4:test`	The word `test` is received by the end device after wake up

2.2.5 Control of GPIOs

Mapping of MEGARFR2 module I/O pins are summarized in Table 2-10.

Table 2-10. GPIO Pin Summary

I/O pin	MEGARFR2-ZIGBIT/ MEGARFR2-XPRO/ MEGARFR2-ZIGBIT-XPRO PinOut
GPIO0	PE2
GPIO1	PE3
GPIO2	PD6
GPIO3	PG2
GPIO4	PE0
GPIO5	PE1
GPIO6	PG5
GPIO7	PD0
GPIO8	PD1

Configure the I/O pins via commands:

Table 2-11. Configure I/O Pins

Command/response	Comment
`ATS120=3 S122=3 S123=3`	Configure GPIO0, GPIO2, GPIO3 for output.
`OK`	
`ATS124=1`	Configure GPIO4, and turn on internal pull-up.
`OK`	

Afterwards, it is possible to control the GPIOs using the corresponding S-registers:

Table 2-12. Control LEDs and Check Switch

Command/response	Comment
`ATS130=1 S132=0 S133=1`	Set GPIO0 and GPIO3 high and GPIO2 low.
`OK`	
`ATS134?`	
`1`	SW2:1 is in the OFF state.
`OK`	

3 SerialNet Commands Summary

This chapter provides a summary of AT commands, S-registers, and response codes supported by SerialNet. General overview of the syntax used in SerialNet is described in Chapter 4 Protocol Syntax while detailed description for all commands is given in Chapter 5 Command Descriptions.

3.1 SerialNet AT Commands

The AT commands implemented in SerialNet fall into the following categories:

- Network configuration
- Network management
- Security configuration
- Data transmission
- Power management
- Generic control
- Host interface control
- Hardware control
- Remote management

Table 3-1 provides a full list of SerialNet AT commands, along with information about supporting node roles, syntaxes, corresponding S-registers (if any), persistence, and references to their detailed documentation in Chapter 5 Command Descriptions.

Table 3-1. AT Commands Summary

Command	Function	Node type (C/R/E)	S-register	Action syntax	Set syntax	Read syntax	Range syntax	Persistence	Reference
Network configuration									
+WPANID	Extended PAN ID	CRE	20, 21		x	x	x	x	5.1.2
+WCHAN	Active channel	CRE	22			x			5.1.3
+WCHMASK	Channel mask	CRE	23		x	x	x	x	5.1.4
+WCHPAGE	Channel page	CRE	25		x	x	x	x	5.1.5
+WAUTONET	Automatic networking	CRE	24		x	x	x	x	5.1.6
+WROLE	Node role	CRE	33		x	x	x	x	5.1.7
+GSN or I4	Device extended address	CRE			x	x			5.1.8
+WSRC	Node short address	CRE	55		x	x	x	x	5.1.9
+WNWKPANID	Short (network) PAN ID	CRE			x	x	x		5.1.10
Network management									
+WJOIN	Start/join the network	CRE		x					5.2.1
+WLEAVE	Leave the network	CRE		x					5.2.2
+WNWK	Request for networking status	CRE		x					5.2.3
+WPARENT	Request for parent address	E				x			5.2.4
+WCHILDREN	Request for children addresses	CR				x			5.2.5

Command	Function	Node type (C/R/E)	S-register	Action syntax	Set syntax	Read syntax	Range syntax	Persistence	Reference
+WNBSIZE	Request for a number of neighbor nodes	CRE				x			5.2.6
+WNB	Request for neighbors' information	CRE				x			5.2.7
S30	Network addressing mode	CRE	30		x	x			5.2.8
+WLQI	Request for LQI	CRE		x					5.2.9
+WRSSI	Request for RSSI	CRE		x					5.2.10
Security configuration									
+WSECON	Enable/disable security	CRE		x					5.3.1
+WSECSTATUS	Set/get security status	CRE		x					5.3.2
+WNETKEY	Set/get network encryption key	CRE		x					5.3.3
+WTCADDR	Set/get trust center address	CRE				x			5.3.4
Data transmission									
D	Send data to a specific address	CRE		x					5.4.2
DB	Send binary data to a specific address	CRE		x					5.4.3
DU	Send broadcast data	CRE		x					5.4.4
DS	Send S-register value to a specific address	CRE		x					5.4.5
+WPING	Ping the node	CRE		x					5.4.6
+WSYNCPRD	Indirect poll rate	CRE	37		x	x	x		5.4.7
+WTIMEOUT	Data delivery timeout	CRE	51			x			5.4.8
+WRETRY	Data retries amount	CRE	52		x	x	x		5.4.9
+WWAIT	Data transmission waiting timeout	CRE	53		x	x	x	x	5.4.10
Power management									
+WPWR	End device sleep parameters	CRE	31, 32		x	x	x	x	5.5.1
+WSLEEP	Force to sleep	CRE		x					5.5.1
+WTXPWR	TX power level	CRE	34		x	x	x	x	5.5.3
Generic control									
Z	Warm reset	CRE		x					5.6.1
&H	Help	CRE		x					5.6.2
%H	Display parameters and S-register values	CRE		x					5.6.3
I, I0	Display product identification information	CRE		x					5.6.4
+GMI or I1	Request manufacturer identification	CRE		x					5.6.5
+GMM or I2	Request model identification	CRE		x					5.6.6
+GMR or I3	Request hardware/software revision identification	CRE		x					5.6.7
&F	Set to factory-defined configuration	CRE		x					5.6.8
+WACALIBRATE	Configure periodic internal clock calibration	CRE		x					5.6.9
+WCALIBRATE	Calibrate internal clock	CRE		x					5.6.10
Host interface commands									
S3	Termination character	CRE	3		x	x		x	5.7.1

Command	Function	Node type (C/R/E)	S-register	Action syntax	Set syntax	Read syntax	Range syntax	Persistence	Reference
S4	Response formatting character	CRE	4		x	x		x	5.7.2
S5	Command editing character	CRE	5		x	x		x	5.7.3
E	Command echo	CRE		x				x	5.7.4
Q	Result code suppression	CRE		x				x	5.7.5
V	Response format	CRE		x				x	5.7.6
X	Result code selection	CRE		x				x	5.7.7
+IPR	Serial port communication rate	CRE			x	x	x	x	5.7.8
+IFC	Serial port flow control	CRE			x	x	x	x	5.7.9
&D	DTR behavior	CRE	50	x				x	5.7.10
S0	Request the latest result code	CRE	0			x			5.7.11
Hardware control									
S120…S128	GPIO configuration	CRE	120 128		x	x		x	5.8.2
S130…S138	GPIO control	CRE	130 138		x	x			5.8.3
S100	A/D configuration	CRE	100		x	x		x	5.8.4
S101…S104	A/D conversion	CRE	101 104			x			5.8.5
S140, S141, S142	PWM configuration	CRE	140 141 142		x	x			5.8.6
S143, S144, S145	PWM frequency control	CRE	143 144 145		x	x			5.8.7
S146, S147, S148	PWM duty cycle control	CRE	146 147 148		x	x			5.8.8
POKE	Writing to hardware registers	CRE		x					5.8.1
PEEK	Reading from hardware registers	CRE		x					5.8.1
Remote management									
+WPASSWORD	Set a password	CRE		x				x	5.9.1
R	Remote execution of AT commands	CRE		x					5.9.2

Note: The second column contains roles of nodes to which a given command is applicable. C stands for coordinator, R for router, and E for end device.

3.1.2 Parameter Persistence

In Table 3-1, many parameters associated with AT commands are indicated as persistent. This means that their values are stored in the MCU’s persistent memory, and in contrast to no persistent parameters, they will not be set to default configuration upon device reset but will be restored to the values saved in persistent memory. However, the value assigned to a persistent parameter by the corresponding AT command is not written to the persistent memory right away. Instead, it is applied to the SerialNet operation, but is kept in RAM.

The networking parameters are stored automatically by BitCloud stack on pre-defined events that are likely to change their values. The application specific parameters are stored immediately when the particular command is sent. All persistent parameters in internal Flash are updated to actual values (if necessary) automatically upon `ATZ` command execution (see Section 5.6.1 “Z” - Warm Reset).

Upon device reset, SerialNet assigns persistent parameters to their values stored in internal Flash. If a parameter value has not been transferred from RAM to Flash, then the old Flash value will be used.

3.2 SerialNet S-registers

An extensive set of S-registers available in SerialNet provides easy read/write access to device and networking parameters. In many cases, AT command functionality is duplicated by certain S-registers to reduce overhead of the serial ASCII protocol.

Table 3-2. SerialNet S-registers

S-register	Parameter	Acceptable operations (R/RW)	Command reference
S0	The latest result code	R	5.7.11
S3	Termination character	RW	5.7.1
S4	Response formatting character	RW	5.7.2
S5	Command editing character	RW	5.7.3
S21, S20	PAN ID	RW	5.1.2
S22	Active channel	R	5.1.3
S23	Channel mask	RW	5.1.4
S24	Automatic networking	RW	5.1.6
S25	Channel page	RW	5.1.5
30	Network addressing mode	RW	5.2.8
S31, S32	Power management	RW	5.5.1
S33	Node role	RW	5.1.7
S34	TX power level	RW	5.5.3
S37	Indirect poll rate	RW	5.4.7
S50	DTR behavior	RW	5.7.10
S51	Data delivery timeout	R	5.4.8
S52	Data retries amount	RW	5.4.9
S53	Data transmission waiting timeout	RW	5.4.10
S55	Own network address	RW	5.1.9
S100	A/D configuration	RW	5.8.4
S101…S104	A/D conversion	R	5.8.5
S120…S128	GPIO configuration	RW	5.8.2
S130…S138	GPIO control	RW	5.8.3
S140, S141, S142	PWM configuration	RW	5.8.6
S143, S144, S145	PWM frequency control	RW	5.8.7
S146, S147, S148	PWM duty cycle control	RW	5.8.8

3.3 SerialNet Result Codes

Result codes appear either synchronously to an AT command indicating the execution status, or asynchronously notifying about specific events in the network or on a SerialNet device. For general description of result code syntax and its configuration options, see Section 4.6 Device Responses.

Table 3-3 provides description and both verbose and numeric forms for result codes available in SerialNet.

Table 3-3. SerialNet Result Codes

Verbose code	Numeric code	Parameters	Description
`OK`	0	None	Command is executed successfully
`ERROR`	4	None	Error occurred during command execution
`DATA`	8	`<addr>,<bcast>,<length>:<data>`	Indicates data reception from a remote node. `addr` is a short (network) address of the source node the data is originating from `bcast` is set to 1 if data are sent by broadcast transmission; otherwise, it is set to 0 `length` is a length of the `<data>` field `data` is a byte sequence of received data Note: The `+WPING` command (see Section 5.4.6 "+WPING" - Ping the Node) results in the following code on the destination node: `DATA <addr>,0,0:`
`EVENT`	7	`:<text>`	`text` is text specifying an event
		`:JOINED`	Indicates that the node has joined the network. Note: Event is returned in auto-network mode only, and not after the `+WJOIN` command.
		`:LOST`	Indicates that the node has lost connection to the network (that is, to its current parent). Note: This event can occur on end-device nodes only, and is not returned after `+WLEAVE` command.
		`:CHILD_JOINED <addr>`	Indicates to the parent that a device with extended address `<addr>` has just joined to it as a child.
		`:CHILD_LOST <addr>`	Indicates to the node that its child end device with extended address `<addr>` has disconnected from the node. Note: This event occurs when the child end device switches to a new parent, when it leaves the network using the `+WLEAVE` command, or when it is not accessible (powered off, no link, etc.) for *3*(sleep_interval + sync_period),* as configured on the parent device by the `+WPWR` and `+WSYNCPRD` commands.
		`:CALIBR`	Indicates that the device has successfully calibrated its internal clock after encountering errors on the serial interface.

4 Protocol Syntax

4.1 Character Formatting and Data Rates

Data transmitted between the host and the SerialNet device over a serial interface conform to the requirements for start-stop data transmission specified in the ITU-T Recommendation V.4 [6]. Parity is even, odd, or not used. Each character has at least one complete stop bit. The module accepts commands using any combination of parity and stop bits supported. These include, at least, the following combinations, each of which consists of up to 10 bits (including the start bit):

7 data bits, even parity, 1 stop bit

7 data bits, odd parity, 1 stop bit

8 data bits, no parity, 1 stop bit

Both the host and the module are able to accept commands at 1200 bits per second, at least. Particular character formatting and the data rate can be changed using appropriate AT commands (see Section 5.7.8 "+IPR" - Serial Port Communication Rate, Section 5.7.9 "+IFC" - Serial Port Flow Control, and Section 5.7.6 "V" - Response Format). The host has the means to explicitly select the data rate and character formatting according to the specifications above.

4.2 Alphabet

For any information exchange between the module and the host, the T.50 International Alphabet 5 (IA5) [6] is used. Only the seven low-order bits of each character are significant and any eighth or higher-order bit(s), if present, are ignored for the purpose of identifying commands and parameters. Lowercase characters (hex codes `0x61` through `0x7A`) are considered identical to their uppercase equivalents (hex codes `0x41` through `0x5A`) when received from the host by the module. Result codes from the module, which are particularly defined, are specified in uppercase.

4.3 Basic Command-line Operations

Command-line editing, echoing, and repeating are done in accordance with Clauses 5.2.2, 5.2.3, and 5.2.4 of the Recommendation V.250. The description below follows the statements introduced in [6].

The module may echo back to the host the characters received from the host, depending on the setting of the E command (see Section 5.7.4 "E" - Command Echo). If enabled, the characters received from the host are echoed at the same rate, parity, and format as those received.

The module checks on the characters coming from the host first to see if they match the termination character, `S3` (see Section 5.7.1 "S3" - Termination Character). Next, it checks the editing character, `S5` (see Section 5.7.3 "S5" - Command Editing Character), before considering any other character. This insures the characters will be properly recognized even though they were set to values the device uses for other purposes. If `S3` and `S5` are set to the same value, the character checked will be treated as a character matching `S3` (as `S3` is checked before `S5`).

The character defined by the `S5` parameter (by default, it is the backspace character, `BS` [hex code `0x08`], see Section 5.7.3 "S5" - Command Editing Character) is intended to be interpreted as a request from the host to the device to delete the previous character. Any control characters (hex codes `0x00` through `0x1F`, inclusive) that remain in the command line after receiving the termination character will be ignored by the SerialNet.

Once the device finds the termination character, it starts processing the command line. The command line starts with `AT` (characters `0x41` and `0x54`) and may contain a sequence of commands with syntax formats described in Section 4.5 Command Syntax.

A command (with associated parameters, if any) may be followed by additional commands in the same command line without using any delimiting character. Some commands may cause the remainder of the command line to be ignored (the D command, for instance, see Section 5.4.2 “D” - Send Data to a Specific Node).

If the command line is started with the 'A/' or 'a/' prefix (hex codes 0x41, 0x2F or 0x61, 0x2F), the device immediately repeats the execution of the preceding command line. No editing is possible, and no termination character is required. With this mechanism, a command line may be repeated as much as desired.

After processing and execution of the command line result code is returned as described in Section 4.6.2 Result Codes.

4.4 Parameter Values

Parameters may be defined as “read-only” (R) or “read/write” (RW). Read-only parameters are used to provide the host with status or identifying information, but are not set by the host. Attempting to set such a parameter will return an error. In some cases (depending on the particular parameter), the module may ignore any attempt to set the value for such a parameter rather than respond with the **ERROR** result code. Read-only parameters may be read and tested.

Read/Write parameters may be set by the host in order to store a value or values for later use. Read/Write parameters may be set, read, and tested.

Parameters may take either a single value or multiple (compound) values. A compound value consists of any combination of numeric values (as defined in the description of the action or parameter command). The comma character (hex code 0x2C) is included as a separator before the second and all subsequent values in the compound value. If a value is not specified as missed (that is, defaults assumed), the required comma separator should be specified; however, trailing comma characters may be omitted if all the associated values are also omitted.

Note: When any of the optional parameters are misused in a command, the command would be performed as if the parameter was omitted. That parameter would be further treated as if the other subsequent command were input, probably causing an **ERROR** message. To avoid confusion, follow the command syntax.

Each parameter value may be either a decimal or hexadecimal number. The choice depends on whether or not a particular command and hexadecimal number is preceded with '0x.' Hexadecimal numbers can represent 16-bit, 32-bit, 64-bit, and 128-bit values.

Decimal numeric constants consist of a sequence of one or more of the characters '0' (hex code 0x30) through '9' (hex code 0x39), inclusive, and can be preceded by minus “-.” The most-significant digit is specified first. Leading '0' characters will be ignored.

Hexadecimal numbers consist of characters “0” through “9” and “A” through “F,” inclusive. The minus sign is not allowed. Leading '0' characters will be ignored. To prevent misinterpretation of hexadecimal numbers in cases when the command containing them is not the last in the AT string, it is strongly recommended to add the leading zeroes. Thus, if a parameter is 32 bits long, it would be eight characters long, and if it is a 64-bit number, it would contain 16 characters, and so on.

As a special case, a string constant appears in the R command only (see Section 5.9.2 “R” - Remote Execution of AT Command) only. Then it is just a sequence of displayable IA5 characters, each in the range of 0x20 to 0x7F, inclusive.

4.5 Command Syntax

Requests from host to the SerialNet device can be differentiated in two types based on their syntax:

- AT-commands
- S-register commands

AT-command syntax overview is given in Table 4-1 and described in details in sections 4.5.2 through 4.5.5, while syntax for S-register commands is given in Section 4.5.6 S-registers Syntax.

Table 4-1. AT Command Syntax Formats

AT-Command type	Syntax
Action command	`AT<command>[<value>]`
Parameter set command	`AT<command>=<value>`
Parameter read command	`AT<command>?`
Testing a range of valid values	`AT<command>=?`

where `<command>` is one of the following:

1. a single character.
2. `'&'` character (`0x26`), followed by a single character.
3. `'%'` character (`0x25`), followed by a single character.
4. `'+'` character, followed by a string of characters.

The characters allowed to be used in `<command>` should be taken from the T.50 International Alphabet [6].

The first three of the `<command>` cases above are referred to as basic commands; they may be of the action command syntax only. Commands beginning with the "+" character are known as extended syntax commands, and can fit all the syntax rules, depending on their type.

Typically, a command that supports the parameter set syntax also supports the range testing syntax.

Some commands allow omitting a `<value>`. If a command does omit one, then it should be immediately followed by another command (or the termination character) in the command line.

After processing and execution of the command line a result code is returned as described in Section 4.6.2 Result Codes.

4.5.2 Action Command Syntax

The format of the action commands, except for the `D`, `DU`, and `S` commands is given in Table 4-2.

The `<value>` may be either a single value parameter or a compound value parameter, as described in Section 4.4 Parameter Values. Some commands may have no parameters at all. Expected value range is noted in the description of a particular command.

Table 4-2. Action Command Syntax

Command type	Syntax	Example	Comment
Action command with no parameters used	`AT<command>`	`AT+WLEAVE` **OK**	Leave the network Result code

Action command with one or more sub-parameters used	`AT<command>`
	`[<value>]`
		`ATX2`
		OK
			`X` - Configures result codes to be indicated to the host
			`2` - Disables events and data indications

4.5.3 Parameter Set Command Syntax

The following syntax is used for a parameter set command:

Table 4-3. Parameter Set Command Syntax

Command type	Syntax	Example	Comment

Parameter set command	`AT<command>=[<value>]`	`AT+WWAIT=4000`
		`OK`
			Set parameter +WWAIT
			Result code

If the named parameter is implemented in the SerialNet, all the mandatory values are specified, and all values are valid according to the definition of the parameter then the specified values should be set and OK status will be returned. In case of an error, the previous values of the parameter are unaffected. For more information on the result code values and configuration see Section 4.6.2 Result Codes.

4.5.4 Parameter Read Command Syntax

The host may determine the current value or values stored in a parameter by using the following syntax:

Table 4-4. Parameter Read Command Syntax

Command type	Syntax	Example	Comment
Parameter read command	`AT<command>?`	`AT+WRETRY?` **`+WRETRY:3`** **`OK`**	Request for parameter `+WRETRY` Returned value Result code

If the named parameter is implemented, its current values are sent to the host in an information text response. The format of this response is described in the definition of the parameter.

Generally, the response string begins with `<command>` followed by the **`':'`** character and the values represented in the same form in which they would be generated by the host in a parameter set command. If multiple values are supported, they will generally be separated by commas, as in a parameter set command. For more details on the information response and result code values and configuration, see Section 4.6 Device Responses.

4.5.5 Parameter Range Command Syntax

Table 4-5. Parameter Range Command Syntax

Command type	Syntax	Example	Comment

Parameter range command	`AT<command>=?`	`AT+WSRC=?`
		`+WSRC:(0000-FFF7)`
		`OK`
			Request for valid range of the short address.
			Returned value (information response).
			Result code.

If the device recognizes the parameter name, it returns an information text response to the host, followed by the **`OK`** result code. The information text response will indicate the values supported by the module for each of the sub-parameters, and, possibly, additional information. The format of this information text response is defined for each parameter. Generally, an information text response starts with a `<command>` followed by the **`':'`** character. For more details on the information response and result code values and configuration, see Section 4.6 Device Responses.

When an action/parameter accepts a single numeric sub-parameter, or the parameter accepts only one numeric value, the set of supported values may be presented in an information text as an ordered list of values. The list should be preceded by left parenthesis, '(' (hex code `0x28`), and closed by right parenthesis, ')' (hex code `0x29`). If that very single value is supported, it should appear in parentheses. If more than one value is supported, then the values may be listed individually, separated by comma characters (hex code `0x2C`). When a continuous range of values is supported, the values appear in the form of the first value in the range and the last value in the range, both separated by a hyphen character (hex code `0x2D`). The specification of single values and value ranges may be alternated within a single information text. Nevertheless, the supported values should be indicated in an ascending order. The following are some examples of value range indications:

Table 4-6. Value Range Indications

Value range examples	Comment
`(0)`	Only the `0` value is supported.
`(1,2,3)`	The values `1, 2`, and `3` are supported.
`(1-3)`	The values `1` through `3` are supported.
`(0,4,5,6,9,11,12)`	The several listed values are supported.
`(0,4-6,9,11-12)`	Alternative expression of the previous list.

The value may be either a single value parameter or a compound value parameter as described in Section 4.4 Parameter Values. Some commands may have no parameters at all. Expected values are noted in the description of a particular command.

When an action/parameter accepts more than one sub-parameter, or the parameter accepts more than one value, the set of supported values may be presented as a list of the parenthetically-enclosed value range strings, separated by commas. For example, the information text in response to testing an action that accepts three sub-parameters and supports various ranges for each of them could appear as follows:

```
(0), (1-3), (0,4-6,9,11-12)
```

This indicates that the first sub-parameter accepts only the 0 value, the second accepts any value from `1` through `3`, inclusively, and the third sub-parameter accepts any of the values `0, 4, 5, 6, 9, 11,` or `12`.

4.5.6 S-registers Syntax

S-registers represent a group of numerical parameters that can be addressed in a special syntax. Each S-register has its own address and value. Some S-registers are standardized by the V.250 recommendations, and are reused in the SerialNet while some of the S-registers are non-standard, defined specifically by the SerialNet firmware.

Commands that begin with the `'S'` character are allowed for S-register access. These differ from other AT commands in some respects. The number following the `'S'` character indicates the referenced "register number." If the number is not recognized as a valid register number (register is omitted), the **`ERROR`** result code is generated.

Immediately following that number, either a `'?'` or `'='` character (hex codes `0x3F` or `0x3D`, respectively) should appear. `'?'` is used to read the current value of the indicated S-parameter. `'='` is used to set the S-parameter to a new value.

Table 4-7. S-registers Syntax

Command type	Syntax	Example	Comment
Reading the S-register	`ATS<reg_number>?`	`ATS0?` **3** **OK**	Read the latest result code. Previous request failed due to out-of-range value. Result code.
Setting the S-register	`ATS<reg_number>=[<value>]`	`ATS23=800` **OK**	Set channel mask to 0x800 Result code

If the `'='` character is used, the new value to be stored in the S-parameter is specified in decimal form following the `'='` character. If no value is given (that is, the end of the command line occurs or the next command follows immediately), the corresponding S-parameter will be set to 0. The ranges of acceptable values are given in the description of each S-register. Section 3.2 SerialNet S-registers gives functional representation of S-registers supported in SerialNet.

4.6 Device Responses

SerialNet device is sending responses back to the host device. There are two types of responses that may be generated by the SerialNet device:

1. Information responses.
2. Result codes.

Any device response consists of three parts: header, response text, and trailer.

The characters generated in the header are determined by the user's setting (see `V` command, see Section 5.7.6 “V” - Response Format). The trailer consists of two characters, namely the ordinal value of parameter `S3`, followed by the ordinal value of parameter `S4`. Response text formatting is different for information responses and result codes.

4.6.1 Information Responses

Information response is a command-specific output returned synchronously for the entered AT-command (for example for parameter read or range commands). It may contain multiple lines, and the text may include any formatting characters. The particular information text format returned by particular commands is specified in the command definition in Chapter 5 Command Descriptions while general response format description is also mentioned in command syntax sections above (see sections 4.5.2 through 4.5.5).

Note: The device may insert intermediate `<CR>` characters in very long information text responses, in order to avoid overflow in the host receive buffers (for example `AT&I`). If intermediate `<CR>` characters are included, the module does not include the character sequences “**0** `<CR>`” (`0x30`, `0x0D`) or “**OK**`<CR>`” (`0x4F`, `0x4B`, `0x0D`), so that the host can avoid false detection of the end of these information text responses.

4.6.2 Result Codes

There are two general types of result codes: final and unsolicited.

Final result codes (**OK/ERROR**) indicate completion of the requested action and readiness to accept new commands from the host. The **OK** result code indicates successful execution of the entered command line while, the **ERROR** result code is generated if:

- entered command is not recognized;
- a sub-parameter is specified for an action that does not imply using sub-parameters;
- too many sub-parameters are specified;
- a mandatory sub-parameter is not specified;
- a value of the wrong type is specified;
- a value that is not within the supported range is specified;
- command-specific error occurred.

If command has returned **ERROR** result code its reason can be immediately obtained via `S0` register (see Section 5.7.11 “S0” - Request for the Latest Result Code).

Unsolicited result codes (such as **DATA**) may not be directly associated with the issuance of a command from the host. They indicate the occurrence of another **EVENT** causing them.

The result code text may be generated as a number or a string, depending on the user-selected setting (see Section 5.7.6 “V” - Response Format).

Command `X` (see Section 5.7.7 “X” - Result Code Selection) controls the generation of result codes, while command `Q` (see Section 5.7.5 “Q” - Result Code Suppression) controls whether they result codes are present at all.

Section 3.3 SerialNet Result Codes summarizes the result codes available in SerialNet in both verbose and numeric forms with their corresponding parameter(s), if any, and a brief description. Each command description in Chapter 5 Command Descriptions lists result codes that may be generated in relation to the command and the circumstances under which they will be issued.

5 Command Descriptions

5.1 Networking Parameters

This section describes SerialNet commands associated with networking parameters. Most of the parameters shall be set on each device according to the desired network characteristics prior to executing the network start/join procedure. Note that if the default setting or persistent value from the Flash (see Section 3.1.2 Parameter Persistence) already has the desired value for a network parameter, there is no need to assign it explicitly again prior to the network start/join.

There are also a number of hard-coded parameters that cannot be changed by AT commands, but which have direct impact on possible network topology and performance. Table 5-1 lists such parameters, and provides their values in SerialNet firmware images with and without security support.

Table 5-1. Values of BitCloud ConfigServer Parameters Applied in SerialNet Firmware

BitCloud ConfigServer parameter	No security support	With security support
`CS_NEIB_TABLE_SIZE`	25	10
`CS_MAX_CHILDREN_AMOUNT`	25	10
`CS_MAX_CHILDREN_ROUTER_AMOUNT`	10	5
`CS_ROUTE_TABLE_SIZE`	40	25
`CS_MAX_NETWORK_DEPTH`	6	5
`CS_NWK_BTT_SIZE`	50	30
`CS_ADDRESS_MAP_TABLE_SIZE`	10	10
`CS_ROUTE_DISCOVERY_TABLE_SIZE`	4	3
`CS_APS_DATA_REQ_BUFFERS_AMOUNT`	4	4
`CS_APS_ACK_FRAME_BUFFERS_AMOUNT`	3	3
`CS_DUPLICATE_REJECTION_TABLE_SIZE`	5	5
`CS_NWK_BUFFERS_AMOUNT`	4	4

Their values shall be taken into account during network establishment and operation. Details about each parameter can be found in BitCloud Stack documentation [10].

5.1.2 "+WPANID" - Set/Get Extended PAN ID

Table 5-2. "+WPANID" - Set/Get Extended PAN ID

Syntax	Explanation
`+WPANID=<value>`	The command sets the extended PAN ID for the device. `value` is the extended PAN ID in the form of a 64-bit hexadecimal number that uniquely identifies the target network. If PAN ID is set to `0`, the coordinator will form a network with the extended PAN ID equal to its extended (MAC) address. Router and end device nodes in such case will join the first available network, irrespective of its extended PAN ID. Notes: 1. Setting the extended PAN ID is possible only when the device is not in the network. 2. Several networks with different PANIDs can be operated in parallel on the same frequency channel.
`+WPANID?`	The command returns the extended PAN ID that is specified on the device for network operation.
`+WPANID=?`	The command requests a valid range for the extended PAN ID value.
`S-register`	`S21` (RW). This register is just keeping a copy of the parameter accessible through the `+WPANID` command. `S20` (R). This register contains the actual extended PAN ID that is used for networking. If the `S21` register is set to `0` and the device is in the network, this register will keep the extended PAN ID of the selected network. If the device has not been connected, this register contains `0`.
`Result codes`	The set command is executed if the device is not in the network and the extended PAN ID is in the valid range. In such case, the device returns `OK` upon completion. Otherwise, the extended PAN ID is ignored, and the device responds with **`ERROR`**.

Examples	`AT+WPANID=10`
	`OK`
	`AT+WPANID?`
	`+WPANID:0000000000000010`
	`OK`
	`AT+WPANID=?`
	`+WPANID:(0000000000000000-FFFFFFFFFFFFFFFE)`
	`OK`

Default value	`0000000000000000`
Persistence	`value` is stored in NV memory.
Node types	Coordinator / router / end device.

5.1.3 "+WCHAN" - Get Active Channel

Table 5-3. "+WCHAN" - Get Active Channel

Syntax	Explanation
`+WCHAN?`	The command requests the channel number (in hexadecimal form) the device is currently operating on. If the node is not in the network, **`FF`** is returned.
S-register	`S22` (R)
Result codes	**`OK`**
Example	`AT+WCHAN?` **`+WCHAN:0B`** **`OK`**
Node types	Coordinator / router / end device.

5.1.4 “+WCHMASK” - Set/Get Channel Mask

Table 5-4. “+WCHMASK” - Set/Get Channel Mask

<table>
<tr><th>Syntax</th><th colspan="2">Explanation</th></tr>
<tr><td>+WCHMASK=<value&gt;</td><td colspan="2">The command sets the channel mask to be enabled for network operation.
value is a 32-bit field (in hexadecimal form) that specifies the channel numbers supported by the node. The five most-significant bits of the channel mask (b31,...,b27) shall be set to 0. The remaining 27 bits (b26, b25,...,b0) indicate support status for each of the 27 valid channels (1 = supported, 0 = unsupported). Available channels are distributed across frequency bands as follows:
780MHz: channel numbers 0 – 3
868MHz: channel number 0
915MHz: channel numbers 1 – 10
2.4GHz: channel numbers 11 – 26
For sub-GHz bands, the corresponding channel page shall be configured by the +WCHPAGE command (see Section 5.1.5 “+WCHPAGE” - Set/Get Channel Page).
A detailed description of the channel mask parameter can be found in clause 6.1.2 of the 802.15.4-2006 standard [9].
Notes: 1. Only channels from frequency bands supported by the platform’s RF chip can be selected in the channel mask.
2. The command is not accessible when the node is joined to a network.</td></tr>
<tr><td>+WCHMASK?</td><td colspan="2">The command returns the actual channel mask as a hexadecimal value.
The returned channel mask can be different from the channel mask set by the +WCHMASK=<value> command, and depends on the hardware capabilities. The cleared bits mark unsupported channels.</td></tr>
<tr><td>+WCHMASK=?</td><td colspan="2">The command returns the channel capability mask in the form of two 32-bit unsigned hexadecimal numbers. It returns 00000800-07FFF800 for 2.4GHz chipsets and 00000001-000007FF for sub-GHz.
Note: Strictly speaking, these two numbers do not represent “range” in its direct sense, but are rather the maximum and minimum values achievable by the composition of corresponding bits.</td></tr>
<tr><td>S-register</td><td colspan="2">S23 (RW)</td></tr>
<tr><td>Result codes</td><td colspan="2">The set command is executed if the node is not in the network and the channel mask is set according to actually supported by HW. In such case, the device returns OK. Otherwise, the channel mask is ignored, and the device responds with ERROR.</td></tr>
<tr><td>Example</td><td>AT+WCHMASK=40000
OK
AT+WCHMASK?
+WCHMASK:00040000
OK
AT+WCHMASK=?
+WCHMASK(00000800-07FFF800)
OK</td><td>Enable only channel 19 (0x13) for operation

Check current channel mask</td></tr>
<tr><td>Default value</td><td colspan="2">00010000 for 2.4GHz chipsets or 00000001 for sub-GHz.</td></tr>
<tr><td>Persistence</td><td colspan="2">value is stored in the NV Memory.</td></tr>
<tr><td>Node types</td><td colspan="2">Coordinator / router / end device.</td></tr>
</table>

5.1.5 "+WCHPAGE" - Set/Get Channel Page

The command is available only for platforms with the Atmel AT86RF212/212B radio part.

Table 5-5. "+WCHPAGE" - Set/Get Channel Page

Syntax	Explanation
`+WCHPAGE=<value>`	The command sets the channel page that will be used for networking. Values 0 and 2 correspond, respectively, to BPSK and O-QPSK modulations on 868/915MHz channels. Value 5 means that the 780MHz frequency band with O-QPSK modulation shall be used. A detailed description of the channel page parameter can be found in clause 6.1.2 of the 802.15.4-2006 standard [9]. Note: The command is not accessible when the node is joined to a network.
`+WCHPAGE?`	The command returns actual channel page.
`+WCHPAGE=?`	The command returns possible channel pages: **0,2,5.**
S-register	`S25` (RW)
Result codes	**OK** if the device contains the RF212 radio chip and is not in the network; otherwise, `ERROR` is returned.

Example	`AT+WCHPAGE=0 OK AT+WCHPAGE?`
	+WCHPAGE:0
	OK
	`AT+WCHPAGE=?`
	+WCHPAGE:(0,2,5)
	OK

Default value	0
Persistence	`value` is stored in the NV memory.
Node types	Coordinator / router / end device.

5.1.6 "+WAUTONET" - Enable/Disable Automatic Networking

Table 5-6. "+WAUTONET" - Enable/Disable Automatic Networking

Syntax	Explanation
`+WAUTONET=<value>`	The command controls the node activity behavior at power up, reset, or when a connection loss is detected. `value` has a Boolean type. `1` implies automatic joining to the network at power up, reset, or network leave, `0` means that automatic joining is disabled and the `+WJOIN` command shall be used for the network start/join procedure.
`+WAUTONET?`	The command requests current automatic networking configuration.
`+WAUTONET=?`	The command requests the range of supported values.
S-register	`S24` (RW)
Result codes	**OK**

Syntax	Explanation
Example	AT+WAUTONET=1 **OK** AT+WAUTONET? **+WAUTONET:1** **OK** AT+WAUTONET=? **+WAUTONET:(0-1)** **OK**
Default value	0 – automatic networking is disabled.
Persistence	value is stored in the NV memory.
Node types	Coordinator / router / end device.

5.1.7 "+WROLE" - Set/Get Node Role (coordinator / router / end device)

Table 5-7. "+WROLE" - Set/Get Node Role (coordinator / router / end device)

Syntax	Explanation	
+WROLE= <value>	The command sets the node role to value as follows: 0 – Coordinator. 1 – Router. 2 – End device. Note: The command is not accessible when the node is joined to a network.	
+WROLE?	The command requests the actual node role.	
+WROLE=?	The command requests the node roles available for the device.	
S-register	S33 (RW)	
Result codes	**OK** is returned if value is in the valid range; otherwise, **ERROR** is returned.	
Example	AT+WLEAVE **OK** AT+WROLE=? **+WROLE:(0,1,2)** **OK** AT+WROLE=2 **OK** AT+WROLE? **+WROLE:2** **OK**	Leave the network Switch to the end device role
Default value	Depends on the firmware version. Typically 1 – Router.	
Persistence	value is stored in the NV memory.	
Node types	Coordinator / router / end device.	

5.1.8 "+GSN" – Set/Get Extended (MAC) Address

Table 5-8. "+GSN" – Set/Get Extended (MAC) Address

Syntax	Explanation
`+GSN=<value>`	The command assigns the device extended (MAC) address. `value` is a 64-bit hexadecimal number that uniquely identifies the device. Shall not be 0 or `0xFFFFFFFFFFFFFFFF` Note: The command is not accessible when the node is joined to a network.
`+GSN?` `I4`	The command returns the device extended (MAC) address in the form of a 64-bit hexadecimal number.
Result codes	`OK` is always returned.

Example	`AT+GSN=FEDCBA0987654321`
	`OK`
	`AT+GSN?`
	`+GSN:FEDCBA0987654321`
	`OK`
	`ATI4`
	`FEDCBA0987654321`
	`OK`
	Just an alias to `I4`

Default value	`0000000000000000` Note: User-defined MAC address shall be a non-zero value less than 0xFFFFFFFFFFFFFFFF. The default value for the supported platforms is not 0000000000000000 as the boards already has the unique MAC address stored inside the user signature page of MCU. This MAC address is a uniquely assigned ID for each ZigBit and owned by Atmel.
Persistence	`value` is stored in NV memory.
Node types	Coordinator / router / end device.

5.1.9 "+WSRC" - Set/Get Short (NWK) Address

Table 5-9. "+WSRC" - Set/Get Short (NWK) Address

Syntax	Explanation
`+WSRC=<value>`	The command assigns the device short (network) address. `value` is a 16-bit hexadecimal number that will be used by the device for communication in the network. It shall be unique within the network. There are two approaches for the short address assignment: • **Stochastic addressing** is applied if `value` is set to `FFFF` when the device is not in the network. In such case, short address is assigned randomly by the stack when the device joins the network. The stack also automatically resolves all possible address conflicts. After network join, this parameter contains the current short address of the node, but after leaving the network, the `+WSRC` value is reset to `FFFF`. Hence the node can get a new short address assigned during the next network join. • **Static addressing** is applied if `value` is set to non-`FFFF` prior to network join. After joining the network, the device will use the assigned value as its short address. Moreover, after leaving the network, the `+WSRC` value will be kept and reused during the next network join attempt, unless explicitly overwritten with another value. If static addressing is used, the user is responsible for ensuring that the node's short address is unique within the network. All nodes across the network shall use the same addressing mode. Notes: 1. The command is not accessible when the node is joined to a network. 2. The coordinator node shall always have its short address set as `0000`. Nodes of other roles shall have nonzero short addresses.
`+WSRC?`	The command returns the device short address in the form of a 16-bit hexadecimal number. If the node is in the network, the command returns its actual short address independently of the addressing scheme used. If the node is not joined to a network, **`FFFF`** is returned if stochastic addressing is used on the device and the user pre-configured address in the case of static addressing.
`+WSRC=?`	The command requests the range of valid addresses.
S-register	`S55` (RW)
Result codes	**`OK`** is returned if `value` is in range; otherwise, **`ERROR`** is returned.

Example	`AT+WSRC=2ABC`
	`OK`
	`AT+WSRC?`
	`+WSRC:2ABC`
	`OK`
	`AT+WSRC=?`
	`+WSRC:(0000-FFF7)`
	`OK`

Default value	`FFFF` Note: The `FFFF` default value implies that stochastic addressing will be used on the device upon network join.
Persistence	`value` is stored in the NV Memory.
Node types	Coordinator / router / end device.

5.1.10 "+WNWKPANID" - Set/Get Short (NWK) PANID

The short PAN ID value is used in ZigBee frame headers during data exchange to identify which network frames belong to. Its 16-bit value is set by the coordinator during network formation.

By default, the short PAN ID is chosen randomly and is intentionally different from the short PAN IDs of other networks present in the same location on channels specified in the channel mask.

Such behavior may lead to the following issue. If the network coordinator leaves the current network while routers continue their operation and then initiates a new network formation, it will establish a network with a short PAN ID different from the short PAN ID of its previous network. Hence, communication between the coordinator and its former network will not be possible.

To avoid the problem described above, the `+WNWKPANID` command can be used to configure a predefined short PAN ID on the network coordinator.

Table 5-10. "+WNWKPANID" - Set/Get Short (NWK) PAN ID

Syntax	Explanation
`+WNWKPANID=<value>`	The command assigns the device short (network) PAN ID of the target network for the device. `value` is a 16-bit hexadecimal number. If the node is configured as the coordinator, then upon the `+WJOIN` command, it will form the network with its short PANID equal to `value`. If `+WNWKPANID` is set to `FFFF`, the stack running on a coordinator automatically assigns the short PAN ID randomly and intentionally different from all the other networks detected while forming its own network. If `+WNWKPANID` is set to `FFFF` on an end device or a router, it will join to any network matching its `+WPANID [ref]` setting. Otherwise it will be able to join only to a network with a target short PAN ID equal to `value`. Notes: 1. The command is not accessible when the node is joined to a network. 2. Using `+WNWKPANID` does not cancel configuration of the `+WPANID` parameter, and both parameters should be used together to achieve the desired behavior.
`+WNWKPANID?`	The command returns device short PANID in the form of 16-bit hexadecimal number.
`+WNWKPANID=?`	The command requests the range of valid addresses.
S-register	Not available.
Result codes	`OK` is returned if `value` is in range; otherwise, `ERROR` is returned.

Example	`AT+WNWKPANID=3A2F`
	`OK`
	`AT+WNWKPANID?`
	`+WNWKPANID:3A2F`
	`OK`
	`AT+WNWKPANID=?`
	`+WNWKPANID:(0000-FFFF)`
	`OK`

Default value	`FFFF` – no predefined short PAN ID is used by the device.
Persistence	`value` is stored in the NV Memory.
Node types	Coordinator / router / end device.

5.2 Network Management Functions

SerialNet commands described in this section execute various network management functionality, including network join and leave operations, obtaining network topology-related information, getting link quality data, etc.

When exploring network topology, it is important to take into account the fact that due to mesh networking; only an end device node can be a child and have a dedicated parent node (coordinator or router) during its lifetime in the network. Router nodes use the coordinator or other routers only as network entry points, and are not associated as direct children after network join. However, if there is enough space in the node's neighbor table, it will contain information about neighbor coordinator/router nodes.

5.2.1 "+WJOIN" - Start/Join to the Network

Table 5-11. "+WJOIN" - Start/Join to the Network

Syntax	Explanation
`+WJOIN`	The command forces the device to form a network (for coordinator node) or join an existing network (for router or end device nodes). The desired network and device characteristics shall be set prior to the `+WJOIN` request using, if necessary, SerialNet commands from Section 5.1 Networking Parameters.
Result codes	**`OK`** is returned if network formation/join is completed successfully; **`ERROR`** is returned if failed. If the node is in the network already, it returns **`OK`** immediately.
Example	`AT+WJOIN` **`OK`**
Node types	Coordinator / router / end device.

5.2.2 "+WLEAVE" - Leave the Network

Table 5-12. "+WLEAVE" - Leave the Network

Syntax	Explanation
`+WLEAVE`	The command forces the node to leave the network. Note: Parameters stored in NV Memory persist even after the node leaves the network.
Result codes	**`OK`** is returned on process completion. If the device was not connected before starting the process, it returns **`ERROR`** immediately.
Example	`AT+WLEAVE` **`OK`**
Node types	Coordinator / router / end device.

5.2.3 "+WNWK" – Get Networking Status

Table 5-13. "+WNWK" – Get Networking Status

Syntax	Explanation	
`+WNWK`	The command requests the current networking status of the device.	
Result codes	**`OK`** is returned if the device is joined to a network; otherwise, it returns **`ERROR`**.	

Example	`AT+WLEAVE`
	`OK`
	`AT+WNWK`
	`ERROR`
		Leave the network first
		Device is not in a network now

Node types	Coordinator / router / end device.	

5.2.4 "+WPARENT" - Get Parent Address

Table 5-14. "+WPARENT" - Get Parent Address

Syntax	Explanation
`+WPARENT?`	The command requests the parent node address the device is associated with. The extended (MAC) address of the parent node is returned as a 64-bit hexadecimal number if the `S30` register is set to `0`. The short (NWK) parent address is returned if the `S30` register is set to `1`. See Section 5.2.8 "S30" - Set Node Addressing Mode for details. Note: This command does not cause network operations, and just returns a copy of the parent address assigned during the joining process.
Result codes	**`OK`** is returned if the module is in the network and has a parent. **`ERROR`** will be returned if the device is not in the connected state or has a node role of coordinator or router.
Example	`AT+WPARENT?` **`+WPARENT:0123456789DEF`** **`OK`**
Node types	End devices.

5.2.5 "+WCHILDREN" – Get Children Addresses

Table 5-15. "+WCHILDREN" – Get Children Addresses

Syntax	Explanation
`+WCHILDREN?`	The command requests the addresses of children end devices associated with the node. The extended (MAC) addresses of children nodes are returned as 64-bit hexadecimal numbers if the `S30` register is set to `0`.The short (NWK) addresses of children nodes are returned if the `S30` register is set to `1`. See Section 5.2.8 "S30" - Set Node Addressing Mode for details. The children addresses returned are delimited by commas. Notes: 1. An end device is removed from the children list if the parent node receives no poll requests from the child during *3*(sleep_interval + sync_period)* time interval as configured on the parent device by `+WPWR` and `+WSYNCPRD` commands. 2. This command does not cause network operations, and just returns copies of the children addresses stored in the parent memory.
Result codes	**`OK`** is returned if the module is in the network, even though there is no child connected yet. **`ERROR`** will be returned if the device is not in the connected state or has an end device node role.

Example	`AT+WCHILDREN?`
	`+WCHILDREN:0123456789ABCDEF,123456789ABCDEF0`
	`OK`

Node types	Coordinator and routers.

5.2.6 "+WNBSIZE" - Get Number of Neighbors

Table 5-16. "+WNBSIZE" - Get Number of Neighbors

Syntax	Explanation
`+WNBSIZE?`	The command requests a number of entries in node's neighbor table. Returned result consists of two values: the first is the current number of occupied entries in node's neighbor table; the second is the maximum possible number of entries (size of the neighbor table).

Syntax	Explanation
Result codes	**OK** is returned if the node is in the network. If the device is not in the connected state, **ERROR** will be returned.
Example	AT+WNBSIZE? **+WNBSIZE:2,5** **OK**
Node types	Coordinator / router / end device.

5.2.7 “+WNB” - Get Neighbor Information

Table 5-17. “+WNB” - Get Neighbor Information

Syntax	Explanation
+WNB <node_role> [,<device_addr>]	The command requests the contents of node’s neighbor table. node_role parameter specifies the node role of neighboring nodes to be extracted from the neighbor table. The following values are accepted: 0 – coordinator. 1 – router. 2 – end device. 3 – all device types. device_addr optional parameter that specifies the address of the neighboring node to be extracted. If the S30 register is set to 0, device_addr is accepted as the short (NWK) address. If the S30 register is set to 1, device_addr is expected to be an extended (MAC) address. See Section 5.2.8 “S30” - Set Node Addressing Mode for details. The command’s information response has the following format: **seqNr ¦ nodeRole ¦ extAddr ¦ nwkAddr ¦ relationship ¦ depth** Where **seqNr** – is the sequence number in the neighbor table **nodeRole** – is the node role of the neighbor. **extAddr** – is the neighbor’s extended address **nwkAddr** – is the neighbor’s network address **relationship** – is the neighbor’s relationship to the current node (0 – parent, 1 – child, 3 – no relationship). **depth** – is the neighbor’s network depth. Notes: 1. A neighbor entry is removed from the table if the node, during a certain interval, doesn’t receive any periodic management frames, expected from the neighbor. If neighbor is a router/coordinator, this interval is 45 seconds (management frames are sent once per 15 seconds). If the neighbor is an end device, then the interval equals 3×*(sleep_interval+sync._period),* as configured on the node by +WPWR and +WSYNCPRD commands. 2. Although right after network join an end device node can have information about several nodes in its neighbor table, only the actual parent node persists in the table while information about other nodes is removed shortly after an end device join. The same is valid for information about an end device neighbor – in a long term period, it is present only in the neighbor table of its parent, and is not directly “visible” for other routers in its neighborhood. 3. This command does not cause network operations, and just returns information from the node’s current neighbor table.
Result codes	**OK** is returned if the node is in the network. If the node is not in the connected state, **ERROR** will be returned.

Syntax	Explanation
Example	AT+WNB 3 **1 | 0 | 0000000000000001 | 0000 |3 | 2** **2 | 1 | 0000000000000002 | 0002 |0 | 1** **OK** AT+WNB 1,2 **1 | 1 | 0000000000000002 | 0002 |0 | 1** **OK**
Node types	Coordinator / router / end device.

5.2.8 “S30” - Set Node Addressing Mode

Table 5-18. “S30” - Set Node Addressing Mode

Syntax	Explanation
`S30=<value>`	The command sets the node addressing scheme to be used by some SerialNet commands. `value` specifies the addressing mode: `0` – extended (64-bit) addressing. `1` – short (16-bit) addressing.
`S30?`	The command requests the current addressing mode.
Result codes	The command returns **OK** if `value` is in range; otherwise, it returns **ERROR**.
S-register	`S30` (RW)
Example	ATS30=0 **OK** AT+WPARENT? **+WPARENT:000100000A3B98CC** **OK** ATS30=1 **OK** AT+WPARENT? **+WPARENT:0000** **OK**
Node types	Coordinator / router / end device.
Default value	`0`
Persistence	`value` is NOT stored in NV Memory.

Note: Setting the addressing mode, the `S30` command affects the performance of the following commands: `+WPARENT?` (see Section 5.2.4 “+WPARENT” - Get Parent Address), `+WCHILDREN?` (see Section 5.2.5 “+WCHILDREN” – Get Children Addresses), and `+WNB` (see Section 5.2.7 “+WNB” - Get Neighbor Information). These commands use an extended (MAC) address if `S30` is set to `0`, but will switch to using short (NWK) addressing if `S30` is set to `1`.

5.2.9 “+WLQI” - Get LQI Value

Table 5-19. “+WLQI” - Get LQI Value

Syntax	Explanation	
`+WLQI <addr>`	The command requests the LQI value for the link to the node with short (NWK) address equal to `addr` specified in 16-bit hexadecimal format. The command returns the actual LQI value in the range of `0…255`. Notes: 1. LQI information can be retrieved for links within a one-hop radius only. 2. An end device can obtain LQI only to its current parent node, and vice versa: LQI to an end device can be obtained only from its current parent node. 3. LQI value is measured during data transmission initiated by the `ATD` command. If `ATD` has not been performed yet, `+WLQI` may return an irrelevant value.	
Result codes	The node returns **`OK`** if the device is in the network and the LQI value for this particular link exists; otherwise, **`ERROR`** will be returned.	

Example	`AT+WLQI 1`
	`+WLQI:254`
	`OK`
		Request LQI for the link to the node with short address 0x0001

Node types	Coordinator / router / end device.	

5.2.10 “+WRSSI” - Get RSSI Value

Table 5-20. “+WRSSI” - Get RSSI Value

Syntax	Explanation	

`+WRSSI`
`<addr>`
	The command requests the RSSI value for the link to the node with short (NWK) address equal to `addr` specified in 16-bit hexadecimal format.
	The command returns the actual RSSI value expressed in dBm. If RSSI is not available, then the value **-91** is returned.
	Notes: 1. RSSI information can be retrieved for links within a one-hop radius only.
	2. An end device can obtain RSSI only to its current parent, and vice versa: RSSI to an end device can be obtained only from its current parent node.
	3. RSSI value is measured during data transmission initiated by the `ATD` command. If `ATD` has not been performed yet, `+WRSSI` may return an irrelevant value.	

Result codes	The node returns **`OK`** if the device is in the network and the RSSI value for this particular link exists; otherwise, **`ERROR`** will be returned.	

Example	`AT+WRSSI 0001`
	`+WRSSI:-80`
	`OK`
		Request RSSI for the link to the node with short address 0x0001
		-80dBm.

Node types	Coordinator / router / end device.	

5.3 Security

SerialNet firmware images with names having `_Security` at the end support standard security mechanism defined in the ZigBee PRO specification [5]. In this scheme, a special 128-bit key (called the network key in ZigBee PRO), the same for all devices, is used in the network to encrypt/decrypt messages with the AES-128 algorithm. The security mechanism can be enabled/disabled using the `+WSECON` command and configured as described below. If security is enabled, then the payload of all data frames sent by the device will be encrypted with the net- work key.

To employ the security mechanism, the network coordinator shall be configured as a trust center (TC) by assigning a +WTCADDR value to its extended address. The TC shall also be programmed with a network key (+WNETKEY command) prior to network start.

Joining a secured network can be organized in an unsecured or secured way using the +WSECSTATUS command. To avoid possible confusion during the network join, all nodes, including the TC, shall have the same +WSECSTATUS configuration prior to network join/start.

For an unsecured join +WSECSTATUS shall be set to 3. In this case, only the TC shall know the network key in advance, while other devices usually only have information about the TC's extended address (+WTCADDR). In such a configuration, when a new device tries to enter the network, a special message is propagated to the TC informing it about the joining device. Then, the TC replies with the transport key command that contains the current network key. However, to allow the joining device to understand this command, it is delivered over the last hop in an unencrypted frame. The joining device accepts the key if the sender address of the received transport key command matches its pre-configured TC's extended address, and rejects the key if the addresses don't match. After the key is accepted, the device is authenticated and starts exchanging data using the received network key.

A secured join completely eliminates the unencrypted key transmission performed in an unsecured join. For a secured join, nodes shall have +WSECSTATUS set to 0 and be pre-configured with the network key (+WNETKEY command) prior to a network join attempt. The +WTCADDR setting is required on the joining device only if it has +WPANID set to 0 and +WSRC set to FFFF (stochastic addressing). In such a configuration, the device expects a transport key command from the TC as in an unsecured join. But in this case, the command payload contains zeros, and, hence, doesn't reveal the secret network key.

The sections below provide detailed descriptions of the commands that shall be used to configure and operate a secured network.

5.3.1 “+WSECON” -Enable/Disable Security

Table 5-21. “+WSECON” - Enable/Disable Security

Syntax	Explanation
+WSECON=<value>	The command enables/disables security for a network join and data exchange on the device. value is a Boolean type. 1 means that the ZigBee PRO standard security mechanism is enabled on the node. 0 implies that encryption is not used by the node. Notes: 1. The command is not accessible when the node is joined to a network. 2. The command configures the CS_SECURITY_ON parameter in the BitCloud stack.
+WSECON?	The command returns value indicating whether security on the node is enabled or not.
+WSECON=?	The command requests the range of supported values.
S-register	Not available.
Result codes	**OK** is returned if value is in range; otherwise, **ERROR** is returned.
Example	AT+WSECON=1 **OK** AT+WSECON? **+WSECON:1** **OK** AT+WSECON=? **+WSECON:(0-1)** **OK**

Syntax	Explanation
Default value	`0`
Persistence	`value` is stored in the NV Memory.
Node types	Coordinator / router / end device.

5.3.2 "+WSECSTATUS" - Set/Get Security Status

Table 5-22. "+WSECSTATUS" - Set/Get Security Status

Syntax	Explanation

`+WSECSTATUS=`
`<value>`
	The command configures the security status on the devices to provided `value` as follows:
	`0` – for secured key transmission; the device should have a pre-configured, nonzero network key prior to network join.
	`3` – device obtains the network key from the trust center via unencrypted transmission over the air.
	`1, 2` – reserved for high-security mode, not supported in SerialNet.
	Notes: 1. The command is not accessible when the node is joined to a network.
	2. The command configures the `CS_ZDO_SECURITY_STATUS` parameter in the BitCloud stack.
	3. On the trust center, independently of `+WSECSTATUS` configuration, `+WNETKEY` shall be set to the non-zero network key to be used for encryption.

`+WSECSTATUS?`	The command returns device security status.
`+WSECSTATUS=?`	The command requests the range of valid statuses.
S-register	Not available.
Result codes	**`OK`** is returned if `value` is in range; otherwise, **`ERROR`** is returned.

Example	`AT+WSECSTATUS=3`
	`OK`
	`AT+WSECSTATUS?`
	`+WSECON:3`
	`OK`
	`AT+WSECSTATUS=?`
	`+WSECSTATUS:(0-3)`
	`OK`

Default value	`0`
Persistence	`value` is stored in the NV Memory.
Node types	Coordinator / router / end device.

5.3.3 "+WNETKEY" - Set/Get Network Encryption Key

Table 5-23. "+WNETKEY" - Set/Get Network Encryption Key

Syntax	Explanation
`+WNETKEY=<val64bit0,val64bit1>`	The command assigns the network key on the device. `val64bit0` and `val64bit1` are 64-bit hexadecimal numbers that compose the 128-bit secret encryption key to be used for data encryption/decryption in the network. Notes: 1. The command is not accessible when the node is joined to a network. 2. Zero value implies no encryption to be used. 3. The command configures the `CS_NETWORK_KEY` parameter in the BitCloud stack.
`+WNETKEY?`	The command returns the network key configuration on the node.
`+WNETKEY=?`	The command requests the range of valid values for the network key.
S-register	Not available.
Result codes	**OK** is returned if `val64bit0` and `val64bit1` are in range; otherwise, **ERROR** is returned.

Example	`AT+WNETKEY=CCCCCCCCCCCCCCCC, BBBBBBBBBBBBBBBB`
	OK
	`AT+WNETKEY?`
	+WNETKEY:CCCCCCCCCCCCCCCC, BBBBBBBBBBBBBBBB OK
	`AT+WNETKEY=?`
	+WNETKEY:(0000000000000000-FFFFFFFFFFFFFFFF),(0000000000000000-FFFFFFFFFFFFFFFF)
	OK

Default value	`0000000000000000,0000000000000000`
Persistence	`val64bit0` and `val64bit1` are stored in the NV Memory.
Node types	Coordinator / router / end device.

5.3.4 "+WTCADDR" - Set/Get Trust Center Address

Table 5-24. "+WTCADDR" - Set/Get Trust Center Address

Syntax	Explanation
`+WTCADDR=<value>`	The command assigns the trust center extended address. `value` is a 64-bit hexadecimal number that defines the extended address of the trust center node in the network. The trust center itself shall set it to its own extended address. The value shall be configured on the device prior to network join. If the device cannot know the TC address in advance, it can use a `FFFFFFFFFFFFFA` value instead. In this case, the device considers the first node replying with the transport key command during network join to be the network's TC. Notes: 1. The command is not accessible when the node is joined to a network. 2. The command configures the `CS_APS_TRUST_CENTER_ADDRESS` parameter in the BitCloud stack.
`+WTCADDR?`	The command returns the trust center address configured on the node.
`+WTCADDR=?`	The command requests the range of valid values.
S-register	Not available.
Result codes	**OK** is returned if `value` is in range; otherwise, **ERROR** is returned.

Syntax	Explanation
Example	AT+WTCADDR=000100001090C96D **OK** AT+WTCADDR? **+WTCADDR:000100001090C96D** **OK** AT+WTCADDR=? **+WTCADDR:(0000000000000001-FFFFFFFFFFFFFFFE)** **OK**
Default value	AAAAAAAAAAAAAAAA
Persistence	value is stored in the NV Memory.
Node types	Coordinator / router / end device.

5.4 Data Transmission

In SerialNet, data can be transmitted in two ways:

- Unicast transmission to a particular node using the D, DS, or +WPING commands
- Broadcast transmission to all nodes using the DU or D commands with broadcast address

It is important that extended (MAC) addresses are not used for data transmission directly; instead, they are substituted by short (network) addresses that are convenient for node replacement in network installation and maintenance.

The route establishment procedure to the target node is implemented inside the stack. It is executed automatically upon a data transmission request, and then if a route exists, data delivery (one-hop or multi-hop) is performed to the destination node.

The following application identifiers are used in SerialNet for all data exchange operations:

Profile ID:	0xC31A
Endpoint ID:	0x01
Cluster ID:	0x00

Note: To ensure safe data transmission over a serial interface between a host and an MCU, it is strongly recommended to set hardware flow control (see Section 5.7.9 “+IFC” - Serial Port Flow Control for details). When running terminal software to control the node, the chosen COM port should be set with the hardware flow control option selected.

5.4.1 Parent Polling Mechanism

Data delivery to an end device over the last hop (that is, from the parent node to the child) is performed using the polling mechanism described below.

Upon receiving a frame destined for its child node or a broadcast frame with a non-exhausted transmission radius and a destination address equal 0xFFFF, the parent node buffers the frame and waits for a poll request from the child. The maximum waiting time is (sleep_interval+3×sync._period), as configured on the parent by the +WPWR and +WSYNCPRD commands.

In awake state, an end device polls its parent node periodically every +WSYNCPRD milliseconds (as configured on the end device). The parent node can transmit a data frame to a child only after receiving a corresponding data poll from it. After data frame reception is completed, the end device issues another data poll request to verify whether there are any frames buffered at the parent.

5.4.2 “D” - Send Data to a Specific Node

Table 5-25. “D” - Send Data to a Specific Node

Syntax	Explanation	

`D <addr>`
`[,[<arq>]`
`[,<length>]]`
`<data>`
	The command sends data to a specific node.
	`addr` is the 16-bit hexadecimal short (network) address of the destination node.
	An optional `arq` parameter (equal to `1` or `0`) controls the ARQ/nonARQ data delivery mode, with `1` (that is, ARQ) as the default, if omitted.
	The `length` parameter specifies the length in bytes of the data portion to be sent. It shall not exceed the maximum allowable number (95 bytes for an un-encrypted frame, and 77 bytes for an encrypted frame). If the `length` parameter is omitted, the maximum possible value is implied by default.
	Data transmission starts either when the specified number of data bytes is received over the serial interface, or when the time interval between two consecutive data symbols exceeds the timeout preset (see Section 5.4.10 “+WWAIT” - Data Transmission Waiting Timeout).
	Notes: 1. `data` should be preceded by `<CR>`(`S3` character, see Section 5.7.1 “S3” - Termination Character). This symbol is not transmitted over the air, and it is not counted in `length`.
	2. If the destination address is a broadcast address (`FFFF` for all nodes or `FFFD` for router/coordinator nodes), the broadcast transmission is performed but only when `arq` is `0`.	

Result codes	If data transmission requires acknowledgement from the destination node (unicast transmission and `arq` is set to `1`), then `OK` is returned only after such acknowledgement frame is received. If no acknowledgement is received after `+WRETRY` transmission attempts (see Section 5.4.9 “+WRETRY” – Data Retries Amount), then `ERROR` is returned as a result code for the command. If an acknowledgement is not requested (`arq` is set to `0`), then `OK` is returned just upon successful data transmission over the first hop. There is no guarantee on frame delivery to the final destination. If first-hop transmission cannot be performed (channel busy, route unknown, etc.), or if the sending node is not in the network, then `ERROR` is returned.	

Example	`ATD 12,1,5 HELLO`
	OK
	`ATD 12 HELLO`
	OK
		Send `HELLO` to the node with address 12 using ARQ.
		The same as above, but the node will be waiting for the timeout expiration before going to the air.

Node types	Coordinator / router / end device.	

5.4.3 "DB" - Send Binary Data to a Specific Node

Table 5-26. "DB" - Send Binary Data to a Specific Node

Syntax	Explanation	

`DB <addr>`
`[,<arq>`
`[,<length>]]`
`<data>`
	The command sends binary data (not encoded in ASCII symbols) to a specific node.
	`addr` is the 16-bit hexadecimal short (network) address of the destination node.
	An optional `arq` parameter (equal to `1` or `0`) controls the ARQ/nonARQ data delivery mode. If omitted, then ARQ is used.
	The `length` parameter specifies the length in bytes of the data portion to be sent. It shall not exceed the maximum allowable number (95 bytes for an un-encrypted frame, and 77 bytes for an encrypted frame). If the `length` parameter is omitted, the maximum possible value is implied by default.
	`data` should be preceded by `<CR>` (`S3` character, see Section 5.7.1 "S3" - Termination Character). This symbol is not included in frame payload to be transmitted over the air, and it is not counted in `length`. Data transmission starts either when the specified number of data bytes is received over the serial interface, or when the time interval between two consecutive data symbols exceeds the timeout preset (see Section 5.4.10 "+WWAIT" - Data Transmission Waiting Timeout). In contrast to the `D` command, data transmission doesn't start unplanned if a `<CR>` code (or `S3` character, see Section 5.7.1 "S3" - Termination Character) occurs inside the `data`.
	Note: If the destination address is a broadcast address (`FFFF` for all nodes or `FFFD` for router/coordinator nodes), the broadcast transmission is performed but only when `arq` is `0`.	

Result codes	If data transmission requires an acknowledgement from the destination node (unicast transmission and `arq` is set to `1`), then an **OK** result code is returned only after such acknowledgement frame is received. If no acknowledgement is received after `+WRETRY` transmission attempts (see Section 5.4.9 "+WRETRY" – Data Retries Amount), then **ERROR** is returned as a result code. If an acknowledgement is not requested (`arq` is set to `0`), then **OK** is returned just upon successful data transmission over the first hop. There is no guarantee on frame delivery to the final destination. If first-hop transmission cannot be performed (channel busy, route unknown, etc.), or if the sending node is not in the network, **ERROR** is returned.	

Example	`ATDB 12,1,5 97CA2`
	OK
	`ATDB 12,0`
	`97CA2`
	OK
		Send `97CA2` to the node with address 12 using ARQ.
		The same as above, but without using ARQ, and with the node waiting for timeout expiration before sending data.

Node types	Coordinator / router / end device.	

5.4.4 “DU” - Send Broadcast Data

Table 5-27. “DU” - Send Broadcast Data

Syntax	Explanation	
DU [<length>] <data>	The command sends data using broadcast transmission. The length parameter specifies the length in bytes of the data portion to be sent. It shall not exceed the maximum allowable number (95 bytes for an un-encrypted frame, and 77 bytes for an encrypted frame). If the length parameter is omitted, the maximum possible value is implied by default. Data transmission starts either when the specified number of data bytes is received over the serial interface, or when the time interval between two consecutive data symbols exceeds the timeout preset (see Section 5.4.10 “+WWAIT” - Data Transmission Waiting Timeout). Notes: 1. ATDU is shorthand for the ATD command with broadcast address (FFFF) as the destination and arq set to 0. 2. Data should be preceded by <CR> (S3 character, see Section 5.7.1 “S3” - Termination Character). This symbol is not transmitted over the air, and it is not counted in length. 3. Data are broadcasted to the whole network (radius 0).	
Result codes	The node responds with **OK** immediately after the transmission if the node itself is in the network. Otherwise, **ERROR** is returned.	
Example	ATDU HELLO **OK**	Send **HELLO** to all nodes in the network.
Node types	Coordinator / router / end device.	

5.4.5 “DS” - Send S-register Value to a Specific Node

Table 5-28. “DS” - Send S-register Value to a Specific Node

Syntax	Explanation	
DS <S-reg>,<addr>[,[<arq>]	The command sends the S-register value to a specific node. The default arq parameter (set to 1 or 0) specifies whether the ARQ or non-ARQ data delivery mode is used. 1 is implied if arq is omitted. The addr destination node address should be a 16-bit hexadecimal short (network) address. S-register data are sent in the form readable by ATS command without the line termination characters. Note: S-registers defined by user extensions are also accessible by this command.	
Result codes	If an acknowledgement is requested (arq is set to 1), the node responds with OK upon receiving an acknowledgement in specified amount of attempts (see Section 5.4.9 “+WRETRY” – Data Retries Amount); otherwise, it returns **ERROR**. If the destination node or the sending node itself is not in the network, **ERROR** is returned. Also, if the specified S-register cannot be read, the command returns **ERROR**, and the node does not send anything to the air.	
Example	ATDS130,2,0 **OK**	Send GPIO0 value to the node with address 2 without using ARQ.
Node types	Coordinator / router / end device.	

5.4.6 “+WPING” - Ping the Node

Table 5-29. “+WPING” - Ping the Node

Syntax	Explanation
`+WPING <addr>`	The command pings the targeted node. `addr` specifies the destination address as a 16-bit hexadecimal short (network) address. This command is equivalent to the `D` command with ARQ and zero data length: `ATD <addr>,1,0`.
Result codes	The node responds with **OK** upon receiving an acknowledgement in specified amount of attempts (see Section 5.4.9 “+WRETRY” – Data Retries Amount); otherwise, it returns **ERROR**. If the destination node or the sending node itself is not in the network, **ERROR** is returned.
Example	`AT+WPING 1` **OK**
Node types	Coordinator / router / end device.

5.4.7 “+WSYNCPRD” - Poll Rate for Requesting Indirect Transactions from the Parent

Table 5-30. “+WSYNCPRD” - Poll Rate for Requesting Indirect Transactions from the Parent

Syntax	Explanation	
`+WSYNCPRD=<rate>`	The command sets the poll interval to the `rate` value, measured in milliseconds. This value is used by the end device as the poll rate for requesting indirect transmission messages from the parent. Coordinator and router use this rate to verify children presence. Notes: 1. On end devices, the `rate` value must not be increased by this command. Otherwise, BitCloud behavior is unpredictable. 2. On routers and coordinators, this parameter must be set to the largest `rate` value among all children. Otherwise, child presence status may be detected incorrectly. 3. This value should be at least two times smaller than the value of `+WTIMEOUT` (see Section 5.4.8 “+WTIMEOUT” - Data Delivery Timeout).	
`+WSYNCPRD?`	The command requests the actual poll rate.	
`+WSYNCPRD=?`	The command requests the allowable range of poll rate values.	
S-registers	`S37` (RW)	
Result codes	**OK** is returned if parameter is within the valid range. Otherwise, **ERROR** is returned.	

Example	`AT+WSYNCPRD=500`
	OK
	`ATS37?`
	1000
	OK
	`AT+WSYNCPRD=?`
	+WSYNCPRD:(10-30000)
	OK
		Set poll rate to 500 milliseconds

Default values	`1000`	
Node types	Coordinator / router / end device.	
Persistence	`rate` is NOT stored in NV Memory.	

5.4.8 "+WTIMEOUT" - Data Delivery Timeout

Table 5-31. "+WTIMEOUT" - Data Delivery Timeout

Syntax	Explanation
+WTIMEOUT?	The command returns the timeout value, in milliseconds. The returned value corresponds to the apscAckWaitDuration variable introduced by ZigBee recommendation [5].
S-register	S51 (R)
Result codes	**OK** is always returned.
Example	AT+WTIMEOUT? **+WTIMEOUT:2800** **OK**
Node types	Coordinator / router / end device.

5.4.9 "+WRETRY" – Data Retries Amount

Table 5-32. "+WRETRY" – Data Retries Amount

Syntax	Explanation	
+WRETRY= <value>	The command sets value to the maximum amount of retransmission attempts that will be automatically performed by the stack for unicast data transmission with ARQ until delivery is confirmed by the final destination. Notes: 1. Retries follow original failed transmission attempt and continue until successful delivery or until value amount of retries is reached. 2. The command configures only APS (end-to-end) level retransmissions. Lower ZigBee layers (NWK/MAC) might perform their own retries before indicating failure to the upper APS layer. Thus total amount of frames resent over the air will be larger than value. 3. ZigBee PRO standard requires that this parameter is set to 3 for compliancy tests.	
+WRETRY?	The command returns the number of retransmissions to be performed for unicast data with ARQ enabled.	
+WRETRY=?	The command requests the range of valid values for maximum amount of retries.	
S-register	S52 (RW)	
Result codes	**OK** is returned if value is within the valid range. Otherwise, **ERROR** is returned.	
Example	AT+WRETRY=1 **OK** AT+WRETRY? **+WRETRY:1** **OK**	Setting +WRETRY to 1.
	AT+WRETRY=? **+WRETRY:(1-5)** **OK**	Getting range of valid values.
Default value	3	
Persistence	value is NOT stored in the NV Memory.	
Node types	Coordinator / router / end device.	

5.4.10 “+WWAIT” - Data Transmission Waiting Timeout

Table 5-33. “+WWAIT” - Data Transmission Waiting Timeout

Syntax	Explanation
`+WWAIT=<value>`	The `value` parameter sets the timeout (in milliseconds) for the module to wait for entering the `D` (see Section 5.4.2 “D” - Send Data to a Specific Node) or the `DU` (see Section 5.4.4 “DU” - Send Broadcast Data) command. If a pause between two consecutive characters coming from the serial interface exceeds the specified timeout, the node will start data transmission even though the data length encountered has not yet reached the number specified by the `length` argument of the D/`DU` command. In this case, the `length` is replaced with its actual value according to the data transmitted.
`+WWAIT?`	The command returns actual timeout `value`.
`+WWAIT=?`	The command requests the range of valid timeouts.
S-register	`S53` (RW)
Result codes	**`OK`** is returned if the `value` is in the range; otherwise, **`ERROR`** is returned.

Example	`AT+WWAIT=500`
	`OK`
	`AT+WWAIT?`
	`+WWAIT:500`
	`OK`
	`AT+WWAIT=?`
	`+WWAIT:(100-5000)`
	`OK`

Default value	`5000`
Persistence	`value` is stored in the NV Memory.
Node types	Coordinator / router / end device.

5.5 Power Management

Because power consumption is a major concern in applications with battery-powered devices, SerialNet provides AT commands that allow switching between awake and sleep modes, as well as setting the transmit power level.

To avoid issues in network stability, coordinator and router nodes are always kept in active mode, and, hence, require continuous power supply.

In addition to power management of the device itself, SerialNet simplifies power management of external peripherals or the host device via the CTS line. If hardware flow control is enabled by the `+IFC` command (see Section 5.7.9 “+IFC” - Serial Port Flow Control), the line becomes high when the node is in the sleep state.

5.5.1 “+WPWR” - Configuration of Sleep/Active Intervals

Table 5-34. “+WPWR” - Configuration of Sleep/Active Intervals

<table>
<tr><th>Syntax</th><th colspan="2">Explanation</th></tr>
<tr><td>+WPWR=<sleep>,
<active&gt;</td><td colspan="2">The command sets the duration of sleep and active intervals for end device nodes.
The sleep duration is specified in 100ms units, but active duration is specified in 10ms units. Zero active period means that the node can be put asleep explicitly only by the +WSLEEP command (in which case it will stay asleep for the given sleep duration).
On a coordinator/router node, sleep interval is used for children tracking, and should be not less than on its children nodes. It is also used as the maximum time interval the data destined for the child can be buffered. See Section 5.4.1 Parent Polling Mechanism for more details.
Notes: 1. Active interval is not guaranteed because going to sleep is not always possible during ongoing data transactions. Thus it is recommended to keep active interval as 0 and control active time from the host using +WSLEEP command. Also host can check the state of the SerialNet device via status of CTS line.
2. Actual resolution of sleep time is ~32ms. Sleep time will be set to the next higher multiple of 32ms. For example in case of 100ms configuration actual sleep time will be 4*32=128ms.
3. The command is not accessible when the node is joined to a network.</td></tr>
<tr><td>+WPWR?</td><td colspan="2">The command requests current sleep/active intervals.</td></tr>
<tr><td>+WPWR=?</td><td colspan="2">The command requests valid ranges of sleep/active intervals.</td></tr>
<tr><td>S-registers</td><td colspan="2">S31, S32 (RW)</td></tr>
<tr><td>Result codes</td><td colspan="2">OK is returned if parameters are within their valid ranges. Otherwise, ERROR is returned.</td></tr>
<tr><td>Example</td><td>AT+WPWR=600,0
OK
AT+WPWR?
+WPWR:600,0
OK
ATS31?
600
OK
AT+WPWR=?
+WPWR:(2-30000),(0-30000)
OK</td><td>Set sleep time to 60 seconds

Verify setting is applied

Get sleep interval via S-register

Get valid ranges for sleep/active intervals</td></tr>
<tr><td>Default values</td><td colspan="2">100,0 (the node sleeps for ten seconds if put asleep by the +WSLEEP command).</td></tr>
<tr><td>Persistence</td><td colspan="2">The sleep,active values are stored in the NV Memory.</td></tr>
<tr><td>Node types</td><td colspan="2">Coordinator / router / end device.</td></tr>
</table>

5.5.2 “+WSLEEP” - Force Node to Sleep

Table 5-35. “+WSLEEP” - Force Node to Sleep

Syntax	Explanation
`+WSLEEP`	The command forces the node into the sleep mode. The node in sleep mode can respond to the subsequent commands with a delay, depending on the sleeping interval specified (see Section 5.5.1 “+WPWR” - Configuration of Sleep/Active Intervals) and DTR configuration (see Section 5.7.10 “&D” - DTR Behavior).
Result codes	**`OK`** is returned for end devices; otherwise, **`ERROR`** is returned. Note: The command is executed as follows: the node returns the result code first, and then it disables any subsequent commands, completes pending operations, and finally goes into the sleep mode. Wake up occurs as scheduled by the `+WPWR` command or DTR interrupt, if enabled.
Example	`AT+WSLEEP` **`OK`**
Node types	End devices.

5.5.3 “+WTXPWR” - TX Power Level

Table 5-36. “+WTXPWR” - TX Power Level

Syntax	Explanation	

`+WTXPWR=`
`<value>`
	The command sets the transmit power level for the device.
	The `value` represents the TX power level, measured in dBm.	

`+WTXPWR?`	The command requests the actual TX power level. Note: Power level resolution is hardware dependent, and may be coarser than 1dB, and so some power values (-4, -6, -8…) may be forbidden despite being within the allowed range. On input, such values are rounded to the nearest allowed value.	
`+WTXPWR=?`	The command requests the allowable range of TX power levels.	
S-register	`S34` (RW)	
Result codes	**`OK`** is returned if `value` is in the valid range and was set to RF correctly; otherwise, **`ERROR`** is returned.	

Example	`AT+WTXPWR=-5`
	`OK`
	`AT+WTXPWR?`
	`+WTXPWR:-5`
	`OK`
	`AT+WTXPWR=?`
	`+WTXPWR:(-17-3)`
	`OK`
		Set -5 dBm TX power level
		Allowed range is from -17 to +3dBm.

Default value	Hardware dependent. Typically maximum value is set.	
Persistence	`value` is stored in the NV Memory.	
Node types	Coordinator / router / end device.	

5.6 Generic Control

5.6.1 “Z” - Warm Reset

Table 5-37. “Z” - Warm Reset

Syntax	Explanation
`Z`	The command instructs the device to execute a warm (software) reset. This command resets the hardware, restores all persistent variables from NV Memory, and restarts the SerialNet firmware. Notes: 1. The command does not issue “leaving the network” signals to other nodes, and, hence, can affect PAN integrity. Therefore, it is better to put the node out of the current network prior to reset using the `+WLEAVE` command (see Section 5.2.2 “+WLEAVE” - Leave the Network). 2. If automatic networking is enabled, then the node will automatically initiate network join after reset. 3. Parameters stored in NV Memory persist after software reset; to erase them, use the `&F` command (see Section 5.6.8 “&F” – Set to Factory Default Configuration). The result code is sent after the reset process is completed and the device is ready for control again. During the reset process, some transients may be observed on the module pins (including GPIO) because of the nature of the MCU used. It is strongly recommended to wait until the **OK** result code is received (or an equivalent numerical code) before sending any new command to the module.
Result codes	**OK** is returned after reset process is completed.
Example	`ATZ` **OK**
Node types	Coordinator / router / end device.

5.6.2 “&H” - Command Help

Table 5-38. “&H” - Command Help

Syntax	Explanation
`&H`	The command outputs a list of valid AT-commands. The listing order may change, depending on the firmware version.
Result codes	**`OK`** is always returned.

Example	`AT&H`
	`E`
	`V`
	`Q`
	`Z`
	`&F`
	`+IPR`
	`+IFC`
	`&D`
	`&H`
	`%H I`
	`+GMI`
	`+GMM`
	`+GMR`
	`+GSN`
	`(skipped…)`
	`S146`
	`S147`
	`S148`
	`OK`

Node types	Coordinator / router / end device.

5.6.3 “%H” - Display Parameters and S-register Values

Table 5-39. “%H” - Display Parameters and S-register Values

Syntax	Explanation
`%H`	The command outputs the values of parameters and S-registers. The listing order may change, depending on the firmware version.
Result codes	`OK` is always returned.

Example	`AT%H`
	`+WPANID: 0000000000000000`
	`+WCHAN: FF`
	`+WCHMASK: 00000800`
	`+WAUTONET: 0`
	`+WPWR: 100,1000`
	`+WROLE: 2`
	`+WSRC: 0001`
	`+WSYNCPRD: 1400`
	`+WTXPWR: 0`
	`+WTIMEOUT: 2800`
	`+WRETRY: 3`
	`+WWAIT: 5000`
	`E: 1`
	`Q: 0`
	`V: 1`
	`X: 0`
	`+IPR: 38400`
	`+IFC: 0,0`
	`+GMI: ATMEL`
	`+GMM: ZIGBIT`
	`+GMR: BitCloud v.3.1.0; SerialNet v.3.0.0`
	`+GSN: 0001000011672CFC`
	(skipped…)
	`S146:0`
	`S147:0`
	`S148:0`
	`OK`

Node types	Coordinator / router / end device.

5.6.4 “I” - Display Product Identification Information

Table 5-40. “I” - Display Product Identification Information

<table>
<tr><th>Syntax</th><th colspan="3">Explanation</th></tr>
<tr><td rowspan="6"><code>I[&lt;value&gt;]</code></td><td colspan="3">The command instructs the node to return information text identifying the device. Information text depends on <code>value</code>, as follows:</td></tr>
<tr><td>Value</td><td>Information text</td><td>Reference</td></tr>
<tr><td>0</td><td>All the identifiers below</td><td></td></tr>
<tr><td>1</td><td>Manufacturer identifier</td><td>Section 5.6.5</td></tr>
<tr><td>2</td><td>Model identifier</td><td>Section 5.6.6</td></tr>
<tr><td>3</td><td>Hardware/software revision identifier</td><td>Section 5.6.7</td></tr>
<tr><td></td><td>4</td><td>Product serial number identifier</td><td>Section 5.1.8</td></tr>
</table>

Syntax	Explanation
	If `value` is omitted, 0 is implied by default.
Result codes	**OK** for any of the aforementioned values; **ERROR** otherwise.
Example	ATI0 **ATMEL** **MEGARF-ZIGBIT** **BitCloud v.3.1.0; SerialNet v.3.0.0 000100001090C3F9** **OK**
Node types	Coordinator / router / end device.

5.6.5 "+GMI" - Get Manufacturer Identifier

Table 5-41. "+GMI" - Get Manufacturer Identifier

<table>
<tr><th>Syntax</th><th colspan="2">Explanation</th></tr>
<tr><td>+GMI?
I1</td><td colspan="2">The command instructs the node to output information text identifying the manufacturer.</td></tr>
<tr><td>Result codes</td><td colspan="2"><b>OK</b> is always returned.</td></tr>
<tr><td>Example</td><td>AT+GMI?
<b>+GMI:ATMEL</b>
<b>OK</b>
ATI1
<b>ATMEL</b>
<b>OK</b></td><td>Just an alias to +GMI</td></tr>
<tr><td>Node types</td><td colspan="2">Coordinator / router / end device.</td></tr>
</table>

5.6.6 "+GMM" - Request for the Model Identifier

Table 5-42. "+GMM" - Request for the Model Identifier

<table>
<tr><th>Syntax</th><th colspan="2">Explanation</th></tr>
<tr><td>+GMM?
I2</td><td colspan="2">The command instructs the node to transmit information text identifying the particular model of the device.</td></tr>
<tr><td>Result</td><td colspan="2">OK is always returned.</td></tr>
<tr><td>Example</td><td>AT+GMM?
<b>+GMM:MEGARF-ZIGBIT</b>
<b>OK</b>
<b>ATI2</b>
<b>MEGARF-ZIGBIT</b>
<b>OK</b></td><td>Just an alias to +GMM</td></tr>
<tr><td>Node types</td><td colspan="2">Coordinator / router / end device</td></tr>
</table>

5.6.7 “+GMR” - Request for the Hardware/Software Revision Identifier

Table 5-43. “+GMR” - Request for the Hardware/Software Revision Identifier

Syntax	Explanation	
`+GMR?` `I3`	This command instructs the node to transmit an information text intended to identify the actual revision of hardware or software product burned into the device.	
Result codes	**`OK`** is always returned.	

Example	`AT+GMR?`
	`+GMR: BitCloud v. 3.2.0; SerialNet v.3.0.0`
	`OK`
	`ATI3`
	`BitCloud v. 3.2.0; SerialNet v.3.0.0`
	`OK`
		Just an alias to `+GMR`

Node types	Coordinator / router / end device.	

5.6.8 “&F” – Set to Factory Default Configuration

Table 5-44. “&F” – Set to Factory Default Configuration

Syntax	Explanation
`&F`	The command instructs the module to set all the parameters (including the persistent variables from Internal Flash) to the factory defaults. This command forces hardware reset just like the `Z` command, and so all the same precautions should be considered. The result code will be issued according to the actual result code suppression setting (see Section 5.7.5 “Q” - Result Code Suppression), response formatting (see Section 5.7.6 “V” - Response Format), and the transmission rate (see Section 5.7.8 “+IPR” - Serial Port Communication Rate) set before execution of this command. Note: `&F` command does not reset the remote management password once it has been set (see Section 5.9.1 “+WPASSWORD” - Set a Password).
Result codes	**`OK`** is always returned.
Example	`AT&F` **`OK`**
Node types	Coordinator / router / end device.

5.6.9 “+WACALIBRATE” - Configure Periodic Internal Clock Calibration

Table 5-45. “+WACALIBRATE” - Configure Periodic Internal Clock Calibration

Syntax	Explanation
`+WACALIBRATE=<value>`	The command requests the device to automatically calibrate the internal clock. `value` is an unsigned integer that determines the period of calibration, in minutes (that is, how many minutes will elapse between consecutive calibrations). The command can be used to prevent frequency drift of the MCU’s internal RC oscillator with temperature. Such drift can impact or even block serial communication with the host.
`+WACALIBRATE?`	The command returns the period of calibration (in minutes).

Syntax	Explanation
`+WACALIBRATE=?`	The command returns the permitted range of values for the period of calibration.
Result codes	**`OK`** is returned on successful command completion. Otherwise, `value` is ignored, and the device responds with **`ERROR`**.

Example	`AT+WACALIBRATE=60`
	`OK`
	`AT+WACALIBRATE?`
	`+WACALIBRATE:60`
	`OK`
	`AT+WACALIBRATE=?`
	`+WACALIBRATE(0-65535)`
	`OK`

Default value	`0`
Persistence	The value is stored in the NV Memory.
Node types	Coordinator / router / end device.

5.6.10 "+WCALIBRATE" - Calibrate Internal Clock

Table 5-46. "+WCALIBRATE" - Calibrate Internal Clock

Syntax	Explanation
`+WCALIBRATE`	The command requests the device to calibrate the internal clock. The command shall be used to prevent frequency drift of the MCU's internal RC oscillator with temperature. Such drift can impact or even block serial communication with the host.
Result codes	**`OK`** is returned on successful calibration. Otherwise, the device responds with **`ERROR`**.
Example	`AT+WCALIBRATE` **`OK`**
Node types	Coordinator / router / end device.

5.7 Host Interface Commands

5.7.1 "S3" - Termination Character

Table 5-47. "S3" - Termination Character

Syntax	Explanation
`S3=<value>`	The command sets the ASCII code to be used as the termination character in command line, response, and result code formatting. The description of the `V` command shows the parameter usage (see Section 5.7.6 "V" - Response Format for details). Section 4.6 Device Responses provides a general overview of the response formatting. `value` may be specified in the range of `0…127`. Note: It is strongly recommended to avoid changing this parameter during network operation.
`S3?`	The command requests the actual ASCII code currently used as the termination character.

Syntax	Explanation
Result codes	The module returns **OK** if `value` is in range; otherwise, **ERROR** is returned. Note: The current value of `S3` is used to terminate the command line when entering the `S3` setting command to specify a new command line termination character. However, the result code will use the new value of `S3`-specified in the command line. For example, if `S3` is currently set to `13` when the '`ATS3=30`' command line is issued, the command line must be terminated with a **<CR>** character, but the result code will use the character with a decimal value of `30`.

Example	`ATS3=13`
	OK
	`ATS3?`
	13
	OK

Node types	Coordinator / router / end device.
Default value	`13` – **<CR>** (carriage return character).
Persistence	`value` is stored in the NV Memory.

5.7.2 "S4" - Response Formatting Character

Table 5-48. "S4" - Response Formatting Character

Syntax	Explanation
`S4=<value>`	The command sets the ASCII code of the character used in response and result code formatting along with the `S3` parameter (see Section 5.7.1 "S3" - Termination Character). The description of the `V` command shows the parameter usage (see Section 5.7.6 "V" - Response Format for details). Section 4.6 Device Responses provides a general overview of the response formatting. `value` may be specified in the range of `0…127`. Note: It is strongly recommended to avoid changing this parameter during network operation.
`S4?`	The command requests the actual ASCII code currently used as the response formatting character.
Result codes	The module returns **OK** if `value` is in the allowed range, and **ERROR** otherwise. Note: The changed value of `S4` will be used to format the result code and information responses immediately after processing the '`S4=<value>`' command. If the value of `S4` is changed in a command line, the result code issued in response to that command line will be formatted using the new value of `S4`.

Example	`ATS4=10`
	OK
	`ATS4?`
	10
	OK

Node types	Coordinator / router / end device.
Default value	`10` – `<LF>` (Line Feed character).
Persistence	`value` is stored in the NV Memory.

5.7.3 “S5” - Command Editing Character

Table 5-49. “S5” - Command Editing Character

Syntax	Explanation
`S5=<value>`	The command sets the ASCII code used as the control character to delete the immediately preceding character in the command line (see Section 4.3 Basic Command-line Operations). `value` may be specified in the range of `0…127`. Note: It is strongly recommended not to set this parameter to any letter or other symbol that can be a part of an AT command. For example, setting it to letter `A`, either upper- or lowercase (ASCII code 65 or 97), would effectively prevent the entering of any subsequent AT- command.
`S5?`	The command requests the actual ASCII code of the command editing character.
Result codes	The module returns **OK** if `value` is in range; otherwise, **ERROR** is returned. Note: The new value of `S5` will be used when editing of subsequent command lines, and will be applied after processing the line containing the `S5` register change.

Example	`ATS5=8`
	OK
	`ATS5?`
	8
	OK

Node types	Coordinator / router / end device.
Default value	`8` – `<BS>` (backspace character).
Persistence	`value` is stored in the NV Memory.

5.7.4 “E” - Command Echo

Table 5-50. “E” - Command Echo

Syntax	Explanation	
`E[<value>]`	This parameter instructs the module to echo characters received by the UART. `value` may be specified as `0` or `1` to disable or enable echoing respectively. If `value` is omitted, `0` is implied.	
Result codes	The module returns **OK** if `value` is 0 or 1; otherwise. **ERROR** is returned.	
Example	`ATE` **OK**	Disable echo
	`ATE1` **OK**	Enable echo
Node types	Coordinator / router / end device.	
Default value	`1` – echoing is enabled.	
Persistence	`value` is stored in the NV Memory.	

5.7.5 “Q” - Result Code Suppression

Table 5-51. “Q” - Result Code Suppression

Syntax	Explanation	
`Q[<value>]`	This parameter instructs the module to transmit result codes to the UART. When result codes are being suppressed, no portion of any intermediate, final, or unsolicited result code – header, result text, line terminator, or trailer (see Section 3.3 SerialNet Result Codes) – is transmitted. Information text transmitted in response to a command is not affected by the setting of this parameter. There are two possibilities for `value`: `0` – The module transmits result codes `1` – Result codes are suppressed and not transmitted If `value` is omitted, `0` is implied.	
Result codes	Nothing will be received for the `ATQ1` command. **`OK`** if `value` is 0; otherwise, the module returns **`ERROR`**.	
Example	`ATQ0` **`OK`**	Enable result codes
	`ATQ1`	Suppress result codes. No **`OK`** will be sent because it is suppressed
Node types	Coordinator / router / end device.	
Default value	`0` – enables result codes.	
Persistence	`value` is stored in the NV Memory.	

5.7.6 “V” - Response Format

Table 5-52, summarizes the usage of response formats. All references to `<CR>` mean the character ASCII code specified in parameter `S3` (see Section 5.7.1 “S3” - Termination Character); likewise, all references to `<LF>` mean the character ASCII code specified in parameter `S4` (see Section 5.7.2 “S4” - Response Formatting Character). Numeric and verbose values for result codes are listed in Section 3.3 SerialNet Result Codes.

Table 5-52. Response Formatting

Device response type	Response format for particular `V` value	
	0	1
Information responses	`<text><CR><LF>`	`<CR><LF><text><CR><LF>`
Result codes	`<numeric code><CR>`	`<CR><LF><verbose code><CR><LF>`

Table 5-53. “V” - Response Format

Syntax	Explanation	
V[<value>]	This parameter defines the contents of the header and trailer transmitted with result codes and information responses (see Section 4.6 Device Responses). value specifies whether result codes are transmitted in numeric or verbose (alphabetic) form as follows: 0 – The result code shall be in numeric format. 1 – The result code shall be in verbose format. The text portion of information responses is not affected by this setting. If value is omitted, 0 is implied.	
Result codes	A result code shows the effect of the setting of this parameter on the format of information text and result codes as follows:	
	0	If value is 0 (because numeric response text is being used).
	OK	If value is 1.
	4	For unsupported values (if previous value was 0).
	ERROR	For unsupported values (if previous value was 1).
Example	ATV1 **OK** ATV0 **0**	0 will be output on the same line because <LF> is not used for formatting of the result code.
Node types	Coordinator / router / end device.	
Default value	1 – verbose format.	
Persistence	value is stored in the NV Memory.	

5.7.7 “X” - Result Code Selection

Table 5-54. “X” - Result Code Selection

Syntax	Explanation	
X[<value>]	This parameter defines whether the module transmits particular result codes (see Section 3.3 SerialNet Result Codes) to the host or not.	
	Value	**Description**
	0	All result codes are sent to the host
	1	**EVENT** result codes are not sent
	2	**EVENT** and **DATA** result codes are not sent
	If value is omitted, 0 is implied.	
Result codes	**OK** if value is from valid range. Otherwise, **ERROR** is returned.	
Example	ATX2 **OK**	Disable events and data indications
Node types	Coordinator / router / end device.	
Default value	1 – all result codes will be sent, excluding **EVENT**.	
Persistence	value is stored in the NV Memory.	

5.7.8 "+IPR" - Serial Port Communication Rate

Table 5-55. "+IPR" - Serial Port Communication Rate

Syntax	Explanation
`+IPR=<value>`	The command specifies the data rate at which the SerialNet will accept commands and respond. At minimum, 1200b/s and 9600b/s are supported, but specific hardware may support an extended set of rates. Note: The rate specified takes effect following the issuance of any result code associated with the current command line, even if subsequent commands in a command line return **`ERROR`**.
`+IPR?`	The command requests the actual communication rate.
`+IPR=?`	The command requests the list of supported rates. This depends on the hardware capabilities of the particular model.
Result codes	The module returns **`OK`** if the requested rate is present in the supported list; otherwise, it returns **`ERROR`**.

Example	`AT+IPR=38400`
	`OK`
	`AT+IPR?`
	`+IPR:38400`
	`OK`
	`AT+IPR=?`
	`+IPR:(1200,9600,38400)`
	`OK`

Node types	Coordinator / router / end device.
Default value	`38400`
Persistence	`value` is stored in the NV Memory.

5.7.9 "+IFC" - Serial Port Flow Control

Table 5-56. "+IFC" - Serial Port Flow Control

<table>
<tr><th>Syntax</th><th colspan="2">Explanation</th></tr>
<tr><td rowspan="5"><code>+IFC=<rx_flow></code>
<code>,<tx_flow></code></td><td colspan="2">The command is used to specify the methods for local flow control over the UART interface between the host and the module. It accepts two numeric sub-parameters:
• <code>rx_flow</code>, specifies the method for the host to control the flow of data received from the module.
• <code>tx_flow</code>, specifies the method for the module to control the flow of data transmitted from the host.</td></tr>
<tr><td><b><code>rx_flow</code></b></td><td><b>Description</b></td></tr>
<tr><td><code>0</code>
<code>2</code></td><td>None
use RTS (request to send) line</td></tr>
<tr><td><b><code>tx_flow</code></b></td><td><b>Description</b></td></tr>
<tr><td><code>0</code>
<code>2</code></td><td>None
use CTS (clear to send) line</td></tr>
</table>

Syntax	Explanation
	Note: It is strongly recommended to use the CTS method for sleeping end devices because in `sleep` the device is not accepting any data from the UART and without flow control there are no means to ensure that commands are not sent when device is in sleep.
`+IFC?`	The command requests the actual flow control settings.
`+IFC=?`	The command requests a list of the flow control settings supported.
Result codes	**`OK`** is returned if the specified flow control combinations are supported; otherwise, it returns **`ERROR`**.

Example	`AT+IFC=2,2`
	`OK`
	`AT+IFC?`
	`+IFC:2,2`
	`OK`
	`AT+IFC=?`
	`+IFC:(0,2),(0,2)`
	`OK`

Node types	Coordinator / router / end device.
Default value	`0,0`
Persistence	`value` is stored in the NV Memory.

5.7.10 "&D" - DTR Behavior

Table 5-57. "&D" - DTR Behavior

<table>
<tr><th>Syntax</th><th colspan="2">Explanation</th></tr>
<tr><td rowspan="3"><code>&D&lt;value&gt;</code></td><td colspan="2">The command specifies how the module manages the DTR line. It is not supported on ATmega256RFR2 platforms.</td></tr>
<tr><td><b><code>Value</code></b></td><td><b>Description</b></td></tr>
<tr><td><code>0</code>
<code>1</code></td><td>The module ignores the DTR line.
The module wakes up if it is sleeping and with a short delay can start processing data coming from the UART.</td></tr>
<tr><td>S-register</td><td colspan="2"><code>S50</code> (RW)</td></tr>
<tr><td>Result codes</td><td colspan="2"><b><code>OK</code></b> is returned if the requested mode is supported; otherwise, <b><code>ERROR</code></b> is returned.</td></tr>
<tr><td>Example</td><td colspan="2"><code>AT&D1</code>
<b><code>OK</code></b></td></tr>
<tr><td>Node types</td><td colspan="2">Coordinator / router / end device.</td></tr>
<tr><td>Default value</td><td colspan="2">0</td></tr>
<tr><td>Persistence</td><td colspan="2"><code>value</code> is stored in the NV Memory.</td></tr>
</table>

5.7.11 “S0” - Request for the Latest Result Code

Table 5-58. “S0” - Request for the Latest Result Code

Syntax	Explanation	
S0?	Request for the result code from the most recently executed command. If that command was completed with an **ERROR** result code, register S0 will contain a nonzero value. Returned values:	
	0	No error.
	1	Syntax error.
	2	Improper number of parameters.
	3	Parameter value(s) is out of range (example: AT+IFC=12,34).
	4	Unspecified error.
	5	Requested value cannot be read (example: +WLQI command for nonexistent link).
	6	Operation is not permitted in current state (example: setting PAN ID in the connected state or +WSLEEP for router).
	7	Operation cannot be completed due to networking problems (example: connection loss).
	8	Data transmission error.
Result codes	Always **OK**	
Example	AT+ABCD **ERROR** ATS0? **1** **OK** AT+IFC=12,34 **ERROR** ATS0? **3** **OK**	Syntax error, invalid command Parameter is out of range
Node types	Coordinator / router / end device.	

5.8 Hardware Control

AT commands described in this section provide control over hardware functionality, such as GPIO, A/D conversion, and PWM. Most of the HW-specific control is possible via PEEK and POKE commands that provide full access to HW registers.

5.8.1 Reading and Writing HW Registers

Table 5-59. “POKE” - Writing to Hardware Registers

<table>
<tr><th>Syntax</th><th colspan="2">Explanation</th></tr>
<tr><td>POKE &lt;type>,
<addr>,<value&gt;</td><td colspan="2">Command writes value to the register specified by addr and type.
For more information about ranges of addresses and possible values, refer to the hardware datasheets.
type defines the register type as follows:
<table>
<tr><th>&lt;type&gt;</th><th>Description</th></tr>
<tr><td>0
1
3</td><td>Radio transceiver registers MCU internal RAM
MCU internal EEPROM</td></tr>
</table>
Note: By using this command, it is not possible to write to MCU internal flash memory.</td></tr>
<tr><td>Result codes</td><td colspan="2"><b>OK</b> is returned if the address is within the allowed range. <b>ERROR</b> is returned if type or address is out of range.</td></tr>
<tr><td>Example</td><td>ATPOKE 1, 142, 00000006
<b>OK</b></td><td>Write value 0x06 (RX_ON command) to the TRX_STATE register of ATmega256RFR2</td></tr>
<tr><td>Node types</td><td colspan="2">Coordinator / router / end device.</td></tr>
</table>

Table 5-60. “PEEK” - Reading from Hardware Registers

<table>
<tr><th>Syntax</th><th colspan="2">Explanation</th></tr>
<tr><td>PEEK &lt;type>,
&lt;addr&gt;</td><td colspan="2">Command reads the value from register specified by addr and type.
For more information about registers available for a certain platform, and ranges of addresses and values, refer to the hardware datasheets.
type defines the register type as follows:
<table>
<tr><th>&lt;type&gt;</th><th>Description</th></tr>
<tr><td>0</td><td>Radio transceiver registers.</td></tr>
<tr><td>1</td><td>MCU internal RAM.</td></tr>
<tr><td>2</td><td>MCU internal flash.</td></tr>
<tr><td>3</td><td>MCU internal EEPROM.</td></tr>
</table></td></tr>
<tr><td>Result codes</td><td colspan="2"><b>OK</b> is returned if the address is within the allowed range.
<b>ERROR</b> is returned if type or address is out of range of available addresses.</td></tr>
<tr><td>Example</td><td>ATPOKE 1, 145, 00000006
<b>OK</b>
ATPEEK 1, 145
<b>00000006</b>
<b>OK</b></td><td>Write and read PHY_TX_PWR register of ATmega256RFR2</td></tr>
<tr><td>Node types</td><td colspan="2">Coordinator / router / end device.</td></tr>
</table>

5.8.2 GPIO Configuration

Table 5-61. GPIO Configuration

<table>
<tr><th>Syntax</th><th colspan="2">Explanation</th></tr>
<tr><td rowspan="7">S<reg>=<value></td><td colspan="2">Sets the configuration of particular GPIO pin.
reg corresponds to the GPIO pins, GPIO0…GPIO8, on the module, and it is in the range of 120…128.
value is applied as follows:</td></tr>
<tr><td><strong><value></strong></td><td><strong>Description</strong></td></tr>
<tr><td>0</td><td>Input pin, no internal pull-up.</td></tr>
<tr><td>1</td><td>Input pin, internal pull-up is turned on.</td></tr>
<tr><td>2</td><td>Tri-state.</td></tr>
<tr><td>3</td><td>Output.</td></tr>
<tr><td colspan="2">Note: Use of internal pull up improves noise immunity, but also results in increased power consumption.</td></tr>
<tr><td>S<reg>?</td><td colspan="2">The command requests the actual GPIO pin configuration.</td></tr>
<tr><td>Result codes</td><td colspan="2"><strong>OK</strong> is returned if value is in the valid range; otherwise, <strong>ERROR</strong> is returned.</td></tr>
<tr><td>Example</td><td>ATS120=1 S121=3
<strong>OK</strong></td><td>Set GPIO0 as input with internal pull up and GPIO1 as output</td></tr>
<tr><td>Default value</td><td colspan="2">2, tri-state.</td></tr>
<tr><td>Persistence</td><td colspan="2">Values are stored in the NV Memory.</td></tr>
<tr><td>Node types</td><td colspan="2">Coordinator / router / end device.</td></tr>
</table>

5.8.3 GPIO Control

Table 5-62. GPIO Control

<table>
<tr><th>Syntax</th><th colspan="2">Explanation</th></tr>
<tr><td rowspan="4">S<reg>=<value></td><td colspan="2">The command assigns value to a particular GPIO pin.
Each pin, GPIO0…GPIO8, of the module is numbered by reg, which is in the range of 130…138, correspondingly.</td></tr>
<tr><td><strong><value></strong></td><td><strong>Description</strong></td></tr>
<tr><td>0
1</td><td>Logical 0
Logical 1</td></tr>
<tr><td colspan="2">Note: This command does not affect any pin configured as input or tri-state.</td></tr>
<tr><td>S<reg>?</td><td colspan="2">The command reads a particular GPIO pin, numbered and coded as above, and returns 0 or 1. If the pin is configured for output or as tri-state, the returned value is not defined.</td></tr>
<tr><td>Result codes</td><td colspan="2"><strong>OK</strong> is returned if value is 0 or 1; otherwise, <strong>ERROR</strong> is returned.</td></tr>
</table>

Syntax	Explanation	

Example	`ATS120=1 S121=3`
	`ATS130?`
	`1`
	`OK`
	`ATS131=0`
	`OK`
		Set GPIO0 as input and GPIO1 as output, both with internal pull-up
		Read GPIO0, it is 1
		Clear GPIO1

Default value	`0`	
Persistence	Values are NOT stored in the NV Memory because GPIO pins are configured as tri-state at startup.	
Node types	Coordinator / router / end device.	

5.8.4 A/D Configuration

Table 5-63. A/D Configuration

Syntax	Explanation	
`S100=<value>`	The command selects the configuration of particular A/D pins. `value` is a hexadecimal number containing a bit field. The four least-significant bits (`b0...b3`) can be used to enable or disable each of four A/D channels. Bits `b4...b7` are ignored in the `value` field. If a bit is cleared, then A/D conversion of a corresponding channel is disabled and the A/D pin goes to the high-impedance state without internal pull-up. Notes: 1. Enabling A/D conversion increases power consumption. 2. Conversion is executed in single conversion mode thus enabling the maximum conversion rate of approximately 5kb/s. 3. Proper conversion results are achieved for if the external reference signal of 1.25V is applied to the `A_VREF` pin. If conversion is disabled on all A/D pins, the `A_VREF` pin is moved to tri-state. 4. Pins AD4…AD7 can be also used as a JTAG port, and then A/D conversion functionality for these inputs is disabled.	
`S100?`	The command requests the actual A/D configuration.	
Result codes	**`OK`** is always returned.	

Example	`ATS100=08`
	`OK`
		Enable conversion on pin AD3

Default `value`	`00` – disable A/D conversion for all four A/D pins.	
Persistence	`value` is stored in the NV Memory.	
Node types	Coordinator / router / end device.	

5.8.5 A/D Conversion

Table 5-64. A/D Conversion

Syntax	Explanation
`S<reg>?`	The command reads a particular A/D pin and returns its value in decimal format. `reg` corresponds to pins AD0…AD3 on the module, and it is in the range of 101…104. If A/D conversion for a particular channel is disabled by the `S100` register, no value is returned.
Result codes	**`OK`** is always returned.

Syntax	Explanation	
Example	ATS100=08 **OK** ATS104? **125** **OK**	Enable conversion on pin AD3 Read AD3 pin
Node types	Coordinator / router / end device.	

5.8.6 PWM Configuration

Table 5-65. PWM Configuration

Syntax	Explanation		
S<reg>=<value>	The command configures a particular PWM channel:		
	PWM channel	**Output pin on ATmega256RFR2**	**S-register**
	0 1 2	PB5 PB6 PB7	140 141 142
	<value>	**Description**	
	0, 2 1 3	Disable PWM channel Enable channel, setting non-inverted output polarity (output is low when duty cycle = 0% and high when duty cycle = 100%) Enable channel, setting inverted output polarity (output is high when duty cycle = 0% and low when duty cycle = 100%)	
	Notes: 1. When a PWM channel is enabled, the corresponding output pin is configured as output to be controlled by that PWM channel. Duty cycle for the channel is set to 0. PWM channel frequency is set to the default value (5kHz) if no channel has yet been opened. Otherwise, the frequency last set for any other channel is used. 2. When a PWM channel is disabled by setting reg to 0 or 2, the corresponding output pin is configured as tri-state and is fully controlled as GPIO.		
Result codes	**OK** is returned if the value is in the valid range; otherwise, **ERROR** is returned.		
S<reg>?	The command requests the current PWM configuration.		
Result codes	**OK** is always returned.		

Example	ATS140=1
	S142=3
	OK
		Enable PWM channel 0, setting noninverted polarity output, and enable PWM channel 2, setting inverted polarity output	

Default value	0, disabled.		
Persistence	value is NOT stored in the NV Memory.		
Node types	Coordinator / router / end device.		

5.8.7 PWM Frequency Control

Table 5-66. PWM Frequency Control

<table>
<tr><th>Syntax</th><th colspan="3">Explanation</th></tr>
<tr><td rowspan="6"><code>S<reg>=<value></code></td><td colspan="3">The command selects the PWM operating frequency for a particular PWM channel.</td></tr>
<tr><td><b>PWM channel</b></td><td><b>Output pin on ATmega256RFR2</b></td><td><b>S-Register</b></td></tr>
<tr><td>0
1
2</td><td>PB5
PB6
PB7</td><td>143
144
145</td></tr>
<tr><td><b><code><value></code></b></td><td colspan="2"><b>PWM frequency</b></td></tr>
<tr><td>0
1
2
3
4</td><td colspan="2">5kHz
10kHz
20kHz
50kHz
100kHz</td></tr>
<tr><td colspan="3">PWM frequency selection for any channel affects all channels (frequency is common for all channels). Changing frequency for any PWM channel results in the reset of the duty cycle to 0 for all channels.</td></tr>
<tr><td>Result codes</td><td colspan="3"><b><code>OK</code></b> is returned if <code>value</code> is in the valid range; otherwise, <b><code>ERROR</code></b> is returned.</td></tr>
<tr><td><code>S<reg>?</code></td><td colspan="3">The command reads the PWM operating frequency for a particular PWM channel, coded as above, and returns 0 to 4.</td></tr>
<tr><td>Result codes</td><td colspan="3"><b><code>OK</code></b> is always returned.</td></tr>
<tr><td>Example</td><td><code>ATS143=2</code>
<b><code>OK</code></b>
<code>ATS144=4</code>
<b><code>OK</code></b>
<code>ATS143?</code>
<b><code>4</code></b>
<b><code>OK</code></b></td><td colspan="2">Set the PWM frequency to 20kHz for PWM channel 0.
Set the PWM frequency to 100kHz for PWM channel 1.
Request the PWM frequency on channel 0. The most recent frequency set is returned.</td></tr>
<tr><td>Default value</td><td colspan="3"><code>0</code> - 5kHz.</td></tr>
<tr><td>Persistence</td><td colspan="3"><code>value</code> is NOT stored in the NV Memory.</td></tr>
<tr><td>Node types</td><td colspan="3">Coordinator / router / end device.</td></tr>
</table>

5.8.8 PWM Duty Cycle Control

Table 5-67. PWM Duty Cycle Control

Syntax	Explanation		
S<reg>=<value>	The command selects the duty cycle `value` for a particular PWM channel.		
	PWM channel	**Output pin on ATmega256RFR2**	**S-Register**
	0	PB5	146
	1	PB6	147
	2	PB7	148
	`value` is an integer number in the range of 0 to 100 representing the PWM duty cycle, in percent. Notes: 1. The duty cycle currently set on the output pin will be changed as soon as the current period of PWM frequency is ended. 2. Resolution of the duty cycle setting depends on the PWM frequency, as below:		
	PWM frequency	**Duty cycle resolution**	
	5kHz	1%	
	10kHz	1%	
	20kHz	1%	
	50kHz	2.5%	
	100kHz	5%	
Result codes	**OK** is returned if `value` is in the valid range; otherwise, **ERROR** is returned.		
S<reg>?	The command reads the duty cycle for a particular PWM channel, in percent.		
Result codes	**OK** is always returned.		
Example	ATS146=45 **OK**	Set duty cycle to 45% for PWM channel 0	
Default value	`0` (%).		
Persistence	`value` is NOT stored in the NV Memory.		
Node types	Coordinator / router / end device.		

5.9 Remote Management

Remote management functions include the password protected AT commands that come from the originating node to a target node. The received AT command sequences are executed on the destination node as if they came from a serial port. Information response and result codes of the command execution are sent back to the originating node in the same form as if they were returned over a serial interface.

Remote execution service can be protected by a 32-bit password that can be set during node installation or manufacturing.

Remote management function is an important tool that allows the organization of commissioning procedures on a PC using commercial, off-the-shelf terminal software.

5.9.1 “+WPASSWORD” - Set a Password

Table 5-68. “+WPASSWORD” - Set a Password for Remote Access

Syntax	Explanation
`+WPASSWORD <psw>`	The command sets a password for the remote execution of AT commands (see Section 5.9.2 “R” - Remote Execution of AT Command) `psw` specifies the new password in form of a 32-bit hexadecimal number. Note: This command is not to be confused with the parameter set commands. Unlike those, it does not include the “`=`” symbol.
Result codes	**`OK`** is always returned.

Example	`AT+WPASSWORD 65432178`
	`OK`

Default value	0
Persistence	The `psw` is stored in the NV Memory. Note: The password cannot be reset to a default value through the *`&F`* command (see Section 5.6.8 “&F” – Set to Factory Default Configuration), but it can be rewritten over the air using the remote execution AT command (see Section 5.9.2 “R” - Remote Execution of AT Command).
Node types	Coordinator / router / end device.

5.9.2 “R” - Remote Execution of AT Command

Table 5-69. “R” - Remote Execution of AT Command

Syntax	Explanation
`R<addr>,<psw>,<cmd>`	The command enables the execution of AT commands on a remote node, with output redirected. `addr` specifies the short (network) address of the destination node. `psw` is a 32-bit hexadecimal number, which is used as password for remote management on this specific node. Synchronization of this password between the command originator and target are upto the host application. `cmd` is a sequence of AT commands without the `AT` prefix. Note: It is strongly recommended not to use the `&H` and `%H` commands for `cmd`, as they produce extremely lengthy output.
Result codes	All the responses and result codes are received from the remote node in text form that can be processed normally. If a connection loss is detected, the **`ERROR`** result code will be returned after a timeout from when the last response packet was received (approx. three seconds). In particular, remote execution of the `+WLEAVE` command will result in an **`ERROR`** code, despite being executed successfully. If the remote execution command is sent to an end device with a sleeping period longer than the timeout, **`ERROR`** will be returned. If the controlled node is not in the PAN, **`ERROR`** will be returned. Remote execution is not allowed for commands that cause the receiving node to send data over the network: `D`, `DU`, `DS`, `+WPING`, `R`. Attempting this will result in an **`ERROR`** code, with the command processing aborted.

Syntax	Explanation	

Example	`ATR1,65432178,+GMM?+WRSSI 2`
	`+GMM:MEGARF-ZIGBIT`
	`+WRSSI:-80`
	`OK`
	`ATR1,65432178,+WLEAVE`
	`ERROR`
		Get model number and RSSI.
		Remove node from network – **`ERROR`** will be returned, but delayed as target node leaves the network.

Node types	Coordinator / router / end device.	

6 Reference

[1] ZigBit® 2.4GHz wireless modules. ATZB-S1-256-3-0-C datasheet.

[2] ZigBit 2.4GHz wireless modules. ATZB-S1-256-3-0-U datasheet.

[3] ZigBit Extension User Guide.

[4] Atmel ATMEGA256RFR2 XPRO User Guide.

[5] ZigBee PRO specification (053474r20).

[6] Serial asynchronous automatic dialing and control. ITU-T Recommendation V.250, 05/99

[7] International Reference Alphabet (IRA) (Formerly International Alphabet No. 5 or IA5). Information Technology – 7-Bit Coded Character Set for Information Interchange, CCIT Recommendation T.50, 09/92.

[8] General Structure of Signals of International Alphabet No. 5. Code for Character Oriented Data Transmission over Public Telephone Networks. ITU-T Recommendation V.4.

[9] IEEE® Std. 802.15.4-2006 IEEE Standard for Information technology – Part 15.4 Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs).

[10] BitCloud Software Development Kit.

[11] ATmega256RFR2 device.

[12] ZigBit Extension programming via Bootloader PC Tool.

[13] Atmel AVR2054: Serial Bootloader User Guide.

7 Revision History

Doc Rev.	Date	Comments
8389C	05/2015	Updated for SerialNet v3.0.0. Description of ATmega256RFR2-specifics is added. Minor improvements through the document.
8389B	06/2012	Updated for SerialNet v2.7.0 in BitCloud SDK v1.14.0.
8389A	02/2011	Initial document release.

Atmel | Enabling Unlimited Possibilities®

Atmel Corporation 1600 Technology Drive, San Jose, CA 95110 USA **T:** (+1)(408) 441.0311 **F:** (+1)(408) 436.4200 | **www.atmel.com**

DISCLAIMER: The information in this document is provided in connection with Atmel products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Atmel products. EXCEPT AS SET FORTH IN THE ATMEL TERMS AND CONDITIONS OF SALES LOCATED ON THE ATMEL WEBSITE, ATMEL ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS AND PROFITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF ATMEL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Atmel makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and products descriptions at any time without notice. Atmel does not make any commitment to update the information contained herein. Unless specifically provided otherwise, Atmel products are not suitable for, and shall not be used in, automotive applications. Atmel products are not intended, authorized, or warranted for use as components in applications intended to support or sustain life.

SAFETY-CRITICAL, MILITARY, AND AUTOMOTIVE APPLICATIONS DISCLAIMER: Atmel products are not designed for and will not be used in connection with any applications where the failure of such products would reasonably be expected to result in significant personal injury or death (“Safety-Critical Applications”) without an Atmel officer's specific written consent. Safety-Critical Applications include, without limitation, life support devices and systems, equipment or systems for the operation of nuclear facilities and weapons systems. Atmel products are not designed nor intended for use in military or aerospace applications or environments unless specifically designated by Atmel as military-grade. Atmel products are not designed nor intended for use in automotive applications unless specifically designated by Atmel as automotive-grade.