EPC C1G2 BATTERY ASSISTED AMBIENT TEMPERATURE LOGGER SENSOR

Check for samples: EVAL01-FENIX-RML

FEATURES

- **EPC C1G2 compliant**
- **ISO 18000-6 Type C compliant**
- **160-bit EPC Bank: Up to 128-bit EPC**
- **96-bit TID Bank: Up to 48-bit Serial Number**
- **Available User Memory: Up to 1008-bit Non Volatile User Data**
- **Extended range in battery assisted passive mode: 20m**
- **Ambient temperature sensor with data logger**
  - **Operation range: -40°C to 85°C**
  - **Accuracy: 0.5°C**
  - **Resolution: 0.0625°C**
- **Data logger functions**
  - **Memory capacity: up to 44k samples**
  - **Programmable sample intervals**
  - **Programmable threshold alerts**
  - **Battery monitor**
  - **Fast data download time**
  - **Battery free data download**
- **Pick to light indicator for visual identification**

DESCRIPTION

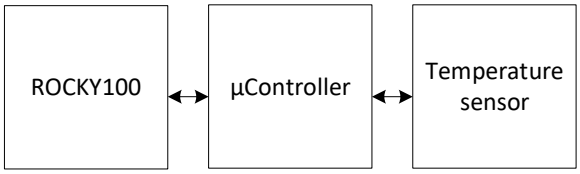
FENIX-RML is an EPC Class-1 Generation-2 (C1G2) RFID tag based on Farsens' sensor technology. Built in a compact PCB format, the tag includes an ambient temperature sensor with logging capabilities.

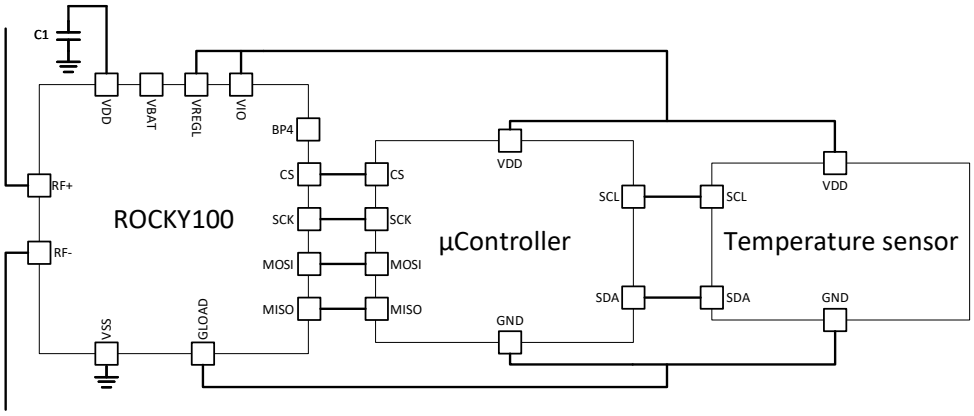
These RFID sensor tags are compatible with commercial UHF RFID readers (EPC C1G2). The device can work with or without batteries for device configuration and data download. The battery is required for data logging during normal operation.

The FENIX-RML can be customized with different antenna design and sizes, depending on the specific application. It can be encapsulated in an IP67 or IP68 casing for usage in harsh environments. It may also be possible to customize the specifications of the sensor upon request.

BLOCK DIAGRAM

The FENIX-RML tag consists of a ROCKY100 IC, a microcontroller and a digital temperature sensor.





The ROCKY100 IC includes the RF front end for UHF RFID power harvesting and communication, a power supply module to generate the required voltage levels, and an EPC C1G2/ISO18000-6C digital processor including a SPI bridge. The SPI bridge can be controlled via EPC C1G2 standard memory access commands.

The operation of measuring and logging ambient temperature is controlled with a microcontroller. Besides the CPU and the memory, the microcontroller includes a SPI module and a I2C interface. Finally, a digital temperature sensor provides the actual temperature measurement.

The configuration and data download of the device can work completely battery free. However, note that the data acquisition and logging requires the assistance of a battery.

BASIC USER INTERFACE

FENIX-RML has two main interfaces: UHF RFID C1G2 and manual. The RFID interface allows detailed configuration and data extraction from the device. However, this can only be done with the help of an RFID reader.

Once the device has been correctly configured, the manual interface can be used for visual status indication and start/stop operations. A RGB LED is included for status indication, and a push button for interacting with the device in the field without the need of further equipment.

CHECK STATUS

A short button push (less than 3 seconds) will trigger a status notification through the RGB LED. The LED will turn on momentarily with the following color codification:

COLOR	DESCRIPTION
Blue	System status OFF
Green	System status ON
Red	Alert indicator

The alert indicator led will turn on continuously along with the status led for threshold alerts. After a small delay, the red led will start to blink if the sensor is running low on battery, independently of the threshold alerts.

CHANGE STATE

In order to change the current system status manually, a two step process is used. First, a long button push (over 3 seconds) is required to trigger the operation. At this point the RGB LED will flash with the color of the new configuration:

COLOR	DESCRIPTION
Blue	System status OFF
Green	System status ON

The user has up to 5 seconds to validate the new status with a new button push. If no validation is performed, the status change will be discarded.

DATA ACQUISITION AND ALERTS

When setting the device to ON state, FENIX-RML will save the sample interval, the timestamp defined by the Real Time Clock (RTC) and the current temperature. Subsequent measurements at the specified time interval will then be logged. During each measurement, the current temperature is compared with the alert thresholds, if they pass those thresholds, a flag for that alert will be set.

LOG MODE

To optimize the logging, FENIX-RML logs temperature measurements in differential mode. This mode saves the initial timestamp, the sample time and the reference temperature. Subsequent reads are considered to be taken after the specified sample interval from the last measurement. The logger is only capable of saving one continuous log. Each time the logger is started, previous data (if any) is discarded.

All following information is stored as the difference between the new measurement and the last one. In order to do so, the data is stored in variable length words following the Extensible Bit Vector (EBV) format. The first bit of each byte of information indicates if the data is completed in this byte ('0'), or if more bytes have to be appended ('1').

CHARACTERISTICS

SYMBOL	PARAMETER	MIN	TYP	MAX	UNIT
RFID					
$r_{operation}$	Operation range full passive		5		m
	Operation range BAP		15		m
	Operation range EBAP		20		m
OPERATING CONDITIONS					
T_{OP_TOP}	Operating temperature range	-40		85	°C
TEMPERATURE SENSOR					
T_{range}	Temperature range	-40		85	°C
T_{acc}	Temperature accuracy				°C
	0 °C to 65 °C		±0.5		°C
	-40 °C to 85 °C		±1		°C
T_{res}	Temperature resolution		0.0625		°C

RF INTERFACE

FENIX-RML uses the integrated UHF RFID interface for wireless communication. This interface complies with the EPC C1G2 standard, so that all the basic item identification functionality is included. The device allows unique item identification with the upto 128 bit length Electronic Product Code (EPC).

This RF interface is used to configure the device, retrieve the logged data and reset the device for a new use. All of this can be performed using standard EPC C1G2 read and write commands, so that any standard compliant reader can be used.

The following image shows the distribution of the memory in FENIX-RML.

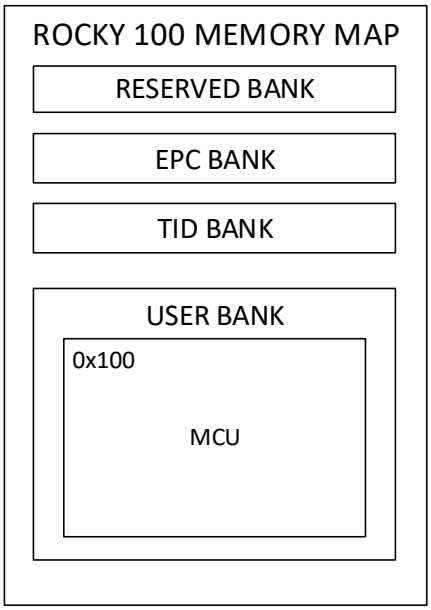


Figure 1: RFID memory distribution of FENIX-RML

When reading in the user bank, all addresses after 0x100 are sent to the MCU. Read commands to specific addresses are then used to transfer data blocks from the reader to the MCU and from the MCU back to the reader. To guarantee compatibility with common readers, it was decided to only use 16-bit addresses. Because of this decision, it is only possible to send to the MCU one byte of data per read, as the other byte indicates the command. All information is to be stored in **little endian** format.

MCU MEMORY

Besides the RFID IC memory, FENIX-RML integrates an additional volatile and non-volatile memory inside the MCU. This memory is partitioned as follows in order to store all the required information.

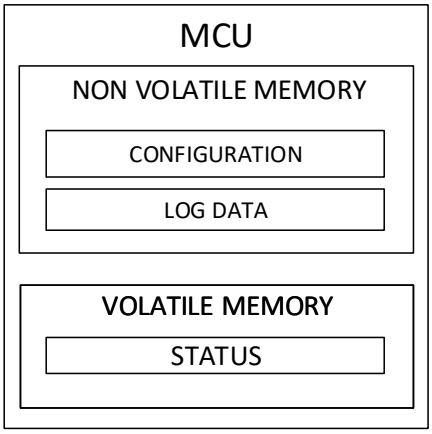


Figure 2: RFID memory distribution of FENIX-RML

CONFIGURATION

The configuration partition contains all the operation parameters required by the device.

LOG DATA

The log data partition contains all temperature log information.

STATUS

The status contains the current ON/OFF status of the logger

CONFIGURATION VARIABLES

All MCU configuration variables are non volatile except for the status. The variables that configure the MCU are the following:

Table 1: Configuration Variables.

TYPE	VARIABLE	ACCESS	DESCRIPTION
[UINT8]	time	get	6 Byte array containing current date components
BOOL	status	get/set	Current FENIX-RML ON/OFF status.
UINT16	rate	get/set	Current sample time.
BOOL	BAP	get/set	Battery Assisted Passive mode Enable/Disable.
INT16	upperTempTh	get/set	Coded upper alert temperature threshold
INT16	lowerTempTh	get/set	Coded lower alert temperature threshold
UINT32	logLines	get	Current logged lines
UINT32	writtenBytes	get	Current written bytes
UINT8	alerts	get	Current triggered alerts
UINT16	column	set	Column for download
UINT8	year	set	RTC years after 2000
UINT8	month	set	RTC month
UINT8	day	set	RTC day
UINT8	hour	set	RTC hour
UINT8	minute	set	RTC minute
UINT8	second	set	RTC second

Note: Coded temperature threshold variables hold the temperature thresholds divided by 0.0625, to decode the temperature threshold, it is then needed to multiply the encoded value by 0.0625.

OPERATION

EPC READING

In order to read the EPC of the tag, commercial EPC C1G2 readers can be used. However, some considerations have to be taken into account.

As the tag has a significant supply capacitor connected to VDD, the power-up of the system will be slow. It can last several seconds. In order to speed up the charge process, the reader shall be configured to send power as continuously as possible.

Once the supply capacitor is charged, the tag will respond with its EPC. From this point on, memory access commands can be used to control additional functionalities via the SPI bridge.

COMMAND SET

In order to command actions from the RFID reader to the FENIX-RML device, the following command structure and command set are defined. The command itself has to be written in the MSB of the 16-bit read address, while the argument is included in the LSB of the address.

Get commands and the erase command do not require an argument, while all set commands should always include a valid argument. When setting two byte variables it is important to first send the LSB, as when setting the MSB it concatenates the received byte with the previously set LSB. For this reason, setting the MSB can be omitted if the byte is empty.

Table 2: Command structure.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CODE								**ARGUMENTS**							

[15:8] **CODE:** Command code.

[7:0] **Arguments:** command dependant arguments.

Table 3: Command set.

CODE	COMMAND	DESCRIPTION
0x01	GET_SENSOR	Get current sample while logging.
0x02	GET_TIME	Get current RTC time.
0x03	GET_STATUS	Get current FENIX-RML status.
0x04	SET_STATUS	Set new FENIX-RML status.
0x05	GET_RATE	Get current sample time.
0x06	SET_RATE	Set new sample time.
0x07	SET_RATE_MSB	Set MSB of new sample time.
0x08	GET_BAP	GET current Battery Assisted Passive mode.
0x09	SET_BAP	SET current Battery Assisted Passive mode.
0x0A	GET_LOG_SIZE	Get current logged lines.
0x0B	GET_WRITTEN_BYTES	Get current written log bytes.
0x0C	GET_ALERTS	Get current flag alerts.
0x0D	GET_COLUMN	Get current column.
0x0E	GET_COLUMN_INCREMENT	Get current column and go to next column.
0x0F	SET_COLUMN	Sets new column for download.
0x10	SET_COLUMN_MSB	Sets new column MSB for download.
0x11	ERASE	Erases log and clears alerts.
0x12	SET_YEAR	Set year of RTC.
0x13	SET_MONTH	Set month of RTC.
0x14	SET_DAY	Set day of RTC.
0x15	SET_HOUR	Set hour of RTC.
0x16	SET_MINUTE	Set minute of RTC.
0x17	SET_SECOND	Set second of RTC.
0x18	GET_UPPERALERT_TH	Get current upper alert threshold.
0x19	SET_UPPERALERT_TH	Set new upper alert threshold.
0x1A	SET_UPPERALERT_TH_MSB	Set new upper alert MSB threshold.
0x1B	GET_LOWERALERT_TH	Get current lower alert threshold.
0x1C	SET_LOWERALERT_TH	Set new lower alert threshold.
0x1D	SET_LOWERALERT_TH_MSB	Set new lower alert MSB threshold.

GET COMMANDS

When performing get commands, the word count of the EPC read should be set accordingly to the size of the variable to be read. Each packet received contains a base header of 4 bytes (2 words). The final word count should then be 2 plus the words needed to contain the variable.

EPC Read

Operation: Read
Memory bank: User Memory
Word Pointer: **GET_COMMAND**
Word Count: 2 + sizeof(type)/2

The answer from the tag to such a request will contain N+1 bytes of data. Assuming that the reader returns the received data in the buffer of bytes *rawdata*, the content of the answer is defined as follows:

rawdata	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	...	...	Byte N
content	HEADER	FW_VER	COLUMN	DATA 1	DATA 2	DATA 3	DATA X	QOS

- HEADER (uint8): datagram header '0xAA'. The header will be set for each packet sent. If header is not set, the following fields have to be discarded.
- FW_VER (uint8): firmware version included in the micro-controller.
- COLUMN (uint8): LSB of the column ready to be downloaded.
- DATA (variable): Requested data (Little Endian).
- QOS (uint8): Quality Of Service provided by ROCKY100. Refer DS-ROCKY100 for further details on this parameter.

QOS	Meaning
0xFF	Sensor working in best conditions
0xEE	Sensor working in good conditions
0xCC	Sensor switched off
0x88	Sensor switched off

GET_ALERTS

The command GET_ALERTS, will return a byte containing the flags for the temperature thresholds and a flag for low battery. A read command to get the alerts is then as follows:

EPC Read

Operation: Read
Memory bank: User Memory
Word Pointer: 3072
Word Count: 3

The answer from the tag to such a request will contain 6 bytes of data. Where as expected, byte 3 will contain the alerts byte. The flag bits are defined as follows for each alert.

rawdata	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
content	Reserved					Lower Alert	Upper Alert	Low Bat

- Low Bat: FENIX-RML is running low on battery. Battery level below 2.55 V.
- Upper Alert: During log, the temperature reached the upper threshold.
- Lower Alert: During log, the temperature reached the lower threshold.

GET_TIME

The command GET_TIME, different from other commands, will return a 6-Byte UINT8 array instead of a single variable. A read command to get the time is then as follows:

EPC Read
Operation: Read
Memory bank: User Memory
Word Pointer: 512
Word Count: 5

The answer from the tag to such a request will contain 10 bytes of data. Assuming that the reader returns the received data in the buffer of bytes *rawdata*, the content of the answer is defined as follows:

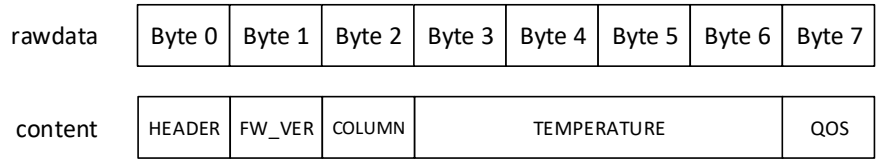
rawdata	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8	Byte 9
content	HEADER	FW_VER	COLUMN	YEAR	MONTH	DAY	HOUR	MIN	SEC	QOS

LIVE TEMPERATURE

While the logger is in operation, it is possible to read the current ambient temperature using standard EPC read commands. The command will request the logger to load in the Rocky buffer the current measurement of the device, which will be ready for the next EPC read with the same command.

EPC Read
Operation: Read
Memory bank: User Memory
Word Pointer: 100
Word Count: 4

The answer from the tag to such a request will contain 8 bytes of data. And the content to the answer is defined as follows.



Where TEMPERATURE is a binary 32, single precision floating point (Little Endian). Value is given in °C.

DOWNLOADING LOG

Downloading the logged temperature data can be accomplished by performing multiple GET_COLUMN_INCREMENT commands. Some readers allow reads of maximum 16 words, taking away the header, each read command can retrieve 28 bytes. Each column will thus contain 28 bytes of data. To determine the number of reads that need to be performed to get all the data, it is necessary to get the current written bytes and divide it by 28 (rounding up the result).
To start a download it is important to first set the initial column to be read (starting at 0). After setting the column, successive GET_COLUMN_INCREMENT commands can be performed for the number of columns previously calculated. If an error occurs during a GET_COLUMN_INCREMENT command (i.e. first byte is not 0xAA). It is recommended to reset the column to be read and retry the GET_COLUMN_INCREMENT command. For convenience, each frame will include in the header the LSB of column that was downloaded. Optional checks can thus be performed to assure the continuity of the download, assuring the continuous increasing of the column byte can avoid an alteration in all future samples.

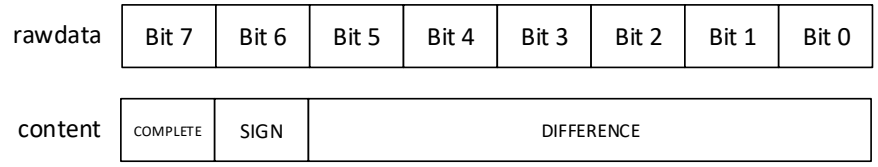
As the logged data is stored in EBV format, it is recommended to first get all the data before processing it. The resulting downloaded data buffer will have the following format:

- TIMESTAMP (uint32): UNIX timestamp containing the time the logger started.
- RATE (uint16): Sample time at which the logger worked.
- INITIAL TEMPERATURE (int16): Temperature at which the logger started (encoded).
- DIFFERENTIAL DATA (EBV): Differential logged temperature data (encoded).

To decode all temperature values, it is simply needed to multiply the encoded value by 0.0625.

GETTING THE DIFFERENTIAL DATA

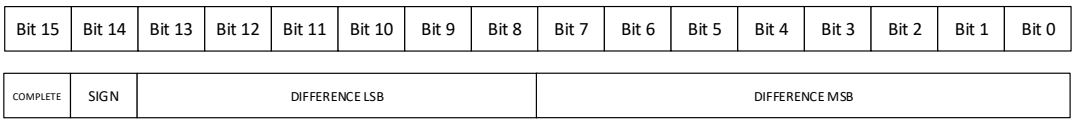
Once the encoded initial temperature has been obtained, it is then possible to obtain the remaining samples from the differential data. The first EBV byte after the initial temperature will have the following format:



- COMPLETE (uint): clear: sample contained in 1 byte, set: sample in 2 bytes

- SIGN (uint): clear: difference is negative from previous result, set: difference is positive
- DIFFERENCE (uint6): encoded differential value from previous sample.

It is important to remark that the variation needed to require 3 bytes for a sample is over the range of the temperature sensor. A sample will always be contained in no more than 2 bytes, this fact can reduce the decoding algorithm complexity. Therefore a byte that has the complete bit marked as set will contain its differential sample as follows:



Where the difference is an uint14 little-endian. After determining the bytes needed for the sample, it is then simply needed to jump to the next sample and repeat the process until all samples are decoded.

PICK TO LIGHT INDICATOR

The pick to light indicator of FENIX-RML can be triggered using standard EPC read commands. The PWM module of the ROCKY100 is used to control an LED. Upon receiving a not-null value write command directed to the PWM trigger register, FENIX-RML will generate the PWM signalling which will make the LED of the device blink according to the active configuration. By default, the device will generate 3 short blinks.

Trigger blink

Operation: Write
Memory bank: User Memory
Word Pointer: 0x91
Data: 0x01

DEMO SOFTWARE

Demonstration software to read and control the FENIX-RML is available in the web. Download the latest software and user guide at: http://www.farsens.com/software.php. Check the website for updated reader compatibility list. Up to the date of writing this document, this is the status of the compatibility list:

Fixed readers			
Manufacturer	**Model**	**Tested HW rev.**	**Tested FW rev.**
Impinj	R420	HLA: 1.00 | PCBA: 4.00	5.12.1
Impinj	R220	-	-
Impinj	R120	-	-
Nordic ID	Sampo	PWM00282	5.4 A
Nordic ID	Stix	PWM00226	5.10 A
Alien	ALR9900	-	14.07.01.00
ThingMagic	Mercury6	1.0	4.19.3.2
Zebra	FX9500	-	1.5.4.348

Handheld readers				
Manufacturer	**Model**	**OS**	**Tested HW rev.**	**Tested FW rev.**
Nordic ID	Merlin	Windows CE 6.0	PWM00193	3.7.0
Zebra	MC9090G	Windows CE	x.xx	x.xx

REFERENCES

The next table shows the available references of the FENIX-RML.

Ref.	Name	Description
42102	EVAL01-FENIX-RML	FENIX-RML, dipole wideband antenna, PCB format

For custom references with other antennas and housings, please contact us at sales@farsens.com.

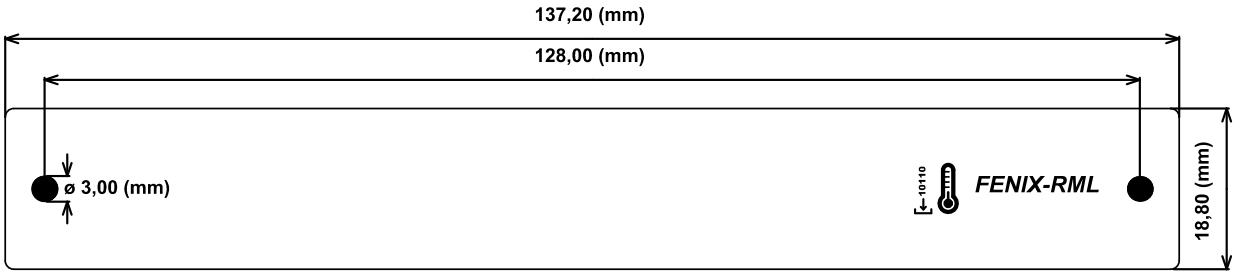
MECHANICAL DIMENSIONS

All dimensions are in millimeters.

DKWB

Valid for reference(s): 42102

2D VIEW



Maximum height: 10mm

3D VIEW