SERMA
PRODUCTIVITY ENGINEERING

Documentation

EVA3011 User Guide

Table of Content

1 Revision History

Version	Date	Changes	Page
Initial Version 1.0	10/2010		

2 Overview

The EVA3011 is especially designed to evaluate the PE3011 HF RFID Integrated Circuit of Productivity Engineering GmbH featuring an integrated temperature sensor, external capacitive sensor interface and real time clock as well as an 8kBit EEPROM for data monitoring. Digital external sensors can also be incorporated with the help of a microcontroller through the SPI interface of the chip.
The EVA3011 serves as a demonstrator and evaluation kit with these features:

- PCB with IC, LED, resistors, sensors and battery
- Two touch sensors to be used as capacitive on/off switches (e.g. box open/closed)
- passive RFID HF transponder with ISO 15693 protocol interface
- recording of time, sensor and temperature events in a defined time interval, either in normal mode or in outband mode
- recording of time and sensor events as interrupt function
- readout of all stored data via HF-RFID interface (ISO15693)
- detailed representation of data in MS Excel format for a graphical view in Windows OS
- default configuration of Evaluation Kit (BAT; L1=R1=R2=0Ohm; PE3011; JP1)
- R3 and LED1 (indicator alarm port - optional)
- JP2 (boost range) not mounted

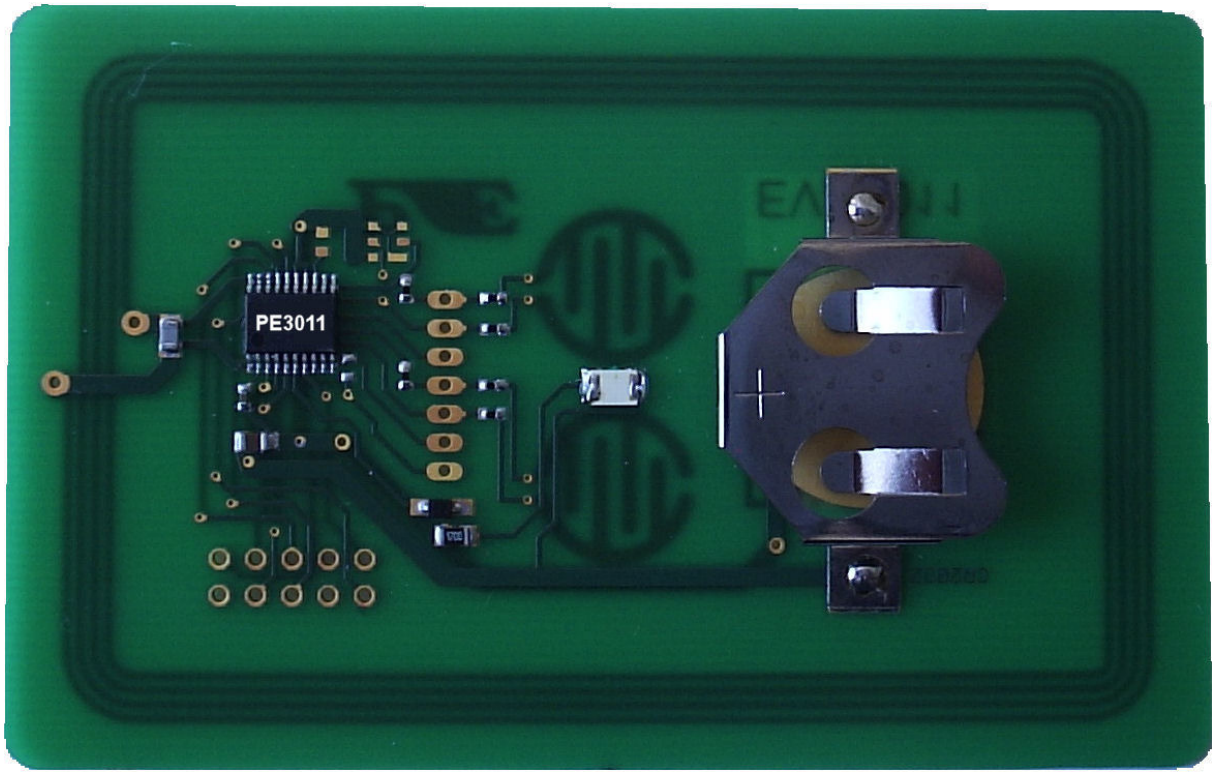


Figure 1 - Evaluation Kit PE3011 v1.0

3 Controlling software „EVA3011“

3.1 Installation

After downloading the software to manipulate the Data Monitor with a SCEMTEC Reader it needs to be installed by executing the file **“Install_EVA3011_1.0.exe”**.
The EVA3011 comes with a graphical user interface for WindowsXP ™ platforms. It is recommended to start with this simple software interface to learn about the basic functionality of the chip on the board.

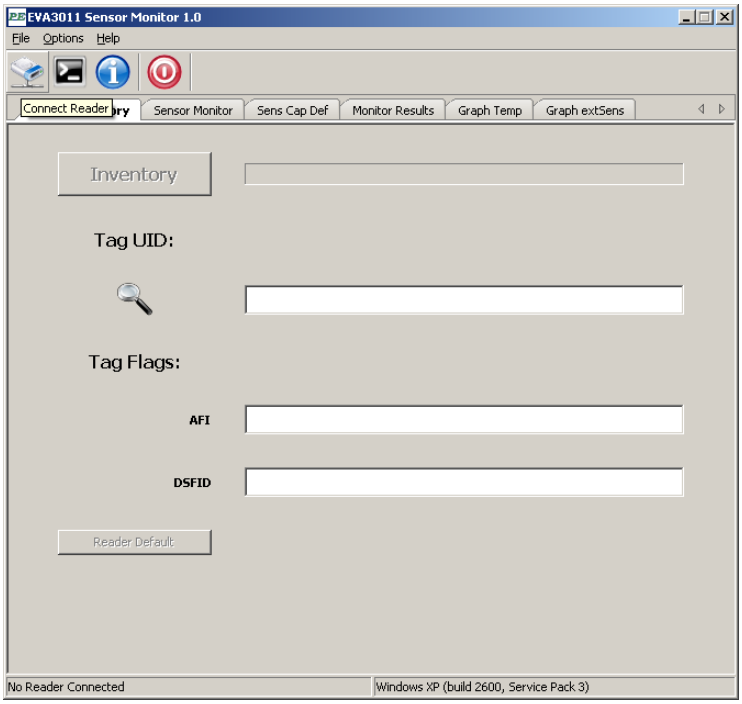
All necessary files will be copied into the target directory and a group will be created in the “Start” Menu. To communicate with SCEMTEC Reader the internal USB to RS232 driver with DLL will be used. Readers from other vendors may not work properly. Typically each reader requires its own USB driver interface. Driver installation can be done through standard software installation or through the “Start” Menu.

SCEMTEC
The driver interface for USB must be installed with the reader software from the reader vendor. For the operation with SCEMTEC a DLL and special internal C++ functions are available. A SCEMTEC reader SIR 2710 can communicate with these functions through the USB to RS232 conversion. The software automatically checks the devices connected to the ports and connects with the reader.

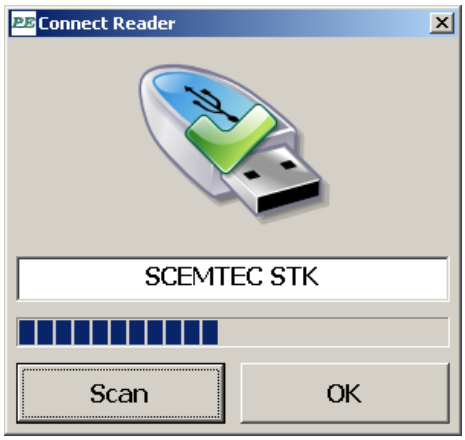
Note: A software driver might or might not be required. More readers can be incorporated by PE on request if a driver is not available from the reader manufacturer. Available source codes are free to be used.

3.2 Software description – Connect Reader

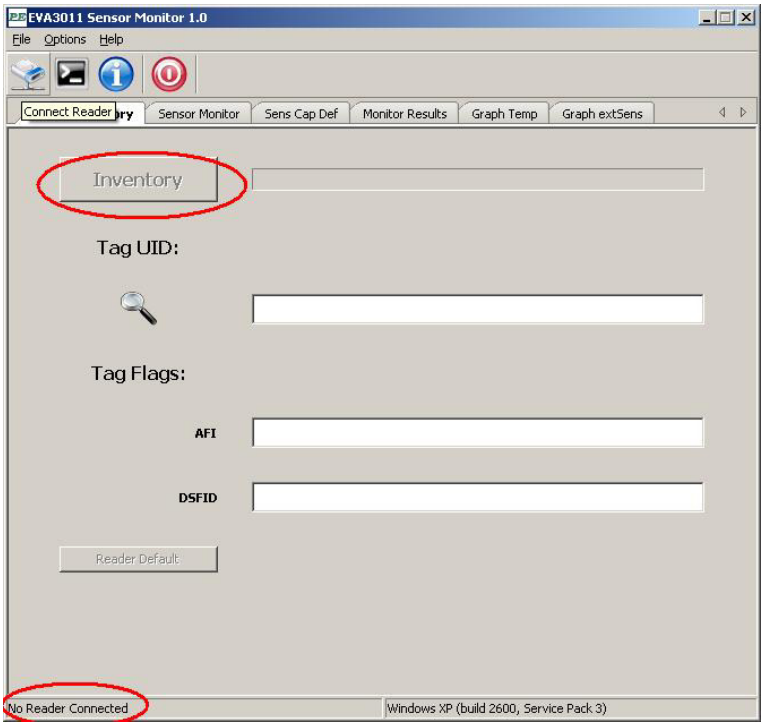
After starting the software (over “Start” Menu) and the access of a HF-Reader the connection can be established. For this action the menu **"File"** and the point **"Connect Reader"** or the icon should be used.



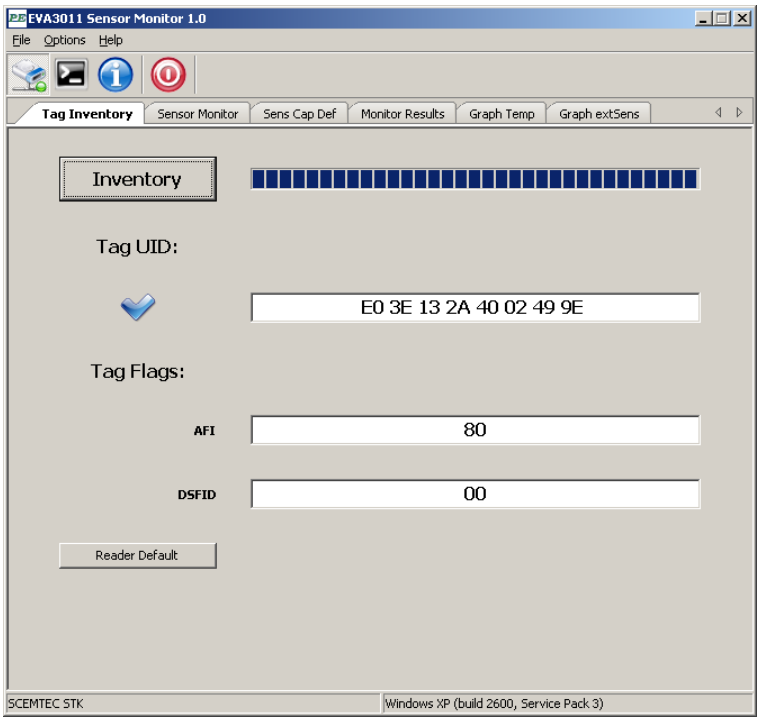
After pressing “Scan” it will now scan the ports for a connected reader. If it finds a valid reader the connection can be established by confirming **"OK"**.



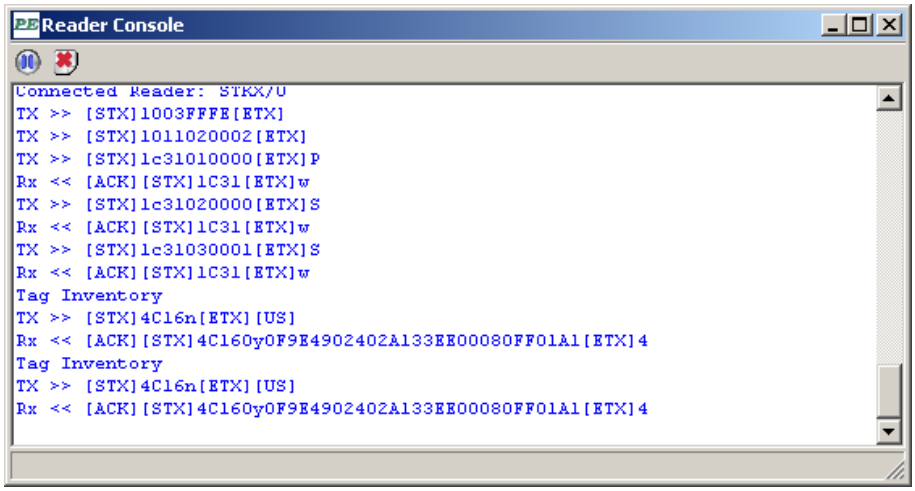
If no reader is found the “Inventory” button is not usable and the software is working without any tag information or activities.



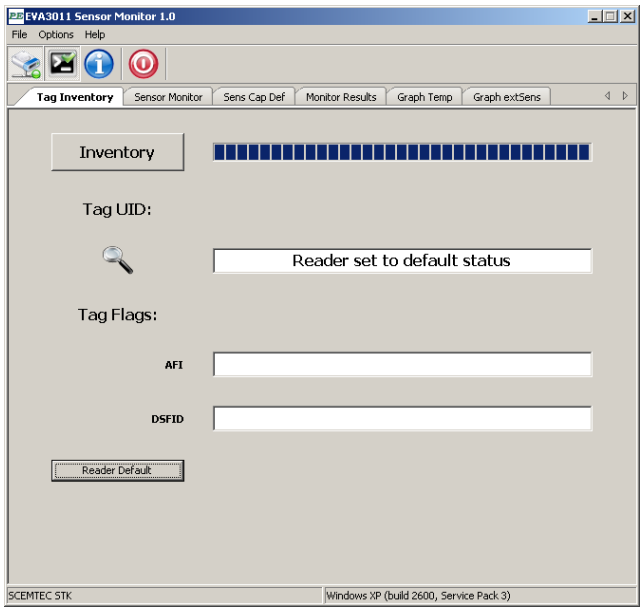
After a successful reader connection the software can **"Inventory"** a tag for UID, AFI and DSFID data.



It is possible to use the Reader Console (menu **"OPTIONS"** -> **"Reader Console"** or the icon) to see the internal data transfer between software reader and tag.



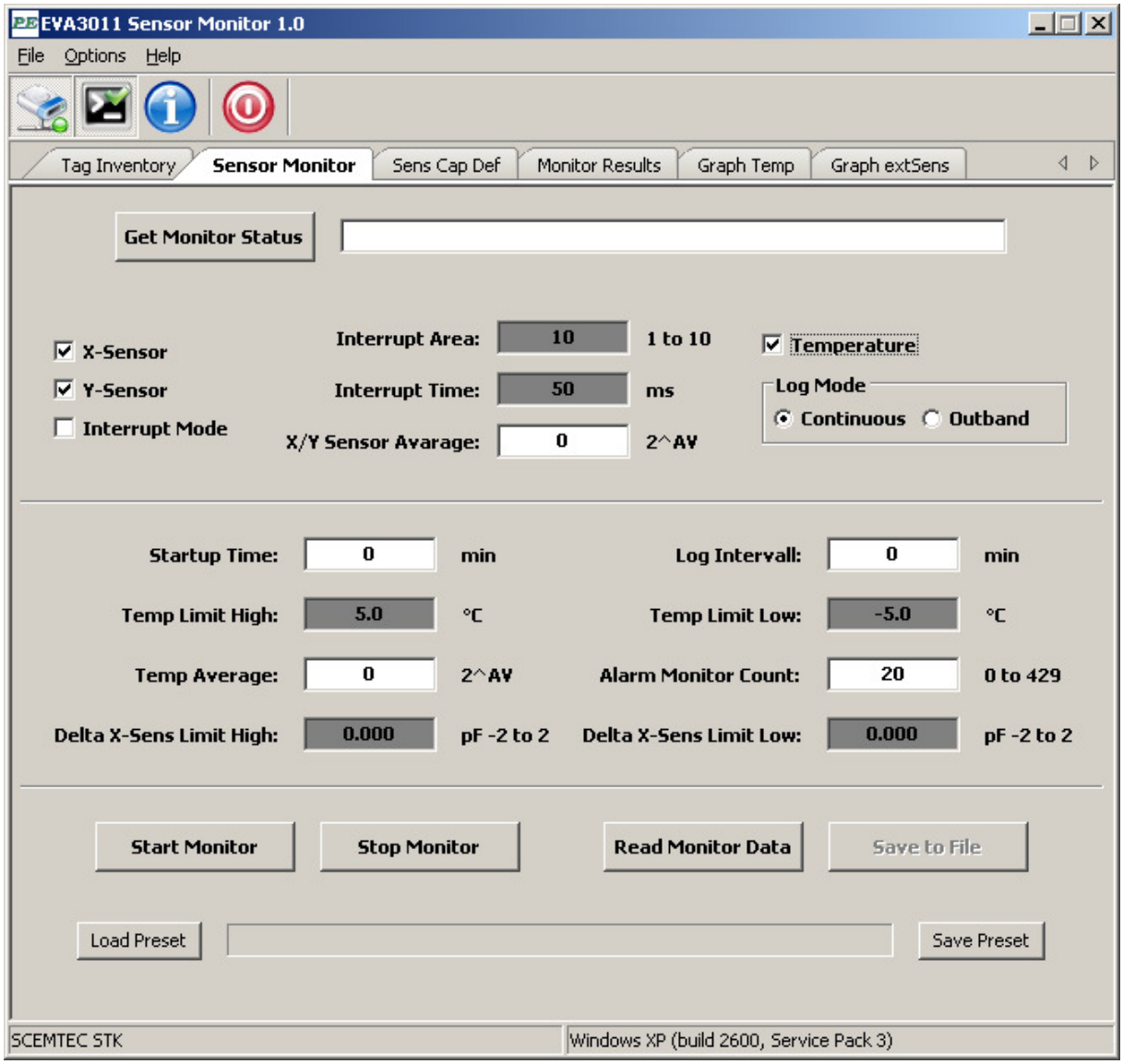
The button **"Reader Default"** sets all reader specific flags to default configuration. This is necessary if the reader is not set to ISO15693 conformity.



This way all reader and tag pre-definitions are completed.

3.3 Software description – Sensor Monitor

To read and manipulate the Sensor Monitor of the PE3011 tag the tab **"Sensor Monitor"** can be used.



Get Monitor Status

At first it is possible to use the button **"Get Monitor Status".** This shows the current status of the Data Monitor in the text field. It is possible to see the actual status of the tag at every cycle of the monitoring.

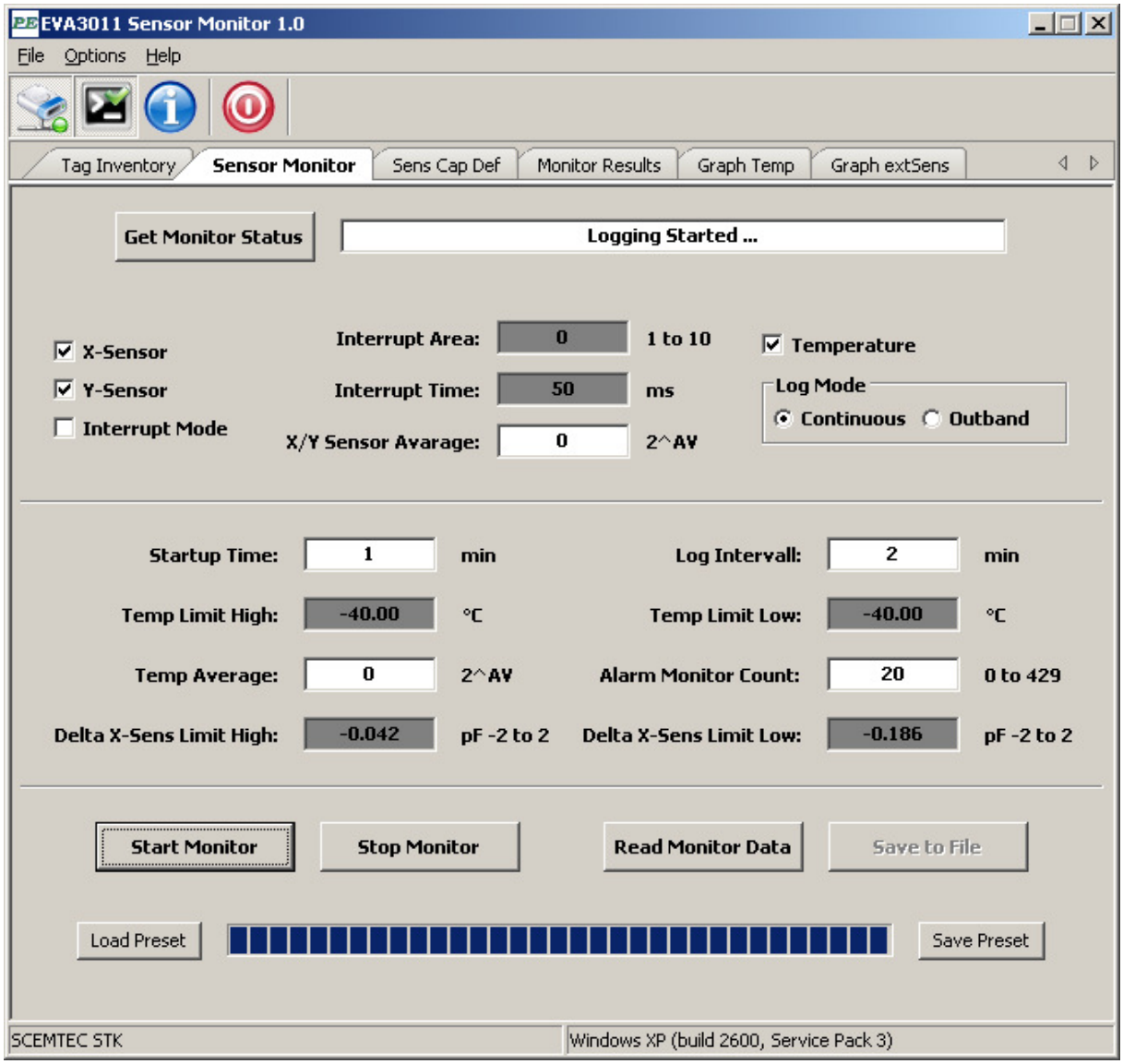
Select X-Sensor, Y-Sensor, Temperature, Interrupt Mode

These check boxes define the available sensor modes to monitor measured values. Basis for these modes are the connected sensors on tag. If one wants to use the Y-Sensor the X-Sensor must be activated and available. In Interrupt Mode only the X-Sensor is usable. If no X or Y-Sensor is activated the Temperature Mode will be used. Depending on sensor mode other set input values for sensing can be activated.

Interrupt Area	Active in Interrupt Mode. Set the number of interrupt measurements for the X-Sensor, (min 1; max 10; default 10; 0x14)
Interrupt Time	Active in Interrupt Mode. Set the time between interrupt measurement in milliseconds (typical is 10 ms) for the X-Sensor, (min 10; max 100; default 10; 0x11)
X/Y Sensor Average	Active if Y and/or X-Sensor is selected. 2 over x measured values are being used to build the mean value, mean value is stored as one X and/or Y value, (min 0; max 7; default 0; 0x11)
Log Mode	The Data Monitor has two **"Log Mode"** definitions in Temperature Mode. In **"Continuous"** mode the chip will sample and log data after the **"Startup Time"** at every **"Log Interval".** In **"Outband"** mode the chip will sample and log data which are higher or lower than the values defined in **"Temp Limit High"** respectively **"Temp Limit Low"** after the **"Startup Time"** at every **"Log Interval" for Temperature mode and if used X/Y-Sensor**.
Startup Time	time before the monitor is starting in minutes, wait-till-log-starts, (min 0; max 6000; default 1; 0x15)
Log Interval	time between measurements in minutes, (min 1; max 255; default 1; 0x15)
Temp Limit High	upper limit in "outband" mode in °C, not usable in Continuous Log Mode, (min -20.0; max 80.0; default 5.0; 0x12)
Temp Limit Low	lower limit in outband mode in °C, not usable in Continuous Log Mode, (min -20.0; max 80.0; default -5.0; 0x12)
Temp Average	2 over x measured values are being used to build the mean value, mean value is stored as one temperature value, (min 0; max 15; default 0; 0x14)
Alarm Monitor Count	set alarm port to high if MonitorCounter > Alarm Monitor Count, (min 0; max 429; default 20; 0x14)
X-Sens Limit High	delta of upper limit to X calibration value in Interrupt Mode in pF, (min -2.000; max 2.000; default 0.000; 0x16)
X-Sens Limit Low	delta of lower limit to X calibration value in Interrupt Mode in pF, (min -2.000; max 2.000; default 0.000; 0x16)

Start Monitor

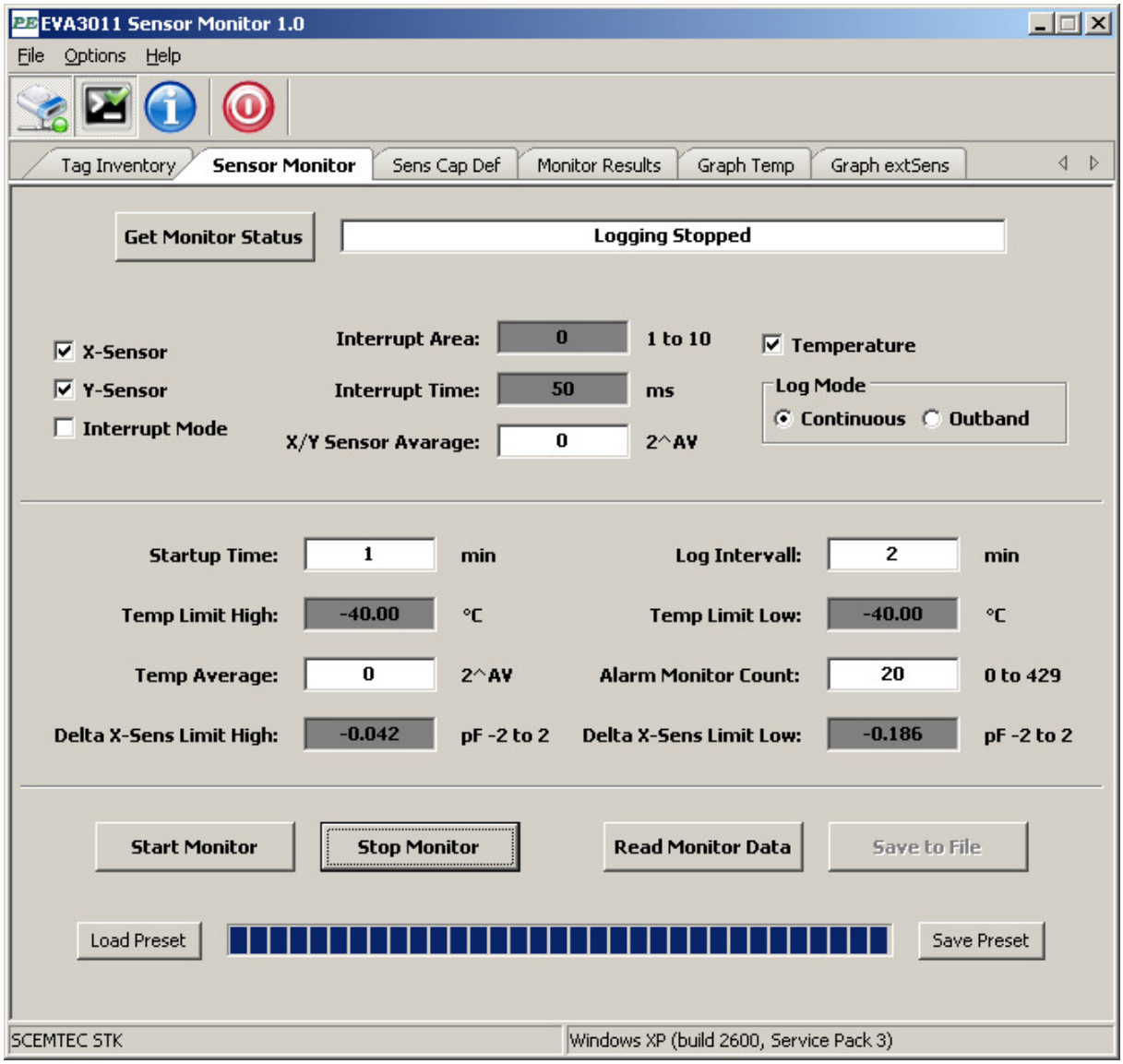
The button **"Start Monitor"** resets the monitor status (interrupts a running measurement), writes all configuration from the Window to the MONREG bank and restarts the measurement Sensor Mode and Log Mode constrained. After this event the PE3011 tag can be used as autonomic sensor measurement system just supported by a battery.



Note: It is necessary to calibrate the sensor on analog capacitive sensor interface to use external sensors. This will be described in tab “Sens Cap Def” -> “Calibration” below.

Stop Monitor

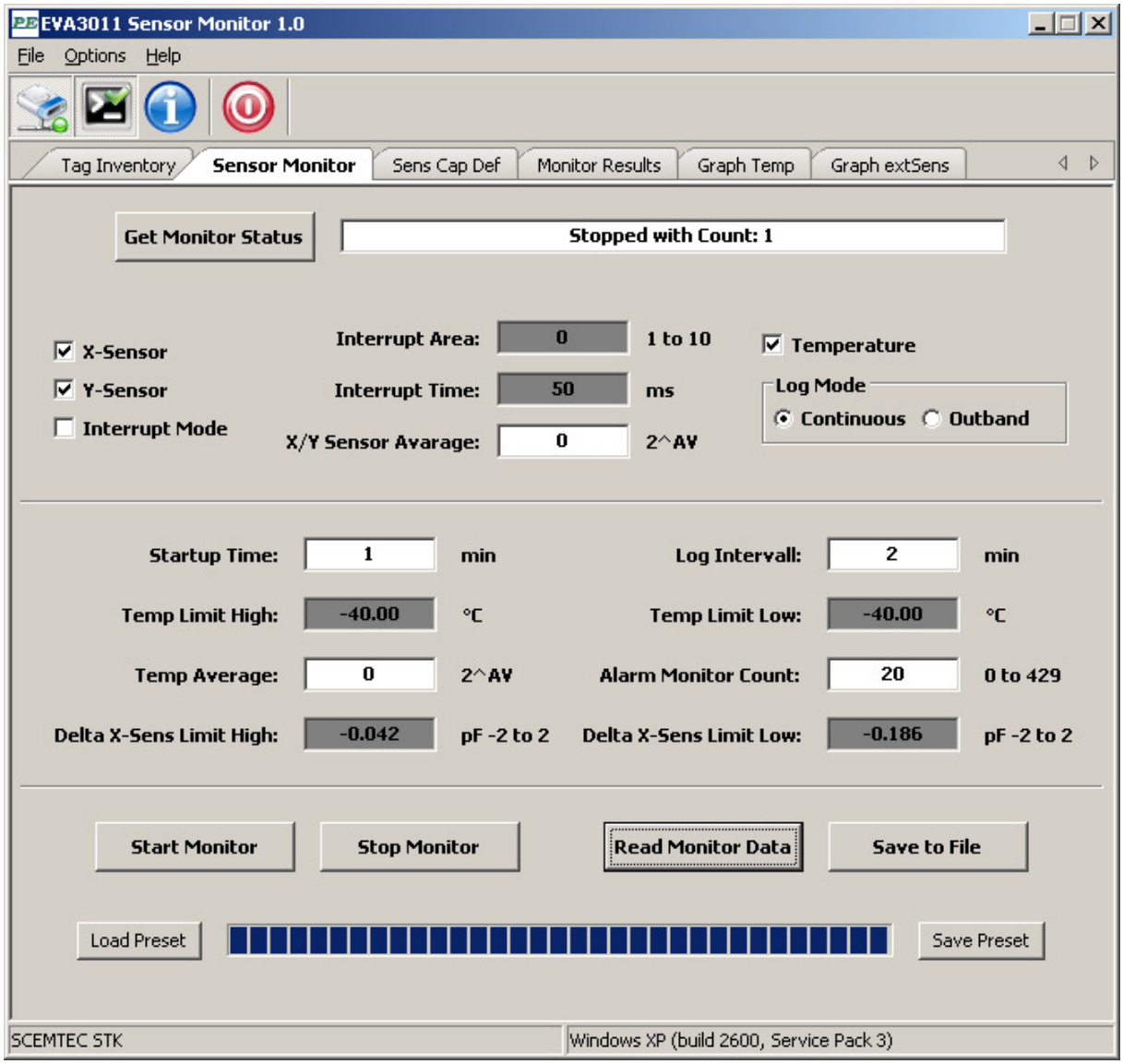
Push the button **"Stop Monitor"** to stop any sensor measurement cycle. This button stops all measurements (timing dependent), saves all actual monitor data and displays the actual status in the Window.



Note: If Monitor started with “Startup Time” then a stop of the monitoring cycle is only possible after this Startup Time. After Startup Time it is possible to stop monitoring at any time.

Read Data Monitor

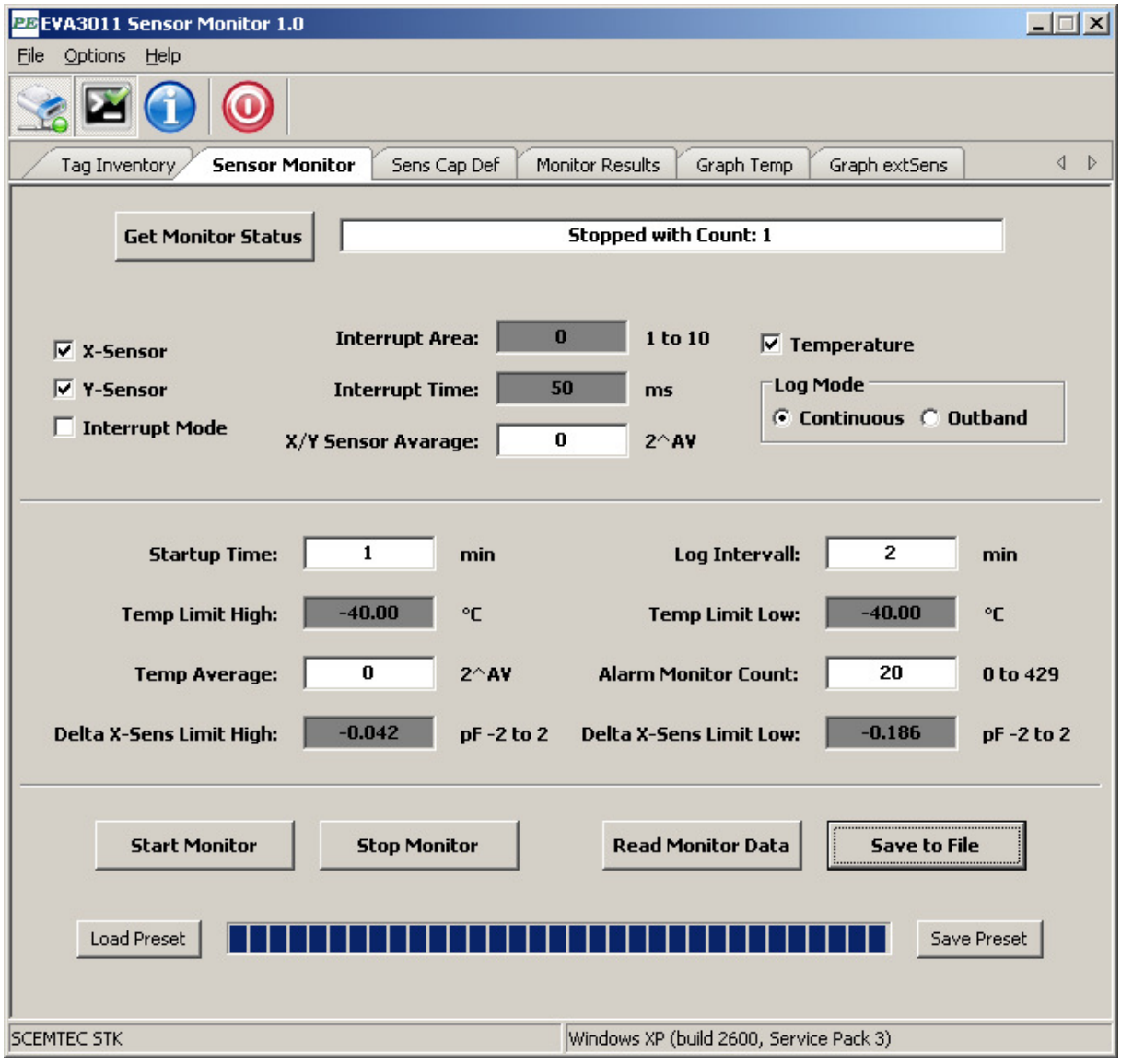
It is possible to read all monitor and tag relevant data at any time from the PE3011 tag. The main information will be displayed in the Window.



A MemoryDump.txt File will be created in the execution directory. This File shows the complete EEPROM data of the tag.

Save to a File

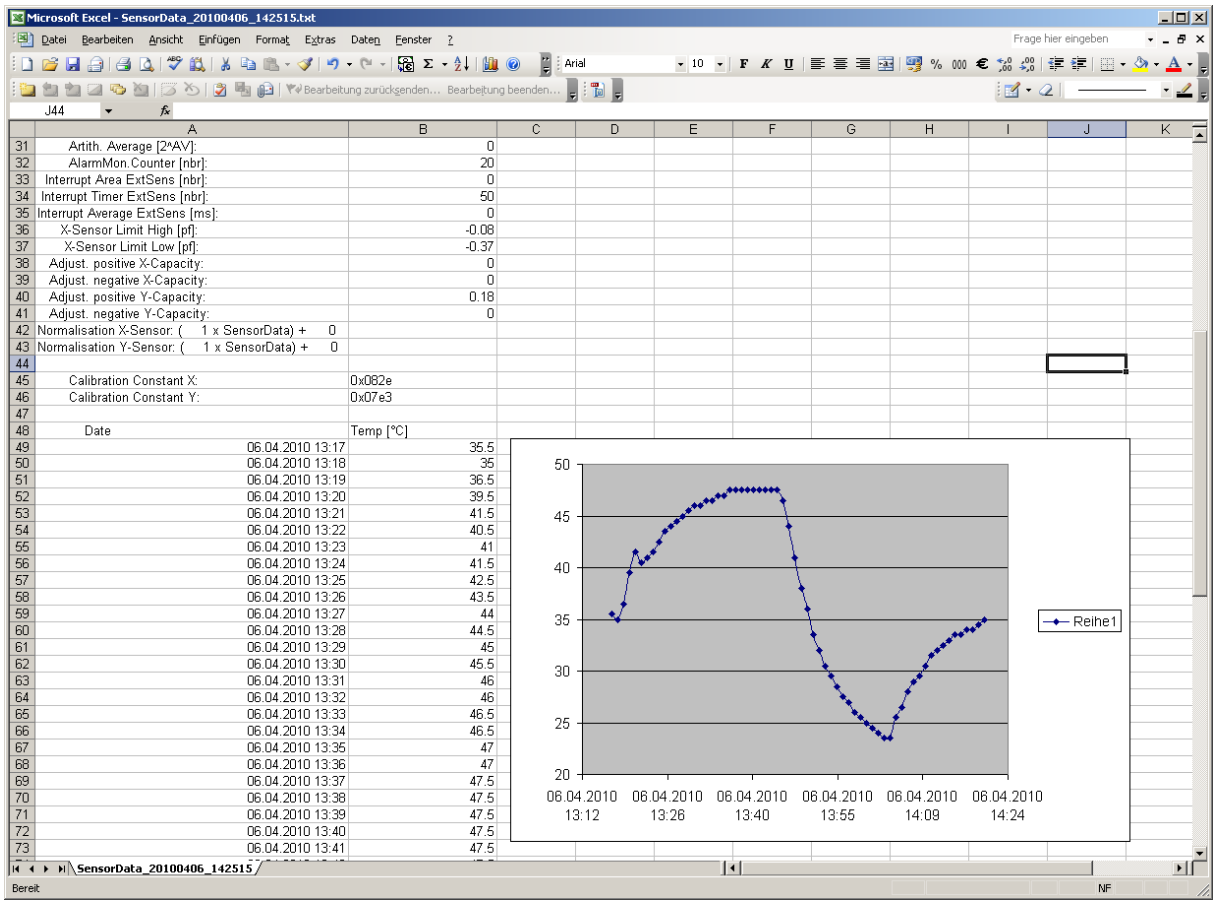
After successful reading of monitor data it is possible to write detailed information to a Text-File (text file with tab-separated values usable in “MS Excel”) in “SensorData_YYYYMMDD_hhmmss.txt” format. Reading back data from the PE3011 can also be done without any battery support.



Save Preset / Load Preset

These buttons allow to save all set parameter in tab “Sensor Monitor” and tab “Sens Cap Def” and their reload into the program. This allows for quick programming of the tag. Data is stored in the execution directory in the file Configuration_ YYYYMMDD_hhmmss.txt. The file should not be changed manually.

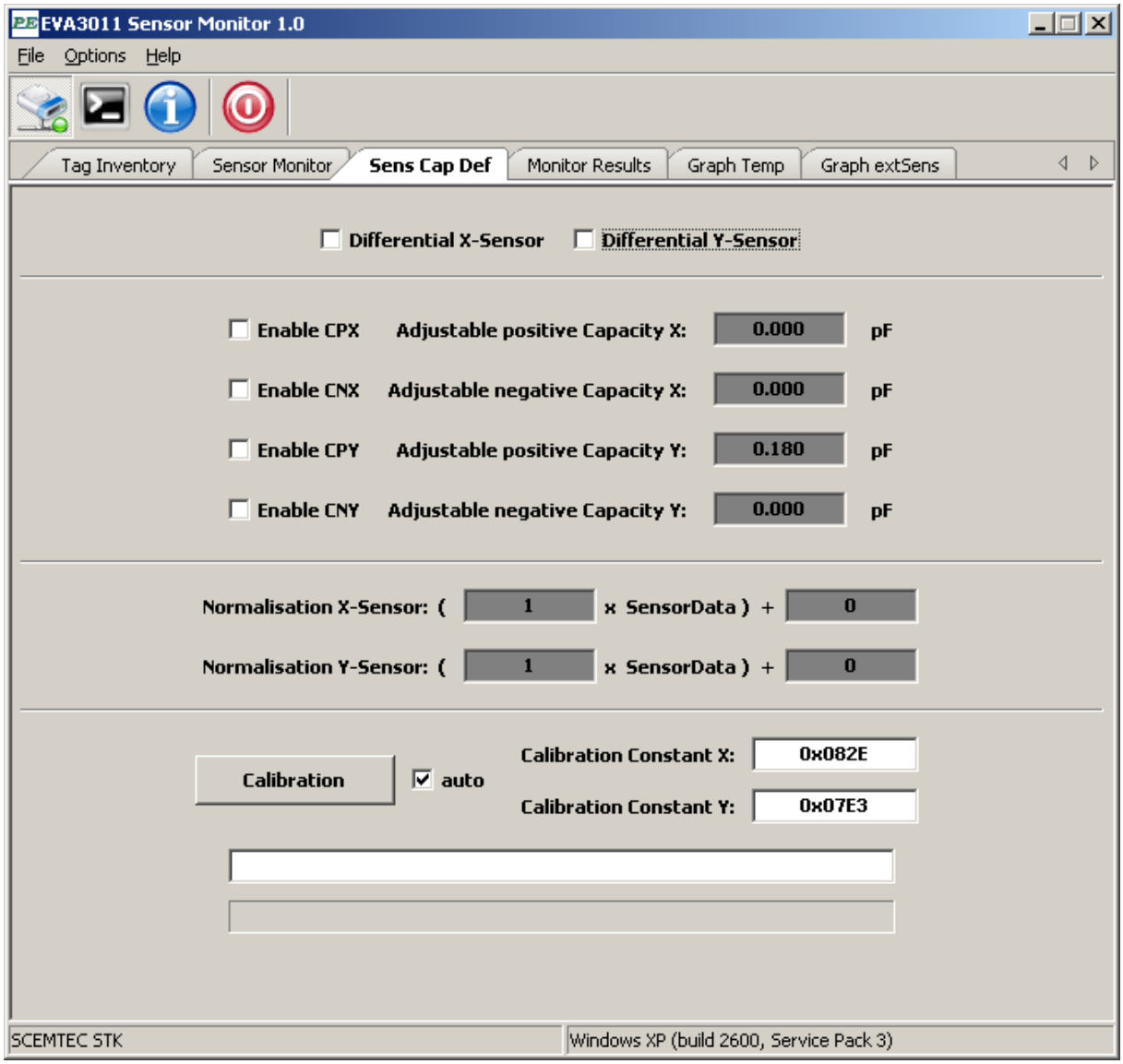
The current status of Data Monitor is displayed below. The fields show the read values, the limit settings and interval times. At the end of the file the measured sensor data and calculated timing data are provided. Measurements can be displayed as a graph using standard MS Excel functions.



Note: To use measured data for a graphical view in MS Excel it is necessary to set the decimal separator to a 'Point' in "Extra"->"Options"->"International".

3.4 Software description – Sens Cap Def

To manipulate the external sensors and calibrate the sensors on Y and/or X-Port this tab was programmed. A normalization function completes this tab.



Differential X-Sensor checkbox to select differential (checked) or single ended (not checked) mode on X-Port if X-Sensor (tab Sensor Monitor) is available. (MONREG 0x11)

Differential Y-Sensor checkbox to select differential (checked) or single ended (not checked) mode on Y-Port if Y-Sensor (tab Sensor Monitor) is available. (MONREG 0x11)

Enable CPX / CNX checkbox to use internal adjustable positive/negative capacity if X-Sensor is available. (MONREG 0x1A)

Enable CPY / CNY checkbox to use internal adjustable positive/negative capacity if Y-Sensor is available. (MONREG 0x1A)

Adjustable positive/negative Capacity X

value to set internal capacity for fine tuning of external sensor on X-Port if Enable is set. (min 0; max 2.000; default 0; MONREG 0x1A), During Calibration with auto mode automatic set.

Adjustable positive/negative Capacity Y

value to set internal capacity for fine tuning of external sensor on Y-Port if Enable is set. (min 0; max 2.000; default 0; MONREG 0x1A) During Calibration with auto mode automatic set.

Normalisation X-Sensor

Gain and offset for measured sensor results on X-Sensor. (MONREG 0x18) default Gain = 1, Offset = 0.

Normalisation Y-Sensor

Gain and offset for measured sensor results on Y-Sensor. (MONREG 0x19) default Gain = 1, Offset = 0.

Calibration (default auto)

Button to calibrate the external Sensor. This function is automatic with active checkbox “auto” and sets the “Calibration Constant” and “Adjustable positive/negative Capacity” for the used external Sensor. If checkbox auto is disabled a manual calibration is possible.

Note: During calibration it is necessary to hold the external sensors in a defined static mode.

Calibration Constant X

Show the value after calibration or after “Load Preset”. (MONREG 0x1B)

Calibration Constant Y

Show the value after calibration or after “Load Preset”. (MONREG 0x1B)

Calibration Status Display

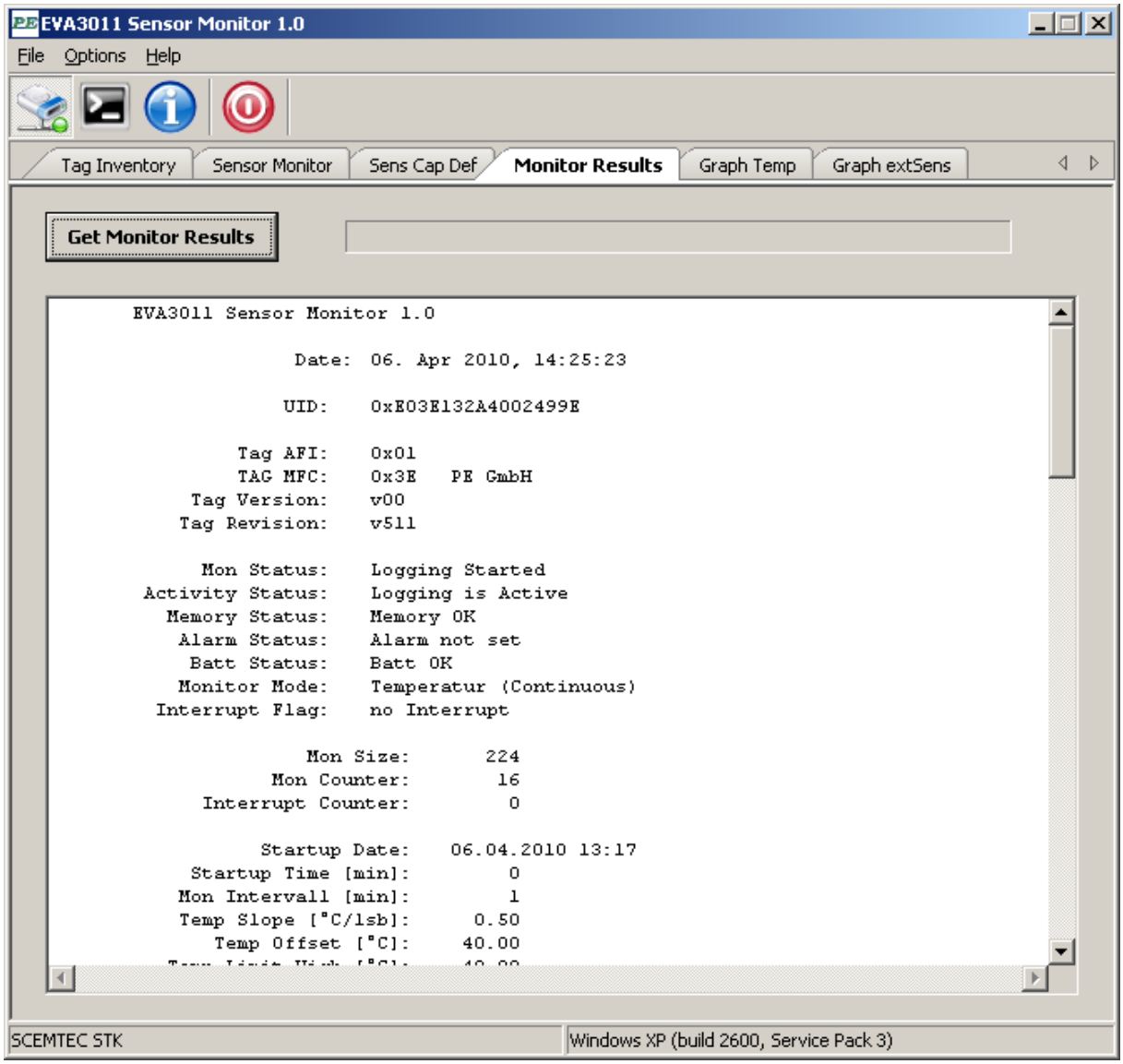
Show the status during calibration.

3.5 Software description – Monitor Results

The tab is designed to read the tag and monitor information directly from tag internal memory.

Get Monitor Results

It is possible to use the button **"Get Monitor Results",** this shows the current content of tag memory as ASCII text.

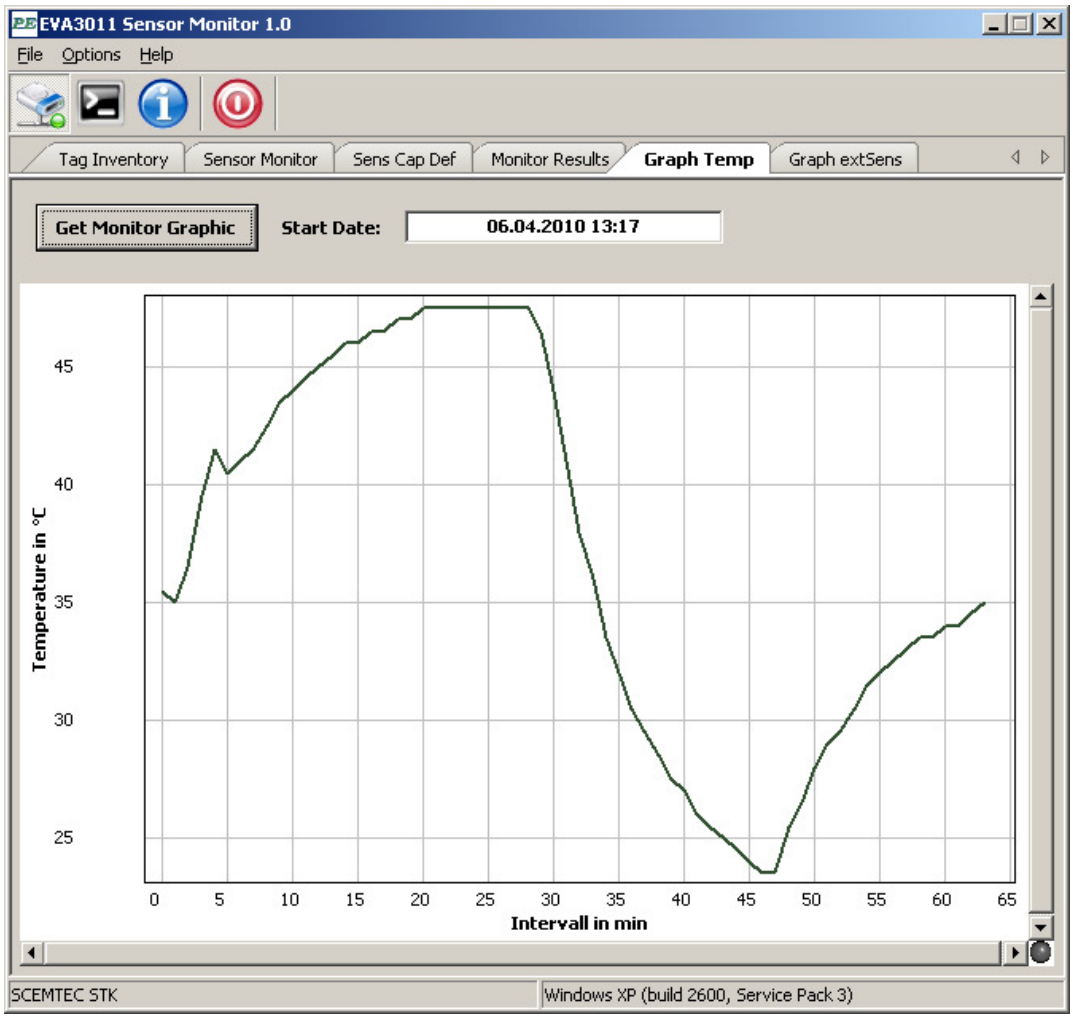


3.6 Software description – Graph Temp

To view a simple graphic diagram on measured temperature data directly from tag it is possible to use this tab. This view is only a memory map if a tag is on reader and measured data are available.

Get Monitor Graphic

It is possible to use the button **"Get Monitor Graphic",** this shows the current monitored time and temperature information in a simple diagram. The x-axis shows timing information in minutes after starting monitoring. Start date is displayed in the window. The y-axis shows the temperature (°C) in relation to timing information.



The zero point of the x-axis is the Start Date and the start of the curve is the first temperature information in minutes after Startup Time (see tab “Sensor Monitor” -> ”Startup Time”).
Hot key functions to manipulate graphic curve:

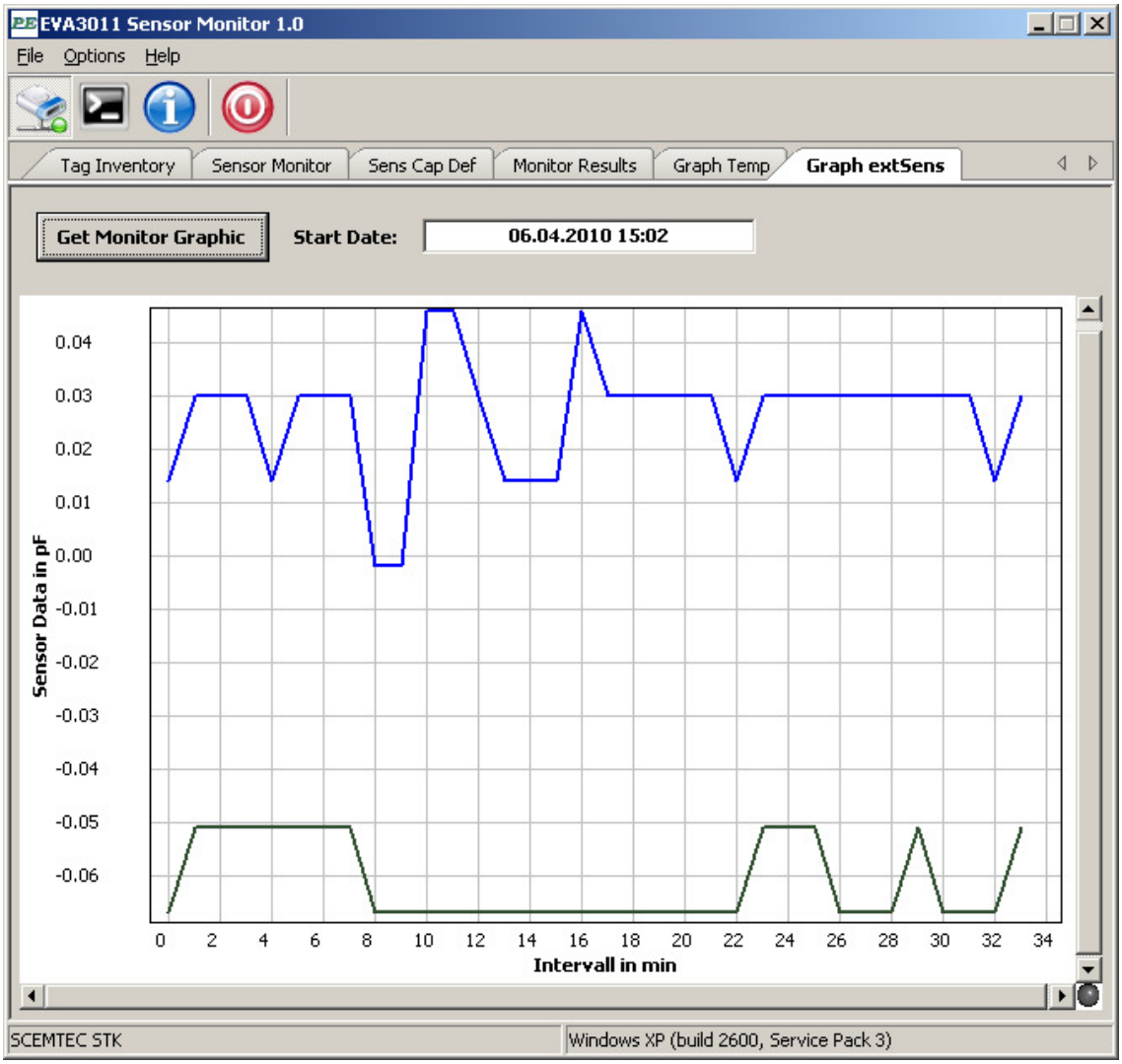
'e'	- zoom in	'z'	- zoom out	's'	- zoom fit	'd'	- zoom in x
'a'	- zoom out x	'w'	- zoom in y	'x'	- zoom out y	'left'	- pan left
'right'	- pan right	'up'	- pan up	'down'	- pan down		

3.7 Software description – Graph extSens

To view a simple graphic diagram on measured external data directly from tag it is possible to use this tab. This view is only a memory map if a tag is in reading distance and measured data are available.

Get Monitor Graphic

It is possible to use the button **"Get Monitor Graphic",** this shows the current monitored time and sensor information in a simple diagram. The x-axis shows timing information in minutes after starting monitoring. Start date is displayed in the window. The y-axis shows the sensor data in relation to timing information.



The zero point of the x-axis is the Start Date and the start of the curve is the first sensor information in minutes after Startup Time (see tab “Sensor Monitor” -> ”Startup Time”).
Hot key functions to manipulate graphic curve:

'e'	- zoom in	'z'	- zoom out	's'	- zoom fit	'd'	- zoom in x
'a'	- zoom out x	'w'	- zoom in y	'x'	- zoom out y	'left'	- pan left
'right'	- pan right	'up'	- pan up	'down'	- pan down		

4 Memory definition

The memory for the Data Monitor is inside the RFID IC and is organized as specified in the ISO15693 standard. The monitor configuration and the trimming values are stored in the MONREG bank.

Tag Bank	EEPROM Block	Data
UID	00h	VICC, DSFID, Flags ISO15693 conform
	01h	AFI ISO15693 conform
	02h	UID[63:32] ISO15693 conform
	03h	UID[31:0] ISO15693 conform
	04h-0Bh	LOCK ISO15693 conform
	0Ch-0Dh	RFU
	0Eh	TempScaling
	0Fh	MonCycle
MONREG	10h	MonCount
	11h	ExtSensFlags
	12h	Temp, RTC
	13h	TMS
	14h	Area
	15h	TimeDef
	16h	IntDataX
	17h	IntDataY
	18h	CoeffNormX
	19h	CoeffNormY
	1Ah	CPCN
	1Bh	CalibConst
MONDAT Read Only!	1Ch	IntDatExtSense
	…	IntDatExtSense
	1Ch+IntDataArea-1	IntDatExtSense
	1Ch+IntDataArea	MonitorData
	…	MonitorData
	MonDataEnd	MonitorData
USER Bank 11b	MonDataEnd+1	UserDat
	…	UserDat
	FFh	UserDat

All measured data will be stored in MONDAT-Bank. In USER-Bank additional data, like monitor start date, can be stored. The edge between USER and MONDAT-Bank can be changed by configuration byte MonDataEnd.

The MONDAT-Bank is defined for measured data from sensor and timing information. For the measurement several Modes are defined - collection of all values, acquisition of all values outside a defined boundary area, interrupt mode for external sensor or a combination of all modes.

Note: PE GmbH delivers samples and production volume ICs in calibrated condition (RTC, TMS). The nature of the chip allows everybody to access AND change these values at any time as long as the MONREG-Bank is not locked. If the MONREG bank will be locked the data monitoring setup can not be changed any more. The values are trimmed in the test process during manufacturing and are guaranteed to be within specified limits.

4.1 Header for PE3011 Tag definition in C++

The complete source code is available from Productivity Engineering GmbH.

```
/*************************************************************
 * Name:    tagdefs.h
 * Purpose: tag defines
 * Author:  Jens Grosse
 * Created: 02-2010
 *************************************************************/

#ifndef __tagdefs__
#define __tagdefs__

// Sensor definitions
#define FSENSSLOPEDIV   1
#define FSENSSLOPESINGLE   2
#define FSENSOFFSET  127
#define FSENSFEMTO2PICO  0.001
#define FYSENSSLOPE   0.1
#define FYSENSOFFSET  0
#define BANK_USER     0x03
#define BANK_RESERVED 0x00
#define ICMFG_ADR     0xA0
#define ICMFG_DATA    0x3E


// memory size
#define MONREG_INTDATAAREA 0x05
#define MONREG_MONDATAEND  0xFA

// memory bank size
#define MIN_UID      0x00
#define MAX_UID      0x0F
#define MIN_MONREG   0x10
#define MAX_MONREG   0x1B
#define DELTA_MONREG MAX_MONREG - MIN_MONREG
#define MAX_USER     0xFF // user bank size (USER_SIZE = MAX_USER - MONDATAEND)
#define MIN_MONDAT   0x1C // reserved bank size (RESERVED_SIZE = MIN_MONDAT + MONDATAEND)
#define MAX_MONDAT   0xFF

// memory addresses epc bank
#define UID_FIRST      0x00
#define UID_TMPSCALING 0x0E
#define UID_UID2       0x02
#define UID_UID1       0x03
#define UID_AFI        0x01

// memory addresses tid bank
#define MONREG_FIRST      0x10
#define MONREG_MONSTATUS  0x10 - MONREG_FIRST
#define MONREG_SENSFLAGS  0x11 - MONREG_FIRST
#define MONREG_TEMPLIMIT  0x12 - MONREG_FIRST
#define MONREG_AREA       0x14 - MONREG_FIRST
#define MONREG_TIMER      0x15 - MONREG_FIRST
#define MONREG_XSENSELIM  0x16 - MONREG_FIRST
#define MONREG_YSENSELIM  0x17 - MONREG_FIRST
#define MONREG_NORMX      0x18 - MONREG_FIRST
#define MONREG_NORMY      0x19 - MONREG_FIRST
#define MONREG_CPCN       0x1A - MONREG_FIRST
#define MONREG_CALIB      0x1B - MONREG_FIRST
#define MONREG_MONSTATUSADR  0x10
#define MONREG_SENSFLAGSADR  0x11
#define MONREG_TEMPLIMITADR  0x12
#define MONREG_AREAADR       0x14
#define MONREG_TIMERADR      0x15
#define MONREG_XSENSELIMADR  0x16
#define MONREG_YSENSELIMADR  0x17
#define MONREG_NORMXADR      0x18
#define MONREG_NORMYADR      0x19
#define MONREG_CPCNADR       0x1A
```

```
// memory addresses user bank
#define USER_STARTDATE 0xFF
#define USER_STARTDATEADR 0xFF

// memory addresses reserved bank
#define MONDAT_FIRST   0x1C

// MONREG block status flag register
#define GET_TEMPSLOPE(reg)      ((reg & 0x0000FF00) >> 8)
#define GET_TEMPOFFSET(reg)     ((reg & 0x000000FF) >> 0)
#define GET_MONDATAEND(reg)     ((reg & 0x000000FF) >> 0)
#define GET_MONCOUNT(reg)       ((reg & 0x000000FF) >> 0)
#define GET_INTADDRCNT(reg)     ((reg & 0x00FF0000) >> 16)
#define GET_TAGAFI(reg)         ((reg & 0x000000FF) >> 0)
#define GET_FIRMA(reg)          ((reg & 0x00FF0000) >> 16)
#define GET_VERSION(reg)        ((reg & 0x00FF0000) >> 16)
#define GET_MSI(reg)            ((reg & 0x0000FFFF) >> 0)
#define GET_MONSTATUS(reg)      ((reg & 0x0000FF00) >> 8)
#define GET_STARTTIME(reg)      ((reg & 0xFFFF0000) >> 16)
#define GET_INTERVALLTIME(reg)  ((reg & 0x000000FF) >> 0)
#define GET_TEMPLIMITHI(reg)    ((reg & 0xFF000000) >> 24)
#define GET_TEMPLIMITLO(reg)    ((reg & 0x00FF0000) >> 16)
#define GET_MWSIZE(reg)         ((reg & 0x00000F00) >> 8)
#define GET_ALARMMONCNT(reg)    ((reg & 0x00FF0000) >> 16)
#define GET_SENSFLAGS(reg)      ((reg & 0x0000000F) >> 0)
#define GET_DIFFFLAGS(reg)      ((reg & 0x00000300) >> 8)
#define GET_INTDATAREA(reg)     ((reg & 0xFF000000) >> 24)
#define GET_INTSAMPLE(reg)      ((reg & 0x00FF0000) >> 16)
#define GET_INTMWSENSE(reg)     ((reg & 0x00000030) >> 4)
#define GET_INTDATAHI(reg)      ((reg & 0x0FFF0000) >> 16)
#define GET_INTDATALO(reg)      ((reg & 0x00000FFF) >> 0)
#define GET_CPX(reg)            ((reg & 0x00003F00) >> 8)
#define GET_CPY(reg)            ((reg & 0x3F000000) >> 24)
#define GET_CNX(reg)            ((reg & 0x0000003F) >> 0)
#define GET_CNY(reg)            ((reg & 0x003F0000) >> 16)
#define GET_GAIN(reg)           ((reg & 0x0FFF0000) >> 16)
#define GET_OFFS(reg)           ((reg & 0x00000FFF) >> 0)
#define GET_CALIBX(reg)         ((reg & 0x00000FFF) >> 0)
#define GET_CALIBY(reg)         ((reg & 0x0FFF0000) >> 16)
#define GET_RTC(reg)            ((reg & 0x0000FFFF) >> 0)
#define GET_CLK1M(reg)          ((reg & 0x0F000000) >> 24)
#define GET_ACTDATE(reg)        ((reg & 0xFFFFFFFF) >> 0)
#define GET_LOGWORD(reg)        ((reg & 0x00FF3FFF) >> 0)

// reserved block data (logsize = 0x1AE)
#define TEMP_START  0x1C // 384 blocks

// status values
#define MONSTATUS_STARTLOG    0x80
#define MONSTATUS_MEMOVERFLOW 0x20
#define MONSTATUS_ACTIVE      0x40
#define MONSTATUS_ALARM       0x04
#define MONSTATUS_BATT        0x08
#define MONSTATUS_OUTBAND     0x02
#define MONSTATUS_INTEXT      0x01
#define SENSFLAG_XSENSOR      0x02
#define SENSFLAG_YSENSOR      0x04
#define SENSFLAG_TEMP         0x01
#define SENSFLAG_INTMODE      0x08
#define DIFFFLAG_DIFFENX      0x01
#define DIFFFLAG_DIFFENY      0x02
#define CPN_CPENX             0x00004000
#define CPN_CPENY             0x40000000
#define CPN_CNENX             0x00000040
#define CPN_CNENY             0x00400000
```

```
// calibration values
#define CAL_TEMPSLOPEDIV  100.0 //
#define CAL_TEMPOFFSETDIV  1.0 //
#define CAL_XSLOPEDIV  0.200 // 200 fF/bit
#define CAL_XOFFSETDIV  0.0 //
#define CAL_YSLOPEDIV  0.200 // 200 fF/bit
#define CAL_YOFFSETDIV  0.0 //

#define CAL_LIMIT2CNTDIF  1000.0 //
#define CAL_LIMIT2CNTSINGLE  500.0 //

#define CAL_CPNSLOPE  0.180 // 180 fF/bit

#endif // __tagdefs__
```

5 Introduction of EVA3011 Hardware

Features PE3011:

- Passive RFID HF transponder chip with integrated sensor monitor
- Compliance according to ISO15693 standard (limited read/write range)
- 8kBit EEPROM read-/writeable access via RF field and/or sensor monitor
- EEPROM memory to store measurement data with time stamp
- Power supply via RF field (for communication) or with battery (sensor monitoring)
- Intelligent power management for different power domains
- Continuous battery control and automatic shutdown
- Extra signal output for “out-of-limits” detection
- Additional function with SPI interface (for using external devices)
- Real Time Clock to provide an accurate clock signal of 8.738 kHz
- Internal temperature sensor and ADC for external capacitive sensors

5.1 PE3011 General Description

The PE3011 is an integrated circuit for tracking and controlling logistics. It monitors temperature, extern capacitive sensor data and related time data of goods during transport or storage. While not in an RFID reader field and so not being supplied through the reader the system draws the required energy from the battery. While in a HF reader field the system is supplied by the reader’s field energy and communicates to the reader based on the standard protocol. Besides standard ISO15693 communication additional functionality to read out temperature or other data is implemented. The integrated SPI interface allows a communication with other external devices like a microcontroller that can provide additional sensor functionality like digital interfaced sensors.

Memory access is granted through the HF reader controller as well as through the data monitoring controller. Both blocks have the same priority. No started memory access will be interrupted by a request for another access. The started access will be finished first before the new access request will be acknowledged. Detailed information can be found in the PE3011 datasheet.

6 How to design an application

Basically the EVA3011 comes as a complete application. The system has a spare connector spot for 2 external capacitive sensors. Sensors can be mounted directly at this location (e.g. humidity). It also features two capacitive touch button areas which can be used as switches along with a metal plate that comes close to the sensor and this way changes the capacitance. This way it is easy to observe an open/closed state of a box. A combination of temperature and humidity monitoring along with a switch function can be realized easily just be adding a capacitive humidity sensor. Good experiences have been made with a sensor of IST AG (www.ist-ag.com), the P14-Rapid. A spare solder location for a Microfab PE1.3N/PE8.0T-3.0SQ (http://www.microfab.de/) capacitive pressure sensor is on the board. Along with a humidity sensor temperature, pressure and moisture can be measured with just this simple system.

6.1 PCB specification

The PCB (printed circuit board) in the original state consists of a network for antenna matching, a battery with a corresponding connector to the PE3011, the two touch sensors and an interface for SPI communication.

Supply voltage (battery) 3.3V

It is possible to assemble more devices, e.g. an optical indication for an active alarm port. Zero ohm jumpers might be necessary to select the different capacitive sensors that can be mounted on the board.

6.2 SPI

When using a microcontroller with SPI it is possible to use also external sensors. The measurement data of these sensors can be written to the EEPROM via the SPI and can be read out through the RFID interface.

Note: Refer to the datasheet of the PE3011 for read/write commands to EEPROM via SPI.

6.3 Alarm Port

The alarm port becomes active, when user defined sensor limits are violated and a measurement cycle is running. Refer to the datasheet of PE3011 for an active alarm. For getting an optical indication when the alarm port is activated, it is necessary to assemble resistor R4 and the LED. The recommend data for R4 and LED can be found in the BOM.
When the alarm port gets activated, the signal is switched to ground and the LED illuminates.

Calculation for series resistor:

$$R_3 = \frac{V_{BAT} - V_F}{I_F}$$

For a red LED with low current the following parameters have to be accounted for:
Forward voltage V_F=1,8V , forward current I_F=2mA, battery voltage 3V:

$$R_3 = \frac{3V - 1{,}8V}{2mA} = 600Ohm$$

R_3=560Ohm shall be chosen here.

Note: The LED can only be applied, when using a 3V battery.

7 Description

The Evaluation Kit was designed to help understand and evaluate the features of the PE3011 HF RFID IC. Used external devices are standard components and do not represent a completely fine-tuned OEM application. The BOM (bill of material) for a final application may look different.

7.1 Evaluation Board Specifications

Parameter	Symbol	Min	Typ	Max	Unit	Notice
Operating temperature	T_{amb}	-40	27	80	℃	
Frequency RF field	f_{rf}		13,56		MHz	
Battery Voltage	V_{BAT}	2,4	3,3	3,6	V	Not possible if LED is used to indicate alarm port
Current Alarm Port	I_{Alarm}		5		mA	Open drain output

7.2 General Schematic

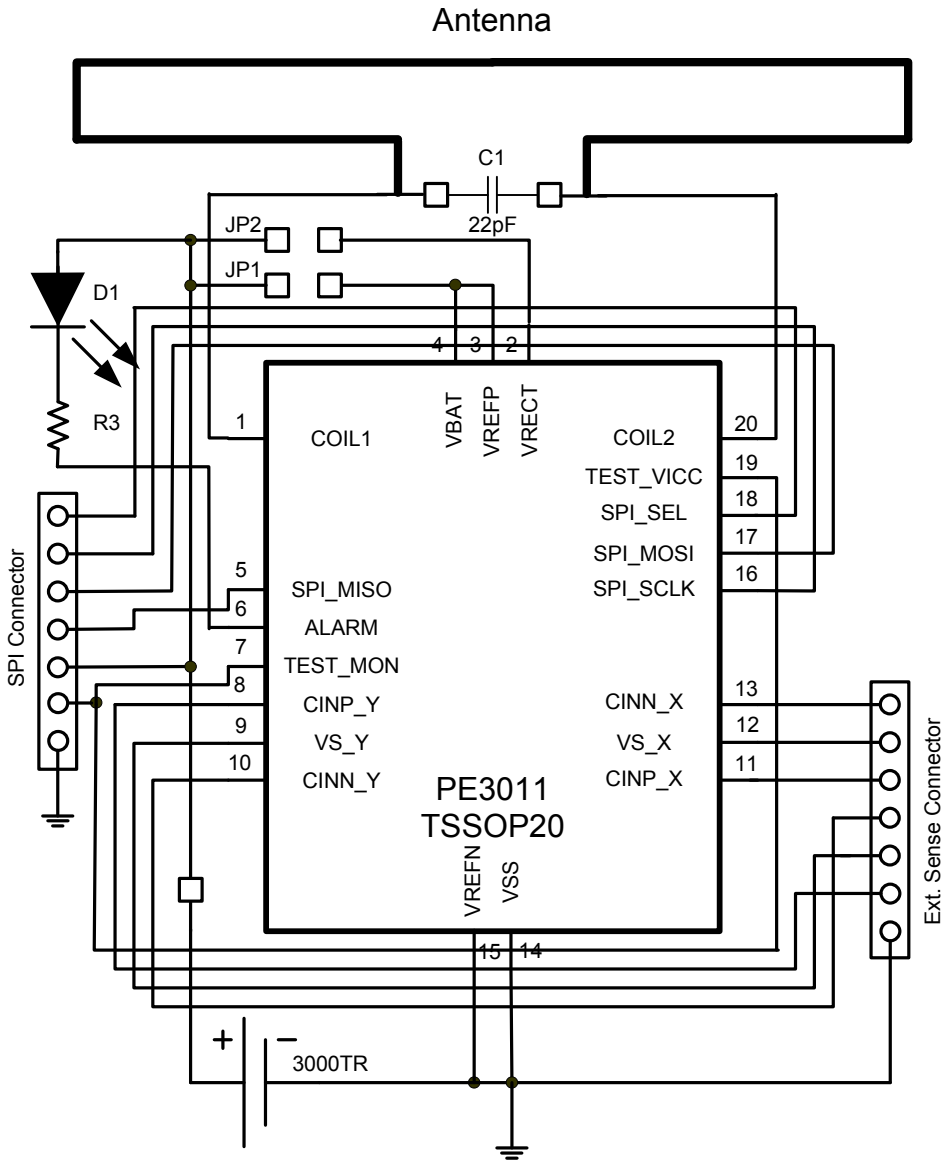


Figure 2 - Basic Schematic of the Evaluation Kit

7.3 Eagle Data

Application documentation and PCB layouts are availabe at the web site www.pe-gmbh.com . Those are meant to serve as basis for new different developments.

7.3.1 Layout

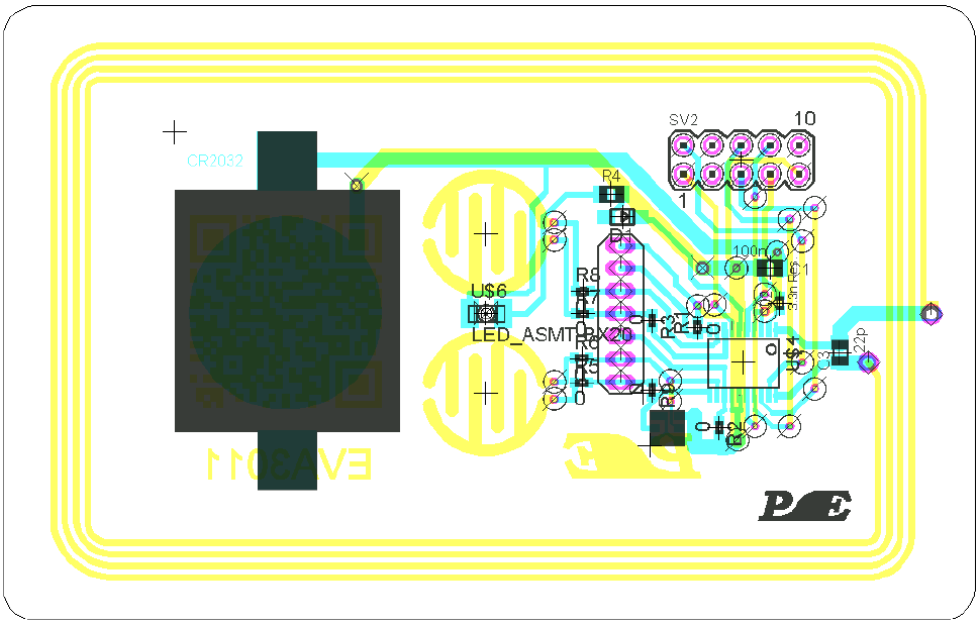


Figure 3 - Layout of the Evaluation Kit (standard credit card format)

7.3.2 Schematic

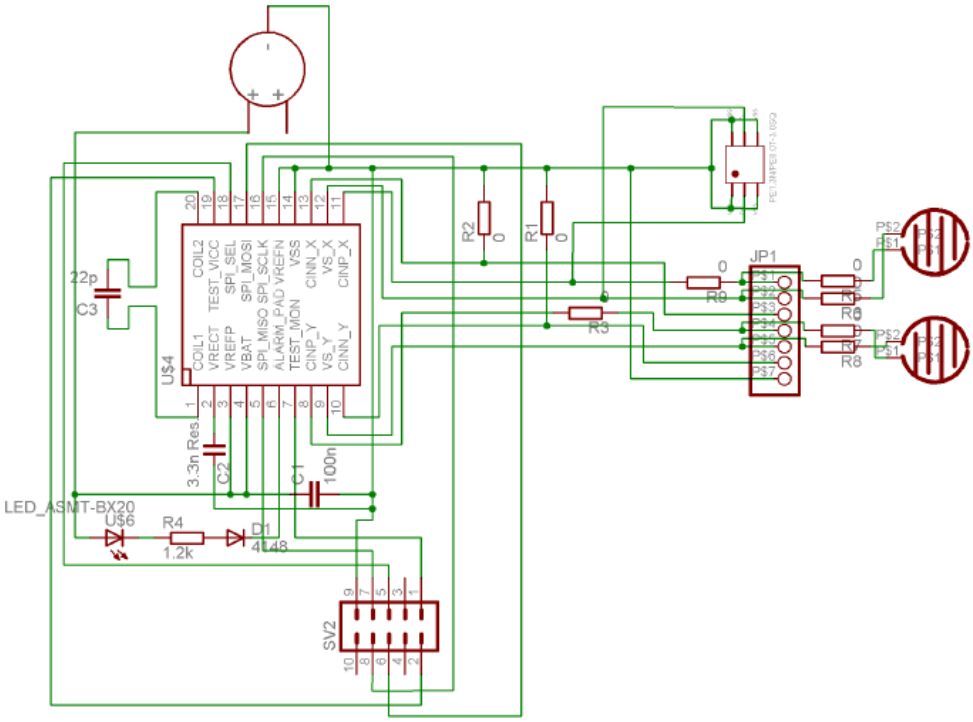


Figure 4 - Complete Eagle Schematic of the EVA3011

C2 is required to overcome read range limitations by buffering the internal digital supply voltage.

8 Operation

8.1 Using only the HF interface

The HF interface is accessible at any time on the EVA3011. The battery has not to be populated for this operational mode. JP1 can be closed or open. The PE3011 can be controlled with a HF Reader. The HF transmission distance can be extended in certain environments by attaching JP1 and JP2 so that the HF functional electrical blocks have battery backup for operation. The antenna network is optimized for good electro-magnetic performance of the antenna with the chip. It is recommended to use the GUI3011 software to configure the PE3011 registers. The chip responds to any standard ISO15693 Reader by sending the UID.

The configuration and measurement commands are described in the PE3011 datasheet.

8.2 Using only the data monitor

For using only the data monitor the battery has to be mounted (default) and JP1 has to be closed.
The PE3011 can be controlled via the SPI interface or the RF field when the GUI3011 software will be used.

The configuration and measurement commands are described in the PE3011 datasheet.

8.3 Using HF interface and data monitor

For using both HF interface and data monitor the same requirements are necessary as described in chapter 8.2
This mode is recommended for measurement tasks.
Accessing the EEPROM is possible in two ways (RFID interface and data monitor). Both have the same priority. If an access to the EEPROM occurs, the memory is locked until the access has finished.
The configuration and measurement commands are described in the PE3011 datasheet.

9 Notes

10 Contact

Germany

Stuttgart

Productivity Engineering
Process Integration GmbH
Behringstrasse 7
D-71083 Herrenberg
Germany
Phone.: +49 (0) 70322798 0
Fax: +49 (0) 70322798 29
Email: info@pe-gmbh.com
Web: www.pe-gmbh.com

Dresden

Productivity Engineering GmbH
Branch
Sachsenallee 9
D-01723 Kesselsdorf
Germany
Phone.: +49 (0) 35204777 00
Fax: +49 (0) 35204777 000
Email: info@pe-gmbh.com