Developing custom firmware for STM 31X and STM 330

1. Introduction

Purpose

This application note will explain:

- How to develop custom firmware applications for STM31x HW and STM330 HW
- Differences between the STM31x and STM330 default firmware

The major purpose of this application note is to show how to create custom firmware applications for STM31x HW and STM330 HW platforms. Although most of the applications can be realized using the default firmware, it can happen that some specific applications require a customized FW. Custom firmware development is possible using the DolphinAPI and DolphinStudio. For this purpose the STM31x and STM330 hardware modules (same hardware core) can be all customer reprogrammed.

To make the custom firmware development as easy as possible we have released the modules default firmware source code. The source code can be downloaded from the software download section: http://www.enocean.com/en/download/. The STM31X HW runs the default firmware called STMSEN. Under the STM330 HW is running the default firmware STMTRH.

NOTE:
The STM31x and STM330 source code is provided on an "AS-IS" basis. Absolutely no warranty of any kind.

Default firmware differences STM31x and STM330

Before starting a custom firmware development it is important to understand the major differences between the default firmware. As both STM31X and STM330 have the same hardware core in theory both STMSEN and STMTRH could run on both HW modules.

The first major difference between STM31X and STM330 hardware module is that STM31X hardware module is not temperature calibrated (you can calibrate STM31X, see AN506 [6]). The second major issue is that STMSEN firmware and interface allocation was design to make it possible to reach the A/D converter rails by connecting an external amplifier to the pins ADIO0, ADIO3, ADIO4 (see AN314 [5]). In STM 330 these pins are not interpreted by the default STMTRH software and are reserved for custom application usage. The following table summarizes the major differences between the default firmware. The last major difference is the way of configuring the sleep period and cyclic retransmission.

	STM330 running STMTRH default firmware	STM31X running STMSEN default firmware
Temperature measurement	Yes	No (you can calibrate STM33X, see AN506 [6])
Temperature calibration values	Yes	No
Rail to Rail A/D measurement	No solution implemented, for set point to eliminate rail errors a lookup table is used which ensures that values between 0x00 an 0xFF will be transmitted.	External amplifier needed for exact measurement
Configuration possibility	CW_0, CW_1 and CP0, CP1	CONF_0 and CONF_1 pins
A/D input channels for external measurement used in firmware	1x analog (setpoint)	3x analog channels 2x analog reference input
Digital input channels used in firmware	1x digital (occupancy)	3x digital
Maskable digital inputs	No	Yes

Customizing the default firmware

Step-by-step start using the source code

This chapter describes how to work with the STMTRH and STMSEN

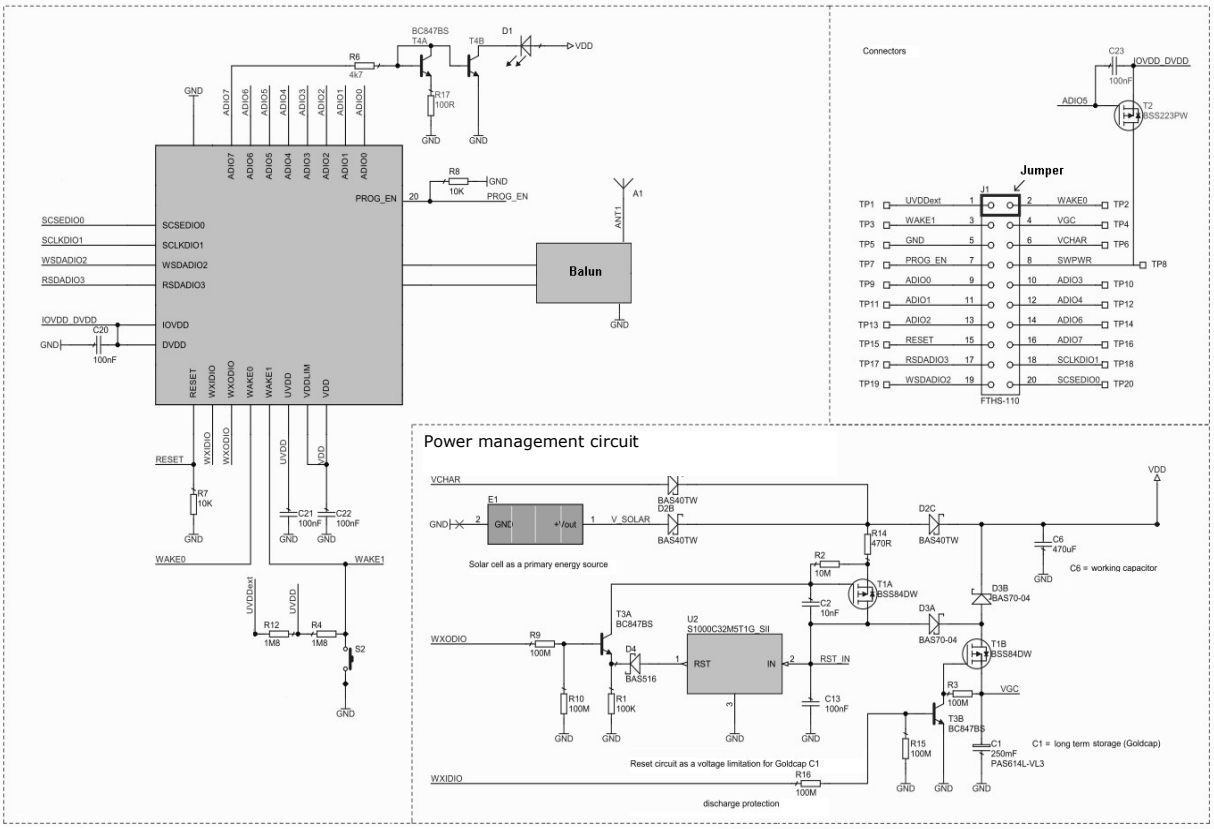
- Make sure you have downloaded and installed the following tools:
  - DolphinAPI
  - DolphinStudio
  - DolphinView
  - Keil uVision and C51 Compiler
- Download the source code zip file from http://www.enocean.com/en/download/
- Extract the zip file to the following directory DolphinAPI\EO3000I_API\Examples\
- Open the project in Keil uVision
- Select the correct frequency target 868 or 315Mhz and compile the software
- Download the software using EOPXIn case you got the following error message:

```
ERROR: Firmware radio settings are not conform with the hardware as
specified in radio_conformity.xml.
```

Copy the radio_conformity.xml that is located in the STMTRH folder to the EnOcean\DolphinStudio\ folder and overwrite the actual file

Functionality described

For the correct module functionality as energy harvesting modules it is essential that the custom FW correctly controls the integrated power-management. Therefore before a custom FW can be developed it is important to understand how the default firmware works.



The STMSEN firmware interprets the HW interfaces in the following way:

1. **LRN** button, represented by S2, connected to **WAKE1.** It is used to send **LRN** telegrams
2. The **WXODIO** and **WXIDIO** pins are in charge of activating power-management circuit depending on specific voltage thresholds.
3. **ADIO7** pin controls the telegram transmission indicator LED D1
4. The analogue external signals **ADIO0, ADIO1** and **ADIO2** are the analogue input pins
5. **ADIO3** and **ADIO4** pins are the Vref+ and Vref- for analogue measurements
6. **ADIO5** activates the external sensors power supply (**SWPWR**) through the T2

transistor (upper right edge) during the measurement.
7. **ADIO6, SCSEDIO0** and **SCLKDIO1** connect to the STM 3xy external digital inputs **DI2, DI0** and **DI1**
8. **WSDADIO2** and **RSDADIO3** set the digital signals **CONF1** and **CONF0** to select the sleep cycle and cyclic sending.

The STMTRH firmware interprets the HW interfaces following way:

1. **WAKE0** is used for external occupancy button
2. **LRN** button, represented by S2, connected to **WAKE1.** It is used to send **LRN** telegrams
3. The **WXODIO** and **WXIDIO** pins are in charge of activating power-management circuit depending on specific voltage thresholds.
4. **ADIO7** pin controls the telegram transmission indicator LED D1
5. The **ADIO0** is for connecting set point dial
6. **ADIO1, ADIO2**, **ADIO3, ADIO4, ADIO6** are not used
7. **ADIO5** activates the external sensors power supply (**SWPWR**) through the T2 transistor (upper right edge) during the measurement.
8. **SCLKDIO1** and **SCSEDIO0** are encoding configuration input (CW_0, CW_1)
9. **RSDADIO3** and **WSDADIO2** are encoding configuration input (CP_0, CP_1)

To understand the firmware functionality in detail, please read the comments in the source code. When customizing STMTRH and STMSEN firmware it has to be ensured that the power-management implementation and HW interface (marked with yellow) is not modified. Failing to do so will lead to improper module functionality.

Power Management function description

The circuit diagram above shows the power management circuit implemented into the STM3XY modules. The charger circuit is controlled via the WXODIO signal which switches automatically (controlled by HW) according to the status of the internal threshold detector (value 1 when Vdd >=2,4V, value 0 otherwise). The SW must only configure the WXODIO pin as Von (using Dolphinstudio and calling function io_init). For details please also refer to Dolphin Core Description documentation. The WXIDIO signal is only used to disconnect the C1 (0.25 F, long term storage) from the STM 3xy load during start-up, in deep sleep mode, and in all active modes during transmitting as explained below. The short term storage working capacitor is the C6 (470 µF).

As the module's supply voltage ramps up, both WXIDIO and WXODIO are LOW. This means that the long term storage C1 is disconnected from the supply (due to WXODIO) and also from the load (due to WXIDIO), permitting a fast start-up through the smaller short term storage C6. WXODIO will always output the state of the internal Von threshold detector (HW controlled). As soon as the supply voltage Vdd of the module rises above the Von threshold level, WXODIO will switch to HIGH state and charging of the long term storage C1 starts (although T1B is still open, charging is possible due to the built in reverse diode of T1B).

Once started, the module will then periodically check the supply voltage (see bVdd_4 variable). When Vdd reaches 3V for the first time , WXIDIO which is software controlled will be switched to HIGH state (see macro GOLDCAP_ON), while the module is in deep sleep mode, connecting the C1 to the module. During high current consumption phases (e.g.

telegram transmissions), WXIDIO will always be set to LOW state to disconnect the long term storage C1 from load and so avoiding the risk of a deep discharge in the event of a power down in the module through a sudden brown-out. In case of no charge (e.g. operation in darkness) the voltage in the storages will go slowly down and during a phase with high current and disconnected long term storage C1 the voltage will fall below the Voff threshold. Then the start-up procedure starts again.

Note:

When the voltage on the C1 reaches a maximal threshold (HW fixed by U2, max. 3.3 V), his RST (HW) output overwrites the WXODIO state and cuts the charging path (T1A) to C1 avoiding an overcharge of the C1. The STM 3xy is however further supplied from power supply directly over D2. VDDLIM connected to Vdd protects STM 3xy against overvoltage.

References

Further details can be found in the following documentation

[1] Dolphin Core Description
http://www.enocean.com/de/enocean_module/tcm-300/

[2] IEEE 754 format
http://en.wikipedia.org/wiki/IEEE_754

[3] IEEE 754 Calculator
http://babbage.cs.qc.edu/IEEE-754/

[4] STM330 source code
http://www.enocean.com/en/download/

[5] AN314 Rail-to-Rail Sensor Applications using the STM 31x
http://www.enocean.com/de/application-notes/

[6] AN506 Dolphin Internal Temperature Sensor Calibration
http://www.enocean.com/de/application-notes/