Application Note

Porting Z-Wave Appl. SW from ZW0102 to ZW0201

Document No.:	**APL10248**
Version:	**4**
Description:	**This document describes the issues related to porting software applications from ZW0102 to ZW0201 platform.**
Written By:	**JFR;SSE**
Date:	**2006-10-23**
Reviewed By:	**LLP**
Restrictions:	**None**

Approved by:

Date	CET	Initials	Name	Justification
2006-10-23	15:07:05	NTJ	Niels Thybo Johansen	

CONFIDENTIAL

REVISION RECORD

Doc. Rev	Date	By	Pages affected	Brief description of changes
1	20050114	SSE JFR	ALL	Minor changes
1.01	20050411	SSE	ALL	Made changes according to NTJ notes.
2	20060105	MVO	All	New 1st page/header/footer contents. New Doc No
3	20060905	JFR	Front page	Change title
4	20061023	JFR	Section 3.8	RF_MODE and IDLE_MODE removed

Table of Contents

1 ABBREVIATIONS

Abbreviation	Explanation
ADC	Analog to Digital Converter
API	Application Programming Interface.
ISR	Interrupt Service Routine
PWM	Pulse Width Modulation
RTC	Real Time Clock
UART	Universal Asynchronous Receiver/Transmitter
WUT	Wake Up Timer

2 INTRODUCTION

2.1 Purpose

The purpose of this document is to give guidelines for the Z-Wave application developer, when porting software applications from ZW0102 to ZW0201 platform.

2.2 Audience and prerequisites

The audience of this document is Z-Wave partners and Zensys A/S.

3 ZW0201 PORTING ISSUES

3.1 System Clock

The system clock (**fsystem)** in ZW0201 is 16Mhz in opposite to 7.3769MHz in ZW0102.

The 8051 timers 0 and 1 runs on half the system clock rate.

3.2 ZW0201 UART

The ZW0201 UART supports the following baud rates: 9600kbps, 38400kbps and 115200kbps. The ZW0201 UART API is extended with the following API functions:

UART_ClearRx	Clear the UART byte received flag. Useful when using serial interrupt service routine.
UART_ClearTx	Clear the UART byte sent flag. Useful when using serial interrupt service routine.
UART_Read	Read the UART receive register. Useful when using serial interrupt service routine.
UART_Write	Write a byte to the UART send register. Useful when using serial interrupt service routine.
UART_Enable	Enable the UART controller.
UART_Disable	Disable the UART controller.

When Porting from ZW0102 to ZW0201, the developer only need to change the baud rates used.

3.3 ZW0201 Timer

The PWM API has the same functions as the ZW0102, except for the new API function ZW_PWMSetup parameter bit mask. The bit mask for the ZW_PWMSetup parameter is as followed:

Bit 0	0 = Timer is inactive and counter is cleared. 1 = Timer is active and counter is enabled.
Bit 1	0 = TIMER mode 1 = PWM mode
Bit 2	0 = Timer counter runs on fsystem/4. 1 = Timer counter runs on fsystem/512.
Bit 3	0 = Timer stops upon overflow. 1 = Timer reloads its counter registers upon overflow.
Bit 4-7	Don't care

3.4 ZW0201 ADC

The Z-Wave ADC API for ZW0201 is changed as follow:

1. The ADC_Init Function has 4 parameters. In ZW0201 dupports the selection of the mode, the upper reference voltage, lower reference voltage and the input pin

2. ZW0201's ADC has 4 input pins. The 4 pins are multiplexed; only one pin is active at one time.

3. ZW0201's ADC has 2 modes. Single conversion mode and continuous mode, continuous mode is only supported in 8 bit mode. The ADC_MULTI_STP_MODE and ADC_MULTI_RST_MODE are not supported in ZW0201.

4. **ADC_On** Is obsolete in ZW0201.The ADC_Off function is used to power down the ADC. The ADC_Init function must be used to power up the ADC.

5. **ADC_SetThresMode** Is used to set the threshold mode. In ZW0201 ADC can trigger when input below/equal or above/equal threshold value.

6. **ADC_SetResolution** Is used to set the ADC resolution. The ZW0201 ADC can either work with 8-bit or 12-bit resolution.

7. **ADC_SetAZPL** Is used to set the auto zero period length. See [1] for more details.

8. **ADC_Buf** is used to enable a buffer between the analog input and the A/D converter. See [1] for more details.

9. **ADC_IsFired**. Is used to check if the conversion result crossed the threshold value.

10. **ADC_SetSamplingRate**. Is obsolete, ZW0201 has a fixed sampling rate.

3.5 ZW0201 TRIAC

The API function TRIAC_Init parameters have changed See [1] for more details.

3.6 RTC

ZW0201 don't have internal support for a Real Time Clock. To implement this functionality, the developer can use a system timer for listening devices. For battery devices (not listening) the WUT can be used, see [1] for more details on WUT.

3.7 Non Initialized Variables

A memory area of 16 SRAM bytes starting at address 0x07DD in controllers and 0x07F0 in slaves will not be initialized when the application starts. This allows the application to have SRAM based variables transferring values from the previous state. A typically case could be to preserve data when the ZW0201 node enters WUT or stop mode. The Z-Wave protocol will initialize the remaining part of the SRAM at wake up on the ZW0201.

The NON_ZERO_START_ADDR constant defines the start address. The following example shows how to define a byte and word in the non-initialized variables area:

```
WORD var1 _at_  NON_ZERO_START_ADDR;

BYTE  var2 _at_  NON_ZERO_START_ADDR + 2;
```

3.8 Power Management

Power management in ZW0201 has 3 new API functions. These functions are:

ZW_SetSleepMode: The ZW0201 can be configured to one of the following power down modes:

1. **STOP_MODE**: ZW0201 wake up based on a hardware reset or an external interrupt.
2. **WUT_MODE**: ZW0201 wake up based on WUT time out or an external interrupt. The ZW0201 is reset when it wake up.

ZW_SetWutTimeout: sets the WUT time out value.

ZW_IsWutKicked: When the ZW0201 wake up from WUT mode, the API function determines if the ZW0201 reset was caused by the WUT.

3.9 Watchdog

The ZW0201 contains a watchdog to increase reliability by detecting software and/or hardware anomalies and reset the ZW0201 if any occur. The watchdog timeout is 1.05 seconds and is fixed. The watchdog is disabled in the protocol by default.

To control the watchdog three API functions are provided.

ZW_WatchDogEnable: Enable the watchdog.

ZW_WatchDogDisable: Disable the watchdog.

ZW_WatchDogKick: Restart the watchdog and thereby preventing it from timing out.

4 REFERENCES

[1] Zensys, INS10247, Instruction, Z-Wave ZW0102/ZW0201 Application Programming Guide