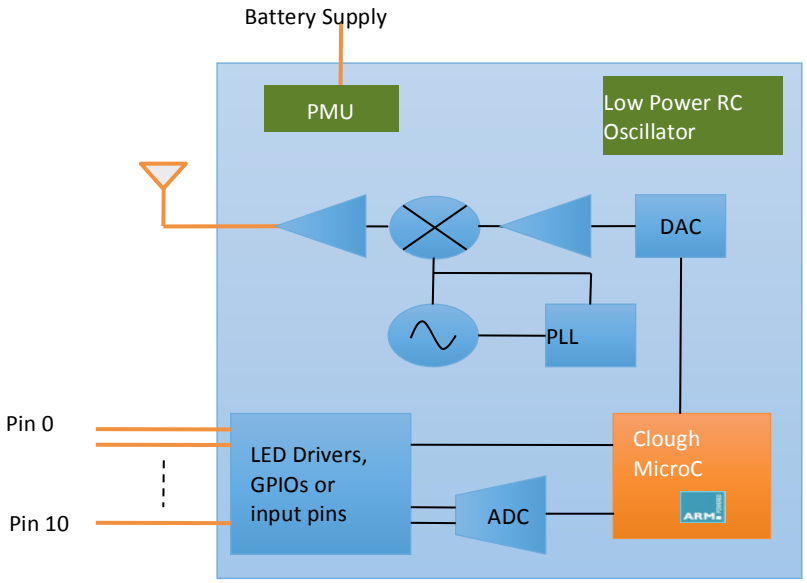
iND80212 “Heimdall Slave”

Heimdall Slave EVKit starter guide

12/4/2015 **Starting Guide**

1.0 TABLE OF CONTENT

2.0 LIST OF FIGURES



4x4 QFN Package

3.0 HEIMDALL SLAVE HARDWARE INTRODUCTION

3.1 HEIMDALL SLAVE IC

Heimdall Slave integrates an ARM Cortex-M0 low cost 32-bit microcontroller containing 160kB of flash program memory and 8kB of SRAM. It implements several general-purpose peripherals. Its main features are:

CPU Architecture:

- ARM Cortex-M0 processor running at 12MHz (Internal RC), 32.768kHz RTC, or 10kHz (Internal auxiliary RC)
- System Tick Timer (SysTick – 24 bits, interruptible)
- Serial Wire Debugger
- Built-in Nested Vectored Interrupt Controller (NVIC)
- Programmable Watch-Dog Timer

Memory:

- 160kByte of Flash Program Memory
- 8kByte of SRAM

Peripherals:

- 310-450MHz Transmitter, supporting selectable ASK/OOK, FSK & DPSK modulation modes
- Runs from a single Lithium cell (may be coin cell)
- Low power mode with internal RC oscillator, Selectable 10kHz / ~250 kHz RC relaxation oscillator
- Integrated PIR sensor interface
- 8-bit ADC with 10 input channels, with selectable input references and input gain block
- 10 General purpose I/O ports, 2 of which has LED current sink capability
- Integrated fractional-N phase locked loop referenced to crystal oscillator
- Red and Blue LED Drivers, with Charge Pump available for Blue LED
- Power Management, Low Battery Detection
- Brown out Reset
- Temperature Sensor
- Wake-up Timer

Package:

- 4x4, 20 pin QFN package

Application:

- Home Automation, Alarm detection, PIR sensor based-system
- Heimdall provides GPIO pins, hardware protocol interfaces, and memory for small applications. Heimdall Slave is designed to be an attractive solution for designers are looking for a basic MCU with 32-bit performance, which has comparable pricing to 8-bit MCUs.
- Figure 1 shows the device block diagram. For more information, please check the Heimdall Slave Datasheet [1]

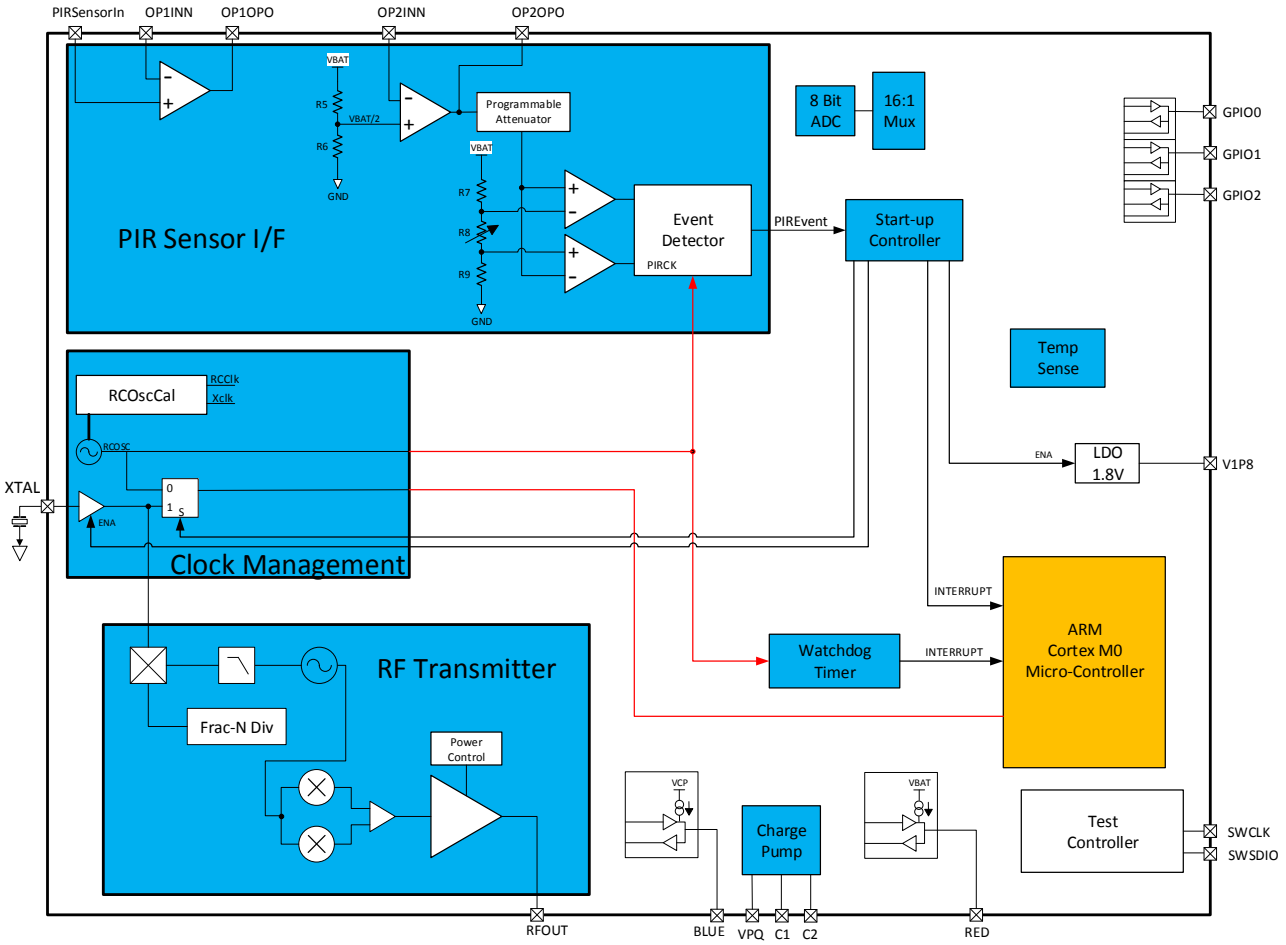


Figure 1 Heimdall Slave block diagram

Indie offers support with software demo programs and the Heimdall Slave evaluation kit described below.

3.2 HEIMDALL SLAVE EVKIT

Heimdall Slave can be evaluated using a small dedicated PCB with PIR sensor and transmitter output. The provided board can be either powered through 3.3V power supply, or independently through single battery. The EVKit schematic is also provided below the boards pictures.

3.2.1 Heimdall Slave EVKit hardware

Figure 2 Heimdall Slave EVKit board

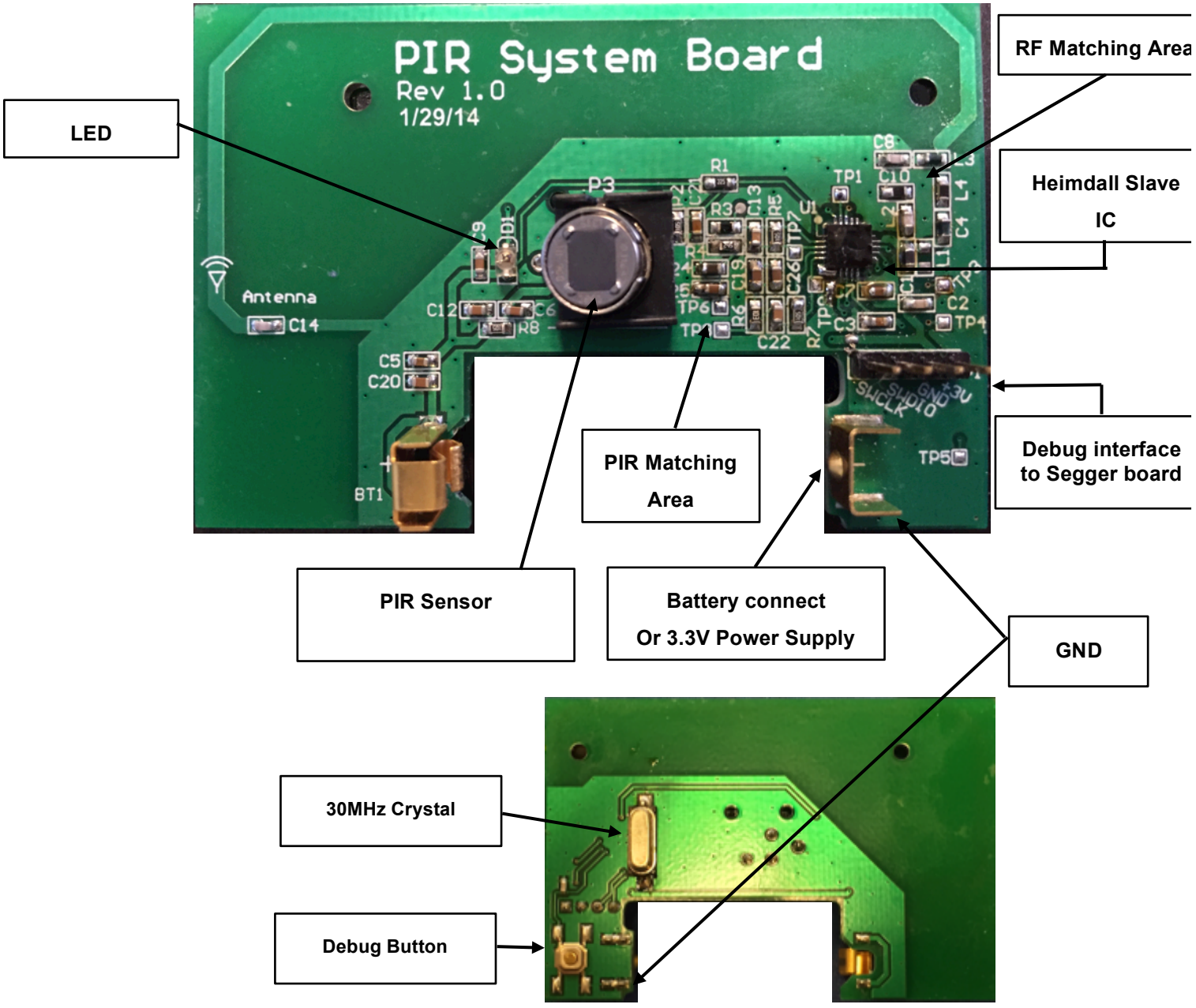
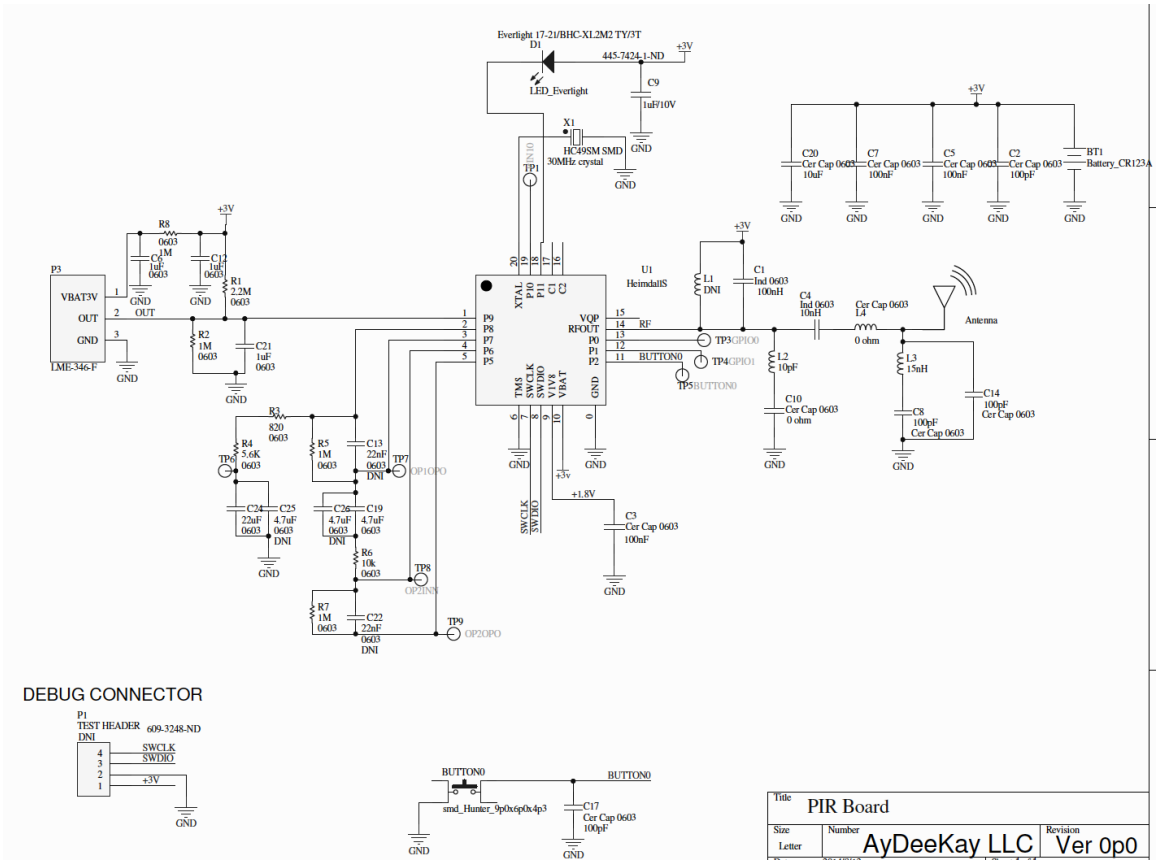


Figure 3 Heimdall Slave EVKit Schematic



3.2.2 Heimdall Slave EVKit software and debugging tools

The current setting, software and demonstration environment are running with IAR system tool chain.

https://www.iar.com

All demo provided by indie can be run with the IAR Size-limited, Kickstart, evaluation license:

- Code size limited license without any time limitation but, no MISRA C support, no power debug functionality, source code for runtime libraries is not included.

https://www.iar.com/iar-embedded-workbench/downloads/

In order to use, evaluate and program Heimdall Slave on the provided board, the standard Segger black box for Arm Cortex M is required:

https://www.segger.com/j-trace-for-cortex-m.html

The Segger box is not provided by default with the Heimdall Slave EVKit, and needs to be purchased separately.

The figure below show the connection between the Segger debugging box and the Heimdall slave EVKit

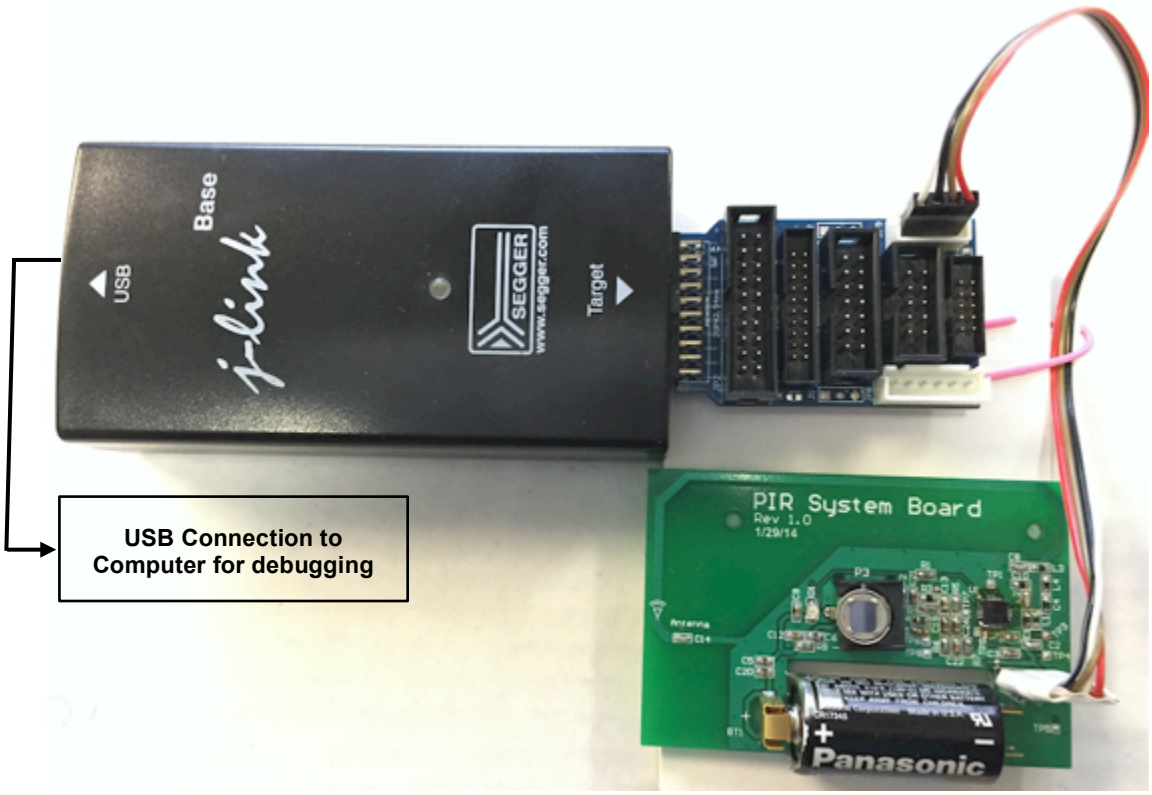
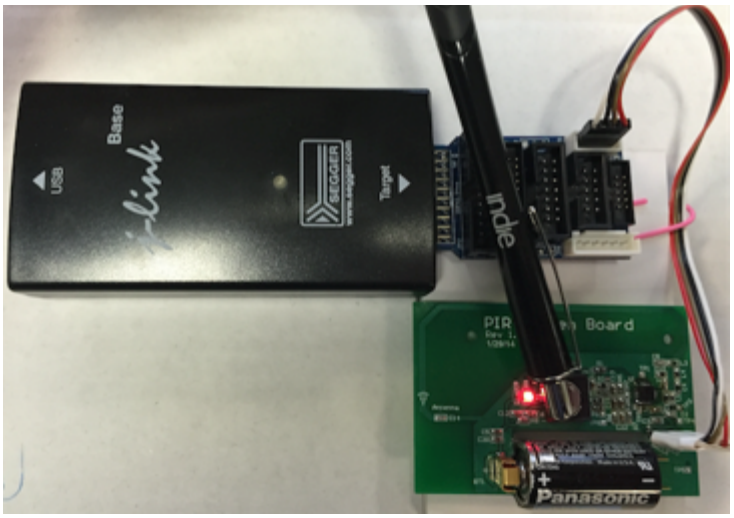


Figure 4 Heimdall Slave and Segger Black box connection

The demo program for PIR is already included and if a battery is inserted, the on-board LED will react to PIR activation. Note that the PIR on the Heimdall Slave EVKIt is set not to be un-sensitive and LED will turn on if sensor is nearly touched



3.2.3 Heimdall Slave EVKit Features

- Heimdall Slave 32-bit Cortex M0

Indie Semiconductor's Heimdall Slave 32-bit ARM Cortex M0. There is a 30MHz crystal for Radio TX function.

- Power Management

Heimdall Slave mini EVKit needs to be powered either through external regulated 3.3V supply (recommended for demo and development) or with a battery.

- Reset

There is no dedicated HW RESET. Device is reset at power-on.

- User Switch

Heimdall Slave mini EVKit has a switch that is connected to port P2. P2 is capable of detecting pin state change to generate interrupt. This can be user programmed for his/her own application. Heimdall Slave has an internal pull-up resistor. Therefore an external resistor is not needed.

- LED indicator

Heimdall Slave mini EVKit has an on-board red color LED connected to Port P11, which has an integrated current-mirror LED driver. This can be user programmed as required for the application.

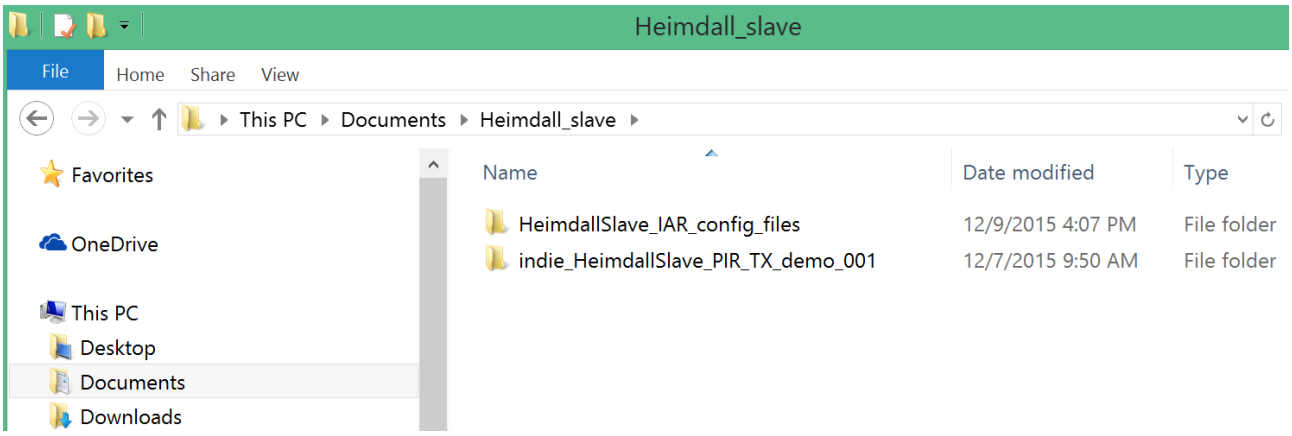
- PIR Sensor

The Heimdall Slave mini EVKit implements an PIR LME-346-F sensor connected to P9

4.0 LAUNCHING DEMO CODE (IAR): STARTING AND CONFIGURATION

- The Heimdall Slave mini EVKit demo software works with IAR open (free) version. The free version allows the creation of downloadable code limited to 16kBytes. All demos can be individually selected and the compiled code fits within the 16kB limit. If more complex programs are compiled it may be necessary to license the full version of IAR.

1) Copy the indie Heimdall Slave Software Development Kit into your working area:



- The /indie_HeimdallSlave_PIR_TX_demo_00"X"/ directory contains the IAR demo projects files. "X" is the demo version released. Please contact indie semiconductor to check the latest available revision
- The /HeimdallSlave_IAR_config_files contains the required config files to run the projects.
    - Before launching IAR, it is necessary to add the several Heimdall Slave config files within the ARM config directories.
    - There are 7 config files, 2 are devices-specific config, and 5 are to be able to program the device through the flashloader.

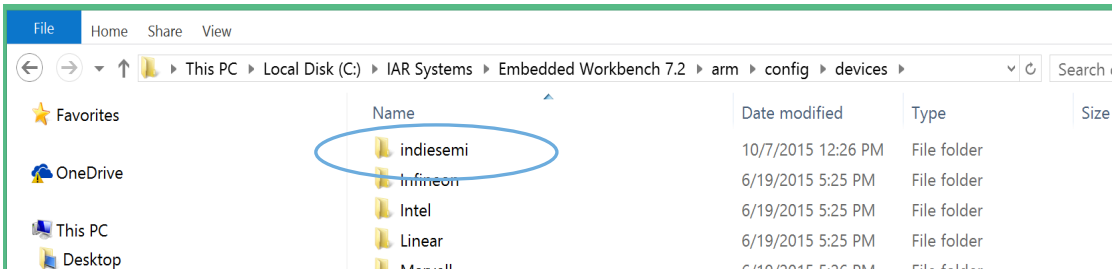
2) Create a "/indiesemi/" sub-directory within the IAR ARM/config/devices area:

The following directory

- **<IAR_PROGRAM_DIR>/arm/config/devices/indiesemi/**

Should then contain the 2 files:

- **HeimdallSlave.i79 and**
- **HeimdallSlave.menu**



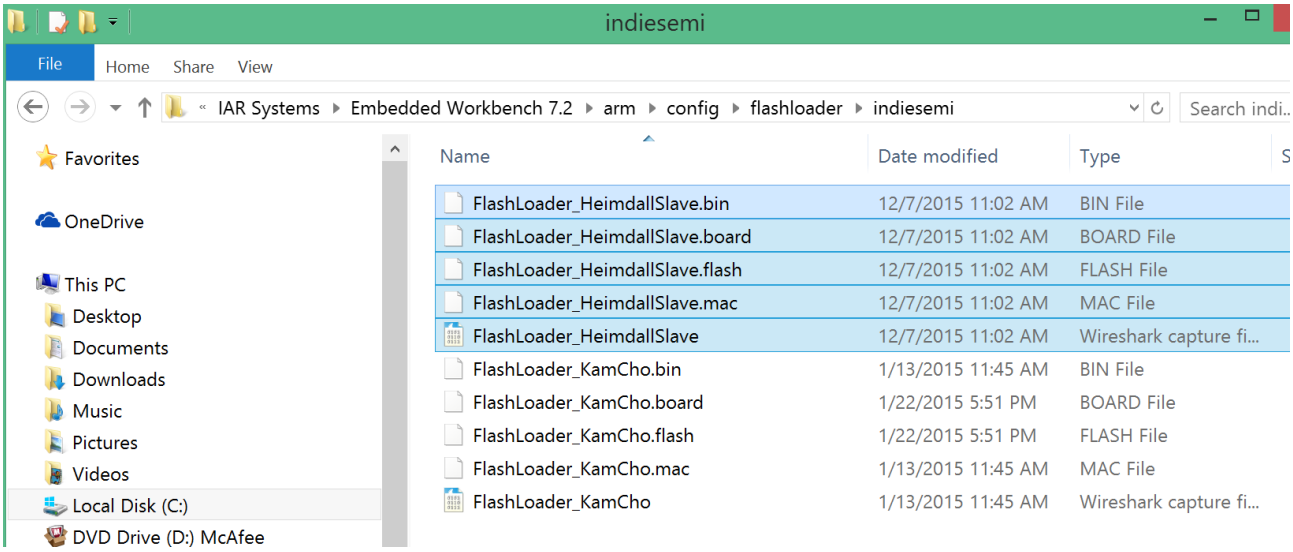
Create an /indiesemi/ sub-directory within the IAR ARM/config/flashloader area

The following directory:

- **<IAR_PROGRAM_DIR>/arm/config/flashloader/indiesemi/**

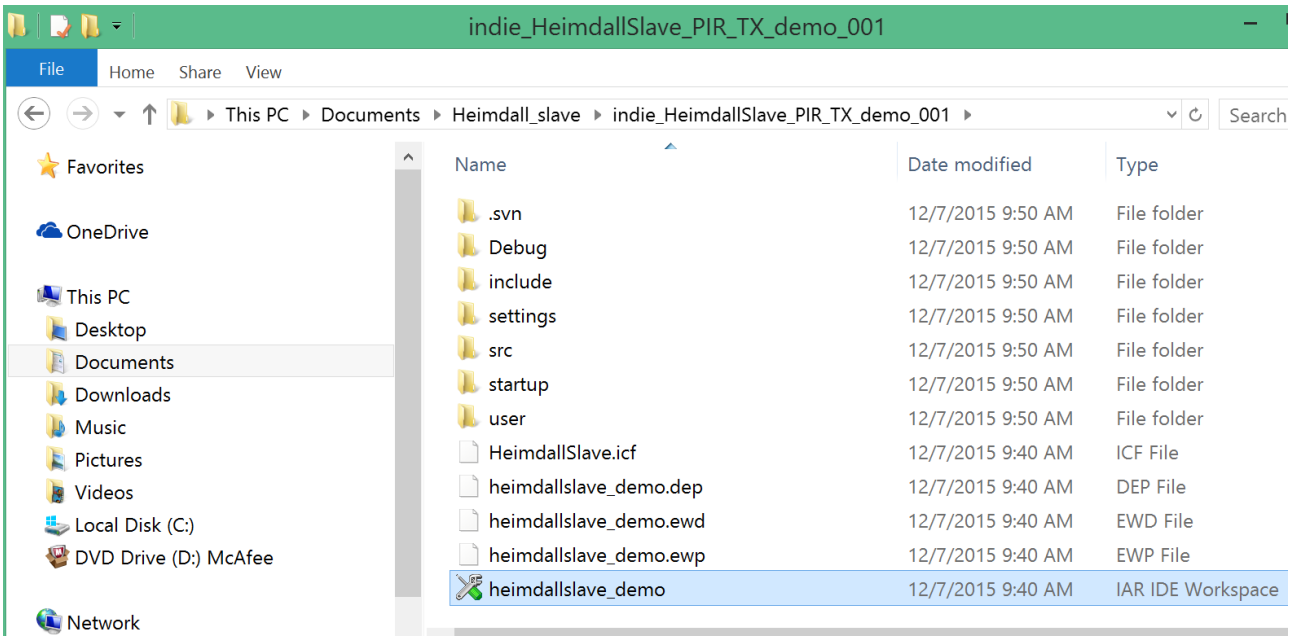
Should then contain the 5 files:

- **FlashLoader_HeimdallSlave.bin**
- **FlashLoader_HeimdallSlave.board**
- **FlashLoader_HeimdallSlave.flash**
- **FlashLoader_HeimdallSlave.mac**
- **FlashLoader_HeimdallSlave.out**

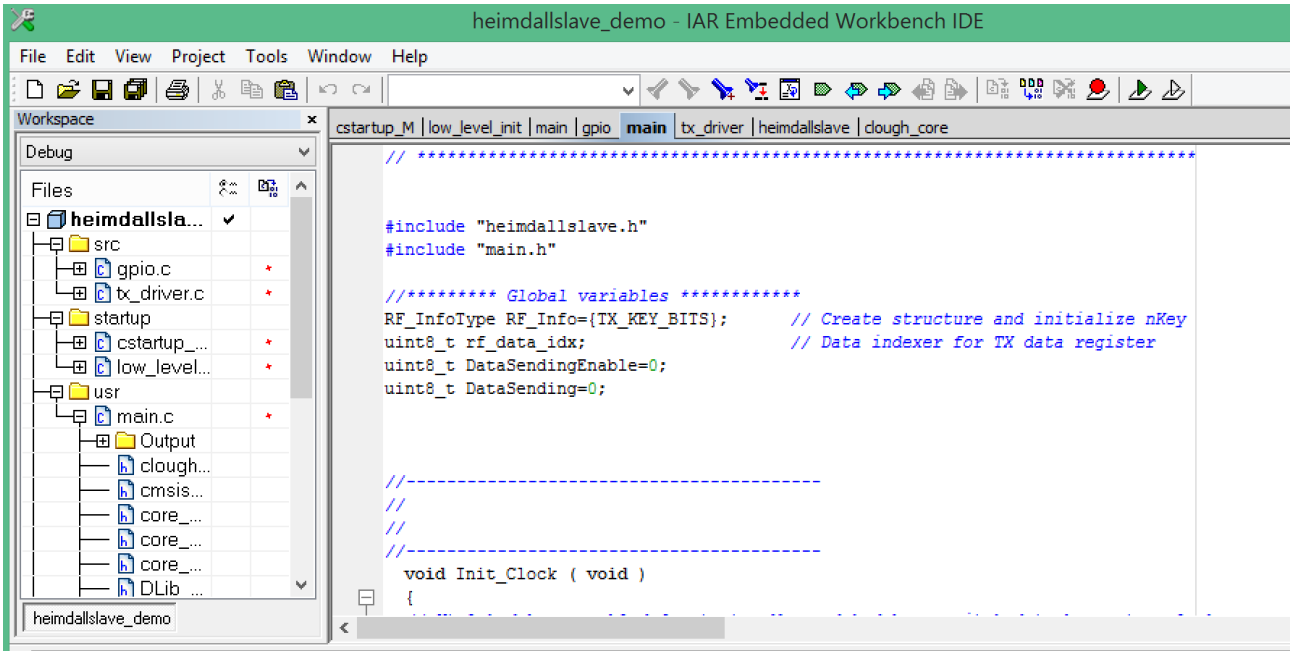


You are now ready to launch IAR and the indie Heimdall Slave demo code:

4) **Go back to the indie Heimdall Slave SDK directory and launch Heimdall SlaveDemo IAR project (IAR IDE Workspace)**



5) The Heimdallslave_demo project will open



5.0 RUNNING THE DEMO

5.1 INTRODUCTION

The purpose of the provided demo is to show the PIR and TX capabilities of the indie Heimdall Slave device.
Once the EVkit is powered either by battery or through external power supply, the software will be active once a PIR event is detected (interrupt) or if one of the port/pin status will change (also as triggering interrupt).
When any of these interrupts occurs, the device will send TX packets and then will go back to sleep.
Combined with a spectrum analyzer, it is easy to see the TX packets sent by Heimdall Slave.

The other capability offered by the EVkit for demo is the button located under the board which, when pressed at power on, makes the software go to “while” loop for ADC measurement.

After the IAR windows has opened, build the program and flash the code into the device.
But in order to prevent Heimdall Slave to get into sleep mode, it is necessary to first press the underneath button to set the program in loop mode. The red led will then turn on and it is possible to flash the code.

5.2 STEP-BY STEP PROCESS

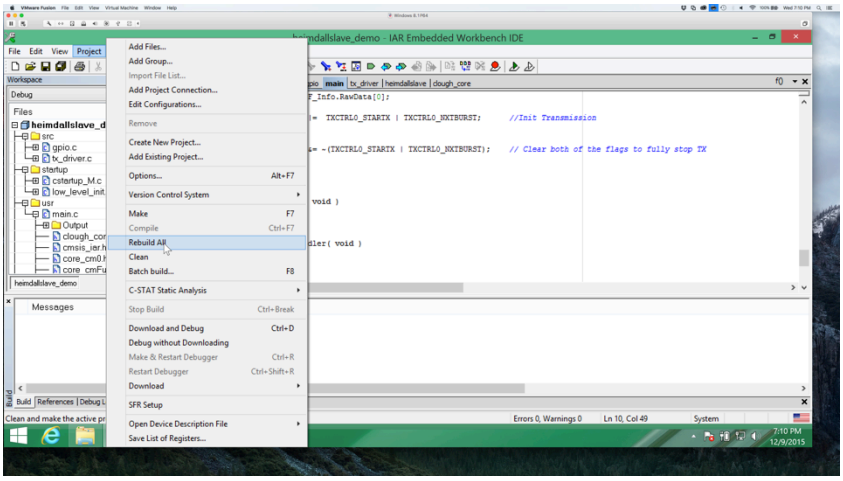
After launching IAR by clicking on heimdallslave_demo.eww file:

- **Reset the EVKit (remove and replace battery or turn off/on power supply)**
- **Set-up board in loop mode:**

Press button and wait for red let to turn on continuously.

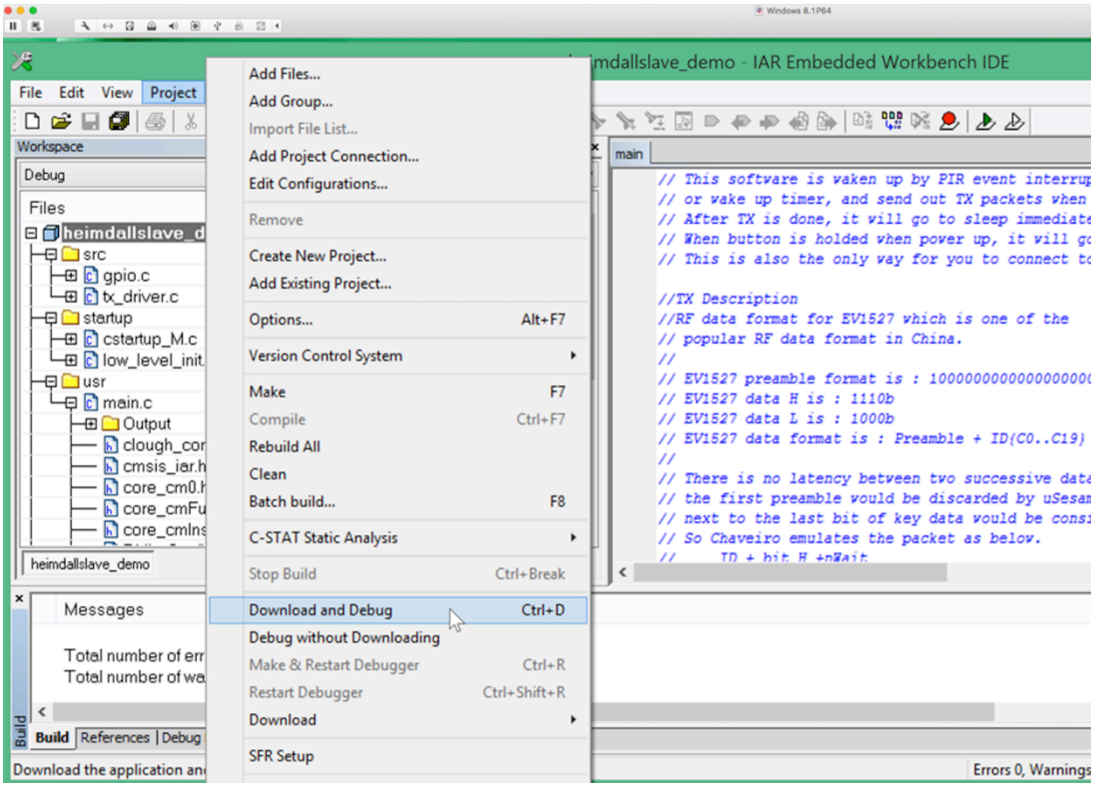
Segger box LED should be green.

- **Build the demo code**

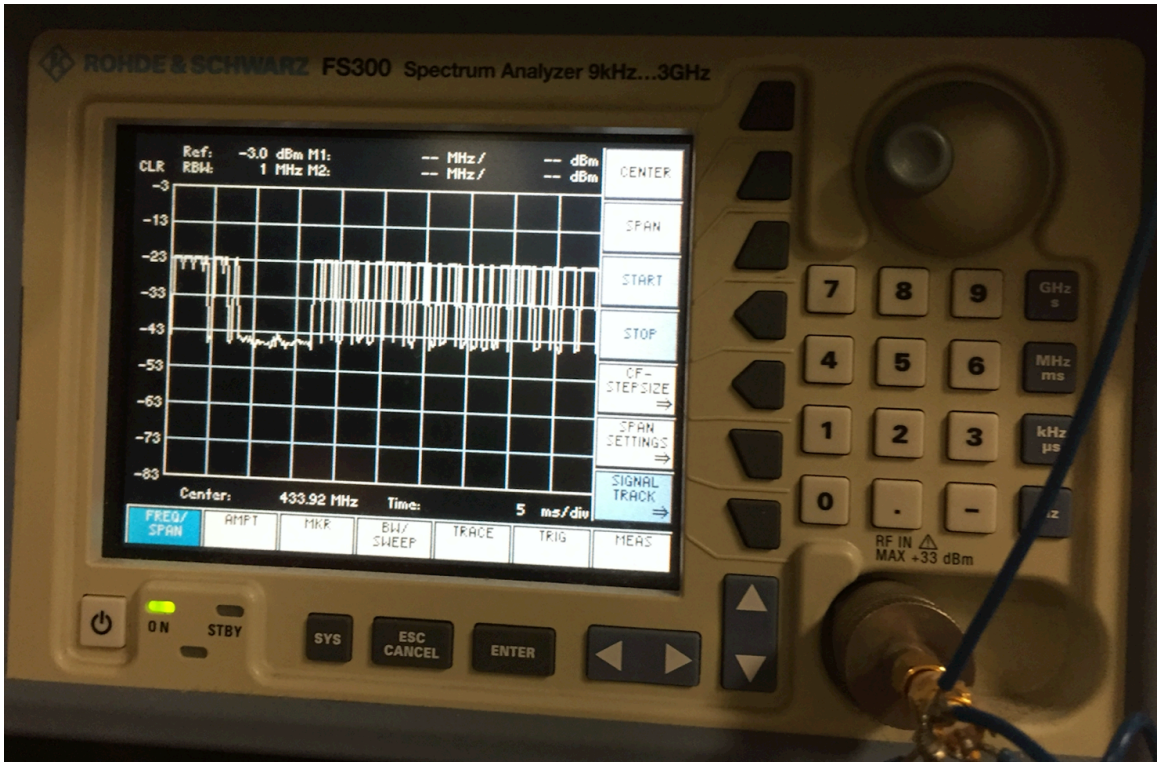


You can ignore the warnings.

- **Download and debug the code into the device:**



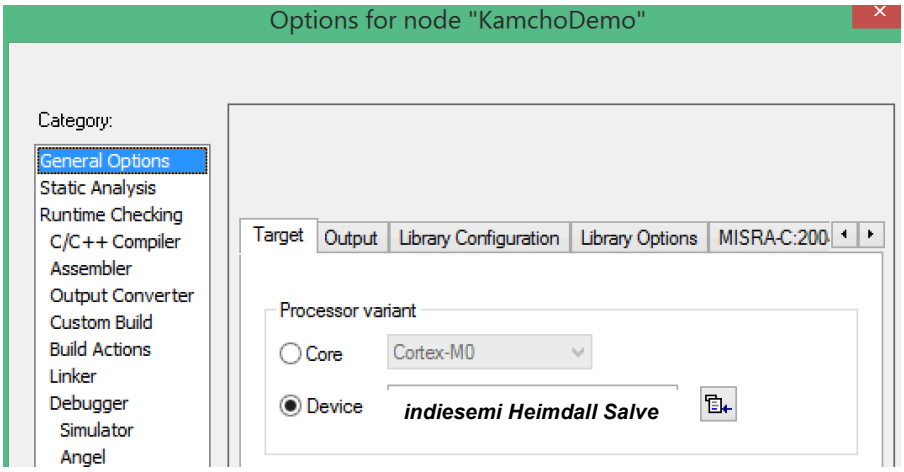
Once the code had been uploaded on the device, disconnect from the Segger box and go through the power cycle. The board can now be used to detect movement around the PIR sensor and transmit a signal each time, which can be detected a spectrum analyzer shows as below:



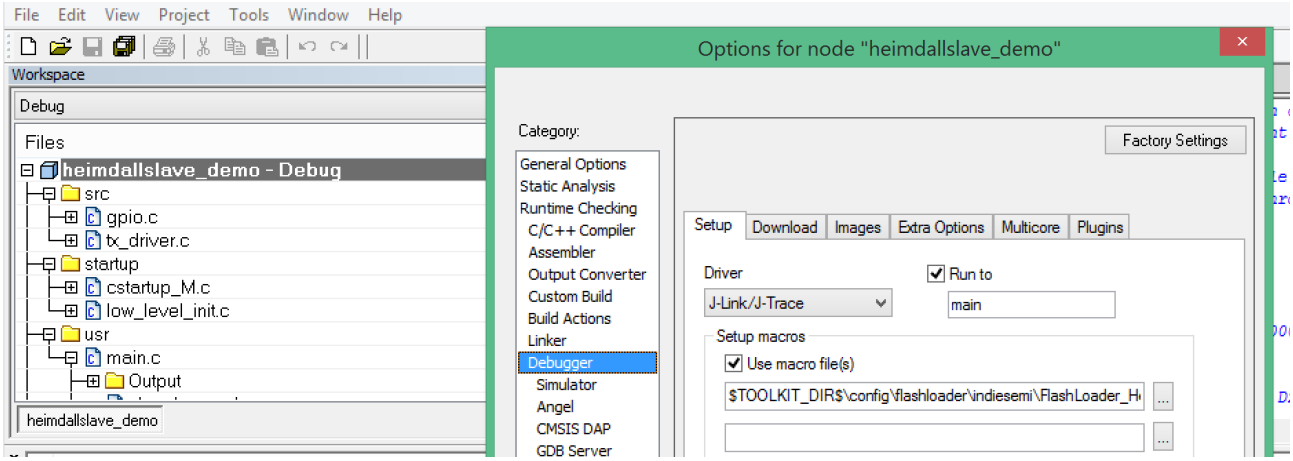
6.0 ANNEX: OVERVIEW OF IAR CONFIG

The IAR environment should be configured for Heimdall Slave demo and Flash programming:

➔ General Options within the IAR Project should indicate the indiesemi Heimdall Slave device



➔ The links to debugger and flash programmer are within the indie Heimdall Slave SDK:



You can now re-Flash the Heimdall Slave evaluation board with the code, and explore its structure.

The Heimdall Slave demo includes all the libraries and .h files needed to check all peripherals as well as all communications.

7.0 REFERENCES

[1] iND80212 data sheet

8.0 REVISION HISTORY

Rev #	Date	Action	By
1.0	12/04/2015	Heimdall Slave mini EVKit starting guide initial draft	CR

9.0 CONTACTS

United States
32 Journey
Aliso Viejo, California 92656, USA
Tel: +1 949-608-0854
sales@indiesemi.com

China
Rm.7B.Flat A, YangGuangHuaYi Building,No.3003,Nanhai Ave
Nanshan District,ShenZhen,China
Tel: +86 18665822385
danielzdz@indiesemi.com

Scotland
5th Floor, The Auction House,
63a George Street
Edinburgh EH2 2JG
Tel: +44 131 718 6378

http://www.indiesemi.com/

Important Notice

indie semiconductor reserves the right to make changes, corrections, enhancements, modifications, and improvements to indie semiconductor products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on indie semiconductor products before placing orders. indie semiconductor products are sold pursuant to indie semiconductor's terms and conditions of sale in place at the time of order acknowledgement. Purchasers are solely responsible for the choice, selection, and use of indie semiconductor products and services described herein. indie semiconductor assumes no liability for the choice, selection, application assistance or the design of Purchasers' products.

No license, express or implied, to any intellectual property right is granted by indie semiconductor by this document.

The materials, products and information are provided "as is" without warranty of any kind, whether express, implied, statutory, or otherwise, including fitness for a particular purpose or use, merchantability, performance, quality or non-infringement of any intellectual property right. Indie semiconductor does not warrant the accuracy or completeness of the information, text, graphics or other items contained herein. indie semiconductor shall not be liable for any damages, including but not limited to any special, indirect, incidental, statutory, or consequential damages, including without limitation, lost of revenues or lost profits that may result from the use of the materials or information , whether or not the recipient of material has been advised of the possibility of such damage.

Unless expressly approved in writing by two authorized indie semiconductor representatives, indie semiconductor products are not designed, intended, warranted, or authorized for use as components in military, space, or aircraft, in systems intended to support or sustain life, or for any other application in which the failure or malfunction of the indie semiconductor product may result in personal injury, death, or severe property or environmental damage.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.