



CH-101 Example Driver Hands On

CH-101 Example Driver Hands On

CH-101 Example Driver Hands On

INTRODUCTION

This exercise demonstrates how to build and run a simple ultrasonic sensing application using the Chirp CH-101 example driver. The application runs on the SmartSonic evaluation board, which uses an Atmel SAMG55 microcontroller. The application can run with either a single sensor or multiple sensors connected to the board.

REQUIRED EQUIPMENT

- SmartSonic evaluation board
- CH-101 sensor daughter board
- Two Micro-USB cables
- Internet connection (if downloading and installing files)

REQUIRED SOFTWARE PACKAGES

- Chirp CH-101 driver example – Atmel Studio 7 project files
- SmartSonic_ExampleDriver.zip
- [Atmel Studio 7](#)
- Terminal emulator of your choice (for example PuTTY or TeraTerm)

CH-101 Example Driver Hands On

TABLE OF CONTENTS

INTRODUCTION	1
REQUIRED SOFTWARE PACKAGES	2
1 INSTALLATION / PREPARATION	4
2 BUILDING THE EXAMPLE APPLICATION AND DRIVER	5
3 PROGRAMMING THE SMARTSONIC BOARD	6
4 RUNNING THE EXAMPLE APPLICATION	9
5 USING ALTERNATE SENSOR FIRMWARE IMAGES	10
6 REVISION HISTORY	11

LIST OF FIGURES

Figure 1. SmartSonic with CH-101 Daughter Card	4
Figure 2. Device Programming Screen	6
Figure 3. Programming Clock Frequency	7
Figure 4. Device Signature and Target Voltage	7
Figure 5. Programming Hex File	8
Figure 6. Successful Programming	8
Figure 7. Example Application Output	9

CH-101 Example Driver Hands On

1 INSTALLATION / PREPARATION

- Download and install Atmel Studio 7 IDE.
- Download and install the CH-101 driver example.
- Install terminal emulator.
- Connect the CH-101 sensor to the DVB board with flex cable(s).
- Connect the SmartSonic board to a Windows PC with the two USB cables.

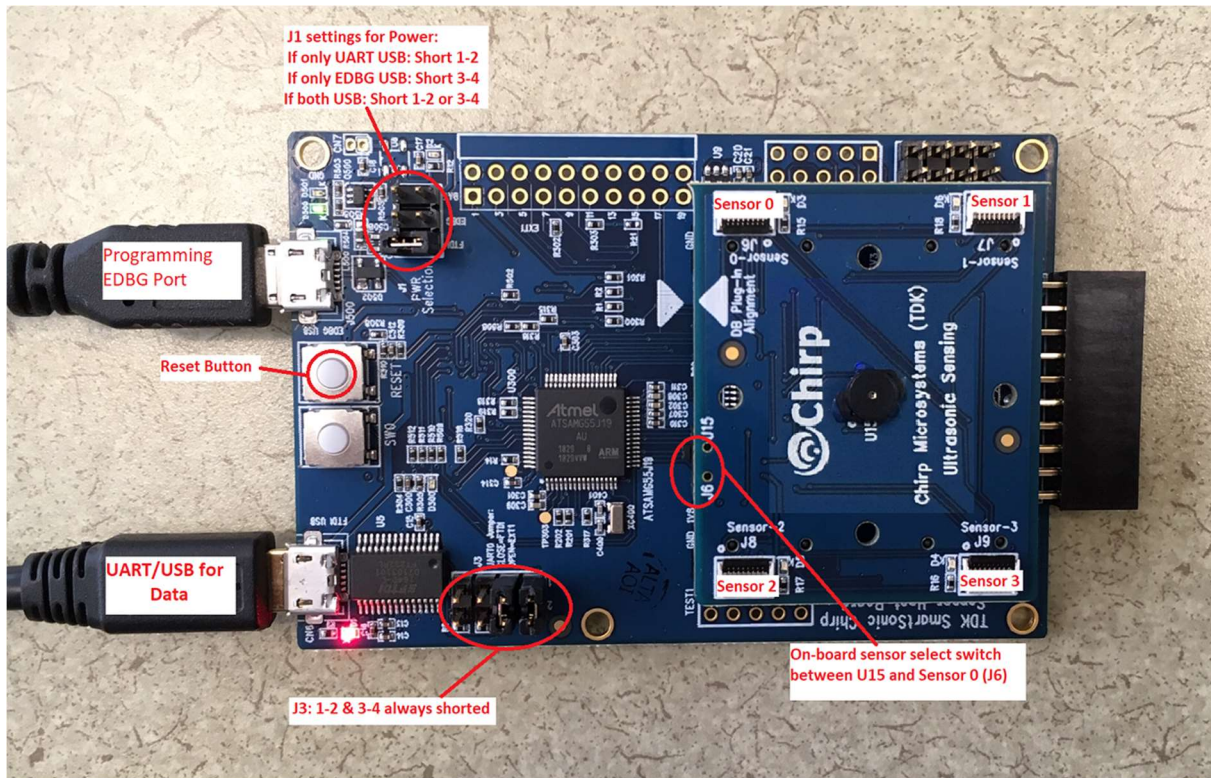


Figure 1. SmartSonic with CH-101 Daughter Card

- Open Windows Device Manager look in the Ports (COM & LPT) list and identify the COM port numbers assigned to the SmartSonic board. There will be two ports associated with the SmartSonic board: EDGB and USB Serial Port. You will need to specify the USB serial port number when using a terminal emulator to display output.

CH-101 Example Driver Hands On

2 BUILDING THE EXAMPLE APPLICATION AND DRIVER

- Open Atmel Studio 7
- Open the driver example project:
 - Open **File** menu
 - If you are using a new System Workbench installation:
 - Select **File > Open > Project/Solution...** > Select the **SmartSonic_ExampleDriver.atsln** file in the project directory.
 - Click **Open**. The program should locate the project files and display the name of the project.
- Build the project:
 - Select **Build > Rebuild Solution**

The project should build successfully (with some warnings). The default build configuration is “Debug” so the build output files will be placed in the Debug sub-directory.

- Warning: **Do NOT** select “**Clean Solution**” or otherwise clean the build output files. Doing so will remove some build definition files, and the next build(s) will fail. If you have this problem, run “**Build Project**” several (3 or 4) times until the build succeeds – the repeated builds will ultimately reconstruct the missing files.

CH-101 Example Driver Hands On

3 PROGRAMMING THE SMARTSONIC BOARD

- Connect the SmartSonic board to a Windows PC with the two USB cables
- Select **Tools > Device Programming**.
- The Device Programming screen will appear:

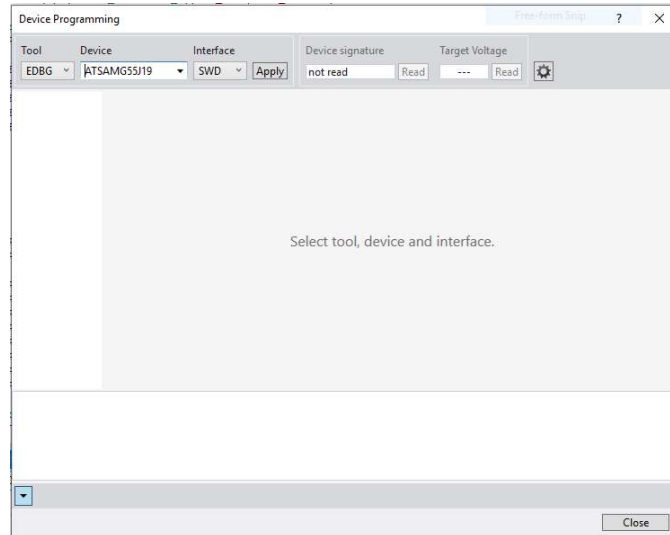


Figure 2. Device Programming Screen

- Verify that the tool is **EDBG**, device is **ATSAMG55J19**, and interface is **SWD**.
- Select **Apply**.
 - **Note:** Atmel Studio 7 may require you to update the EDBG debug interface firmware on the SmartSonic board before continuing. Follow the on-screen instructions to update the EDBG firmware.

- The Device Programming screen will prompt to set the programming clock frequency:

CH-101 Example Driver Hands On

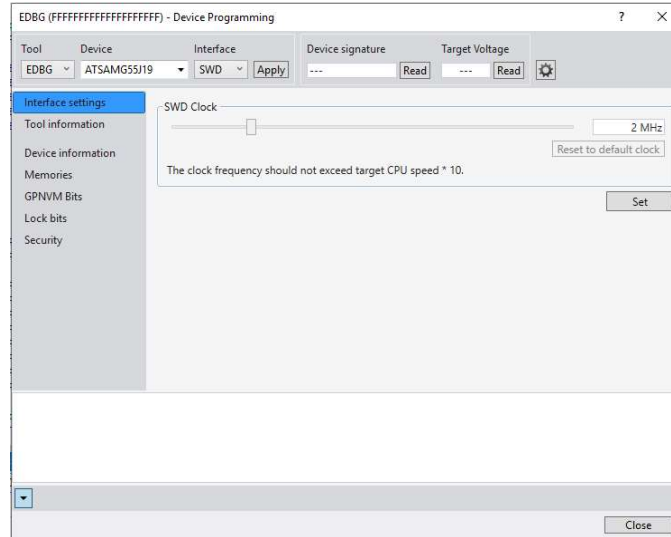


Figure 3. Programming Clock Frequency

- Leave the clock frequency at the default and select **Set**.
- Select **Read** near the **Device signature** field.
- The Device programming menu should look as follows:

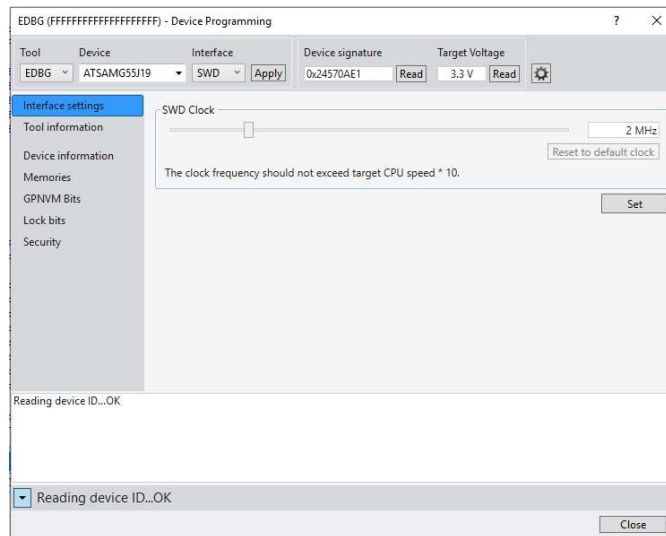


Figure 4. Device Signature and Target Voltage

- Select **Memories** on the Device Programming menu.
- The Device Programming menu will prompt for the name of the hex file to program:

CH-101 Example Driver Hands On

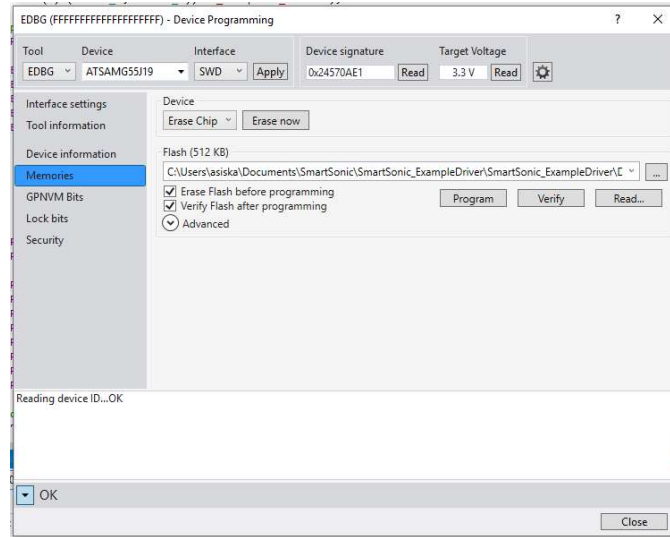


Figure 5. Programming Hex File

- From the Device Programming screen scroll to the project’s debug directory and select the **SmartSonic_ExampleDriver.hex** file.
- Select **Program**. Your SmartSonic board is successfully programmed when the Device Programming screen displays the following on the bottom left:

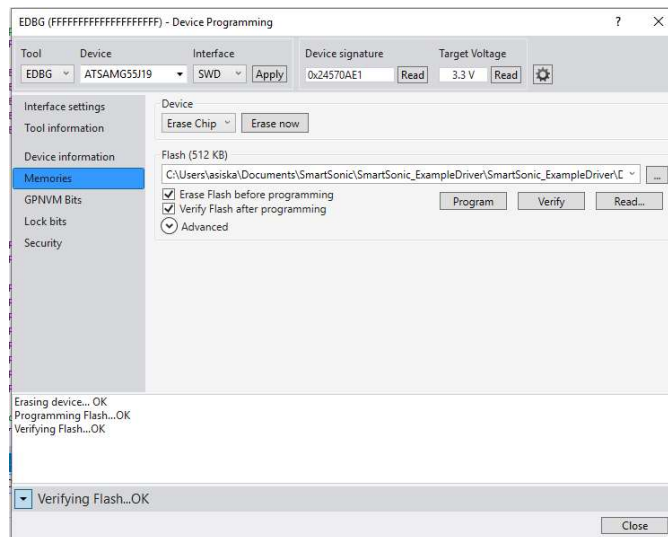


Figure 6. Successful Programming

CH-101 Example Driver Hands On

4 RUNNING THE EXAMPLE APPLICATION

- Start the terminal emulator program and open/configure the COM port assigned to the SmartSonic board:
 - **921600 baud** (non-standard but should work with most terminal emulators)
 - **8 bits data, no parity, 1 stop bit**
 - **New-line sequence = Line Feed only (no carriage return)**
- Reset the SmartSonic board using the board's reset button (next to the Programming EDGB connector).
- Status messages from the application will appear on the terminal output, followed by summary data from the sensor initialization (device frequency, etc.).
- Range measurement data from the sensor device(s) will be output in a continuous loop. Note that distance measurements are expressed in **millimeters**.

```

COM9 - Tera Term VT
File Edit Setup Control Window Help
Set Resolution to Normal 12-bit
CHIRP dongle-0 0x45 register 0x00 and 0x01 = 0a, 02
CHIRP dongle-1 0x45 register 0x00 and 0x01 = 00, 00
CHIRP dongle-2 0x45 register 0x00 and 0x01 = 00, 00
CHIRP dongle-3 0x45 register 0x00 and 0x01 = 00, 00
Chirp Sensor Idd: 155 uA
Chirp Microsystems CH-201 Driver Example
Compile Time: May 6 2019 08:19:57
Version: 0.5
CH-201 firmware: lr16_multithresh_03.hex
Starting timers... timer init OK..
Programming CH-201...chirp programming OK..

Sensor: 0 PT: 2880 Freq: 73477 BW: 0 SF: 0

Port 0: threshold 0: level = 5000 length = 26
Port 0: threshold 1: level = 2000 length = 13
Port 0: threshold 2: level = 800 length = 17
Port 0: threshold 3: level = 400 length = 23
Port 0: threshold 4: level = 250 length = 10
Port 0: threshold 5: level = 175 length = 0

Port 0: Range: 252.6 Amplitude: 1382 134 IQ samples copied
Port 0: Range: 486.2 Amplitude: 3214 134 IQ samples copied
Port 0: Range: 402.2 Amplitude: 4267 134 IQ samples copied
Port 0: Range: 419.4 Amplitude: 2845 134 IQ samples copied
Port 0: Range: 429.2 Amplitude: 3685 134 IQ samples copied
Port 0: Range: 422.9 Amplitude: 3640 134 IQ samples copied
Port 0: Range: 478.2 Amplitude: 4781 134 IQ samples copied
Port 0: Range: 473.0 Amplitude: 2766 134 IQ samples copied
Port 0: Range: 554.4 Amplitude: 2485 134 IQ samples copied
Port 0: Range: 467.5 Amplitude: 2447 134 IQ samples copied
Port 0: Range: 470.6 Amplitude: 4531 134 IQ samples copied
Port 0: Range: 458.1 Amplitude: 4002 134 IQ samples copied
Port 0: Range: 413.2 Amplitude: 2688 134 IQ samples copied
Port 0: Range: 412.9 Amplitude: 3564 134 IQ samples copied
Port 0: Range: 423.9 Amplitude: 2129 134 IQ samples copied
Port 0: Range: 432.6 Amplitude: 2972 134 IQ samples copied
Port 0: Range: 483.6 Amplitude: 3526 134 IQ samples copied
Port 0: Range: 475.6 Amplitude: 3197 134 IQ samples copied
Port 0: Range: 482.9 Amplitude: 3625 134 IQ samples copied
Port 0: Range: 485.8 Amplitude: 3832 134 IQ samples copied
Port 0: Range: 422.9 Amplitude: 3325 134 IQ samples copied
Port 0: Range: 425.7 Amplitude: 4364 134 IQ samples copied
Port 0: Range: 428.8 Amplitude: 4073 134 IQ samples copied
Port 0: Range: 433.4 Amplitude: 3281 134 IQ samples copied
Port 0: Range: 434.4 Amplitude: 2716 134 IQ samples copied
Port 0: Range: 439.1 Amplitude: 2762 134 IQ samples copied
Port 0: Range: 436.8 Amplitude: 2673 134 IQ samples copied
Port 0: Range: 433.9 Amplitude: 4038 134 IQ samples copied
Port 0: Range: 430.9 Amplitude: 4712 134 IQ samples copied
Port 0: Range: 433.6 Amplitude: 5394 134 IQ samples copied
    
```

Figure 7. Example Application Output

5 USING ALTERNATE SENSOR FIRMWARE IMAGES

By default, the example application uses standard Chirp “GPR” (General Purpose Rangefinder) sensor firmware. This firmware provides good performance over various distances under most conditions. Normally, the standard firmware is recommended, and no changes are required.

For special applications using CH-101 sensors, two options are available. Either or both may be selected by defining symbols in the **inc/main.h** header file.

To use a special firmware version that is optimized for short-range performance, define **USE_SHORT_RANGE** in **main.h**. This special firmware provides more measurement resolution at close distances. However, the maximum range for the sensor is reduced significantly (by a factor of 4).

To use a special firmware version that uses less power when the device is idle, define **USE_LOW_POWER** in **main.h** (at or near line 107). This version has a significant disadvantage, however – the performance of the CH-101 sensor is negatively affected by sunlight. Therefore, it is not recommended for most applications.

6 REVISION HISTORY

Revision Date	Revision	Description
08/02/2019	1.0	Initial Release

This information furnished by Chirp Microsystems, Inc. ("Chirp Microsystems") is believed to be accurate and reliable. However, no responsibility is assumed by Chirp Microsystems for its use, or for any infringements of patents or other rights of third parties that may result from its use. Specifications are subject to change without notice. Chirp Microsystems reserves the right to make changes to this product, including its circuits and software, in order to improve its design and/or performance, without prior notice. Chirp Microsystems makes no warranties, neither expressed nor implied, regarding the information and specifications contained in this document. Chirp Microsystems assumes no responsibility for any claims or damages arising from information contained in this document, or from the use of products and services detailed therein. This includes, but is not limited to, claims or damages based on the infringement of patents, copyrights, mask work and/or other intellectual property rights.

Certain intellectual property owned by Chirp Microsystems and described in this document is patent protected. No license is granted by implication or otherwise under any patent or patent rights of Chirp Microsystems. This publication supersedes and replaces all information previously supplied. Trademarks that are registered trademarks are the property of their respective companies. Chirp Microsystems sensors should not be used or sold in the development, storage, production or utilization of any conventional or mass-destructive weapons or for any other weapons or life threatening applications, as well as in any other life critical applications such as medical equipment, transportation, aerospace and nuclear instruments, undersea equipment, power plant equipment, disaster prevention and crime prevention equipment.

©2016—2017 Chirp Microsystems. All rights reserved. Chirp Microsystems and the Chirp Microsystems logo are trademarks of Chirp Microsystems, Inc. The TDK logo is a trademark of TDK Corporation. Other company and product names may be trademarks of the respective companies with which they are associated.