APPLICATION NOTE

Atmel AVR057: Internal RC Oscillator Calibration for ATtiny4/5/9/10/20/40

8-bit Atmel Microcontrollers

Features

- Calibration of the internal 8MHz RC Oscillator for Atmel® ATtiny4/5/9/10/20/40 devices
- Calibration implemented with the Atmel AVR918 (Using the Atmel TPI) and the Atmel AVR911 (AVROSP)
- Adjustable RC frequency with ±1% accuracy
- Tune RC oscillator at any operating voltage and temperature
- Tune RC oscillator to any frequency within specification

Introduction

The ATtiny4/5/9/10/20/40 devices feature an internal 8MHz RC oscillator with prescalar and calibration register (OSCCAL). The internal RC oscillator of these devices is factory calibrated for ±10% accuracy at 3V supply voltage and at 25°C. But most of the applications needs more accurate clock as their basic requirement. For such applications AVR® offers a way to calibrate the internal RC oscillator. The internal RC oscillator can be user calibrated to ±1% accuracy by modifying the OSCCAL register.

This application note provides an easy procedure for calibrating the internal 8MHz RC oscillator of the ATtiny4/5/9/10/20/40 devices. It uses the AVR918 (Using Atmel Tiny Programming Interface) application note for programming and calibration with the AVROSP as the software frontend.

Table of Contents

1. The internal RC oscillator

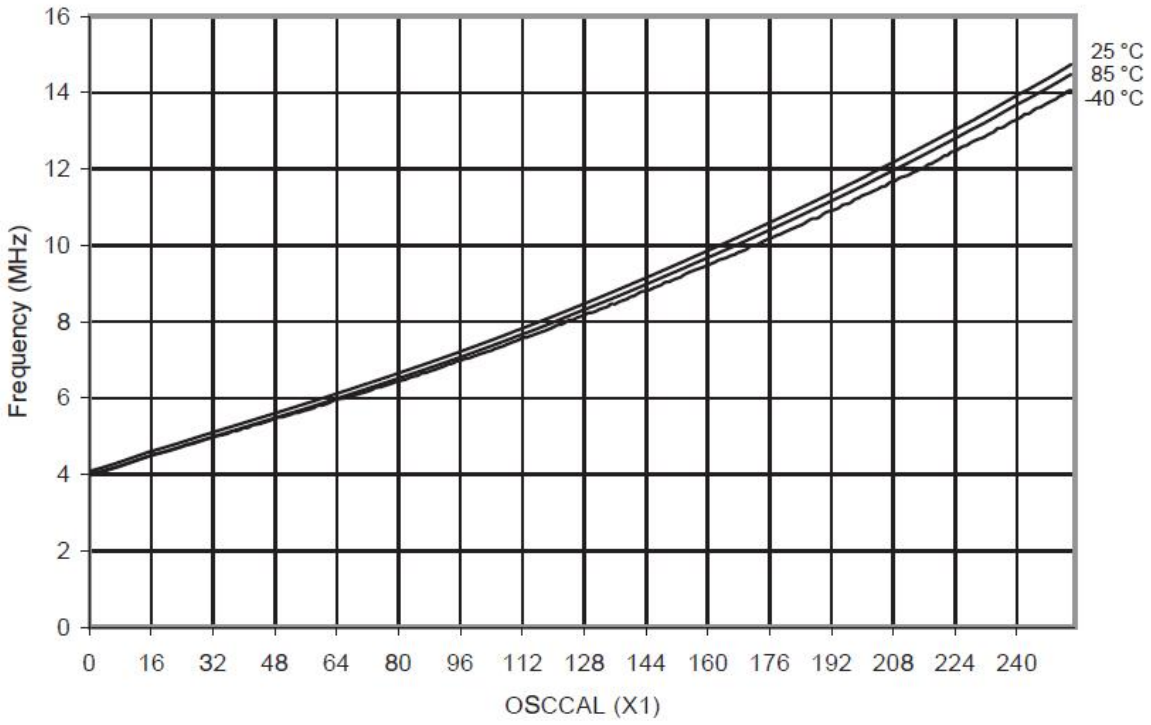
Majority of the AVR MCUs have an internal calibrated RC oscillator as one of the clock sources. This internal RC oscillator runs without connecting any external components to the MCU and can be used as the system clock.

In production the internal RC oscillator of the Atmel ATtiny4/5/9/10/20/40 devices is calibrated at 3V. The accuracy of the factory calibration is within ±10%. If a design's need for accuracy is beyond what can be offered by the standard calibration in factory by Atmel, it is possible to perform a secondary calibration of the RC oscillator. By doing this it is possible to obtain frequency accuracy within ±1%. A secondary calibration can thus be performed to improve or tailor the accuracy or frequency of the oscillator.

1.1 User calibration

The ATtiny4/5/9/10/20/40 devices contain a register called OSCCAL (OSCillator CALibration) register. The frequency of the internal RC oscillator will vary depending on the value loaded into the OSCCAL register. As an example Figure 1-1 shows the typical variation of the internal RC oscillator frequency of the ATtiny10 device with respect to the OSCCAL value at 3V supply voltage.

Figure 1-1. Internal RC oscillator frequency variation with respect to OSCCAL value (for ATtiny10 at V_{CC} = 3V).



From the above plot it can be inferred that the frequency of the internal RC oscillator varies with the OSCCAL value as well as the temperature. Hence for a particular operating environment the internal RC oscillator can be calibrated to ±1% accuracy by varying the OSCCAL register.

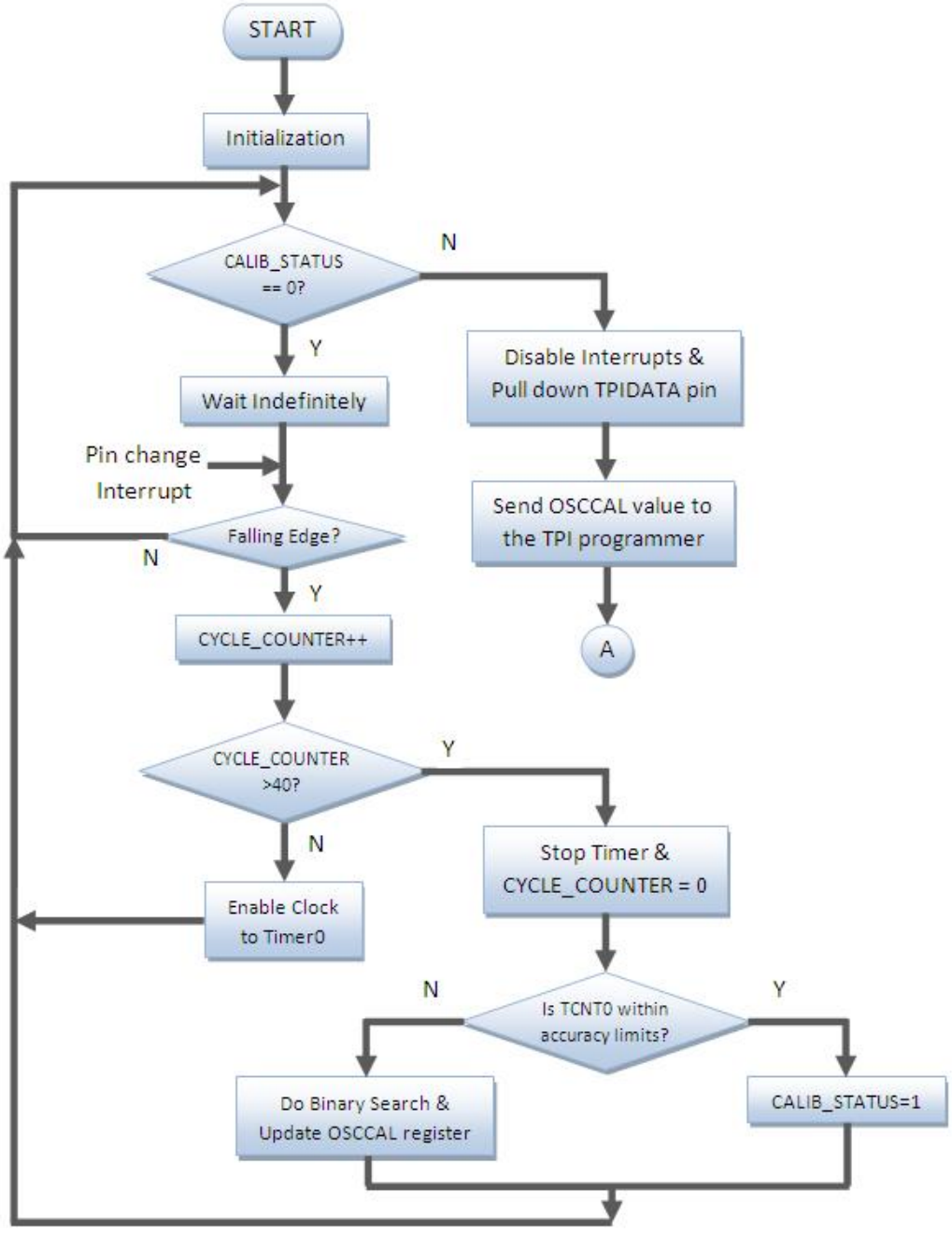
The OSCCAL value corresponding to the factory calibration will be saved into the production signature row of the device. Hence every time after reset the device loads the saved OSCCAL value from production signature row into the OSCCAL register. This makes the device to run the internal RC oscillator at ±10% accuracy by default. The user application can then calibrate the internal RC oscillator by modifying the OSCCAL register value continuously until it reaches the specified accuracy.

2. Calibration protocol

The protocol implemented in this application note will calibrate the internal RC oscillator based on externally applied 32kHz signal. The process of calibration is accomplished with the Atmel AVR918 TPI programmer itself. Detailed explanation of how the TPI programmer is used for calibration is explained in the upcoming section.

This section simply explains how the target device performs the calibration based on the 32kHz signal applied to its TPICLK pin. Figure 2-1 provides the flowchart of the main calibration routine running inside the target MCU.

Figure 2-1. Flowchart for the main calibration routine.



The I/O pin of the TPI programmer which is connected to the TPIDATA pin of the target is configured as input with internal pull-up enabled. The 32kHz signal is applied to the TPICLK pin of the target.

2.2 Workflow

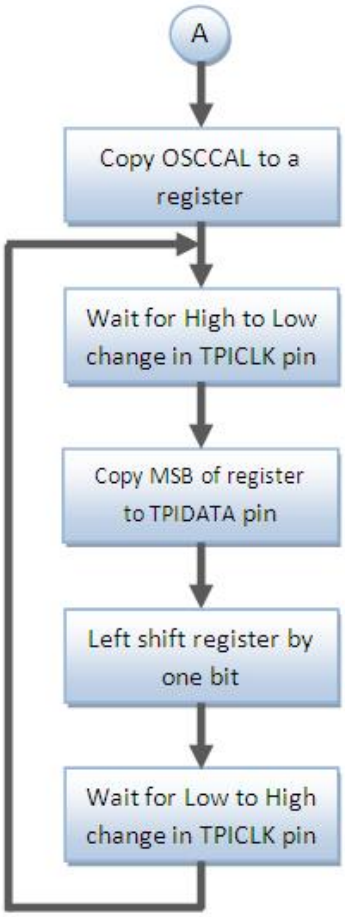
As a brief outline of the process the protocol uses the 16-bit timer of the target device which runs with the I/O clock without any prescaling. For Atmel ATtiny4/5/9/10 devices Timer0 is used and for Atmel ATtiny20/40 Timer1 is used. The protocol first initializes the OSCCAL register with an initial value (typically half the maximum) and starts the timer based on a falling edge of the calibration clock applied to the TPICLK pin. The timer is kept running for 40 calibration clock cycles. After 40 cycles the timer is stopped and the TCNT value is checked whether it is inside the limits of accuracy. If the value is more than the maximum limit the OSCCAL is reduced and if the value is less than the minimum limit the OSCCAL is increased and again the timer is run for next 40 calibration clock cycles. This process is continued until the TCNT value is within the accuracy limits.

Step by step implementation of the calibration code is explained below.

1. The target device starts up with the initialization routine. This routine does the following steps.
    - Allocates registers for program variables and initialize the variables
    - Calculates the frequency limits and timer count limits for specified accuracy
    - Initiates step size to half the maximum OSCCAL value
    - Selects internal RC oscillator as clock source (This setting is default after reset)
    - Configure system clock prescaler to clock divided by 1 (Default configuration is divide by 8)
    - Configure TPIDATA pin as output and TPICLK pin as input with internal pullup
    - Enable pin change interrupt on TPICLK pin
    - Load OSCCAL value with initial step size (127)
    - Set global interrupt enable bit
2. After initialization the main code checks whether the variable CALIB_STATUS is zero. If so then it waits indefinitely until it becomes non-zero. If the variable becomes non-zero it stops the calibration process and starts to send the OSCCAL value bit by bit which will be explained shortly.
3. When the TPICLK pin starts receiving 32kHz signal it triggers the pin change interrupt which has been enabled during initialization. Whenever a pin change interrupt is triggered the target checks whether it is a falling edge. It ignores a rising edge.
4. If falling edge is detected it increments the CYCLE_COUNTER variable and checks whether it has reached the maximum value (40). If not the clock to the timer is enabled and the CPU is returned to main.
5. If CYCLE_COUNTER reaches the limit the TCNT value is checked whether it is within the accuracy limits.
    - If TCNT is more than the maximum limit then the OSCCAL is reduced by half of the step size.
    - If TCNT is less than the minimum limit then the OSCCAL is increased by half of the step size.
    - If TCNT is within the limits then the TPIDATA pin is pulled down to indicate the TPI programmer that the calibration is completed and the interrupts are disabled. Once the calibration is over the CPU executes the routine for transmitting the OSCCAL value to the TPI programmer.
6. If TCNT is not within limits the TCNT register is cleared, CYCLE_COUNTER is cleared and step 4 continues.

The TPI programmer will always monitors for a low level on the TPIDATA pin. If the TPIDATA pin is pulled low it understands that the calibration on the target is completed and stops providing calibration clock to the TPICLK pin. Now the OSCCAL value has to be read back from the target. Since the target (ATtiny4/5/9/10) does not has any communication interface the I/O pin of the TPI programmer connected to the TPICLK pin of the target is toggled to output the OSCCAL value bit by bit on the TPIDATA line. The bits from the TPIDATA pin are accumulated to form the OSCCAL value at the TPI programmer. Figure 2-2 depicts the workflow of this process.

Figure 2-2. Flowchart for the OSCCAL read back.



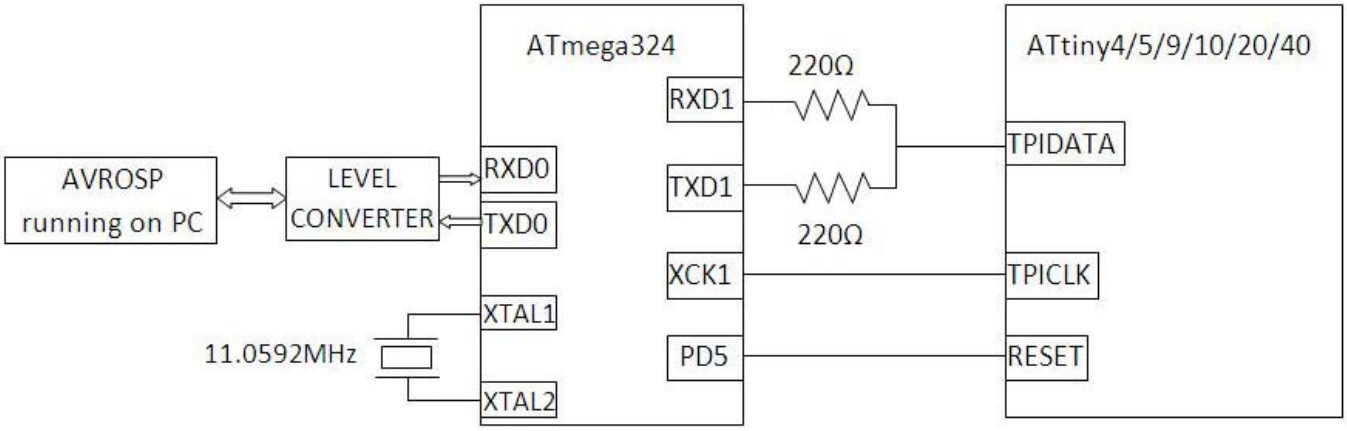
During calibration the TPI programmer acts as normal I/O pin controller controlling signals on the I/O pins of the target device. In other words the target is not under programming mode and the reset line is not under the control of the TPI programmer. Once the calibration is successful and the TPI programmer has the OSCCAL value it holds the reset line and enters into programming mode.

If the calibration fails (that is if the TPIDATA line is not pulled low before the AVROSP timeout) the AVROSP timeout occurs and an error message will be popped up in the command window.

3. Hardware setup

The application note does the programming of the target device as well as calibration of the target internal RC oscillator using Atmel AVR918 as TPI programmer and Atmel AVR911 as software frontend. Figure 3-1 shows the block schematics of the hardware setup.

Figure 3-1. Hardware setup – block schematics.



AVR918 example code uses the Atmel ATmega324 as the TPI programmer since it has two USART modules in it. The ATmega324 is connected with an external 11.0592MHz crystal which acts as the clock source with the CKDIV8 fuse disabled. The USART0 module is operated asynchronously connected to the computer's RS232 port (after proper level conversion) and is used for communicating with the AVROSP running on the computer. The USART1 module is operated synchronously which is used for the TPI programming and for calibration.

The commands for programming the target is same as explained in the AVR918 application note. For calibration the '-Y' command should be used. When a '-Y' command is sent to the TPI programmer through AVROSP the TPI programmer (ATmega324) does the following steps.

1. Saves current values of PORT and DDR registers corresponding to USART1 and disables the USART1 module.
2. Configures TXD1 & RXD1 pins as input with internal pull-up.
3. Configures XCK1 pin as output to send calibration clock.
4. Starts the Timer 1 to output 32kHz signal on XCK1 pin (OC1B pin – Timer 1 operated in CTC mode with OC1B toggling on compare match).
5. Waits until RXD1 is pulled low.
6. Once the RXD1 is pulled low timer is stopped and signal on XCK1 pin is toggled to make the target send the OSCCAL value. For every toggling the programmer accumulates the value seen on RXD1 pin. This happens for eight times (1 byte OSCCAL value).
7. It then enables the USART1 module, restores the PORT and DDR values and sends back the OSCCAL value to the AVROSP.

The –Y command should be provided with the byte address of the flash to where the OSCCAL value is written. Also in the command line the –Y command should be accompanied with the programming commands of final application hex code. For example the –Y command should be used as below.

avrosp.exe –dattiny40 –s –Y5FF –e –pf –vf –ifuser.hex

When the above command is executed, after calibration the AVROSP reads the user.hex file and programs the target device with the hex file. While programming the hex file the AVROSP also programs the final OSCCAL value to the specified address (0x05FF byte address) of the flash. If the specified byte address is within the user hex file application code range then error message will be thrown.

4. Quick start guide

1. Download and extract the AVR057 zip file.
2. Open the folder AVR918_HEX_FILES. This folder contains the hex file of the TPI programmer. There will be three hex files available in this folder. One for the Atmel ATtiny4/5/9/10 devices, one for the Atmel ATtiny20 device and the third one for the Atmel ATtiny40 device. Select the required hex file and program the Atmel ATmega324 device with the hex file.
3. Connect an 11.0592MHz crystal between the XTAL1-XTAL2 pins of the ATmega324 and modify the fuse settings such that external crystal oscillator is selected as clock source and the CKDIV8 fuse is disabled.
4. Connect the TPI programmer to the computer as well as to the target device as shown in the block schematic in Figure 3-1.
5. Once the connections are done open command prompt and navigate to the Debug folder inside the AVR057 folder where the avrosp.exe file resides. Now execute the following commands.

   mode com1 baud=115200 parity=n data=8

   avrosp.exe –dattiny40 –Y0

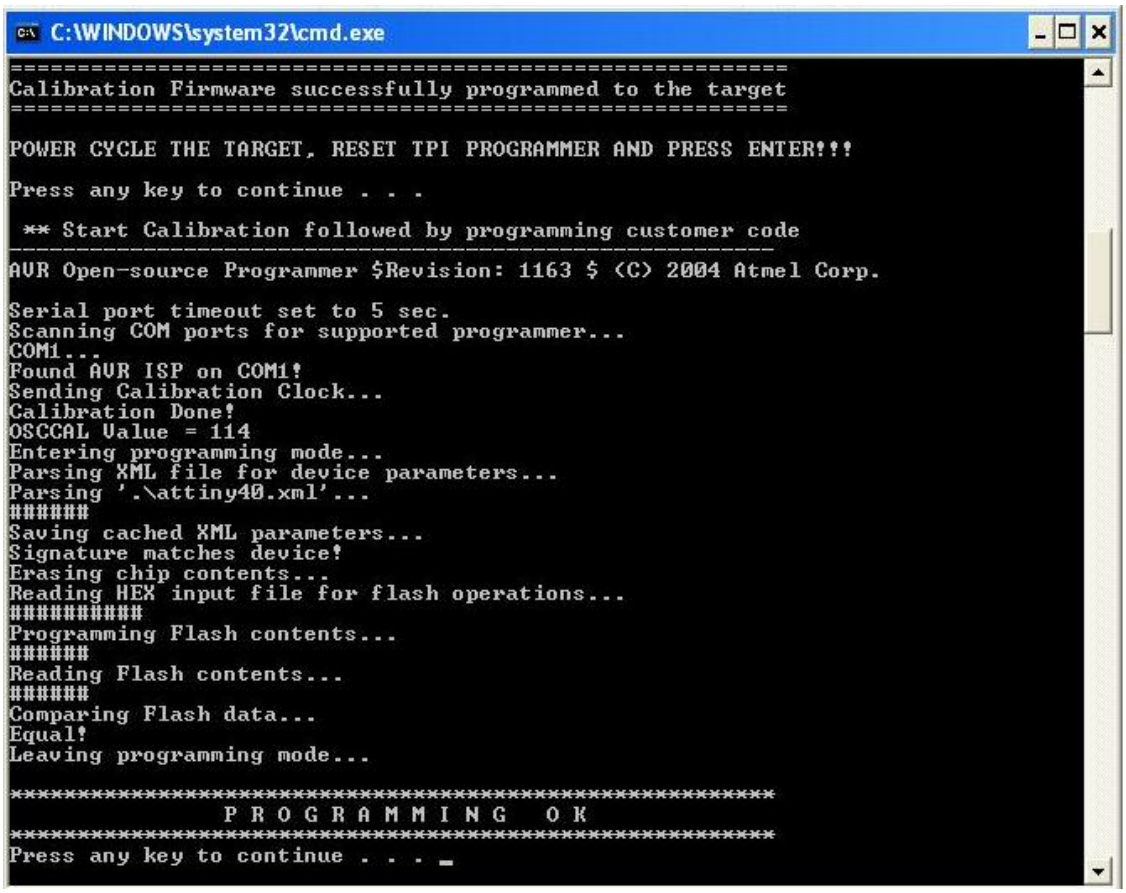
   The first command configures the COM port of the computer and the second command start the calibration. The calibration clock will be now available on the PD4 (XCK1) pin of the TPI programmer (ATmega324). Using an oscilloscope measure the frequency of the clock signal on this pin (should be approximately 32kHz).

   Note: The AVROSP timeout error will be thrown on the command prompt. This error happens because we have not loaded the calibration firmware to the target. Ignore the error and reset the TPI programmer.
6. Now open the AVR057 folder extracted from the zip file and open the Atmel Studio 6 Solution file inside the folder.
7. Open the AVR057.asm file from the solution explorer and specify the device name in the first line of the code. For example if the device is ATtiny40 then mention “.equ ATTINY = 40”.
8. Specify the calibration clock frequency in the line “.equ CALIB_CLOCK_FREQ = 31800” with the frequency measured in step 5.
9. Change the device in the Atmel Studio 6 project using Project -> AVR057 Properties -> Device -> Change Device.
10. Save the project and rebuild the project using Build -> Rebuild solution. The avr057.hex file will be generated inside the Debug folder of the project. The Debug folder also contains the avrosp.exe file as well as a batch file (avr057.bat) for convenience.
11. Open the avr057.bat batch file with notepad and specify the device name in the CPU field. Also specify the final application code hex file name in the CUSTOMERCODE field (currently it is set to user.hex) and save the batch file.
12. Copy the final application code hex file into the Debug folder of the AVR057 where the batch file exists and execute the batch file.
13. The batch file will first program the calibration firmware into the target device. Once the programming is successful the target device needs to be cycle powered. The batch file displays this message as shown in Figure 4-1.

Figure 4-1. Output after programming the target device with the AVR057.hex file.

14. Now cycle power the target device, reset the TPI programmer and hit a key enter on the computer's keyboard.
15. Once this is done the calibration starts followed by the programming of the user code. The final OSCCAL value is written to the flash location which is mentioned along with the –Y command. Figure 4-2 shows the command output when the calibration and user code programming are successful.

Figure 4-2. Output after calibration with the OSCCAL value displayed.



5. Revision history

Doc. rev.	Date	Comments
42038A	10/2012	Initial document release

Atmel Corporation
2325 Orchard Parkway
San Jose, CA 95131
USA
Tel: (+1)(408) 441-0311
Fax: (+1)(408) 487-2600
www.atmel.com

Atmel Asia Limited
Unit 01-5 & 16, 19F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
HONG KONG
Tel: (+852) 2245-6100
Fax: (+852) 2722-1369

Atmel Munich GmbH
Business Campus
Parkring 4
D-85748 Garching b. Munich
GERMANY
Tel: (+49) 89-31970-0
Fax: (+49) 89-3194621

Atmel Japan G.K.
16F Shin-Osaki Kangyo Building
1-6-4 Osaki
Shinagawa-ku, Tokyo 141-0032
JAPAN
Tel: (+81)(3) 6417-0300
Fax: (+81)(3) 6417-0370