Freescale Touch Sensing Software 2.5 Kinetis Preview

Release Notes

PRODUCT:	Freescale TSS
PRODUCT VERSION:	2.5 Kinetis Preview
DESCRIPTION:	Freescale Touch Sensing Software Library, version 2.5
RELEASE DATE:	November 9th 2010

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or +1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals", must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

Rev. 6
11/2010

Table of Contents

1 Introduction

This document describes the Freescale Touch Sensing Software (TSS) version 2.5 released for ARM®Cortex™-M4 Kinetis processor family.

This is the preview version of the TSS 2.5 library for the Kinetis family. It is based on and is backward compatible with TSS version 2.0. The features described here are not ported back to HCS08 and ColdFire® V1 platforms. A full TSS 2.5 release will bring the new library features to all Kinetis, HCS08, and ColdFire® V1 platforms.

1.1 Requirements

1.1.1 System Requirements

The ARM®Cortex™-M4 version of the library was developed, compiled and tested with IAR Embedded Workbench for ARM Version 5.50.6 and Version 6.10.

The system requirements are defined by the development tools requirements. There are no special host system requirements for hosting the Freescale TSS distribution.

Minimum PC configuration:

- As required by CodeWarrior® Development Studio

Recommended PC configuration:

- 2 GHz processor – 2 GB RAM - 2 GB free disk space

Software requirements:

- OS: As required by development tools (Windows XP SP2 or later recommended)

1.1.2 Target Requirements

The Freescale TSS in this release supports the ARM®Cortex™-M4 families of microcontrollers. The requirements for the target hardware are same as the operating requirement of your evaluation or custom board (power supply, cabling, jumper settings etc).

There are physical parameters that directly affect the Touch Sensing performance like electrodes design, PCB routing, parasitic capacitance at processor pins etc. Refer to appropriate Application Notes related to the software-based capacitive measurements available on www.freescale.com/touchsensing .

This release of Freescale TSS contains application examples for the following boards:

- TWR-K60N512 Tower board with MK60N512
- TWR-K40X256 Tower board with MK40X256

The boards are available for purchase at Freescale web site.

1.2 Special Instructions

1.2.1 Installation Instructions

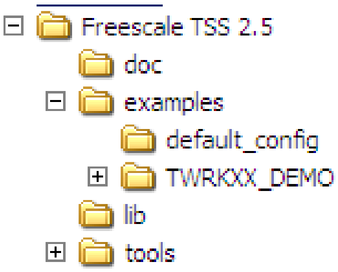
Run the self-extracting executable and proceed according to instructions on the screen. Refer to Touch Sensing Software User Guide available in the Start/Programs menu after installation.

2 Release Content

This is release version 2.5 of the Freescale Touch Sensing Software. The content is described in the following table.

Deliverable	Location	Status
TSS Library Files	**<install_dir>/lib/**	
ARM®Cortex™-M4	.../lib/TSS_ARM.a	new
Master library header file	.../lib/TSS_API.h	updated
Support header files	.../lib/*.h	reworked
Compile-time configuration library files	.../lib/*.c	reworked
Examples	**<install_dir>/examples**	
Example of user configuration header file	.../examples/default_config/ TSS_SystemSetup.h	updated
Example applications for TWR-K60n512 board	.../examples/ TWRKXX_DEMO	new
Example applications for TWR-K40x256 board	.../examples/ TWRKXX_DEMO	new
Documentation	**<install_dir>/doc**	
User Documentation	.../doc/	updated
PC Host Tools	**<install_dir>/tools**	
System Setup Creator tool	…/tools/System Setup GUI.exe	updated
Check for Latest Version tool	…/tools/webchk.exe	updated

The following picture shows the Freescale TSS directories installed to the user host computer:



```
⊟ Freescale TSS 2.5
     doc                    ---  Documentation
  ⊟  examples               ---  Configuration examples
          default_config         and example applications
       ⊞  TWRKXX_DEMO
     lib                    ---  TSS Library Files
  ⊞  tools                  ---  PC Host Tools
```

2.1 Example Applications

Example applications are included in the library distribution in the *examples* folder.

- twrk60n512 demo application for the TWR-K60N512 board servicing all four electrodes of the board and demonstrating use of Keypad decoder. TSI module is used for electrode sensing. Touch status of each electrode is indicated by LED placed inside the electrode area. The application is available in the */examples/TWRKXX_DEMO* folder.
- twrk40x256 demo application for the TWR-K40X256 board servicing all four electrodes of the board and demonstrating use of Keypad decoder. TSI module is used for electrode sensing. Touch status of each electrode is indicated by LED placed inside the electrode area. The application is available in the */examples/TWRKXX_DEMO* folder.

3 New Features

Freescale is committed to maintain this product and to deliver updates and enhancements timely. This section describes the major changes and new features implemented in this release.

In comparison to version 2.0, the TSS 2.5 Kinetis Preview release implements the following features:

- **TSS Library**
  - ARM®Cortex™-M4 support added. The TSS_ARM.a precompiled library for ARM®Cortex™-M4 was added into the lib directory.
  - Touch Sense Input (TSI) hardware module added. The module enables robust hardware-driven capacitance measurements to be performed without CPU intervention. The TSS library has been reworked to enable such a “background” processing and save CPU time and power consumption.
  - GPIO (ATL, CTS) and other low-level layer sensing algorithms are still available in the TSS library and may be used to detect touch on electrodes connected to non-TSI pins.
  - TSI active mode clock configuration parameters added into the TSS_SystemSetup.h file.
  - TSI bit-resolution parameter (TSS_TSI_RESOLUTION) added into the TSS_SystemSetup.h file. The TSS code automatically manages the TSI module runtime configuration to achieve the desired resolution.
  - ATL and CTS low-level layer sensing algorithms are no longer mutually exclusive. The CTS method can be applied to the selected electrodes just like an ordinary method additional to GPIO, TSI, and other. The electrode type is set up in the TSS_SystemSetup.h file.
  - The TSS low-level layer now enables to use the following methods: GPIO, CTS and TSI. The other methods (PTI, KBI, TPM, FTM) known from TSS 2.0 will be supported in the follow up library version. Selection of the electrode type is performed in TSS_SystemSetup.h by the TSS_En_Type macro.
  - Various trigger mechanisms are added. A new automatic trigger may help to achieve periodic electrode sampling and let the TSI module to drive the period also for non-TSI electrodes. In addition to the automatic trigger, two manual triggers may be used to reduce complexity of conditional execution of TSS_Task when any kind of periodicity is required.
  - Source code files are renamed and the code is refactored:
    - ATL_SENSOR_TIMEOUT and ATL_SENSOR_PRESCALER macros a renamed to TSS_SENSOR_TIMEOUT and TSS_SENSOR_PRESCALER.
    - ATL_Timer.h file renamed to TSS_Timer.h file.
    - Macros with ATL_HW_TIMER_ prefix were renamed to use TSS_HW_TIMER_ prefix in TSS_Timer.h file.
    - ATL_Sensor.h and ATL_Sensor.c files renamed to TSS_Sensor.h and TSS_Sensor.c.
    - Interrupt handlers for the low level layer methods moved to the appropriate TSS_SensorXXX.c which enables to assign interrupt vector number automatically.

4 Release Description

4.1 Supported Features

- TSI HW module support for ARM®Cortex™-M4 Kinetis processors.
- Software-only capacitive touch sensing for. ARM®Cortex™-M4 Kinetis processors.
- Backward compatibility with TSS 1.x and 2.0.
- Easy to use and integrate with existing user applications.
- Electrode malfunction detection.
- Support for up to 64 electrodes.
- Compile-time configurable using a single header file (at user application level).
- Graphical System Setup Creator utility to help creating the configuration header file.
- Several touch-detection and capacity measurement algorithms available:
    - TSI, GPIO, CTS.
- Advanced Key Detector signal processing layer.
- Decoded signals available to user application.
- Easy to use decoding structures with callback event notification.
    - Keypad – keyboard decoder supporting multiple key press and auto-repeat.
    - Slider – handling linearly distributed electrodes as a single up-down or left-right control.
    - Rotary – handling ring-distributed electrodes as a single jog-dial-like control.
- Supports up to 16 instances of decoding controls.

4.2 Limitations

- This is the Kinetis Preview version of the library only. New features are not available to HCS08 and ColdFire® V1 platforms.
- TSI and GPIO are the only available sampling methods. The pin interrupt and timer input capture methods will be implemented in the future library version.
- Advanced Low Power and Wake-up features of the TSI module are not enabled in the current library version.

5 Release History

Version 1.0 (September 21st 2009)

- First public release of the library with an example for LG32-based TSSEVB Rev.B evaluation board.

Version 1.1 (January 27th 2010)

- TSS.lib
  - Baseline Tracking bug fixed. The baseline was updated slowly in case of negative delta value.
  - TSS_ERROR_KEYPAD_NOT_IDLE state removed from TSS Keypad Decoder.
  - DC Tracker init value changed from 64 to 100.
  - Number of CTS measurements allowed to be interrupted by user application before a timeout occurs was increased from 20 to 128.
  - Setting the System Reset bit in the TSS System Configuration Register makes TSS to restart immediately.
  - atl_u8SampleIntFlag variable definition moved from ATL_Sensor.h to C code.
- TSSEVB_SINGLE application example
  - I2CDvr.c file updated so the sensitivity can be set from EGT.
- TSSEVB_DEMO application example
  - I2CDvr.c file updated so the sensitivity can be set from EGT.
  - BUSclk changed to 20 MHz if CTS sensing algorithm selected. This enhances algorithm sensitivity.
  - SCI baud rate setting fixed since 20MHz bus clock is used with CTS sensing algorithm selected. Needed to properly communicate with COMM JM60 device.
  - Electrodes Sensitivity changed for Washing Machine demo application if CTS sensing algorithm selected

Version 2.0 (August 23rd 2010)

- ColdFire® V1 support added. The TSS.lib precompiled library for HCS08 family was renamed to TSS_S08.lib and the TSS_CFV1.a precompiled library for ColdFire® V1 was added into the same directory.
- The IIR filter feature was implemented at the Key detector level. The filter processes capacitance values obtained from low-level routines and works with both ATL and CTS algorithms. Use of this feature is optional, enable it in TSS_SystemSetup.h.
- The Noise Amplitude Filter function was implemented in the ATL and CTS low level. The user can define the noise amplitude to be filtered. Noise peaks greater than the defined amplitude are filtered by the system, thus disregarding the noisy sample. Use of this feature is optional; enable it in TSS_SystemSetup.h together with setting the Noise Amplitude Filter sizes for each electrode.

- The SWI feature which can be enabled in the TSS registers is available only for HCS08 version of the TSS library. The OnFault callback feature was added to enable handling of a fault situation on both HCS08 and ColdFire® V1 processors. Specify name of application callback function as the TSS_ONFAULT_CALLBACK parameter in the TSS_SystemSetup.h file.
- Baseline balancing algorithm was simplified in the Key detector code.
- The ATL low level layer now enables to use also GPIO Port Interrupt, KBI and TPM Input Capture modules to improve sensitivity. Use of this feature is optional, enable it in TSS_SystemSetup.h.
- New TSS_Task "sequencing" feature enables to divide task processing to several steps where each electrode is acquired in separate TSS_TaskSeq call. When all electrodes are processed, the decoders are handled all at once in the last TSS_TaskSeq call.
- Default electrode pin state was changed to logic-high when measurement is idle. This helps to achieve lower power consumption in low power modes. The only exception is that a pin is set to logic-low state when timer timeout occurs (electrode charge timeout). Timeout may be a symptom of short-grounded electrode, so setting the output pin to logic low prevents high current sourced from pin and achieves lower power consumption.
- ATL_SENSOR_TIMEOUT and ATL_SENSOR_PRESCALER macros were moved to TSS_SystemSetup.h and are now configurable. Default value of ATL_SENSOR_TIMEOUT was set to 511, ATL_SENSOR_PRESCALER was set to 2.
- Macros with ATL_TIMER_ prefix were renamed to use ATL_HW_TIMER_ prefix in ATL_Timer.h in order to differentiate it from ATL_IC_TIMER_macros used for Timer Input Capture method.
- ATL HW Timer Interrupt handler moved from inside of library to the ATL_Sensor.c which enables to assign interrupt vector number automatically.
- GPIO Pin Interrupt-based measurement method added.
- FTM timer support added.
- The type of ATL Low Level routine return value was changed from UINT8 to UINT16, making it more general for large capacitance differences between electrodes.
- The ATL_ElectrodesSetState function code was reduced in size and was renamed to ATL_ElectrodesSetStateHigh. The function now only sets all electrodes to logic output-high state as this is the only stat really used.
- Checking of Fault timeout and the u8FaultCnt counter variable was added to ATL_SampleElectrode function. The timeout is set by macro ATL_FAULT_TIMEOUT in ATL_Sensor.h.
- Fixed issues:
  - When more than seven controls were used, the tss_cau8BuffMask[] array in TSS_SystemSetupData.c was not defined properly.
  - Removed warning messages when no control is used. The tss_pau8EventsBuff[] and tss_acpsCSStructs[] arrays were not correctly defined in TSS_SystemSetupData.c.
  - Removed redundant Warning messages if Slew Rate and Strength registers do not exist.

- **TSSEVB_SINGLE application example**

    - ATL_SENSOR_TIMEOUT and ATL_SENSOR_PRESCALER macros were moved to TSS_SystemSetup.h. Macro ATL_HW_TIMER_TIMEOUT set to 1023, ATL_SENSOR_PRESCALER set to 2.
    - ATL_TimerIsr vector assignment removed from .prm file. This is now done automatically by the TSS library code.

- **TSSEVB_DEMO application example**
    - ATL_SENSOR_TIMEOUT and ATL_SENSOR_PRESCALER macros were moved to TSS_SystemSetup.h. Macro ATL_HW_TIMER_TIMEOUT set to 1023, ATL_SENSOR_PRESCALER set to 2.
    - ATL_TimerIsr vector assignment removed from .prm file. This is now done automatically by the TSS library code.
    - Electrode pins where an alternative KBI or TPM channel feature is available were reconfigured in TSS_SystemSetup.h to use new type of measurement.
- **Processor Expert support**

  Processor Expert TSS Component v1.0 included in the form of PEupd package. The component may help to configure the TSS library in an easy to use graphical environment.

Version 2.5 (Kinetis Preview 9th 2010)

- TSI module added.
- Library implementation for the ARM®Cortex™-M4 Kinetis processor family.
- Demo applications provided for TWR-K60n512 and TWR-K40x256 boards. For details, refer to New Features.