Document Number: TSSRN
Rev. 17, 08/2013

Freescale Touch Sensing Software 3.1.0
Release Notes

PRODUCT:	Freescale TSS
PRODUCT VERSION:	3.1.0
DESCRIPTION:	Freescale Touch Sensing Software Library, version 3.1.0
RELEASE DATE:	August, 2013

© 2013 Freescale Semiconductor, Inc. All rights reserved.

Table of Contents

1 Read Me First

This document describes the Freescale Touch Sensing Software version 3.1.0 released for Freescale HCS08, ColdFire V1, ColdFire+, ARM®Cortex®-M0+ and ARM®Cortex®-M4 Kinetis processor families.

1.1 Requirements

1.1.1 System Requirements

The following tools were used to develop and test the library code and example applications:

- CodeWarrior Development Studio for MCU Version 10.4 - Build B130425
  - covering the HCS08, ColdFire V1, ColdFire+, Kinetis K and L platforms.
- CodeWarrior Development Studio for Microcontrollers Version 6.3.
  - covering the HCS08 and ColdFire V1 platforms.
- IAR Embedded Workbench® for ARM® Version 6.50.2.4585
  - covering the Kinetis K and L platforms.
- ARM-MDK™ uVision® 4.71.2.0
  - covering the Kinetis K and L platforms.
- Cosmic Tools for Freescale 68HC08/HCS08 family v4.6.8.
  - covering the HCS08 platforms.

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

Minimum PC configuration:

- As required by CodeWarrior Development Studio, IAR Embedded Workbench or ARM MDK uVision

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 code in this release supports the Freescale HCS08, ColdFire V1, ColdFire+, Kinetis K and Kinetis L families of microcontrollers in general. There are no special requirements for the target hardware other than what your evaluation or custom board requires for its operation (power supply, cabling, jumper settings etc).

There are physical parameters which directly affect the Touch Sensing performance like electrodes design, PCB routing, parasitic capacitance at processor pins etc. Discussing these parameters is out of scope of this document. Please refer to appropriate Application Notes related to the software-based capacitive measurement available on freescale.com/touchsensing .

This release of Freescale TSS library contains application examples for:

- TSSEVB TSS Evaluation Board with MC9S08LG32
- DEMOQE Demonstration Board with MCF51QE128 + KITPROXIMITYEVM board
- DEMOACKIT Demonstration Board with MCF51AC256 + KITPROXIMITYEVM board

- KWIKSTIK Development board with MK40X256
- FRDM-KL02Z Development board with MKL02Z32
- FRDM-KL05Z Development board with MKL05Z32
- FRDM-KL25Z Development board with MKL25Z128
- FRDM-KL26Z Development board with MKL26Z128
- TWR-K20D50M Tower System module with MK20DX128
- TWR-K20D72M Tower System module with MK20DX256
- TWR-K40X256 Tower System module with MK40X256
- TWR-K53N512 Tower System module with MK53N512
- TWR-K60N512 Tower System module with MK60N512
- TWR-K60D100M Tower System module with MK60DN512
- TWR-K70F120M Tower System module with MK70FN1M
- TWR-MCF51QM Tower System module with MCF51QM128
- TWR-MCF51JF Tower System module with MCF51JF128
- TWR-S08PT60 Tower System module with MC9S08PT60
- TWR-KL05Z48M Tower System module with MKL05Z32
- TWR-KL25Z48M Tower System module with MKL25Z128
- TWR-KL46Z48M Tower System module with MKL46Z256

These development boards are available for purchase at Freescale web site.

1.1.3 Freescale MQX™ RTOS Integration

This version brings examples demonstrating integration of TSS library in MQX applications. The MQX version included in this release is derived from MQX version 4.0.1.

The MQX examples require the serial console to be set up properly to display output and get user input. By default, the MQX console is routed to the virtual serial line implemented by the OSBDM or OSJTAG USB connection. For OSBDM firmware versions prior to 31.00, you need a special Terminal Utility from the OSBDM Virtual Serial Toolkit to access the console. For firmware versions 31.00 and later, the virtual serial line appears as a standard COM port on the Host PC and can be accessed by any terminal utility. Visit www.pemicro.com/osbdm for more details on OSBDM.

The serial console can be also reconfigured and redirected to a different communication channel in the MQX BSP as needed. In order to test the TSS with any other MQX configuration, please visit the freescale.com/mqx and download the full Freescale MQX RTOS installation package. Refer to MQX Release Notes and MQX Getting Started documentation included in the full MQX package for more information about MQX configuration and build process.

1.1.4 Processor Expert Integration

The Processor Expert TSS Component is included directly in the official release of the Components Library of Processor Expert available integrated in the CodeWarrior version 10.4 and DriverSuite v10.1 and later

1.2 Special instructions

1.2.1 Setup 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.

1.2.2 Update of OSBDM firmware on TSSEVB

If an older version of TSSEVB board is used you will probably need to update the OSBDM firmware in order the communication with FreeMASTER tool works properly. To re-program the OSBDM firmware, follow the instructions in OSBDM-JM60_Users_Guide section. Update OSBDM-JM60 Firmware with Freescale JM60 GUI Application and use J7 header for this purpose. The OSBDM-JM60_Users_Guide document together with the last OSBDM firmware can be downloaded from the Freescale web site at frescale.com.

1.2.3 Update of OSBDM firmware on Freescale Tower System processor modules

Upon starting the first debugger session with a new Tower System Module, the CodeWarrior may prompt you to upgrade the OSBDM firmware to the latest version. Follow the instructions on the screen and complete all necessary steps before running your first TSS application.

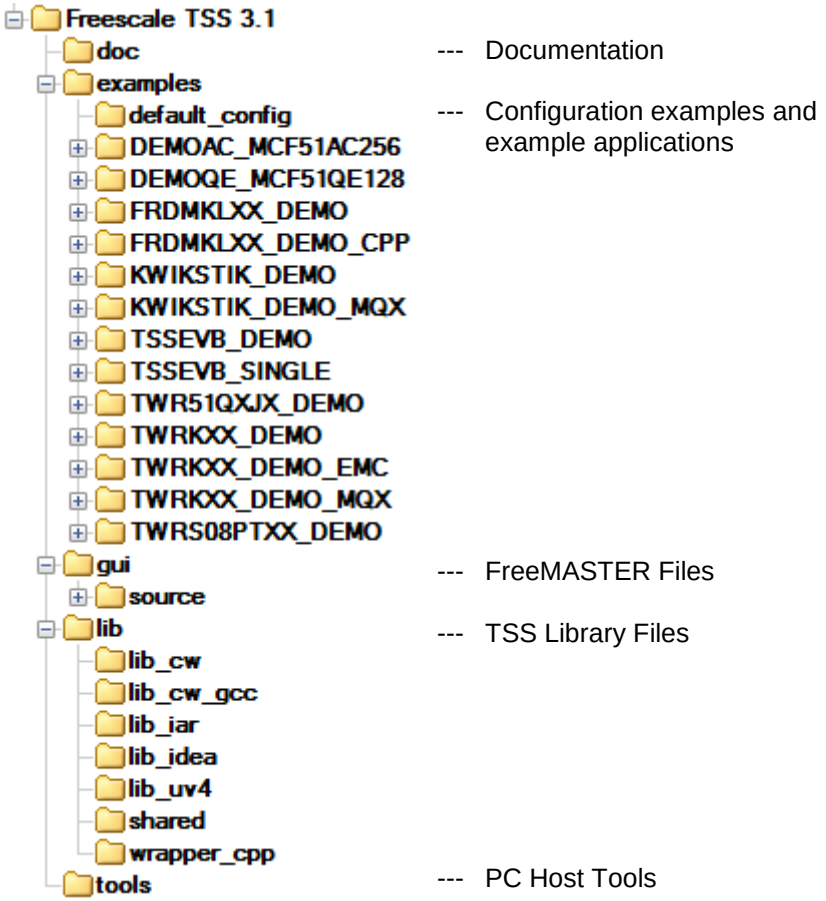
2 Release Content

This is release version 3.1.0 of the Freescale Touch Sensing Software. The content is described in the following table.

Deliverable	Location	Status
TSS Library Files	**<install_dir>/lib/**	
CodeWarrior version of library for ARM®Cortex™-M4, Freescale	.../lib/lib_cw/TSS_KXX_M4.a	Updated
CodeWarrior version of library file for HCS08	.../lib/lib_cw/TSS_S08.lib	Updated
CodeWarrior version of library file for ColdFire+ & V1	.../lib/lib_cw/TSS_CFV1.a	Updated
CodeWarrior version of library for ARM®Cortex™-M0, GCC	.../lib/lib_cw_gcc/libTSS_KXX_M0.a	Updated
CodeWarrior version of library for ARM®Cortex™-M4, GCC	.../lib/lib_cw_gcc/libTSS_KXX_M4.a	Updated
CodeWarrior version of library for ARM®Cortex™-M4 FPU, GCC	.../lib/lib_cw_gcc/libTSS_KXX_M4_FPU.a	New
IAR version of library file for ARM®Cortex™-M0	.../lib/lib_iar/TSS_KXX_M0.a	Updated
IAR version of library file for ARM®Cortex™-M4	.../lib/lib_iar/TSS_KXX_M4.a	Updated
ARM uVision version of library file for ARM®Cortex™-M0	.../lib/lib_uv4/TSS_KXX_M0.lib	Updated
ARM uVision version of library file for ARM®Cortex™-M4	.../lib/lib_uv4/TSS_KXX_M4.lib	Updated
Cosmic version of library file for HCS08	.../lib/lib_idea/TSS_S08.lib	New
Master library header file	.../lib/ shared/TSS_API.h	Updated
Support header files	.../lib/shared/*.h	Updated
Compile-time configuration library files	.../lib/shared/*.c	Updated
Examples	**<install_dir>/examples**	
Example of user configuration header file	.../examples/default_config/TSS_SystemSetup.h	Updated
Example applications for TWR-S08PT60 + GUI	.../examples/TWRS08PTXX_DEMO	Updated
Example applications for TWR-MCF51JF + GUI	.../examples/TWR51QXJX_DEMO	From 3.0
Example applications for TWR-MCF51QM + GUI	.../examples/TWR51QXJX_DEMO	From 3.0
Example applications for TSSEVB + GUI	.../examples/ TSSEVB_DEMO	From 3.0
Example applications for TSSEVB + GUI single electrode	.../examples/ TSSEVB_SINGLE	From 3.0
Example applications for DEMOQE + GUI	.../examples/ DEMOQE_MCF51QE128	From 3.0
Example applications for DEMOACKIT + GUI	.../examples/ DEMOAC_MCF51AC256	From 3.0
Example applications for TWR-K20D50M board + GUI	.../examples/ TWRKXX_DEMO	Updated
Example applications for TWR-K20D72M board + GUI	.../examples/ TWRKXX_DEMO	Updated
Example applications for TWR-KL05Z48M board + GUI	.../examples/ TWRKXX_DEMO	Updated
Example applications for TWR-KL25Z48M board + GUI	.../examples/ TWRKXX_DEMO	Updated
Example applications for TWR-KL46Z48M board + GUI	.../examples/ TWRKXX_DEMO	New
Example applications for TWR-K40X256 board + GUI	.../examples/ TWRKXX_DEMO	Updated
Example applications for TWR-K53N512 board + GUI	.../examples/ TWRKXX_DEMO	Updated
Example applications for TWR-K60N512 board + GUI	.../examples/ TWRKXX_DEMO	Updated
Example applications for TWR-K60D100M board + GUI	.../examples/ TWRKXX_DEMO	New
Example applications for TWR-K70F120M board + GUI	.../examples/ TWRKXX_DEMO	Updated
Example applications for FRDM-KL02Z48M board + GUI	.../examples/ FRDMKLXX _DEMO	New
Example applications for FRDM-KL05Z48M board + GUI	.../examples/ FRDMKLXX _DEMO	New
Example applications for FRDM-KL25Z48M board + GUI	.../examples/ FRDMKLXX _DEMO	Updated
Example applications for FRDM-KL26Z48M board + GUI	.../examples/ FRDMKLXX _DEMO	New
EMC example applications for TWR-KL05Z48M board + GUI	.../examples/ TWRKXX_DEMO_EMC	New
EMC example applications for TWR-KL25Z48M board + GUI	.../examples/ TWRKXX_DEMO_EMC	New
EMC example applications for TWR-KL46Z48M board + GUI	.../examples/ TWRKXX_DEMO_EMC	New
MQX example applications for TWR-K40X256 board + GUI	.../examples/ TWRKXX_DEMO_MQX	Updated
MQX example applications for TWR-K53N512 board + GUI	.../examples/ TWRKXX_DEMO_MQX	Updated
MQX example applications for TWR-K60N512 board + GUI	.../examples/ TWRKXX_DEMO_MQX	Updated
MQX example applications for TWR-K70F120M board + GUI	.../examples/ TWRKXX_DEMO_MQX	Updated
C++ wrapper example applications for FRDM-KL02Z48M board	.../examples/ FRDMKLXX _DEMO_CPP	New
C++ wrapper example applications for FRDM-KL05Z48M board	.../examples/ FRDMKLXX _DEMO_CPP	New
C++ wrapper example applications for FRDM-KL25Z48M board	.../examples/ FRDMKLXX _DEMO_CPP	New
C++ wrapper example applications for FRDM-KL26Z48M board	.../examples/ FRDMKLXX _DEMO_CPP	New

Example applications for KWIKSTIK board + GUI	.../examples/ KWIKSTIK_DEMO	From 3.0
MQX example applications for KWIKSTIK board + GUI	.../examples/ KWIKSTIK_DEMO_MQX	Updated
FreeMASTER Files	**<install_dir>/gui**	
FreeMASTER source files	…/gui/source/	Updated
FreeMASTER universal GUI project for TSS	…/gui/TSS_FMSTR_GUI.pmp	Updated
Documentation	**<install_dir>/doc**	
User Documentation	.../doc/	Updated
PC Host Tools	**<install_dir>/tools**	
Check for the Latest Version tool	…/tools/webchk.exe	Updated

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



2.1 Example Applications

There are several example applications included in the TSS distribution in the *examples* folder. Use any of the applications with the FreeMASTER GUI project located in the *gui* subfolder in each of the examples.

- **TSSEVB_DEMO** application for the HCS08LG32 servicing all electrodes of the TSSEVB evaluation board and demonstrating three types of decoders (Keypad, Rotary and Slider). Status of electrodes and values of each decoder control is displayed on the LCD display. The demo application is located in the */examples/TSSEVB_DEMO* folder.
- **TSSEVB_SINGLE** is a simple application servicing the one E1 electrode of the TSSEVB evaluation board. It is located in the */examples/TSSEVB_SINGLE* folder.
- **DEMOQE, DEMOAC** applications for the MCF51QE128 and MCF51AC256 MCUs servicing eight electrodes of the external KITPROXIMITYEVM touch pad board. The application demonstrates use of Keypad decoder. The application is available in the */examples/DEMOQE_MCF51QE128* folder.
- **TWRK20DX50, TWRK20DX72, TWRKL05Z48, TWRKL25Z48, TWRKL46Z48, TWRK53N512, TWRK40X256, TWRK60N512, TWRK60D100M, TWRK70F120M** Kinetis demo applications servicing all on-board electrodes and also external touch pad daughter boards TWRPI-KEYPAD, TWRPI-ROTARY, TWRPI-SLIDER, TWRPI-ROTARY2, TWRPI-TOUCHPAD and TWRPI-PROXIMITY. The applications demonstrate use of Keypad, Rotary and Analog Slider/Rotary decoders with the TSI capacitance measurement module, different triggering modes and processor wake up from Low Power mode by an electrode touch. Touch status of each electrode is indicated by LED placed inside the electrode area. The applications are available in the */examples/TWRKXX_DEMO* folder.
- **TWRKL05Z48, TWRKL25Z48, TWRKL46Z48** Kinetis L demo applications are also prepared in a version demonstrating the AFID keydetector and TSI noise mode feature. This demo is prepared to sustain EN61000-4-6 test. The demo presents just simple key based TWRPI boards like TWRPI-KEYPAD, TWRPI-ROTARY, TWRPI-SLIDER, and TWRPI-KEYPAD2. The applications are available in the */examples/TWRKXX_DEMO_EMC* folder.
- **TWRK40X256, TWRK53N512, TWRK60N512, TWRK70F120M** demo applications are also prepared in a version demonstrating the integration with the MQX real time operating system. The integration is performed at the HMI layer which can be used to extend or replace functionality of the mechanical push buttons. The MQX example applications are available in the */examples/TWRKXX_DEMO_MQX* folder.
- **TWR51QM,** TWR51JF ColdFire+ demo applications servicing all on-board electrodes and external TWRPI touch pad boards. This example demonstrates use of the TSI capacitance measurement module and the same features as the TWRKXX_DEMO application. The applications are available in the */examples/TWRQXJX_DEMO* folder.
- **TWRS08PT60** demo application for the TWR-S08PT60 board servicing four on-board electrodes and external TWRPI touch pad boards. This example demonstrates use of the TSI capacitance measurement module and the same features as the TWRKXX_DEMO application. The applications are available in the */examples/TWRS08PTXX_DEMO* folder.
- **FRDM-KL02Z, FRDM-KL05Z, FRDM-KL25Z, FRDM-KL26Z** Kinetis L demo applications servicing two on-board electrodes in Analog Slider control configuration. TSI module is used for the touch sensing. Touch status of each electrode is indicated on the tri-color LED. The applications are available in the */examples/ FRDMKLXX_DEMO* folder.

- **FRDM-KL02Z, FRDM-KL05Z, FRDM-KL25Z, FRDM-KL26Z** Kinetis L demo applications are also prepared in a version demonstrating the C++ Wrapper approach. The C++ Wrapper brings simplified object oriented API for TSS control. The C++ Wrapper example applications are available in the */examples/FRDMKLXX_DEMO_CPP* folder.
- **KWIKSTIK** demo based on K40X256 device servicing six on-board electrodes. The application demonstrates use of Keypad control. Both TSI module and GPIO measurement method is used for the touch sensing. Touch status of each electrode is indicated on the onboard LCD display. FreeMASTER visualization GUI can be connected via TWR-SER board connected to TWR-ELEV Tower System Elevator bus. The application is also available in the MQX version.

3 What's New

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.

Comparing to previous TSS 3.0.1 release, this version implements the following new features:

- **TSS Library**
  - The TSS library supports Kinetis KL02, KL46 and KL26 family.
  - Cosmic compiler support was added. The TSS_S08.lib precompiled library available for the application done with the Cosmic compiler.
  - The libTSS_KXX_M4_FPU.a precompiled library is available for ARM®Cortex™-M4 MCU with FPU support and for CodeWarrior GCC compiler.
  - New AFID Keydetector (Advanced Filtering and Integrating Detection) can be optionally enabled. The AFID touch detection is based on using two IIR filters with different depths (one short/fast, the other long/slow) and on integrating the difference between the two filtered signals. This algorithm is more immune against noise, but is not compatible with other noise-cancellation techniques like shielding.
  - TSI Noise mode is supported for KLxx family. The noise mode feature is an alternative hardware measurement method intended for touch detection in an extremely noisy environment. Optimal when enabled along with the robust AFID Keydetector.
  - mbed online compiler is supported. C++ wrapper was developed for object oriented access to TSS features. New example applications demonstrating the C++ wrapper layer are available.
  - Configuration of controls in TSS_SystemSetup.h file was changed. A requirement of sequential assignment of electrodes is no longer applicable. Electrodes can now be assigned to a control directly which enables to reuse the same TSI inputs in more controls.
  - Idle scan rate function has been removed from all decoders as obsolete.
  - AutoTriggerModuloValue register and LowPowerScanPeriod were removed as they were not compatible with the latest versions of TSI module. The registers can be still configured through TSS_TSI_SMOD and TSS_TSI_LPSCNITV macros.
  - Baseline is now updated also in proximity and shielding modes. The update period is defined in SlowDCTrackerFactor register, which is a multiplication factor used with standard DCTrackerRate register value.
  - Enabling or disabling controls now also enables or disables all electrodes assigned to the control. A user does not need to configure ElectrodeEnablers registers manually if any control is enabled or disabled.
  - New API functions were added to support store and restore of the TSS configuration. This is used for a recovery of the TSS configuration after low power wake up. TSS_StoreElectrodeData(), TSS_StoreModulesData() and TSS_StoreAllSystemData() functions copy data to user memory area. TSS_LoadElectrodeData(), TSS_LoadModulesData() and TSS_LoadAllSystemData() functions copy data back to TSS configuration.
  - TSS enables to configure minimum level of sensitivity value in Automatic Sensitivity Calibration. To enable this feature use TSS_USE_AUTO_SENS_CALIB_LOW_LIMIT macro. Limit value is defined by initial value of Sensitivity register.

- Low power wake-up threshold can be initialized from dc-tracker baseline. To enable this feature use TSS_USE_LOWPOWER_THRESHOLD_BASELINE macro.
- Workaround for low power errata on TSI modules in Kinetis silicon versions 2 and 3 is now available. To enable this feature use TSS_USE_LOWPOWER_CALIBRATION macro.

- **Example Demo applications**
  - All existing demo applications were updated for new features of TSS 3.1.0.
  - File structure of TWRKXX_DEMO and FRDMKLXX project was simplified with regards to support of multiple boards with the same code. The following boards are newly supported by the examples:
    - The FRDMKL02 demo application for the FRDM-KL02Z board was added.
    - The FRDMKL05 demo application for the FRDM-KL05Z board was added.
    - The FRDMKL26 demo application for the FRDM-KL26Z board was added.
    - The TWRKL46 demo application for the TWR-KL46Z board was added.
    - The TWRK60D100M demo application for the TWR-K60D100M board was added.
  - TWRPI-PROXIMITY daughter board is supported in all examples and boards which provide TWRPI connector.
  - TWRKXX_DEMO_EMC was added for demonstration of the AFID keydetector and TSI noise mode feature.
  - TWRK15_DEMO demo application for the TWR-K15E256 board has been removed as obsolete.

- **FreeMASTER GUI**

  The TSS directly supports FreeMASTER GUI which enables to setup and analyse the TSS application. Each demo application contains its own FreeMASTER GUI project located in the *<example folder>/gui* folder. The FreeMASTER support files are stored in the *<install_dir>/gui* folder together with the universal FreeMASTER GUI project.

4 Release Description

4.1 Supported Features

- TSI module support for Kinetis K, Kinetis L, ColdFire+ and HCS08 PTxx family of Freescale processors.
- Simple API, easy to use and integrate to existing user applications.
- Electrode malfunction detection.
- Support for up to 64 electrodes.
- Compile-time configurable using a single header file (at user application level).
- Processor Expert component to generate the TSS project and configuration is available in component library of latest CodeWarrior Development Studio for MCU Version 10.x
- TSI and GPIO touch-detection and capacitance measurement algorithms available.
- Two versions of Key Detector signal processing layer, basic detector and AFID method with extended noise immunity.
- Decoded capacitance signals available to user application.
- Low Power wake up function enabled by TSI module.
- Three triggering modes (ALWAYS, SW, AUTO).
- Easy to use decoder controls 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.
    - Analog Slider – similar to Slider control, but consisting of fewer electrodes and achieving a better resolution.
    - Analog Rotary – similar to Rotary control, but consisting of fewer electrodes and achieving a better resolution.
    - Matrix – detecting touch on 2D X/Y matrix of electrodes with gesture detection support.
- Supports up to 16 instances of controls.

4.2 Limitations

- The Low Power and Wake-up features of the TSI module are available on Kinetis and ColdFire+ devices only (not the HCS08 PTxx family).
- The AFID Keydetector algorithms are based on mathematical filters rather than on baseline tracking and delta signal evaluation. This makes the AFID incompatible with shielding noise avoidance method and with the under-water operation. All other TSS features like analog controls, low power wake-up, or proximity are fully supported.

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 measurement which is 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 Keydetector 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 (April 18th 2011)

- **TSS Library**
  - ARM®Cortex™-M4 support is added. The TSS_KXX.a ARM®Cortex™-M4 precompiled library for IAR Embedded Workbench was added into the lib/lib_iar directory and version for CodeWarrior development Studio v10.1 was added into the lib/lib_cw directory.
  - Touch Sense Input (TSI) hardware module support added. The new TSI 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 and low power 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. This automatic management can be limited by paramaters specified in TSS_SystemSetup.h .
  - ATL and CTS low level layer sensing algorithms are no longer mutually exclusive. The CTS method can be applied to 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 supports all following measurement methods also known from TSS 2.0: GPIO, PTI, KBI, TIC (Timer Input Capture), CTS and TSI. Most of these methods are available on all HCS08, ColdFire V1 and Kinetis platforms. The TSI is available on Kinetis only. The KBI is available on HCS08 and ColdFire V1 platforms only. Selection of the electrode type is performed in TSS_SystemSetup.h by the TSS_En_Type macro.
  - Various trigger mechanisms were 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. The “automatic” and “sw” trigger function are available just on MCU’s which contains TSI module

- Both TSS_Task and TSS_TaskSeq functions now return TSS_STATUS_PROCESSING status if the measurement is in progress and TSS_STATUS_OK if the measurement was finished and data processed.
- MCU Wake-up function added. The TSS is able to wake-up MCU from low power mode by TSI module. This function is available on MCU’s with TSI module only.
- A TSS_USE_SIGNAL_LOG configuration macro added to make the TSS_FillSignalLogBuffer array available to the user application. This array contains instant signal electrode values.
- Checksum protection and data corruption checkings of TSS structures can be disabled by TSS_USE_DATA_CORRUPTION_CHECK macro. This may help to reduce Flash memory footprint of TSS.
- A TSS_ENABLE_DIAGNOSTIC_MESSAGES configuration macro added to enable compile-time warnings which remind user not to forget installing TSS interrupt handlers.
- The new TSS_freemaster.c file was added into the lib structure. This file enables FreeMASTER application to visualize internal TSS registers and signals. The FreeMASTER connection is enabled by TSS_USE_FREEMASTER_GUI configuration macro in the TSS_SystemSetup.h file.
- The tss_fOnInit callback was added. This callback function is called during TSS Init. The user application should implement this function and enable required peripheral clocks, port multiplexer etc.
- Source code files were renamed and the code was refactored:
  - ATL_SENSOR_TIMEOUT and ATL_SENSOR_PRESCALER macros were 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 on HCS08 and ColdFire V1 platforms.

- **Example Demo applications**

  All existing demo applications were updated for new features of TSS 2.5.

- **FreeMASTER GUI**

  TSS directly supports FreeMASTER GUI which provides possibility to setup and analyse TSS solution. Each demo application contains its own FreeMASTER GUI project located in the *<example folder>/gui* folder. The FreeMASTER support files are placed in the *<install_dir>/gui* folder together with universal FreeMASTER GUI project.

- **Processor Expert support**

  Processor Expert TSS Component for CodeWarrior Development Studio for Microcontrollers Version 6.3 was updated for new features of TSS 2.5. The new Processor Expert TSS Component for CodeWarrior Development Studio for MCU Version 10.1 was

added into the install package. The component may help to configure the TSS library in an easy to use graphical environment.

Version 2.5.1 ColdFire+ Preview (May 11th 2011)

- **TSS Library**
  - ColdFire+ support is added. All TSS registry configurable features are available. The TSS_CFV1.a ColdFire V1 library for CodeWarrior development Studio v10.1 is reused, because the library is fully compatible with ColdFire+ architecture.
  - Touch Sense Input (TSI) hardware module, GPIO and PTI low level layer sensing algorithms are available for ColdFire+ in this release.
- **Example Demo applications**

  TWR51JF demo application for the TWR-MCF51JF board was added.

Version 2.5.2 ColdFire+ Preview (May 27th 2011)

- **TSS Library**
  - Timer Input Capture (TIC) and CTS low level layer sensing algorithms were added, so the TSS low level layer supports on ColdFire+ all measurement methods known from TSS 2.5: GPIO, PTI, KBI, TIC (Timer Input Capture), CTS and TSI.
  - The issue with incorrect optimization pragma usage in CTS method in CodeWarrior was fixed. The user does not need to use TSS_USE_PE_COMPONENT macro anymore for solving of this issue.
- **Example Demo applications**

  TWR51QM demo application for the TWR-MCF51QM board was added.

Version 2.6 (March 27th 2012)

- **TSS Library**
  - The TSS 2.6 release combines the TSS 2.5 version and TSS 2.5.2 ColdFire+ Preview version. It brings software library with unified API for ARM®Cortex™-M4, ColdFire V1, ColdFire+ and HCS08 platforms and different capacitance measurement algorithms.
  - The pre-compiled libraries are available in lib/lib_cw and lib_iar directory for CodeWarrior development Studio v10.1 and for IAR Embedded Workbench.
  - ARM MDK (Keil) uVision 4 support for ARM®Cortex™-M4 platforms was added. The libraries are available in the lib/lib_uv4 directory.
  - HCS08 S08PTxx family support added. The TSS low level layer now supports new version of TSI hardware module available on the S08PTxx family. Two low-level sampling algorithms are available. A default one is the Kinetis-like approach with a support of interrupt-driven measurement performed on CPU background. The second version is simpler, smaller and optimized for S08 platform where the sampling takes place in the TSS_Task body. Auto triggering is not available in the simpler version. The simple version is enabled by TSS_USE_SIMPLE_LOW_LEVEL configuration macro.
  - The RTC timer is used as auto triggering source for S08PTxx TSI module. The TSS_USE_AUTOTRIGGER_SOURCE macro which defines the auto trigger source must be set to RTC on PT60 platform and TSI0 on Kinetis platforms.

- The SW trigger is now enabled for all measurement methods and platforms, not only the TSI-based platforms.
- Shielding function was added. The function performs subtraction between capacitive electrode signal and a signal of so-called shielding electrode. This approach helps to compensate environment noise or signal drift of the electrode. The function is enabled by TSS_USE_SIGNAL_SHIELDING in TSS_SystemSetup.h file. If enabled the macro TSS_Ex_SHIELD_ELECTRODE should be used to define shielding electrode for each touch electrode.
- User application may assign private data pointer to each control if TSS_USE_PRIVATE_CONTROL_DATA is set. If enabled, use the TSS_SetControlPrivateData() and TSS_GetControlPrivateData() functions to manipulate the data pointer.
- For GPIO-based measurement methods, the electrode voltage level the electrode is driven to in an idle state is logic high by default. It can be set to a low level by TSS_USE_DEFAULT_ELECTRODE_LEVEL_LOW macro.
- The on-fault callback function now contains a parameter which informs the application which electrode has caused the fault state. The callback is called for each electrode separately, not once per TSS_Task loop.
- TSS_GetSystemConfig(), TSS_GetKeypadConfig(), TSS_GetSliderConfig() and TSS_GetRotaryConfig() functions were implemented to standardize read access to control configuration registers.
- The TSS now provides the __TSS_VERSION__ macro which defines a version of TSS in unified format.
- Default electrode sensitivity was changed from 127 to 63.
- The following new macros may be used to reduce code size footprint by disabling selected features:
  - TSS_USE_DATA_CORRUPTION_CHECK macro enables or disables Data Corruption Check function
  - TSS_USE_TRIGGER_FUNCTION macro enables or disables the trigger feature and possibility to set registers related to triggering
  - TSS_USE_STUCK_KEY macro enables or disables the Stuck Key detection function
  - TSS_USE_NEGATIVE_BASELINE_DROP macro enables or disables the function for negative baseline drop for low signal cases.
- Some source code files were renamed and the code was overall refactored:
  - The TSS_USE_TRIGGER_SOURCE macro was renamed to TSS_USE_AUTOTRIGGER_SOURCE.
  - TSS_SystemConfig() function was renamed to TSS_SetSystemConfig()
  - TSS_KeypadConfig() function was renamed to TSS_SetKeypadConfig()
  - TSS_SliderConfig() function was renamed to TSS_SetSliderConfig()
  - TSS_RotaryConfig() function was renamed to TSS_SetRotaryConfig()
  - TSS_TSI_SCANC_EXTCHRG_LOW_LIMIT macro was renamed to TSS_TSI_EXTCHRG_LOW_LIMIT

- TSS_TSI_SCANC_EXTCHRG_HIGH_LIMIT macro was renamed to TSS_TSI_EXTCHRG_HIGH_LIMIT
- TSS_TSI_GENCS_PS_LOW_LIMIT macro was renamed to TSS_TSI_PS_LOW_LIMIT
- TSS_TSI_GENCS_PS_HIGH_LIMIT macro was renamed to TSS_TSI_PS_HIGH_LIMIT
- TSS_TSI_SCANC_AMCLKS macro was renamed to TSS_TSI_AMCLKS
- TSS_TSI_SCANC_AMPSC macro was renamed to TSS_TSI_AMPSC
- TSS_TSI_SCANC_AMCLKDIV macro was renamed to TSS_TSI_AMCLKDIV
- TSS_TSI_GENCS_LPCLKS macro was renamed to TSS_TSI_LPCLKS

- **Example Demo applications**
  - All existing demo applications were updated for new features of TSS 2.6.
  - ARM MDK (Keil) uVision 4 example applications were added.
- **FreeMASTER GUI**

  The TSS directly supports FreeMASTER GUI which enables to setup and analyse the TSS application. Each demo application contains its own FreeMASTER GUI project located in the *<example folder>/gui* folder. The FreeMASTER support files are stored in the *<install_dir>/gui* folder together with universal FreeMASTER GUI project.
- **System setup tool**

  The standalone System Setup Creator tool used in previous versions to generate the TSS header file is no longer available. The Processor Expert is a better and more robust alternative to this tool and is also well connected with information about the MCU device and electrode pins.
- **Processor Expert support**

  New version of Processor Expert TSS Component was added into the install package. The component may help to configure the TSS library in an easy to use graphical environment.
- **Example Demo applications**

  KWIKSTIK_DEMO

  KWIKSTIK_DEMO_MQX

  TWRKXX_DEMO_MQX

  TWRS08PTXX_DEMO

Version 3.0 (August 24th 2012)

- **TSS Library**
  - The TSS 3.0 release brings software library with unified API for ARM®Cortex™-M0, ARM®Cortex™-M4, ColdFire V1, ColdFire+ and HCS08 platforms and different capacitance measurement algorithms.
  - The pre-compiled libraries are available in lib/lib_cw, lib_cw_gcc, lib_iar directory and lib_uv4 for CodeWarrior development Studio v10.x, IAR Embedded Workbench and ARM-MDK uVision.

- New Auto Sensitivity Calibration (ASC) feature can be enabled by defining TSS_USE_AUTO_SENS_CALIBRATION. The ASC uses a signal noise analysis and past touch statistics to configure electrode sensitivity automatically. Sensitivity can still be defined by the user; the values will be set as initial sensitivity values for ASC function.
- The standard Keypad was extended for a possibility to define combinations of low level electrodes which will be reported as a single keypad control key. These combinations are defined in the form of array of 16 bit masks where bits set to one identify electrodes used. See more details for TSS_Cx_KEYS_GROUP settings in the documentation.
- A Proximity mode and proximity electrode can be enabled by ProximityEn bit. It can be combined with the Low Power function. The proximity mode uses a special configuration and may be used to detect hand approaching to electrode from a larger distance. Special electrode design is needed.
- Shielding electrodes and Water tolerant operation. With this new feature, the TSS is able to detect electrode touches even when electrodes are covered with water. Special electrode design is needed.
- Analog decoders implemented. The decoders are similar to standard Slider/Rotary ones but operate with fewer electrodes achieving a better touch position resolution. Position and range registers are extended to 8 bit range (range 0-255).
- 2D Matrix decoder has been added. It contains horizontal and vertical registers for position and range definition. The matrix decoder supports also a multi-touch in a limited form of gesture parameter reporting. To test the Matrix decoder use TWRPI-TOUCHPAD board.
- The simple low level routines specification was extended to all MCU's from HCS08 and ColdFire V1 family. The option is available for TSI and GPIO measurement method.
- The following GPIO-based measurement methods were removed and are no longer supported: KBI, PTI, TIC, CTS.
- Some source code files were renamed and the code was generally re-factored:
  - TSS_TSI_SCANC_DELVOL was replaced by TSS_TSI_DELVOL for TSS_TSI_VERSION 1 or TSS_TSI_DVOLT for TSS_TSI_VERSION 2
  - TSS_TSIL_CS2_DELVOL changed to TSS_TSIL_DVOLT
  - TSS_SAMPLE_ERROR_SMALL_AUTOTRG_PERIOD error status was changed to TSS_SAMPLE_ERROR_SMALL_TRG_PERIOD

- **Example Demo applications**
  - All existing demo applications were updated for new features of TSS 3.0.
  - The TWR-KL25Z48M, TWR-K15E256 and FRDM-KL25Z boards are now supported with the example applications.
  - TWR-KXX demo was extended to support new TWRPITSS-SLIDER2, TWRPI-KEYPAD2, TWRPITSS-ROTARY3, TWRPI-TOUCHPAD, TWRPI-SHIELD1 and TWRPI-SHIELD2 boards.

- **FreeMASTER GUI**

  The TSS directly supports FreeMASTER GUI which enables to setup and analyse the TSS application. Each demo application contains its own FreeMASTER GUI project located in

the *<example folder>/gui* folder. The FreeMASTER support files are stored in the *<install_dir>/gui* folder together with universal FreeMASTER GUI project.

Version 3.0.1 (September 21st 2012)

- **TSS Library**
  - The TSS library supports KL05 family.
  - Water Tolerance function was improved. The Shielding electrodes in Water Tolerance mode are not managed by the Automatic Sensitivity Calibration function anymore.
  - Fixed issues:
    - The TSI channel was incorrectly addressed in TSS_SensorTSIL.c.
    - If no electrodes is enabled then stuck could happen in TSS_SensorTSI.c.
- **Example Demo applications**
  - The TWRKL05 demo application for the TWR-KL05Z48 board was added.
  - The TWRKL25 demo can be now connected with FreeMASTER GUI via serial connection
  - The TWR-KXX demo was tuned for Water Tolerance function
  - The error message in FreeMASTER GUI for TSSEVB demo was removed
- **FreeMASTER GUI**

The TSS directly supports FreeMASTER GUI which enables to setup and analyse the TSS application. Each demo application contains its own FreeMASTER GUI project located in the *<example folder>/gui* folder. The FreeMASTER support files are stored in the *<install_dir>/gui* folder together with universal FreeMASTER GUI project.

Version 3.1.0 (August, 2013)

- The KL02, KL46 and KL26 family is supported now. TSI noise mode is supported and new AFID keydetector is available to increase robustness of touch detection in noisy environments. See more details in Chapter 3 “What is New” above.

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

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

Freescale reserves the right to make changes without further notice to any products herein. Freescale makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale 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 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 does not convey any license under its patent rights nor the rights of others. Freescale sells products pursuant to standard terms and conditions of sale, which can be found at the following address: freescale.com/SalesTermsandConditions.

Freescale, the Freescale logo, CodeTest, CodeWarrior, ColdFire, and Kinetis are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. ColdFire+, CoreNet, and Tower are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners. ARM, ARM Powered Logo, ARM Cortex-M0, and ARM Cortex-M4 are registered trademarks of ARM Limited.

© Freescale Semiconductor, Inc. 2013.

Document Number: TSSRN
Rev. 17
08/2013