AN8650.01

SX8650 WinCE® 6.0 Touch Screen Driver

Table of Contents

1 Introduction

The SX8650 WinCE™ 6.0 driver was developed to help users of the SX8650 touch screen controller to quickly use the device and to shorten software development time. This driver was implemented using the standard WinCE touchpad driver layer known as PDD, platform-dependant layer. The driver was developed and tested using a SX8650EVK with an Atmel AT91SAM9261-EK evaluation board.

2 Connections

The SX8650 device must be connected to the host processor running the touch screen driver. This driver was developed and tested using the AT91SAM9261-EK board with SX8650EVK.

The following connections are required:

- Power +3.3v, and ground from the Atmel board prototype area.
- SDA for I^2C™ data.
- SCL for I^2C™ clock.
- Pen-down interrupt signal NIRQ.

On the SX8650EVK board, connections are made to JP1 to provide the I^2C and interrupt signal. The four analog signals from the touch screen are made to J3 on the SX8650EVK. When using the SX8650EVK with touchscreen and microprocessor external to the EVK, remove all jumpers from JP1 and J3. For details on JP1 and J3, refer to the SX8650EVK user's guide.

On the AT91SAM9261-EK board, the included on-board touch controller ADS7843 is disconnected and replaced with the connections show in Figure 1. Refer to the AT91SAM9261-EK users guide for relevant documentation about connections to that board.

™ Windows is a registered trademark of Microsoft Corporation in the United States and other countries
™ I^2C is trademark of NXP Semiconductor

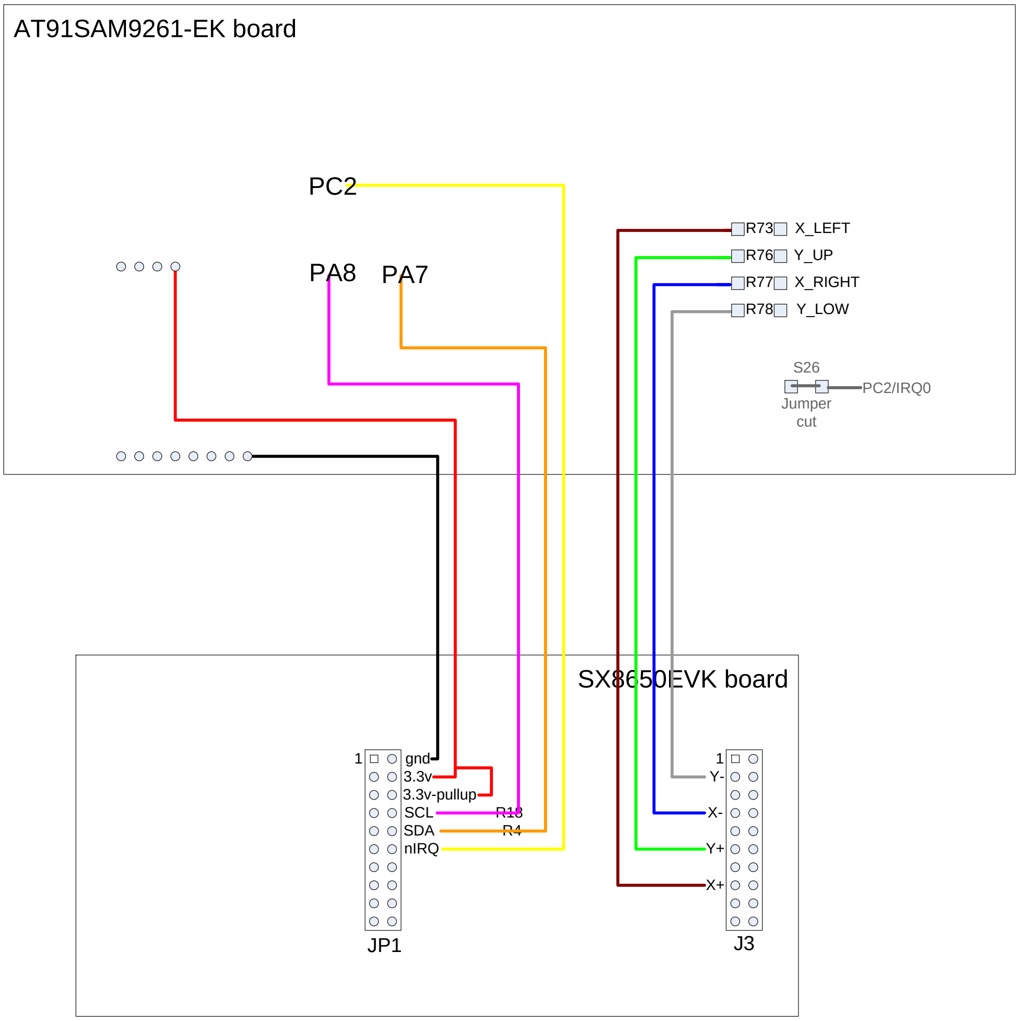

Figure 1 - SX8650 connection to AT91SAM9261-EK board

3 I²C interface

The I²C bus is the control and data bus through which the host processor sends commands and reads the analog values. The SX8650 driver uses the I²C bus using the standard WinCE stream interface. This partitions the processor-specific details of driving the I²C to the BSP for the microprocessor in use. The SX8650 driver communicates over the I²C using `DeviceIoControl()` calls to read and write to the SX8650 device. The SX8650 driver can be used on any WinCE BSP which implements the stream I/O interface to provide I²C communication.

For the hardware connection, the I²C signals are connected to PA8 = SCL and PA7 = SDA on the AT91SAM9261-EK. PA8 is available on J16-45, and PA7 is available on J16-44. Verify the pull-up resistor on the I²C bus: R18 and R4 on the SX8650EVK should be approximately 2.2kohm for sufficiently fast rise time on the I²C bus.

The I²C driver software (source file SX8650I2C.c) opens the I²C peripheral on the AT91SAM9261 using a call to `CreateFile()` with the filename of “I2C1:”. All I²C communication takes place over the standard stream interface of `DeviceIoControl()` calls.

4 Touch Device Driver

On the AT91SAM9261-EK board, the NIRQ signal is connected to PC2/IRQ0 pin available on J16-73. The on-board ADS7843 is disconnected by cutting the S26 jumper.

The device driver is implemented in the file SX8650Touch.cpp.

The SX8650 is used in `PENTRG` mode, meaning that the NIRQ signal will be inactive when no pen is on the touch screen. When the SX8650 device detects the pen is down on the screen, the device will first take samples, and then assert the NIRQ pin after the samples are ready to be read by the driver software. After the samples have been read from the device, the NIRQ will de-assert. If the pen continues to be held down on the screen, the SX8650 device will again start a conversion cycle and assert the NIRQ when new samples are ready. The rate at which NIRQ will become asserted, is determined by the `POWDLY` setting multiplied by the number of channels selected to be converted in the channel mask register. See SX8650 datasheet for the rate formula. WinCE 6.0 only cares about X and Y, so only acquisitions on these two channels are made on every cycle. If the operating system supports touch pressure in the future, the Z1 and Z2 channels can be enabled to report pressure to the operating system.

To initialize the driver, WinCE calls the `DdsiTouchPanelEnable()` function. The SX8650 is reset and initialized for `PENTRG` operation, and the channel mask is set to convert X and Y channels. `KernelIoControl()` is called to cause the assertion of NIRQ to call `DdsiTouchPanelGetPoint()` function. If the NIRQ pin is asserted, the analog values will then be read from the SX8650 device and reported to the operating system.

The SX8650 device does not uniquely report the pen-up condition. Instead, the software driver sets the variable `gdwIstTimeout` to detect the pen-up condition. If the pen has been lifted, a timeout will occur. This timeout condition can be detected when `DdsiTouchPanelGetPoint()` is called with NIRQ (PC2) de-asserted.
For further understand of the timeout behavior, refer to the `TouchPanelpISR()` function in `tchmain.c` in the WinCE operating system.

Often when the pen is lifted from the screen, the very last sample will contain error due to insufficient pressure upon the screen while the sample is being taken. In order to prevent wild cursor movement during pen lifting, the driver compares each sample with the previous. If the change of value is too large compared to previous, the driver will tell WinCE to ignore this sample, and instead use the previous sample when reporting the pen up condition.

The BSP for AT91SAM9261-EK has the behavior of calling twice `DdsiTouchPanelGetPoint()` every time the NIRQ signal is asserted. The flag `bIgnoreCall` is used to ignore the 2nd spurious call, and simply return the previous coordinates. When using this driver on another platform, it is recommended to verify this behavior on your BSP.

The only platform-dependant portion of SX8650Touch.cpp is the handling of the NIRQ signal, in this case PC2 on the AT91SAM9261 microprocessor. The driver can work on any platform by replacing these three items:

- The call to `pio_get_value()` to read the NIRQ state.
- The pin assignment `dwLogIntr` passed to `KernelIoCtrl()` to enable interrupt from the NIRQ pin.
- The included files `AT91SAM9261_oal_intr.h`, `at91sam9261_gpio.h`, and `atmel_gpio.h` to support the above two items.

4.1 Interrupt rate considerations

The interrupt rate generated by the SX8650 is the `POWDLY` settings multiplied by the number of channels enabled in the channel mask register.

Although the primary intention of `POWDLY` is to accommodate the R/C time constant of the resistive touch screen, additional delay may be required according to the ability of the microprocessor to handle the interrupt rate while still providing adequate resources to application software. When increasing the interrupt rate of the SX8650, the developer must consider the CPU time used by the I^2C driver, graphics subsystem and application software which must run while the pen is down on the touch screen.

The pen-trigger mode of the SX8650 permits faster interrupt rate due to the only I^2C activity on each interrupt is reading the samples.

5 Using the driver with the Adeneo BSP

Using the SX8650 driver doesn't require any modification to the BSP source code, only the ADS7843 driver is replaced.

5.1 Platform.bib

The touch screen driver is implemented as a DLL, meaning the driver can be replaced by editing platform.bib.

The platform.bib is changed by commenting the included touch driver, and replacing with the sx8650 touch driver:

```
; -----------------------------------------------------------------------
IF BSP_AT91SAM9261EK_TOUCHSCREEN
;     at91sam9261ek_touchscreen.dll
$(_FLATRELEASEDIR)\at91sam9261ek_touchscreen.dll        NK  SHK
      sx8650_touchscreen.dll  $(_FLATRELEASEDIR)\sx8650_touchscreen.dll           NK
SHK
ENDIF BSP_AT91SAM9261EK_TOUCHSCREEN
; -----------------------------------------------------------------------
```

5.2 Registry Setting

The "DriverName" setting must be changed, in the file Touchscreen.reg found in the directory:
%_TARGETPLATROOT%\SRC\DRIVERS\Touchscreen\Dll\

```
[HKEY_LOCAL_MACHINE\HARDWARE\DEVICEMAP\TOUCH]
      "DriverName"="sx8650_touchscreen.dll"
      "MaxCalError"=dword:10
```

5.3 Add SX8650 Source Code Files

The driver source code files must be copied into the touchscreen driver directory of the BSP. Copy the following files into `TARGETPLATROOT%\SRC\DRIVERS\Touchscreen\Dll\`

- `SX8650Touch.cpp`: the touch screen driver
- `SX8650I2C.c`: implements communicating with SX8650 device over the I^{2}C stream driver
- Sources: contains build instructions for the above two files

5.4 Compiling the Driver with the BSP

The driver may now be compiled by following the instructions in the Adeneo BSP install notes.

6 References

- [1] SX8650 datasheet

- [2] AT91SAM9261-EK board
http://atmel.com/dyn/products/tools_card.asp?tool_id=3820

- [3] Windows on AT91
http://www.at91.com/windows4sam/bin/view/Windows4SAM/WebHome

The information presented in this document does not form part of any quotation or contract, is believed to be accurate and reliable and may be changed without notice. No liability will be accepted by the publisher for any consequence of its use. Publication thereof does not convey nor imply any license under patent or other industrial or intellectual property rights. Semtech assumes no responsibility or liability whatsoever for any failure or unexpected operation resulting from misuse, neglect improper installation, repair or improper handling or unusual physical or electrical stress including, but not limited to, exposure to parameters beyond the specified maximum ratings or operation outside the specified range.

SEMTECH PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS. INCLUSION OF SEMTECH PRODUCTS IN SUCH APPLICATIONS IS UNDERSTOOD TO BE UNDERTAKEN SOLELY AT THE CUSTOMER'S OWN RISK. Should a customer purchase or use Semtech products for any such unauthorized application, the customer shall indemnify and hold Semtech and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs damages and attorney fees which could arise.

Contact Information

Semtech Corporation
Advanced Communications and Sensing Products Division
200 Flynn Road, Camarillo, CA 93012
Phone (805) 498-2111 Fax : (805) 498-3804