

Application Note

Using the NXP ADC with Resistive Touch Screens

INTRODUCTION

A resistive touch screen is a sensor that translates the physical location of a touch at a point (X,Y) in a rectangular area into a voltage that represents the X value and a voltage that represents the Y value. Many LCD modules come equipped with resistive touch screens. Such screens can use 4 wires, 5 wires, 7 wires, or 8 wires for generating screen bias voltages and reading back the touch point voltage.

In the past, reading touch coordinates from a resistive touch screen into a microcontroller required a dedicated touch screen controller chip or a complex network of external switches wired to the microcontroller's on-chip analog-to-digital converter (ADC). More recently, NXP microcontrollers such as the LH75400/01/10/11 family, LH79524/25 family and the LH7A404 have an on-chip ADC with built-in touch screen biasing circuitry that permits a glue-less interface between the touch screen sensor and the microcontroller. The NXP ADC controls all touch screen bias voltages and records all measurements without CPU intervention.

For each touch screen interface — 4-wire, 5-wire,

7-wire, or 8-wire — this Application Note describes how to wire the touch screen to the ADC, how to program the ADC, and how to design the system software to interact with the ADC.

This note begins by reviewing the fundamentals of how the ADC works with different touch screen sensors to produce the desired X-Y coordinate reading. Experienced users will find this especially useful when using the ADC in a manner not described in this note. For setting and algorithm information only, skip the next few sections, and go straight to the Hookup and Programming section.

ADC PRINCIPLES

NXP's ADC uses a Successive Approximation Register (SAR) type converter.

SAR ARCHITECTURE

While there are various SAR implementations, the basic architecture is simple. Figure 1 shows this architecture.

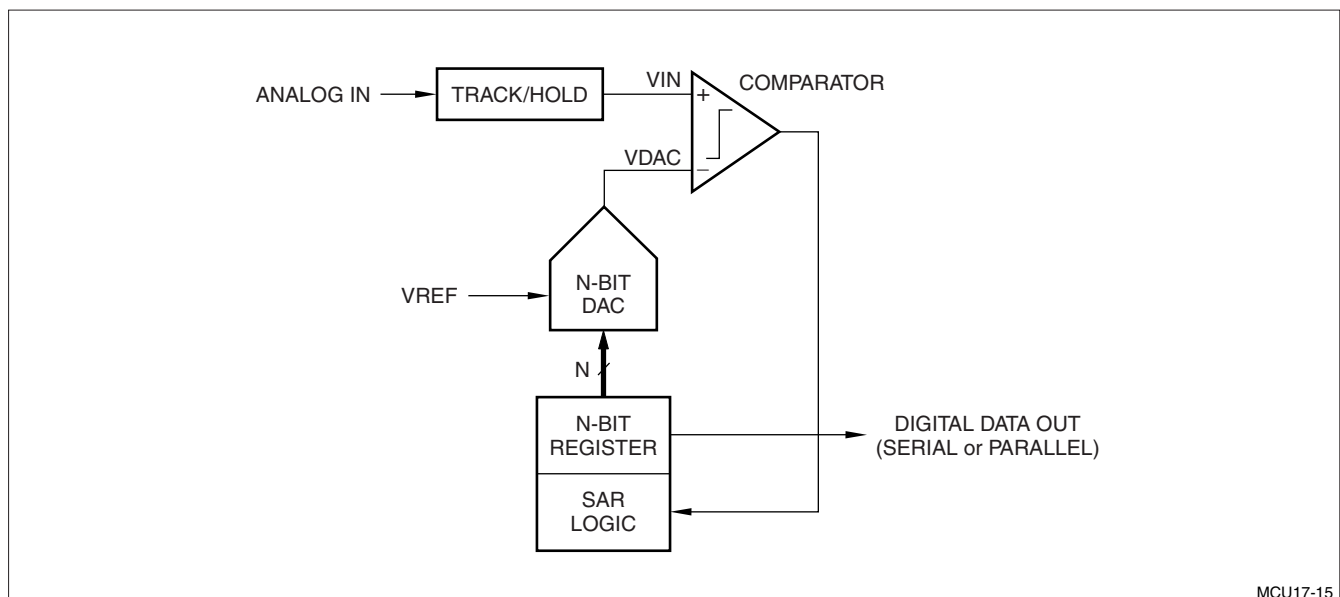


Figure 1. Simplified N-bit SAR Architecture

MCU17-15

The analog input voltage (V_{IN}) is held on a track/hold. The N-bit register is set to mid-scale (100...0, where the most-significant bit is set to 1) to implement the binary search algorithm. This forces the DAC output (V_{DAC}) to be $V_{REF} \div 2$, where V_{REF} is the reference voltage provided to the ADC. Then a comparison is performed to determine whether V_{IN} is less than or greater than V_{DAC} :

- If V_{IN} is less than V_{DAC} , the comparator output is a logic LOW and the most-significant bit of the N-bit register is cleared to 0.
- If V_{IN} is greater than V_{DAC} , the comparator output is a logic HIGH (or 1) and the most-significant bit of the N-bit register remains set to 1.

The SAR control logic then moves to the next bit down, forces that bit HIGH, and conducts another comparison. The SAR control logic repeats this sequence until it reaches the least-significant bit. When the conversion is complete, the N-bit digital word is available in the register.

Figure 2 shows an example of a 4-bit conversion. In this figure, the Y-axis and the bold line show the DAC output voltage.

In this example:

1. The first comparison shows that $V_{IN} < V_{DAC}$. Consequently, bit [3] is set to 0. The DAC is then set to 0b0100 and the second comparison is conducted.
2. In the second comparison, $V_{IN} > V_{DAC}$, so bit [2] remains at 1. The DAC is then set to 0b0110 and the third comparison is conducted.
3. In the third comparison, bit [1] is set to 0 and the DAC is then set to 0b0101 for the last comparison.
4. In the final comparison, bit [0] remains at 1 because $V_{IN} > V_{DAC}$.

Four comparison periods are necessary for a 4-bit ADC. Generally, an N-bit SAR ADC requires N comparison periods and will not be ready for the next conversion until the current conversion is completed. This explains why the ADC is power- and space-efficient.

It is very important that the DAC portion of the NXP ADC accepts both a positive and negative external reference.

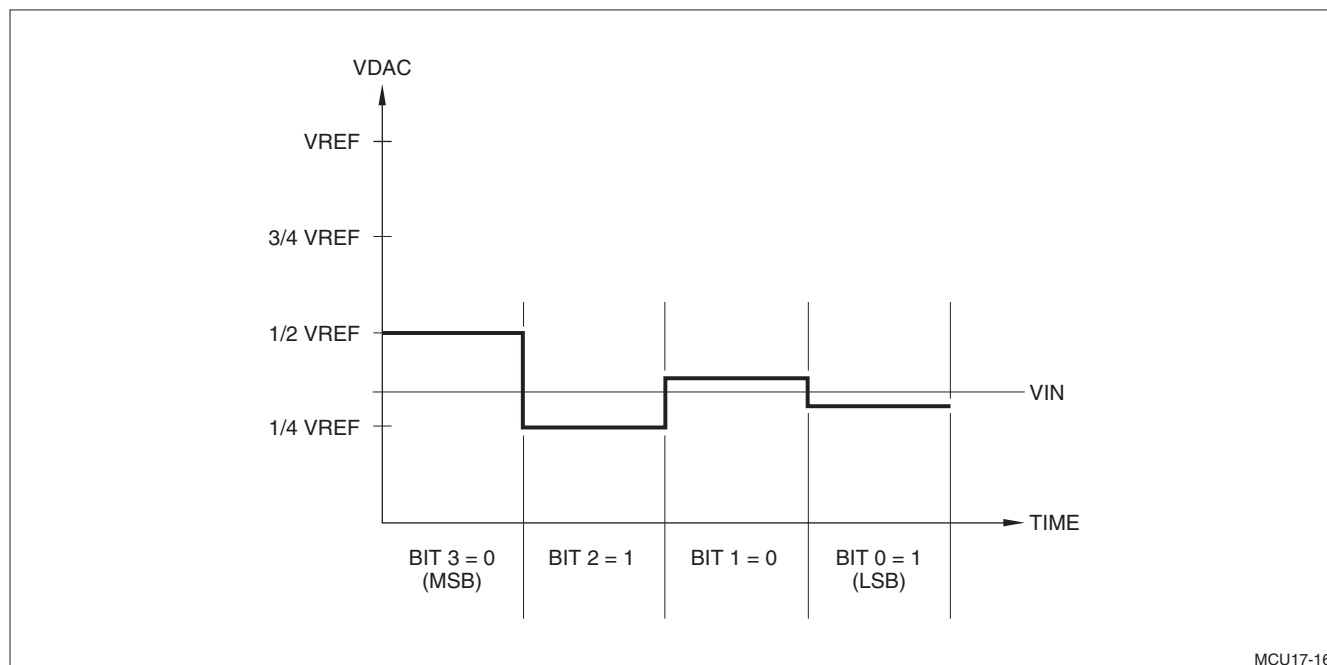


Figure 2. Example of a 4-bit SAR ADC Operation

MCU17-16

TOUCH SCREEN PRINCIPLES

A touch screen is made out of two transparent layers, one on top of the other. 4-wire and 8-wire touch screens are made out of two layers of a transparent resistive material that have a uniform surface resistance. 5-wire and 7-wire touch screens are made with one resistive layer and one conductive layer. Normally, a resilient material is used to keep the two layers separated. When enough pressure is applied to the surface of a touch screen (for example, with a stylus or your finger), the top layer is brought into contact with the bottom layer.

All resistive touch screens use the voltage divider principle to generate voltages that represent X and Y coordinates. See Figure 3.

A voltage divider is created by connecting two resistors in series. The top resistor (R1 in Figure 3) is connected to a positive reference voltage (V_{REF}) and the bottom resistor (R2 in Figure 3) is connected to ground. The voltage measured at the point where the two resistors meet is directly proportional to the value of the bottom resistor.

To measure a coordinate in a particular direction on a resistive touch screen, bias a resistive layer by applying V_{REF} to one edge of the resistive layer and connecting the other edge of that layer to ground. Connect the unbiased layer to the high impedance input of an ADC. When pressure on the touch screen is enough to bring the two layers into contact, the resistive surface is divided into two resistors. The resistance value is directly proportional to the distance from the touch point to the biased edge. The resistance between the touch point to the edge that is wired to ground is similar to the bottom resistor in the voltage divider. Therefore, the voltage measured on the unbiased layer is directly proportional to the distance between the touch point and the edge that is wired to ground.

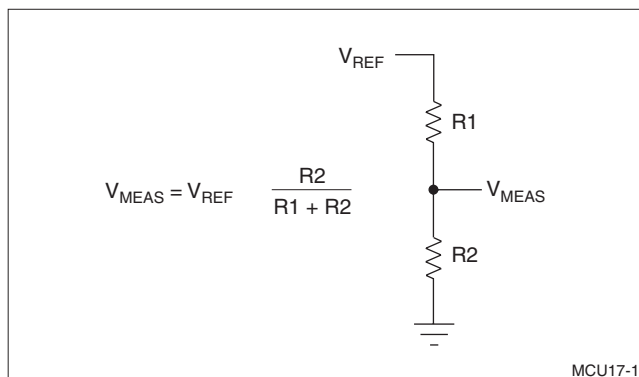


Figure 3. Equivalent Voltage Divider Circuit

4-Wire Touch Screens

To make a 4-wire touch screen, two resistive layers are used. One layer has a vertical bus bar at the left edge of the screen and a second vertical bus bar along the right edge of the screen. The other layer has a horizontal bus bar at the bottom of the screen and a second horizontal bus bar at the top of the screen. See Figure 4.

To measure along the X axis, bias the left bus bar to 0 V and bias the right bus bar to V_{REF} . Connect either the top or the bottom bus bar to the ADC, and make a measurement when the top layer is in contact with the bottom layer.

To measure along the Y axis, bias the top bus bar to V_{REF} and the bottom bus bar to 0 V (or the other way around for raster Y coordinates). Connect the ADC input to either the left or right bus bar, and measure the voltage when the top layer is in contact with the bottom layer.

Figure 5 shows a simplified model of the 4-wire touch screen when the two layers are in contact. For the best results with a 4-wire touch screen, connect the bus bar set to V_{REF} to the positive reference input of the ADC and connect the bus bar set to 0 V to the negative reference input of the ADC.

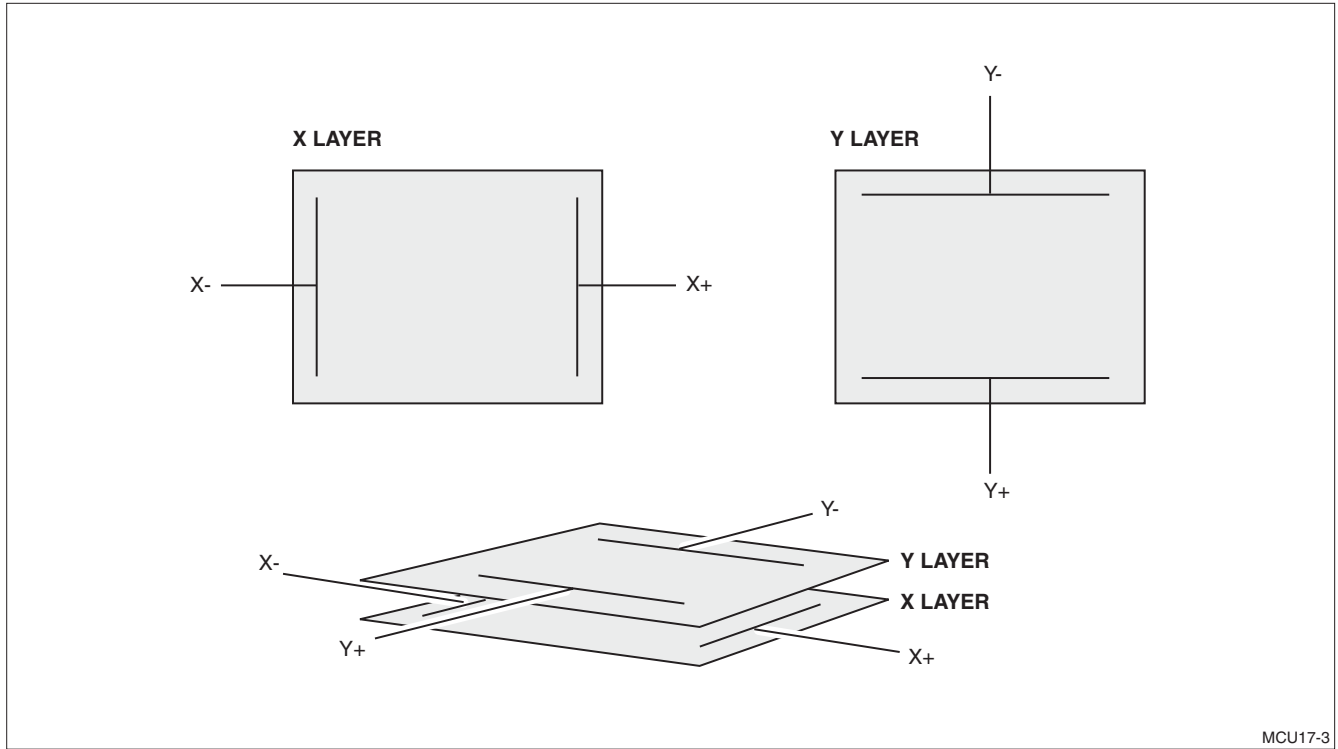
5-Wire Touch Screens

To make a 5-wire touch screen, a resistive layer and a conductive layer are used. The conductive layer has a contact, usually along one edge. The resistive layer has a contact point at each corner.

To measure along the X axis, bias the upper left corner and lower left corner to V_{REF} and the upper right corner and lower right to ground. Because left and right corners are at the same voltage, the effect is about the same as attaching bus bars along the left and right edges, similar to the method used with the 4-wire touch screen. See Figure 6.

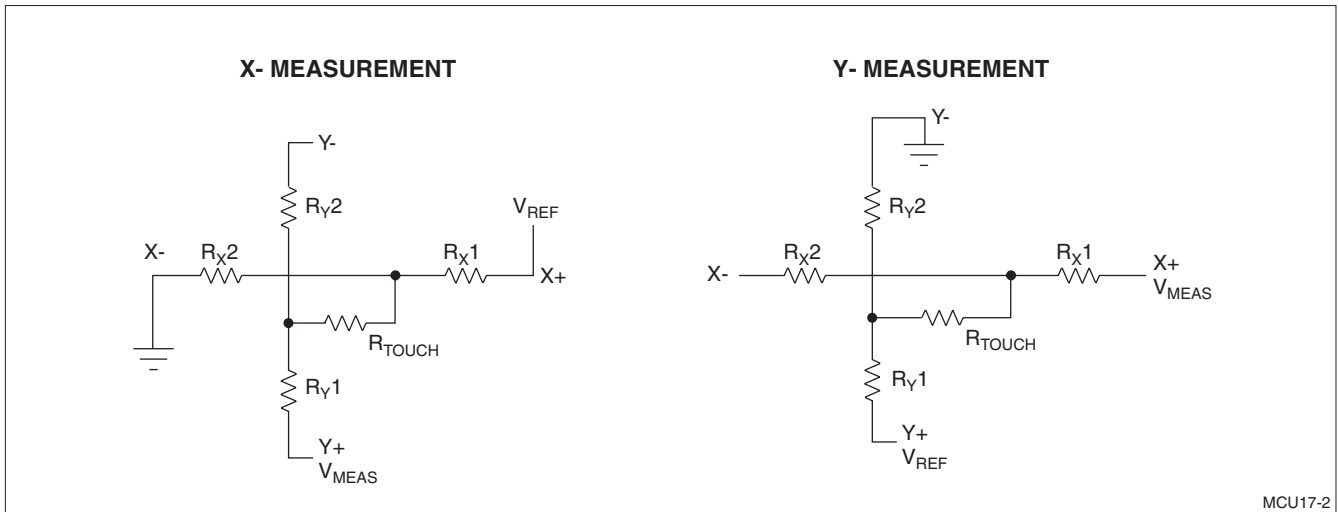
To measure along the Y axis, bias the upper left corner and upper right corner to V_{REF} and bias the lower left corner and lower right corner to 0 V. Because upper and lower corners are at the same voltage, the effect is about the same as attaching bus bars along the top and bottom edges, similar to the method used with the 4-wire touch screen. This measurement algorithm is nice because it makes the upper left and lower right voltages invariant; however, if using raster coordinates, it makes the X and Y directions backwards.

Figure 7 shows a simplified model of the 5-wire touch screen when the two layers are in contact. For best results, connect the upper left corner (biased to V_{REF}) of the 5-wire touch screen to the positive reference of the ADC and connect the lower-left corner of the 5-wire touch screen (biased to 0 V) to the negative reference of the ADC.



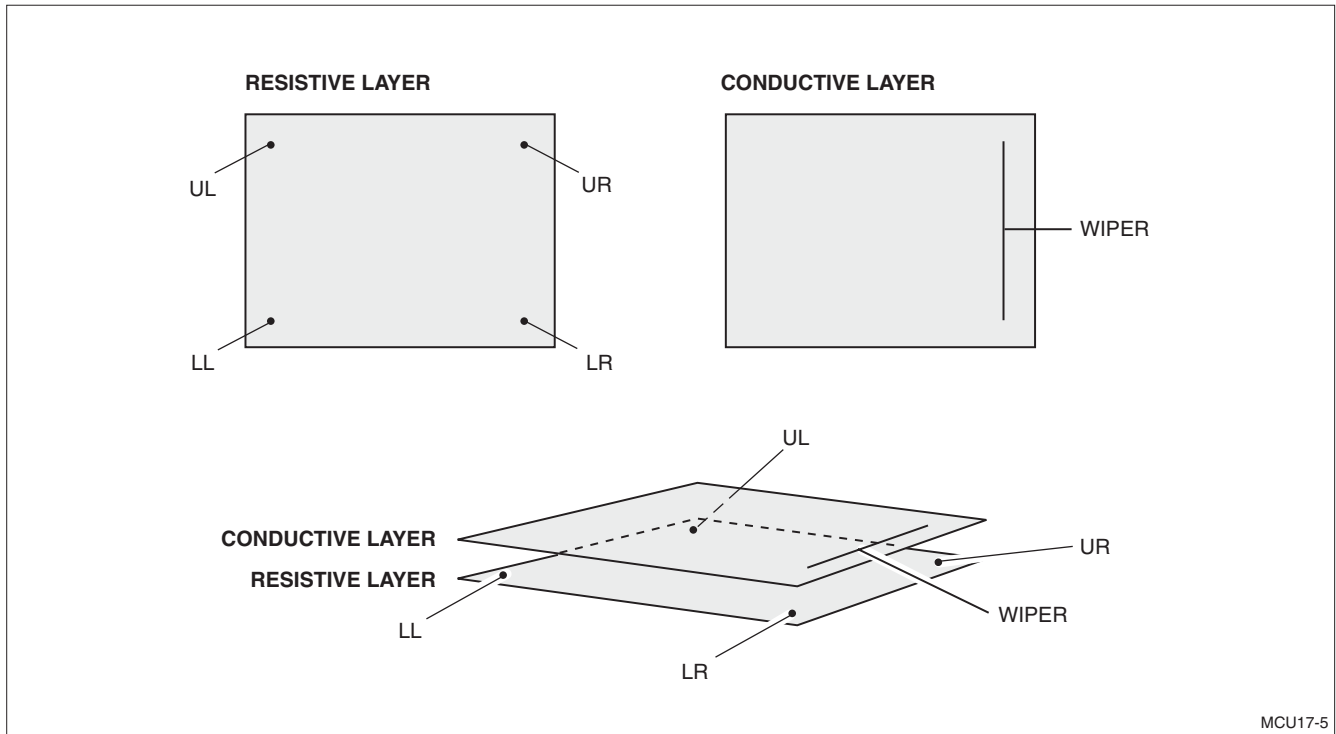
MCU17-3

Figure 4. 4-wire Touch Screen



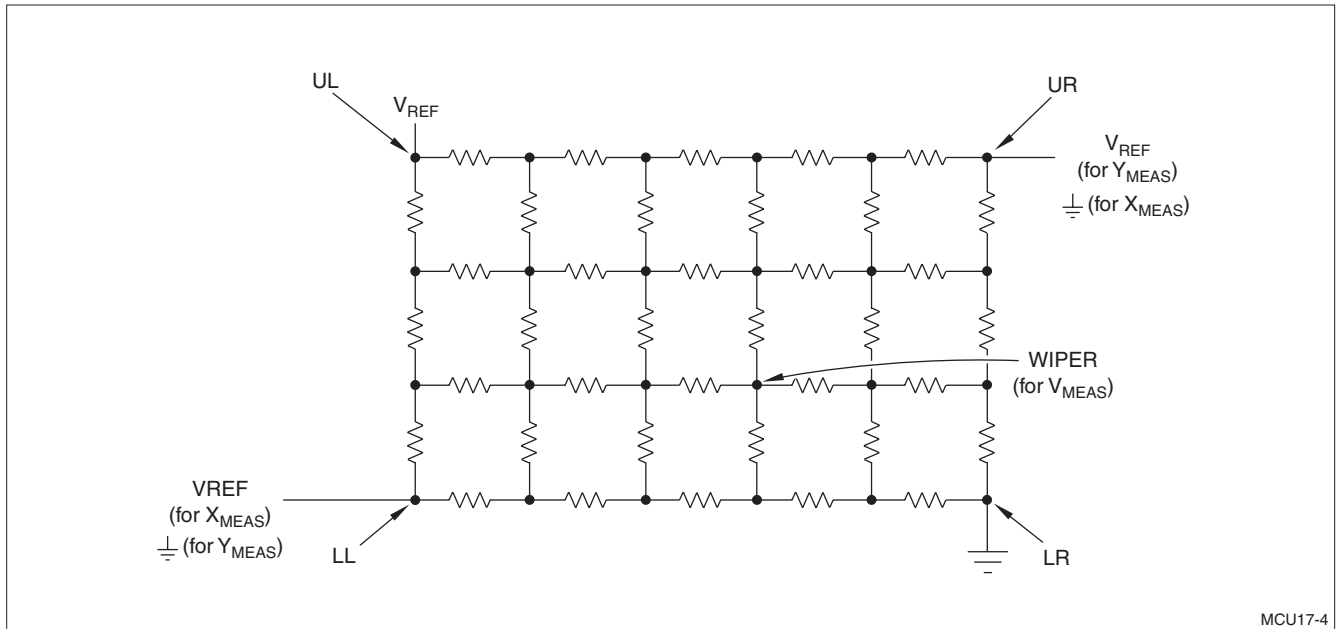
MCU17-2

Figure 5. 4-wire Touch Screen Equivalent Circuit



MCU17-5

Figure 6. 5-wire Touch Screen



MCU17-4

Figure 7. 5-wire Touch Screen Equivalent Circuit

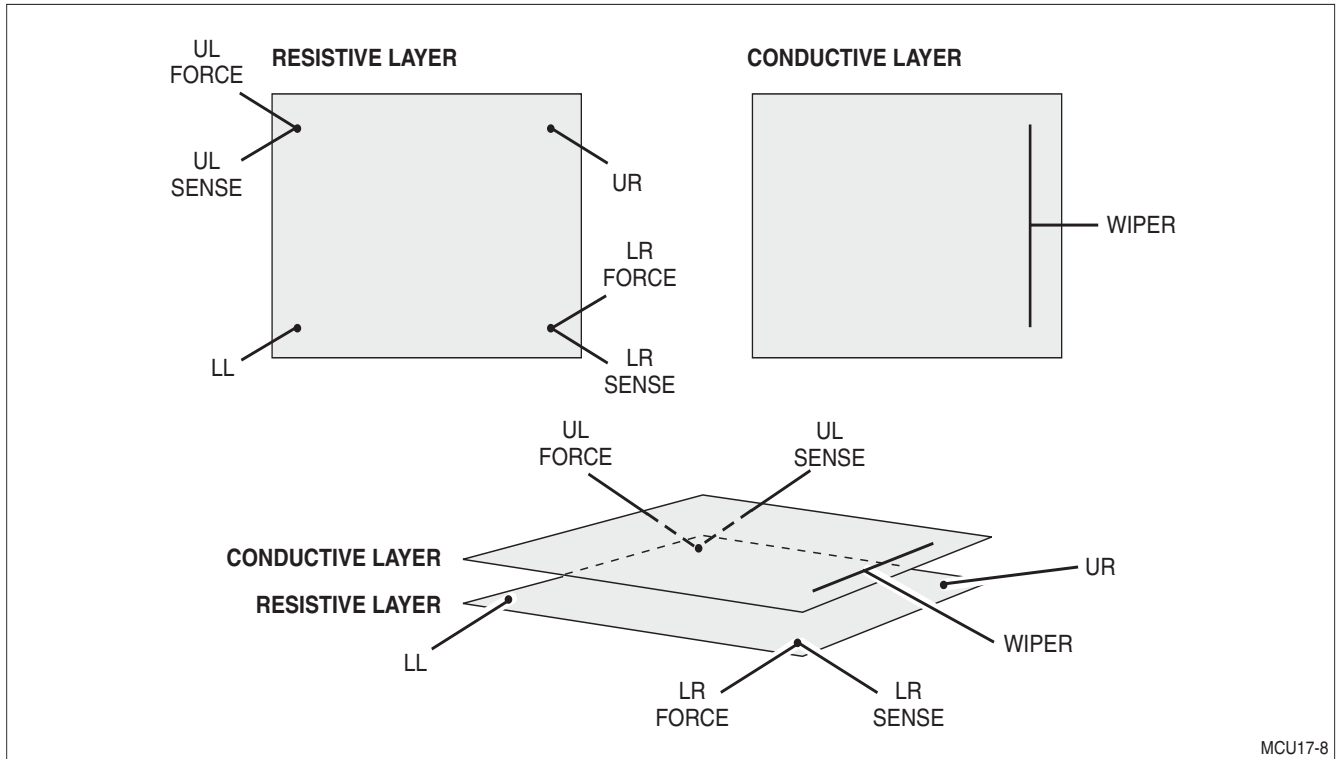
7-Wire Touch Screens

A 7-wire touch screen is made the same way as a 5-wire touch screen, except a second wire is added to the upper left and lower right corners. See Figure 8.

During a screen measurement, apply V_{REF} to the upper left corner on one wire and attach the other wire to the positive reference of the SAR ADC's DAC. During a

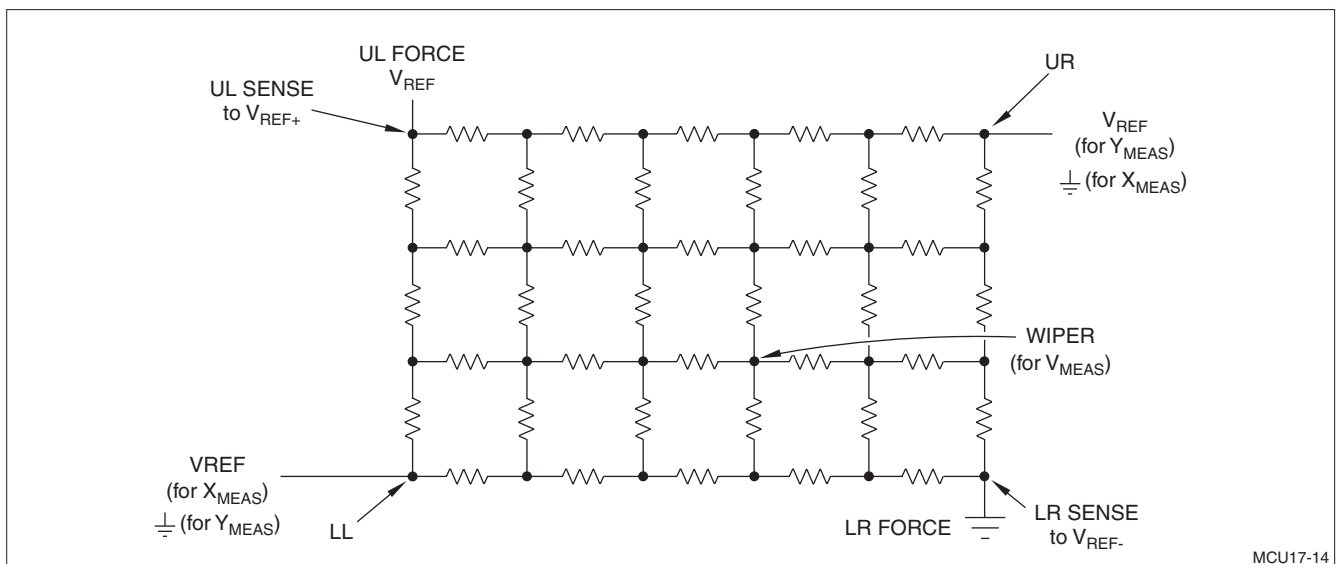
screen measurement, apply 0 V to the lower right corner on one wire and attach the other wire to the negative reference of the SAR ADC's DAC. The conductive layer is still used to measure the voltage from the voltage divider.

Figure 9 shows a simplified model of the 7-wire touch screen when the two layers are in contact.



MCU17-8

Figure 8. 7-wire Touch Screen



MCU17-14

Figure 9. 7-wire Touch Screen Equivalent Circuit

8-Wire Touch Screens

An 8-wire touch screen is made the same way as a 4-wire touch screen, except a second wire is added to each bus bar. See Figure 10. For the V_{REF} bus bar, apply V_{REF} on one wire and use other wire as the positive reference for the SAR ADC's digital-to-analog converter (DAC). For the 0 V bus bar, apply 0 V on one wire

and use the other wire as the negative reference for the SAR ADC's DAC. Any of the four wires on the unbiased layer can be used to measure the voltage from the voltage divider.

Figure 11 shows a simplified model of the 8-wire touch screen when the two layers are in contact.

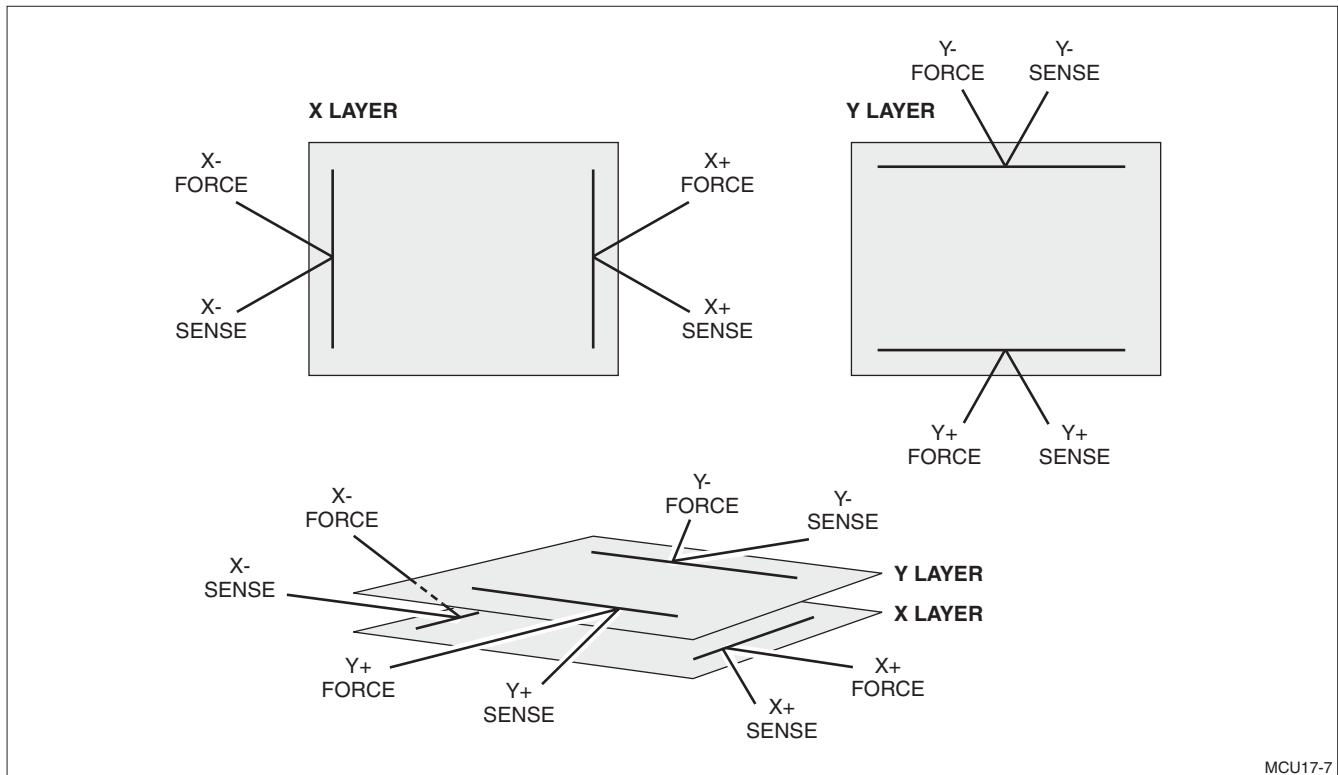


Figure 10. 8-wire Touch Screen

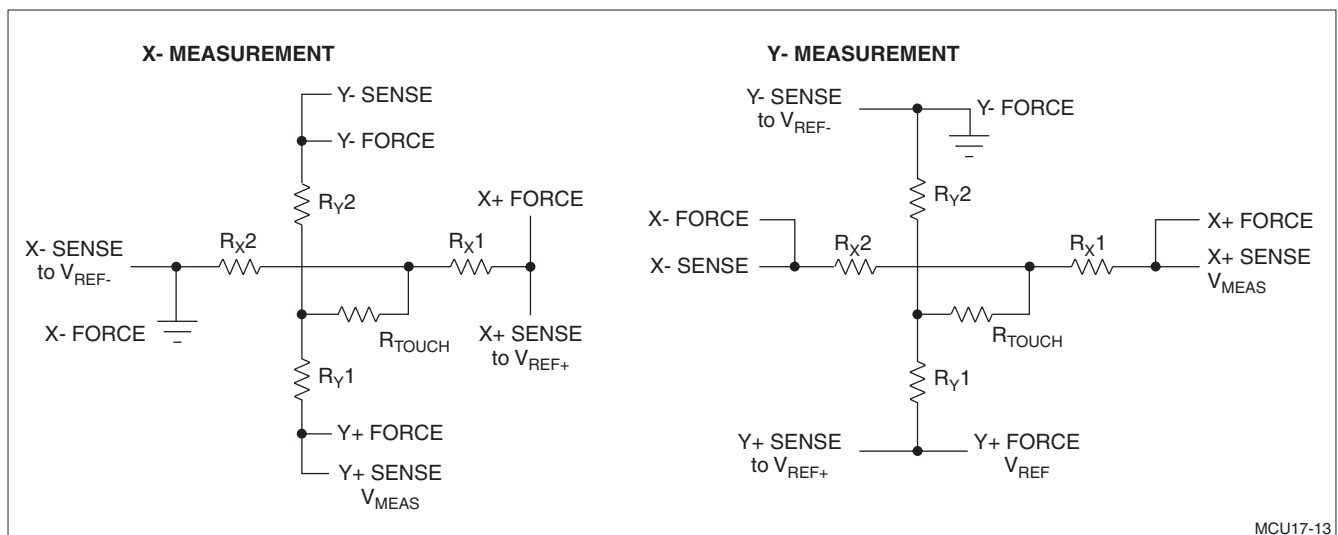


Figure 11. 8-wire Touch Screen Equivalent Circuit

SENSING TOUCH AND NO-TOUCH CONDITIONS

For all touch screens, you can sense touch and no-touch conditions by pulling one layer up with a weak pull-up resistor and pulling the other layer down with a strong pull-down transistor. If the voltage measurement of the pulled-up layer is above some logic threshold voltage, then there is no touch; otherwise, there is a touch. A complication with this method is that the touch screen is a giant capacitor. As well, you may need to add capacitance to the touch screen wires to filter out LCD-induced noise. A weak pull-up connected to a large capacitor yields a long rise time that could result in false touch detection. A work-around for this problem is described in the Basic ADC Setup section.

For 4-wire and 8-wire touch screens, measure the contact resistance, shown as R_{TOUCH} in Figure 5 and Figure 11. R_{TOUCH} is directly proportional (more or less) to the touch pressure.

To measure touch pressure, you will need to know the resistance of either one or both layers of the touch screen. The formulas shown in Figure 12 show how to make the calculations. However, be aware that if the Z1 measurement is close to or equal to 0 (this happens when the touch point is near the X bus bar that is tied to ground during the measurement), your calculation will have numerical problems. Most applications will be better served by using the weak pull-up method.

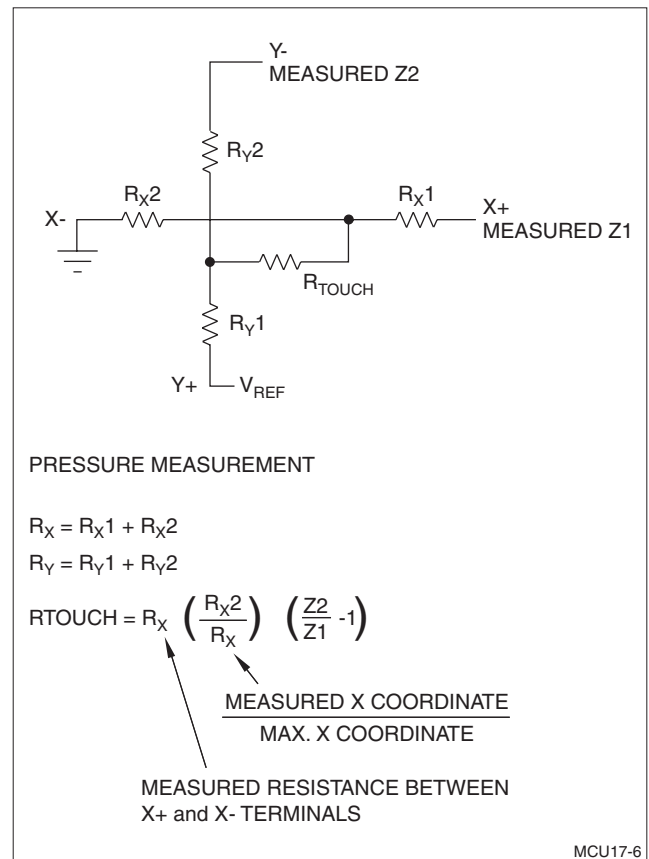


Figure 12. 4-wire Touch Screen Pressure Measurement

USING THE NXP ADC

The NXP ADC module consists of the ADC core, a measurement result FIFO, a positive input multiplexer, a negative input multiplexer, a positive reference multiplexer, a negative reference multiplexer, a measurement sequencer, and a bias and control network.

Except for the LH7A404, all ADC module I/O pins pass through the GPI MUX. This multiplexer allows you to configure each pin as either a general purpose digital input or an analog pin. When configuring an ADC module I/O pin as a digital input, the GPI MUX attaches a digital input buffer to the pin. When configuring an ADC module I/O pin as an analog pin, the GPI MUX disconnects the pin's digital input buffer from the pin and connects the buffer to ground. To minimize leakage current in low-power applications, either wire all unused ADC I/O pins to ground or program the unused ADC module I/O pins to be analog pins. The remainder of this Application Note assumes that you have programmed all the required ADC module I/O pins as analog pins.

On the LH75400/01/10/11 family, the ADC positive input multiplexer can sample any of the eight ADC pins: AN0-AN4, AN6, AN8, and AN9. On the LH7A404, the ADC positive input multiplexer can sample any of the ten ADC pins. To save pins on the LH75400/01/10/11, NXP only brought out eight of the ten pins. This explains the gaps in the analog pin nomenclature on the LH75400/01/10/11. On all chips, the negative input multiplexer can select either the negative reference input or VSSA_ADC as the negative input for the ADC. On all chips, the positive reference multiplexer can select either the on-chip reference or one of three external pins as the positive reference. On all chips, the negative reference multiplexer can select either VSSA_ADC or one of three other external pins as the negative reference. These multiplexers feed the analog inputs to the SAR ADC core. See Figure 13 for the ADC block diagram. Note that the LH7A404 has two additional analog inputs to the 11 to 1 MUX.

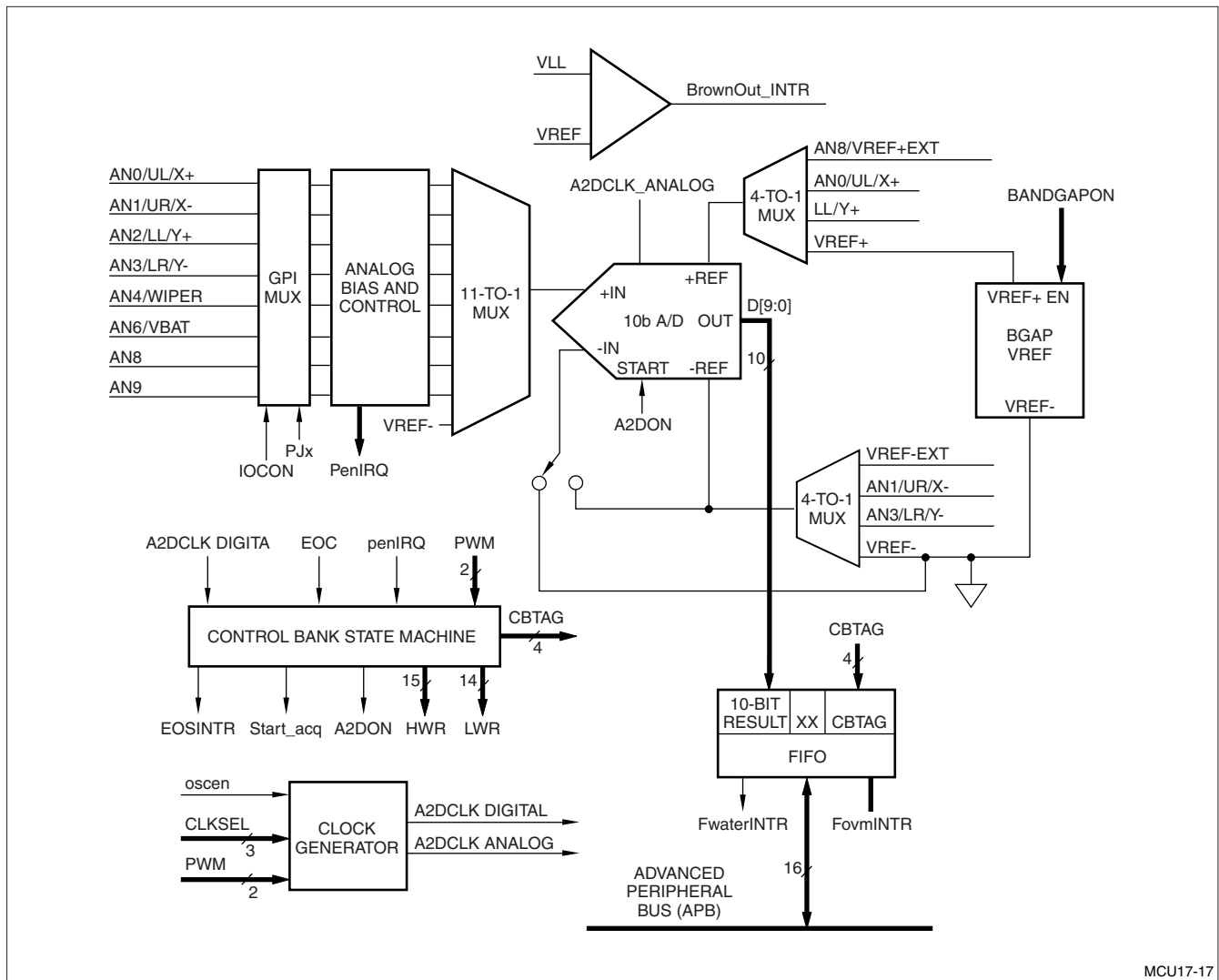
The ADC bias and control network can switch many of the analog pins to VDDA_ADC or VSSA_ADC via low resistance analog switches. In addition, the bias and control network can switch weak pull-up resistors onto the AN0 and AN4 pins. Figure 14 shows the bias and control network. Note that the LH7A404 has two additional analog inputs to the 11-to-1 MUX.

The measurement sequencer controls when a series of measurements start, which analog switches are activated to bias the touch screen, when these switches are activated, which pins are used as analog inputs to the ADC core, and how long the sampling interval is. The measurement sequencer is very powerful, but it is also somewhat complex. For measurement sequencer settings information, skip the next sections, and go straight to the configuration section for the appropriate touch screen.

The A2DCLK clock signal determines the time base for the sequencer and the ADC core. The A2DCLK frequency is programmed in the Power Configuration Register. See the 'Optimizing the System' section for the tradeoffs in choosing the A2DCLK.

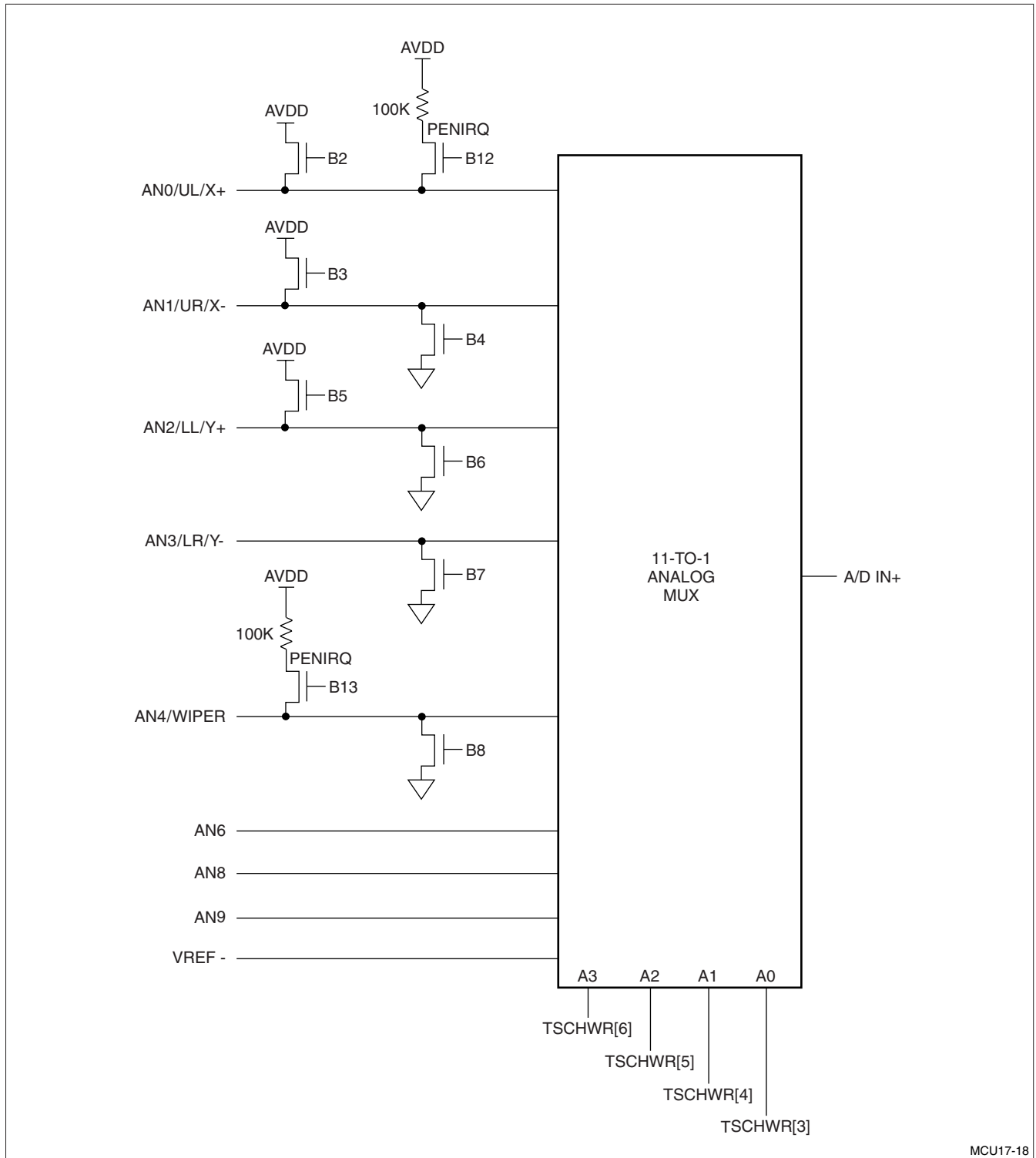
The measurement sequencer is a state machine that sends information to the bias and control block, the multiplexers, and the analog core. The information the sequencer sends is called a Control Word. This Control Word is 32 bits long. Because the ADC is designed to operate on a 16-bit on-chip data bus, the Control Word is broken into the Control High Word Register and the Control Low Word Register. To see the current state of the signals that the sequencer is feeding to the rest of the ADC, have your program read the Control High Word Register and Control Low Word Register.

The Control High Word Register contains the current settings for the positive and negative input multiplexers, the positive reference multiplexer, and the measurement settling time. The Control Low Word Register contains the current state of the negative reference multiplexer and the Bias and Control Network's control bits. The settings for the Control Word Registers are shown in Table 1 through Table 5.



MCU17-17

Figure 13. LH75400/01/10/11 ADC Block Diagram



MCU17-18

Figure 14. LH75400/01/10/11 Bias and Control Network

Table 1. HW Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	SETTIME									INP			INM	RefP		
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
ADDR	0xFFFC3000 + 0x00															

Table 2. HW Register Definitions

BITS	NAME	DESCRIPTION
31:16	///	Reserved Read as zero
15:7	SETTIME	Number of Clock Cycles Number of clock cycles that the ADC allows for the input signal to settle to within required accuracy before beginning conversion. Used with bits [10:8] of the Power Control Register to set the acquire time in clock cycles. For example, Frequency In (f_{IN}) 2 MHz (500 ns period) PC[10:8] = 010 (i.e., divide f_{IN} by 4) HW[15:6] = 000100000 (i.e., 32 cycles) Therefore, acquire time is $500 \text{ ns} \times 4 \times 32 = 64 \mu\text{s}$
6:3	INP	In+ Mux Determines the signal connected to the positive input of the ADC. See Table 3.
2	INM	In- Mux Determines the signal connected to the negative input of the ADC. 0 = Ref- (output of the Ref- Mux) 1 = GND
1:0	RefP	Ref+ Mux Determines the signal connected to the positive reference of the ADC. 00 = VREF+ (positive terminal of the internal bandgap reference) 01 = AN0 (UL/X+) 10 = AN2 (LL/Y+) 11 = AN8

NOTE: While the examples in Table 1 and Table 2 are specific to LH75400/01/10/11, minor differences will exist. For those other parts, use this information as a reference.

Table 3. In + Mux Definition

IN+	BIT6	BIT5	BIT4	BIT3
AN0 (UL/X+)	0	0	0	0
AN1 (UR/X-)	0	0	0	1
AN2 (LL/Y+)	0	0	1	0
AN3 (LR/Y-)	0	0	1	1
AN4 (Wiper)	0	1	0	0
RESERVED (AN5 on LH7A404)	0	1	0	1
AN6	0	1	1	0
RESERVED (AN7 on LH7A404)	0	1	1	1
AN8	1	0	0	0
AN9	1	0	0	1
VREF -	1	0	1	0
VREF -	1	0	1	1
VREF -	1	1	0	0
VREF -	1	1	0	1
VREF -	1	1	1	0
VREF -	1	1	1	1

Table 4. LW Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///		BIASCON												RefM	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
ADDR	0xFFFC3000 + 0x04															

Table 5. LW Register Definitions

BIT	NAME	DESCRIPTION
31:14	///	Reserved Read as zero.
13:2	BIASCON	Bias Control These bits drive the FETs, as shown in Figure 14. B2 in the figure corresponds to bit [2] in this register.
1:0	RefM	Ref- Mux Determines the signal connected to the negative reference of the ADC during Idle Mode. 00 = VREF- (negative terminal of the internal bandgap reference) 01 = AN1 (UR/X-) 10 = AN3 (LR/Y-) 11 = AN9

NOTE: While the examples in Table 4 and Table 5 are specific to LH75400/01/10/11, minor differences will exist. For those other parts, use this information as a reference.

The measurement sequencer state machine starts in the IDLE state. During the idle state, the Control High Word contains the value of the Idle High Word and the Control Low Word contains the value of the Idle Low Word. This means that the Idle Low Word programs the state of the switches in the Bias and Control network. When a measurement is triggered by touching the touch screen, the state machine waits for the number of A2DCLK periods programmed into the idle time bit field of the Idle High Word register. Then, if the ADC still detects a touch on the touch screen, the sequencer advances to the GET_DATA state.

In the GET_DATA state, the measurement sequencer fetches the Control Word from the Control Bank array. The Control Bank is an array of 16 Control High Word values and 16 Control Low Word values. The sequencer accesses the Control Bank with indexes 0 through 15. The NOC bit field of the Power Configuration Register is programmed with the total number of fetches the sequencer will make from the Control Bank minus 1.

When the sequencer enters the GET_DATA state, the Control Bank index is 0. The sequencer loads the Control High Word from Control Bank High Word[0] and loads the Control Low Word from the Control Bank Low Word[0]. The new value in the Control Low Word causes the switches in the bias and control network to switch into the programmed state. Once the new control word is loaded, the sequencer enters the WAIT_CONV state. In the WAIT_CONV state, the sequencer waits for the number of A2DCLK periods programmed into the settling time bit field of the Control Bank High Word[0]. This wait allows the voltages, the bias, and control switch settings applied to the touch screen time to stabilize. Next, the ADC core takes a measurement. When the measurement is complete, the ADC core signals the measurement sequencer to fetch the conversion result. The measurement sequencer advances to the END_OF_SEQ state in which it fetches the conversion result, stores it in the result FIFO, and increments the Control Bank index.

The process that starts with fetching a new control word and ends with storing a measurement result repeats until the sequencer has taken the number of measurements programmed in the NOC bit field of the Power Configuration Register. Once all measurements in a sequence are complete, the sequencer sets the EOSINTR_UM bit in the Interrupt Status Register.

The measurement sequencer can be programmed to trigger when it detects that the touch screen has been touched. In this mode, the sequencer will automatically re-trigger for as long as a touch on the screen is detected. The measurement sequencer can also be programmed to trigger on software command or to trigger continuously. The 'Making Polled Measurements' and 'Making

Interrupt-driven Measurements' sections of this document describe which modes are most appropriate.

HOOKUP AND PROGRAMMING

This section provides wiring diagrams and step-by-step programming guides. For a complete C-language hardware abstraction for the ADC, along with configurations for all touch screens this note describes, visit the SMA web site at www.nxp.com. Note that IOCON and VIC Register Settings apply to the LH75400/01/10/11 only. For other devices, check the chip's User's Guide for comparable registers.

Basic ADC Setup

No matter what type of touch screen you are trying to measure, you should configure the measurement sequencer to perform the following steps.

1. Take a touch detection measurement.
2. Measure X.
3. Measure Y.
4. Take a touch detection measurement.

The assumption is that if you were touching the touch screen at the beginning of the measurement sequence and you were touching it at the end of the measurement sequence, then you were touching it during the X and Y measurements.

To measure whether you have touched the screen, program the bias and control network to attach a weak pull-up to the AN0 pin and connect a strong pull-down to the AN4 pin (5-wire and 7-wire screens) or the AN3 pin (4-wire and 8-wire screens). Program the reference input multiplexers to use the on-chip 2.0 V reference. Program the positive input multiplexer to measure on AN0. The negative input is the same as the negative reference. For example, if you choose a measurement threshold of 1/3 full scale, then you will detect a touch when the voltage on AN0 is $2.0 \text{ V} \div 3 = 0.66 \text{ V}$. That is plenty of noise margin over a 3.3 V range.

The problem you are likely to encounter is that the pull-up is very slow to pull the screen all the way up to VDDA_ADC. The solution is, before measuring the touch condition, pre-charge the AN0 pin to VDDA_ADC using the strong pull-up bit 2 of the Bias and Control Network. If you are touching the touch panel, then the strong pull-down, which the measurement sequencer switches in on the next step, will pull the AN0 pin down very quickly. If you are not touching the touch panel, then the weak pull-up will hold the AN0 pin at VDDA_ADC for the duration of touch detection measurement. All of the register configurations described in the touch screen specific Hookup and Programming Setup sections use this method for touch detection.

4-wire Touch Screen Hookup and Programming Setup

Figure 15 shows a 4-wire touch screen wired to AN0 - AN3, with its optional capacitors and filters.

Programming Steps:

1. Program all writable ADC registers to 0.
2. Flush the result FIFO by reading the result register while the FIFO Status Register bit, FEMPTY (2), is 0.
3. Program the chip registers, as shown in Table 7.
4. Use the polled or interrupt-driven measurement algorithm.

Once you have initially configured the ADC registers, see the Making Polled Measurements and the Making Interrupt-driven Measurements sections for algorithms to use.

Table 6. Pin Wiring

TOUCH SCREEN WIRE	ADC PIN NAME
X+	AN0
X-	AN1
Y+	AN2
Y-	AN3

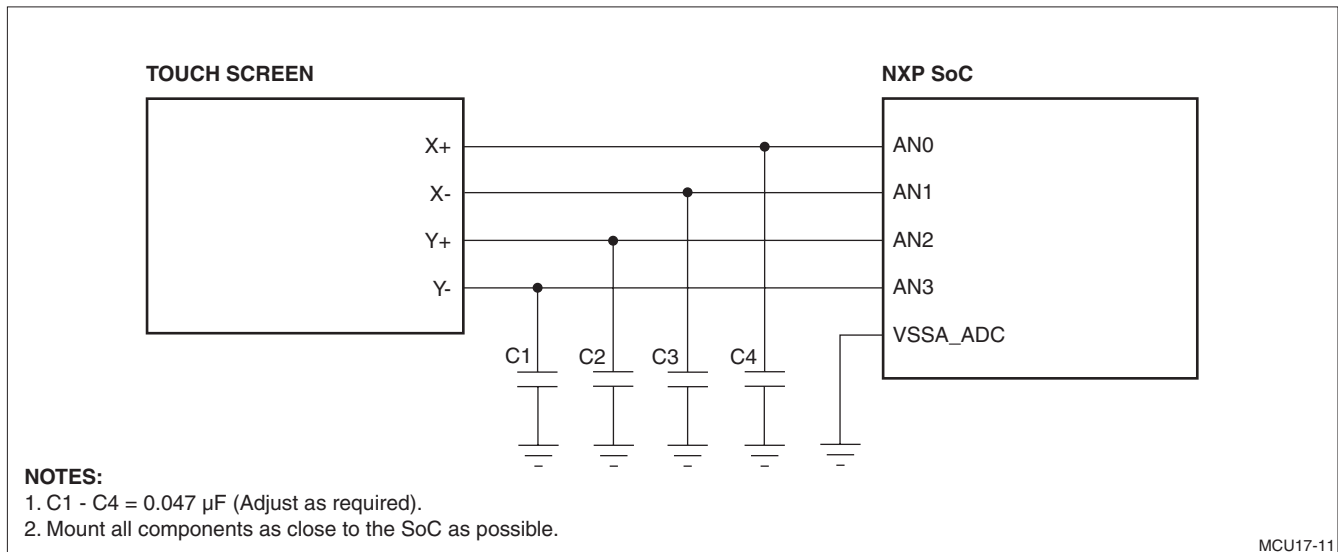


Figure 15. 4-wire Touch Screen Hookup

NOTE: While these examples are specific to the LH75400/01/10/11 parts, minor differences will exist. For those other parts, use this information as a reference.

Table 7. 4-wire Register Configurations

REGISTER NAME	REGISTER ADDRESS	VALUE AND COMMENTS ^{1, 2}
Vectored Interrupt Controller Configuration Register³		
IntEnableClear	0xFFFFF014	0b00000000000010B000000000000000 B = program bit 16 to 0 if you are using the combined ADC interrupt to handle the Brown Out interrupt. NXP recommends that you use the separate Brown Out Interrupt (VIC source 17) to handle brownout conditions. Therefore, program bit 16 to 1.
I/O Configuration Registers³		
ADC_MUX	0xFFFE5028	0b000000000xx0x0x0
ADC Registers		
High Word Register	0xFFFC3000	Read-only; do not program
Low Word Register	0xFFFC3004	Read-only; do not program
Results Register	0xFFFC3008	Read-only; do not program
Interrupt Masking	0xFFFC300C	0b0000000000000000 for polled mode 0b000000000100P100 for interrupt mode P = enable PENIRQ Interrupt per Algorithm.
Power Configuration	0xFFFC3010	0b00000CCC01101011 CCC = clock rate; see 'Optimizing the System' This performs a 12-conversion sequence.
General Configuration	0xFFFC3014	0b0000000000000T10 for polled mode Set T bit per algorithm 0b0000000000000TBP for interrupt mode Set T, B, and P bits per Algorithm
General Status	0xFFFC3018	Read-only; do not program
Interrupt Status	0xFFFC301C	Read-only; do not program
FIFO Status	0xFFFC3020	Read-only; do not program
High Word Control Bank Registers (SSSSSSSS bits determine the Settling Time)		
Register 0	0xFFFC3024	0bSSSSSSSSSS0000000 (pre-charge)
Register 1	0xFFFC3028	0bSSSSSSSSSS0000000 (touch detect)
Register 2	0xFFFC302C	0bSSSSSSSSSS0010001 (measure X)
Register 3	0xFFFC3030	0bSSSSSSSSSS0010001 (measure X)
Register 4	0xFFFC3034	0bSSSSSSSSSS0010001 (measure X)
Register 5	0xFFFC3038	0bSSSSSSSSSS0010001 (measure X)
Register 6	0xFFFC303C	0bSSSSSSSSSS0000010 (measure Y)
Register 7	0xFFFC3040	0bSSSSSSSSSS0000010 (measure Y)
Register 8	0xFFFC3044	0bSSSSSSSSSS0000010 (measure Y)
Register 9	0xFFFC3048	0bSSSSSSSSSS0000010 (measure Y)
Register 10	0xFFFC304C	0bSSSSSSSSSS0000000 (pre-charge)
Register 11	0xFFFC3050	0bSSSSSSSSSS0000000 (touch detect)
Register 12	0xFFFC3054	0b0000000000000000
Register 13	0xFFFC3058	0b0000000000000000
Register 14	0xFFFC305C	0b0000000000000000
Register 15	0xFFFC3060	0b0000000000000000

Table 7. 4-wire Register Configurations (Cont'd)

REGISTER NAME	REGISTER ADDRESS	VALUE AND COMMENTS ^{1, 2}
Low Word Control Bank Registers		
Register 0	0xFFFC3064	0b000000000000100 (pre-charge)
Register 1	0xFFFC3068	0b0001000010000000 (touch detect)
Register 2	0xFFFC306C	0b000000000010101 (measure X)
Register 3	0xFFFC3070	0b000000000010101 (measure X)
Register 4	0xFFFC3074	0b000000000010101 (measure X)
Register 5	0xFFFC3078	0b000000000010101 (measure X)
Register 6	0xFFFC307C	0b000000010100010 (measure Y)
Register 7	0xFFFC3080	0b000000010100010 (measure Y)
Register 8	0xFFFC3084	0b000000010100010 (measure Y)
Register 9	0xFFFC3088	0b000000010100010 (measure Y)
Register 10	0xFFFC308C	0b00000000000100 (pre-charge)
Register 11	0xFFFC3090	0b0001000010000000 (touch detect)
Register 12	0xFFFC3094	0b0000000000000000
Register 13	0xFFFC3098	0b0000000000000000
Register 14	0xFFFC309C	0b0000000000000000
Register 15	0xFFFC30A0	0b0000000000000000
Idle High Word	0xFFFC30A4	0b0000000000000000 (for polled mode) 0bIIIIIIII00000000 (for interrupt mode) IIIIIIII = idle time (see Optimizing the System)
Idle Low Word	0xFFFC30A8	0b0000000000000000 (for polled mode) 0b000A0000A0000000 (for interrupt mode) A = 1 during touch detect, 0 otherwise
Masked Int. Status	0xFFFC30AC	Read-only; do not program
Interrupt Clear	0xFFFC30B0	0b000000000000011 write only

NOTES:

1. AHB peripheral register configurations are shown as 32-bit binary numbers.
2. APB peripheral register configurations are shown in binary, least significant 16 bits only.
The most significant 16 bits must be programmed to 0.
3. These Register Settings pertain to the LH75400/01/10/11. Consult the chip User's Guide for the device you are using.

5-wire Touch Screen Hookup and Programming Setup

Figure 16 shows a 5-wire touch screen wired to AN0 - AN4 with its optional capacitors and filters.

Programming Steps:

1. Program all writeable ADC registers to 0.
2. Flush the result FIFO by reading the result register while FIFO Status Register bit, FEMPTY (2), is 0.
3. Program chip registers, as shown in Table 9.
4. Use the polled or interrupt-driven measurement algorithm.

Once you have initially configured the ADC registers, see the Making Polled Measurements and Making Interrupt-driven Measurements sections for algorithms to use.

Table 8. Pin Wiring

TOUCH SCREEN WIRE	ADC PIN NAME
UL	AN0
UR	AN1
LL	AN2
LR	AN3
Wiper	AN4

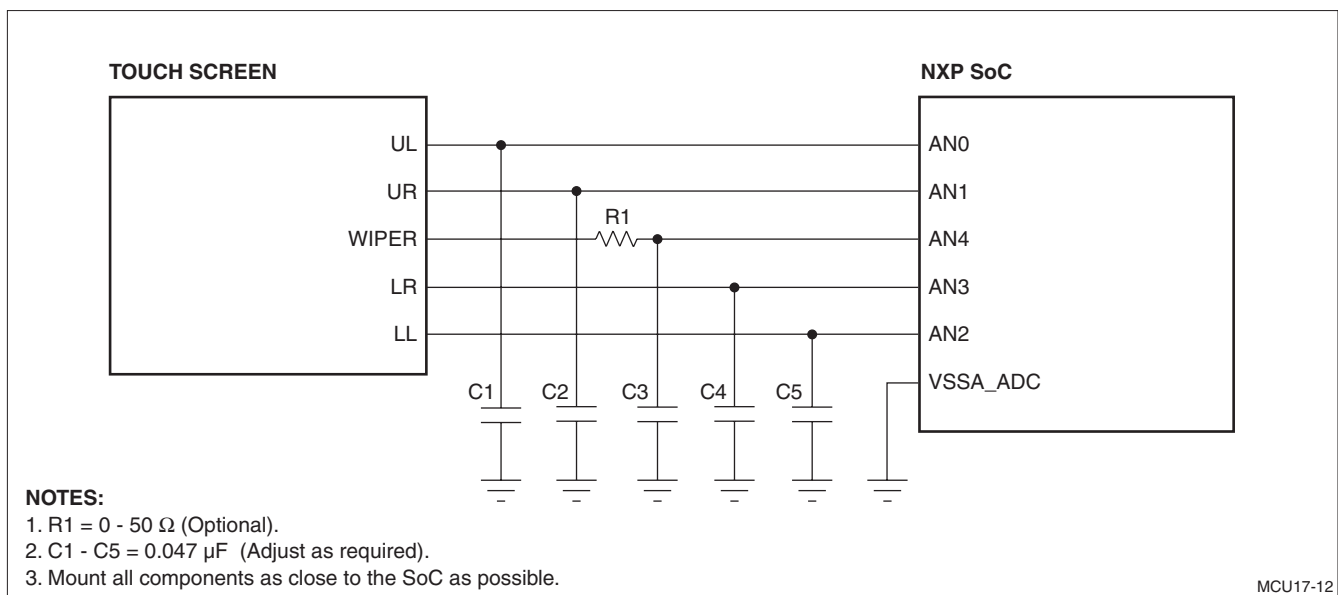


Figure 16. 5-wire Touch Screen Hookup

NOTE: While these example are specific to the LH75400/01/10/11 parts, minor differences will exist. For those other parts, use this information as a reference.

Table 9. 5-wire Register Configurations

REGISTER NAME	REGISTER ADDRESS	VALUE AND COMMENTS ^{1, 2}
Vectored Interrupt Controller Configuration Register³		
IntEnableClear	0xFFFFF014	0b00000000000010B000000000000000 B = program bit 16 to 0 if you are using the combined ADC interrupt to handle the Brown Out interrupt. NXP recommends that you use the separate Brown Out Interrupt (VIC source 17) to handle brownout conditions. Therefore, program bit 16 to 1.
I/O Configuration Registers (configure pins as ADC pins instead of digital inputs)³		
ADC_MUX	0xFFFE5028	0b00000000xx0x0x0
ADC Registers		
High Word Register	0xFFFC3000	Read-only; do not program
Low Word Register	0xFFFC3004	Read-only; do not program
Results Register	0xFFFC3008	Read-only; do not program
Interrupt Masking	0xFFFC300C	0b0000000000000000 (for polled mode) 0b00000000100P100 (for interrupt mode) P = enable PENIRQ Interrupt per algorithm.
Power Configuration	0xFFFC3010	0b00000CCC01101011 CCC = clock rate; see 'Optimizing the System' This performs a 12-conversion sequence
General Configuration	0xFFFC3014	0b000000000000T10 (for polled mode) Set T bit per algorithm 0b000000000000TBP (for interrupt mode) Set T, B, and P bits per algorithm
General Status	0xFFFC3018	Read-only; do not program
Interrupt Status	0xFFFC301C	Read-only; do not program
FIFO Status	0xFFFC3020	Read-only; do not program
High Word Control Bank Registers (SSSSSSSS bits determine the Settling Time)		
Register 0	0xFFFC3024	0bSSSSSSSS0000000 (pre-charge)
Register 1	0xFFFC3028	0bSSSSSSSS0000000 (touch detect)
Register 2	0xFFFC302C	0bSSSSSSSS0100001 (measure X)
Register 3	0xFFFC3030	0bSSSSSSSS0100001 (measure X)
Register 4	0xFFFC3034	0bSSSSSSSS0100001 (measure X)
Register 5	0xFFFC3038	0bSSSSSSSS0100001 (measure X)
Register 6	0xFFFC303C	0bSSSSSSSS0100001 (measure Y)
Register 7	0xFFFC3040	0bSSSSSSSS0100001 (measure Y)
Register 8	0xFFFC3044	0bSSSSSSSS0100001 (measure Y)
Register 9	0xFFFC3048	0bSSSSSSSS0100001 (measure Y)
Register 10	0xFFFC304C	0bSSSSSSSS0000000 (pre-charge)
Register 11	0xFFFC3050	0bSSSSSSSS0000000 (touch detect)
Register 12	0xFFFC3054	0b0000000000000000
Register 13	0xFFFC3058	0b0000000000000000
Register 14	0xFFFC305C	0b0000000000000000
Register 15	0xFFFC3060	0b0000000000000000

Table 9. 5-wire Register Configurations (Cont'd)

REGISTER NAME	REGISTER ADDRESS	VALUE AND COMMENTS ^{1, 2}
Low Word Control Bank Registers		
Register 0	0xFFFC3064	0b0000000000000100 (pre-charge)
Register 1	0xFFFC3068	0b0001000100000000 (touch detect)
Register 2	0xFFFC306C	0b0000000010110110 (measure X)
Register 3	0xFFFC3070	0b0000000010110110 (measure X)
Register 4	0xFFFC3074	0b0000000010110110 (measure X)
Register 5	0xFFFC3078	0b0000000010110110 (measure X)
Register 6	0xFFFC307C	0b0000000011001110 (measure Y)
Register 7	0xFFFC3080	0b0000000011001110 (measure Y)
Register 8	0xFFFC3084	0b0000000011001110 (measure Y)
Register 9	0xFFFC3088	0b0000000011001110 (measure Y)
Register 10	0xFFFC308C	0b0000000000000100 (pre-charge)
Register 11	0xFFFC3090	0b0001000100000000 (touch detect)
Register 12	0xFFFC3094	0b0000000000000000
Register 13	0xFFFC3098	0b0000000000000000
Register 14	0xFFFC309C	0b0000000000000000
Register 15	0xFFFC30A0	0b0000000000000000
Idle High Word	0xFFFC30A4	0b0000000000000000 (for polled mode) 0bIIIIIIII00000000 (for interrupt mode) IIIIIIII = idle time (see 'Optimizing the System')
Idle Low Word	0xFFFC30A8	0b0000000000000000 (for polled mode) 0b000A0000A0000000 (for interrupt mode) A = 1 during touch detect, 0 otherwise
Masked Int. Status	0xFFFC30AC	Read-only; do not program
Interrupt Clear	0xFFFC30B0	0b0000000000000011 write only

NOTES:

1. AHB peripheral register configurations are shown as 32-bit binary numbers.
2. APB peripheral register configurations are shown in binary, least significant 16 bits only.
The most significant 16 bits must be programmed to 0.
3. These Register Settings pertain to the LH75400/01/10/11. Consult the chip User's Guide for the device you are using.

7-wire Touch Screen Hookup and Programming Setup

Figure 17 shows a 7-wire touch screen wired to AN0 - AN4, AN8 and AN9, along with its optional capacitors and filters.

Programming Steps:

1. Program all writeable ADC registers to 0.
2. Flush the result FIFO by reading the result register while FIFO Status Register bit, FEMPTY (2), is 0.
3. Program chip registers, as shown in Table 11.
4. Use the polled or interrupt-driven measurement algorithm.

Once you have initially configured the ADC registers, see the Making Polled Measurements and Making Interrupt-driven Measurements sections for algorithms to use.

Table 10. Pin Wiring

TOUCH SCREEN WIRE	ADC PIN NAME
UL Excite	AN0
UL Sense	AN8
UR	AN1
LL	AN2
LR Excite	AN3
LR Sense	AN9
Wiper	AN4

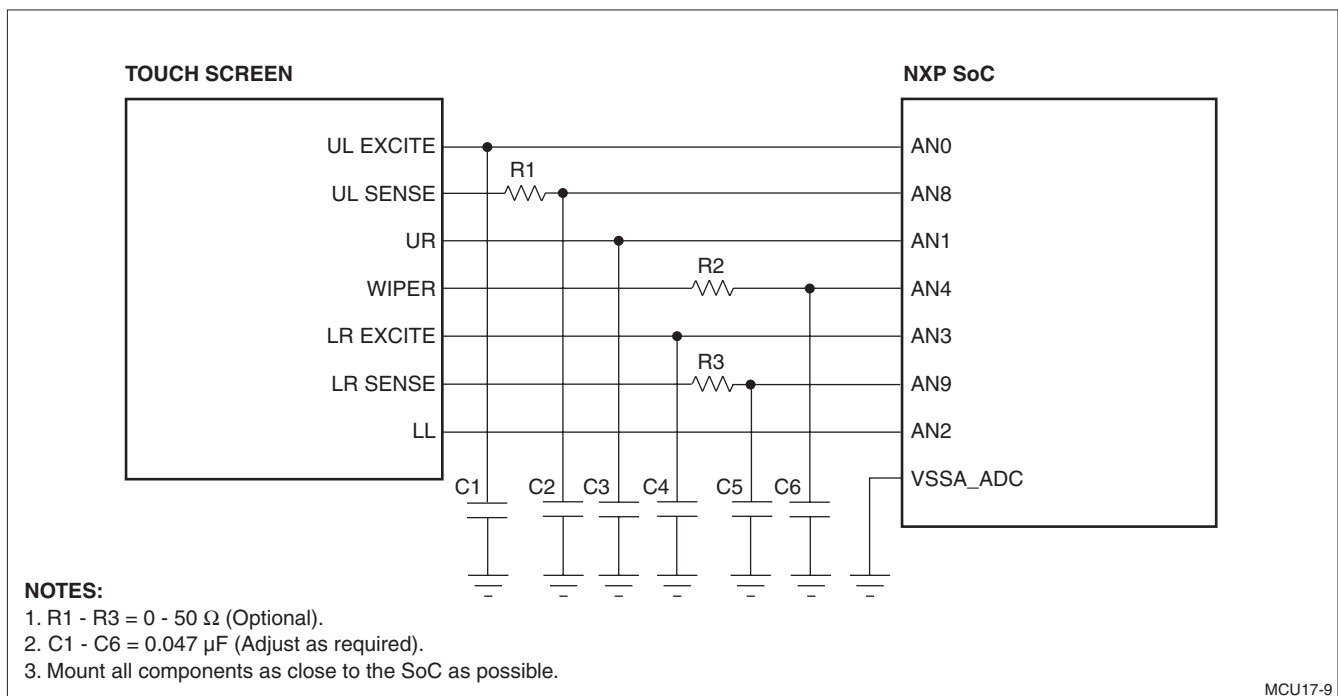


Figure 17. 7-wire Touch Screen Hookup

NOTE: While these examples are specific to the LH75400/01/10/11 parts, minor differences will exist. For those other parts, use this information as a reference.

Table 11. 7-wire Register Configurations

REGISTER NAME	REGISTER ADDRESSES	VALUE AND COMMENTS ^{1, 2}
Vectored Interrupt Controller Configuration Register³		
IntEnableClear	0xFFFFF014	0b00000000000010B000000000000000 B = program bit 16 to 0 if you are using the combined ADC interrupt to handle the Brown Out interrupt. NXP recommends that you use the separate Brown Out Interrupt (VIC source 17) to handle brownout conditions. Therefore, program bit 16 to 1.
I/O Configuration Registers (configure pins as ADC pins instead of digital inputs)³		
ADC_MUX	0xFFFE5028	0b00000000000000x0
ADC Registers		
High Word Register	0xFFFC3000	Read-only; do not program
Low Word Register	0xFFFC3004	Read-only; do not program
Results Register	0xFFFC3008	Read-only; do not program
Interrupt Masking	0xFFFC300C	0b0000000000000000 for polled mode 0b000000000100P100 for interrupt mode P = enable PENIRQ Interrupt per algorithm.
Power Configuration	0xFFFC3010	0b00000CCC01101011 CCC = clock rate; see 'Optimizing the System' This performs a 12-conversion sequence.
General Configuration	0xFFFC3014	0b00000000000000T10 (for polled mode) Set T bit per algorithm 0b00000000000000TBP (for interrupt mode) Set T, B, and P bits per algorithm
General Status	0xFFFC3018	Read-only; do not program
Interrupt Status	0xFFFC301C	Read-only; do not program
FIFO Status	0xFFFC3020	Read-only; do not program
High Word Control Bank Registers (SSSSSSSS bits determine the Settling Time)		
Register 0	0xFFFC3024	0bSSSSSSSS0000000 pre-charge
Register 1	0xFFFC3028	0bSSSSSSSS0000000 (touch detect)
Register 2	0xFFFC302C	0bSSSSSSSS0100011 (measure X)
Register 3	0xFFFC3030	0bSSSSSSSS0100011 (measure X)
Register 4	0xFFFC3034	0bSSSSSSSS0100011 (measure X)
Register 5	0xFFFC3038	0bSSSSSSSS0100011 (measure X)
Register 6	0xFFFC303C	0bSSSSSSSS0100011 (measure Y)
Register 7	0xFFFC3040	0bSSSSSSSS0100011 (measure Y)
Register 8	0xFFFC3044	0bSSSSSSSS0100011 (measure Y)
Register 9	0xFFFC3048	0bSSSSSSSS0100011 (measure Y)
Register 10	0xFFFC304C	0bSSSSSSSS0000000 (pre-charge)
Register 11	0xFFFC3050	0bSSSSSSSS0000000 (touch detect)
Register 12	0xFFFC3054	0b0000000000000000
Register 13	0xFFFC3058	0b0000000000000000
Register 14	0xFFFC305C	0b0000000000000000
Register 15	0xFFFC3060	0b0000000000000000

Table 11. 7-wire Register Configurations (Cont'd)

REGISTER NAME	REGISTER ADDRESSES	VALUE AND COMMENTS ^{1, 2}
Low Word Control Bank Registers		
Register 0	0xFFFC3064	0b0000000000000100 (pre-charge)
Register 1	0xFFFC3068	0b0001000100000000 (touch detect)
Register 2	0xFFFC306C	0b0000000010110111 (measure X)
Register 3	0xFFFC3070	0b0000000010110111 (measure X)
Register 4	0xFFFC3074	0b0000000010110111 (measure X)
Register 5	0xFFFC3078	0b0000000010110111 (measure X)
Register 6	0xFFFC307C	0b0000000011001111 (measure Y)
Register 7	0xFFFC3080	0b0000000011001111 (measure Y)
Register 8	0xFFFC3084	0b0000000011001111 (measure Y)
Register 9	0xFFFC3088	0b0000000011001111 (measure Y)
Register 10	0xFFFC308C	0b0000000000000100 (pre-charge)
Register 11	0xFFFC3090	0b0001000100000000 (touch detect)
Register 12	0xFFFC3094	0b0000000000000000
Register 13	0xFFFC3098	0b0000000000000000
Register 14	0xFFFC309C	0b0000000000000000
Register 15	0xFFFC30A0	0b0000000000000000
Idle High Word	0xFFFC30A4	0b0000000000000000 (for polled mode) 0bIIIIIIII00000000 (for interrupt mode) IIIIIIII = idle time; see 'Optimizing the System'
Idle Low Word	0xFFFC30A8	0b0000000000000000 (for polled mode) 0b000A0000A0000000 (for interrupt mode) A = 1 during touch detect, 0 otherwise
Masked Int. Status	0xFFFC30AC	Read-only; do not program
Interrupt Clear	0xFFFC30B0	0b0000000000000011 write only

NOTES:

1. AHB peripheral register configurations are shown as 32-bit binary numbers.
2. APB peripheral register configurations are shown in binary, least significant 16 bits only.
You must program the most significant 16 bits to 0.
3. These Register Settings pertain to the LH75400/01/10/11. Consult the chip User's Guide for the device you are using.

8-wire Touch Screen Hookup and Programming Setup

Figure 18 shows an 8-wire touch screen wired to AN0 - AN4, AN8, and AN9, along with its optional capacitors and filters and the required external P-channel MOSFET.

Programming Steps:

1. Program all writeable ADC registers to 0.
2. Flush the result FIFO by reading the result register while FIFO Status Register bit, FEMPTY (2), is 0.
3. Program chip registers, as shown in Table 13.
4. Use the polled or interrupt-driven measurement algorithm.

Once you have initially configured the ADC registers, see the Making Polled Measurements and Making Interrupt-driven Measurements sections for algorithms to use.

Table 12. Pin Wiring

TOUCH SCREEN WIRE	ADC PIN NAME
X+ Sense	AN0
X- Sense	AN1
X+ Excite	Drain of P-channel FET. Source to VDDA_ADC and gate to X-Excite
X- Excite	AN4
Y+ Sense	AN8
Y- Sense	AN9
Y+ Excite	AN2
Y- Excite	AN3

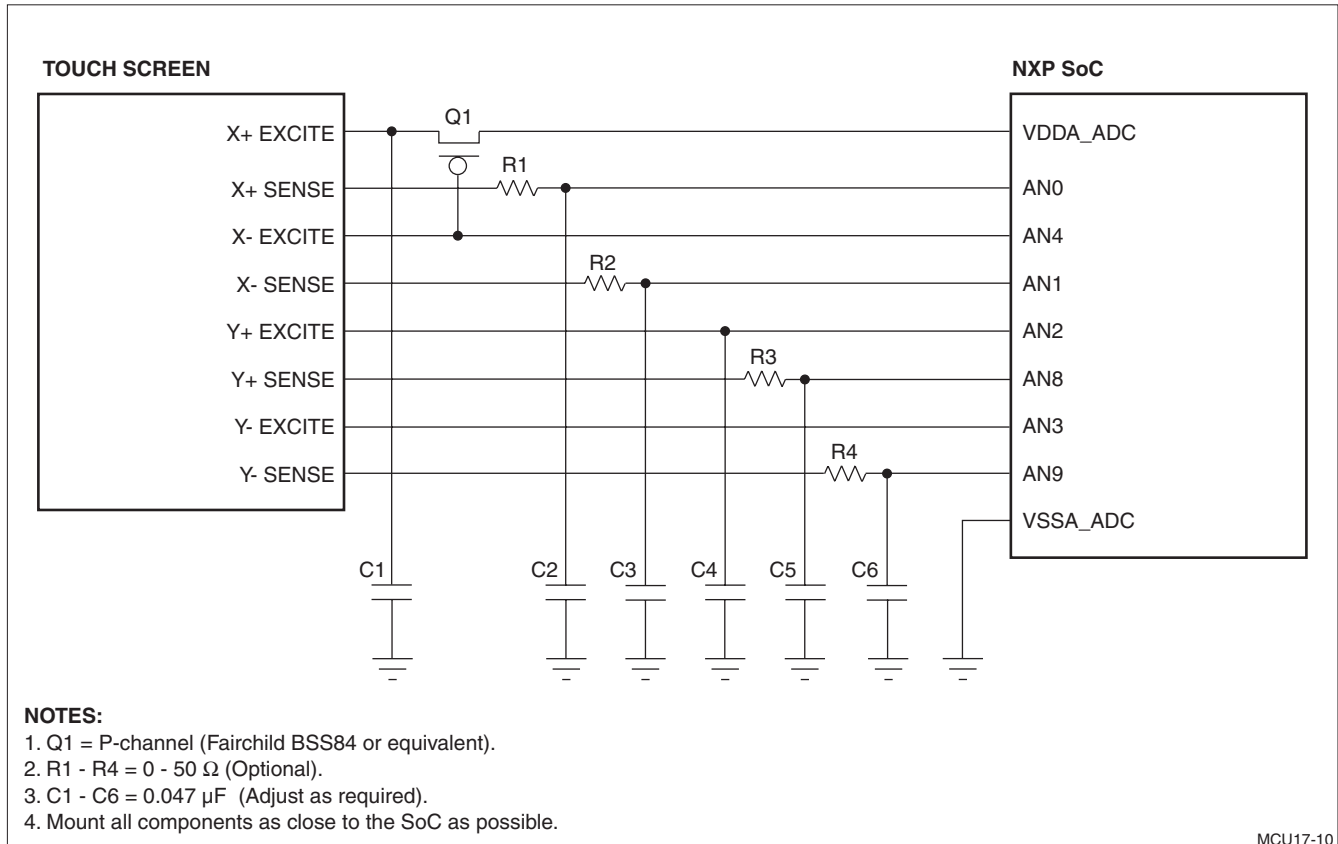


Figure 18. 8-wire Touch Screen Hookup

NOTE: While these examples are specific to the LH75400/01/10/11 parts, minor differences will exist. For those other parts, use this information as a reference.

Table 13. 8-wire Register Configurations

REGISTER NAME	REGISTER ADDRESS	VALUE AND COMMENTS ^{1, 2}
Vectored Interrupt Controller Configuration Register³		
IntEnableClear	0xFFFFF014	0b000000000000010B0000000000000000 B = program bit 16 to 0 if you are using the combined ADC interrupt to handle the brown out interrupt. NXP recommends that you use the separate Brown Out Interrupt (VIC source 17) to handle brownout conditions. Therefore, program bit 16 to 1.
I/O Configuration Registers (configure pins as ADC pins instead of digital inputs)³		
ADC_MUX	0xFFFE5028	0b00000000000000x0
ADC Registers		
High Word Register	0xFFFC3000	Read-only; do not program
Low Word Register	0xFFFC3004	Read-only; do not program
Results Register	0xFFFC3008	Read-only; do not program
Interrupt Masking	0xFFFC300C	0b0000000000000000 for polled mode
		0b000000000100P100 for interrupt mode
		P is enable PENIRQ Interrupt per algorithm.
Power Configuration	0xFFFC3010	0b00000CCC01101011
		CCC is clock rate; see optimizing the system
		This performs a 12 conversion sequence
General Configuration	0xFFFC3014	0b0000000000000T10 for polled mode
		Set T bit per algorithm
		0b0000000000000TBP for interrupt mod
		Set T, B and P bits per algorithm
General Status	0xFFFC3018	Read-only; do not program
Interrupt Status	0xFFFC301C	Read-only; do not program
FIFO Status	0xFFFC3020	Read-only; do not program
High Word Control Bank Registers (SSSSSSSS bits determine the Settling Time)		
Register 0	0xFFFC3024	0bSSSSSSSS0000000 (pre-charge)
Register 1	0xFFFC3028	0bSSSSSSSS0000000 (touch detect)
Register 2	0xFFFC302C	0bSSSSSSSS1000001 (measure X)
Register 3	0xFFFC3030	0bSSSSSSSS1000001 (measure X)
Register 4	0xFFFC3034	0bSSSSSSSS1000001 (measure X)
Register 5	0xFFFC3038	0bSSSSSSSS1000001 (measure X)
Register 6	0xFFFC303C	0bSSSSSSSS0000011 (measure Y)
Register 7	0xFFFC3040	0bSSSSSSSS0000011 (measure Y)
Register 8	0xFFFC3044	0bSSSSSSSS0000011 (measure Y)
Register 9	0xFFFC3048	0bSSSSSSSS0000011 (measure Y)
Register 10	0xFFFC304C	0bSSSSSSSS0000000 (pre-charge)
Register 11	0xFFFC3050	0bSSSSSSSS0000000 (touch detect)
Register 12	0xFFFC3054	0b0000000000000000
Register 13	0xFFFC3058	0b0000000000000000

Table 13. 8-wire Register Configurations (Cont'd)

REGISTER NAME	REGISTER ADDRESS	VALUE AND COMMENTS ^{1, 2}
Register 14	0xFFFC305C	0b0000000000000000
Register 15	0xFFFC3060	0b0000000000000000
Low Word Control Bank Registers		
Register 0	0xFFFC3064	0b0000000000000100 (pre-charge)
Register 1	0xFFFC3068	0b0001000010000000 (touch detect)
Register 2	0xFFFC306C	0b0000000100000001 (measure X)
Register 3	0xFFFC3070	0b0000000100000001 (measure X)
Register 4	0xFFFC3074	0b0000000100000001 (measure X)
Register 5	0xFFFC3078	0b0000000100000001 (measure X)
Register 6	0xFFFC307C	0b0000000010100011 (measure Y)
Register 7	0xFFFC3080	0b0000000010100011 (measure Y)
Register 8	0xFFFC3084	0b0000000010100011 (measure Y)
Register 9	0xFFFC3088	0b0000000010100011 (measure Y)
Register 10	0xFFFC308C	0b0000000000000100 (pre-charge)
Register 11	0xFFFC3090	0b0001000010000000 (touch detect)
Register 12	0xFFFC3094	0b0000000000000000
Register 13	0xFFFC3098	0b0000000000000000
Register 14	0xFFFC309C	0b0000000000000000
Register 15	0xFFFC30A0	0b0000000000000000
		0b0000000000000100 (pre-charge)
Idle High Word	0xFFFC30A4	0b0000000000000000 for polled mode
		0bIIIIIIII00000000 for interrupt mode
		IIIIIIII is idle time (see Optimizing the System)
Idle Low Word	0xFFFC30A8	0b0000000000000000 for polled mode
		0b000A0000A0000000 for interrupt mode
		A = 1 during touch detect, 0 otherwise
Masked Int. Status	0xFFFC30AC	Read-only; do not program
Interrupt Clear	0xFFFC30B0	0b0000000000000011 write only

NOTES:

1. AHB peripheral register configurations are shown as 32-bit binary numbers.
2. APB peripheral register configurations are shown in binary, least significant 16 bits only.
You must program the most significant 16 bits to 0.
3. These Register Settings pertain to the LH75400/01/10/11. Consult the chip User's Guide for the device you are using.

Converting Result FIFO Values into Voltages and Screen Coordinates

The result FIFO returns a 16-bit number with these characteristics:

The most significant 10 bits are the ADC result.

The least significant 4 bits are the index into the Control Bank array when the measurement was taken.

Bits 4 and 5 are reserved.

To isolate the conversion result, shift the value read from the result FIFO to the right 6 times (divide by 64).

To convert a measurement FIFO value into volts, multiply the right-shifted measurement result by the reference voltage (the full scale voltage) and divide by 0x3FF (the full scale conversion).

To convert an X coordinate measurement into a screen coordinate, multiply the right-shifted result register value by (the screen width – 1) and divide by 0x3FF.

To convert a Y coordinate measurement into a screen coordinate, multiply the right-shifted result register value by (the screen height – 1) and divide by 0x3FF. Be careful to do the multiply before the divide to maximize precision.

If the touch X coordinates seem to be decreasing as the touch point moves from left to right, don't change the wiring. Instead, subtract the X coordinate from the screen width – 1. Depending on whether you are using Cartesian coordinates or raster coordinates for your display system, Y coordinates either have to increase or decrease as the touch point moves from the bottom to the top of the screen. If the Y coordinates are increasing when they should be decreasing, subtract the computed Y coordinate value from the screen height – 1.

Sometimes, X and Y are transposed due to the naming convention on the touch screen. Again, don't change the wiring. Exchange the X measurement and the Y measurement in your calculations. The order in which you measure X and Y does not matter.

There are a number of errors associated with converting touch screen measurements into LCD pixel coordinates. Some have to do with the ADC, and some have to do with the physical attachment of the touch screen to the LCD. An excellent article by Carlos E. Vidales in the May 2002 issue of *Embedded Systems Programming* called 'How to Calibrate Touch Screens' describes the theory behind touch screen calibration and provides C source code for routines that do it.

Making Polled Measurements

Polling is the best way to make measurements when you are using the channels of the ADC that are not hooked up to the touch screen for other measurements. Set up the measurement sequencer for soft-triggered measurements by writing 0b10 to the SSM field of the General Configuration register. To trigger a measurement, write a 1 to the SSB bit of the General Configuration register. To save power, make sure the Idle Low Word is 0 so it does not cause the Bias and Control Network to bias or pull up on either touch screen layer.

Set up an operating system task at the appropriate priority to poll the ADC in an infinite loop. This algorithm assumes that the touch screen measurements are taken before any other measurements.

To take a polled measurement:

1. Write 1 to the EOSINTC bit of the Interrupt Clear Register.
2. Sleep 100 ms. You may increase or decrease this value to achieve a slower or faster sampling rate.
3. Trigger a measurement by writing 1 to the SSB bit field of the General Configuration Register.
4. Poll the Interrupt Status Register until the EOSINTR_UM bit is set.
5. Read and discard the first measurement (pre-bias the sense layer to VDDA_ADC).
6. Read the second measurement. This is the first touch detect measurement.

If the measured voltage is above threshold:

- a. Read and discard the remaining ten ADC-related measurements.
- b. Record that there is no touch detected.
- c. Skip to step 10.

Otherwise:

- a. Read and average the next four measurements. These are the X coordinate measurement.
 - b. Read and average the next four measurements. These are the Y coordinate measurement.
 - c. Read and discard the next measurement. (Pre-bias the sense layer to VDDA_ADC.)
7. Read the last touch screen measurement. This is the second touch detect measurement. If the measured voltage is above threshold, discard previous X and Y measurements and record that there is no touch detected. Otherwise, record that a touch was detected.

8. If there was no touch detected and the previous state was 'touch detected,' report to the rest of the system that the touch screen is no longer being touched. You can report to the system by posting a message to a global variable, queue, or mailbox.
9. If the previous state was 'no touch detected' and the current state is 'touch detected,' or if the previous and current state was 'touch detected' and the new X or Y coordinate is different from the previous X or Y coordinate, report to the rest of the system that a touch is detected at the X and Y coordinates.
10. Make the current state of the screen (touch state and coordinates) the new previous state of the touch screen.
11. Read the remaining non-touch screen results from the result FIFO (if any), and process them accordingly.

Making Interrupt-driven Measurements

If you are using the ADC as a touch screen controller only, you can minimize power consumption by using interrupts. The algorithm this section describes uses a timer interrupt to set the touch screen coordinate measurement rate. It uses the PENIRQ interrupt to detect screen touches and only biases the touch screen for measurement for as long as a touch is detected on the touch panel. You will need an interrupt handler for the PENIRQ, the timer, and the end-of-measurement-sequence interrupts.

To use a timer interrupt to set the touch screen coordinate measurement rate, configure the timer to generate an interrupt at the desired rate and automatically clear the count register. Set the timer interrupt handler to:

1. Clear the timer interrupt.
2. Trigger a measurement by writing 1 to the SSB bit field of the General Configuration Register.

Set up an end of sequence handler to:

1. Stop the timer.
2. Write 1 to the EOSINTC bit of the Interrupt Clear Register.
3. Read and discard the first measurement (pre-bias the sense layer to VDDA_ADC)
4. Read the second measurement. This is the first touch detect measurement. If the measured voltage is above threshold, read and discard the remaining ten ADC-related measurements. Record that there is no touch detected. Skip to step 11.

- Otherwise,
5. Read and average the next 4 measurements. These are the X coordinate measurement.
 6. Read and average the next 4 measurements. These are the Y coordinate measurement.
 7. Read and discard the next measurement. (pre-bias the sense layer to VDDA_ADC).
 8. Read the last touch screen measurement. This is the second touch detect measurement. If the measured voltage is above threshold, discard previous X and Y measurements and record that there is no touch detected. Otherwise, record that a touch was detected.
 9. If there was no touch detected and the previous state was touch detected, report to the rest of the system (e.g., post a message to a global variable, queue or mailbox) that the touch screen is no longer being touched. If the previous state was no touch detected and the current state is touch detected, or if the previous and current state was touch detected and the new X or Y coordinate is different from the previous X or Y coordinate, report to the rest of the system that a touch is detected at the X and Y coordinates.
 10. Make the current state of the screen (touch state and coordinates) the new previous state of the touch screen.
 11. If there was a touch detected, start the timer and exit. Otherwise,
 12. Change the Idle Word Register to bias the touch screen for touch detect.
 13. Enable the Touch Detect Interrupt.
 14. Change the SSM bit field of the General Configuration Register to 0b01 to enable touch-triggered measurements.
 15. Exit.

Finally, set up a touch detect interrupt handler (PENIRQ) to:

1. Disable the touch detect interrupt.
2. Clear the touch detect interrupt.
3. Change the Idle Word Register so that the touch screen is completely unbiased during the idle state.
4. Enable soft-triggered measurements by writing 0b10 to the SSM bit field of the General configuration register.
5. Clear the timer interrupt.
6. Clear the timer.

To initialize the system:

1. Disable IRQ Exceptions.
2. Set up the ADC as described in the appropriate Hookup and Programming section for the type of screen being used.
3. Set up a timer to generate a periodic interrupt at the desired rate.
4. Install the interrupt handlers.
5. Set up the ADC to start a measurement on touch detect by writing 0b01 to the SSM field of the General Configuration Register.
6. Enable IRQ Exceptions.

The touch screen will be biased for touch detection. When the screen is touched, a measurement is triggered and the touch detection interrupt handler starts. The touch detection interrupt handler changes the ADC to soft-triggered mode. When the touch triggered measurement completes, the end of sequence interrupt will happen. If the touch is still detected, then the new coordinates will be sent to the system, the timer will start, and the timer interrupt will trigger a new measurement periodically. If there was a false touch detection or if you stop touching the screen, the end of sequence handler discards the measurement and re-arms touch detection.

OPTIMIZING THE SYSTEM

Once the system is operating, you can adjust it for optimum performance. This section discusses the most common issues.

A2DCLK Frequency, Idle Time, and Settling Time Programming

The A2DCLK frequency, idle time setting, and settling time settings are a tradeoff between measurement accuracy, false measurement trigger prevention/touch responsiveness, and power consumption. The A2DCLK frequency can be no more than 2 MHz and no less than 200 kHz or else the ADC won't function properly.

To maximize measurement accuracy and minimize false touch measurement triggers, program as slow an A2DCLK and as long a settling time and idle time as possible.

On the other hand, the biggest power drain of touch screen measurements comes when the ADC is biasing a resistive layer with 0 V along one edge and VDDA_ADC on the other edge. Since the screen remains biased during the entire settling time, minimize the settling time and maximize the A2DCLK to keep the period the screen is biased as short as possible.

To minimize the time it takes for a touch to trigger a measurement, minimize the idle time.

Use an oscilloscope to adjust the settling time for each step in the measurement sequence. Hook up the touch screen and program the ADC as described in the appropriate section. Start with the slowest possible A2DCLK and the longest possible settling time. Set the ADC for continuous measurements (SSM bit field of the General Configuration Register is 0b11). Touch the screen with a stylus. Put a scope probe on each touch screen bias voltage and measurement point and observe how long it takes to stabilize after it switches. Back off the settling time until the bias voltages and measured voltages are just stable. To get the best look at the switching, touch the corner of the screen closest to the bias voltage.

Improving Signal-to-Noise Ratio

Resistive touch screens can be good collectors of noise. You will notice a problem with touch screen noise if you are trying to draw a straight line and you wind up with a jagged one, or if you are trying to drag an object across the screen and you see it shaking even though your stylus is perfectly still.

Touch screens sit right on top of the LCD, so they are subjected to the LCD timing signals and the backlight/frontlight inverter noise. They are also big surfaces exposed to the outside world and are subject to all the noise in the environment. Remember that for a 10-bit ADC with a 3.3 V reference, it only takes about 3.2 mV of noise to add 1 LSB of error to a measurement.

If you are having problems with noise, you can add capacitors and ferrite beads to the biasing signals. Because they have separate biasing and sensing wires, 5-wire, 7-wire, and 8-wire screens allow active filters to be added to the ADC inputs in severe noise environments. Be aware that the more filtering applied to the sensing wires, the longer the settling time likely to be required.

The ADC setups for touch screen measurement in this note assume that you want to average four measurements in the X direction and four measurements in the Y direction for each coordinate pair. If there is random noise on the wires, four averages will improve the signal to noise ratio of the measurement by 1 LSB. If you know that the noise is below the Nyquist frequency of your ADC system (unlikely), you can apply digital filtering to achieve better noise suppression than averaging will provide.

Any extra measurements cost power, so if the environment is not noisy or if the extra measurement precision is not required, reduce the number of measurements to 1 in the X direction and 1 in Y direction.

Non-Linearity

Everything in this Application Note assumes that the touch screen resistive layers and the ADC operate in a perfectly linear fashion. The NXP ADC and most small touch screens are sufficiently linear for most applications. If you have minimized noise and calibrated the touch screen using the algorithm described here and you are still observing non-uniform coordinate measurements for uniform stylus movement, contact your touch screen manufacturer for assistance on linearizing your measurements.

REFERENCES

Carlos E. Vidales, 'How to Calibrate Touch Screens,' *Embedded Systems Programming*, May 2002. (Also available on the web in the back issue archive at: www.embedded.com)

Hampshire Company, Inc. AN103, 'How an Analog Resistive Touch Screen Works.'

Victor Marten, 'Human Input Devices Based on Resistive Touch Screen Sensors,' *ECN*, May 2001.

ANNEX A: Disclaimers (11)

1. t001dis100.fm: General (DS, AN, UM, errata)

General — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information.

2. t001dis101.fm: Right to make changes (DS, AN, UM, errata)

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

3. t001dis102.fm: Suitability for use (DS, AN, UM, errata)

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in medical, military, aircraft, space or life support equipment, nor in applications where failure or malfunction of a NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors accepts no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

4. t001dis103.fm: Applications (DS, AN, UM, errata)

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

5. t001dis104.fm: Limiting values (DS)

Limiting values — Stress above one or more limiting values (as defined in the Absolute Maximum Ratings System of IEC 60134) may cause permanent damage to the device. Limiting values are stress ratings only and operation of the device at these or any other conditions above those given in the Characteristics sections of this document is not implied. Exposure to limiting values for extended periods may affect device reliability.

6. t001dis105.fm: Terms and conditions of sale (DS)

Terms and conditions of sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at <http://www.nxp.com/profile/terms>, including those pertaining to warranty, intellectual property rights infringement and limitation of liability, unless explicitly otherwise agreed to in writing by NXP Semiconductors. In case of any inconsistency or conflict between information in this document and such terms and conditions, the latter will prevail.

7. t001dis106.fm: No offer to sell or license (DS)

No offer to sell or license — Nothing in this document may be interpreted or construed as an offer to sell products that is open for acceptance or the grant, conveyance or implication of any license under any copyrights, patents or other industrial or intellectual property rights.

8. t001dis107.fm: Hazardous voltage (DS, AN, UM, errata; if applicable)

Hazardous voltage — Although basic supply voltages of the product may be much lower, circuit voltages up to 60 V may appear when operating this product, depending on settings and application. Customers incorporating or otherwise using these products in applications where such high voltages may appear during operation, assembly, test etc. of such application, do so at their own risk. Customers agree to fully indemnify NXP Semiconductors for any damages resulting from or in connection with such high voltages. Furthermore, customers are drawn to safety standards (IEC 950, EN 60 950, CENELEC, ISO, etc.) and other (legal) requirements applying to such high voltages.

9. t001dis108.2.fm: Bare die (DS; if applicable)

Bare die (if applicable) — Products indicated as Bare Die are subject to separate specifications and are not tested in accordance with standard testing procedures. Product warranties and guarantees as stated in this document are not applicable to Bare Die Products unless such warranties and guarantees are explicitly stated in a valid separate agreement entered into by NXP Semiconductors and customer.

10. t001dis109.fm: AEC unqualified products (DS, AN, UM, errata; if applicable)

AEC unqualified products — This product has not been qualified to the appropriate Automotive Electronics Council (AEC) standard Q100 or Q101 and should not be used in automotive critical applications, including but not limited to applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors accepts no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is for the customer's own risk.

11. t001dis110.fm: Suitability for use in automotive applications only (DS, AN, UM, errata; if applicable)

Suitability for use in automotive applications only — This NXP Semiconductors product has been developed for use in automotive applications only. The product is not designed, authorized or warranted to be suitable for any other use, including medical, military, aircraft, space or life support equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors accepts no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.